Rule-Based Modelling of Vegetation Dynamics

By

Brian S. McIntosh

PhD

The University of Edinburgh

2002

I declare that this thesis has been composed by myself and that the research reported here has been conducted by myself unless otherwise indicated.

Brian S. McIntosh

Acknowledgements

Professional

I must deeply thank Robert Muetzelfeldt, my supervisor, for introducing me to logic and argument. His uncompromising and simultaneous attention to both the finest detail and the bigger picture interspersed with marvellously irreverent humour helped me immeasurably. I would like to sincerely thank Colin Legg and Stefano Mazzoleni for imparting their invaluable knowledge and experience and for giving me the opportunity to collaborate in their European research work. Thank you to Peter Csontos for the initiating some of this research, for sharing knowledge and for a splendid sense of humour. Although personal contact was infrequent, electronic communication with Margarita Arianoutsou was of great help in knowledge acquisition and model design. For this I thank her. My thanks also go to Nick Winder for providing me with many enlightening discussions about science and philosophy and an ideal intellectual environment in which to finish my work. In addition, I would like to offer my thanks to the Modulus project (EU FP4 grant ENV4-CT97-0685) team – our meetings influenced me enormously. Last of all my wider thanks go to all the members of The Institute of Ecology and Resource Management at The University of Edinburgh for providing a stimulating and friendly atmosphere.

Personal

I would like to thank my wife, Kirstin Davies, along with my family, Sheila, Gordon and Neil for providing the framework that underlies and supports my life. I could do no better. My thanks must also go to my friends in Edinburgh who provided me with many nights of joy and respite from the rigours of a Doctorate. Cheers!

Financial

First, I would like to acknowledge The Carnegie Trust for the Universities of Scotland, by whom I was supported as a Scholar for the first three years of my research. Second, I would like to acknowledge financial support from three EU FP research grants – ModMED II (ENV4-CT95-0139), ModMED III (ENV4-CT97-0680) and Modulus (ENV4-CT97-0685). I would also like to acknowledge the financial support of The Spatial Modelling Centre, Kiruna, Sweden, particularly under the Directorship of Professor Nick Winder, with whom I found gainful employment and time to write this thesis during the year 2001. Last of all, the occasional, irregular, but much needed financial contributions made by my parents must also be thanked.

Abstract

Although there has been over one hundred years of scientific investigation no general, predictive understanding of vegetation structure and dynamics has been developed. A substantial corpus of community- and species-level knowledge has however been accrued detailing the structure and dynamics of specific vegetation systems along with some general principles of vegetation behaviour. The corpus of available vegetation knowledge is characterised by its fragmented form and by the way in which relationships between different ecological quantities tend to be expressed non-quantitatively. Much of the corpus is only held informally and composed of deterministic factual or conditional statements. Despite its form, this thesis demonstrates that available ecological knowledge can be usefully employed for predictive modelling of vegetation dynamics under different conditions. The thesis concentrates on Mediterranean vegetation.

Modelling using available knowledge requires special methods. Suitable techniques should be capable of representing available knowledge with minimal modification and be capable of soundly reasoning with this knowledge to make predictions. Traditional quantitative modelling techniques based on matrix algebra or differential equations cannot represent or reason with non-quantitative knowledge, rendering them unsuitable. Deterministic state transition approaches are capable of representing some community-level knowledge. However it is not clear how to generally reason with such knowledge for prediction and, in addition, current approaches do not handle the dynamics of non-disturbance environmental factors well. Techniques such as FATE that model species dynamics based on functional attributes have been demonstrated to be useful but lack a general formulation, each based on a hard-wired set of attributes. Qualitative Reasoning (QR) techniques like Qualitative Process Theory (QPT) and QSIM contain useful concepts for representing and reasoning with non-quantitative knowledge but utilise branching simulation methods, have difficulties in reasoning about changes in discretely valued variables (the rate of change problem) and have various restrictions which affect their ability to represent ecological knowledge. Qualitative algebras such as SIMAO are not suitable for simulation. Techniques from the field of knowledge-based systems (KBS), particularly the use of first-order logic for representation and reasoning, although not providing an 'off-the-shelf' solution do offer a coherent framework within which to develop suitable techniques.

Using a mixture of concepts and techniques from deterministic state transition and functional attributes modelling, Qualitative Reasoning and knowledge-based systems, three ontologically distinct modelling systems are developed to demonstrate the utility of available knowledge for modelling vegetation dynamics. All three systems use declarative, rule-based approaches based on first-order logic and are composed of a set of representational constructs along with a separate system for reasoning with these constructs to make predictions. A method for reasoning about change in non-quantitative model variables is developed based upon time and direction of change. This 'temporal reasoning system' provides a solution to the rate of change problem and may offer a general way of modelling with non-quantitative knowledge. To illustrate, a different model of Mediterranean vegetation dynamics is developed and run under different conditions for each system. The capabilities and possible problems of each system in terms of ecological validity, knowledge representation and reasoning are discussed. The general utility of rule-based approaches to modelling vegetation dynamics are also discussed along with the implications of the modelling systems developed for the activities of decision-support and ecological theory development.

Table of Contents

Chapter 1 Vegetation Dynamics – Knowledge and Modelling

1.1 Introduction and Aims

1.1.1 Motivation

Vegetation forms the primary production element of most ecosystems critically providing the main food source for entire food chains. Where natural vegetation occurs important reservoirs of biodiversity exist and where natural vegetation exists humans can find valuable resources such as grazing fodder, medicine and wood for fuel and building. Natural vegetation can play a pivotal role in preventing the spread of desertification in arid and semi-arid climates. In addition natural or semi-natural landscapes provide an important focus for tourism-based economic activities and can provide much valued aesthetic qualities to our environment. Given that our understanding is incomplete and the effects of our land-use policies often unclear there is substantial importance in the continued development of methods that can help us better manage natural vegetation resources.

As with other fields of scientific endeavour problems in vegetation dynamics can be divided into those that are concerned with advancing our understanding of the processes and patterns found in the world (pure or 'research' problems) and those that are concerned with better informing our management of resources (applied or 'policy' problems) (Engelen *et al.* 2000). The ability to explore the implications of current understanding to detect inconsistencies, predict behaviours and explore theories and hypotheses is crucial for the continuing process of theory development in vegetation ecology. The ability to explore the effects of management decisions on vegetation structure and dynamics is similarly crucial for purposes of land-use planning and conservation, particularly in degraded or marginal areas, where some courses of action may have debilitating and potentially irreversible consequences. Both are well served by modelling techniques that can represent and reason with the types of incomplete knowledge that are available.

For example, it is now commonly agreed that vegetation change occurs as a result of processes and mechanisms operating at the level of the population or individual (Peet and Christensen 1980, Pickett *et al.* 1987, Smith and Huston 1989, Pickett and Kolasa 1989, Burrows 1990, Luken 1990, Glenn-Lewin *et al.* 1992, Pickett *et al.* 1994). Indeed many of the processes have been identified and explored along with many of the constraining and determining factors. However it is not clear how different processes and factors affect the populations of different species and how the resultant population dynamics combine to produce observed community-level patterns. Characterising different species non-quantitatively by ecological function (Noble and Slatyer 1980, Grime 1985, Noble and Gitay 1996, Gitay and Noble 1997, Lavorel *et al.* 1997, Weiher *et al.* 1999) permits ecologists to ask questions regarding how function at one level of scale relates to pattern at another, higher level i.e. how do the processes and behaviours of different functional types at the species-level lead to the formation and dissolution of different vegetation structures? Under some conditions species interact to produces stable structures and predictable sequences whilst under other conditions the outcomes of interactions are complicated and unpredictable. Exploring relationships between organisational scales will be crucial in the development of a robust understanding of why, how and when vegetation changes. The provision of general modelling tools suited to the types of knowledge and data used in functional attributes research will be useful for doing so.

Regarding vegetation as a resource, land managers require information upon which to base their decisions. It is important that management plans be formulated with as full an understanding of consequence as possible. If the wrong course of action is taken vegetation resources may be degraded or lost with consequent economic and biodiversity impacts. For example how frequently should a particular type of vegetation be burned to maintain its current state and exclude invading species? The complexity of vegetation response and the potential novelty of management action can however mean that it is not at all obvious what will happen for a given action. Given that it is not usual to have a precise quantified understanding of the relationships between vegetation, environment and management options (Luken 1990), and that obtaining such an understanding is likely to be expensive or impossible, possessing methods capable of predictive modelling using available knowledge types will be a useful addition to the decision-support armoury (Roberston *et al.* 1995).

Ecologists often only have an imprecise, partial understanding of the relationships between the different objects and quantities involved in vegetation structure and dynamics – knowledge is incomplete. Traditional vegetation modelling methods use quantitative methods such as differential or difference equations and matrix algebra. As such they require formalised and precise quantitative specification of the relationships between ecological quantities along with appropriate quantitative data for parameterisation and initialisation. Such understanding and data is typically not readily available.

However, this lack of precise quantitative understanding and data does not mean that nothing is known about how vegetation is structured and changes at different scales. Quite the opposite in fact; ecologists have been steadily accruing knowledge regarding vegetation change since the pioneering work of Cowles (Cowles 1899a, Cowles 1899b, Cowles 1899c, Cowles 1899d, Cowles 1910, Cowles 1911) and Clements (Clements 1904, Clements 1916, Clements 1928). Although a coherent body of theory generally capable of explaining and predicting what will happen under given circumstances has not been developed (Pickett and Kolasa 1989, Smith and Huston 1989, Burrows 1990, Luken 1990), substantial understanding exists with many of the underlying processes and mechanisms identified and partially understood (Begon and Mortimer 1986, Pickett *et al.* 1987, Burrows 1990, Luken 1990, Glenn-Lewin and van der Maarel 1992, Peet 1992, van der Valk 1992 and Silvertown and Lovett Doust 1993). The types of partial and imprecise knowledge that have been accrued will be termed available knowledge. It is the contention of this thesis that such knowledge can be formalised and used for the purpose of predicting how vegetation might change over time under different conditions. The problem is how.

1.1.2 Scale and Vegetation

Vegetation structure and dynamics are dependent on the scale of observation (Wiens 1989, Johnson 1996). Scale (of observation) may mean various things:

1. Spatial scale e.g. patch and gap.
2. Temporal scale e.g. decadal, annual, monthly, weekly, diurnal, hourly.
3. Organisational scale e.g. community, species and individual.

Where, when, how and for what duration we observe vegetation will determine what we see. Different patterns will be found at different scales and at different combinations of scales and these patterns may also vary with the species and species mixtures under observation (Wiens 1989, Johnson 1996, Dale 1999). Dynamics at different scales are not independent - feedback from broad scales constrains fine scale operation whilst fine scale processes provide the causal mechanisms that underlie broad scale phenomena (Wiens 1989, Pickett *et al.* 1994).

Throughout this thesis vegetation will be examined using the organisational hierarchy of scale detailed in Table 1. In the hierarchy to be used, each division of scale is composed of many units of the division below e.g. communities are composed of many populations which are composed of many individuals. The focus of this thesis is to develop methods of predicting community-level patterns over time. Scales above the community are outwith the scope of this work. However attention will be paid to finer scales of operation for it is they that control the processes underlying community-level dynamics.

Scale of Observation	Organisational Level
Broad	Global
	Ecosystem / Landscape
⇩	Community
	Species / Population
Fine	Individual Plant

Table 1 Organisational hierarchy for vegetation

1.1.3 A Definition of Vegetation Dynamics

Different types of dynamic are observed in vegetation (see Figure 1 and Table 2) and there are also different ways of classifying these patterns – by process, by cause, by scale etc. (see Mueller-Dombois and Ellenberg 1974, Burrows 1990, Glenn-Lewin and van der Maarel 1992, Peet 1992). Vegetation dynamics will be examined in terms of species compositional and / or structural change. Glenn-Lewin and van der Maarel (1992) provide a classification of temporal pattern in vegetation at different spatial and temporal scales. The focus of this work is to examine dynamics at the community-level – what they term 'vegetation change'.

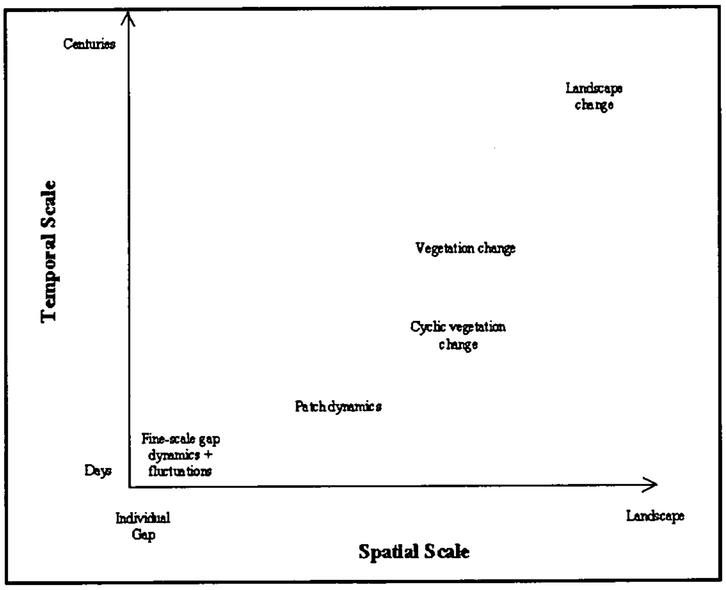


Figure 1 Types of vegetation dynamic by temporal and spatial scale

Each element of a landscape vegetation mosaic can be considered to be a community, a unit of relatively homogeneous vegetation, which may or may not be identical to and may or may not grade continuously with neighbouring landscape elements. This view is similar to Naveh and Liebermann's (1985) conception of the *landscape unit*, a homogeneous stand of vegetation larger than the individual plants it contains. At this scale community-level or vegetation change is observed.

Vegetation change can be roughly defined as the change in vegetation that results from changes in species composition caused by differential growth and establishment resulting from various biotic and abiotic environmental influences. Vegetation change may take various forms and produce different temporal patterns, operating over a time-scale ranging from a few years to several decades and over a spatial scale ranging from one to tens of hectares. (Burrows 1990) identifies different types of vegetation change dynamics that exist and which must be accounted for by any theory of vegetation (see Table 2).

1.	Colonisation and sequential replacements following the formation of new sites or following the disturbance of established vegetation.
2.	Direct replacement following the disturbance of established vegetation.
3.	Cyclic replacements in response to endogenous or exogenous influences.
4.	Fluctuating replacements in response to exogenous influences.
5.	Vegetation maintained by frequent or continual disturbance.
6.	Vegetation in relative equilibrium for a time (composition unchanging for more than one generation of the dominant species).
7.	Long-term, gradual change in response to autogenic or allogenic influences.

Table 2 Types of vegetation change (adapted from Burrows 1990)

Traditionally, vegetation change is classified in two ways – by the starting state from which the change proceeds and by the nature of the processes behind the change (Burrows 1990, Luken 1990, Glenn-Lewin and van der Maarel 1992, Peet 1992). Vegetation change that proceeds from bare ground with no initial propagule bank is termed primary vegetation change. Vegetation change that proceeds from an already vegetated state or from bare ground with a propagule bank is termed secondary vegetation change. Peet (1992) argues that most cases of vegetation change are intermediate between the two extremes of primary and secondary. Another division concerns autogenic and allogenic change. Autogenic change is driven by plant mediated processes whereas allogenic change is driven by processes that are not plant mediated. Most cases result from a mixture of both autogenic and allogenic influences (Glenn-Lewin and van der Maarel 1992).

A third means of classifying vegetation change is into progressive and retrogressive change. This division is generally only used along with notions of succession and climax.

Progressive change is taken to mean a change 'towards' a climax state and generally involves increases in vegetation biomass density, stature and structural complexity. Conversely, retrogressive change is taken to mean vegetation change 'away' from a given climax state, usually through disturbance (grazing, fire etc.) and generally involves a reduction in biomass, stature and structural complexity.

A fourth way to classify vegetation change is into sequential replacement and cyclic replacement types. Species replacement occurs when one or more species replace one or more other species in an area of vegetation over a period of time. If replacement eventually results in species that were previously present in the vegetation re-occurring after a period of absence then the process of vegetation change is said to involve cyclical replacement. If there is no re-occurrence of once present species, then the process of vegetation change is said to involve sequential replacement.

1.1.4 Thesis Aims

The primary goal of this thesis is to demonstrate that it is possible to utilise the types of knowledge currently available to ecologists for predictive modelling of vegetation dynamics at the scale of the community. To demonstrate that this is indeed possible this thesis aims to:

1. Evaluate the capabilities of existing methods for representing and reasoning with available knowledge types with the aim of determining how to model using such knowledge.
2. Develop, implement, test and evaluate techniques for predictive modelling of vegetation dynamics that use available knowledge.

1.2 A Categorisation of Knowledge for Modelling

As a starting point to this study some basic questions must be asked regarding knowledge and modelling. What kinds of available knowledge do we have concerning vegetation dynamics? How can this knowledge be represented and reasoned with to predict how vegetation will change over time under different conditions? What are the implications of representing and reasoning with knowledge in different ways? To answer these questions we must have an understanding of the different types of knowledge that are used in constructing models.

Formal models can be viewed as being composed of four types of knowledge – structural, value, relationship and reasoning. The first three types of knowledge can be treated as being declarative - they represent what is held to be true in terms of facts and conditional statements. The fourth type of knowledge is procedural and involved in using the other types of knowledge to accomplish the task of calculating and updating model variables over time.

Schut and Bredeweg (1996) define a model as being:

> '... a representation of a phenomenon, conceptualised according to a particular goal, for which an inference procedure exists that allows the derivation of new information about that phenomenon.'

Under this definition, structural, value and relationship knowledge are the representational component whilst reasoning knowledge is the inference procedure to be used.

The following sub-sections will describe each of these knowledge types and explain how they relate to modelling vegetation dynamics. During this thesis the underlying view of models taken either explicitly or implicitly is one based on this categorisation. Note that this is not the only possible categorisation possible but it is one that is useful here for the purposes of modelling with non-quantitative variables and relationships.

1.2.1 Structural Knowledge

Models are abstractions of the real world, using variables to represent real and theoretical quantities. The choice of which variables and objects to include in a model creates the model world's structure. This choice is based on structural or ontological knowledge and belief about the world and the relative importance of different quantities for the phenomena and problem being modelled. All models have structure and, for construction, all models require some structural knowledge about the phenomenon being modelled. Levins (1966) termed the degree to which a model's structure corresponds to the (perceived) structure of the real world, the degree of realism of that model.

In addition to domain quantities and objects, models can be structured to represent different components of quantities and objects separately. For example quantities can be considered to possess separate level or amount values and derivative values (direction or rate of change). As will be explored in later chapters structurally splitting components of quantity value can provide a useful way in which to formalise non-quantitative understanding.

1.2.2 Value Knowledge

Real world quantities can be measured and given a value using a variety of scales and units. Model quantities, or variables, are no different. During the process of model development, in addition to constructing model structure, each variable must be given a value. Typically most model variables are valued using the sets of real or integer numbers or sometimes using a binary scale (0 or 1, yes or no) but these are not the only possibilities. Available knowledge about vegetation at either the community or species-levels may be either only roughly quantitative (e.g. we know how much biomass there is for a particular species within a given interval) or not quantitative at all (e.g. a plant species is a facultative resprouter or the vegetation at a site is *Pinus halepensis* woodland).

To represent and reason with different types of value knowledge we will introduce the idea of a support set (Leitch *et al.* 1990) – the set of legal values that a variable can take for a given model. Similar ideas to the support set have been used by other authors for giving values to variables, including for example, quantity spaces (Forbus 1984, Guerrin 1991, Guerrin 1992, Kuipers 1986, Kuipers 1994) and quality spaces (Guerrin 1995). Different variables may have different types of support set, some quantitative, some not. The set of real numbers is probably the most commonly used support set in ecological modelling.

To illustrate the concept of the support set, consider a variable whose value is only known in terms of a range of possible values (an interval) rather than a single value. In such a case we only have approximate or what we will term qualitative knowledge about the variable's value e.g. the growth rate of a plant species is between 0-5 cm/yr. To represent this value we can use the set of reals within a statistical or fuzzy logic framework or alternatively we can define a set containing the model-relevant intervals of values ordered with respect to their numerical magnitude e.g. we could measure a vegetation type's or species' growth rate using a three element support set such as {(0-5), (5-15), (15-20)} where the units are cm/yr and each interval represents a biologically important or problem relevant range of values. In doing so, we are representing the value of the variable using a qualitative support set. One of the contentions of this thesis is that the form of available ecological knowledge is amenable to this type of value representation.

Both single-valued quantitative (real and integer) and qualitative support sets are said to be ordered due to the magnitude relations that hold between their elements e.g. the previous qualitative support set is ordered because (0-5) $\leq$ (5-10) $\leq$ (10-15). Support sets in which the elements neither represent a numerical value nor are ordered with respect to each other are examples of unordered linguistic support sets e.g. the support set {obligate seeder, facultative seeder, resprouter} uses value elements with no inherent numerical interpretation and which are ordered in an arbitrary way. It is possible to order linguistic support sets with respect to some other quantity using order of magnitude operators such as >, < etc. This creates an ordered linguistic support set. For example {grassland, shrubland, forest} is an ordered linguistic support set taking vegetation types ordered according to maximum height as value elements.

We will return to support sets in Chapter 3, but for now it is sufficient to note that there are different types, differentiated by the nature of the underlying value-scale (single-valued quantitative, qualitative or linguistic) and whether magnitude or other relations hold between the elements of the set or not (ordered or unordered).

The type of support set best used for any given model depends on a combination of the dynamics of the phenomena being modelled, the purpose of the model (the question being asked) and the knowledge available. There is no need to automatically choose the highest precision support set for model variables (Leitch *et al.* 1990).

1.2.3 Relationships

Ecological model variables are usually connected by directed influences into networks. Model variables influence each other in often complex chains of causality with feedback loops that operate over time. Influences between variables in ecological models tend have directed causality. That is if there is a relationship between two variables then that relationship is usually an influence operating from one variable to another - from an independent variable to a dependent variable.

The exact form of an influence between variables is termed a functional relationship. Functional relationships between variables valued using quantitative support sets are usually expressed as algebraic equations which describe either how the independent variable causes change in the dependent variable or how the independent variable constrains change in the dependent variable. Functional relationships between non-quantitative support set variables are difficult or impossible to represent using traditional algebraic structures. For example if an independent variable, X, is valued using the qualitative support set {low, medium, high} and a dependent variable, Y, is valued using the qualitative support set {low, high}, how can the functional relationship between them be represented? Standard algebraic operators cannot be used as they rely on the properties of the real or integer support sets. The properties of the two qualitative support sets are not clear so alternative representational means may be necessary. One of the aims of this thesis is to identify and utilise ways of representing functional relationships between variables of different support sets such that they represent available ecological knowledge concerning vegetation dynamics.

As well as functions (algebraic or otherwise) to determine the values of variables, relationship knowledge can take the form of order of magnitude statements using operators like >, ≤ or =. Such relationships compare the value of a variable of interest with the values of one or more influencing variables and in doing so represent constraints on value.

It should be noted that representing relationships between variable values in a model is treated separately here from the process of calculating values. Relationship knowledge is treated here as being declarative in form – it states what is known to be true about how the values of different variables influence and constrain each other.

1.2.4 Reasoning

Once the variables in a model have been selected based upon structural knowledge of the domain being modelled, once the support sets have been selected for each variable and the functional relationships established between variables, a means of reasoning must be employed to utilise the knowledge contained within the model to calculate and update variable values during simulation. The procedural knowledge concerning how to calculate and update variables will simply be termed reasoning. Variables valued using numerical support sets such as the real numbers can be calculated or updated using mathematical equations and standard numerical calculation algorithms. However, such reasoning is either difficult or impossible to apply to qualitative or linguistically valued variables. For example, given a species with low biomass and a high growth rate at time t_1, how can biomass be updated to provide a value for time t_2? A standard algebraic equation and numerical algorithm cannot be used as the variables are not valued quantitatively, their underlying support sets not necessarily having the same properties or being amenable to arithmetic operators in the same way as real valued variables. Methods for reasoning about change in variable values appropriate to the type of support set must be employed. It is one of the major aims of this thesis to develop and utilise different types of reasoning suitable for use with available ecological knowledge. Chapter 3 will examine and evaluate various existing ways of reasoning with non- quantitative support sets whilst Chapter 4 will describe the approach to be used in the modelling frameworks detailed within this thesis.

1.3 Implications of Current Understanding for Modelling

1.3.1 The Format and Structure of Available Knowledge

Available knowledge about vegetation dynamics tends to be held informally, either in verbal or textual descriptions in published or unpublished form possibly derived from the qualitative interpretation of quantitative results (Guerrin 1991), or in the accumulated knowledge and beliefs possessed by domain experts; ecologists. By informal here we mean that the knowledge is not expressed in the syntax of some formal system such as mathematics or first order logic.

Available knowledge is often situation or condition specific and may be more phenomenological than mechanistic (*sensu* Pickett *et al.* 1994). The behaviour of different objects and quantities is often only understood in terms of roughly what happens under particular sets of conditions. A precise and general understanding of how different objects and quantities relate to each other is usually not available. This means that available structure and relationship knowledge tends to take the form of facts and rules expressed using approximate terminology and coarse-grained values rather than with precise definitions and quantitative measurements. Values and relationships between quantities may only be expressed in qualitative and linguistic terms and the simultaneous use of different support sets can be viewed as making available vegetation knowledge 'value heterogeneous'.

For example many species attributes can be viewed using linguistic support sets such as {sprouter, seeder} whilst community and species-level quantities can be viewed as being valued using approximate, qualitative support sets like {low, medium, high} to reflect imprecision in understanding. Relations may be expressed in terms of direction of change under different conditions or in terms of known correspondences between quantity values. For example:

> 'Under repeated burning at high altitude vegetation in the *Cedrus atlantica* sequence will tend to change towards spiny therophyte cover (Le Houérou 1981).'
>
> 'Under grazing, species of low growth form will tend to increase (Burrows 1990).'

'Low light levels correspond to low germination rates for *Pinus halepensis* (Arianoutsou and Ne'eman 2000).'

The rules, facts and descriptions that constitute available vegetation knowledge can be viewed as a set of knowledge fragments. Some knowledge fragments are directly related, referencing each other and creating explicit dependencies whilst others do not. This means that as a whole, the body of available ecological knowledge concerning vegetation lacks coherence. It is not in the form of a well-structured and tested theory with derivable consequences and implications. However, despite not forming a uniformly coherent body of understanding, it is the contention of this thesis that ecological knowledge fragments are nonetheless valuable sources of understanding that can be utilised for model-based prediction.

Over the course of the subsequent chapters it will be shown that available knowledge fragments can be co-ordinated to make predictions about the outcome of given situations over time. To do so will require the development of forms of reasoning suitable for the different types of available knowledge. For example, it is not immediately obvious how to coherently update model variable values based upon knowledge fragments that describe relationships between quantity values in terms of non-quantitative support sets.

It should be noted that none of these points imply that the many investigations undertaken in vegetation science since the days of Clements have yielded few formal, quantitative results. On the contrary, formal results, including many mathematical relationships between ecological quantities, have been uncovered and presented in the literature. However these tend to be site and study specific, requiring significant abstraction into informal, qualitative, linguistic and approximate terms before they can be interpreted and used practically (Guerrin 1991). Generalisations, rules and principles in ecology do exist, but appear not to have the same precision as their counterparts in physics and chemistry (Loehle 1987).

1.3.2 Modelling Vegetation using Available Knowledge

Using available knowledge presents special problems and places particular requirements on the techniques that can be used for modelling vegetation dynamics. Issues concerning the representation of non-quantitative support sets, heterogeneous variable values and relationships and how to calculate and update variables valued in such ways need to be addressed before available knowledge can be successfully deployed for predictive modelling.

The issues concerned can be stated as a minimal set of functionality requirements for modelling vegetation dynamics using available knowledge:

Structure:

- Representation of vegetation using vegetation types, community attributes and properties or species-level attributes and properties.
- Representation of the processes, mechanisms and causes that influence vegetation including biological, environmental and disturbance related causes.
- Representation of vegetation-environment feedbacks when required.

Value:

- Representation of heterogeneous variable values using support sets appropriate to the type of available knowledge. This may involve the use of support sets other than the reals or integers.

Relationships:

- Representation of quantitative, non-quantitative and mixed support set functional relationships between variables.

Reasoning:

- Reason coherently with available knowledge fragments so as to achieve accurate and sound calculation of model variables, regardless of support set type.

1.4 Research Context and Collaboration

The research carried out for this thesis was partly carried out under the auspices of three separate European Union Environment Framework projects – ModMED II (ENV4-CT95-0139), ModMED III (ENV4-CT97-0680) and Modulus (ENV4-CT97-0685). The first two projects were concerned with modelling Mediterranean ecosystem dynamics at the landscape-level using a range of models and modelling techniques from the individual-level to the community-level. The third project was concerned with applying models produced by other EU-funded research projects to the task of providing practical integrated environmental decision support to regional governments in two Mediterranean countries. Consequently, although the techniques to be developed during this thesis have wider application, they were developed within a Mediterranean context and will be illustrated using models of Mediterranean vegetation dynamics.

The models produced were designed in collaboration with various ecologists from the ModMED and Modulus projects using knowledge acquisition (KA) methods. KA sessions were carried out using a more-or-less unstructured interview format, a recognised approach to acquiring knowledge for modelling (Gordon 1989, Windon Jr. and Massey 1991). No formal protocols were employed although influence diagrams, tables and written if...then rules were used as appropriate. The sessions were carried out both as general domain knowledge acquisition behind the development of techniques to model with available knowledge and within the contexts of the three projects mentioned above as part of the development process behind specific models of Mediterranean vegetation dynamics.

Details of the ecologists who took part in knowledge acquisition exercises:

1. Dr. Colin J. Legg
   Institute of Ecology and Resource Management, The University of Edinburgh, UK.
2. Prof. Stefano Mazzoleni
   Istituto di Botanica, Facoltà di Agraria, Università di Napoli 'Federico II', Italy.
3. Prof. Margarita Arianoutsou
   Dept. of Ecology and Systematics, National and Kapodistrian University of Athens, Greece.
4. Dr. Peter Csontos
   Dept. of Plant Taxonomy and Ecology, Lorand-Eötvös University, Hungary.

Individuals will be specifically credited during the course of this thesis where appropriate.

1.5 Thesis Outline

Chapter 2 will evaluate the utility of techniques that are capable of representing and reasoning with available knowledge for predictive modelling of vegetation dynamics. It will argue that current techniques, although offering partial solutions to the problem of modelling with available knowledge, do not provide a suitable 'off the shelf' solution. Certain concepts and notions have been shown to be useful but a new approach is required to utilise them effectively and to solve certain existing problems. The outline of the new approach to be developed in Chapter 4, 5 and 6 will be described in Chapter 3. Chapters 4, 5 and 6 will detail three rule-based modelling systems developed to address this need, describing their representational and reasoning methods and illustrating their capabilities with various models of Mediterranean vegetation dynamics. Chapter 7 will discuss the findings of the thesis and outline future directions for this type of work.

Chapter 2 Modelling with Non-Quantitative Knowledge – A Methodological Evaluation

2.1 Introduction

2.1.1 Vegetation Modelling

Typical vegetation models tend to use process-based or probabilistic state transition formalisms. The process approach (see for example Cropper and Carter Ewel 1987, Bossel and Schäfer 1989, Aber and Federer 1992, Mauchamp *et al.* 1994, Pakeman *et al.* 1995, Ryan *et al.* 1996) concentrates on quantitatively modelling ecological systems in terms of the storage and flow of matter (carbon, water etc.) or individuals in such a way as to represent ecosystem or species-level processes such as growth, senescence etc. The typical process model is based on the use of differential or difference equations.

The probabilistic state transition or Markov Chain approach involves representing vegetation in terms of discrete states with varying probabilities of transitioning between one state and another. The essence of the method is to represent an area of land by means of a state vector, each element of which represents the relative cover occupied by a particular community state or species. The state vector is then multiplied by an empirically estimated matrix of transition probabilities between states (e.g. the probability of state A changing to state B) to obtain the next time-step's state vector. Despite various criticisms (see Facelli and Pickett 1990) Markov models have received widespread and continued attention from vegetation ecologists (for example see Usher 1979, Legg 1980, Usher 1981, Isagi and Nakagoshi 1990, Usher 1992, Legg 1995, Logofet and Lesnaya 2000).

Both approaches use traditional algebraic, differential, difference or matrix equations and require detailed data for parameterisation and initialisation. This makes them incapable of either representing or reasoning with non-quantitative value and relationship knowledge.

2.1.2 Aims of the Evaluation

Representing and reasoning with these types of knowledge is not a new activity, even within the domain of ecology. The artificial intelligence field of Qualitative Reasoning (QR) is concerned with these problems but almost entirely within physical and often engineering domains. Knowledge-based (KB) system research, another field of endeavour within artificial intelligence, has had an impact on modelling through the development and application of various techniques to tackle problems of representing and reasoning with expert domain knowledge. Within ecology some attempts have been made to develop and use models and methods for modelling that utilise non-quantitative domain knowledge such as May's qualitative modelling of ecosystem stability (May 1973), Noble and Slatyer's Vital Attribute approach to predicting vegetation dynamics (Noble and Slatyer 1980), work on qualitative modelling of hydroecological systems carried out by Guerrin (1991), Guerrin (1992) and Heller and Struss (1996), the work of Guerrin *et al.* (1997) on qualitative modelling of salmon population dynamics and the QR-based models of Brazilian Cerrado vegetation presented by Salles (1997) and Salles and Bredeweg (1997).

As a result various methods already exist that are capable of representing and reasoning with non-quantitative knowledge. The aim of this chapter is to assess the suitability of a range of candidate techniques for modelling vegetation dynamics based on available knowledge. More specifically, this chapter aims:

1. To determine the capabilities of existing techniques able to represent and reason with non-quantitative value and relationship knowledge.
2. To evaluate the suitability of these techniques for modelling vegetation dynamics using available vegetation knowledge.

In Chapter 3 the approach to be taken and techniques to be used for modelling vegetation dynamics during the remainder of this thesis will be detailed.

2.1.3 Evaluation Method

2.1.3.1 Procedure

The assessment of each candidate will take place in two parts. First, the literature on each candidate technique will be reviewed. Second, the ability of each technique to represent and reason with available ecological knowledge will be assessed based upon the results of the review plus an evaluation of their functional capabilities. To carry out the functionality evaluation the capabilities of each technique will be rated using the criteria shown in Table 3.

Functionality	Rating
Structural Representation	
Community (vegetation) types	
Community attributes	
Community properties	
Species attributes	
Species properties	
Inter-specific interactions	
Intra-specific interactions	
Non-disturbance environmental influences	
Disturbance influences	
Vegetation-environment feedback and dynamics	
Value Representation	
Support sets:	
Real values	
Integer values	
Qualitative values	
Ordered linguistic values	
Unordered linguistic values	
Direction of change	
Relationship Representation	
Quantitative relationships	
Non-quantitative relationships	
Mixed support set relationships	
Reasoning	
Reasoning with quantitative relationships	
Reasoning with non-quantitative relationships	
Reasoning with mixed relationships	
Time-driven simulation	
Deterministic reasoning with state variables	
Non-deterministic reasoning with state variables	

Table 3 Functionality Evaluation Criteria

2.1.3.2 Functionality Evaluation

The functionality criteria to be used were chosen for their relevance to the knowledge and types of knowledge that are available regarding vegetation structure and dynamics, in particular the set of minimum functionality requirements (section 1.4.3). They are grouped into categories following the taxonomy of knowledge for modelling detailed in Chapter 1.

Structural Representation criteria:

These criteria detail each candidate technique's ability to handle different structural aspects of available ecological knowledge i.e. what ecological quantities and types of quantities can be represented. Sufficient knowledge is available to structure models at two organisational scales – community and species. Consequently aspects of both these scales are included as representational criteria. Attributes are defined as approximately time invariant quantities that characterise community types or species e.g. palatability. Properties are defined as potentially dynamic quantities that are typically used to represent the state of vegetation e.g. biomass, growth rate etc. The inter-specific and intra-specific interaction criteria refer to the ability of a candidate technique to represent influences between quantities representing different or the same species. Non-disturbance environmental and disturbance influence criteria detail whether a technique can represent the effect of the environment on vegetation.

The 'vegetation-environment feedback and dynamics' criteria details whether a technique can represent dynamic influences from vegetation to environment as well as from environment to vegetation.

Value Representation criteria:

The types of support set that a technique can use to represent variable values. Available vegetation knowledge shows that knowledge about ecological quantities is value heterogeneous i.e. quantities can be given values from a variety of support sets. Representation of different value types is therefore an important capability. In addition this category details whether techniques can represent direction of change values i.e. increasing, decreasing, steady, the value towards which a variable value is heading or numerical derivatives. There is non-quantitative value and relationship knowledge available concerning the direction in which various ecological quantities change under different conditions, potentially making this a useful representational capability.

Relationship Representation criteria:

The types of quantitative, non-quantitative (qualitative *or* linguistic) and mixed value relationships that can be represented by a technique, where 'mixed value' refers to the use of some mixture of quantitative, qualitative and linguistic variables in a single relationship statement. Some ecological knowledge is available regarding quantitative relationships between quantities but more is available in the form of non-quantitative or mixed relationships. No further distinction is made here regarding types of relationships since it is not clear what form non-quantitative and mixed relationships may take. Part of the function of this chapter is to review and assess methods for doing just this.

Reasoning criteria:

As mentioned in Chapter 1, representing relationships between variables and reasoning with the knowledge they express are considered separately in this thesis. A technique may provide good representational capabilities but not be able to reason with that knowledge to calculate variable values hence the separate treatment here. The criteria:

- *Reasoning with quantitative relationships:*

  Ability to calculate / infer quantitative variable values or changes in value from quantitative relationship statements.
- *Reasoning with non-quantitative relationships:*

Ability to calculate / infer non-quantitative variable values or changes in value from non-quantitative relationship statements.

- *Reasoning with mixed relationships:*

  Ability to calculate / infer quantitative *or* non-quantitative values or changes in value from mixed support set relationship statements.
- *Time-driven simulation:*

  Does the technique provide a simulation-clock driven means of reasoning about changes in variable values over time? Time-driven simulation, where the values of state variables are passed from one time-step to another, is an appropriate reasoning approach if the type of problems being addressed by a model involve answering questions relating to the values of quantities over specific periods of time or at specific points in time e.g. how will tree biomass change over a period of 10 years immediately post-fire or what will the average height of a patch of vegetation be 3 years after grazing stops? Being typical in vegetation ecology, this thesis aims to address problems of this type.
- *Deterministic reasoning with state variables:*

  When state variables are updated / their new value inferred, does the technique enforce determinism in the solution? That is, does it only permit one possible solution for each update process? Enforcing single solutions may require more structural and relationship knowledge than is available but allows more definite predictions to be made when running a model. Note the subtle difference between determinism in producing one solution and determinism with regards non-random. Under the current definition a stochastic model, by producing a single solution per calculation iteration, would also be deterministic.
- *Non-deterministic reasoning with state variables:*

  When state variables are updated / their new value inferred, does the technique permit multiple solutions to be generated if there is insufficient relationship knowledge to constrain the variable update process? Such reasoning may be appropriate if there is insufficient or indeterminate domain knowledge available (Robertson *et al.* 1995). Non-deterministic reasoning here does not necessarily imply any form of stochasticity.

For each criteria the capability of each technique will be rated using a 4 point scale similar to that employed by Page (1994). The scale should be interpreted as:

1. Functionality not recognised (not mentioned in any sources).
2. Functionality recognised (mentioned in at least 1 source and not demonstrated at all or demonstrated incorrectly) or, possible but not recognised (theoretically possible).

3. Functionality demonstrated (correctly demonstrated in at least 1 source).
4. Functionality conclusively demonstrated (with *substantial* or *incontrovertible* evidence).

In addition to the criteria-based functionality evaluation, some candidate techniques will be evaluated in terms of their ability to represent and reason with a standard ecological modelling problem. This additional evaluation step will be performed on the Qualitative Reasoning techniques, which have so far received only limited ecological application. As such they possess a set of capabilities for representing and reasoning with non-quantitative value and relationship knowledge but it is not clear how appropriate they are for handling ecological knowledge. Other techniques have received sufficient ecological application to make this extra step unnecessary.

Table 4 lists the techniques to be evaluated.

Category	**Technique**
Ecological modelling techniques	Deterministic state-transition modelling
	Functional attributes modelling
Knowledge-Based Systems & Knowledge-Based Dynamic Modelling	Basic principles of KBS, various applications of KBS to vegetation ecology, various rule-based and logic-based ecological models
Qualitative reasoning	QSIM
	Qualitative Process Theory
	SIMAO symbolic algebra system

Table 4 Candidate techniques for the evaluation

2.2 Ecological Modelling Techniques

2.2.1 Deterministic State Transition Modelling

2.2.1.1 Review

Vegetation can be pictured as occupying one of a set of discrete states (vegetation types), vegetation dynamics then viewed as being transitions between these states. This is the conceptual model underlying early successional, climax theories of vegetation dynamics and more recent community-level ideas such as the Kinetic concept (White 1979). Despite being shown not to universally apply, this conceptual model has been and continues to be used as the basis for predicting vegetation dynamics under some conditions. Recently Hobbs (1997) noted the potential usefulness of the approach to examining vegetation dynamics in response to environmental change.

The deterministic state transition approach represents vegetation using a single state variable, vegetation type, which can be influenced by other variables representing environmental and disturbance influences and integer time counters representing stand age. Vegetation state here can be viewed as a linguistic variable, each value element corresponding to a vegetation type. Relationships between vegetation state value and other variables can be viewed as sets of 'if ... then' rules representing how changes in state are related to influencing variable values. For example, with grazing valued using presence / absence:

> **IF** the vegetation of a stand is in 'shrubland' state **AND** there is grazing on the stand **THEN** the stand will change state to 'degraded shrubland'.

Reasoning about change in state value can be achieved by applying state transition rules once per time-step to each stand of vegetation being modelled. In addition to instant change rules such as the one above it is also possible to reason about developmental (non-disturbance) induced change by keeping a track of concepts such as time in state (length of time a stand has been at a particular state value) then comparing current time in state with required stand age values for particular transitions to occur. This requires reasoning with mixed support set relationships such as:

> **IF** the vegetation of a stand is in 'shrubland' state **AND** the stand age $\geq$ 30 years **THEN** the stand state will change to 'open forest'.

The Columbia River Basin Succession Model (CRBSUM) (Keane *et al.* 1996) uses deterministic state transitions organised into environmental condition specific series (successional pathways). In CRBSUM vegetation states are called 'succession classes' and represent a combination of cover type and structural stage. Vegetation series are formed from many such classes interlinked by possible transitions into a multiple pathway network. Each successional pathway represents the possible ways in which vegetation can change under a given set of conditions and terminates in a single stable end-state, the Potential Vegetation Type (PVT). Different stands of vegetation across a landscape may therefore follow different successional pathways depending on the biophysical conditions present.

Vegetation stands are modelled as being in a particular successional class and having a particular succession age - a time counter representing the relative developmental age of the class with respect to the PVT of the pathway being followed. Time drives CRBSUM simulation. Vegetation dynamics are modelled as changes in a stand's successional class within a successional pathway caused either by disturbance, management action or by the processes of vegetation development. If there is no disturbance or management succession age increments by 1 per year and transitions occur as and when appropriate depending on the age and pathway concerned. If there is a disturbance event or management action then either the succession age will be changed in a disturbance and class specific way but class will remain constant or class will change and succession age will be set equal to a new value specific to the new class. CRBSUM can represent and reason about the effects of different disturbance and management action severities.

The TELSA System for exploring vegetation dynamics at the landscape scale (Kurz *et al.* 2000) also uses successional classes representing a combination of species composition and structural stage to model vegetation at the community level. Classes are then connected into successional pathways with each transition associated with a particular required transition time. Vegetation dynamics are then modelled in the same way as CRBSUM with the occurrence of disturbance modelled probabilistically for each class. The pathways and classes behind both CRBSUM and TELSA are typically derived from workshop-based knowledge elicitation exercises involving domain experts (Kurz *et al.* 2000).

2.2.1.2 Functionality Evaluation

The functionality checklist for the deterministic state transition approach is detailed in Table 5. Although the literature is not extensive, the approach has been demonstrated as possessing a suitable ontology and the representational capabilities to capture certain aspects of available structural knowledge (vegetation type and causal influences on vegetation type), value knowledge (vegetation states – an unordered linguistic support set) and relationship knowledge (how vegetation state changes under different environmental and disturbance conditions and how long it takes to do so). The approach provides a means of reasoning about change in vegetation type over time depending on environmental and disturbance conditions. This is achieved in CRBSUM and TELSA through reasoning with disturbance and management action types and severities, integer valued succession age and succession age required to change variables.

The ontology of vegetation states and state transitions is suited to modelling Mediterranean vegetation under some conditions. Such vegetation can sometimes be composed of well-characterised, identifiable states (Nahal 1981, Tomaselli 1981a, Tomaselli 1981b) and, due to a relatively high degree of resilience (Keeley 1986), follow well-known, predictable transition pathways under different sets of conditions. However, the deterministic state transition approach will not always be suitable, where continuous change, difficulties in type classification and a lack of resilience in structure mean that vegetation dynamics cannot easily be represented or modelled in terms of discrete states and their transitions.

Functionality	Rating
Structural Representation	
Community (vegetation) types	3
Community attributes	1
Community properties	1
Species attributes	1
Species properties	1
Inter-specific interactions	1
Intra-specific interactions	1
Non-disturbance environmental influences	2
Disturbance influences	3
Vegetation-environment feedback and dynamics	1
Value Representation	
Support sets:	
Real values	1
Integer values	3
Qualitative values	1
Ordered linguistic values	1
Unordered linguistic values	3
Direction of change	1
Relationship Representation	
Quantitative relationships	1
Non-quantitative relationships	1
Mixed support set relationships	3
Reasoning	
Reasoning with quantitative relationships	1
Reasoning with non-quantitative relationships	1
Reasoning with mixed relationships	3
Time-driven simulation	3
Deterministic reasoning with state variables	3
Non-deterministic reasoning with state variables	1

Table 5 Deterministic State Transition Modelling Functionality

Where the approach is ecologically applicable it is not however clear that current implementations offer sufficient flexibility in their representation and reasoning capabilities to adequately capture available knowledge or accurately model vegetation dynamics. For example, in CRBSUM and TELSA each vegetation series represents all the values of vegetation state that can occur under one set of biophysical conditions. Different vegetation series are needed to model vegetation under different sets of conditions. No means of representing and reasoning with knowledge regarding how environmental conditions may change at a site during a run is provided. This means that CRBSUM and TELSA cannot model a site's vegetation changing from one developmental series to another due to environmental change nor can they model effects of environmental change such that a site's vegetation series and next state remain the same (constant direction of change) but the time (successional age) required for the site's vegetation to change state increases or decreases (varying rate of change). There is no separation in reasoning about direction and rate of vegetation change in the two systems.

2.2.2 Functional Attributes Modelling

2.2.2.1 Review - Vital Attributes

One of the most influential approaches to functional attributes modelling of vegetation has been the Vital Attributes (VA) scheme (Noble and Slatyer 1977, Noble and Slatyer 1978, Noble and Slatyer 1980, Noble 1985, Noble and Gitay 1996). The VA scheme is based upon 4 axioms describing how vegetation is thought to operate (Noble and Gitay 1996). From these axioms two sets of functional attributes were deduced with the aim of predicting vegetation change in response to disturbance – attributes affecting persistence and arrival at a site (10 attributes) and attributes affecting establishment and growth at a site (3 attributes). Species are then described in terms of one persistence/arrival attribute and 1 establishment/growth attribute e.g. DI species have the persistence/arrival attribute D meaning that they always have propagules available at a site and they have the establishment / growth attribute I, which means that they are only able to establish during the short period of reduced competition immediately after a disturbance. Taking the two attribute sets together gives rise to 13 biologically feasible combinations that are likely to be common. Each of the 13 combinations defines a VA plant functional type. Each functional type is also described by the time taken to reach certain critical life history stages - time to reproductive maturity *(m)*, life-span *(l)* and time taken for all propagules to be lost from a site *(e)*.

On a given patch, species are represented as belonging to one of the VA functional types and also as being in one of four life stages (presence/absence only) – juveniles, mature, propagules (only) and locally extinct. There can be as many species types present as desired on any patch. Each functional type transitions between its life-stages under disturbed and undisturbed conditions according to a different set of rules (see Noble and Slatyer 1980 for details). For example take an instance of species type CT that takes 10 years to reach maturity (i.e. m = 10). The CT species type is prone to local extinction under frequent disturbance due to its inability to reproduce before reaching the mature state. The CT life-stage transition rules can be represented thus:

> **IF** the species type is in the juvenile state **AND** there is a disturbance **THEN** the species type will transition to locally extinct.

IF the species type is in the juvenile state **AND** the time in that state ≥ m (where m = 10) **AND** there is no disturbance **THEN** the species type will transition to mature.

IF the species type is in the mature state **AND** there is a disturbance **THEN** the species type will transition to juvenile.

IF the species type is in the mature state **AND** there is no disturbance **THEN** the species type will remain in the mature state.

The rules mean that if this species type is at a site in juvenile life-stage and a disturbance occurs then it will transition to a locally extinct life-stage due to it only being able to persist if mature plants are present. If the disturbance occurs after 10 years then it will, at year 10, transition to a mature life-stage before transitioning back to a juvenile life-stage when the disturbance occurs.

Different combinations of species types with different critical life-stage times will produce different patterns of vegetation dynamic (replacement sequences) based upon their life-stage transition rules and the disturbance regime they are subjected to. If all species at a site are taken as being in juvenile stage it is possible to derive replacement sequences for a given set of species under different disturbance conditions based upon a complete exploration of the possible behaviours arising from their type, their life stages, the required times to change life-stage (m, l and e) and different disturbance regimes (see Noble and Slatyer 1980 for details). Defining a qualitative state as a unique combination of life-stage presence/absence information for a given set of species types, a replacement sequence completely enumerates all possible qualitative states and how they are linked i.e. what state transitions are possible.

2.2.2.2 Review – FATE

The FATE (Functional Attributes in Terrestrial Ecosystems) model of Moore and Noble (1990) involves representing vegetation as occupying a patch of land in the same way as VA. Patches contain life-stage specific cohorts of functional types – propagules, germinants, immature plants and mature plants. The size (abundance) of each life-stage cohort (numbers of plants) is represented internally by a real value but presented as model output using a qualitative {low, medium, high} support set. The dynamics of functional type life-stages are then modelled with respect to three sets of processes:

1. Life history (processes - dispersal, propagule storage, germination, establishment, growth, maturation and senescence).
2. Environmental / competition response (processes – tolerance of conditions and niche relationships).
3. Disturbance response (process – escape, death and resprouting).

The life-stages and processes used represent an extension of the VA approach. For each process each functional type will have one or more 'parameter' values that may or may not be life-stage specific. Values may be of different types including binary yes/no values and qualitative values chosen from one of a few different support sets such as {low, medium, high}. Where qualitative scales are used the qualitative values are mapped onto single numeric values for internal computation. Functional type parameter values determine how they behave and how vegetation changes over time on a patch.

FATE uses 3 sub-models to model the dynamics of the functional types – life-history, competition and disturbance response. For each functional type the life-history sub-model uses various rules and equations to determine the presence or absence of a propagule pool, the rate at which the propagule pool decreases (propagule mortality), the actual germination rate based upon each species' maximum germination rate plus its propagule pool size plus the ageing of plants from immature to mature and the death of mature plants, both depending upon age. It is not clear from Moore and Noble (1990) how the actual size of life-stage cohorts is determined by FATE.

The competition sub-model is involved in modelling inter-specific interactions with regard to access and utilisation of a generalised resource. Each functional type life-stage corresponds to a competitive stratum, used to represent competitive relationships between life-stages of the same and different species types. Strata are represented using integers, with there typically being 5 or less in any model. As cohorts of each functional type move from one life-stage to another they also move up a strata level. The stratum level to which each functional type's life-stages correspond depends on the size and growth form of the species type in reality. No guidance on how to classify species types into competitive strata is given. Higher-level strata have preferential access to the generalised resource with actual access for any given life-stage cohort determined by the total size of all cohorts in higher strata. Cohorts of equivalent strata are assumed not to compete.

Resource availability is outputted using the support set {low, medium, high} but, like life-stage cohort size, computed using real numbers that are subsequently mapped onto the output scale. Although the actual calculation procedure for resource utilisation is not clear from Moore and Noble (1990) the effects on cohorts are. Available resources are calculated for each strata using some function. Each life-stage cohort from each functional type is then either killed or survives the resource availability level based on the application of survival rules.

FATE models the effects of event-based disturbances (fire etc.) on functional types using its disturbance response sub-model. For each life-stage cohort of each functional type, rules are used to determine whether none, few, half, most or all plants are unaffected by, are killed by and resprout after a disturbance event. If a life-stage cohort resprouts its age is reset to some life-stage and functional type specific value to represent the loss in tissue incurred during the disturbance.

Both Moore and Noble (1993) and Pausas (1999) report using FATE successfully.

2.2.2.3 Review - Other Functional Attribute Techniques

Armstrong and Milne (1995) describe how functional attributes could be used to model the response of vegetation to grazing. Although they do not develop a full model they characterise the functional attributes of *Nardus stricta* and make some predictions about the species under different conditions. Ellison and Bedford (1995) employ functional attributes within a cellular automata framework to model the dynamics of a wetland. They identified 4 functional types and used 10 life-history characteristics. CA rules based upon these characteristics of each type were used in combination with various probabilistic equations to determine lateral growth between cells and death within a cell. Franklin *et al.* (2001) detail LANDIS, a spatially explicit landscape model based upon the dynamics of functional types in response to disturbance. The model characterises the landscape as a grid, each cell containing presence / absence information for age-specific cohorts from a set of functional types based on the work of Noble and Gitay (1996). A mixture of probabilistic and deterministic rules was then employed to determine the establishment, persistence, mortality and response to disturbance of functional types.

2.2.2.4 Functionality Evaluation

Table 6 details the functionality checklist for the functional attributes modelling approach. The approach, which is really a group of related techniques rather than a single coherent method, has been shown to be useful for modelling vegetation dynamics using both non-quantitative and quantitative value and relationship knowledge. The approach is capable of making coarse-grained predictions about the direction and form of vegetation change based on knowledge about how species types behave in relation to their environment, mainly disturbance factors.

Functionality	Rating
Structural Representation	
Community (vegetation) types	2
Community attributes	1
Community properties	1
Species attributes	4
Species properties	3
Inter-specific interactions	1
Intra-specific interactions	3
Non-disturbance environmental influences	3
Disturbance influences	4
Vegetation-environment feedback and dynamics	1
Value Representation	
Support sets:	
Real values	3
Integer values	3
Qualitative values	3
Ordered linguistic values	1
Unordered linguistic values	4
Direction of change	1
Relationship Representation	
Quantitative relationships	3
Non-quantitative relationships	3
Mixed support set relationships	3
Reasoning	
Reasoning with quantitative relationships	3
Reasoning with non-quantitative relationships	3
Reasoning with mixed relationships	3
Time-driven simulation	3
Deterministic reasoning with state variables	3
Non-deterministic reasoning with state variables	1

Table 6 Functional Attributes Modelling Functionality

The techniques that go to make up the approach are capable of representing certain aspects of available structural, value and relationship knowledge. Relationships between non-quantitative variables and between variables of mixed support set are essentially expressed using 'if ... then' rules. Relationship rules are then reasoned with in conjunction with time counters and disturbance event occurrences to determine replacement sequences that describe all possible states and trajectories for a given stand of vegetation or to simulate the dynamics of a given stand under given conditions over time.

However current techniques contain various restrictions and problems that adversely affect their ability to model using available knowledge. First, all current functional attributes techniques lack generality in their representation and reasoning capabilities. The attributes that are used are hard-wired in terms of their names (structure), possible values and relationships with the implication that only knowledge concerning those attributes can be represented and reasoned with. It is not immediately clear how to represent and reason with species-level knowledge more generally based upon the methods of functional attribute modelling.

Second, although genuinely capable of representing and reasoning with linguistic and mixed relationships and genuinely capable of representing qualitative values, current functional attributes techniques have no adequately demonstrated capability in reasoning with qualitative relationships or mixed relationships involving qualitative values. For example, the VA approach only uses presence / absence to value life-stage abundances whilst FATE, although outputting abundance values using qualitative values, actually uses quantitative (real) values and relationships to reason about change 'behind the scenes'. Crucially, it is not clear from Moore and Noble (1990) how many of the key calculations of FATE are actually carried out. It may be necessary to represent and reason with greater precision than presence / absence to adequately capture dynamics.

Third, VA does not possess a well-developed separation of direction and rate of change for discretely valued variables. This means that VA cannot represent changes in the length of time required for discrete variables to change e.g. life-stage. VA represents the time required to change from one stage to another using a constant. In combination with its presence / absence representation this means that if a life-stage transition is due to occur after 10 years, the age of a cohort is 9 years and a disturbance occurs that moves the cohort back a life-stage then it will take the whole 10 years for the transition to be possible again. VA contains no notion of disturbance or other environmental factors only slightly increasing or decreasing the time required for a transition to occur. FATE addresses this issue by increasing the resolution of life-stage abundance and by resetting cohort ages in disturbance- and type-specific ways. This is a possible solution but it is not clear how to define cohort age and or how to reset it in this way. A more general method would be preferable for reasons of simplicity and consistency.

Fourth, the representation of, and reasoning about, relationships pertaining to inter-specific interactions is either absent or similarly hard-wired. Bond and van Wilgen (1996) note that VA adopts a model of non-interaction between species, solely relying on species attributes to determine the likely outcome of vegetation change. They also note that because of good evidence to suggest such interactions are dynamically important that VA may be limited in application. FATE provides such interaction capability.

Fifth, none of the current functional attribute techniques handle environmental dynamics or feedbacks between vegetation and environment. In vegetation such as that found in the Mediterranean, where factors such as water availability and flammable litter build-up are important in determining when, where and how vegetation changes, it may be necessary to include non-disturbance environmental dynamics to adequately simulate vegetation.

2.3 Artificial Intelligence Techniques

2.3.1 Introduction

Due to their ability to represent and reason with different types of knowledge and their focus on solving complex problems, it has been recognised that techniques from the field of artificial intelligence (AI) are well suited to the domain of ecology (Noble 1987, Davis *et al.* 1989, Rykiel 1989, McRoberts *et al.* 1991, Robertson *et al.* 1991, Fedra 1995, Robertson *et al.* 1995, McGlade 1999). The symbolic computation capabilities of AI methods permit representation and reasoning of non-quantitative value and relationship knowledge and in doing so provide a set of methods for modelling and decision-analysis (Shannon *et al.* 1985, Noble 1987, Kitzmiller 1988, McRoberts *et al.* 1991, Fedra 1995).

We will evaluate two categories of AI techniques - those broadly belonging to the field of knowledge-based systems (KBS) and knowledge-based modelling, and those belonging to the field of qualitative reasoning (QR).

2.3.2 Knowledge-Based Systems and Knowledge-Based Dynamic Modelling Techniques

2.3.2.1 Knowledge-Based Systems Review

One of the first types of KBS to be developed were 'diagnostic expert systems' such as the medical consultation system, MYCIN (Shortliffe 1976). Since then a wide range of techniques and applications have been developed across a range of domains. The act of constructing a KBS can be viewed as a process of modelling the world (Clancey 1993, Ford *et al.* 1993, Gaines 1993, Luger and Stubblefield 1993).

Knowledge-based systems are useful for addressing problems in domains where understanding is imprecise and/or indeterminate and conclusions are arrived at using heuristic (rule of thumb) rather than formal mathematical reasoning. They are capable of representing non-quantitative and heuristic domain knowledge using knowledge representation schemes often based on predicate calculus and 'if ... then' rules. KBS are generally composed of variables related to each other by logical operators (i.e. and, or , not) within rules rather than by mathematical expressions. In addition they possess a characteristic architecture whereby knowledge is represented separately from the system employed to make inferences from it. The separation of the knowledge-base from inference engine is a key component of KBS. It is a modular design that facilitates the revision and updating of knowledge, making such systems easier to maintain. In addition, it permits system developers to concentrate on specifying what is known to be true about a domain and the knowledge to be represented, free from concerns about processing and computation.

Various types of inference procedure can be employed in KBS. Two of the most common are forward chaining and backward chaining. With forward chaining, the inference engine attempts to determine possible consequences from a given body of knowledge. This is a form of deduction and is appropriate to use if the consequences of current or postulated domain knowledge need to be explored. It requires a set of domain facts along with a set of valid rules for generating new facts. With backward chaining the inference engine attempts to determine whether a particular goal (or hypothesis) is true given a particular knowledge-base. This is essentially a form of abduction – reasoning from a true conclusion to the premises that may have caused the conclusion, and it is appropriate if the truth of a hypothesis or fact is to be determined given a particular knowledge-base. Forward chaining is easier to implement than backward chaining, which is more often indeterminate (i.e. producing multiple solutions) and may involve the postulation of several intermediate hypotheses in order to determine the truth of the original goal. Shannon *et al.* (1985) and Luger and Stubblefield (1993) provide more detail on KBS representation methods and inference procedures.

Vegetation ecology is a domain characterised by complex causality, simultaneous operation over multiple spatial and temporal scales and incomplete knowledge about process, pattern and constraint. Consequently, knowledge-based systems have been noted by various authors as offering great potential for addressing ecological problems (Nanninga and Davis 1985, Davis *et al.* 1989, McGlade 1999). Vegetation ecology in particular has been noted by various authors as potentially benefiting from the application of KBS techniques (Noble 1987)

Much ecological knowledge can be expressed in terms of 'if .. then' rules, one of the key representational constructs used in KBS. Such rules could be used to determine the value of intermediate variables in during simulation. When many such fragments of knowledge are encoded and held within a knowledge-base, non-obvious and non-trivial conclusions can be derived through the application of a suitably designed inference engine.

Noble (1985) details an expert system designed to provide advice on classifying species into functional types according to the Vital Attributes scheme of (Noble and Slatyer 1980)). Sequential questions and 'if ... then' rules are used to determine the attributes and type of each species.

Davis *et al.* (1989) detail an expert system shell (GEM-II) and a knowledge-base (FIRES) for fire management and risk assessment developed within it. FIRES was constructed based upon empirical knowledge and contained 105 'if ... then' rules with estimates of certainty and explanations. The rules were organised into a fire behaviour KB and a separate fire effects KB. An example fire behaviour KB rule:

> **IF** the season is cool **AND** the 3pm humidity is less than 40% **AND** the wind strength is moderate **AND** the fuel type is one of [annual, sorhum, perennial] **AND** the degree of curing is 100% **THEN** there is strong evidence that the fire danger is low (certainty 90%)

The fire effects rules represented available knowledge concerning the effects of flames of seven different height classes on over ninety species by height (eight different classes). Both KBs were developed such that quantitative information could be represented where appropriate and needed. For example, consider the quantitative fire effect rule:

> **IF** available biomass < (2 * grass biomass) **THEN** the fire type is a grass fire.

The two KBs were then jointly queried from the GEM-II shell to determine fire risk and effect on vegetation using a forward chaining procedure.

2.3.2.2 Review of Knowledge-Based Dynamic Modelling

In addition to standard KBS techniques, which can be viewed as modelling the world statically (i.e. without a temporal dimension), there exist a range of knowledge-based techniques and applications for modelling the world dynamically. Most of these techniques use at least some of the principles and methods of KBS, primarily logic and rule-based representation and reasoning.

Extending the notion of rule-based expert systems to incorporate change over time, Plant and Loomis (1991) detail a model-based reasoning system that forms part of an integrated agricultural decision support system. They note that static KBS are limited with respect to their ability to predict dynamics like vegetation growth and develop a rule-based qualitative simulation methodology to counter this problem.

The fundamental element in their system is the component, representing storages of material or things. A model can have many components and each component possesses a discrete state value and a rate value. States represent the amount of a quantity (e.g. 10 litres of water). Rates represent the direction of change and can either be decreasing, zero or increasing. They use 'if ... then' rules to specify how the states and rates of each component are influenced by other component states and rates and inputs (exogenous variables). Some rules may jointly determine both item rates and states in the same rule whereas other rules may only determine rate based upon state(s). All rules are used to determine how item rates and states change dynamically over time through the application of a forward chaining algorithm which, for each time point (time is represented using integers), accepts a set of input and component values then tests and fires all rules until no further component value changes are detected (constant system state). Time is then incremented by 1 and the procedure repeated under the next set of input values. The procedure used to reason about change in quantity values imposes an equilibrium dynamics assumption on model variables.

Starfield *et al.* (1989) develop a rule-based model to support management of a coastal lake based using discretely valued variables to predict how the biomass of various biotic components (e.g. phytoplankton, reeds, 3 species of underwater plant) might change over time. The biotic components were valued using the same 5 element qualitative support set whilst the 2 abiotic variables (salinity and lake level) were valued using 6 element and 3 element qualitative support sets respectively. Formulated from empirical observations, a set of 'if ... then' rules were used to update biotic variable values once every 3 months (once per season). Each rule specifies the direction of value change and the amount (how many value elements). For example:

IF salinity = value 5 **THEN** the reed biomass will die back slowly (drop 1 value per quarter).

Using their simple rule-based model they demonstrate that it is possible to simulate ecological dynamics based on only coarse-grained, non-mechanistic and incomplete knowledge.

In the context of providing decision-support in ecology Robertson *et al.* (1995) explore a range of logic-based techniques for representing and reasoning with available ecological knowledge. They describe various approaches to modelling agro-forestry systems using coarse-grained ecological knowledge with the aim of predicting the effects of different management options.

They detail a process by which such systems can be modelled, first of all by identifying state and intermediate variables then by constructing a signed digraph of model structure and relationship nature (positive or negative influence directions). A three element qualitative support set ({low, medium, high}) is used to value all variables. To carry out qualitative simulation on such models the influences on state variables are resolved from one time-step to another through application of a general algorithm whereby, for each state variable, the values of all positive influences are averaged, as are the values for all negative influences. If the positive average > the negative average then the value of the state variable concerned is increased by 1 element at the next time-step. If the negative average > the positive average then the state variable value is decreased by 1 element at the next time-step. It is not clear how qualitative values are to be averaged and not clear whether such an activity can be done at all in a meaningful manner i.e. is there a general procedure for calculating the average influence 'strength' for any set of *n* influencing real-world quantities? Will the average of low + low + medium always be the same irrespective of the actual 3 quantities and the support sets involved? However, they acknowledge the simplicity of their approach and that it may suffer from problems.

2.3.2.3 Functionality Evaluation

Table 7 details the knowledge-based system and modelling functionality capability rating. The general features and principles of KBS and KB modelling techniques make them well suited to representing and reasoning with available vegetation knowledge. Using factual and rule-based symbolic representation there is little or no *a priori* restriction on the value and relationship knowledge that can be expressed between ecological quantities. In the same way that high-level natural language statements can be represented, so coarse-grained qualitative or linguistic values and relationships can be expressed. In addition, the separation of representation from reasoning provides useful modularity and flexibility, permitting model structure, value and relationship knowledge to be specified declaratively, relatively free from computational concerns.

Functionality	Rating
Structural Representation	
Community (vegetation) types	3
Community attributes	3
Community properties	3
Species attributes	3
Species properties	3
Inter-specific interactions	2
Intra-specific interactions	2
Non-disturbance environmental influences	3
Disturbance influences	3
Vegetation-environment feedback and dynamics	1
Value Representation	
Support sets:	
Real values	3
Integer values	3
Qualitative values	3
Ordered linguistic values	3
Unordered linguistic values	3
Direction of change	3
Relationship Representation	
Quantitative relationships	3
Non-quantitative relationships	3
Mixed support set relationships	3
Reasoning	
Reasoning with quantitative relationships	3
Reasoning with non-quantitative relationships	3
Reasoning with mixed relationships	3
Time-driven simulation	3
Deterministic reasoning with state variables	3
Non-deterministic reasoning with state variables	2

Table 7 Knowledge-based systems and modelling functionality

However none of the specific approaches and examples detailed offer an 'off-the-shelf' solution. Rather, with some potentially re-useable constructs and concepts, they provide a framework within which domain and problem-specific solutions can be developed. Through application in state transition and functional attributes modelling as well as the knowledge-based techniques detailed in this section, it has been shown that 'if ... then' rules can be used to represent a range of quantitative, non-quantitative and mixed support set relationships. It is relatively easy to express conditional knowledge relating the value(s) of one quantity to the value(s) of others and to represent causal relations between quantities using the formalism of logic. This knowledge could be used to in determining intermediate variable values using standard non-temporal deductive inference methods. However it is less obvious to reason with rule-based knowledge to make useful, accurate predictions regarding dynamics. Forward chaining may provide a useful deductive framework but does not in itself offer a solution to reasoning about change over time.

Due to the multi-scale, complex nature of ecological dynamics, different ecological quantities are likely to change at different rates (Wiens 1989). The KBS techniques examined do not provide a satisfactory general means of dealing with changes in discretely valued variables over time, particularly where variables may change at different and varying rates. Ideas involving separate consideration of variable state (value), direction and rate of change may offer a solution.

Two of the key problems to address in modelling using available knowledge:

1. The *direction of change* problem - determining how discretely valued state variables are changing. How to represent relationships to determine direction of change and how should they be reasoned with to model changes in direction of change over time?
2. The *rate of change problem* – given a particular direction of change, when should a discrete state variable change from one value to the next? For example, if a plant species has a low biomass and is increasing, at what point in time should the species' biomass be updated to medium? In QR this is sometimes called the *state variable problem.*

It is not clear how to reason about change in discretely valued state variable values based upon direction and rate of change. Any given change in value (direction) may take one or more time-steps to occur depending on the rate of change as determined by the values of influencing variables (the *conditions*). If the change in value takes more than a single time-step there is a possibility that either the rate of change (number of time-steps to change value) or the direction of change may change one or more times in response to changes in conditions. How should dynamic changes in rate or direction of change be handled?

The method of separately reasoning with a set of rules to determine a quantity's direction of change (rate) then using this rate to update the quantity's state used by Plant and Loomis (1991) does not provide a solution. They assume equilibrium dynamics and impose a condition whereby discrete variable values *must* change up or down a value during each time-step unless the rate is calculated as zero. This approach cannot handle dynamics whereby a variable takes several time-steps to change value or never reaches equilibrium. Starfield *et al.'s* (1989) notion of using rules to determine the number of discrete states that a qualitatively valued variable changes up or down its support set every time-step fails for the same reasons – it cannot handle variables taking more than one time-step to change a single value in a given direction.

Unfortunately it is improbable that the idea employed by Robertson *et al.*(1995) in determining the 'average' direction (positive, neutral or negative) of all influences on a discretely valued quantity as a means of determining change will be of use. It is unlikely that such averaging can be meaningfully carried out in a general way for ecological quantities. Ecological quantities may be related in complex, non-linear ways, making their relationships resistant to simple averaging e.g. will two but opposite influences (one positive and one negative) always average to zero? Without further information this cannot be resolved.

2.3.3 Qualitative Reasoning Techniques

2.3.3.1 Introduction

The artificial intelligence field of qualitative reasoning has been in existence since the early-mid 1980's when various authors such as de Kleer and Brown (1983), de Kleer and Brown (1984), Forbus (1984) and Kuipers (1986) proposed methods of representing and reasoning about the behaviour of physical systems without resort to complicated quantitative models. Williams and de Kleer (1991) state that 'the heart of the qualitative reasoning enterprise is to develop computational theories of the core skills underlying engineers, scientists and just plain folk's ability to hypothesise, test, predict, create, optimise, diagnose and debug physical mechanisms'. Bonissone and Valavanis (1985) denote this 'common-sense' knowledge. Using imprecise and indeterminate (i.e. incomplete) knowledge and information, human experts have been widely noted within the QR literature as being able to reason about the behaviour of physical systems without resorting to detailed, quantitative methods. Kuipers (1994) notes that using incomplete knowledge to reason about the world is ubiquitous and that a primary motivation for QR is to represent and reason with coarse-grained descriptions that are based upon and capture qualitatively important behavioural distinctions regarding the systems they model. The field of QR has endeavoured to develop systems for representing and reasoning based on such distinctions. This has resulted in a range of techniques capable of handling structural, non-quantitative and mixed value and relationship knowledge being developed and applied, mostly within engineering and related fields.

Outwith such areas, QR techniques have received some attention particularly in those fields where knowledge is imprecise and / or indeterminate. It has been recognised that ecology is one such field. Consequently, several QR and qualitative simulation techniques have been applied to and developed for the modelling of biological and ecological systems e.g. Guerrin (1991), Guerrin (1992), Guerrin *et al.* (1994), Guerrin *et al.* (1997), Hunt and Cooke (1994), Kuipers (1994), Salles (1997) and Salles and Bredeweg (1997). QR techniques have demonstrated capabilities in modelling with non-quantitative knowledge and may provide an 'off the shelf' method for modelling using available vegetation knowledge. Consequently they will be given an in-depth evaluation.

Several reviews are available covering the major concepts and techniques developed by the QR field (Bonissone and Valavanis 1985, Rajagopalan 1986, Forbus 1988, Cohn 1989, Guerrin *et al.* 1994). There are three major QR techniques and the field is categorised according to them:

- Component-based QR - the Confluences system of de Kleer and Brown (1983) and de Kleer and Brown (1984).
- Constraint-based QR - QSIM (Kuipers 1986, Kuipers 1994).
- Process-based QR - Qualitative Process Theory (QPT) (Forbus 1984).

In addition there has been interest in the application of symbolic qualitative algebras like SIMAO (Guerrin 1991, Guerrin 1992) in the ecological modelling and QR literature. SIMAO will be evaluated as an example of this type of technique.

Both the component-based and constraint-based QR techniques utilise qualitative differential equations (QDEs) as the basic construct with which to represent relationships. QDEs are symbolically valued versions of ordinary differential equations (ODEs). Component-based QR adds to this by constraining the structure of models to a particular set of constructs that go to make up the 'device ontology'. The device ontology in the Confluences system (de Kleer and Brown 1983, de Kleer and Brown 1984) structures the world in terms of well-defined component models that act to modify the properties of materials (namely 'pressure' and 'flow') and which are connected into a fixed topology by means of conduits, channels for transporting materials. Only a single material can be modelled in any given Confluences model. All the components taken together comprise the whole device with each component being described in 'local terms' – keeping whole device structure and behaviour separate from component specification.

Components can operate in one or more qualitative states depending on conditions (e.g. a valve may be open or closed) and each state is modelled using a set of QDEs. A qualitative simulation process called envisioning is used to produce a *total envisionment* for each Confluences model. The Confluences total envisionment is a complete description of all possible qualitative states for the whole device described in terms of component model states and the transitions between them. Each qualitative whole device state is a single combination of component model states that are described by one or more sets of QDE variable value assignments. Transitions between component states are described by means of a directed graph where states are nodes and legal transitions are arcs. Transition graphs are constructed by a rule-based method analogous to solving simultaneous differential equations (see de Kleer and Brown 1984 for more details).

As Salles (1997) notes the device ontology is too restrictive to be widely applied in modelling with ecological knowledge. The ontology was designed to model well-bounded engineering devices such as electronic circuits where only a single 'substance' is present throughout the whole system and where the system has a discrete, non-changing boundary. The ontology cannot handle the types of systems that occur in ecology where boundaries are less clear and may change depending on model purpose (with behavioural implications), where the inclusion of exogenous variables (possibly non-material in form e.g. a fire trigger) is the norm and where it is not always clear how to decompose the 'system' into components or quantities. The device ontology assumes that components are readily identifiable and their qualitative behaviours are already known. This is not the norm in ecology. In addition, the ontology cannot represent structural knowledge where the notions of flow and manipulation of materials do not apply such as in vegetation state transition models nor can it represent situations where multiple substances are involved.

For these reasons the component-based approach will not be fully evaluated. However, the approach's use of QDEs to represent relationships and some of its methods of reasoning with them (e.g. qualitative calculus, the use of qualitative derivatives, generate and test constraint satisfaction and propagation) may be useful for modelling with available vegetation knowledge. Constraint-based QR (QSIM) also uses QDEs (not syntactically identical) to represent relationships and, along with process-based QR (QPT), also utilises many of the same basic methods for reasoning about change in qualitatively valued variables. So, despite not being evaluated separately, potentially useful features from component-based QR will still be evaluated.

2.3.3.2 Ecological Modelling Evaluation Method

Due to a lack of published ecological applications, the job of evaluating QR techniques with regards their ability to represent and reason with available vegetation knowledge is more difficult. Salles (1997) carried out some limited evaluation and application of QPT and SIMAO to modelling vegetation dynamics. However his analysis was inconclusive as will be discussed. To counter the lack of material for the evaluation process each of the QR techniques will be 'tested' using a standard ecological modelling problem – two competing species. Up until the present there has been no comparative evaluation of the ability of different QR techniques to represent and reason with standard ecological modelling problems. The evaluation to follow represents a novel analysis of each technique.

Mathematically two competing species can be modelled using the Lotka-Volterra-Gause (LVG) equations (see for example Ricklefs 1990). Figure 2 details a graphical representation of the LVG model using System Dynamics or compartment-flow notation.

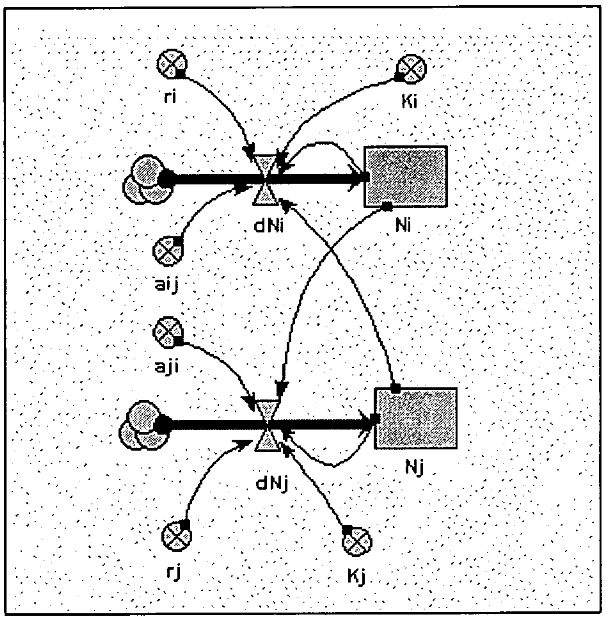


Figure 2 System Dynamics representation of the LVG 2 species competition model

The LVG equations represent a simple 2 species population model based upon the logistic growth equation extended to incorporate inter-specific interactions through the notion of competition coefficients that represent the effect of one species on another.

The Lotka-Volterra-Gause Population Model

$dN_i / dt = r_i N_i (1 - N_i / K_i - \propto_{ij} N_j / K_i)$ Eq.1

$dN_j / dt = r_j N_j (1 - N_j / K_j - \propto_{ji} N_i / K_j)$ Eq.2

where,

dN_n / dt = rate of change over the time t in the population size of species n.

r_n = the *per capita* birth rate of species n such that $0 \leq r_n \leq 1.0$.

N_n = the population size of species n.

K_n = the population size of species n at carrying capacity.

$\propto_{n\,m}$ = the competitive effect of species m on species n expressed as a number such that $0 \leq \propto_{n\,m} \leq 1.0$.

For the purposes of the evaluation procedure it will be assumed that $0 \leq K_n \leq 100$. Expressed as a set of difference equations suitable for numerical simulation, the LVG model becomes:

$N_{i(t)} = N_{i(t-1)} + r_i N_{i(t-1)} (1 - N_{i(t-1)} / K_i - \propto_{ij} N_{j(t-1)} / K_i)$ Eq.3

$N_{j(t)} = N_{j(t-1)} + r_j N_{j(t-1)} (1 - N_{j(t-1)} / K_j - \propto_{ji} N_{i(t-1)} / K_j)$ Eq.4

where,

$N_{n(t)}$ = the size of the population of species n at time t.

$N_{n(t-1)}$ = the size of the population of species n at time t - 1.

The LVG equations incorporate various species-level structural and behavioural features like intra- and inter-specific competition, differences in reproductive success between species and change in 'community' (albeit a simple two species 'community') structure over time. Techniques to be used for modelling vegetation dynamics based upon available knowledge may need to be capable of incorporating such features.

Each QR technique will be reviewed. Next, the way in which each QR technique represents the structure, value and relationships expressed by the LVG model will be detailed. This will either result in a 2 species competition model being produced for each candidate technique (depending on the capabilities of the techniques and the size of the model construction tasks) or a description of how each technique *could* go about the task. Next the reasoning capabilities of each technique will be explored. The 2 species model produced by each technique will be simulated forward and the results compared with the behaviour produced by an equivalently parameterised and initialised mathematical LVG model. Where this is not possible, the process by which each technique *could* simulate / reason about the dynamics of 2 competing species will be described.

The motivation behind comparing qualitative model behaviour with quantitative model behaviour is to provide a sound, objective basis for evaluating the results of different qualitative modelling formalisms. The idea of using quantitative model results derived by applying mathematical reasoning as a 'gold standard' has been recognised by several authors (Struss 1988, Salles and Bredeweg 1997). It should be noted though that this approach only applies to comparing purely quantitative models with identically structured non-quantitative models that use variables valued with underlying numerical support sets i.e. quantitative and qualitative variables. Comparing quantitative models with models that are structurally different or use linguistically valued variables cannot be directly achieved through comparing results from a simulation exercise.

2.3.3.3 Constraint-Based QR

Review

Representation

QSIM (Kuipers 1986, Kuipers 1994) is the primary example in the QR literature of the constraint-based approach to qualitative reasoning. The technique is essentially a qualitative version of differential calculus and does not enforce a strict ontology beyond structuring models in terms of variables that are continuous and continuously varying functions of time ('reasonable functions of time'). QSIM therefore differs from component- and process-based QR, both of which impose more structured ontologies and both of which are more concerned with representing and reasoning with the deep knowledge that underlies human reasoning about the physical world (Rajagopalan 1986).

The basis to QSIM models are qualitative differential equations (QDEs), representational structures which Kuipers (1994), in his formal derivation of QSIM, proves are actually differential equations and not just similar to them. Kuipers (1994) talks about particular ordinary differential equations (ODEs) as 'satisfying a QDE' and shows that any given QDE can represent many ODEs.

Physical systems are modelled in QSIM as sets of QDEs, each defined by four elements - <V, Q, C, T>. V is the set of variables used by the QDE, Q is the set of quantity spaces used to value the variables in V, C is the set of constraints (algebraic equations) that apply to the variables in V and T is the set of transition rules for the QDE – rules that define, in terms of values of the variables in V, the domain of applicability of each QDE. As the values of variables in V change during a qualitative simulation the QSIM model may transition from one QDE to another depending on the domains for each QDE specified in T. The same variables are used in different QDEs but they may be related to each other in different ways. The idea is to capture qualitatively different operating regions for physical systems e.g. an overflowing container behaves differently from a partially full one.

All variables in a QDE must be reasonable functions of time. However, rather than using the real numbers, QDE variables are valued during simulation using <qmag, qdir> tuples where qmag refers to magnitude (or amount or level) and qdir refers to qualitative derivative. The possible values for qmag are represented using the notion of a quantity space, a concept akin to support sets. Quantity spaces are ordered sets of 'landmark' values, $\{l_1 < l_2 \ldots < l_n\}$ each of which represent numeric values marking qualitatively important distinctions in a quantity's value. For example, the quantity space for the volume of a container could be {0 ... VMAX ... ∞} where the maximum volume, VMAX, is used as it demarks a point of behavioural change for the container. Landmark values may be symbolic or numeric. As QSIM variables are abstractions of real-valued quantities most quantity spaces contain the landmarks $-\infty$, 0 and $+\infty$ unless not required. The value of qmag can either be equal to a landmark value (e.g. qmag = l_1) or the interval between landmark values (e.g. qmag $\in (l_1, l_2)$). The value of qdir can either be 'inc' if the rate of change of the variable concerned is > 0 (i.e. $f'(t) > 0$), 'std' if $f'(t) = 0$ and 'dec' if $f'(t) < 0$.

Constraints (algebraic equations) are used to express relationships between different variables with each variable in a QDE required to appear in at least one constraint. QSIM provides 7 basic constraints to express relations between two variables (addition, multiplication, minus, derivative of, constant, monotonic increasing function, monotonic decreasing function), several advanced non-monotonic constraints (saturating increasing and decreasing functions, concave up and down functions) plus 4 additional constraints to express multivariate relations (multivariate monotonic increasing and decreasing functions, signed-sum, sum, sum-zero). For example, the QSIM addition constraint with its arithmetic meaning:

(add x y z) $\equiv$ x(t) + y(t) = z(t)

The monotonic increasing and decreasing function constraints represent incomplete knowledge that one variable is directly (or inversely) proportional to another:

(M+ x y) $\equiv$ y(t) = f(x(t)), $f \in$ M+, *where* M+ is the set of directly proportional functions.

In addition to the basic structure, each constraint can have a set of corresponding values specified to hold between the variables whose relationship is represented. Corresponding values serve to relate constraints to quantity space values. For example, take the QSIM monotonic increasing function constraint between two quantities, container volume and water level with respective quantity spaces {-∞ ... 0 ... full ... +∞} and {-∞ ... 0 ... top ... +∞):

((M+ amount level) (0 0) (full top))

This represents additional relationship knowledge that the container volume is 0 if and only if (iff) the water level is 0, and that the container volume is 'full' iff the water level is 'top'. Other constraints can similarly be made more precise through the use of corresponding values. There is no limit to the number of corresponding values that can be used. It is important to note that different quantity spaces are completely un-related unless specified through the use of corresponding values in variable constraints.

The qualitative state of a QSIM model is then represented as a set of variables, each valued using a <qmag, qdir> tuple. A QSIM model assumes a new state when a qualitative change occurs i.e. a variable reaches a landmark value or moves off of one to an interval between landmark values.

Reasoning

The basic aim of the QSIM qualitative simulation process is to obtain a set of possible behaviours for a QSIM model that are consistent with the specified QDE constraints. Qualitative simulation determines all possible qualitative states consistent with a QSIM model and a given starting state, organised into a 'behaviour tree'.

Kuipers (1994) views the qualitative simulation process as taking a QSIM model and a starting qualitative state to prove a theorem of the form:

> QSIM model $\wedge$ QState (t_0) $\rightarrow$ or(Behaviour $_1$, ... , Behaviour $_n$)
>
> *where,*
>
> Behaviour = [QState (t_0), QState (t_0, t_1), ... , QState (t_n)]

The computational basis to the qualitative simulation algorithm (algorithm QSIM – Kuipers 1994 p.103) is viewing the process as solving a constraint satisfaction problem (CSP). Kuipers (1994) defines a CSP as involving a set of (QDE) variables (V), a set of value sets (quantity spaces) describing the possible values for those variables (D) and a set of (QDE) constraint relations (P) where each constraint refers to a subset of variables in V and every variable in V appears in at least one constraint. An assignment is defined as being an n-tuple of variable values from D and a valid assignment as an assignment whose n-tuple satisfies every constraint in P. The solution to a CSP is a set of valid assignments.

As the basic CSP solution algorithm of generating every possible assignment and testing them against QDE constraints is generally intractable, QSIM uses a filtering algorithm (Cfilter - Kuipers 1994 p.83) to prune the assignment search space as it works. Cfilter forms the basis for the operation of the qualitative simulation algorithm, which is a form of breadth-first search.

Before qualitative simulation can begin each QSIM model must be initialised. The generation of an initial state or states, QState (t_0), is a separate process from simulation. Based upon a set of initial QDE variable values a constraint propagation algorithm (algorithm State-Completion, Kuipers 1994 p.93) is employed to produce all possible complete and consistent states. Complete states are defined as those where every variable has a completed <qmag, qdir> tuple. If the set of initial variable values is complete then no constraint propagation occurs. If the set of initial values is incomplete the State-Completion algorithm essentially 'fills in the blanks' by positing values for non-valued variables, evaluating them against the QDE constraints (see below for details) then propagating the changes around the other constraints until a complete QState is found. The algorithm Cfilter is then used to determine all possible QState (t_0) completions consistent with the first one to be derived. These comprise the set of initial model states.

The QSIM qualitative simulation algorithm:

1. Initialise the QSIM agenda with a set of complete initial states, QState (t_0).
2. If the agenda is empty stop simulation.
3. If the agenda is not empty then pop a state S from the agenda i.e. take the first state.
4. For each variable in the QDE use the set of QSIM qualitative *successor rules* (see below) to determine possible next value(s) – the set D.
5. Determine all successor states (union of variable values) consistent with D and the QDE constraints (achieved through the process of *constraint evaluation*, see below).
6. For each successor state assert the relations successor(S, S_i) and predecessor(S_i, S) (*where* S = current state, S_i = successor state) and add new behaviours to the behaviour tree / transition graph.
7. Apply global filters to each successor state (see below) to prune results.
8. Add eligible successor states (i.e. that pass the application of global filters) to the agenda.
9. Continue from step 2.

If during simulation the domain of applicability for a QDE is not satisfied by the value of its variables then the QDE ceases to be used and another becomes 'active' instead depending on the set of QDE transition rules.

During the initialisation and simulation processes the consistency of new variable values are checked against the constraints in the QDE being used. The process of constraint evaluation is fundamental to QSIM and is based upon using sign algebra to determine whether the constraints are satisfied by possible value assignments. QSIM uses four operators to express the set of conditions associated with each constraint in terms of relations over sign-valued terms (Table 8).

$[x]_0$ = sign(x) = [+] if $x > 0$, [0] if $x = 0$, [-] if $x < 0$	$[x]_{x0}$ = sign($x - x_0$), *where* x_0 serves as a reference value for x	$[x']$ = $[dx/dt]$ = sign(dx/dt)	$[x]_\infty$ = [+] if $x = +\infty$, [0] if x is finite, [-] if $x = -\infty$

Table 8 QSIM sign-valued operator definitions

Each QSIM constraint (addition, multiplication etc.) has a set of conditions that must be satisfied. For example, assignments to the addition constraint (add x y z) must satisfy 3 constraints during constraint evaluation. The first condition:

> ([x'], [y'], [z']) must satisfy [x'] + [y'] = [z'] *where* qualitative addition is defined as a relation returning a truth value (Table 9).

If, during constraint evaluation a value assignment fails to satisfy any condition, that assignment is discarded.

Add	[+]	[0]	[-]
[+]	[+]	[+]	[+] / [0] / [-]
[0]	[+]	[0]	[-]
[-]	[+] / [0] / [-]	[-]	[-]

Table 9 QSIM qualitative addition definition (note that three possibilities can return true for both [-] + [+] and [+] + [-])

Time in QSIM is represented as alternating between 'distinguished time points' and intervals between distinguished time-points. No absolute time-scale is employed. 'Distinguished time-points' are defined as occurring whenever a qualitative variable's qmag value moves from an interval between landmark values to a single landmark value. Time-intervals occur between time points to represent continuity of change in variable values i.e. to represent variables having to change continuously between their landmark values.

The process of determining the successor value to a QSIM variable is called *limit analysis* and involves the use of two sets of *successor rules* (see Kuipers 1994 p.99-100). These rules determine, for every possible combination of <qmag, qdir> value tuples, how a variable's value may change. There are two basic ways in which a variable may change value during simulation – from a distinguished time-point to an interval between distinguished time-points or from an interval between distinguished time-points to a distinguished time-point. Correspondingly there are two sets of successor rules – *P-Successors*, which govern changes in value from a time-point to a time-interval and *I-Successors*, which govern changes in value from a time-interval to a time-point. The rules are given in Table 10.

Qualitative simulation may never terminate as the agenda may never empty (Kuipers 1994). This occurs as QSIM QDEs usually do not sufficiently constrain possible dynamics thereby creating a degree of indeterminacy. As a result for any given state there are often multiple eligible successor states and a branching simulation process. During simulation QSIM can dynamically generate new landmark values for variables if they are found to behave in a qualitatively distinct way previously unspecified (see Kuipers 1994 for details). This can exacerbate the branching behaviour of a simulation. Indeed branching can often become intractable.

P-Successors		**I-Successors**	
<qmag, qdir> t_i	**<qmag, qdir> (t_i, t_{i+1})**	**<qmag, qdir> (t_i, t_{i+1})**	**<qmag, qdir> t_i**
$< l_j$, std >	$< l_j$, std >	$< l_j$, std >	$< l_j$, std >
$< l_j$, std >	< (l_j, l_{j+1}), inc >	< (l_j, l_{j+1}), inc >	$< l_{j+1}$, std >
$< l_j$, std >	< (l_{j-1}, l_j), dec >	< (l_j, l_{j+1}), inc >	$< l_{j+1}$, inc >
$< l_j$, inc >	< (l_j, l_{j+1}), inc >	< (l_j, l_{j+1}), inc >	< (l_j, l_{j+1}), inc >
$< l_j$, dec >	< (l_{j-1}, l_j), dec >	< (l_j, l_{j+1}), inc >	< (l_j, l_{j+1}), std >
< (l_j, l_{j+1}), inc >	< (l_j, l_{j+1}), inc >	< (l_j, l_{j+1}), dec >	$< l_j$, std >
< (l_j, l_{j+1}), dec >	< (l_j, l_{j+1}), dec >	< (l_j, l_{j+1}), dec >	$< l_j$, dec >
< (l_j, l_{j+1}), std >	< (l_j, l_{j+1}), std >	< (l_j, l_{j+1}), dec >	< (l_j, l_{j+1}), dec >
< (l_j, l_{j+1}), std >	< (l_j, l_{j+1}), inc >	< (l_j, l_{j+1}), dec >	< (l_j, l_{j+1}), std >
< (l_j, l_{j+1}), std >	< (l_j, l_{j+1}), dec >	< (l_j, l_{j+1}), std >	< (l_j, l_{j+1}), std >

Table 10 QSIM successor rules

Global filters are applied during the simulation process to help reduce the proliferation of branching and to help prevent the appearance of physically impossible qualitative states. Kuipers (1994) details 9 'state filters' that filter out states based upon current or immediate predecessor state information and 4 'behavioural filters' that filter out possible states based on information from all states leading up the current one. An example state filter is the *no change filter* which removes any time-point states at which no variables change qualitative value.

Recognised Problems

The values within the sets {-,0,+}, {-,0,+,?} or {-∞,0,+∞} used by QSIM operate like intervals, where for example (-) represents the interval (-∞,0), (0) represents the interval with 0 width (0,0) and ? represents the interval (-∞,+∞). Any application of constraints to them turns out to be some form of interval arithmetic subject to some of the same problems (Struss 1988, Williams 1991). The nature of the problems that may be experienced with sign intervals do not disappear when more detailed interval representations are used such as finite landmark quantity spaces - in fact they get worse (Struss 1988). Some of the major properties and problems:

1. *Ambiguity of sign addition and subtraction*:

e.g. [x] + [y] $\approx$ 0 (*where* $\approx$ denotes qualitative equality) is true if <x,y> is <0,0>, <+,-> or <-,+>. This is one of the major causes of the excessive branching problem experienced in QR.

2. *Constraints on intervals are sensitive to the direction in which they work*:
   This means that mathematically equivalent expressions may produce different results because the way in which the constraint has been solved e.g. (taken from Struss 1988) x1-x2 = x3 may not produce the same results as x1 = x2+x3 illustrated by (3,5) = (1,2) + (2,3), whereas (3,5) - (1,2) = (1,4) $\neq$ (2,3).
3. *Lack of associativity for operations on finite landmark QS other than* $\{-\infty, 0, +\infty\}$:
   e.g. D = (A+B)+C may not produce the same result as D = A+(B+C) – the order of calculation in model expressions may determine the solution obtained.
4. *Lack of distributivity for operations on finite landmark QS other than* $\{-\infty, 0, +\infty\}$:
   e.g. A*(B+C) = (A*B)+(A*C) under normal arithmetic but under interval arithmetic only the weaker relation A*(B+C) $\subseteq$ (A*B)+(A*C) holds.
5. *The 'Selection' Problem*:
   If a variable appears more than once in a set of constraints then reasoning with the constraints to infer quantity values may produce results with unnecessarily wide intervals e.g. (taken from Kuipers 1994) solving y = x – x where x$\in$ (0,∞) gives y = (0,∞) - (0,∞) = (-∞,∞) gives a wider and more ambiguous interval.
6. *Weakness of qualitative equality*:
   Using the qualitative equality relation $\approx$ we can not substitute 'equal for equal' in a model expression as equality is no longer transitive as it is in ordinary arithmetic – thus a$\approx$b and b$\approx$c does not mean that a$\approx$c.

The existence of these problems means that it may not be possible to directly represent model relationships in their standard mathematical form to ensure that qualitative constraint results are not term or calculation order dependent or subject to producing unnecessarily wide intervals. Instead real-valued algebraic relationships may need to be manipulated prior to representation as QSIM sign-valued constraints. This may not be a trivial task and may not be possible at all (Kuipers 1994 provides some examples). For example, Struss (1988) shows that there is no QR technique operating on a finite set of landmark values other than $\{-\infty,0,+\infty\}$ that preserves the desireable properties of associativity and distributivity.

With respect to QSIM, Struss (1988) shows that:

1. Solutions / predictions are sensitive to changes of expression.

2. Solutions / predictions are sensitive to changes of order.
3. Wrong solutions / predictions may be produced.

Ecological Modelling Evaluation

Representation

Kuipers (1994) describes a method for transforming certain 'suitable' ODEs into QDEs through a process called structural abstraction in which the ODE is re-written as a set of simultaneous equations (one for each ODE sub-expression) using only the operators addition, multiplication and negation along with functions of continuous and non-zero derivative. Each simultaneous equation can then be easily re-written as a qualitative constraint.

Eq.1 and Eq.2 were taken as the basis for constructing a QDE representation of the LVG model. The right-hand side of these equations can vary in terms of sign and magnitude during a simulation depending on the relative population sizes, the relative reproductive success of the species and their relative competitive strengths. Expanding the brackets gives:

$$dN_i / dt = r_i N_i - r_i N_i^2 / K_i - \propto_{ij} r_i N_i N_j / K_i \qquad \text{Eq.5}$$

$$dN_j / dt = r_j N_j - r_j N_j^2 / K_j - \propto_{ji} r_j N_j N_i / K_j \qquad \text{Eq.6}$$

The right-hand side of each equation now consists of 3 components (the additive terms taken from right to left) – recruitment, intraspecific competition mortality and interspecific competition mortality. However, the components behave non-linearly with respect to each other and time making representation using QSIM constraints difficult. For example, the recruitment component will continue to increase as population size increases but if it approaches the carrying capacity (K_n) it will be increasingly opposed and eventually equalled by intraspecific competition induced mortality. This will zero the rate of change (dN_n/dt). Interspecific mortality will simply continue to increase in proportion to the population size of the competing species until they level off at carrying capacity. If the interspecific competition component is large enough then the mortality induced may cause the affected species to stabilise at a level lower than its carrying capacity. If it is even stronger then the affected species may assume a negative rate of population change and possibly become extinct ($N_n = 0$). Figure 3 details the QSIM constraint network used to represent the LVG model. Figure 4 defines the QDE model using the syntax of QSIM.

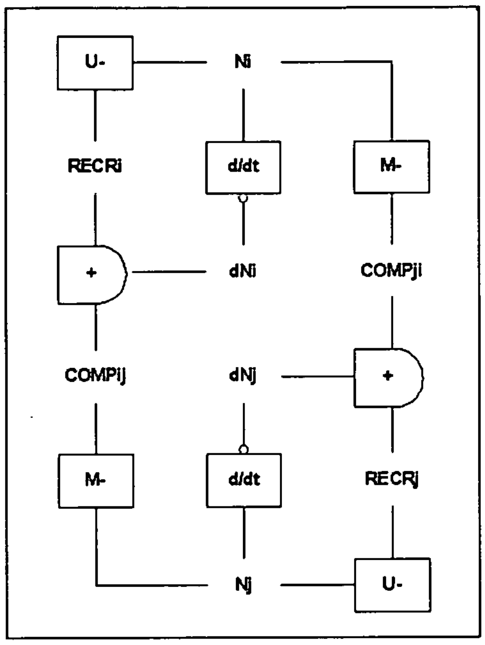


Figure 3 LVG model qualitative constraint diagram (unboxed nodes are QDE variables, boxed nodes are QSIM constraints, arcs denote how variables are related using constraints)

QDE for LVG 2 species competition model

```
(define-QDE plant-community
        (quantity-spaces
                (Ni (0 RMAXNi Ki +∞))
                (RECRi (0 RMAXRi +∞))
                (COMPji (-∞ 0))
                (dNi (-∞ 0 +∞))
                (Nj (0 RMAXNj K +∞))
                (RECRj (0 RMAXRj +∞))
                (COMPij (-∞ 0))
                (dNj (-∞ 0 +∞)))
        (constraints
                ((M- Ni COMPij)              (0 0))
                ((U- Ni RECRi)               (0 0) (RMAXNi RMAXRi) (Ki 0))
                ((add COMPij RECRi dNi)      (-∞ +∞ 0) (0 0 0))
                ((d/dt Ni dNi))
                ((M- Nj COMPji)              (0 0))
                ((U- Nj RECRj)               (0 0) (RMAXNj RMAXRj) (Kj 0))
                ((add COMPji RECRj dNj)      (-∞ +∞ 0) (0 0 0))
                ((d/dt Nj dNj)))
        (transitions))
```

Figure 4 QSIM QDE version of the LVG model

The main model variables used (one for each species) were population size (N_n), rate of change of population size (dN_n), absolute recruitment rate ($RECR_n$) and interspecific competition mortality rate ($COMP_{nm}$). The recruitment rate and the intraspecific competition mortality rate are both functions of a species' population size. Intraspecific competition mortality rate will always be equal to zero when $N_n = 0$. It was therefore viewed as a reduction in recruitment rate rather than a separate factor and the two components were lumped together into a single recruitment rate variable ($RECR_n$).

The QSIM 'concave up' function constraint (i.e. a parabola pointing up), U-, with value correspondences was used to capture the fact that $RECR_n = 0$ when $N_n = 0$, will increase directly with N_n up to a point after which it will decline with $RECR_n = 0$ again when $N_n = K_n$. To represent the function in QSIM a value correspondence between N_n and $RECR_n$ needs to be used to locate the apex of the parabola – the point at which the function changes from direct to inverse proportionality. On the N_n variable quantity space this point is marked by the value $RMAXN_n$, the maximum absolute recruitment rate. The corresponding absolute recruitment rate point in the dependent variable, $RECRR_n$ specifies the apex fully. Note, that although the apex has been specified, the actual numeric values represented by $RMAXN_n$ and $RECRR_n$ will be different depending on conditions (r_n, K_n and $\propto_{nm}$) - the correspondence does not represent a single static point. The amount of mortality caused by interspecific competition ($COMP_{n\ m}$) was represented as a variable with a quantity space of value range from 0 to $-\infty$, related to the population size of the species exerting the competitive effect using a monotonic decreasing function. The rate of change of population size (dN_n) was then defined as $RECR_n$ plus $COMP_{nm}$ using the add constraint and dN_n related to N_n using the derivative of constraint. It was not clear how to represent the two species having different values for r_n, K_n and $\propto_{nm}$ using QSIM constraints and quantity spaces as could be done in the numeric LVG model. QSIM provides no order of magnitude operators to express differences in the values of these parameters.

Qualitative Simulation

The simulation was initialised using a single complete state with species *i* at carrying capacity and species *j* at less than carrying capacity but greater than zero. The QSIM LVG model was hand-simulated following the QSIM qualitative simulation algorithm. No global filters were applied. Table 11 details the results.

The transition from distinguished time-point t_0 to the interval (t_0, t_1) is a point to interval so the P-Successor rules were used. Only some of the possible states that could be generated at time (t_0 , t_1) were generated and listed in Table 11. It became obvious during the hand-simulation that the recognised problem of QSIM branching excessively was occurring. Based on the P-Successors a total of 36 possible states (unique value assignments) could be generated. After testing these states against the LVG QDE constraints a total of 4 unique successor states were possible. Each of these could then go on to branch during the next time-step, an interval to point transition using the I-Successor rules. This step would generate a total of 21,760 possible states based on the I-Successor rules. Despite the fact that this number would be significantly reduced under constraint evaluation, there is no point in continuing the simulation, because of the intractability in computation and branching.

Time	**$RECR_i$**	**$COMP_{ji}$**	**dN_i**	**N_i**	**$RECR_j$**	**$COMP_{ij}$**	**dN_j**	**N_j**
t_0	<0, std>	<(-∞, 0), std>	<0, std>	<K_i, std>	<(0, RMAX R_j), inc>	<(-∞, 0), dec>	<(0, +∞), inc>	<(0, RMAX N_j), inc>
(t_0, t_1)	<0, std>	<(-∞, 0), std>	<0, std>	<K_i, std>	<(0, RMAX R_j), inc>	<(-∞, 0), dec>	<(0, +∞), inc>	<(0, RMAX N_j), inc>
	<0, std>	<(-∞, 0), std>	<(-∞, 0), dec>	<(RMAX N_i, K_i), dec>	<(0, RMAX R_j), inc>	<(-∞, 0), dec>	<(0, +∞), inc>	<(0, RMAX N_j), inc>
	<0, std>	<(-∞, 0), dec>	<0, std>	<K_i, std>	<(0, RMAX R_j), inc>	<(-∞, 0), dec>	<(0, +∞), inc>	<(0, RMAX N_j), inc>
	<0, std>	<(-∞, 0), dec>	<(-∞, 0), dec>	<(RMAX N_i, K_i), dec>	<(0, RMAX R_j), inc>	<(-∞, 0), dec>	<(0, +∞), inc>	<(0, RMAX N_j), inc>

Table 11 LVG QSIM simulation results

Functionality Evaluation

Table 12 details the functionality evaluation for constraint-based QR, based on the assessment of QSIM. Where 'non-quantitative' relationships and reasoning are specified in the table this refers to qualitative support sets only. QSIM has no linguistic knowledge representation capabilities. Where mixed support sets are specified this refers to mixed qualitative-quantitative only.

Although QSIM handled the task, representing the LVG model proved problematic for a couple of reasons. First, it was not obvious how to actually transform the two simultaneous equations into QDE form because of the non-linear behaviour of different terms. In the end it is still not entirely clear if the QDE representation used was the 'best for the job'. Second, QSIM provides no means of representing and reasoning with order of magnitude relations such as hold between the numeric LVG's parameters (e.g. $r_i > r_j$). The use of such relationships may have reduced the ambiguity of the qualitative simulation by providing a means of representing the fact that one species might be a better competitor than the other. Instead, QSIM produces all possible states consistent with a QDE e.g. all states corresponding to one species outcompeting the other, all states corresponding to the other species outcompeting and so on until the outcome of all possible orders of magnitude relations holding between the LVG parameters have been produced.

Functionality	Rating
Structural Representation	
Community (vegetation) types	1
Community attributes	1
Community properties	2
Species attributes	1
Species properties	2
Inter-specific interactions	3
Intra-specific interactions	3
Non-disturbance environmental influences	2
Disturbance influences	1
Vegetation-environment feedback and dynamics	2
Value Representation	
Support sets:	
Real values	1
Integer values	1
Qualitative values	4
Ordered linguistic values	1
Unordered linguistic values	1
Direction of change	4
Relationship Representation	
Quantitative relationships	2
Non-quantitative relationships	4
Mixed support set relationships	2
Reasoning	
Reasoning with quantitative relationships	1
Reasoning with non-quantitative relationships	4
Reasoning with mixed relationships	2
Time-driven simulation	1
Deterministic reasoning with state variables	1
Non-deterministic reasoning with state variables	4

Table 12 Constraint-based QR functionality

Some general issues must be raised concerning QSIM's value and relationship representation capabilities. In his formal derivation of QSIM Kuipers (1994) states that each variable in a QSIM model must be, amongst other things, *continuous* and *continuously differentiable.* Ecological knowledge is not always concerned with variables that are reasonable functions of time either because linguistic or a mixture of support set types are used or because continuously valued variables are related to other variables in ways that mean they are no longer continuously differentiable. This can arise through sharp non-linearities in numeric functions, the existence of distinct operating regions (discontinuities and basins of attraction) or through there only being knowledge available relating continuous variable values to discrete variable values. In the latter case non-differentiable jumps in quantitative variable value may need to be introduced to capture the relationships between continuous and discrete variables e.g. if x_t = low then $y_{t+1} = y_t + 20$.

QSIM can handle discontinuities in the functions that govern quantity behaviour by representing each continuously differentiable part of a function as a separate QDE, then transitioning between QDEs to represent discontinuities. However using QDE transitions to represent discontinuities is not an elegant or easy to use solution for modelling systems where there are multiple quantities, all capable of changing discontinuously and all with different behavioural modes. For example, take a hypothetical QSIM model composed of 3 species each represented by a separate biomass quantity each of which has 3 behavioural modes (locally extinct, growing & stabilising). This model would require a separate QDE to cover each combination of behavioural mode for all 3 species = 3x3x3 = 27 separate QDEs with transitions defined between them; rather cumbersome.

QSIM cannot handle discontinuities whereby the level of a quantity suddenly/instantly changes leaving the same function governing its dynamics. Such discontinuities may occur through disturbance like grazing or fire (sudden removal of tissue/biomass). Both of these processes can be considered to be continuous, but only on temporal scales that are extremely fine compared to that of normal biomass growth dynamics. To smoothly reason about fire affecting biomass would require a time-step of seconds, minutes or hours, not the normal weeks or months. This discrepancy makes a case for considering such behaviour as discontinuous and outwith QSIM's scope.

In addition, the types of relationships that QSIM can represent are limited and may be insufficient to characterise complex, non-linear relationships as may be found in ecology. From Kuipers (1994) it would appear that QSIM is not capable of representing or reasoning with variables that are increasing functions of one or more variables and decreasing functions of one or more other variables simultaneously. Where QSIM can specify non-linear functions (e.g. saturating response curves) the QSIM specification relies on the use of as many value correspondences as possible to reduce ambiguity. Without such specification non-linear function constraints are likely to be a source of ambiguity and excessive branching in simulations.

QSIM provides a solution to the *direction of change problem* by representing and reasoning with qualitative derivatives to determine how discretely valued variables are changing.

However, QSIM does not offer a solution to the *rate of change problem.* QSIM represents time using a relative scale and only represents and reasons with direction of change information, not rate of change information. This means that it is not possible to determine when variables will change value with respect to each other along an absolute time-scale. Instead QSIM employs envisionment to determine all possible changes in value for all variables. Knowing that a particular variable will change value before another variable could substantially reduce the number of possible successor states. The lack of rate of change information causes QSIM to suffer from an additional problem – the *branching problem.* This is a result of representing and reasoning with indeterminate information based on weak relations.

Indeed, the hand-simulation of the QDE LVG model showed that QSIM rapidly became excessively ambiguous in its predictions. Although this ambiguity may be partially accounted for by a lack of value correspondences and by the information poor quantity spaces used, it is unclear how these quantity spaces could be refined much further except by dynamic landmark creation, a QSIM feature that would exacerbate branching in any case. The populations of species have few intuitively obvious or definable landmark values, unlike physical systems, where system behaviour is more likely to be qualitatively understood and landmark values already identified. It is likely that with an ecological model of a realistic size ambiguity would become even more of a problem using QSIM. This problem makes QSIM's usefulness for predictive modelling questionable. Indeed, the use of a non-absolute time representation during simulation means that QSIM cannot address the types of discrete time-centred questions that this thesis is interested in answering.

2.3.3.4 Process-Based QR

2.3.3.4.1 Review

Representation

Process-based QR is typified by Qualitative Process Theory or QPT (Forbus 1984). As such the approach will be evaluated using QPT. A few other techniques have been derived from QPT, such as GARP (Bredeweg 1992) and Qualitative Representation for Plants (Hunt and Cooke 1994) in order to address particular problems or domains in a more tailored manner.

QPT enforces a stricter ontology than QSIM for organising and reasoning with knowledge – that of the process. This means that the technique is capable of more than just qualitative simulation and can be used for tasks such as explanation, causal reasoning and intelligent tutoring. The basic principle behind QPT's ontology is the *sole mechanism assumption*, which states that all physical changes are the result, either directly or indirectly, of the action of processes.

QPT models physical systems as collections of objects of different types, each of which are characterised by sets of properties; discretely valued quantities representing continuous functions of time. Property values are represented as having two components, an amount (A) and a derivative (D), both of which have separate magnitudes ($_m$) and signs ($_s$), thus; ($<A_m , A_s>$, $<D_m , D_s>$).

Magnitudes are valued using quantity spaces (see below) whilst signs are valued using elements from the set of signs ({-1, 0, +1}). Typically in QPT the value of D_m is not known but D_s is and so the qualitative state (or value) of a property is usually represented using the pair <amount, derivative>. When specifying relationships between properties the notation (M Q t) is used to indicate the value of Q at time t where Q can refer to any component of a property e.g. (M A_m[water(C_1)] t_0) refers to the amount magnitude of water in container 1 at time 0, whereas (M D_s[water(C_1)] t_0) refers to the sign of the derivative of the water in container C1 at time 0.

QPT quantity spaces (QS) consist of a set of elements each representing a qualitative value of interest (either a distinct point or an interval) and a set of orderings between elements. QS are used to define the possible values that quantity amount magnitudes can take on and to express orderings between quantity values and constants and between quantity values and other quantity values. Orderings between elements are defined using order of magnitude operators (>, < etc.). All QS contain the element 'zero' to permit amount signs to be determined and to facilitate element ordering. Two adjacent elements with no intervening elements are defined as 'neighbour points'. If all elements of a quantity space are ordered with respect to each other then the quantity space is completely specified, otherwise it is incomplete. Incomplete quantity spaces can be a cause of branching during simulation as will be described below (under *Reasoning*).

Individual properties are related to each other using qualitative proportionalities, weak relations that represent the direction of functional dependence between two variables and which are similar in essence to the monotonic function constraints of QSIM. Qualitative proportionalities are also called indirect influences (see below). *x* QP+ *y* represents an increasing monotonic function whilst x QP- y represents a decreasing monotonic function both such that $x = f(y)$. No loops in qualitative proportionality dependence may exist i.e. A QP+ B and B QP+ A are not permitted to simultaneously hold. As with QSIM function constraints, proportionalities can be supplemented with value correspondences for the purpose of providing a more precise definition. The operator QP+/- denotes an 'anonymous' function. Functions can also be custom defined. To set the value of a property to be equal to a function the following syntax is used:

> $A_m[x]$ = dummy(y) ≡ the amount magnitude x equals the value of the function 'dummy' operating on y.

Note that a function can take more than one argument (the example above only used one – y) if defined to do so.

Processes are held as being the sole agents of change in QPT. Consequently they have an important and distinguished effect on property values represented separately from proportionalities (or *indirect influences*) and termed *direct influences*. I+(Q, n) states that some property n is a direct positive influence on some property Q, I-(Q, n) represents a direct negative influence whilst I+-(Q, n) represents an unspecified influence. Direct influences only operate when specified to do so by an active process (see below) and can be viewed as causing changes in independent variables. Indirect influences are used to relate non-process affected 'dependent' variables and to propagate the effect of processes along networks of such relationships. No property can be simultaneously affected by a direct and an indirect influence. Figure 5 illustrates further using an example from (Rajagopalan 1986). Some computational details will be covered but this is necessary to fully explain the difference between the two types of influence.

The diagram on the left of Figure 8 represents two QPT model variables – amount of water (in a container) and flow rate (out of the container). Amount of water is directly negatively influenced by the value of flow rate (the higher the flow rate, the less water there will be) whilst flow rate itself is an increasing function of the amount of water (the more water there is the higher the flow rate will be), an "indirect influence" in QPT terms. Using System Dynamics graphical notation the right hand diagram in Figure 5 shows how this situation could be represented using a compartment (rectangle), an outflow (thick arrow with 'valve') and an influence from the compartment to the outflow (thin arrow). Here the flow is acting like a negative term on the right-hand side of a differential equation i.e. d Amount Water / dt = - Flow Rate.

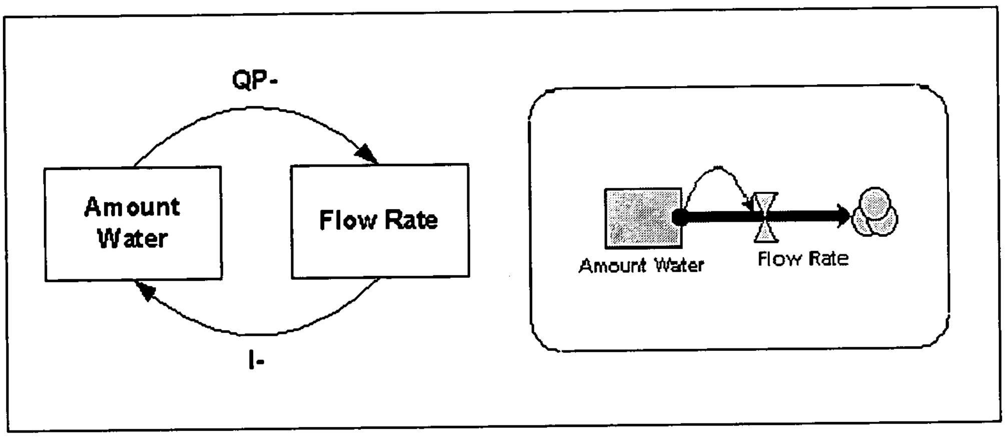


Figure 5 Representing dynamic feedback in QPT and System Dynamics (see text for explanation)

In reality Amount Water at some time (interval or point) t_1 will determine the Flow Rate at some later time t_2. This in turn will determine the Amount Water at some even later time t_3. There are various ways of modelling this chain of dependencies over time and the dynamics of the quantities involved. In System Dynamics simulation time is represented using discrete points with a fixed length of time between. Variable behaviour is only reasoned about at each discrete time-point – behaviour during intervals is not explicitly handled. At any given time-step all variable values are determined using an integrative process that follows the maxim '*calculate rate, update state*'. For example, the value of Amount Water at time-point t-1 would be used to determine the value of Flow Rate at time-point t. The calculated value for Flow Rate would then be used to determine the value for Amount Water at time-point t. Here the previous time-point's state variable value is used to determine the effect of the process (the value of Flow Rate), which is then used to determine the current time-step's state variable value.

In QPT a different method is used based on reasoning with the explicit temporal ordering of dependencies between Amount Water and Flow Rate rather than calculating change only at discrete time-steps. An active process (detailed later) is required to switch on the direct influence from Flow Rate to Amount Water, potentially causing a change in its value through an integrative process comprised of two steps - influence resolution and limit analysis (full details later). Any change in the value of Amount Water may cause Flow Rate to also change value by propagating through the qualitative proportionality. The feedback loop is then closed until the next step of qualitative simulation. The value of Flow Rate at some time t_0 is used to determine how the value of Amount Water at some later time t_1 may change and in turn determine the value of Flow Rate for some even later time t_3. In this way QPT tries to closely reproduce the temporal ordering of changes in reality free from an imposed discretisation.

In QPT the world is represented as collections of objects. Each collection of objects is represented using an individual view (IV), a frame-like construct composed of four elements:

- *Individuals:*

  Names the object(s) contained within the IV and lists their properties. An IV may refer to a single object or a collection of objects of different types. The same object may appear in more than one IV.
- *Preconditions:*

  A set of statements about relationships between individuals that are always true whenever the IV is 'active' and which are external to a QPT model. Relationships expressed as preconditions refer to things which cannot be affected by processes during simulation.
- *Quantity Conditions:*

  A set of statements that must be satisfied for the IV to be active regarding inequalities between the properties of individuals referred to in (1) or regarding the status of other IVs or process views (PVs – see below). These statements can be viewed as internal conditions and refer to things which can be affected by the action of processes during simulation.
- *Relations:*

  A set of relations that hold between the properties of the individuals in the IV whenever the IV is active. These relations are typically expressed as equality / inequality statements and qualitative proportionalities between properties.

Collections of objects are subject to processes, the sole agent of change. Processes are defined using process views (PVs), a representational structure very similar to the IV. The first four elements are identical to those in the IV but, refer to the PV rather than an IV:

1. *Individuals.*
2. *Preconditions.*
3. *Quantity Conditions.*
4. *Relations.*
5. *Influences:*

A set of direct influences imposed on the properties of the individuals specified in (1) when the PV is active.

Reasoning

QPT uses envisionment to perform qualitative simulation. Attainable envisionment is achieved through the use of a program called GIZMO (Forbus 1984) whilst total envisionment is achieved through an assumption-based truth maintenance systems called the Qualitative Process Engine (QPE) (Forbus 1990). For the purposes of this evaluation, the simpler approach technique taken by GIZMO will suffice.

The basic simulation algorithm for determining possible successor states to a given starting state in QPT (as used by GIZMO):

1. Find all active process and view instances in current situation.
2. Resolve direct then indirect influences to determine D_s values – *influence resolution.*
3. Determine what value and state transitions are possible - *limit analysis.*
4. Select a transition from the list of possible transitions.
5. Compute and store the corresponding next state for the transition.
6. Repeat from step 4 until all possible transitions have been computed then,
7. Go to step 2 until finished.

QPT models are generally composed of a library of IVs and PVs. A separate View Instance (VI) is created for every collection of objects in a QPT model that satisfies the 'individuals' element of a specified IV. Similarly, for every collection of objects in a QPT model that satisfies the 'individuals' element of a specified IV, a separate Process Instance (PI) is created. VIs and PIs are said to be active whenever their quantity conditions are satisfied. The set of active VIs constitute a situation's view structure whilst the set of active PIs constitute its process structure. Both view and process structures may change during envisionment.

Given a set of statements regarding the individuals present in a physical situation and the values of their properties (amounts *and* derivatives) the process of determining the active VIs and PIs completes the 'initialisation' of a QPT model. If there are no active PIs then the situation is static.

Time in QPT models is represented in terms of contiguous intervals where instants are represented as intervals of very short duration. The time-scale is relative.

The process of reasoning about change in a QPT model is done in two parts – *influence resolution* and *limit analysis*. Throughout QPT simulation the A_m and D_s values for all properties are stored and updated as appropriate. Influence resolution is used to update the D_s values (direction of change) for properties in the currently active VIs. Limit analysis then performs an integrative step to determine changes in A_m values and in the process and view structures. Direct influences are 'resolved' before indirect influences. The influence element of each active PI specifies the properties that are being directly influenced and by what. Properties, although they may not be simultaneously indirectly and directly influenced, can be directly influenced by multiple properties at the same time. The D_s value of a directly influenced property is set equal to sum of its direct influence directions as calculated using the QPT qualitative addition operator (Table 13). This operator is based on sign-algebra and may return an ambiguous result for the operations [+1] + [-1] and [-1] + [+1] if there is insufficient information regarding the A_m values of the influencing and influenced properties. To determine whether one A_m is >, < or = to another orderings between quantity space elements can be used. If the ordering between elements is unknown the simulation will branch to represent there being multiple possible outcomes. A further constraint occurs through enforcing that all change be continuous. This means that a property's D_s value cannot jump from +1 to –1. It must go through 0.

	-1	**0**	**+1**
-1	-1	-1	N1
0	-1	0	1
+1	N1	+1	+1

Table 13 QPT qualitative addition operator for Ds[A]+Ds[B] (N1: if Am[A] > Am[B] then Ds[A], if Am[A] < Am[B] then Ds[B], if Am[A] = Am[B] then Ds = 0)

Once all the possible D_s values for all directly influenced properties have been determined, the newly posited D_s values propagate through to the properties which they indirectly influence. Indirect influences are specified in the relations element of the active VIs and PIs. Resolving indirect influences is done in the same way, by qualitatively adding the D_s values of the independent variables. Where indirectly influenced properties indirectly influence other properties changes may propagate further. Again, ambiguity may arise without sufficient A_m ordering information. Forbus (1984) states that domain-specific knowledge can reduce the ambiguity of influence resolution but does not specify how this can be formally achieved in QPT. The process of influence resolution continues until all indirectly influenced property D_s values have been determined. Those properties neither directly nor indirectly influenced have $D_s = 0$.

Once the all property D_s values have been determined the integrative process of Limit Analysis is used to determine which properties change A_m value, how they change value and if the changes in value cause the process or view structures to change. Quantity hypotheses are formed through an exhaustive generate and test process to posit possible ways in which directly influenced property A_m values may change. Various rules are applied to direct and constrain the generation of these value changes. The most basic of these refer to D_s values. For example, if $D_s < 0$ then a property's A_m value will decrease over some time interval. Assuming continuity of change means that a property's A_m value cannot skip elements in its QS. If $A_m[P] < A_m[Q]$ was true and $D_s[P] = +1$, then the only relative change that can be true next is that $A_m[P] = A_m[Q]$. This constraint preserves the continuity of change. Forbus (1984) p.192 provides two table which show how property A_m values may change with respect to each other (i.e. their ordering) based upon their D_s values. In addition QPT uses the *equality change law* which states that if two properties are at equality they will change value before two properties that are not equal due to change from equality only taking an instant whereas change towards equality may take much longer. The use of domain specific knowledge in generating and constraining value change is alluded to in Forbus (1984) but not detailed. Where there are multiple possible value changes the simulation will branch and consider each branch separately from that point on.

Those quantity hypotheses that result in a change in process or view structure are called limit hypotheses. They mark a qualitative transition in the structure and behaviour of the system being modelled through making some VIs and PIs inactive and other ones active.

Various causes of branching in QPT simulation can be identified:

1. If a property's QS ordering is incomplete then more than one neighbour point may exist making it impossible to determine which value the property may move to next.
2. If a process directly influences more than one property it may not be possible to decide which property changes value first so a branch will occur.
3. Several processes may be active at once creating ambiguity in influence resolution.

Different ways of depicting the results of QPT simulation exist. One way is to draw a parameter history in terms of episodes and events (see Forbus 1984 p.193 for an example) that details how the values of different model properties change along a relative time-scale in relation to each other. Another way is to draw a qualitative state transition graph or use a tabular format.

Recognised Problems

As with QSIM, QPT suffers from the *rate of change problem* and consequently also from the *branching problem.* The technique of envisioning, employed to get round weak and indeterminate relations between properties creates ambiguous simulations. Ambiguity can become excessive, forcing domain-specific knowledge to be employed in pruning the possible state space. These problems have been noted (D'Ambrosio 1987, Salles 1997).

Ecological Modelling Evaluation

Representation

The LVG model was represented using 2 individual views and 2 process views. Figure 6 depicts the causal relations that hold between the properties of interest under 2 of the possible view structures, both with the same process structure.

Two individual views were specified (see Figure 7) – one for a growing population and one for a stabilising population. In both individual views the populations are represented identically in terms of properties – a population size (N_n), growth rate (GR_n) and competitive effect on the other species ($COMP_{mn}$). Basically, the growing population view applies when $N_n < K_n$ i.e. a species still has the capacity to increase. This is reflected in the QP+ relation that holds between GR_n and N_n. The stabilising view applies when $N_n \geq K_n$. Under the stabilising view GR_n is related to N_n by QP- to reflect mortality due to intraspecific competition outweighing any recruitment. There was no direct way of representing differences in specific reproductive rate (r_n) or competitive effect ($\propto_{mn}$) using the QPT notation. However, using inequality statements related to QS elements it was possible to define $K_n > K_m$ (see below). For simulation $K_i > K_j$ was used.

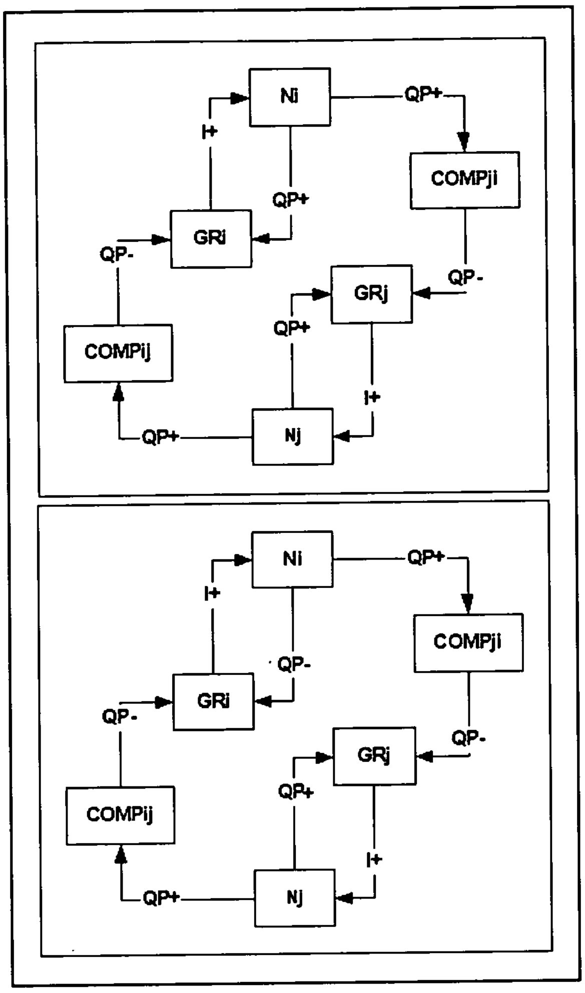


Figure 6 QPT representation of the LVG model (top diagram represents 2 active Growing Population IVs with an active Two Species Population Growth PV, bottom diagram represents 1 active Growing Population IV and 1 active Stabilising Population IV with an active Two Species Population Growth PV)

Growing Population Individual View	**Stablising Population Individual View**
Individuals	**Individuals**
Object population species $_n$	Object population species $_n$
population species $_n$ has property (N_n)	population species $_n$ has property (N_n)
population species $_n$ has property (GR_n)	population species $_n$ has property (GR_n)
population species $_n$ has property ($COMP_{mn}$)	population species $_n$ has property ($COMP_{mn}$)
Preconditions	**Preconditions**
-	-
Quantity Conditions	**Quantity Conditions**
$A_m[N_n] > 0$	$A_m[N_n] \geq K_n$
$A_m[N_n] < K_n$	**Relations**
Relations	GR_n QP- N_n
GR_n QP+ N_n	

Figure 7 QPT LVG individual view definitions

Two process views (see Figure 8) were used to represent the LVG model. QPT permits the objects to exist or not to exist via making VIs active or inactive. Due to the fact that it is possible for one species to become extinct in the LVG model ($N_n = 0$) whilst the other is still present a monoculture population growth PV was defined whereby only one species is present ($A_m[N_n] > 0$). If both species reach $N_n = 0$ then simulation will terminate as there are no species present. With both species populations above zero the two species population growth PV holds. Under this process the $COMP_{mn}$ variables are both indirectly influenced and a direct influence is asserted to hold between GR_n and N_n. This direct influence also holds in the monoculture PV.

Monoculture Population Growth Process View	**Two Species Population Growth Process View**
Individuals	**Individuals**
Object population species $_n$	Object population species $_i$
population species $_n$ has property (N_n)	population species $_i$ has property (N_i)
population species $_n$ has property (GR_n)	population species $_i$ has property (GR_i)
population species $_n$ has property ($COMP_{mn}$)	population species $_i$ has property ($COMP_{ji}$)
Preconditions	Object population species $_j$
Active Growing Population VI OR	population species $_j$ has property (N_j)
Active Stabilising Population VI	population species $_j$ has property (GR_j)
Quantity Conditions	population species $_j$ has property ($COMP_{ij}$)
$A_m[N_n] > 0$	**Preconditions**
Relations	2 active Growing Population VIs OR
-	2 active Stabilising Population VIs OR
Influences	1 active Growing + 1 active Stabilising Population VI
I+ (N_n GR_n)	**Quantity Conditions**
	$A_m[N_i] > 0$
	$A_m[N_j] > 0$
	Relations
	$COMP_{ji}$ QP+ N_i
	$COMP_{ij}$ QP+ N_j
	Influences
	I+ (N_i GR_i)
	I+ (N_j GR_j)

Figure 8 QPT LVG model process view definitions

View and process structure labellings:

V1 – 2 active growing population VIs.

V2 – 1 active growing population VI and 1 active stabilising population VI.

V3 – 2 stabilising population VIs.

P1 – active monoculture population growth PI.

P2 – active two species population growth PI.

The quantity spaces and orderings used for each variable:

N_n - $\{0 < K_j < K_i < +\infty\}$

GR_n - $\{-\infty < 0 < +\infty\}$

$COMP_{mn}$ - $\{0 < +\infty\}$

The value correspondences used to further specify direct and indirect influences:
correspondence $((A_m[N_i], 0), (A_m[COMP_{ji}], 0))$.
correspondence $((A_m[N_j], 0), (A_m[COMP_{ij}], 0))$.
This states that when the population sizes are equal to 0 then corresponding competitive effect properties are also equal to zero.

Note that no correspondence is used to relate GR_n to N_n when $N_n = 0$ or to K_n. Under these circumstances either the view or process structure will change, making the appropriate changes. For example if $N_n = 0$ then the species will cease to exist, altogether removing GR_n. There is no need to explicitly set the property A_m to 0.

Qualitative Simulation

Using the notation $[x] = \langle A_m[x], D_s[x]\rangle$ (where x = a property) to represent the values of properties the QPT LVG simulation was started with view structure V1 and process structure P2.

The simulation was started with both species $N_n = \langle(0, K_j), +1\rangle$, both species $COMP_{mn} = \langle(0, +\infty), +1\rangle$ and both species $GR_n = \langle(0, +\infty), +1\rangle$. Under P2 both populations are directly influenced by GR_n. Influence resolution results in only one possibility – that $D_s[N_n]$ for both populations = [+1]. Propagating this on through the qualitative proportionalities yields $D_s[COMP_{mn}] = [+1]$ if the (unknown) $D_m[N_n] > D_m[COMP_{nm}]$, $D_s[COMP_{mn}] = [0]$ if $D_m[N_n] < D_m[COMP_{nm}]$ and $D_s[GR_n] = [+1]$. At limit analysis (see review) the properties may change thus:

$COMP_{ji}\ t_0 \Rightarrow (\langle(0, +\infty), +1\rangle \vee \langle(0, +\infty), 0\rangle)\ t_1$

$GR_i\ t_0 \Rightarrow (\langle(0, +\infty), +1\rangle)\ t_1$

$N_i\ t_0 \Rightarrow (\langle(0, K_j), +1\rangle \vee \langle K_j, +1\rangle)\ t_1$

$COMP_{ij}\ t_0 \Rightarrow (\langle(0, +\infty), +1\rangle \vee \langle(0, +\infty), 0\rangle)\ t_1$

$GR_j\ t_0 \Rightarrow (\langle(0, +\infty), +1\rangle)\ t_1$

$N_j\ t_0 \Rightarrow (\langle(0, K_j), +1\rangle \vee \langle K_j, +1\rangle)\ t_1$

Combined this generates a total of 16 possible quantity hypotheses of which 8 are limit hypotheses, under which the view structure would change to V2 whilst maintaining the process structure as P2. Table 14 details the starting state and some of the possible state changes that may occur at time t_1. The degree of branching which had already began despite there only having been a single update 'step' was too large for hand-simulation. The run was therefore stopped.

Time	Structure	$COMP_{ji}$	GR_i	N_i	$COMP_{ij}$	GR_j	N_j
t_0	V1 P2	<(0, +∞), +1>	<(0, +∞), +1>	<(0, K_j), +1>	<(0, +∞), +1>	<(0, +∞), +1>	<(0,K_j), +1>
t_1	V1 P2	<(0, +∞), +1>	<(0, +∞), +1>	<(0, K_j), +1>	<(0, +∞), +1>	<(0, +∞), +1>	<(0,K_j), +1>
	V1 P2	<(0, +∞), +1>	<(0, +∞), +1>	<(0,K_j), +1>	<(0, +∞), +1>	<(0, +∞), +1>	<(0,K_j), +1>
	V2 P2	<(0, +∞), +1>	<(0, +∞), +1>	<(0, K_j), +1>	<(0, +∞), +1>	<(0, +∞), +1>	<K_j, +1>
	V2 P2	<(0, +∞), +1>	<(0, +∞), +1>	<(0,K_j), +1>	<(0, +∞), +1>	<(0, +∞), +1>	<K_j, +1>

Table 14 QPT LVG model hand simulation model results

Functionality Evaluation

The functionality evaluation for process-based QR based on the characteristics of QPT is given in Table 15. For the structural criteria the work of Salles (1997) and Salles and Bredeweg (1997) in using QPT for modelling vegetation dynamics was taken as demonstrating QPT's capabilities.

QPT was able to represent the LVG model but not without some problematic issues. As with QSIM it was not obvious how to actually transform the model equations into QPT form and in the end it was not obvious whether the representation designed was the 'best for the job' in terms of demonstrating QPT's capabilities. For example, the choice could have been made to disaggregate each population change (GR_n) variable into separate growth and mortality processes. However the QPT model was designed to be as close as possible to the structure of the LVG equations so an aggregated rate of change was used.

QPT has better capabilities for representing equality and inequality relations between variable values and can use the same quantity space to value different variables as QPT quantity spaces are simply distinguished values (intervals or points) with ordering relations. Distinguished values for different variables can be represented and ordered in the same QS as was done for N_n. In QSIM each variable has a different QS and correspondences can only be used as subsidiary information within constraints. However this did not help in representing differences between the specific reproductive rate and specific competitive effect of each species. For example, it is possible for a species with a higher specific reproductive rate to have a lower absolute reproductive rate than a species with a lower specific reproductive rate. It was not clear how to represent this using QPT.

Functionality	Rating
Structural Representation	
Community (vegetation) types	1
Community attributes	1
Community properties	2
Species attributes	1
Species properties	3
Inter-specific interactions	3
Intra-specific interactions	3
Non-disturbance environmental influences	3
Disturbance influences	3
Vegetation-environment feedback and dynamics	2
Value Representation	
Support sets:	
Real values	1
Integer values	1
Qualitative values	4
Ordered linguistic values	1
Unordered linguistic values	1
Direction of change	4
Relationship Representation	
Quantitative relationships	1
Non-quantitative relationships	4
Mixed support set relationships	1
Reasoning	
Reasoning with quantitative relationships	1
Reasoning with non-quantitative relationships	4
Reasoning with mixed relationships	1
Time-driven simulation	1
Deterministic reasoning with state variables	1
Non-deterministic reasoning with state variables	4

Table 15 Process-Based QR Functionality

Rather than trying to specify non-linear relationships (e.g. 'concave up') between variables using value correspondences, QPT permits the structure and relationships in a model to entirely change. This means that non-linearities and discontinuities between variables can be represented using different IVs and PVs. In the LVG model this can be seen in the relations element of the two IVs used. The changing effect of population size on growth rate was represented using opposite QP relations in two different IVs.

However Forbus (1984) offers no guidance on how to decompose and represent a physical situation in terms of processes and flows. Salles (1997) models species dynamics using a QPT-based formalism with some success. The task tackled in his 'LifeCycle III' and LifeCycle IV' models was not one of model transformation but one of designing a single species population model. The choice was made to represent the world using a plant population IV, an environmental conditions IV and a population growth PV in LifeCycle III.

However he did not represent or reason about the non-linear parabolic shape of the relationship between population size and growth rate – growth rate remained QP+ to population size throughout.

In LifeCycle IV a different view and process structure was employed to address the non-monotonic nature of some of the relationships involved. To represent the effects of intraspecific competition on establishment Salles (1997) used two direct influences from a property called 'establishment rate' which was QP+ to cover. Establishment rate directly negatively influenced the number of germinating seeds and also positively influenced the number of seedlings. Containing such opposing influences, the model was ambiguous under simulation. LifeCycle IV represented the effects of intraspecific competition in controlling establishment but did not clearly represent the reproductive rate increasing up to a certain point beyond which intraspecific competition-caused mortality causes reproductive rate to fall until eventually no reproduction occurs at all.

So there are different ways in which to represent individuals and processes in QPT but it is not obvious which are the best under which circumstances and for what problems and which contain the most ambiguity and result in the most branching. The use of dynamic IV and PV structures enhances the possibilities for representing changing structure and relationships, particularly non-linear changes. However the basic relationship constructs used by QPT are still information weak and inherently monotonic. They may not be able to adequately represent ecological phenomena.

QPT solves the *direction of change problem* but not the *rate of change problem* and in doing so, just like QSIM, ends up suffering from the *branching problem.* The use of a relative time-scale, information weak relations and value representations and the branching nature of the envisionment simulation process produce excessive ambiguity in QPT simulations. It is likely that this ambiguity could be reduced by increasing the resolution of the QSs used, by further specifying QS value orderings and by further specifying inequalities, equalities and correspondences between variable values. In addition, the *branching* problem is partly a problem of influence resolution – where there are opposing influences the simulation is likely to branch without subsidiary information. Salles (1997) proposes a method based on D'Ambrosio (1987) for solving this by annotating each influence with a strength measurement valued using a 5 element QS {very weak, weak, zero, strong, very strong}. Opposing influence strength measurements can then be added together using the SIMAO qualitative algebra (see next section for details) to produce an overall direction to the influence e.g. I+ 'very strong' + I- 'weak' = I+ 'strong'. The evaluation of SIMAO presented in the next section will show that, despite this being an attractive idea for generically solving the problems of influence resolution, the SIMAO algebra is unsound for this type of operation, particularly if more than two opposing influence strengths are being added.

The excessive ambiguity in the hand-simulation shows that QPT is likely to be of little use in answering questions regarding absolute time-based dynamics. In addition, as with QSIM, although the LVG model suited the stipulation that all model variables be continuous functions of time, available ecological knowledge is value heterogeneous, presenting a problem for wider application of the technique.

2.3.3.5 Symbolic Qualitative Algebra – SIMAO

Review

SIMAO (System for the Interpretation of Measurements, Analyses and Observations) (Guerrin 1991, Guerrin 1992) is a system for processing quantitative, qualitative and linguistic values that has been developed within the field of ecology. It is designed, amongst other things, to allow the calculation of variable values using the fixed qualitative support set of very low, low, medium, high, very high represented by the ordered set of symbols {pp, p, m, f, ff} where pp < p < m < f < ff. Guerrin terms this set of values the SIMAO quantity space (QS) and defines each element as representing an adjacent crisp interval. Under this definition the element p for example represents a range of values termed 'low' whilst the element m represents a range of values termed 'medium' such that there are no other possible values between p and m.

SIMAO attempts to integrate the use of heterogenous values acquired from measurements, analyses or observations of a system to be represented and used as quantity values. Different value types are mapped by appropriate sets of 'transfer rules' onto their equivalent qualitative SIMAO QS value. Through the use of the various operators that constitute the SIMAO qualitative algebra these qualitative values can then be manipulated and combined in algebraic equations to predict the values of other quantities. It should be noted that Guerrin (1991) and Guerrin (1992) only apply the SIMAO algebra to deducing the value of quantities given other quantity values once e.g. given that water temperature is low (p), pH is standard (m) and wind is high (f), what will the oxygen level be? They do not enter into dynamic simulation. Their deduced values are intended to represent short-term prediction of system state (1-2 days into the future). Salles *et al.* (1996) and Salles (1997) use the SIMAO qualitative algebra as the mathematical formalism in a system to predict changes vegetation but again only to predict change in state over a single time-step.

Ecological Modelling Evaluation

Introduction

Although only used for single time-step inferences by Guerrin (1991), Guerrin (1992) and Salles (1997) it is interesting to ask the question – can the SIMAO qualitative algebra provide a general means of calculating over some longer period of time with heterogeneous value and relationship knowledge? Salles (1997) concludes that the system has strong potential utility for capturing the mathematical structure of qualitative relationships. To investigate this question the LVG model was represented using SIMAO operators in place of algebraic operators then simulated using a simple time-driven loop. The simulation was performed for 20 time-steps under different conditions and compared with quantitative LVG results after they had been mapped onto the SIMAO QS.

The evaluation exercise was designed to determine:

1. For a particular model simulated for a given number of steps, *t*, using equivalent operators (e.g. instead of arithmetic addition +, using the SIMAO qualitative addition operator Q-add given by [+]), how well do SIMAO results for the main state variables match numerical results mapped onto the SIMAO QS?
2. For the main state variables of concern do different model parameterisations have any effect on the closeness of match?
3. For the main state variables of concern, what effect on closeness of match does initialising the numerical simulation with state variable values in the middle of a QS element have compared to starting at an extreme end of the same QS element have e.g. given that $20 < p \leq 40$, what effect on matching does starting a quantity *x* at 30 have as opposed to starting it at 40?
4. What effect does model calculation order have on the results produced by the two systems?
5. For the main state variables of concern what effect do different numerical value $\rightarrow$ SIMAO QS mappings have on closeness of match?
6. Is SIMAO in some sense a qualitative equivalent to ordinary quantitative arithmetic?
7. Can SIMAO be recommended for use as part of a predictive modelling system operating with non-quantitative and mixed value and relationship knowledge?

LVG Representation: The Real-valued Quantity Problem

For any model, SIMAO may have problems representing the values of certain quantities, certain terms and certain operations. Before testing SIMAO with the LVG model it was necessary to address some such problems; the 'real-valued quantity problem' and two 'quantity calculation' problems.

Some model quantities may have a value range that extends over the negative and positive numbers. This may create problems when trying to map the SIMAO 5 element QS onto the value ranges of such quantities. An example of a real-valued quantity in the LVG model is the rate of change of population size given, for species i, by either dN_i / dt or $r_i N_i (1 - N_i / K_i - \propto_{ij} N_j / K_i)$.

Population change in the LVG model is considered as a quantity that represents how much by and in what direction the population will change over a time step. This means that population change may have a positive, zero or negative value and correspondingly represent population growth, stability or decline. When mapping the SIMAO QS onto this value range it does not make sense to say that a negative value is simply a very low or {pp} change in population size or that zero represents a medium {m} or 'normal' change. Zero represents no change rather than medium-valued change.

Consequently it was decided that to effectively represent the LVG model using the SIMAO system there must be no negatively valued terms. This was achieved through simple algebraic re-arrangement of the equations. Taking equations 3 and 4 and expanding the brackets gives:

$$N_{i(t)} = N_{i(t-1)} + r_i N_{i(t-1)} - r_i N_{i(t-1)}^2 / K_i - r_i \propto_{ij} N_{i(t-1)} N_{j(t-1)} / K_i \qquad \text{Eq.7}$$

$$N_{j(t)} = N_{j(t-1)} + r_j N_{j(t-1)} - r_j N_{j(t-1)}^2 / K_j - r_j \propto_{ji} N_{j(t-1)} N_{i(t-1)} / K_j \qquad \text{Eq.8}$$

All quantities in equations 7 and 8 now have value ranges limited to zero and the positive numbers making them suitable for representation within SIMAO.

LVG Representation: Quantity Calculation Problems

In the original differential (eqs. 1 and 2), difference (eqs. 3 and 4) and algebraically rearranged (eqs. 7 and 8) versions of the LVG model there are terms which involve division.

SIMAO does not have a qualitative equivalent to the division operator, posing a problem. This can be solved through introducing a new term to the model, b_n, representing the value of $1 / K_n$. Rewriting equations 7 and 8 to incorporate b_n gives:

$$N_{i(t)} = N_{i(t-1)} + r_i N_{i(t-1)} - b_i r_i N_{i(t-1)}^2 - b_i r_i \propto_{ij} N_{i(t-1)} N_{j(t-1)} \qquad \text{Eq.9}$$

$$N_{j(t)} = N_{j(t-1)} + r_j N_{j(t-1)} - b_j r_j N_{j(t-1)}^2 - b_j r_j \propto_{ji} N_{j(t-1)} N_{i(t-1)} \qquad \text{Eq.10}$$

The value range for the variable K_n may be mapped onto the SIMAO QS and then the SIMAO values for K_n mapped directly through a form of inverse correspondence onto SIMAO values for b_n i.e. if $K_n = \{ff\}$ then $b_n = \{pp\}$, if $K_n = \{f\}$ then $b_n = \{f\}$, if $K_n = \{m\}$ then $b_n = \{m\}$ etc.

SIMAO provides no power operator and so to allow calculation of the terms $N_{i(t-1)}^2$ and $N_{j(t-1)}^2$ the LVG model as represented by equations 7 and 8 must be rewritten to give:

$$N_{i(t)} = N_{i(t-1)} + r_i N_{i(t-1)} - b_i r_i N_{i(t-1)} N_{i(t-1)} - b_i r_i \propto_{ij} N_{i(t-1)} N_{j(t-1)} \qquad \text{Eq.11}$$

$$N_{j(t)} = N_{j(t-1)} + r_j N_{j(t-1)} - b_j r_j N_{j(t-1)} N_{j(t-1)} - b_j r_j \propto_{ji} N_{j(t-1)} N_{i(t-1)} \qquad \text{Eq.12}$$

Despite several problems, the LVG model as given by equations 11 and 12 can be represented using the SIMAO system. These equations are difference equation equivalents to the original model (eqs. 1 and 2) and importantly do not involve real-valued quantities, division, powers or calculations explicitly involving the number 1 - all features that SIMAO cannot represent.

<u>Test Model Versions</u>

This section details the LVG model versions used for the test procedure. First the quantitative version:

$$N_{i(t)} = N_{i(t-1)} + r_i N_{i(t-1)} - r_i N_{i(t-1)}^2 / K_i - r_i \propto_{ij} N_{i(t-1)} N_{j(t-1)} / K_i \qquad \text{Eq.7}$$

$$N_{j(t)} = N_{j(t-1)} + r_j N_{j(t-1)} - r_j N_{j(t-1)}^2 / K_j - r_j \propto_{ji} N_{j(t-1)} N_{i(t-1)} / K_j \qquad \text{Eq.8}$$

Two SIMAO versions of the LVG model were tested to explore the effect of term calculation order on results. The first ordering corresponded directly to the term ordering of the numerical model given above and was termed SIMAO 1. In this ordering the calculated inflow (i.e. $+ r_n N_{n(t-1)}$) for each species will be added to $N_{n(t-1)}$ before the two calculated outflows (i.e. $- r_n N_{n(t-1)}^2 / K_n$ and $- r_n \propto_{nm} N_{n(t-1)} N_{n(t-1)} / K_m$) are subtracted from $N_{n(t-1)}$. The second ordering differed from the numerical model given above and was termed SIMAO 2. In this ordering the outflows are subtracted before the inflow is added. It is important to note that using ordinary arithmetic both orderings will produce the same results. If SIMAO is to offer a general purpose qualitative calculation method then this property should be preserved.

SIMAO 1:

$N_{i(t)} = N_{i(t-1)} + r_i N_{i(t-1)} - b_i r_i N_{i(t-1)} N_{i(t-1)} - b_i r_i \propto_{ij} N_{i(t-1)} N_{j(t-1)}$ Eq.11

$N_{j(t)} = N_{j(t-1)} + r_j N_{j(t-1)} - b_j r_j N_{j(t-1)} N_{j(t-1)} - b_j r_j \propto_{ji} N_{j(t-1)} N_{i(t-1)}$ Eq.12

Substituting the numerical operators with their SIMAO equivalents (multiplication operator kept implicit),

$N_{i(t)} = N_{i(t-1)} [+] r_i N_{i(t-1)} [-] b_i r_i N_{i(t-1)} N_{i(t-1)} [-] b_i r_i \propto_{ij} N_{i(t-1)} N_{j(t-1)}$ Eq.13

$N_{j(t)} = N_{j(t-1)} [+] r_j N_{j(t-1)} [-] b_j r_j N_{j(t-1)} N_{j(t-1)} [-] b_j r_j \propto_{ji} N_{j(t-1)} N_{i(t-1)}$ Eq.14

SIMAO 2:

Switching the calculation order from Eq.11 and Eq.12 (multiplication operator kept implicit),

$N_{i(t)} = N_{i(t-1)} [-] b_i r_i N_{i(t-1)} N_{i(t-1)} [-] b_i r_i \propto_{ij} N_{i(t-1)} N_{j(t-1)} [+] r_i N_{i(t-1)}$ Eq.15

$N_{j(t)} = N_{j(t-1)} [-] b_j r_j N_{j(t-1)} N_{j(t-1)} [-] b_j r_j \propto_{ji} N_{j(t-1)} N_{i(t-1)} [+] r_j N_{j(t-1)}$ Eq.16

Quantitative → Qualitative Value Mapping: Issues

SIMAO uses a five element quantity space or 'QS', symbolically represented by the ordered set {pp, p, m, f, ff}, as the basis for algebraic and arithmetic operations. In the context of the LVG simulation test it was necessary to map the numerical value ranges of all variables involved onto the 5 elements of the SIMAO QS:

1. To facilitate the identical initialisation and parameterisation of numerical and SIMAO versions of the LVG model.
2. To ensure that the numerical LVG results could be translated into QS values for comparison with the SIMAO LVG results.

Following Struss (1988) the set of numerical values for a variable x was termed $D_{\text{quant}(x)}$. The function QS(x)def$n$: $D_{\text{quant}(x)} \mapsto$ QS maps the possible quantitative values for each quantity onto their corresponding qualitative values on the SIMAO QS, where n is a number used to identify different QS(x)def$n$ mappings for the same variable x. For the simulation test two QS(x)def$n$ mappings will be used – QS(x)def1 and QS(x)def2. The inverse of the mapping

QS(x)def$n$ defines how to translate back from qualitative QS to numeric variable values. QS(x)def$n$ mappings provides the same function as SIMAO measurement transfer rules (called TR-m).

Given that the SIMAO algebra operates purely symbolically upon the elements of the QS the definition of the mapping QS(x)def$n$ may be crucial in determining whether the SIMAO simulation test results are 'correct' with respect to the numerical LVG results produced. However, it is not clear from Guerrin (1991) or Guerrin (1992):

1. How important the definition of QS(x)def$n$ is to the 'correctness' of SIMAO qualitative algebra *or*,
2. How to define such $D_{quant(x)} \mapsto$ QS mappings. Guerrin (1991), with regards the mappings used for a hydroecology example, states simply that they were 'expert defined'.

It is also unclear from Guerrin (1991) and Guerrin (1992) whether the definition of a numerically valued quantity's QS(x)def$n$ mapping should refer to mapping that quantity's numerical value range onto a relative scale or an absolute scale. To explain, an example; species i is known to be capable of per capita reproduction (r_i) within the range (0.0 – 0.5) whereas species j is known to be capable of per capita reproduction (r_j) within the range (0.0 – 1.0) and that no species has a per capita reproduction rate > 1.0. Using the SIMAO QS should QS(r_i)def$n$ (the quantitative $\rightarrow$ qualitative value mapping for r_i) be defined with respect to:

1. The quantity's relative value range so that $\{(pp \leq 0.1), (0.1 < p \leq 0.2), (0.2 < m \leq 0.3), (0.3 < f \leq 0.4), (0.4 < ff \leq 0.5)\}$?
2. The quantity's absolute range across both species (in this example the range (0.0 – 1.0)) so that perhaps $\{(pp \leq 0.2), (0.2 < p \leq 0.4), (0.4 < m \leq 0.6), (0.6 < f \leq 0.8), (0.8 < ff \leq 1.0)\}$?
3. Or either, it doesn't matter to the results?

It was decided to test the results produced by the SIMAO version of the LVG model against two different mappings for the state variables (or quantities) of interest – N_i and N_j. Single mappings were used for each of the remaining model variables.

Quantitative $\rightarrow$ Qualitative Value Mapping: Definition Procedure

Each version of the LVG model being tested involves 10 different variables using a mixture of relative and absolute $D_{quant(x)} \rightarrow$ QS mappings (see Table 16 for details). It was decided to map the state variables N_i and N_j relatively. This meant that, for each variable, the mapping from $D_{quant(x)} \rightarrow QS$ was defined with respect to the maximum and minimum values for the species concerned irrespective of the value range of the corresponding variable in the other species. The maximum numeric values for N_i and N_j are given by the carrying capacity variables, K_i and K_j. Absolute scale variables had their $D_{quant(x)} \rightarrow QS$ mapping defined with respect to the absolute maximum and minimum values from the corresponding variables in both species.

The QS(*x*)def*n* mapping definition procedures for all variables are detailed in Table 17.

$D_{quant(x)} \rightarrow$ QS Mapping Definition	LVG Model Variables
Relative	$N_{i(t)}$, $N_{j(t)}$,
Absolute	r_i, $\propto_{ji}$, K_i, b_i, r_j, $\propto_{ij}$, K_j, b_j

Table 16 Mapping definitions for LVG model variables

Variable	QS(*x*)def*n*	Mapping Definition
r_i, r_j, $\propto_{ij}$, $\propto_{ji}$	QS(x)def1 + QS(x)def2	Divide value range into 5 equal intervals
K_i, K_j	QS(x)def1	Divide value range (0 - 100) into 5 equal intervals
	QS(x)def2	Divide value range (0 - 100) into 10 equal intervals then make pp = lowest single interval, p = next lowest 2 intervals, m = the next 4 intervals, f = next 2 intervals and ff = last (highest) interval
b_i, b_j	QS(x)def1 + QS(x)def2	Map K_i and K_j then use an inverse correspondence between K_i and K_j and b_i and b_j respectively such that if K_i = f then b_i = p and if K_i = pp then b_i = ff
$N_{i(t)}$, $N_{j(t)}$	QS(x)def1	Divide the value range ($0 - K_n$) into 5 equal intervals
	QS(x)def2	Divide value range ($0 - K_n$) into 10 equal intervals then make pp = lowest single interval, p = next lowest 2 intervals, m = the next 4 intervals, f = next 2 intervals and ff = last (highest) interval

Table 17 QS(*x*)def*n* mapping definition procedures

QS(*x*)def1 was therefore defined as the QS(*x*)def*n* mapping where the numerical value ranges for N_i and N_j were mapped onto the QS through the division of their value ranges into 5 equal crisp intervals with the lowest corresponding to pp, the next lowest to p and so on. QS(*x*)def2 was defined as being the QS(*x*)def*n* mapping where the numerical value ranges for N_i and N_j were mapped onto QS through the division of the value range into 10 equal crisp intervals with the lowest interval corresponding to pp, the next two lowest intervals corresponding to p, the next 4 to m, the next 2 to f and the last (highest) interval to ff.

For the simulation test each LVG model variable was assigned an initial value from the SIMAO QS. These values were used to initialise the numeric and SIMAO versions of the model. As part of the testing process (see next section for details), for some model runs the initial values of variables in the numeric LVG were calculated as being in the middle of their assigned QS element value (e.g. if K_i = f = 60 – 80 then the numeric model r_i value = 70). For other runs they were calculated as being at the extreme high-end of their assigned QS element value (e.g. if K_i = f = 60 – 80 then the numeric model r_i value = 80). The effect of increasing the numeric value of the variables K_i and K_j (carrying capacities) for some runs was to effectively increase the maximum possible numeric values for the state variables N_i and N_j. This had implications for defining the QS(*x*)def*n* mappings. Take for instance the situation whereby the numeric LVG variable K_i was set equal to the middle of the f element interval on the absolute scale mapping {(0 < pp ≤ 20), (20 < p ≤ 40), (40 < m ≤ 60), (60 < f ≤ 80), (80 < ff ≤ 100)}. Under this situation the maximum possible value of N_i is 70 and so QS(N_i)def1 = {(0 < pp ≤ 14), (14 < p ≤ 28), (28 < m ≤ 42), (42 < f ≤ 56), (56 < ff ≤ 70)}. However, under the condition where K_i is set equal to the extreme high-end value of the f element interval, the maximum possible value of N_i is 80 and so QS(N_i)def1 = {(0 < pp ≤ 16), (16 < p ≤ 32), (32 < m ≤ 48), (48 < f ≤ 64), (64 < ff ≤ 80)}.

To accomodate the fact that different QS(*x*)def*n* mappings were required to handle varying the numeric start values of K_i and K_j , the notion of mapping 'sub-definitions' was employed. Each mapping sub-definition followed the same principles with respect to the state variables N_i and N_j (i.e. division of each value range into 5 intervals for QS(*x*)def1 etc.) but was defined with respect to the *possible* numeric value ranges of the state variables N_i and N_j as dependent upon K_i and K_j. The mapping sub-definitions were denoted QS(*x*)def*n.m* so that for example, QS(*x*)def*n*.1 denotes a mid-value for K_i and K_j whilst QS(*x*)def*n*.2 denotes an extreme high-end value for K_i and K_j. The complete set of four QS(x)def*n.m* mappings employed for the test are detailed in Appendix 1.

Simulation Test: Purpose and Procedure

To evaluate SIMAO with respect to numerical simulation it was decided to explore the effect of:

1. Changing one numerical model parameter quantity qualitatively i.e. in terms of its SIMAO QS value (to answer SIMAO evaluation point 2).
2. Varying the start value of the numerical LVG state variables within the interval represented by their QS elements (to answer evaluation point 3).

3. Changing the calculation order of the SIMAO representations (SIMAO1 vs. SIMAO2) (to answer evaluation point 4).
4. Using two different mappings (QS(*x*)def1 and QS(*x*)def2) to translate numerical results onto the 5 element SIMAO QS (to answer evaluation point 5).

To address point (1) the value of the competition coefficients $\alpha_{j\,i}$ and $\alpha_{i\,j}$ were varied. Parameterisation 1 ($\alpha_{j\,i} > \alpha_{i\,j}$) represented a situation of asymmetric competition between the species *i* and *j*. Parameterisation 2 ($\alpha_{j\,i} = \alpha_{i\,j}$) represented a situation where neither species had a competitive advantage. Under both parameterisations species *i* was given a higher per capita reproduction rate (i.e. $r_i > r_j$).

To address point (2) the numerical LVG model was simulated for each QS(*x*)def*n* mapping and under each parameterisation first of all with start values in the middle of the interval represented by the appropriate QS element and secondly with start values at the high extreme end of the same interval. Thus for each QS(*x*)def*n* mapping the numerical LVG model was simulated under four starting conditions – one for asymmetric competition with 'middle' starting values, one for asymmetric competition with 'extreme' starting values, one for equal competition with 'middle' starting values and one for equal competition with 'extreme' starting values.

To address point (3), two representations of the numerical LVG model were constructed in SIMAO – SIMAO1 (isomorphic to the numerical model) and SIMAO2 (different calculation order compared to the numerical model). Both representations (SIMAO 1, SIMAO 2) were simulated under both parameterisations - asymmetric and equal competition.

Point (4) was addressed by simulating the numerical LVG model under the two mappings QS(*x*)def1 and QS(*x*)def2 for both model parameterisations and start values.

With the exception of the competition coefficient parameters, every variable in the SIMAO and numeric models was allocated a QS element start value to be held constant for all simulations. In all simulations normal arithmetic precedence applied. Simulation run details are given in Table 18 and initialisation and parameterisation details in Table 19. The simulation test procedure was:

1. Numerical model runs:

a. Initialise and parameterise variables by mapping variable QS values onto numerical values using the inverse of the appropriate QS(*x*)def*n.m* mapping and middle / extreme condition.

b. Run model for 20 time-steps.

c. Map numerical state variable values onto QS values using appropriate QS(*x*)def*n.m* mapping.

2. SIMAO model runs:

   a. Initialise and parameterise model.

   b. Run model for 20 time-steps.

3. Collate numerical and SIMAO model results.

Simulation Number	**QS(*x*)def*n.m***	**Parameterisation (number : details)**	**Numeric Variable Values w.r.t. QS Interval**	**LVG Representation**
1	1.1	1 : $\propto_{ji} > \propto_{ij}$	Middle	Numerical
2	1.2	1 : $\propto_{ji} > \propto_{ij}$	Extreme	Numerical
3	1.1	2 : $\propto_{ji} = \propto_{ij}$	Middle	Numerical
4	1.2	2 : $\propto_{ji} = \propto_{ij}$	Extreme	Numerical
5	2.1	1 : $\propto_{ji} > \propto_{ij}$	Middle	Numerical
6	2.2	1 : $\propto_{ji} > \propto_{ij}$	Extreme	Numerical
7	2.1	2 : $\propto_{ji} = \propto_{ij}$	Middle	Numerical
8	2.2	2 : $\propto_{ji} = \propto_{ij}$	Extreme	Numerical
9	n/a	1 : $\propto_{ji} > \propto_{ij}$	n/a	SIMAO 1
10	n/a	2 : $\propto_{ji} = \propto_{ij}$	n/a	SIMAO 1
11	n/a	1 : $\propto_{ji} > \propto_{ij}$	n/a	SIMAO 2
12	n/a	2 : $\propto_{ji} = \propto_{ij}$	n/a	SIMAO 2

Table 18 LVG model simulation test: run details

Variable	**Simulation Numbers**	**Value (expressed on QS)**
N_i	1 – 12	p
r_i	1 - 12	f
$\propto_{ji}$	1 - 4, 9,11	f
$\propto_{ji}$	5 - 8, 10, 12	m
K_i	1 - 12	f
b_i	9 - 12	p
N_j	1 - 12	f
r_j	1 - 12	p
$\propto_{ij}$	1 - 4, 9,11	p
$\propto_{ij}$	5 - 8, 10, 12	m
K_j	1 - 12	f
b_j	9 - 12	p

Table 19 LVG model simulation test: variable initialisation and parameterisation details

Simulation Test: Results Presentation

The summarised results for the simulation tests carried out are presented below in Tables 20 – 23.

Time (t)	Simulation 1 (Numeric, Middle)		Simulation 2 (Numeric, Extreme)		Simulation 9 (SIMAO 1)		Simulation 11 (SIMAO 2)	
	N_i	N_j	N_i	N_j	N_i	N_j	N_i	N_j
0	p	f	p	f	p	f	p	f
1	m	f	m	f	m	f	m	f
2	m	f	m	f	f	f	f	m
3	f	f	f	f	pp	pp	ff	p
4	f	f	f	f	p	pp	ff	pp
5	f	f	f	m	m	pp	ff	pp
6	f	f	f	m	f	pp	ff	pp
7	f	m	f	m	pp	pp	ff	pp
8	ff	m	f	m	p	pp	ff	pp
9	ff	m	f	m	m	pp	ff	pp
10	ff	m	ff	m	f	pp	ff	pp
11	ff	m	ff	m	pp	pp	ff	pp
12	ff	m	ff	m	p	pp	ff	pp
13	ff	m	ff	m	m	pp	ff	pp
14	ff	m	ff	p	f	pp	ff	pp
15	ff	m	ff	p	pp	pp	ff	pp
16	ff	m	ff	p	p	pp	ff	pp
17	ff	m	ff	p	m	pp	ff	pp
18	ff	m	ff	p	f	pp	ff	pp
19	ff	m	ff	p	pp	pp	ff	pp
20	ff	m	ff	p	p	pp	ff	pp

Table 20 Predicted qualitative values under parameterisation 1 and QS(x)defl

Time (t)	Simulation 3 (Numeric, Middle)		Simulation 4 (Numeric, Extreme)		Simulation 10 (SIMAO 1)		Simulation 12 (SIMAO 2)	
	N_i	N_j	N_i	N_j	N_i	N_j	N_i	N_j
0	p	f	p	f	p	f	p	f
1	p	f	m	f	pp	f	m	f
2	m	f	m	f	pp	f	f	f
3	m	f	m	f	pp	f	ff	f
4	m	f	m	f	pp	f	ff	m
5	m	f	m	f	pp	f	ff	p
6	m	f	m	f	pp	f	ff	pp
7	f	f	m	f	pp	f	ff	pp
8	f	f	m	f	pp	f	ff	pp
9	f	f	m	f	pp	f	ff	pp
10	f	f	m	f	pp	f	ff	pp
11	f	f	m	f	pp	f	ff	pp
12	f	f	m	f	pp	f	ff	pp
13	f	f	f	f	pp	f	ff	pp
14	f	f	f	f	pp	f	ff	pp
15	f	f	f	f	pp	f	ff	pp
16	f	f	f	f	pp	f	ff	pp
17	f	f	f	f	pp	f	ff	pp
18	f	f	f	f	pp	f	ff	pp
19	f	f	f	f	pp	f	ff	pp
20	f	f	f	f	pp	f	ff	pp

Table 21 Predicted qualitative values under parameterisation 2 and QS(x)defl

Time (t)	Simulation 5 (Numeric, Middle)		Simulation 6 (Numeric, Extreme)		Simulation 9 (SIMAO 1)		Simulation 11 (SIMAO 2)	
	N_i	N_j	N_i	N_j	N_i	N_j	N_i	N_j
0	p	f	p	f	p	f	p	f
1	p	f	m	f	m	f	m	f
2	m	f	m	f	f	f	f	m
3	m	f	m	f	pp	pp	ff	p
4	m	f	m	m	p	pp	ff	pp
5	m	f	m	m	m	pp	ff	pp
6	f	m	f	m	f	pp	ff	pp
7	f	m	f	m	pp	pp	ff	pp
8	f	m	f	m	p	pp	ff	pp
9	f	m	f	m	m	pp	ff	pp
10	f	m	f	m	f	pp	ff	pp
11	f	m	f	m	pp	pp	ff	pp
12	f	m	f	m	p	pp	ff	pp
13	f	m	f	m	m	pp	ff	pp
14	f	m	f	m	f	pp	ff	pp
15	f	m	f	m	pp	pp	ff	pp
16	f	m	f	m	p	pp	ff	pp
17	f	m	f	m	m	pp	ff	pp
18	f	m	f	m	f	pp	ff	pp
19	f	m	f	m	pp	pp	ff	pp
20	f	m	f	m	p	pp	ff	pp

Table 22 Predicted qualitative values under parameterisation 1 and QS(x)def2

Time (t)	Simulation 7 (Numeric, Middle)		Simulation 8 (Numeric, Extreme)		Simulation 10 (SIMAO 1)		Simulation 12 (SIMAO 2)	
	N_i	N_j	N_i	N_j	N_i	N_j	N_i	N_j
0	p	f	p	f	p	f	p	f
1	p	f	m	f	pp	f	m	f
2	m	f	m	f	pp	f	f	f
3	m	f	m	f	pp	f	ff	f
4	m	f	m	f	pp	f	ff	m
5	m	f	m	f	pp	f	ff	p
6	m	f	m	f	pp	f	ff	pp
7	m	f	m	f	pp	f	ff	pp
8	m	f	m	f	pp	f	ff	pp
9	m	f	m	f	pp	f	ff	pp
10	m	f	m	f	pp	f	ff	pp
11	m	f	m	m	pp	f	ff	pp
12	m	f	m	m	pp	f	ff	pp
13	m	f	m	m	pp	f	ff	pp
14	m	f	m	m	pp	f	ff	pp
15	m	f	m	m	pp	f	ff	pp
16	m	f	m	m	pp	f	ff	pp
17	m	m	m	m	pp	f	ff	pp
18	m	m	m	m	pp	f	ff	pp
19	m	m	m	m	pp	f	ff	pp
20	m	m	m	m	pp	f	ff	pp

Table 23 Predicted qualitative values for under parameterisation 2 and QS(x)def2

The LVG model simulation test results were assessed by ranking the closeness of match between numerical and SIMAO versions as either good, poor or no match. First of all, assessments were made regarding the closeness of match between numerical and SIMAO results for the two state variables N_i and N_j. The rank definitions used for assessment were:

1. Good match:
   a. The same overall qualitative behaviour (trend) for the variable produced numerically must be reproduced e.g. if the numerical model shows a decline in value from time-step (t) 1 to t 10 then the SIMAO results should also show a decline.
   b. In addition, the SIMAO model results must show a high degree of similarity to those produced numerically. More precisely, the SIMAO results must contain all the qualitative values produced numerically and at most be 3 time-steps ahead or behind the numerical results.
2. Poor match:
   a. The same overall qualitative behaviour (trend) for the variable as produced numerically must be reproduced e.g. if the numerical model shows a decline in value from t 1 to t 10 then the SIMAO results should show also show a decline.
   b. The SIMAO results need not show a high degree of similarity to those produced numerically. It is sufficient that the overall behaviour predicted is the same.
3. No match:
   a. The qualitative behaviour produced by the numerical model is not reproduced by the SIMAO model e.g. numerically there is a decline but SIMAO predicts stability.

To assess the overall closeness of match of the SIMAO model to the numerical version, the individual closeness of match ranks for the two state variables were combined to produce single overall ranks – again, either good, poor or no match. This was achieved through the following procedure - for each combination of parameterisation, QS(x)def$n$ mapping and numeric start value position (middle or extreme), the overall closeness of match was defined as being equal to the lowest individual state variable rank achieved by SIMAO. For example, if under one combination N_i was predicted with a poor match and N_j with no match by SIMAO then the overall closeness of match would be no match. The rationale behind this definition was simply that it does not matter whether SIMAO produced a good match for one variable if it produced a worse match for another – the system is only as good as the worst prediction produced. The assessed results are detailed in Tables 24 and 25.

	Parameterisation 1: $r_i > r_j$, $\propto_{ii} > \propto_{ij}$							
	QS(x)def1				QS(x)def2			
	Middle Values		Extreme Values		Middle Values		Extreme Values	
	N_i	N_j	N_i	N_j	N_i	N_j	N_i	N_j
SIMAO1	No	Poor	No	Poor	No	Poor	No	Poor
SIMAO2	Poor	Poor	Poor	Poor	Poor	Poor	Poor	Poor
	Parameterisation 2: $r_i > r_j$, $\propto_{ii} = \propto_{ij}$							
	QS(x)def1				QS(x)def2			
	Middle Values		Extreme Values		Middle Values		Extreme Values	
	N_i	N_j	N_i	N_j	N_i	N_j	N_i	N_j
SIMAO1	No	Good	No	Good	No	No	No	No
SIMAO2	Poor	No	Poor	No	Poor	Poor	Poor	Poor

Table 24 Degree of matching between results produced by standard arithmetic and those produced by the SIMAO qualitative algebra system for the LVG model under two different parameterisations and two different QS(*x*)def*n* mappings

	Parameterisation 1: $r_i > r_j$, $\propto_{ii} > \propto_{ij}$			
	QS(x)def1		QS(x)def2	
	Middle Values	Extreme Values	Middle Values	Extreme Values
SIMAO1	No	No	No	No
SIMAO2	Poor	Poor	Poor	Poor
	Parameterisation 2: $r_i > r_j$, $\propto_{ii} = \propto_{ij}$			
	QS(x)def1		QS(x)def2	
	Middle Values	Extreme Values	Middle Values	Extreme Values
SIMAO1	No	No	No	No
SIMAO2	No	No	Poor	Poor

Table 25 Overall degree of matching between results produced by standard arithmetic and those produced by the SIMAO qualitative algebra system for the LVG model under two different parameterisations and two different QS(*x*)def*n* mappings

Simulation Test: Results Interpretation

The two SIMAO versions of the LVG model matched the numerically produced results to different degrees of closeness for six out of the eight numerical simulations. SIMAO1, which was isomorphic to the numerical model, produced an overall 'no match' result for all 8 numerical simulations. Under parameterisation 1, the numerical model predicted N_i as showing a gradual rise and N_j as showing a gradual decline. For the same conditions, SIMAO1 predicted that N_i would show cyclical behaviour (p $\rightarrow$ m $\rightarrow$ f $\rightarrow$ pp $\rightarrow$ p ...) and N_j would show a sudden drop to pp ('bust' behaviour). Under parameterisation 2 the numerical model predicted N_i as showing a gradual rise and N_j as either remaining stable at f (QS(*x*)def1) or gradually declining (QS(*x*)def2). SIMAO1 predicted a sudden rise in N_i ('boom' behaviour) and stable behaviour for N_j. Under both parameterisations, SIMAO1's prediction of the behaviour of at least one of the two state variables was qualitatively different from the numerical predictions produced (irrespective of the start value position used).

SIMAO2, which was not isomorphic to the numerical model, produced 'poor match' results for 6 out of the eight numerical simulations and 'no match' for the other two. The 'poor match' results showed the same general trends to the numerically produced results but where the numerical model produced gradual rises and declines, SIMAO2 produced 'boom' and 'bust' behaviours. Where SIMAO2 produced a 'no match' result (parameterisation 2, QS(x)def1, both middle and extreme start values) it predicted a 'boom' for N_i and steady behaviour for N_j compared to the numerical prediction of a gradual rise for N_i and a gradual decline for N_j.

Neither SIMAO version of the LVG model produced a 'good match' to the numerical model under any parameterisation, start value position or QS(x)def$n$ mapping. In fact, SIMAO1 and SIMAO2 produced qualitatively different results during the LVG testing process, highlighting a serious problem with the SIMAO system – that equation calculation order may affect the correctness of the results produced. In addition, SIMAO1 consistently produced 'no match' results unlike SIMAO2 where results quality varied with the parameterisation and QS(x)def$n$ mapping used. This highlights another serious problem for the SIMAO system – that the correctness of the results produced may be affected by run conditions and/or the $D_{quant(x)} \mapsto$ QS mapping used.

Additionally it can be seen from the results that SIMAO models only appear able to predict four possible basic behaviours for a state variable – 'boom', 'bust', stable or cycle. Of these the first three were the more commonly demonstrated.

Simulation Test: Discussion

Introduction

The LVG model testing process shows that SIMAO at best provides 'poor match' results compared to those derived using numerical arithmetic and at worst provide 'no match' behaviour which may be completely erroneous. SIMAO was shown to produce different predictions depending upon calculation order, model parameterisation and QS(x)def$n$ mapping used. In short, SIMAO cannot be said to provide a sound means of calculating with qualitative values. Some possible explanations for the lack of soundness:

1. The number of elements in the QS set used to represent the possible qualitative values that LVG model variables may take may be too few or too many.
2. The QS(x)def$n$ mappings used may be inappropriate for the LVG model.
3. The operation of the SIMAO qualitative algebras i.e. operator definitions and properties.

QS: Number of Elements

Guerrin (1991) states that the five element QS suited the hydroecology problem he tackled 'naturally' but that generally the number of elements to be included in a QS should be determined by model purpose and application. Guerrin (1992) does not discuss the generality of the SIMAO QS or how to construct or use alternative QS.

The five element QS set {very low, low, medium, high, very high} appears to provide sufficient variation to capture the dynamics of the variables of interest (in accordance with the 'minimum required variation rule' for constructing QS given by Salles and Bredeweg (1997) without being so simple that interesting behaviour is missed or so complex that over-specification becomes a problem. The same qualitative value set has been used independently by Uhrmacher *et al.* (1997) to represent and model an ecological system (the Biosphere II project carbon cycle). They found that the five qualitative values used in the SIMAO QS were sufficient to accurately represent numerical knowledge and that a larger number of values merely increased computational cost with no real return in terms of increased ability to distinguish qualitative behaviours. Consequently there is no strong reason to suspect that the definition of the QS set used (i.e. the number of elements) in the LVG tests was the reason for the poor performance of SIMAO.

The Mapping Problem

As SIMAO algebraically and arithmetically manipulates variable values upon a purely symbolic basis (i.e. without reference to their underlying numerical value ranges), a key issue when using either system is how to map each measurement quantity's numerical value range onto the SIMAO QS. The correct choice of mapping may be crucial in determining the soundness of any results.

Neither mapping employed during the LVG model test produced consistently better results than the other, with results soundness seemingly more affected by run conditions. However, the closeness of match between qualitative and numerical simulation results was affected to some degree by the choice of QS(x)def$n$ mapping used e.g. SIMAO 2 overall produced better results under QS(x)def2 than QS(x)def1.

As mentioned, Guerrin (1991) states that each variable in his hydroecology example was mapped onto the SIMAO QS in an 'expert-defined' manner with each interval 'specific to each variable in a given context'. The two QS(x)def$n$ mappings used for the LVG tests were not chosen for their particular relevance to any quantities in the LVG model, rather they were chosen experimentally upon the basis that they are two reasonable mappings that could have been applicable.

Based on the apparent success of SIMAO's application to the field of hydroecology (Guerrin 1991, Guerrin 1992) where predictions were found to be consistent with expert opinion, it may be concluded that SIMAO's lack of success in simulating the LVG model was due in part to the QS(x)def$n$ mappings employed.

To blame SIMAO's lack of success upon the QS(x)def$n$ mappings used is attractive and quite easy. However, if true it means that for the sound application of SIMAO to any given problem the user faces the problem of trying to discover suitable QS(x)def$n$ mappings. It is not at all clear how to do this and not at all clear that it can actually be done. It may be the case that for every new quantity a comprehensive search for a suitable mapping or mappings may be required. This would make using the SIMAO system expensive in terms of labour.

The third factor that may be responsible for the failure of SIMAO to match numerically produced results may be the definition of the operators that constitute its qualitative algebra. For the purposes of more fully exploring the SIMAO qualitative algebra system the properties of its operators were investigated

SIMAO Operator Testing: Purpose and Procedure

A complete exploration of all operator properties was not performed, rather the operators used in the SIMAO representations of the LVG model were chosen - the Q-add or [+], the Q-sub or [-] and the Q-mul or [*] operators. Operator definitions can be found in Guerrin (1992).

The properties of these SIMAO operators were tested using simple two and three term equations. Each variable in each equation was given the same value range ($-\infty$ - $+\infty$) which was mapped onto the SIMAO QS using the mapping QS(*term*)def1 defined as:
QS(*term*)def1 = $\{(pp \leq 20), (20 < p \leq 40), (40 < m \leq 60), (60 < f \leq 80), (80 < ff)\}$.

The mapping QS(*term*)def1 was chosen for its similarity to QS(*x*)def1 used in the simulation test. Both numerical and SIMAO versions of the equations were expressed using identical QS values. Numerical variable values were taken as the mid-point value of their QS element interval. The equations were then solved for both versions and the results compared in terms of their QS values.

The following list details the properties investigated and methods used:

1. [+]: associativity tested using three term equations, commutativity tested using two and three term equations, SIMAO soundness compared to numerical results mapped onto QS values using QS(*term*)def1.
2. [-]: associativity tested using three term equations, commutativity tested using two term equations, SIMAO soundness compared to numerical results mapped onto QS values using QS(*term*)def1.
3. [*]: associativity tested using three term equations, commutativity tested using two and three term equations, SIMAO soundness compared to numerical results mapped onto QS values using QS(*term*)def1.

Associativity refers to insensitivity in a calculation to operation order. For example:

$3 + 4 + 5 = 3 + (4 + 5)$

Commutativity refers to insensitivity to term order in a calculation. For example:

$5 + 8 = 8 + 5$

Distributivity refers to the operation of an external factor on each term within a bracket. For example in the following the multiplication operator is distributive over the addition operator:

$5 \times (2 + 4) = (5 \times 2) + (5 \times 4)$

SIMAO Operator Testing: Results Presentation

Table 26 presents the summarised results for the operator tests. Appendix 1 contains full details.

From the testing procedure the SIMAO Q-add ([+]) operator appears to have the same properties (associativity and commutativity) as the numerical arithmetic addition or + operator. However the qualitative operator did not produce sound results compared to the numerical results under QS(*term*)def1.

With respect to subtraction (SIMAO's Q-sub or [-]) performed upon 3 term equations, SIMAO appears to have an associative operator. This is not a property shared by the subtraction operator found in standard numerical arithmetic. Like numerical arithmetic, SIMAO's subtraction operators are non-commutative.

Operator	**System**	**Associativity**	**Commutativity**	**Distributivity**	**Soundness**
+	Arithmetic	Yes	Yes	No	Yes
[+]	SIMAO	Yes	Yes	No	No
-	Arithmetic	No	No	No	Yes
[-]	SIMAO	Yes	No	No	No
x	Arithmetic	Yes	Yes	Over addition	Yes
[*]	SIMAO	No.	Yes w.r.t. eq's of < 3 terms	No	No

Table 26 Comparison of operator properties and soundness from ordinary arithmetic and SIMAO

Numerical mutiplication is associative, commutative and distributive over addition (e.g. 10 x (30 + 50) = (10 x 30) + (10 x 50)). The SIMAO Q-mul or [*] operator was found to be non-associative (calculation order will determine the results obtained) and only commutative with respect to equations of 2 terms. As soon as the [*] operator was applied to equations containing 3 terms, the order of the terms determined the solution obtained. Q-mul was not distributive over the appropriate addition operator and did not produce sound results.

<u>SIMAO Operator Testing: Results Interpretation</u>

The results of the tests performed upon the SIMAO qualitative algebra operators show that the system does not have the same properties as numerical arithmetic. Caution must therefore be employed when applying the system to qualitatively solving equations. The non-standard properties (w.r.t. to numerical arithmetic) of the qualitative operators may provide a stronger explanation than the $D_{quant(x)} \mapsto QS$ mapping definition problem as to why SIMAO produced numerically unsound results during the LVG simulation test.

To further explore whether SIMAO produced unsound simulation results because of the properties of its operators and not because of the QS(*x*)def*n* mappings employed, consider the following example using the [+] operator. The SIMAO [+] operator solves qualitative addition by taking the maximum of the two terms being operated on. This is only an approximate definition of addition, which understandably, as has been shown, produces unsound results. For example under SIMAO addition the expression m + f = f. This solution may be reasonable under certain QS(*x*)def*n* mappings although not those used in the LVG tests where taking the QS(*term*)def1 it follows that 50 (m) + 70 (f) = 120 (ff), not f as predicted. In addition consider the solution of f + f + f + f + ... ∞ = f. This is clearly non-sensical with respect to numerical addition. The definitions for qualitative subtraction, multiplication and division are similarly approximate.

It is reasonable therefore, based on the results of this investigation, to state that the numerically unsound results produced by SIMAO are most likely due to the operator definitions used. Indeed, Guerrin (1991) states that the SIMAO qualitative algebra has limitations caused by the definition of its operators (using Q-add as an example) and that the system's properties need to be improved to achieve general application within ecology. Guerrin (1992) also admits that the SIMAO algebra has deficiencies in operator definitions. It should be noted that the findings of the simulation and operator tests are in opposition to the findings of Salles (1997) who also investigated the ability of SIMAO to manipulate variable values for predictive purposes, finding the system to be 'a well developed qualitative algebra'.

Functionality Evaluation

Table 27 details the functionality evaluation for SIMAO. The system possesses the capability to represent available structural knowledge regarding vegetation although, with the exception of the species-based modelling of Salles (1997), these abilities go undemonstrated. Regarding value and relationship knowledge representation SIMAO has demonstrated capabilities with quantitative, qualitative and linguistic support sets. However, based upon the results of the modelling evaluation SIMAO has no demonstrated capability in reasoning accurately with relationship knowledge of any kind.

The notion of a generic system for manipulating heterogeneous value knowledge is attractive for it would solve many of the problems faced in modelling with available ecological knowledge. The modelling evaluation carried out has shown that SIMAO does not provide such a system. An interesting question to ask next is whether the failure of SIMAO within the context of the LVG test is due to the operator definitions it currently employs (which might be improved upon) or whether there is a more fundamental flaw in the symbolic qualitative algebra approach.

Functionality	Rating
Structural Representation	
Community (vegetation) types	2
Community attributes	2
Community properties	2
Species attributes	3
Species properties	3
Inter-specific interactions	2
Intra-specific interactions	3
Non-disturbance environmental influences	3
Disturbance influences	3
Vegetation-environment feedback and dynamics	1
Value Representation	
Support sets:	
Real values	3
Integer values	3
Qualitative values	3
Ordered linguistic values	3
Unordered linguistic values	3
Direction of change	1
Relationship Representation	
Quantitative relationships	3
Non-quantitative relationships	3
Mixed support set relationships	3
Reasoning	
Reasoning with quantitative relationships	2
Reasoning with non-quantitative relationships	2
Reasoning with mixed relationships	2
Time-driven simulation	2
Deterministic reasoning with state variables	3
Non-deterministic reasoning with state variables	1

Table 27 SIMAO Qualitative Algebra Functionality

SIMAO arithmetically reasons with value knowledge according to rules defined purely with respect to the symbols of the SIMAO QS. These rules do not refer at all to the properties of each variable's underlying support set. Numerical operators in arithmetic are defined with respect to known properties of the underlying support sets – the real numbers or the integers. Consequently they produce results that are sound and consistent with respect to the properties of the support sets of the variables being manipulated. The operators of SIMAO are not defined with respect to any support set. It perhaps not surprising therefore that they yield unsound and inconsistent results when compared to numerical arithmetic. It is unlikely that SIMAO's algebra would fare any better when manipulating linguistically valued variables for it less clear how to characterise the properties of such support sets. Indeed, serious doubts must be raised as to whether it is possible to develop a general system for reasoning with heterogeneous value and relationship knowledge without reference to support set properties. Certainly SIMAO has not achieved this aim.

2.4 Conclusions

Despite the range of techniques and applications addressing the use of non-quantitative and mixed quantitative non-quantitative value and relationship knowledge no 'off the shelf' solution is available. Certainly a range of tested partial solutions exist. It is worthwhile recapping the two major problems to be faced when modelling with discrete, non-quantitative value and relationship knowledge – the *direction of change problem* (how is the variable changing?) and the *rate of change problem* (how long will it take the variable to change?). For the goals of this thesis these problems must be solved with respect to an absolute time-scale and without generating new problems like the *excessive branching problem*. Constructs capable of representing knowledge for use in calculating and / or updating discrete variable values and methods for reasoning coherently and accurately with those constructs to predict vegetation dynamics over time are required. The potential utility of the partial solutions examined in this Chapter and the outline of the approach to be developed during the remainder of this thesis will be detailed next.

Chapter 3 Integrating Quantitative, Qualitative and Linguistic Knowledge for Modelling Vegetation Dynamics

3.1 The Need for a New Approach

Predictive modelling of vegetation dynamics is concerned with trying to determine answers to questions regarding if, how and when vegetation will change under different management options, environmental conditions, species mixtures, spatial arrangements etc. Such modelling may be used for scenario-based decision support in a practical context or for exploring the consequences of theories and hypotheses in virtual experimentation. For both uses the typical mode of questioning involves determining change over absolute time-intervals or at absolute time-points. Example questions:

1. Will species x become locally extinct over the next 20 years if a patch of land occupied by it is subjected to fire disturbance with a mean return interval of 5 years?
2. Will vegetation gradually increase in height and cover over the next 10 years given that the available species pool is comprised of species $s_1 \ldots s_n$ and the set of environmental conditions e are in operation?

Available knowledge regarding the structure and dynamics of vegetation is characterised by its informal, fragmented form and by its value and relationship heterogeneity. It is often imprecise, with quantities valued using qualitative or linguistic support sets, and incomplete, the form of relationships between different quantities typically only partially understood. A means of integrating and utilising these support set types along with quantitative knowledge (where it is available and useful) is required.

Although a range of techniques and applications have been developed to use non-quantitative and mixed quantitative - non-quantitative value and relationship knowledge, no 'off the shelf' solution is available suitable for predictive modelling of vegetation dynamics. Those ecological modelling techniques capable of representing and reasoning with available value and relationship knowledge lack a general formulation (use hard-wired variables), cannot adequately reason about direction and rate of change separately (with implications for model behaviour), and contain various deficiencies regarding their abilities to model the effects of environmental change on vegetation and the effects of feedback between vegetation and environment.

Knowledge-based systems and knowledge-based modelling techniques have demonstrated that 'if ... then' rules and inference procedures such as forward chaining can be used effectively for representing and reasoning with quantitative and non-quantitative ecological knowledge. However they do not adequately address reasoning about change in discretely valued variables over time. Some authors have proposed techniques involving separation of knowledge and reasoning concerning direction of change, rate of change and variable state (value). These ideas offer promise but have not been adequately developed so far.

QR techniques utilise the idea of separately representing and reasoning with qualitative direction of change and variable value (level in QSIM, amount magnitude in QPT) for modelling physical systems. However, they use information-weak, mostly monotonic, functions supplemented with correspondences between values to specify relationships between variables. Simulation is then performed using the incomplete relationship specified by the function plus its value correspondences to generate possible consistent states from a given starting state (attainable envisionment) or all possible system states (total envisionment) over a relative time-scale. As seen this tends to lead to ambiguity and the *excessive branching problem.* This problem is likely to be exacerbated in modelling standard ecological problems, which tend to be more complex than the LVG competition model.

In addition, QR techniques can only utilise qualitatively valued variables that conform to the 'reasonable function of time' criteria. This means that they cannot be extended to include linguistic values, which, as they change may cause discontinuities in value to arise, making QR unsuitable for modelling with heterogeneous values and relationships.

Last, purely symbolic algebras like SIMAO are unlikely to be of use due to their inherent unsoundness.

Two main obstacles stand in the way of predictive modelling based on available knowledge:

1. The *direction of change problem.*
2. The *rate of change problem.*

The following sections will sketch an outline of the approach to be developed during the remainder of this thesis to address these concerns.

3.2 Approach Outline

3.2.1 Variable Values

As discussed in Chapter 1 the notion of the support set will be used to specify legal values for model variables. Four different types of support set can be distinguished with respect to their properties and meaning (see Table 28). The values within support sets shall simply be called elements.

Support Set Type	**Underlying Continuous Scale?**	**Ordered / Unordered?**	**Corresponds to**	**Examples**
Quantitative	Yes	Ordered	Ratio & interval data	Real numbers
Qualitative	Yes	Ordered	Ordinal data	{low, medium, high}
Ordered linguistic	No	Ordered	Ordinal data	{grass, shrub, tree}
Unordered linguistic	No	Unordered	Nominal data	{shrub, grass, tree}

Table 28 Support set types

For qualitative support sets the elements used will represent either intervals or point-values. All values will be assumed to be contiguous with no intervening elements. This is unlike QSIM where the intervals between landmarks were used but not enumerated in quantity spaces. All non-quantitative support sets shall be constructed according to the principle of *minimum required variation.* This states that the number of elements used should be kept to the minimum required to adequately describe value variation for the modelling purpose at hand. The principle necessarily entails that only those elements with important behavioural meaning should be distinguished.

The approaches to modelling to be developed in Chapters 4, 5 and 6 will use these four support set types. Differences and similarities in representing and reasoning with relationships to determine variables valued using different support sets will be explored and discussed as the techniques are developed.

It should be noted that, despite their underlying continuous scale, qualitative values cannot be added or subtracted as with interval data without problems. For example, if $10 <$ medium ≤ 20 and $0 \leq$ low ≤ 10 then medium – low = {0, 20}. This indeterminacy cannot be avoided without further information. If the elements of the qualitative support set are known but not their mapping onto a numerical scale then it is impossible to say what the result of a calculation like 'medium minus low' would be. All that can be said, using arithmetic subtraction is that the result is either medium or low. For this reason qualitatively valued variables will not be subjected to arithmetic operations and are considered to be more like ordinal than interval data.

3.2.2 Relationships based on Factual Knowledge or Weak Functions?

It may be reasonable and profitable to posit simple relationships such as monotonic increasing or decreasing as holding between quantities in physical, engineering-based systems. The important quantities, how they are related and the way in which they interact to produce overall behaviour is restricted, often simple and typically already known, albeit imprecisely. Behaviourally important 'landmark' values are often easily identified for quantities (e.g. boiling temperature of a liquid, the maximum capacity of a tank etc.) and typically constant over time, and the size of such systems (the number of interacting quantities) tends to be relatively small. In addition, the aim of QR modelling of physical systems is to reproduce the results of quantitative models with less detailed information and computational effort and / or to simulate the types of commonse-sense reasoning employed by human experts whilst providing some form of explanatory capability (Leitch *et al.* 1990).

With plant communities, like other ecological systems, it is typically less clear how to bound and define the 'system', how to decompose the 'system' into components, which quantities to use to represent the chosen components, how those quantities are related to each other and what behaviour will be produced when they interact. In addition, these decisions may vary dramatically depending on the questions being asked and phenomena being addressed. Vegetation ecology is an immature field with an incomplete understanding of process and pattern. The main aims of the science are to determine which quantities are relevant, how they are related, what will happen over time under different conditions and to use this understanding in formulating policy. Following the ideas of Adaptive Management (AM) (Walters 1986, Lee 1993), even management actions in ecology should be viewed as experimental and designed to test and improve current understanding. They should not be viewed as control activities that manipulate well-understood objects in precise ways.

To posit weak relationships between ecological quantities then use a simulation technique such as envisionment seems unproductive for advancing the state of ecological knowledge or for producing management information. Given that perhaps only a few value correspondences may be known and that quantity spaces for ecological quantities are less easily definable in terms of landmark values than those of physical systems, there will almost inevitably be an excessive number of consistent states. This is likely to be true even if quantity spaces are more precisely defined as occurred in the process-based QR simulations of Salles (1997) where species' population sizes were valued using the QS {zero, low, medium, high, maximum}. Enumerating the consistent states with respect to a relative time-scale does not produce a tractable set of testable consequences useful for advancing understanding. As Kuipers (1994) notes each QSIM QDE corresponds to many possible ODEs. The set of possible behaviours consistent with a set of weak functional relationships will itself be consistent with many possible explanations, many possible ways in which the ecological system could be working.

Excessive branching can be countered with global filters when envisioning well-understood physical systems as the possible qualitative behaviours of the whole system are likely to be known in advance. Salles (1997), in simulating Brazilian vegetation dynamics using process-based QR (GARP) was forced to reduce excessive branching by disregarding some model relationships, redefining the set of possible global states (vegetation types) and by pruning the states produced according to some global rules. To control simulation in this way requires that system behaviour be well understood otherwise the modeller risks discarding valid and interesting states and transitions. Although the Brazilian vegetation examined by Salles (1997) may have been suitably well understood in terms of vegetation type transitions, it is very unlikely that the same is always true elsewhere. Vegetation does not necessarily behave in predictable ways.

To be useful for modelling vegetation dynamics a method capable of generating testable predictions concerning how discrete variable values and system state will change over absolute time is required. Given that relationships between ecological quantities are typically poorly understood it is argued here that representing and reasoning only with ecological facts such as value correspondences is more appropriate and useful than representing and reasoning with ecological facts embedded inside information-weak posited relationships. Based on the evaluation of QR techniques it would appear that the only outcome from positing and using such weak relationships, even if used with factual knowledge, is ambiguity and *excessive branching*. This is not useful, particularly when it is considered that such relationships may not even hold in reality. In addition excessively ambiguous simulations do not constitute useful management advice – they simply highlight the uncertainty of which ecologists and managers are already aware.

The techniques to be developed during the remainder of this thesis will demonstrate that it is more useful to simply represent what is known about vegetation in terms of factual and conditional statements representing correspondences and inequalities between variable values. Through representing knowledge fragments that define relationships in terms of such statements without positing weakly-specified functions it will be demonstrated that meaningful and tractable sets of predictions can be generated. It will be argued that such predictions and the systems for generating them can be used for increasing ecological understanding and producing management information.

3.2.3 The Use of Logic

Deterministic state transition, functional attributes and knowledge-based systems / modelling have all shown that logic in the form of factual and conditional statements ('if ... then' rules) can be easily used to represent non-quantitative and mixed support set relationships. First-order logic is capable of declaratively representing non-quantitative and mixed relations along with model structure (variables and influence directions) as demonstrated by Muetzelfeldt *et al.* (1989), Robertson *et al.* (1991) and Robertson *et al.* (1995). Correspondences and inequalities between different variable values can easily be represented using predicated facts (e.g. correspondence(X, Y)) or rules (e.g. **IF** X = low **AND** Y < medium **THEN** Z = low).

The fragmented form of available vegetation knowledge makes it very amenable to declarative representation separate from concerns of reasoning and computation. The splitting of knowledge and reasoning (control) elements is one of the main features of any knowledge-based system and provides several major benefits:

- Knowledge may be expressed declaratively (i.e. as true statements), without concern for the (procedural) application of that knowledge; an act which Sterling and Shapiro (1994) contend 'clears the mind' and can act as a 'tool for thinking'.
- Knowledge-bases may be easily changed because the knowledge represented within a KBS is expressed declaratively and not embedded within a control structure (reasoning system). This means that different knowledge-bases, given that they use the same knowledge representation syntax, may be 'plugged into' the same reasoning system easing update and facilitating re-use.

The techniques to be developed will be based around a knowledge-based systems / modelling architecture whereby models are represented using a mixture of facts and 'if ... then' conditional statements held in rule-bases. A separate reasoning system based on the principles of forward chaining will be used to control simulation execution and reason with the knowledge fragments represented in each rule-base to deduce how model variables will change over time.

One of the major tasks in modelling using available knowledge is the development of adequate representational constructs. A set of such constructs is termed a knowledge representation (KR) scheme Stillings *et al.* (1987). In the context of the techniques to be developed each KR scheme can be viewed as a modelling language in the same way that the graphical constructs behind compartment-flow modelling systems like Simile (Muetzelfeldt 1995) and STELLA.

The logic-based programming language Prolog will be used to implement both the knowledge representation and reasoning components of the techniques to be developed. Prolog was chosen for the following reasons:

- It is high-level and very easy to read (Muetzelfeldt *et al.* 1989, Guerrin 1991)
- It supports a declarative style of programming and knowledge representation (Muetzelfeldt *et al.* 1989, Guerrin 1991).
- Being a form of first-order logic, Prolog is very flexible, applicable to any problem domain, and satisfies the basic requirements of a general purpose representation formalism (Moore 1982).
- Prolog is driven by the Prolog interpreter, an 'engine' that includes powerful built-in symbolic manipulation and reasoning facilities (Guerrin 1991, Clocksin 1997).

Predicates provide the structural basis for all rules and facts in Prolog and will form the basis to the modelling system KR schemes to be developed. Figure 9 details and example predicate taken from the modelling system detailed in Chapter 4. All predicates have their name before their brackets and contain their arguments within their brackets. Each argument represents a piece of information and can be instantiated with a particular value or be a 'logical variable' and represent an unspecified value. Predicate names must start with a lower case character and argument names follow the same rule unless an argument denotes an logical variable in which case it should start with an uppercase character.

```
transition_rule(Site,T,State,State_next).
```

Example:

```
transition_rule(Site,T,euphorbia_dendroides,poor_halophytic_gra
ss):-
        series(Site,1),
        disturbance_event(Site,T,grazing,high).
```

which can be read as,
IF a site's vegetation is following vegetation series 1 **AND** there is currently high intensity grazing disturbance influencing the site **THEN** the vegetation at that site will change from *Euphorbia dendroides* towards a poor halophytic grassland.

Figure 9 Example of a Prolog predicate

All Prolog predicates have two possible forms - facts or rules. Facts are used to represent unconditional knowledge and are structured as predicates ending in a period stop after the last bracket. Conversely, rules (such as `transition_rule` above) are used to represent conditional knowledge and use a general "if ... then" structure composed of two sections, a 'head' and a 'tail'. The 'head' is that part lying before the ':-' or 'if' symbol and it details the consequences of the rule succeeding as well as serving to pass any information to the 'tail' that may be required. The 'tail' or 'body' of the rule is that part which lies after the ':-' symbol. It details the conditions that must be met for the rule to succeed.

In a rule, conditions may either be various types of operation such as arithmetic and Boolean comparison or queries to other predicates (facts or other rules). This feature allows very complex statements to be built up with conditions nested to any depth. Nesting provides Prolog with considerable expressive and representational power.

3.2.4 Representing and Reasoning with Derivatives and Levels

The direction of some knowledge-based modelling work and the basis to most QR is separately representing and reasoning with variable derivatives and levels. Qualitative direction of change can usefully be represented and reasoned with using the set {inc, std, dec} or {-1, 0, +1}. In doing so QR techniques solve the *direction of change* problem for qualitatively valued variables but not the *rate of change* problem. Neither QSIM nor QPT can represent the magnitude of rate of change with the result that ambiguous and excessively branching simulations are produced. Salles (1997) notes in his concluding discussion that representing and reasoning with the magnitude of derivatives is an open and interesting research topic.

Rate of change magnitudes can be represented using qualitative support sets themselves (e.g. low rate of change, medium rate of change etc.). It is not clear that knowledge is available regarding the magnitudes of rates of change in this form. Alternatively the speed at which a quantity is changing can be used to determine when it changes value. For example under some circumstances it may be known that plant biomass will take 5 years to go from low to medium whilst under others it will take 10 years. It will be contended that knowledge of this form is more readily available at both community and species-levels. The techniques to be developed for modelling vegetation based on available knowledge will use one set of rules to determine the direction of change for discretely valued variables and another separate set of rules to determine how long it will take for a single change in value to occur. Increases in time to change value correspond to decreases in the rate of change whilst decreases in the time required to change correspond to increases in rate of change. By reasoning separately with direction and rate the techniques to be developed will be shown to be capable of handling situations whereby a variable's direction of change remains constant but its rate of change changes. In addition by representing rate of change using a temporal measure changes in discrete variable values can be modelled as occurring at specific points along an absolute time-scale.

To accomplish this in a generic way (i.e. not support set or variable specific) a temporal reasoning system will be developed to update discrete variable values based upon their direction of change and time required to change. The system will demonstrate that, despite not being based on an underlying numerical scale, the concepts of direction and rate of change can equally be applied to linguistically valued variables albeit with different semantics.

3.2.5 A Modelling Systems Approach

Rather than producing one-off models with hard-wired variables, a more general approach will be taken whereby the techniques to be developed will each be structured as individual modelling systems. Using a technique's KR scheme as a modelling language it will be possible for a user (ecologist, ecological modeller) to representing available knowledge and specify models to suit the vegetation and questions being asked within certain ontological and representational constraints. The separate reasoning system will be responsible for producing simulations. This is in line with the aims of weak decision-support Robertson *et* *al.* (1995) to provide users (ecologists, vegetation managers etc.) with sets of tools for representing and reasoning with ecological knowledge rather than fixed structure models.

3.2.6 Summary

The following points can be made regarding the modelling approach to be developed:

- *Heterogeneous variable values:*

  The approach to be developed will be capable of representing and reasoning with quantitative, qualitative and linguistic values as required by available knowledge.
- *Relationships based on factual knowledge:*

  Positing weak relations serves only to heighten ambiguity in simulation. The approach to be developed will concentrate on representing and reasoning with ecological facts concerning how quantities are related through the use of correspondences and inequalities.
- *Logic-based representation:*

  A declarative rule-based approach will be taken to modelling, following the principles of KBS and separating representation from reasoning. First-order logic, implemented using Prolog will be used for the task.
- *Derivatives and levels*:

  Separate rules will be used to determine the direction and rate of change of non-quantitative variables. Rate of change will be modelled based on knowledge concerning the length of time required for change in value to occur and form the basis to a temporal reasoning system.
- *Modelling systems*:

  Rather than produce one-off hard-wired models of vegetation, a modelling systems approach will be pursued providing solutions to classes of problems rather than single problems.

Following these points, three ontologically distinct modelling systems will be developed and described over the next three chapters, each capable of modelling using different aspects of available vegetation knowledge. Starting from its most primitive form, ending in its most advanced, and running through the three chapters will be the development of the general temporal reasoning system for modelling using heterogeneous value and relationship knowledge. At the end of each chapter the technique detailed will be discussed in terms of the utility for modelling vegetation dynamics, the characteristics of the KR scheme used and the properties of the reasoning system employed. The utility of the temporal reasoning system developed, how it compares to other techniques and of rule-based modelling and modelling with available knowledge in general will be discussed in the final Chapter of the thesis.

Chapter 4 Modelling Vegetation Dynamics I – A Rule-Based State Transition Approach

4.1 System Basics

4.1.1 System Ontology and Model Structure

Deterministic state transition modelling is a recognised method for predicting vegetation dynamics using coarse-grained understanding based on classifying vegetation into a series of discrete states or types and treating dynamics in terms of transitions between states under different conditions. The conceptual ideas underlying the approach can be traced to the succession to climax theories and community-unit ideas of Clements (1904, 1916, 1928) and the ‘orthodox climax theory’ that developed over the course of the early 20th Century (Burrows 1990). Vegetation dynamics is now largely held as not conforming to the expectations of these theories (Smith and Huston 1989, Burrows 1990, Glenn-Lewin and van der Maarel 1992) but the state transition approach is still noted as useful under certain conditions (Keane *et al.* 1996, Hobbs 1997).

Mediterranean vegetation types can be relatively well defined and dynamics in terms of state transitions similarly well characterised under some conditions (Quezel 1981a, Tomaselli 1981a). In addition Mediterranean vegetation is often relatively resilient (Keeley 1986), compositionally and structurally resisting environmental and disturbance influences, making its response to certain influences, particularly fire, relatively predictable. This can make the state transition approach suitable for modelling Mediterranean vegetation dynamics under some circumstances.

Current implementations of the approach have two major limitations (see Chapter 2). First, they do not clearly separate direction of vegetation change from rate of vegetation change and consequently cannot handle changes in rate that do not affect direction. Second, other than disturbance events, they cannot handle the effects of changing environmental conditions upon a given stand. The Rule-Based Community Level Modelling 1 (RBCLM1) system is the first of the three modelling systems developed during the course of this research and aims to address these points. Note however that the aim of the RBCLM is not to be a modelling system capable of providing mechanistic insight, but rather to provide a modelling system capable of better using state transition understanding where it is available and applicable.

The RBCLM1 represents the world as a single patch of land, a site, containing a vegetation component and an environment component (see Figure 10).

A site's vegetation component is represented using four concepts:

- *State*:
  The main variable of interest in any RBCLM1 model is vegetation state. Vegetation state is an unordered linguistic state variable where each value element corresponds directly to a real-world community type defined by some classification criteria. The set of possible states and the set of possible transitions between those states together define a site's vegetation series and constrain how vegetation may change over time. Different vegetation series can occur under different environmental conditions. If a site's environmental conditions change so it may its vegetation series.
- *Time in State:*
  Vegetation takes time to change from one state to another. The length of time that a site's vegetation has been in its current state under its current direction of change (see below) is represented using the notion of 'time-in-state', represented by the symbol 'T-in'. Time-in-state along with 'direction of change' and 'time required to change state' form the basis to modelling vegetation dynamics in the RBCLM1 and the approach taken to addressing the direction and rate of change problems.
- *Direction of Change:*
  Vegetation state changes in response to environmental conditions. Different vegetation types respond differently to different combinations of conditions so the way in which vegetation at a site changes state depends on its current state and current environmental conditions. The state towards which a site's vegetation is changing is said to be the vegetation's direction of change ('D-Delta').
- *Time Required to Change State:*
  The transition from one state to another takes time. Time required to change state ('T-Delta') is a function of the present state, the direction of change and environmental conditions. If conditions change T-Delta can change whilst D-Delta remains constant or alternatively both can change. The two concepts are used in the RBCLM1 to permit situations to be modelled whereby rate of change is affected by a change in the environment but direction of change is not. Systems like CRBSUM cannot handle such scenaria. The way in which different D-Delta and T-Delta values occur under different

conditions can be viewed as expressing coarse-grained relationship knowledge between vegetation type and environmental factors.

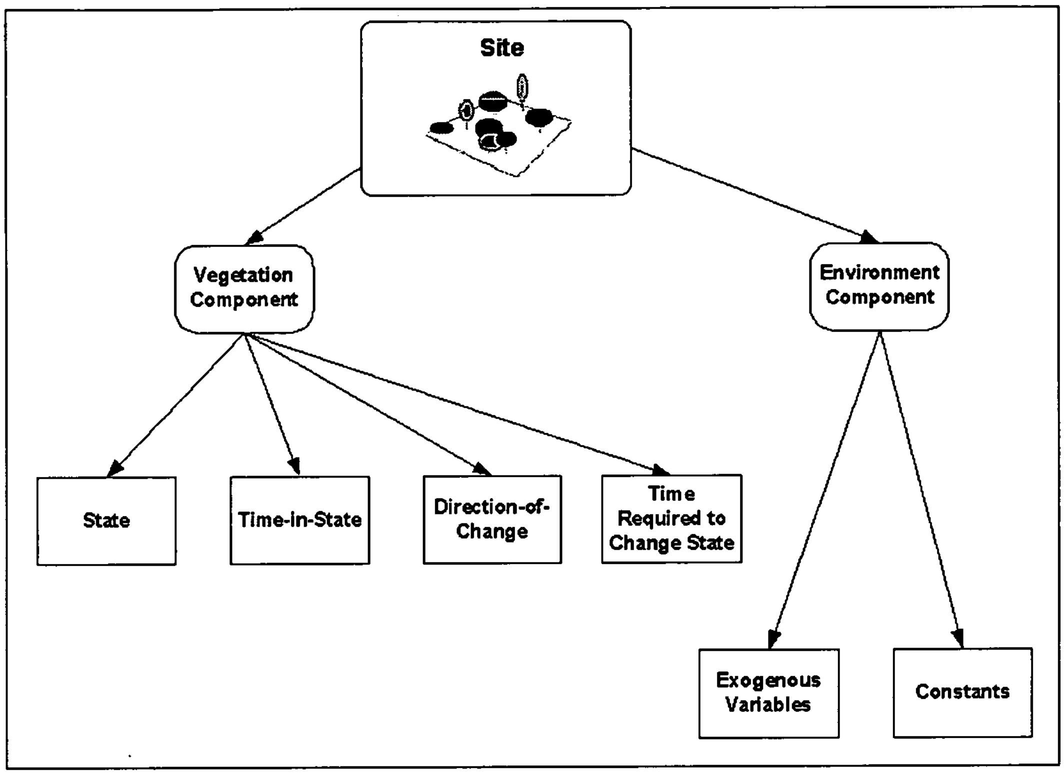


Figure 10 Informal RBCLM1 system ontology (arcs represent 'has a' relationships, rectangular boxes represent single model variables or sets of variables)

A site's environment component is represented using a set of environmental properties categorised into two different types of variables:

- *Exogenous variables:*

  Exogenous variables are used to represent those influences whose values may change during the course of a simulation run e.g. amount of precipitation, temperature, number of sheep etc. Disturbance factors, both prolonged influences such as grazing, and sudden events such as fire, are also represented using exogenous variables.

- *Constants:*

  Constants are used to represent those influences whose values remain constant for a given site over the course of a simulation run e.g. altitude, distance from sea etc.

The RBCLM1 system ontology provides the language for and constraints within which individual models must be developed rather than being a fixed structure model. This is also true of the modelling systems described in Chapters 5 and 6.

In any RBCLM1 model the structure of the vegetation component is the same - state, T-In, D-Delta and T-Delta. Defining the vegetation series for each model (i.e. the states and transitions that are possible) is however the task of the model developer. Each model can have many possible vegetation series, each representing the constraining and controlling effects of different environmental conditions on the states and transitions that are possible e.g. a model may have nutrient rich substrate series and a nutrient poor substrate series with different states and transitions occurring in each. Models may have one vegetation series or many series and vegetation may change from one series to another depending on environmental conditions. For example, taking the previous nutrient example, if a site's substrate became nutrient poor mid-run, the vegetation series would change to simulate the controlling effect of nutrient status on possible vegetation dynamics. Reasoning about T-Delta and D-Delta separately in addition to allowing a site's vegetation series to change during a run in response to environmental change gives the RBCLM1 greater flexibility than present approaches to deterministic state transition modelling. The inclusion of vegetation series also provides a means of more easily spatialising the system where vegetation at different points can be modelled as following different series (different states and transition possibilities).

RBCLM1 models do not necessarily share the same environmental component structure. There is no need to represent a site's environment if it is not required for a particular model. What variables are to be included in the environment component and what support sets should be used for those variables is left to the model developer. This allows the environmental influences that are important for particular modelling problems to be included.

4.1.2 Simulation Overview

The RBCLM1 employs 'if...then' rules and unconditional facts to represent and run vegetation models. At each time-step during a simulation run, a set of 'if ... then' state transition rules are used to determine the vegetation's direction of change based upon current vegetation state and the value of any environmental variables influencing the site at that point in time. The RBCLM1 reasoning system then uses knowledge encoded in a T-Delta rule-base along with a specific update algorithm to calculate the time required to change state value.

The T-In variable is then updated by the RBCLM1 reasoning system based upon whether or not the current state differs from the previous state. Last, the vegetation state is updated based on whether or not the current T-In is greater than or equal to the current T-Delta value. If it is the state changes according to the direction of change, if not it stays the same.

4.1.3 System Architecture

4.1.3.1 Overview

The system is made up of three logical modules (see Figure 11). One module, the reasoning system is 'hard-wired', providing the core reasoning and simulation functionality required for all RBCLM1 models. The other two modules, the knowledge-base (KB) and simulation data-file are changeable, providing the specific knowledge and data that comprise each individual model.

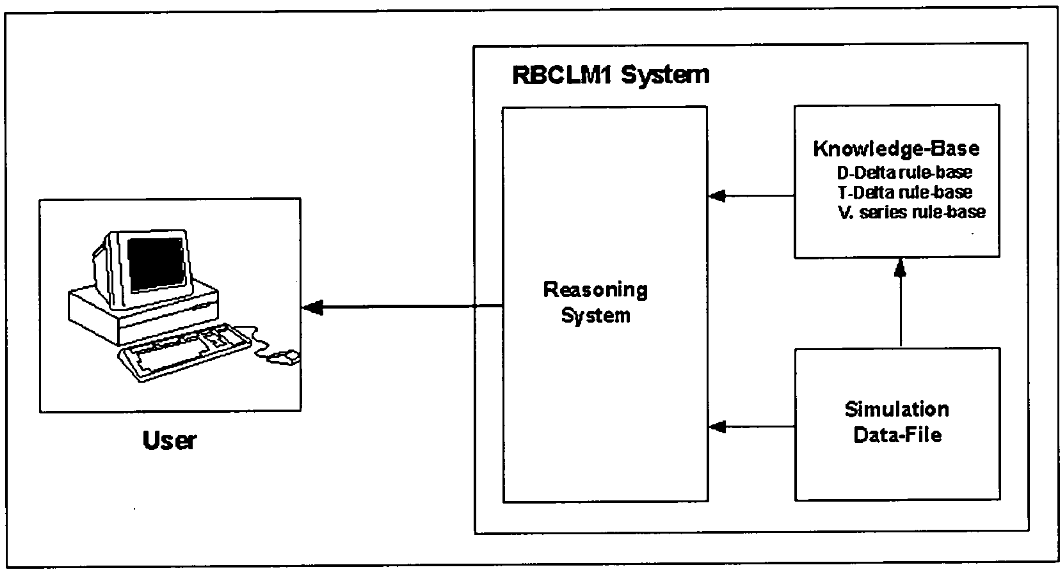


Figure 11 RBCLM1 system architecture and operation (arcs represent information flows resulting from Prolog predicate calls)

4.1.3.2 Reasoning System

The reasoning system is responsible both for driving simulations and for accessing and reasoning with the knowledge contained in both the KB and simulation data-file. It contains rules for calculating the time required to change state, for updating time in state, determining when state transitions occur and for determining disturbance events. In addition, the reasoning system contains procedures for displaying the results of simulation runs.

4.1.3.3 Knowledge-Base

Each KB can be regarded as a separate model of vegetation dynamics. Each KB contains rules that determine how vegetation will change state under different conditions, rules that determine which series of states a site's vegetation belongs to and facts specifying the time required to change state for each transition under various conditions.

Each KB module is divided into three sections:

- *State transition (D-Delta) rule-base*:

  Rules for determining the direction of change based upon the current state and the current environmental influences.
- *Time required to change state (T-Delta) rule-base*:

  Rules and/or facts detailing how long it takes for each state transition to occur under different environmental conditions.
- *Vegetation series (v. series) rule-base*:

  Rules and/or facts that are used to determine which vegetation series a site's vegetation belongs to, typically based upon environmental influences.

4.1.3.4 Simulation Data-File

Each simulation data-file contains facts and / or rules that detail the environment component of each site for a given model and for a given run. The simulation data-file contains information detailing the values for all exogenous variables and constants for each site for the course of a simulation.

4.2 Knowledge Representation

4.2.1 Knowledge Representation Scheme – Introduction and Notation

The basis to modelling using the RBCLM1 system is the system's knowledge representation, or KR, scheme. The KR scheme provides a set of representational constructs that can be regarded as a language for modelling vegetation dynamics using the state transition ontology already outlined. The RBCLM1 KR scheme is expressed in Prolog (see Chapter 3).

The next four sub-sections detail the predicates used by the RBCLM1 KR scheme to represent ecological knowledge. The predicates used by the RBCLM1 to represent knowledge will all be expressed in fact form (i.e. 'head' section only) even if they may more commonly be used as rules. Examples will show how each predicate can be used as a rule if this form is applicable. Table 29 details the arguments used by some or all of the predicates in the RBCLM1 KR scheme.

Argument Name	Typical Value	Description
Name	grazing	Variable or quantity name
Series_label	1	Name or number of a modelled vegetation series
Site	1	Site number (or name) being simulated
State	poor_halophytic_grass	Vegetation type for the current time-step
State_next	juniperus_phoenicia	Vegetation type for the next time-step
T	28	The simulation time-step when a particular event will occur or exogenous variable value will apply
T_delta	4	Time required to change state
Value	medium	The current value of a variable (except vegetation state)

Table 29 RBCLM1 KR scheme argument notation

4.2.2 Knowledge-Base: State Transition Rule-Base

The way in which vegetation is changing at a site is represented by the notion of direction of change or D-Delta i.e. the state towards which vegetation is developing. D-Delta will typically be determined by different combinations of environmental condition values. For example:

> **IF** there is a fire event **THEN** poor quality halophytic grassland will change towards a 'coastal burnt' community.

> **IF** there is no grazing **OR** grazing of low **OR** medium intensity **THEN** poor quality halophytic grassland will change towards a *Euphorbia dendroides* dominated shrub community.

As well as disturbance events such as grazing and fire, other environmental properties such as soil type, rainfall etc. may influence the direction of vegetation change. Note here that knowledge concerning the amount of time required for each transition to occur (T-Delta value) is kept separate. It is possible that each transition may take different lengths of time to occur depending on the conditions at a site and how they change during a run. The T-Delta values associated with each transition under each set of conditions are held in the T-Delta database. How the Reasoning System calculates and updates T-Delta values will be detailed later.

The rules detailing D-Delta are held in the transition rule-base using the predicate `transition_rule` (see Figure 12).

```
transition_rule(Site,T,State,State_next).
```

Example:

```
transition_rule(Site,T,euphorbia_dendroides,halophytic_grass):-
      series(Site,1),
      disturbance_event(Site,T,grazing,high).
```

which can be read as,
IF a site's vegetation is following vegetation series 1 **AND** there is currently high intensity grazing disturbance influencing the site **THEN** the vegetation at that site will change from *Euphorbia dendroides* towards halophytic grassland.

Figure 12 RBCLM1 `transition_rule` predicate

4.2.3 Knowledge-Base: Time Required to Change State Database

As well as determining the direction of vegetation change at a site we also need to know how long the change is going to take if we are interested in making predictions using discrete time-steps. For each possible state transition each combination of influencing variable values is represented as having an associated 'T-Delta' value. For example:

> **IF** there is either zero **OR** low intensity grazing **THEN** the transition from poor quality halophytic grassland to *Euphorbia dendroides* dominated shrubland will take 4 years.
>
> **IF** there is medium intensity grazing **THEN** the transition from poor quality halophytic grassland to *Euphorbia dendroides* dominated shrubland will take 6 years.
>
> **IF** there is either low **OR** medium intensity grazing **AND** the substrate is calcareous **THEN** the transition from *Phlomis* dominated phrygana to degraded phrygana will take 10 years.

The RBCLM1 uses a time-unit and time-step length of 1 year. Each T-Delta value represents the typical length of time in years that it takes for a state transition to occur under a given set of conditions (combination of influencing variable values). Values for influencing variables represent the average or most extreme value over the period of a year. Under some circumstances it may be that the most extreme (i.e. lowest or highest) influencing variable values exert a more important controlling effect on vegetation dynamics than the average value for a year. In such cases transition D-Delta and T-Delta values should be made conditional on those extreme values. In Mediterranean vegetation for example the annual thermal minima and maxima exert a more important controlling effect on vegetation than average temperature (Quezel 1981b).

Due to the RBCLM1 using an annual time-step the effects of seasonal variation are assumed to be included in the T-Delta values. Seasonality in climatic variables tend to cause fluctuations in vegetation rather than species replacement dynamics (Glenn-Lewin and van der Maarel 1992). As such they do not cause directional change in composition and / or structure and are ignored in RBCLM1 models. Only those combinations of influences that cause directional change are included.

The RBCLM1 represents T-Delta knowledge using the predicate `t_delta` (Figure 13).

`t_delta(Site,T,State,State_next,T_delta).`

Example 1:

```
t_delta(Site,T,poor_halophytic_grass,euphorbia_dendroides,4):-
      disturbance_event(Site,T,grazing,low).
```

which can be read as,
IF there is low intensity grazing **THEN** the state transition from poor_halophytic_grass to euphorbia_dendroides will take 4 years.

Example 2:

```
t_delta(Site,T,poor_halophytic_grass,coastal_burnt,0):-
      disturbance_event(Site,T,fire,yes).
```

which can be read as,
IF there is a fire disturbance event **THEN** the state transition from poor_halophytic_grass to coastal_burnt vegetation will take 0 years i.e. instantly (where fire events are represented using a binary yes/no or presence/absence switch).

Figure 13 RBCLM1 `t_delta` predicate

T-Delta values of 0 are used to represent those combinations of influences that cause 'instant' (i.e. within one time-step) change in vegetation state. An example might be fire. T-

Delta values greater than 0 are used to represent those combinations of influences that exert a more gradual dynamic force e.g. soil moisture levels, grazing etc.

4.2.4 Knowledge-Base: Vegetation Series Rule-Base

Areas of vegetation under different environmental conditions such as proximity to the sea (increased aerial salt concentration from seaspray) and altitude often follow quite different vegetation dynamics (sequences of possible states and transitions). The concept of the vegetation series is a notion for characterising and constraining the way in which the vegetation of sites with similar environmental conditions can change. Ecologists sometimes represent differences in the way that vegetation changes at the community-level using the notion of vegetation series. Organising D-Delta and T-Delta rules by series can be a useful way of structuring an RBCLM1 model at the design stage.

Rules can be used to determine which series a site belongs to based upon appropriate environmental conditions. Then when defining D-Delta and T-Delta rules, rather than having to specify the full set of conditions upon which the rule is dependent, vegetation series can be used as a condition. The predicate `series` is used to represent vegetation series knowledge (Figure 14).

```
series(Site,Series_label).
```

Example 2:

```
series(Site,1):-
      exogenous_variable(Site,_,altitude,medium).
```

which can be read as,
IF the altitude of a site is medium **THEN** the vegetation will follow series 1.

Figure 14 RBCLM1 `series` predicate

If a site's environmental conditions change during a run then the site's vegetation series may also change. Changes in series may be induced by changes in factors that exert an overall influence or constraint on species composition such as soil nutrient status. Such environmental changes can be simulated during an RBCLM1 model run by specifying exogenous variable value changes (see section 4.2.5).

4.2.5 Simulation Data-File: Input Variables

The simulation data-file is used to store the predicates that represent the environment component of each site. This includes predicates to represent exogenous variable names and values and predicates to represent constant names and values. Typical examples of exogenous variables in RBCLM1 models are disturbance factors such as grazing, fire events and environmental influences such as soil moisture and soil nutrients. Vegetation related factors that play a role in spatial processes such as seed distribution can also be represented as exogenous variables. Typical examples of constants in RBCLM1 models are altitude and exposition. Each simulation data-file specifies all modelled exogenous variable and constant values for all sites for the whole length of a simulation run. The simulation data-file also stores information about the priority of effect for each disturbance factor included in a model (see below for details).

The predicate `exogenous_variable` (Figure 15) is used by the RBCLM1 to represent knowledge about exogenous variables whilst the predicate `constant` is used to represent knowledge about constants (Figure 16). The value of exogenous variables or constants may be easily altered by changing the Value argument in their predicates.

`exogenous_variable(Site,T,Name,Value).`

Example1:

```
exogenous_variable(1,_,wind_speed,medium).
```

which can be read as,
The wind_speed at site 1 is medium regardless of the current simulation time (note the '_' character used to show that the time argument, T, is to be ignored).

Example 2:

```
exogenous_variable(1,T,rainfall,low):-
        T<7.
```

which can be read as,
IF the current simulation time, T, is less than 7 (time-steps) **THEN** the rainfall at site 1 will be low.

Figure 15 RBCLM1 `exogenous_variable` predicate

`constant(Site,Name,Value).`

Example:

```
constant(1,distance_from_sea,far).
```

which can be read as,
Site 1 has a distance_from_sea equal to far.

Figure 16 RBCLM1 `constant` predicate

Disturbance events and factors generally tend to cause vegetation to change in specific ways, from a current state towards a particular next state. This may take a few months or years where the disturbance has a gradual effect (e.g. continued moderate grazing pressure) or may occur over the course of a few hours to a day where the disturbance has a sudden and strong influence on vegetation structure (e.g. a fire event). Other environmental factors (e.g. soil type) tend to constrain vegetation to particular series rather than inducing change in a specific direction.

The RBCLM1 reasoning system limits the number of disturbance events that can exert an effect on a site's vegetation to one per time-step (i.e. one per year). Those exogenous variables that are considered as disturbance factors should be registered as having a 'priority of effect' in the simulation data-file using the predicate priority.

It is possible that the simulation data-file contains more than one exogenous variable capable of causing a disturbance during a single time-step. To decide which exogenous variable will actually cause a disturbance the reasoning system uses a priority system. If for example both fire and grazing are scheduled to occur during the same time-step the reasoning system will decide which will have the dominant effect based upon the pre-determined 'priority of effect' values specified in the simulation data-file. The priority values for each type of disturbance under the general coastal Mediterranean model to be described in section 4.4 are as follows:

- Fire - 1.
- Erosion – 2.
- Grazing - 3.

Thus, if there are 2 or more possible disturbance event types for the same site during the one time-step then the one with the highest priority w ill exert an effect, where 1 = highest priority (in the case of the above three factors this would result in fire).

```
priority(Name,Priority_value).
```

Example:

```
priority(fire,1).
```

which can be read as,
The exogenous variable fire has a disturbance effect priority of 1.

Figure 17 RBCLM1 `priority` predicate

4.3 Reasoning System

4.3.1 Operation

The RBCLM1 reasoning system is embedded into a stand-alone simulation engine that keeps track of time and acts as a control system for multi-time-step simulation runs. Together these components access and use the knowledge contained within knowledge-base and simulation data-file to provide discrete time-step predictions of vegetation dynamics under a range of conditions. To run a multi time-step simulation the RBCLM1 reasoning system and simulation engine follows the steps detailed in Figure 18.

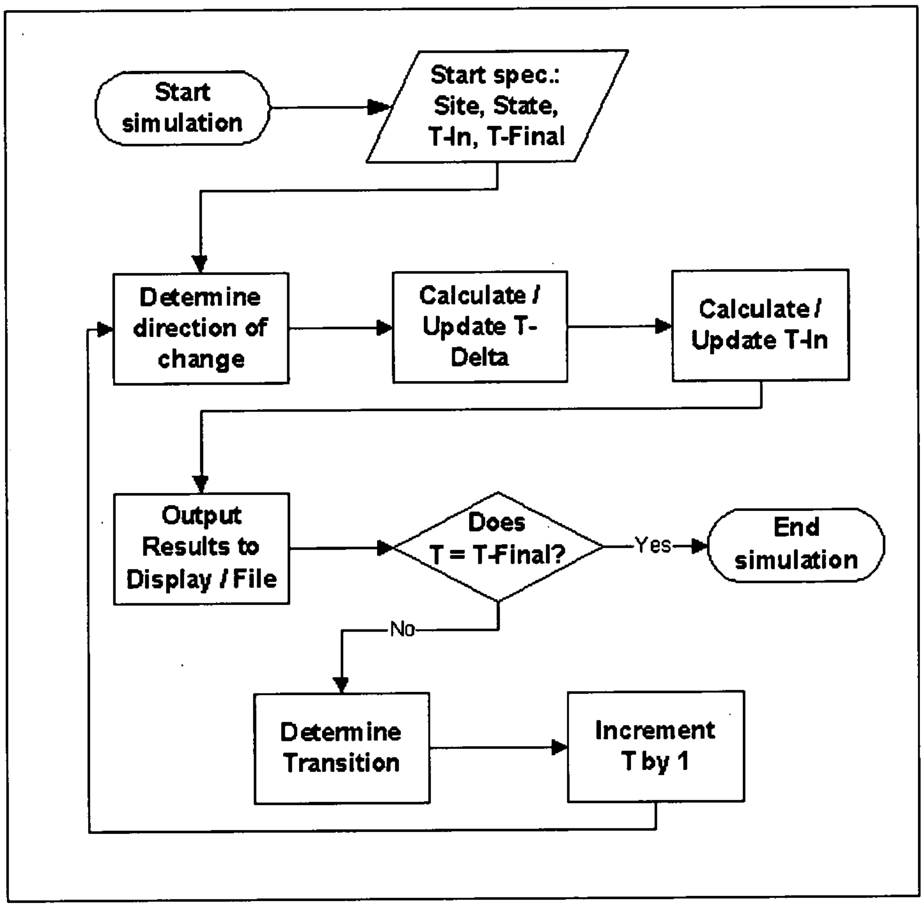

Start simulation
Start spec.: Site, State, T-In, T-Final
Determine direction of change
Calculate / Update T-Delta
Calculate / Update T-In
Output Results to Display / File
Does T = T-Final?
Yes
End simulation
No
Determine Transition
Increment T by 1

Figure 18 RBCLM1 reasoning system operation flowchart

The user, in addition to supplying a correctly constructed KB and simulation data-file, must also initialise the simulation by supplying the number (or name) of the site to be simulated, the current vegetation state and time in state value (T-In) for that site plus the desired length of the simulation run (T-Final). If only one site is being simulated then the run may be as many time-steps long as desired. If multiple sites are being simulated then each site should be simulated for one time-step only before the next site is simulated. The RBCLM1 reasoning system and simulation engine itself does not have spatial simulation capabilities. However, as shown by Mazzoleni and Legg (2001) the RBCLM1 can be called by other simulation engines suitable for this task such as the ModMED Landlord spatial modelling environment.

Each time-step the RBCLM1 goes through the update loop depicted in Figure 21. First of all, the transition rule-base are used to determine the current direction of change. Then, the T-Delta is either calculated afresh or updated (see section 4.3.3 for details) and the time-in-state either incremented by 1 if the state hasn't changed since the previous time-step and reset to 1 if it has. The current time-step's results are then outputted to the screen or to a file depending on where the output stream is directed. The results outputted are state, time in state, direction of change, time required to change state, the operational disturbance name (the exogenous variable with the lowest disturbance priority) and the operational disturbance value. At this point the simulation will terminate if the desired end-time has been reached, otherwise the T-In and T-Delta figures are compared to determine whether a state transition occurs before the time counter increments by 1 (next T = current T +1) and the simulation loops restarts with updated variable values passed.

4.3.2 Reasoning System Predicates: Disturbance Event Handling

The reasoning system determines which disturbance event affects a site's vegetation during any given time-step. This is achieved using the reasoning system predicate `disturbance_event`, which may be used as a condition in the direction of change rule-base predicate `transition_rule`. `disturbance_event` accesses the simulation data-file to determine what (if any) exogenous variable is causing a disturbance during the given time-step using the priority system, then checks the operational disturbance against the specified disturbance (e.g. low grazing). If the specified disturbance is the same as the operational disturbance then the predicate `disturbance_event` succeeds as a condition. If not it will fail causing the predicate that queried (or called) it to also fail. Figure 19 illustrates this use of `disturbance_event`.

```
disturbance_event(Site,T,Name,Value).
```

Example (for use as a condition within the predicate `transition_rule`*):*

```
transition_rule(Site,T,euphorbia_dendroides,halophytic_grass):-
        series(Site,1),
        disturbance_event(Site,T,grazing,high).
```

which can be read as,
IF a site's vegetation is following vegetation series 1 **AND** there is currently high intensity grazing disturbance influencing the site **THEN** the vegetation at that site will change from *Euphorbia dendroides* towards halophytic grassland.

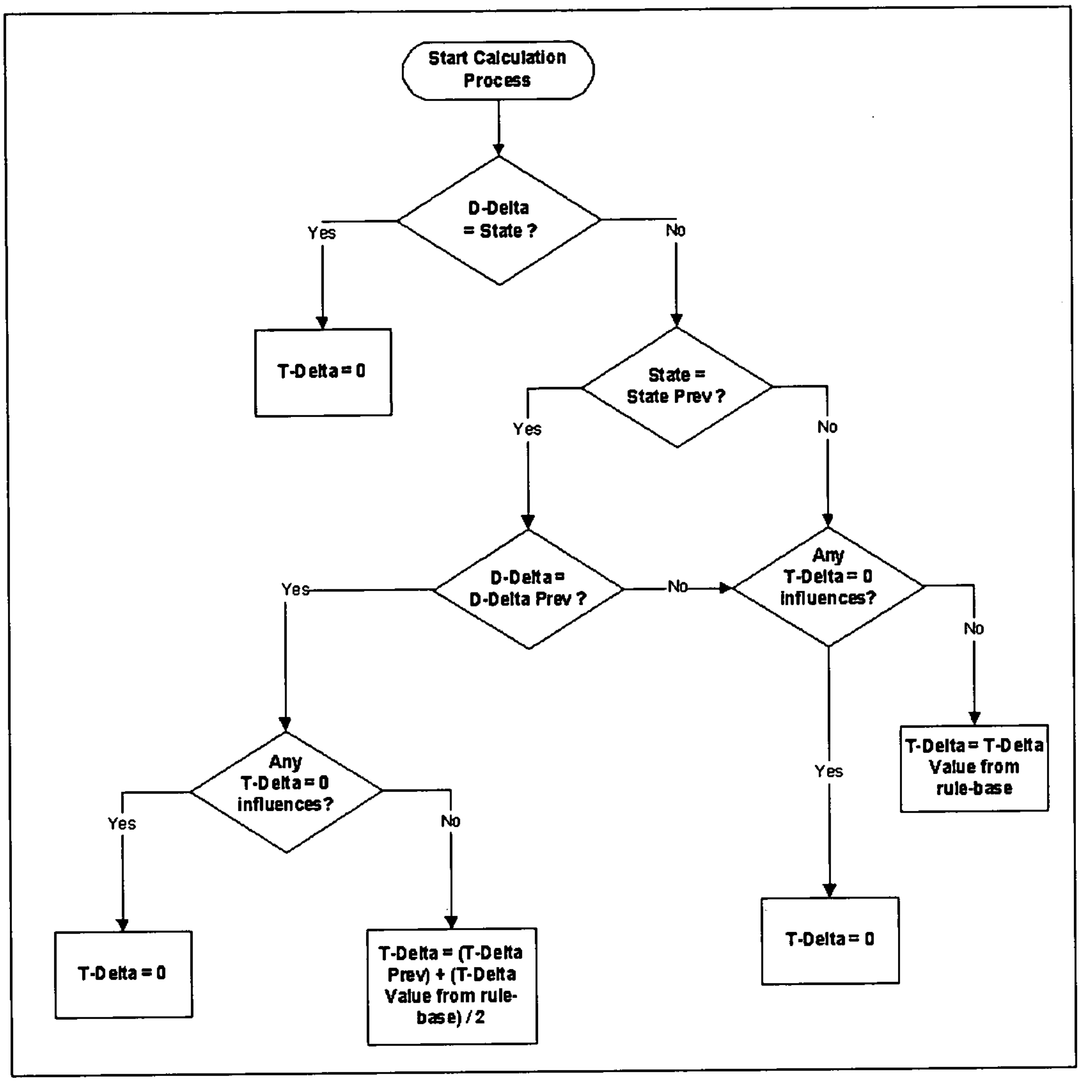
Figure 19 RBCLM1 `disturbance_event` predicate

4.3.3 Time Required to Change State Calculation and Updating

The representation of T-Delta values for each value of each exogenous variable being modelled has already been described. This section will detail how the actual T-Delta values for each direction of change and set of environmental conditions are calculated and updated over time from the facts held in the T-Delta database. Figure 20 details the algorithm involved.

If a site's vegetation is in a stable state (i.e. if it's direction of change, D-Delta is the same as the current vegetation state) then T-Delta is set to zero as no directional change is occurring. If directional change is occurring (i.e. D-Delta ≠ the current vegetation state) then the current T-Delta value first of all depends on whether the site's vegetation has changed state from the previous time-step or not. If it has then T-Delta must be calculated afresh. If the state hasn't changed from the previous time-step then current T-Delta may depend on the previous time-step's T-Delta value.

Taking the first case, where vegetation state has changed between the previous and current time-steps, if there are any 'instant effect' disturbances in operation then T-Delta is set to 0. 'Instant effect' disturbances are represented as having an individual T-Delta value of 0 in the T-Delta database. If there are no such disturbances then the T-Delta value for the particular state and direction of change under consideration will be set equal to that specified by the T-Delta rule which matches current environmental conditions.



Start Calculation Process
D-Delta = State ?
Yes
No
T-Delta = 0
State = State Prev ?
Yes
No
D-Delta = D-Delta Prev ?
Yes
No
Any T-Delta = 0 influences?
No
T-Delta = T-Delta Value from rule-base
Any T-Delta = 0 influences?
Yes
No
Yes
T-Delta = 0
T-Delta = (T-Delta Prev) + (T-Delta Value from rule-base) / 2
T-Delta = 0

Figure 20 RBCLM1 Time Required to Change State Calculation and Update Algorithm (D-Delta = current direction of change, D-Delta Prev = previous time-step's direction of change, State = current vegetation state, State Prev = previous time-step's vegetation state, T-Delta = current time required to change state, T-Delta Prev = previous time-step's time required to change state)

The process of calculating T-Delta for the second case, where vegetation has not changed state between time-steps, can be further divided. Where the direction of change has changed between the previous and current time-steps (i.e. the environmental conditions have changed, causing the vegetation to develop differently), T-Delta is calculated afresh from the T-Delta rule-base. Where the direction of change has stayed the same, two methods can be used to calculate T-Delta. Again, if any 'instant effect' disturbances are in operation, T-Delta is set to 0. If however there are no such disturbances, then a new T-Delta value is calculated by querying the T-Delta rule-base to determine an 'intermediate' value then averaging this with the existing T-Delta value kept in memory from the previous time-step, as follows:

$\text{T-Delta}_t = (\text{T-Delta}_{t-1} + \text{T-Delta}_i) / 2$

Where,

T-Delta_t = current time required to change state

T-Delta_{t-1} = previous time-step's combined T-Delta value

T-Delta_i = temporary 'intermediate' T-Delta value from rule-base

This permits the handling of cases where environmental conditions change such that, although the direction of change remains constant, the rate of vegetation change in that direction is altered. In doing so the concern over handling direction and rate of change separately is addressed.

4.3.4 Reasoning About the Occurrence of State Transitions

Once per time-step the reasoning system determines whether a state transition is to occur or not using the following two rules:

IF a site's vegetation's T-In value is greater than or equal to the current combined T-Delta value **THEN** a transition occurs and the next state set equal to the direction of change.

IF the next vegetation state under the current direction of change is the same as the present state (i.e. the vegetation is in a steady state) **OR** the vegetation's T-In value is less than the current combined T-Delta value **THEN** no transition occurs and the next state is automatically set equal to the value of the current state.

4.3.5 Time in State Calculation and Updating

If either the state or the direction of change change, then the T-In counter is reset. The update algorithm for T-In is detailed in Figure 21.

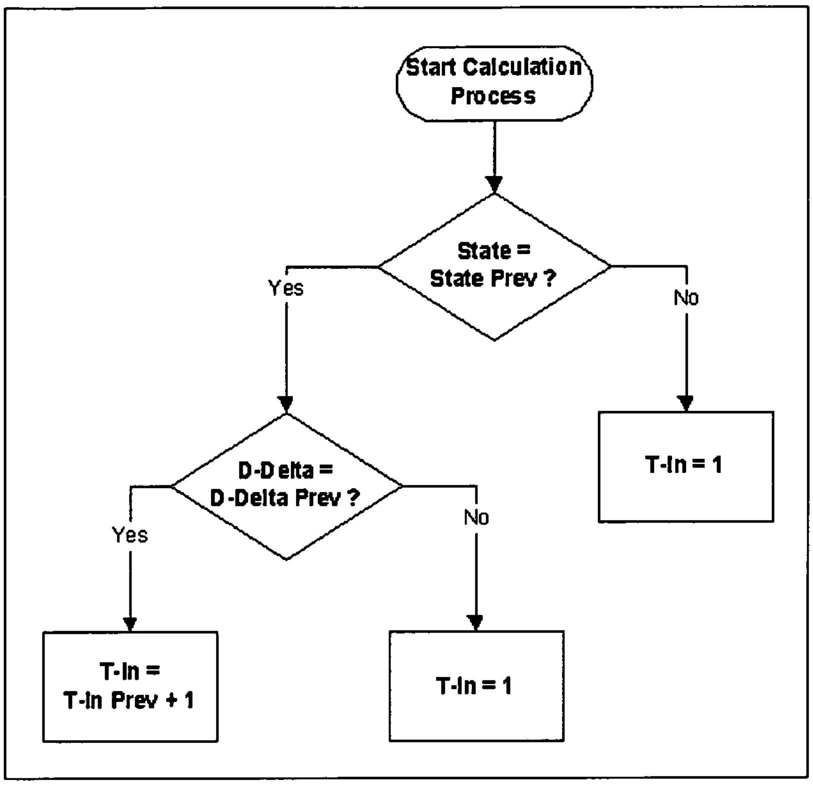


Figure 21 RBCLM1 time in state update algorithm (D-Delta = current direction of change, D-Delta Prev = previous time-step's direction of change, State = current vegetation state, State Prev = previous time-step's vegetation state, T-In = current time in state, T-In Prev = previous time-step's time in state)

4.4 Example Application – Capri Vegetation Model

4.4.1 Introduction

The RBCLM1 system was initially developed for a model of vegetation dynamics for the Island of Capri constructed as part of the ModMED II project. The initial Capri model only had one vegetation series with a total of six states typical of coastal areas and two exogenous variables – grazing, a gradual effect disturbance and fire, an instant effect disturbance. The model was developed in collaboration with the ModMED ecologists Prof. Stefano Mazzoleni and Dr. Colin Legg through various informal knowledge acquisition and consultative sessions. During the ModMED III project the model was developed further in collaboration with Prof. Stefano Mazzoleni, Dr. Colin Legg and Dr. Peter Csontos.

In addition to the original Capri RBCLM1 model, a state transition model of the vegetation of the Argolida region in Greece was developed in collaboration with Prof. Margarita Arianoutsou, Dr. Colin Legg and Prof. Stefano Mazzoleni as part of the ModMED III project under contract to the MODULUS project.

This section aims to provide documentation and simulation results for the Capri model to illustrate the RBCLM1 system in terms of model specification and operation. First of all some of the results from the knowledge acquisition exercises that were held during model development will be detailed to show the correspondence between available community-level dynamics knowledge and the constructs provided by the RBCLM1 system.

4.4.2 Community-Level Dynamics: Available Knowledge

The KA exercises yielded knowledge concerning a set of four different vegetation series defined by the factors 'distance from sea' and 'exposition' and specified in terms of states, possible transitions and the effect of different conditions on the time required to change state. The KA sessions were carried out using an informal semi-structured process that utilised tables for collecting and discussing knowledge concerning vegetation dynamics. Each table corresponded to a vegetation series. Table 30 details a section of the knowledge table detailing vegetation series 2 (see Figure 26).

Succeeding State Preceding State	**2.01 Coastal Bare Ground**	**2.02 Halophytic Vegetation**	**2.03 Annual Grassland**
2.01 Coastal Bare Ground		G0-G3 T-Delta = 1	
2.02 Halophytic Vegetation	Fire T-Delta = 0	G0-G2 T-Delta = n/a	G3 T-Delta = 2
2.03 Annual Grassland			G2-G3 T-Delta = n/a

Table 30 Extract from a Capri knowledge table showing conditions required for state transitions and associated T-Delta values (G = grazing where 0 = none 1 = low 2 = moderate & 3 = high, a T-Delta value of n/a is used to indicate a stable state)

The KA exercises demonstrate the way in which Mediterranean ecologists sometimes characterise vegetation dynamics. Knowledge regarding the relationships between vegetation types and environmental conditions was framed in phrases like 'under high grazing it generally takes 2 years for halophytic vegetation to change to annual grassland' which were then represented in tabular form for clarity. These phrases specify the way in which vegetation is known (or thought) to change under different conditions and the time required to do so. Although no explicit separation of direction of change from conditions and rate of change is contained within these phrases it is easy to split the knowledge into D-Delta and T-Delta rules for representation in an RBCLM1 model. For example 'under high grazing it generally takes 2 years for halophytic vegetation to change to annual grassland' can be represented using the two rules:

IF vegetation state = halophytic vegetation **AND** grazing = high **THEN** D-Delta = annual grassland.

IF vegetation state = halophytic vegetation **AND** grazing = high **AND** D-Delta = annual grassland **THEN** T-Delta = 2.

These rules can then easily be represented using the appropriate RBCLM1 D-Delta and T-Delta predicates. The close correspondence between acquired knowledge and the constructs of the RBCLM1 KR scheme facilitated knowledge representation and model construction.

4.4.3 Context

The Island of Capri lies off the Western Coast of the Campania region of Southern Italy within the humid Mediterranean bioclimatic type (Mazzoleni 1992). The Island has the typical Mediterranean climatic features of a relatively wet autumn and winter and a long, hot summer period with drought conditions (Mazzoleni 1992). The underlying geology is that of an alluvial plain and the Island supports a very rich flora including grasslands, maquis and forests (Mazzoleni 1992).

4.4.4 Structure and Value Knowledge

Figure 22 details the structure of the Capri Vegetation model including both the generic vegetation structure common to all RBCLM1 models and the model-specific environmental structure. The vegetation component uses 22 vegetation states organised into 4 different vegetation series (see Figure 23). The vegetation series were included in the Capri model to permit application of the model to different areas on the Isle of Capri and to facilitate use of the RBCLM1 for landscape-level modelling in the ModMED projects where representing different sites with different conditions across a landscape was necessary. The environment component is composed of two constants, distance from sea and exposition, and three exogenous variables, grazing, fire and cutting. Both distance from sea and exposition determine a site's series. Grazing, fire and cutting operate as disturbance factors. The effects of fire (priority 1) take priority over those of cutting (priority 2) and grazing (priority 3). Grazing and fire determine D-Delta and T-Delta.

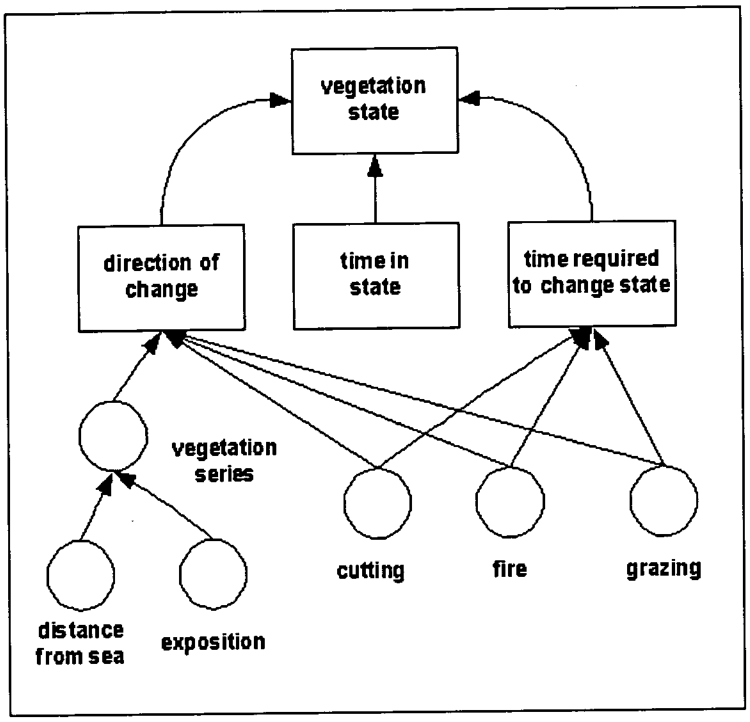


Figure 22 RBCLM1 Capri Vegetation Model structure (rectangular nodes represent state variables, circular nodes represent constants and intermediate or exogenous variables, arcs represent influence relationships)

Figure 23 shows the possible states for each of the four vegetation series included in the model. Four vegetation series were used:

> {'very close to the sea', 'close to the sea', 'far from the sea, northerly exposition', 'far from sea, southerly or plateau exposition'}.

The constant, distance from sea, was valued using a qualitative support set that can be mapped onto a numerical scale (metres):

> {very near, near, far} $\rightarrow$ {$(0 \le x \le 20), (20 < x \le 100), (x > 100)$}

The constant, exposition, was valued using a qualitative support set that can be mapped approximately onto a combination of slope direction (real, degrees with 0^0 or $360^0 \equiv$ magnetic north) and slope angle (real, degrees, with $0^0 \equiv$ horizontal and $90^0 \equiv$ vertical):

> {northerly, plateau, southerly} $\rightarrow$ {$[(0 \le x \le 90, 270 \le x \le 360), (x > 10)], [(0 \le x \le 360), (0 \le x < 10)], [(90 < x < 270), (x > 10)]$}

where,

The form of the combined slope direction and slope angle support set is,

{[(slope direction $_1$), (slope angle $_1$)] ... [(slope direction $_n$), (slope angle $_n$)]}

The values here should be taken as indicative rather than hard-and-fast. Some sites with a slope angle of slightly more than 10^0 may be considered as plateau and similarly, some with a slope angle slightly above or below 10^0 may be considered as having a northerly or southerly exposition. A degree of 'expert' judgement is required to classify a site's exposition.

The exogenous variable, grazing (intensity), was measured using the qualitative support set:

{zero, low, medium, high}

This support set corresponds to the approximate intensity of grazing effect at a site. Intensity is a function of grazing animal numbers and the species of grazer. The RBCLM1 does not provide a means of calculating the value of exogenous variables such grazing intensity. Rather, it requires that intensity be given.

The exogenous variables, fire and cutting, were both measured using a simple binary support set:

{yes, no}

'Yes' corresponds to their having been a fire or cutting event at a site during a given time-step, 'no' to there not having been. Neither fire nor cutting are explicitly modelled as having different potential effect severities i.e. varying effects depending on factors like rainfall etc. The effects of fire and cutting in the Capri Model are to induce instant state transitions on those states where they are known to exert an effect on vegetation.

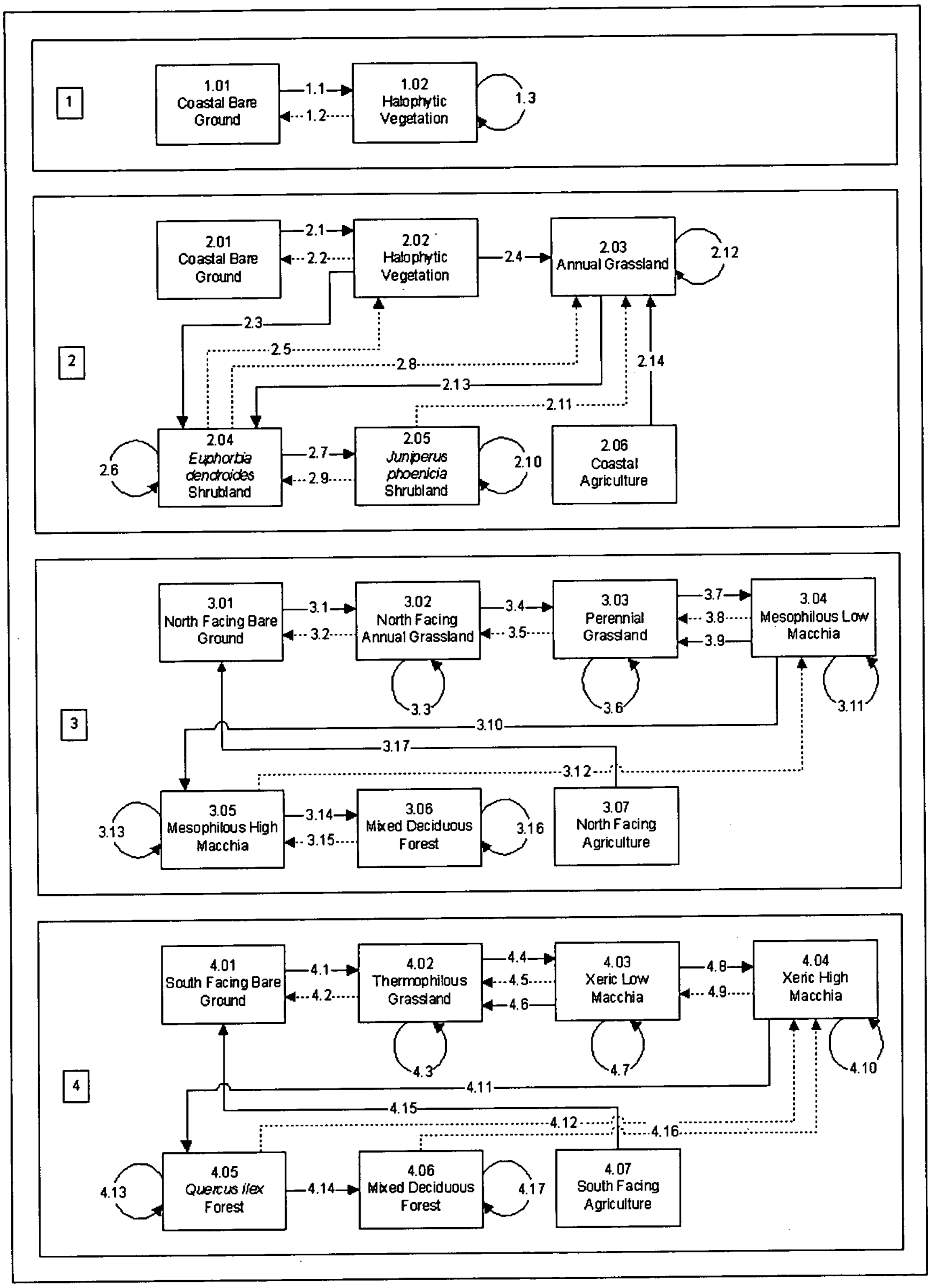

1
1.01 Coastal Bare Ground
1.1
1.2
1.02 Halophytic Vegetation
1.3
2
2.01 Coastal Bare Ground
2.1
2.2
2.02 Halophytic Vegetation
2.4
2.03 Annual Grassland
2.12
2.3
2.5
2.8
2.13
2.11
2.14
2.04 *Euphorbia dendroides* Shrubland
2.6
2.7
2.9
2.05 *Juniperus phoenicia* Shrubland
2.10
2.06 Coastal Agriculture
3
3.01 North Facing Bare Ground
3.1
3.2
3.02 North Facing Annual Grassland
3.3
3.4
3.5
3.03 Perennial Grassland
3.6
3.7
3.8
3.9
3.04 Mesophilous Low Macchia
3.11
3.10
3.17
3.12
3.05 Mesophilous High Macchia
3.13
3.14
3.15
3.06 Mixed Deciduous Forest
3.16
3.07 North Facing Agriculture
4
4.01 South Facing Bare Ground
4.1
4.2
4.02 Thermophilous Grassland
4.3
4.4
4.5
4.6
4.03 Xeric Low Macchia
4.7
4.8
4.9
4.04 Xeric High Macchia
4.10
4.11
4.15
4.12
4.16
4.05 *Quercus ilex* Forest
4.13
4.14
4.06 Mixed Deciduous Forest
4.17
4.07 South Facing Agriculture

Figure 23 RBCLM1 Capri Vegetation Model state transition diagram by vegetation series, *where*, 1 = very close to the sea, 2 = close to the sea, 3 = far from the sea, northerly exposition, 4 = far from sea, southerly or plateau exposition (rectangular nodes represent states, numbered arcs represent state transitions under specific conditions)

4.4.5 Relationship Knowledge

Vegetation series was determined using four rules:

> **IF** 'distance from sea' is equal to 'very near' **THEN** the vegetation series is 'very close to the sea'.
>
> **IF** 'distance from sea' is equal to 'near' **THEN** the vegetation series is 'close to the sea'.
>
> **IF** 'distance from sea' is equal to 'far' **AND** 'exposition' is equal to 'northerly' **THEN** the vegetation series is 'far from the sea, northerly exposition'.
>
> **IF** 'distance from sea' is equal to 'far' **AND** 'exposition' is equal to 'southerly' **THEN** the vegetation series is 'far from the sea, southerly exposition'.

Relationships between vegetation type and environment are represented using the two notions of D-Delta and T-Delta. Figure 23 details the possible states and state transitions that may occur according to vegetation series. Table 31 details the state transitions shown in Figure 23 by series and number describing the environmental conditions necessary for the D-Delta value represented by the succeeding state to be true and the associated T-Delta values under different conditions.

It should be noted that certain transitions, notably those from agricultural vegetation types, require 'abandonment = yes'. This simply means that any site in an agricultural state, if not tended, will automatically undergo the transition listed. No 'abandonment' exogenous variable is included in the model, such transitions occur automatically. Abandonment is included so that the model can be used together with land-use models that can re-allocate agricultural land to natural vegetation (the process of abandonment). This facilitates the modelling of old field succession.

Number	Preceeding State	Succeeding State	Conditions Required
1.1	Coastal Bare Ground	Halophytic Vegetation	Grazing = *any value* (1)
1.2	Halophytic Vegetation	Coastal Bare Ground	Fire = yes (0)
1.3	Halophytic Vegetation	Halophytic Vegetation	Grazing = *any value* (n/a)
2.1	Coastal Bare Ground	Halophytic Vegetation	Grazing = *any value* (1)
2.2	Halophytic Vegetation	Coastal Bare Ground	Fire = yes (0)
2.3	Halophytic Vegetation	*Euphorbia dendroides* Shrubland	Grazing = zero (2); low (2); medium (4)
2.4	Halophytic Vegetation	Annual Grassland	Grazing = high (2)
2.5	*Euphorbia dendroides* Shrubland	Halophytic Vegetation	Fire = yes (0)
2.6	*Euphorbia dendroides* Shrubland	*Euphorbia dendroides* Shrubland	Grazing = medium (n/a)
2.7	*Euphorbia dendroides* Shrubland	*Juniperus phoenicia* Shrubland	Grazing = zero (10); low (15)
2.8	*Euphorbia dendroides* Shrubland	Annual Grassland	Cutting = yes (0)
2.9	*Juniperus phoenicia* Shrubland	*Euphorbia dendroides* Shrubland	Fire = yes (0)
2.10	*Juniperus phoenicia* Shrubland	*Juniperus phoenicia* Shrubland	Grazing = *any value* (n/a)
2.11	*Juniperus phoenicia* Shrubland	Annual Grassland	Cutting = yes (0)
2.12	Annual Grassland	Annual Grassland	Grazing = medium (n/a); high (n/a)
2.13	Annual Grassland	*Euphorbia dendroides* Shrubland	Grazing = zero (2); low (1)
2.14	Coastal Agriculture	Annual Grassland	Abandonment = yes (0)
3.1	North Facing Bare Soil	North Facing Annual Grassland	Grazing = *any value* (n/a)
3.2	North Facing Annual Grassland	North Facing Bare Soil	Fire = yes (0)
3.3	North Facing Annual Grassland	North Facing Annual Grassland	Grazing = high (n/a)
3.4	North Facing Annual Grassland	Perennial Grassland	Grazing = zero (3); low (4); medium (6)
3.5	Perennial Grassland	North Facing Annual Grassland	Fire = yes (0)
3.6	Perennial Grassland	Perennial Grassland	Grazing = medium (n/a); high (n/a)
3.7	Perennial Grassland	Mesophilous Low Macchia	Grazing = zero (2); low (3)
3.8	Mesophilous Low Macchia	Perennial Grassland	Fire = yes (0); Cutting = yes (0)
3.9	Mesophilous Low Macchia	Perennial Grassland	Grazing = high (2)
3.10	Mesophilous Low Macchia	Mesophilous High Macchia	Grazing = zero (4); low (6)
3.11	Mesophilous Low Macchia	Mesophilous Low Macchia	Grazing = medium (n/a)
3.12	Mesophilous High Macchia	Mesophilous Low Macchia	Fire = yes (0); Cutting = yes (0)
3.13	Mesophilous High Macchia	Mesophilous High Macchia	Grazing = medium (n/a); high (n/a)
3.14	Mesophilous High Macchia	Mixed Deciduous Forest	Grazing = zero (10); low (20)
3.15	Mixed Deciduous Forest	Mesophilous High Macchia	Fire = yes (0); Cutting = yes (0)

Number	Preceeding State	Succeeding State	Conditions Required
3.16	Mixed Deciduous Forest	Mixed Deciduous Forest	Grazing = *any value* (n/a)
3.17	North Facing Agriculture	North Facing Bare Soil	Abandonment = yes (0)
4.1	South Facing Bare Ground	Thermophilous Grassland	Grazing = *any value* (1)
4.2	Thermophilous Grassland	South Facing Bare Ground	Fire = yes (0)
4.3	Thermophilous Grassland	Thermophilous Grassland	Grazing = medium (n/a); high (n/a)
4.4	Thermophilous Grassland	Xeric Low Macchia	Grazing = zero (7); low (10)
4.5	Xeric Low Macchia	Thermophilous Grassland	Fire = yes (0); Cutting = yes (0)
4.6	Xeric Low Macchia	Thermophilous Grassland	Grazing = high (3)
4.7	Xeric Low Macchia	Xeric Low Macchia	Grazing = medium (n/a)
4.8	Xeric Low Macchia	Xeric High Macchia	Grazing = zero (5); low (7)
4.9	Xeric High Macchia	Xeric Low Macchia	Fire = yes (0); Cutting = yes (0)
4.10	Xeric High Macchia	Xeric High Macchia	Grazing = medium (n/a); high (n/a)
4.11	Xeric High Macchia	*Quercus ilex* Forest	Grazing = zero (10); low (20)
4.12	*Quercus ilex* Forest	Xeric High Macchia	Fire = yes (0); Cutting = yes (0)
4.13	*Quercus ilex* Forest	*Quercus ilex* Forest	Grazing = low (n/a); medium (n/a); high (n/a)
4.14	*Quercus ilex* Forest	Mixed Deciduous Forest	Grazing = zero (20)
4.15	South Facing Agriculture	South Facing Bare Ground	Abandonment = yes (0)
4.16	Mixed Deciduous Forest	Xeric High Macchia	Fire = yes (0); Cutting = yes (0)
4.17	Mixed Deciduous Forest	Mixed Deciduous Forest	Grazing = *any value* (n/a)

Table 31 RBCLM1 Capri Vegetation Model state transition details (';' means logical 'or', ',' means logical 'and', numbers in brackets next to condition values indicate associated T-Delta values where 'n/a' indicates a stable state)

4.4.6 Model Results

The Capri Vegetation Model was run under various conditions to verify and assess the utility of the RBCLM1 system. This section will present the results from two sets of runs, both of which were aspatial i.e. only one site simulated at a time for multiple time-steps. Both sets of runs used a site following the 'far from the sea, southerly exposition' vegetation series. The first set of runs kept grazing intensity at a constant zero whilst increasing the frequency of fire disturbances from one run to the next. The second set of runs kept grazing intensity at a constant 'high' value whilst increasing the frequency of fire disturbances in the same way. Mediterranean vegetation tends to be resilient to fire, the disturbance only temporarily altering community composition and structure (Trabaud 1994). However, increasing fire frequency free from grazing is known to cause increasingly severe degradation in Mediterranean vegetation (Trabaud 1991, Blondel and Aronson 1999). Under fire and grazing operating together normal post-fire recovery can be prevented and vegetation degradation can occur (Naveh 1990). The two Capri simulation experiments were designed to see if the model and system could reproduce these known patterns.

4.4.6.1 Experiment 1 – No Grazing, Increasing Fire Frequency

	Run Number			
	1	2	3	4
Starting State:	4.01	4.01	4.01	4.01
Constants: Distance from Sea: Exposition:	Far South	Far South	Far South	Far South
Exogenous Variables: Cutting: Fire: Grazing:	No No Zero	No Yes – *once per 10 time-steps.* No – *all other time-steps* Zero	No Yes – *once per 5 time-steps.* No – *all other time-steps* Zero	No Yes *–every time-step.* Zero
Run Length (time-steps):	100	100	100	100

Table 32 Conditions for RBCLM1 Capri Vegetation Model experiment 1

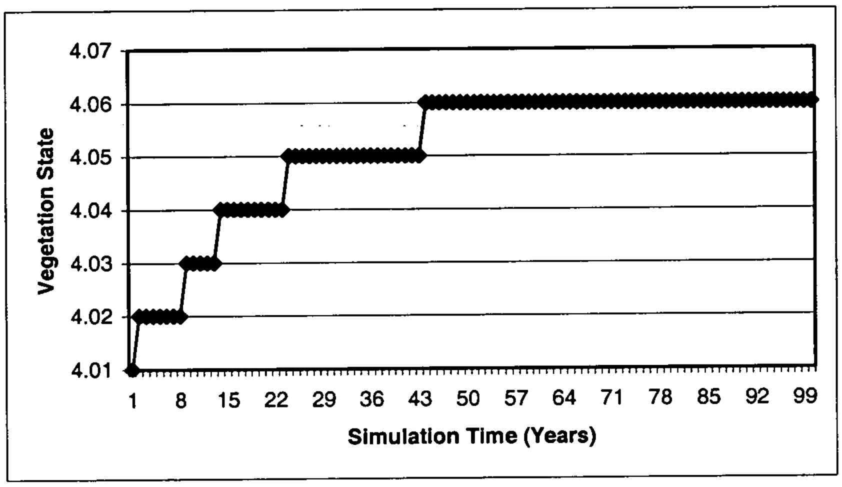


Figure 24 Results for experiment 1 run 1 – no fire

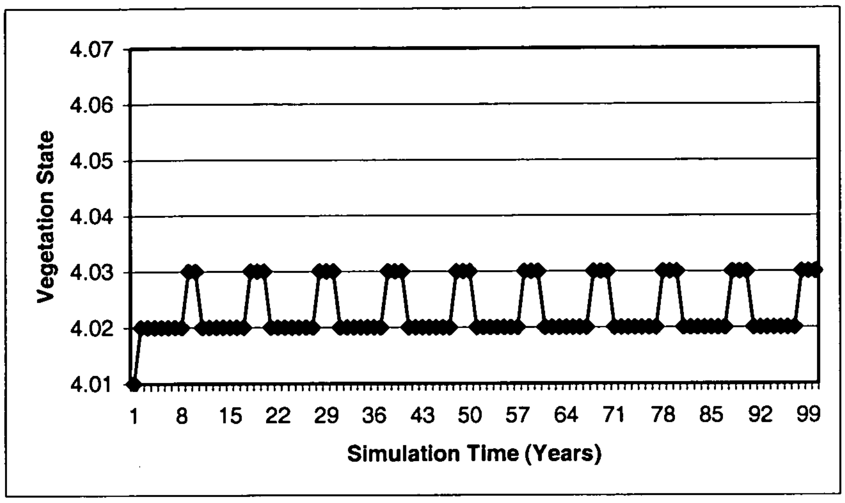


Figure 25 Results for experiment 1 run 2 – fire MRI = 10

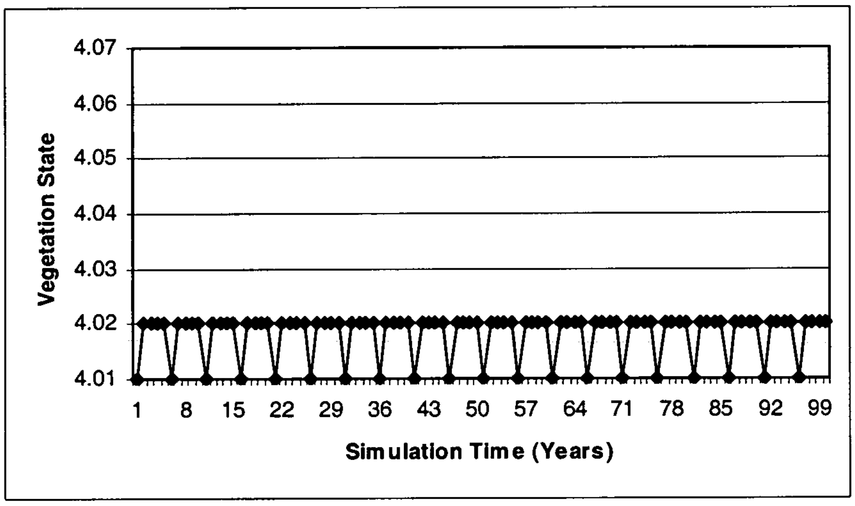


Figure 26 Results for experiment 1 run 3 – fire MRI = 5

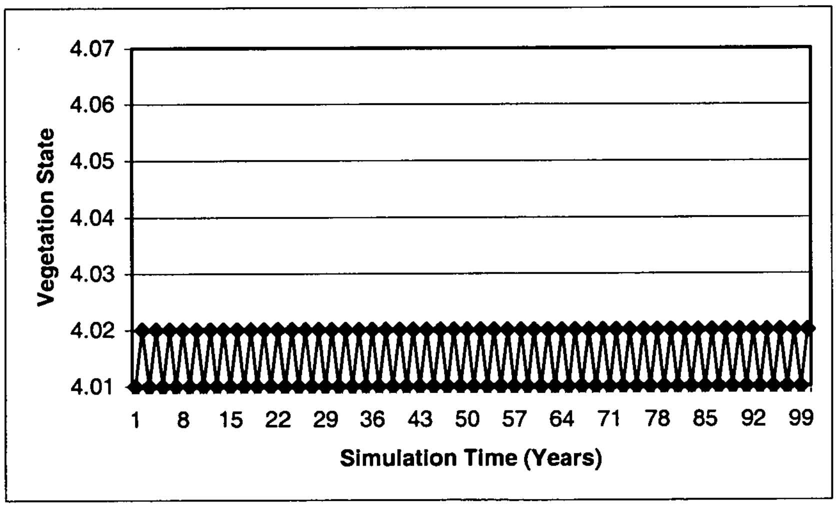


Figure 27 Results for experiment 1 run 4 – fire MRI = 1

The results for experiment 1 show that, under constant conditions of zero grazing intensity and no fire disturbance (Figure 24), vegetation grows steadily from bare ground to a mixed deciduous forest cover state (4.06) in around 45 years. It takes different lengths of time to develop through the states preceding the formation of the deciduous forest cover, only taking around 1 year to change from bare ground (4.01) to thermophilous grassland (4.02) but taking longer to develop from high macchia (4.04) to *Quercus ilex* forest (4.05) and from *Quercus ilex* forest to mixed deciduous cover. Under a fire disturbance regime with a fire MRI of 10 years, the vegetation is prevented from developing into a forest cover, instead oscillating between thermophilous grassland and low machhia (4.03). As the MRI decreases (fire frequency increases) to 5, vegetation starts to oscillate between bare ground and thermophilous grassland, mostly staying in the grassland state. This oscillation is maintained at an MRI of 1 year but the length of time spent in the grassland state decreases such that equal time is spent in both states. In summary, the degree of vegetation degradation increases with increasing fire frequency. However it must be noted that the manner in which vegetation degradation is modelled can be subjected to criticism. These will be detailed in section 4.5.

4.4.6.2 Experiment 2 – High Grazing, Increasing Fire Frequency

	Run Number			
	1	2	3	4
Starting State:	4.01	4.01	4.01	4.01
Constants:				
Distance from Sea:	Far	Far	Far	Far
Exposition:	South	South	South	South
Exogenous Variables:				
Cutting:	No	No	No	No
Fire:	No	Yes – *once per 10 time-steps.* No – *all other time-steps*	Yes – *once per 5 time-steps.* No – *all other time-steps*	Yes –*every time-step.*
Grazing:	High	High	High	High
Run Length (time-steps):	100	100	100	100

Table 33 Conditions for RBCLM1 Capri Vegetation Model experiment 2

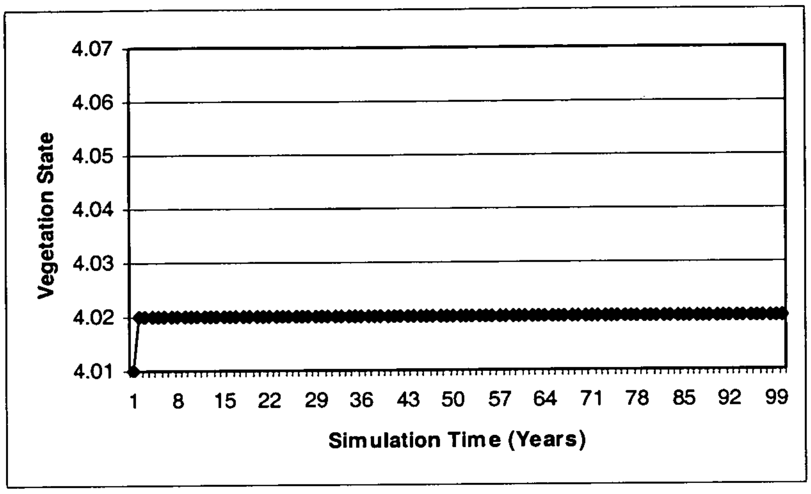


Figure 28 Results for experiment 2 run 1 – no fire

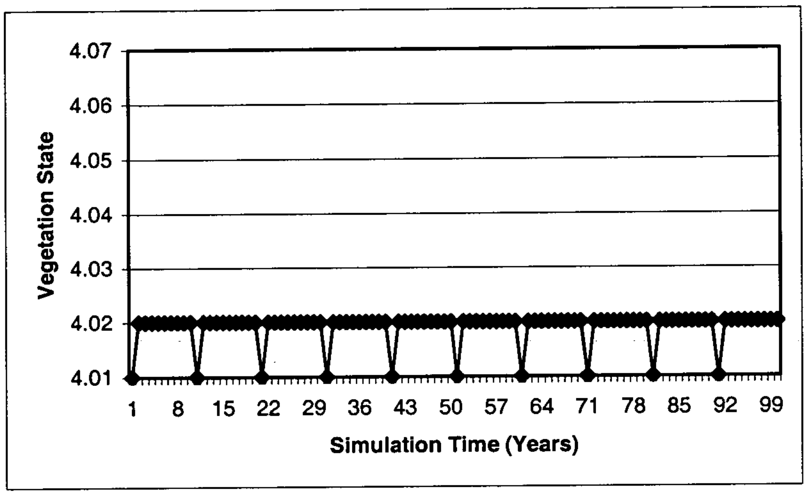


Figure 29 Results for experiment 2 run 2 – fire MRI = 10

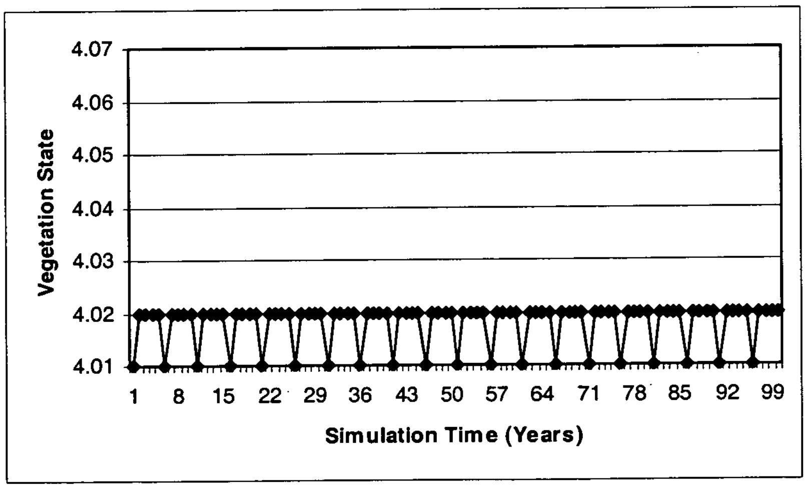


Figure 30 Results for experiment 2 run 3 – fire MRI = 5

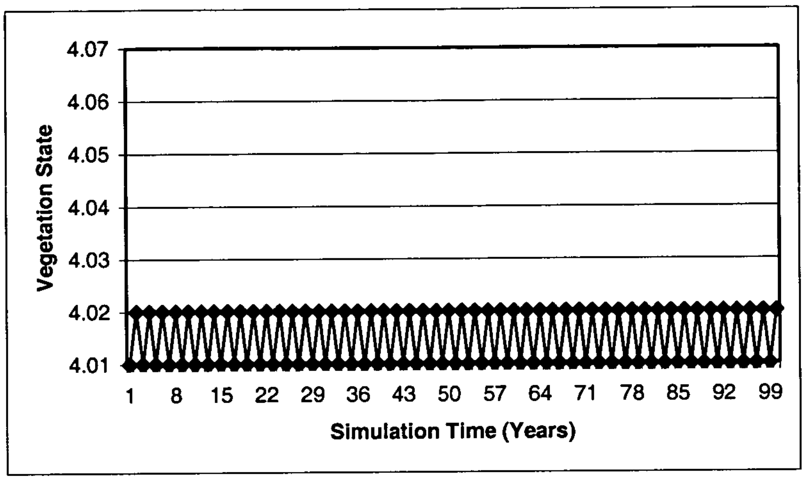


Figure 31 Results for experiment 2 run 4 – fire MRI = 1

The results for experiment 2 show that the effects of high intensity grazing alone (no fire disturbance) are sufficient to prevent vegetation in series 4 from developing into forest cover. Under such conditions, vegetation develops into thermophilous grassland and is then maintained in this state. The degradative effects of high intensity grazing and fire together are such that under a relatively infrequent fire disturbance regime (MRI = 10 years) vegetation begins to oscillate between bare ground and thermophilous grassland. As the MRI decreases further, the length of time spent in the thermophilous grassland state decreases until, just as under experiment 1, when the MRI is equal to 1 year, the vegetation oscillates between the two states, spending an equal amount of time in each. Grazing and fire are shown to interact strongly in the Capri model although the model behaviour can be criticised as unrealistic. The results will be critiqued next.

4.5 Discussion

4.5.1 Modelling Vegetation Dynamics – RBCLM1 System Utility

The results of the experimental runs with the Capri Vegetation Model demonstrate that it possible to represent and reason with knowledge fragments concerning the way in which vegetation changes at the community-level based upon a separation of direction and rate of change (D-Delta and T-Delta). The simulation results demonstrate that the RBCLM1 system is capable of replicating some of the dynamics that would be expected under conditions of increasing fire frequency and of increasing fire frequency in combination with grazing (Le Houérou 1981, Naveh 1990).

With respect to the dynamics characterisation of Burrows (1990), the RBCLM1 System is capable of replicating types 1, 2 3, 5 and 6:

- *Colonization and sequential replacement (Burrows dynamic type 1):*
  Bare ground can be represented as a 'vegetation' state. The Capri model demonstrates that is possible to simulate the dynamic of colonisation leading to a state change from bare ground to some other state (e.g. halophytic grass). In addition the sequence of states produced by the Capri experiments simulates the effects of sequential replacements.
- *Direct replacement following the disturbance of established vegetation (Burrows dynamic type 2):*
  In several of the simulation runs (see Figures 26, 27, 29, 30 and 31) the vegetation state preceding a disturbance rapidly re-establishes post-fires demonstrating the Capri model's ability to simulate direct replacement dynamics.
- *Cyclic replacement in response to endogenous or exogenous influences (Burrows dynamic type 3):*
  Simulation runs such as run 2 from experiment 1 (Figure 25) show cyclical vegetation change in response to an endogenous factor – fire with a 10 year MRI.
- *Fluctuating replacements in response to exogenous influences (Burrows dynamic type 4):*
  The temporal and structural resolution of vegetation is too coarse for this dynamic to be modelled. Such fluctuations normally occur on a time-scale of a year or less (e.g. seasonal vegetation changes) and may not involve sufficient compositional or structural change to warrant being modelled as a state transition. It should be noted that an inability to reproduce such dynamics does not necessarily adversely affect the RBCLM1 system's utility as a vegetation modelling system. Such fluctuations may not be important enough to include in a model, particularly if the nature of the fluctuations do not influence the overall direction and form of longer-term dynamics.
- *Vegetation maintained by frequent or continual disturbance (Burrows dynamic type 5):*
  Experiment 2 run 1 (Figure 28) demonstrates that the RBCLM1 can model vegetation that is maintained by continual disturbance. In this case it is thermophilous grassland maintained by high grazing.
- *Vegetation in relative equilibrium for a time (Burrows dynamic type 6):*
  Runs like experiment 1 run 3 (Figure 26) and experiment 2 runs 2 and 3 (Figures 29 and 30) demonstrate the ability of the RBCLM1 to model vegetation remaining at a steady equilibrium state, only punctuated by fire disturbance.

- *Long-term, gradual change in response to autogenic or allogenic influences (Burrows dynamic type 7):*
  State-based vegetation modelling characterises vegetation dynamics using a fixed set of vegetation types and transitions. Under long-term environmental change the way in which species interact and the available species pool are open to change (Glenn-Lewin and van der Maarel 1992). This is likely to render the classification of states and transitions between them invalid as species interactions produce new states and new transitions. Under the novel conditions of long-term environmental change a deeper, mechanistic understanding of vegetation is likely to be required for effective prediction. The RBCLM1 cannot therefore be said to be capable of modelling this dynamic type.

However there are a number of possible limitations to the RBCLM1 approach. Overall the influence of increasing fire frequency (no grazing) is correctly modelled as preventing the formation of mature shrub and forest vegetation types. The interaction between fire and grazing is shown to cause more severe vegetation degradation, the expected qualitative outcome. However the results under the highest frequency fires (MRI = 1, Figures 27 and 31) are unrealistic - sufficient vegetation growth to fuel fresh fire events would not be present after only one year of post-fire recovery. Without adding more states (e.g. 'thermophilous grass freshly burned', 'thermophilous grass ready to burn') or modelling the dynamics of properties like biomass, the RBCLM1 cannot prevent such unrealistic disturbances from happening. In the RBCLM1 all instances of each vegetation state are modelled as being functionally identical in terms of their response to environmental conditions or disturbance factors. This may result in important dynamics (both in the direction and rate of change) being difficult or impossible to capture with implications for behavioural soundness. For example, although vegetation may be classified as being thermophilous grassland after one year of post-fire growth, it should not be treated as a 'mature' example of the vegetation type. The vegetation should be treated as having insufficient cover and biomass to fuel a fire and therefore essentially non-flammable. However variation in functional response depending on vegetation property values cannot be represented by the RBCLM1 system.

In addition, the RBCLM1 'priority of effect' based disturbance handling system may also affect behaviour. It is possible in reality that more than one disturbance factor may influence and affect the structure and dynamics of a patch of vegetation per year with important consequences for state, direction of change and timing of change. For example, it is possible that grazing may occur for certain months of year, influencing vegetation structure and dynamics, and that a fire may occur after or between grazing periods and also affect the vegetation. However, the RBCLM1 system cannot represent or reason about multiple disturbance factor situations.

As well as multiple disturbance effects per time-step being possible, the combined effect of multiple disturbances on vegetation state, direction of change and rate of change may be qualitatively different from the effect of each disturbance in isolation. If a disturbance affects a patch of vegetation but doesn't immediately alter its state it may have changed certain structural properties and in doing so, potentially altered its direction and timing of change. For example, if grazing occurs on new shoots soon after a fire, the combined effect may be to prevent re-growth of the vegetation. Different instances of each vegetation type will, in reality, be different structurally. Some of these differences may have important consequences for dynamics.

In addition, the RBCLM1 system provides no means of modelling the potentially dynamically important processes and feedbacks that exist between vegetation and environment such as with respect to litter fall, decomposition and soil nutrient status etc. In an RBCLM1 model, vegetation does not affect the environment component of a site. The influences are purely one way, making it impossible for the model developer to use the RBCLM1 system to model situations where vegetation-environment coupling is dynamically important. This is the third problem with the RBCLM1 system. It should be noted however that modelling one-way influences is an improvement over previous state transition approaches, which cannot handle changes in factors like precipitation during a run.

To address the possible problems of variation in functional response within a vegetation type and the problems associated with the RBCLM1 disturbance handling system, a more precise way of representing and reasoning about the state of vegetation is required. Two main options are available for doing so:

1. To reduce the length of the time-unit used.
2. To use a more detailed representation of vegetation structure (i.e. a set of variables) to model change in vegetation state.

Option (1) is not really feasible with respect to purely state-based modelling of vegetation dynamics using available knowledge. Community-level dynamics, other than those caused by instant effect disturbances such as fire, tend to be relatively slow. Using the types of vegetation classification systems that ecologists do, changes between states tend to occur on a scale of years to decades, setting a limit on the temporal resolution that can be employed. To model vegetation under option (1) requires that option (2) also be adopted so that smaller scale dynamics can be represented. In the next two chapters different ways of pursuing option (2) will be explored with the aim of producing more capable modelling systems. In addition, methods for representing the environment component using state and intermediate variables and for representing relationships between vegetation and environment based on available knowledge will also be explored.

4.5.2 Knowledge Representation

The RBCLM1 KR scheme is capable of representing linguistic value and relationship knowledge concerning the direction of vegetation change and quantitative (integer) value and relationship knowledge concerning the rate of vegetation change. Using rules to express the conditions under which particular directions and rates of change are known to occur is a form of value correspondence – correspondence between combinations of influencing variable values and single D-Delta and T-Delta values. Based upon the results of the Capri KA exercises this KR scheme is structurally very similar to the knowledge that ecologists have available regarding state-based vegetation change. Such structurally similarity should facilitate model design and the use of ecological knowledge. In addition, the use of vegetation series to group states and transitions should also reduce the complexity of rule construction, further facilitating model design.

However, the KR scheme does not explicitly represent the structure of models, the support sets of variables or the direction of relationships between variables. For system flexibility, modularity and ease of update / revision reasons this is undesirable. If more complicated models are to be considered it would be better to maintain separation between knowledge about possible values (support sets) and knowledge about how those values change (relationships). Developing a cleaner separation between declarative representation of model structure (variables and what influences what) and possible variable values would facilitate non-simulation oriented reasoning tasks. For example there is currently no easy way to query any RBCLM1 model to get information about the range of possible vegetation states and transitions, a feature that could be useful for decision-support oriented modelling where managers wish to interrogate model structure.

For constants and exogenous variables, where support sets are not detailed anywhere within an RBCLM1 model, this means that models are incomplete without external documentation. If such documentation is unavailable this may prevent effective use and understanding of an RBCLM1 model. Again, there is a need for a separate representation of variable support sets to ensure that RBCLM1 models are complete within themselves.

4.5.3 Reasoning and Simulation

Reasoning about change in vegetation state is achieved based upon reasoning about direction of change, time required to change state and time in state (under current conditions). Vegetation state is an example of a variable measured using an unordered linguistic support set. The RBCLM1 demonstrates that rules for determining D-Delta and T-Delta specific to each set of conditions can be used to effectively reason about change in this type of variable. Each transition is unique, both in form (preceeding and succeeding state values) and conditions required, making it impossible to construct a general method for reasoning about change analogous to the way arithmetic or algebra provide general reasoning methods for numerical variables. Case-specific reasoning has to be employed.

The averaging algorithm used to calculate and update T-Delta appears capable of reproducing the types of behaviour that would be expected under various conditions. However individual-model validation is required. It may be that T-Delta varies in a non-linear way with changing conditions, requiring either a more sophisticated averaging algorithm or case-specific treatment of each set of conditions. Further testing against actual field data is required to determine whether such changes are required.

RBCLM1 models are wholly deterministic although stochasticity of some form could potentially be included through including stochastically varying exogenous variables in a model. Pure determinism was chosen as an approach partly for reasons of parsimony and partly because the ecological knowledge upon which the RBCLM1 is based was found to be deterministic and concerned with spatial and temporal scales in Mediterranean vegetation where the effects of stochasticity can be largely ignored. The RBCLM1 models dynamics based on knowledge about how vegetation responds and changes under known conditions. Outwith these conditions it is not clear how vegetation will change and as such the knowledge upon which RBCLM1 models are based can be viewed as having limited applicability. What the bounds to this applicability are will vary according to the vegetation and environment under consideration and require investigation to identify before applying the RBCLM1 to real-world problems.

Chapter 5 Modelling Vegetation Dynamics II – A Rule-Based Community Attributes and Properties Approach

5.1 System Basics

5.1.1 Introduction

The RBCLM1 system treats all instances of each vegetation type as being functionally very similar and cannot model the effects of more than one disturbance occurring in a year. In addition the RBCLM1 cannot represent feedbacks between vegetation and environment. Such relationships can be important in determining vegetation dynamics. Two main options exist to resolve the first problem – reduce the time-unit length and use a more detailed representation of vegetation structure. The second modelling system to be developed during this investigation, the Rule-Based Community-Level Modelling 2 (RBCLM2) system, will explore both methods. To solve the environmental feedback problem the RBCLM2 will represent the environment using state and intermediate variables as well as exogenous variables and constants.

5.1.2 System Ontology and Model Structure

As well as belonging to a discrete type, vegetation can be thought of in terms of attributes and properties. Vegetation attributes can be viewed as representing the average behavioural characteristics of vegetation types e.g. palatability. Vegetation properties can be viewed as the quantities that describe the state of vegetation over time. Using the ontology of vegetation types, attributes and properties the RBCLM2 can represent vegetation structure in more detail than in the RBCLM1. By reasoning about changes in vegetation state based upon vegetation property values the RBCLM2 aims to more accurately model vegetation dynamics. Importantly, to utilise available knowledge types, the RBCLM2 is designed to permit quantitative, qualitative and linguistic values and relationships to be used.

The RBCLM2 represents the world as a single patch of land, a site, with each site composed of a vegetation component and an environment component (see Figure 32).

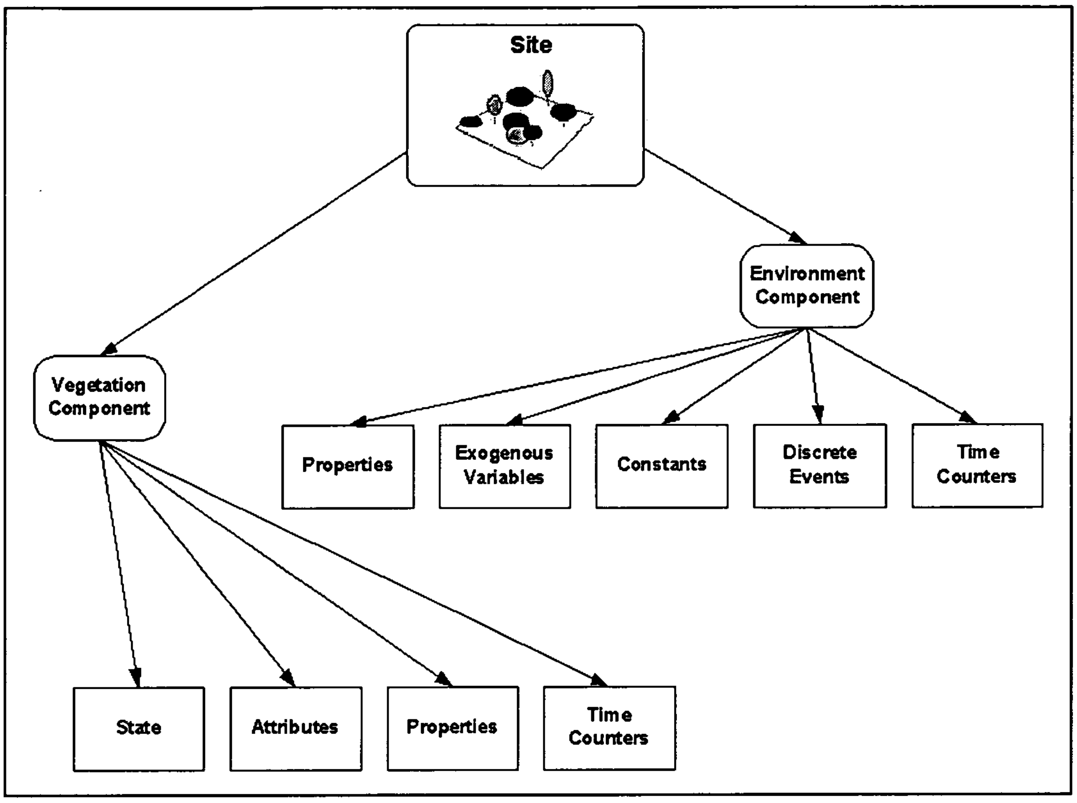


Figure 32 Informal RBCLM2 system ontology (arcs represent 'has a' relationships, rectangular boxes represent single model variables or sets of variables)

A site's vegetation component is represented using four concepts:

- *State:*

  One of the main state variables of interest in any RBCLM2 model is vegetation state. Vegetation at a site is represented as belonging to a discrete state or vegetation type with different state values linked by transitions to form vegetation series. Changes in vegetation state can be made dependent on any other model variables as desired and appropriate to the knowledge available and model purpose.

- *Attributes:*

  The way in which a site's vegetation properties change over time depends upon how that site's vegetation behaves. Vegetation attributes, a type of constant, are used to capture the basic behavioural or functional characteristics of each vegetation type. Vegetation attributes can be used to calculate the values of other model variables, most particularly vegetation properties and the severity of disturbance effects.

- *Properties:*

  Along with the variable vegetation state, vegetation properties are used to characterise how vegetation is structured and changes over time. Vegetation properties are either state or intermediate variables whose values are specific to the vegetation of each site e.g. canopy height, actual growth rate. Vegetation property values tend to be dependent on the value of attributes, other vegetation properties and environmental component variables.

- *Time Counters:*

  The RBCLM2 also employs a set of time counter variables. State variable time counters are used to keep a track of how long non-quantitative variables have been at their current value to assist reasoning about changes in value. They serve the same purpose as the 'T-In' variable associated with vegetation state in the RBCLM1. Intermediate variable time counters can be used in various ways for calculation and update of vegetation and environmental properties and to represent temporal aspects of vegetation dynamics e.g. time since last fire.

Each site's environment component is represented using five concepts:

- *Properties:*

  Environmental properties are the environmental equivalent of vegetation properties. They represent the state and dynamics of each site's environment and, being either state or intermediate variables, can vary with time. Example properties include soil moisture and soil nitrogen level etc. They are included in the RBCLM2 to provide a means of modelling vegetation-environment feedbacks and can be made dependent on any other variable type.
- *Exogenous Variables:*

  Exogenous variables are used to represent those environmental influences whose values may change during the course of a simulation run but whose values are not determined within an RBCLM2 model e.g. amount of precipitation, number of grazing animals, temperature etc.
- *Constants:*

  Constants are used to represent those environmental influences whose values remain constant for a given site over the course of a simulation run e.g. altitude, distance from sea etc.
- *Discrete Events:*

  Discrete events can be viewed as a form of binary exogenous variable where 0 = no and 1 = yes. Discrete events may trigger disturbance events but are not disturbances in themselves e.g. a spark may trigger a fire but will not cause damage itself. The effects of disturbance events are represented using environmental properties such as fire severity.

- *Time Counters:*

  Serve the same purpose as vegetation time counters but applied to environmental properties.

The division between properties (vegetation and environment) and attributes (for vegetation) or constants (for environment) was chosen to highlight the difference between dynamic and static model variables. This was felt to be important for capturing the notion of vegetation properties being determined by attributes and environmental properties being determined by constants.

Vegetation state is a linguistic variable, discrete events are binary and time counters are integer. All other variables can be quantitative, qualitative or linguistic. Mixtures of support sets can be used. Prolog "if ... then" rules represent knowledge detailing how each RBCLM2 model's state variables (vegetation state, vegetation properties, environmental properties and both vegetation and environmental time counters) change from one value to another under given sets of conditions. Each state variable 'update rule' states the nature of the possible change in value and the necessary conditions that must be met for the change in value to occur. State variable update rules may use, amongst other variables, the values of intermediate variables as conditions, so another set of 'if...then' rules, intermediate variable determination rules, are used to represent knowledge regarding their values.

The RBCLM2 system was designed to be more flexible than the RBCLM1, which enforced a strict method for reasoning about change in state (D-Delta, T-Delta and T-In). At the time of design this was felt necessary to permit the range of available knowledge to be represented. Although the model components outlined above can be included in an RBCLM2 model there is no need for all components to be included. The only strictly necessary elements are vegetation state and some means of reasoning about change in state. This may, but not necessarily, be T-Delta and T-In based. Other options include reasoning about change in vegetation state based on vegetation property values and disturbance events. For example, ecologists often classify vegetation and reason about change in vegetation type based on criteria such as height and cover. However most RBCLM2 models will be composed of vegetation state, some number of attributes and properties and an environmental component of some description.

The RBCLM2 provides a set of mandatory representational constructs and a set of optional constructs ('convenience predicates') for the model developer to use. It does not enforce a particular method for reasoning about change in non-quantitative state variables or for calculating intermediate variables. The representational constructs provided can be used in various ways as best suits the knowledge available and purpose of the model being designed. The RBCLM2 reasoning system enforces less than the RBCLM1 reasoning system, rather it operates more like a simulation control algorithm. As will be seen this places the onus of ensuring that discrete state variables change values in appropriate directions at appropriate rates in the hands of the model developer. The implications of designing the RBCLM2 in this manner will be explored later.

5.1.3 Simulation Overview

The RBCLM2 does not have a hard-wired time-unit – the time-unit should reflect model purpose and available knowledge. A monthly time-unit is however appropriate to the scale at which the system represents and reasons. One time-step is equal to one time-unit in an RBCLM2 model. At each time-step during a simulation update rules are applied to each environmental property state variable then each vegetation property state variable. Once these variables have been updated, vegetation state (type) is then updated by means of a state transition rule, another form of update rule. After vegetation state has been updated, all time counter state variables are also updated. During the process of state variable updating the values of constants and exogenous variables can be obtained and used through looking-up a simulation datafile. The values for intermediate variables are not calculated once per time-step, rather they are calculated 'on-the-fly' as required by other rules e.g. if a biomass (state variable here) update rule requires the value of growth rate (intermediate variable here), it simply fires the appropriate intermediate variable determination rule. This does not however violate the 'calculate rate then update state' model computation rule as each time-step's on-the-fly intermediate calculations are made based upon the previous time-step's state variable values or the current time-step's exogenous variable values. The freshly updated state variable values are not asserted until the end of the time-step after all calculation has been completed. Intermediate variable determination rules may themselves use other intermediate variables as conditions, again simply by firing their determination rules as necessary

5.1.4 System Architecture

The RBCLM2 system is composed of seven logical modules that can be grouped into four as detailed in Figure 33. Each module will be described in the following sections.

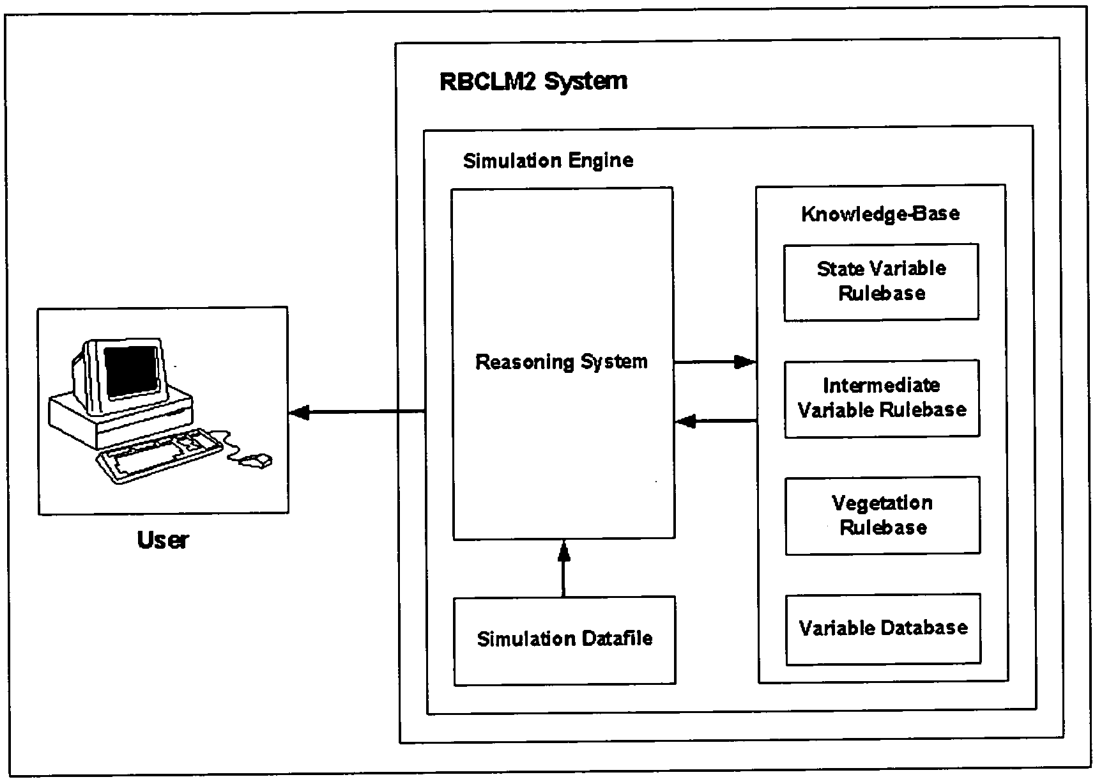


Figure 33 RBCLM2 system architecture and operation (arcs represent information flows resulting from Prolog predicate calls)

5.1.4.1 Simulation Engine

The RBCLM1 reasoning system fulfilled the role of controlling simulation runs and of updating state variables. The RBCLM2 uses 2 modules to perform these tasks. The simulation engine is in charge of the mechanical details of initialisation, execution and simulation time keeping, run termination and output. The reasoning system is in charge of updating state variables. Different simulation engine versions are required for use in different 'simulation modes' – stand-alone or within an external environment. Currently there are three implemented simulation engines:

- Stand alone simulation engine (SASE) for use when RBCLM2 models are executed directly using SICStus Prolog.
- ModMED simulation engine (MMSE) for use when RBCLM2 models are used within the ModMED Landlord simulation environment (Mazzoleni and Legg 2001).
- MODULUS simulation engine (MOSE) for use when RBCLM2 models are used within the MODULUS Geonamica© simulation environment (Engelen *et al.* 2000).

MMSE, MOSE and SASE all contain the same basic functionality but are tailored to meet the needs of the embedding simulation environment e.g. SASE displays results directly to the screen whereas MMSE and MOSE do not. Section 5.3 describes the operation of the simulation engine.

5.1.4.2 Reasoning System

The RBCLM2 reasoning system, along with the simulation engine, constitutes the operational core of the system. It is hardwired within the system irrespective of simulation mode. The reasoning system has the responsibility of updating model state variables in the correct sequence, determining the type of disturbance event that is occurring at a given site during the current time-step (if any) and providing various data manipulation features common to all simulation modes.

5.1.4.3 Knowledge Base (KB)

Each KB is a model of a vegetation system in the same way as the RBCLM1 KB module. The RBCLM2 KB is structured differently however - from up to four different file components (detailed below). It is possible to build a library of KB components for different vegetation systems or for the same vegetation system. The user need only 'plug' in the appropriate module without worrying about compatibility or the effect on other modules.

State Variable Rule-Base

Contains rules detailing how vegetation state may change from one type to another, how vegetation and environmental properties may change value and how time counter state variables are to be updated. This component must be included in all RBCLM2 models as the state variable 'vegetation state' is mandatory.

Intermediate Variable Rule-Base

Contains rules for determining intermediate variable vegetation and environment property values including the values of disturbance factors from the values of other variables within a model. This component may or may not be used in a particular model depending on whether intermediate variables are employed or not.

Vegetation Database

Contains facts concerning the attributes of each vegetation state. This component is optional depending on whether a model uses vegetation attributes or not.

Variable Database

Contains information on variables that is used when converting from qualitative to quantitative measurement scales (or vice versa) for input/output (I/O) purposes when the RBCLM2 is being used within simulation environments such as Landlord.

5.1.4.4 Simulation Datafile

The simulation datafile component provides initial state variable values, values for all model constants and values for all exogenous variables (including discrete events) for the duration of each simulation run. The simulation datafile component may either be a file constructed by the model developer before a run or a set of dynamic predicates held in memory and asserted and retracted during a run in response to prompting from an external simulation environment. Prolog provides the facilities to write self-modifying programs through the use of built in predicates called assert and retract. Typically, if the RBCLM2 is being used in stand-alone mode, the simulation datafile component will be a pre-constructed file and, if being used within a larger simulation environment the simulation datafile component will be a set of dynamic predicates.

5.2 Knowledge Representation

5.2.1 Knowledge Representation Scheme – Introduction and Notation

Table 34 details the arguments used by the RBCLM2 KR scheme.

Argument Name	**Typical Value**	**Prolog Term Classification**	**Description**
Env_details	[(soil_fertility,low)]	Data structure (list)	Tuples representing environmental property state variable names and values.
Name	soil_fertility	Constant (atom)	Variable name.
Priority_value	1	Constant (number)	The priority value associated with a disturbance event type.
Site	1	Constant (number)	Site number (or name).
State	halophilous_grass	Constant (atom)	Vegetation type during the current time-step.
State_next	juniperus_phoenicia	Constant (atom)	Vegetation type for the next time-step.
T	28	Constant (number)	The simulation time when a particular event will occur or input variable value will apply. Used in the stand alone Simulation Data component.
Time_details	[(time_in_state,305),(time_since_fire,100)]	Data structure (list)	Tuples representing time counter state variable names and values.
Value	low	Constant (atom)	The value of a (non-state) variable.

Argument Name	Typical Value	Prolog Term Classification	Description
Value1	medium	Constant (atom)	The current value of a state variable.
Value2	high	Constant (atom)	The value of a state variable for the next time-step.
Veg_details	[(canopy_height,205.60)]	Data structure (list)	Vegetation property state variable names and values for the site being simulated

Table 34 RBCLM2 KR scheme argument notation

5.2.2 Knowledge Base: State Variables Rule-Base

5.2.2.1 Introduction and Usage

All RBCLM2 models must have the state variable `veg_state` (vegetation state) and may have any number of other state variables representing vegetation and environmental properties or time counters. The inclusion of further state variables is the basis to the RBCLM2 more accurately modelling vegetation dynamics than the RBCLM1.

Excepting vegetation state, which is treated separately, state variable names and current values are represented in the RBCLM2 by Prolog data structures called lists. Lists are arbitrarily long data structures composed of many elements. The list structures used to represent RBCLM2 state variables are composed of elements containing both the name of a variable and its value. One tuple is used per variable and one list per site. The common structure for each state variable list structure is as follows:

[(Name $_1$, Value $_1$), (Name $_2$, Value $_2$) ... (Name $_n$, Value $_n$)]

Environmental property state variables are represented by the list `Env_details`, vegetation property state variables by `Veg_details` and time counter state variables by `Time_details`. Vegetation state is also a state variable but is represented using an atom rather than a list as it will only have one value for each site at any given time whereas there may be multiple environmental or vegetation property and time counter state variables for every site.

An example of the Veg_details list:

[(canopy_height, 200), (cover, low)]

Update rules are used to represent knowledge on how environmental, vegetation and time counter state variables may change value under the influence of other model variables. State variables are updated once per time-step through checking and, if required, changing the current value as determined by the appropriate set of update rules. There are 4 predicates that must be used in RBCLM2 models for representing update rules:

- `veg_state_update:`

  Represents knowledge on how vegetation type changes.
- `env_property_update:`

  Used to represent how environmental property state variables change.
- `veg_property_update:`

  Represents how vegetation property state variables change.
- `time_counter_update:`

  Used to represent how any time counters used by a model change.

There are no strict rules enforced within the RBCLM2 concerning how state variables should be updated. However, during the course of the research that led to the development of the RBCLM2 various common structures and ways of representing knowledge about how community-level properties change value emerged. These are detailed here to illustrate some, but not all, of the ways that different fragments of ecological knowledge can be represented using update rules.

5.2.2.2 Updating State Variables with Quantitative Support Sets

Quantitative state variable updating can be achieved through the use of algebraic equations embedded in 'if ... then' rules. If a quantitative state variable is only dependent on quantitative variables then the formulation of a suitable expression for updating is methodologically straightforward. If however a quantitative state variable is dependent on one or more non-quantitative variables in addition to its own quantitative value from the previous time-step, then formulating an expression is less straightforward. The state variable value should be updated in a way that is specific to each combination of non-quantitative influencing variable values. For example, taking a quantitative state variable, X the following simple rule set could be used to determine change:

IF A = low **THEN** $X_{t+1} = X_t + 5$

IF A = medium **THEN** $X_{t+1} = X_t + 8$

IF A = high **THEN** $X_{t+1} = X_t + 10$

This particular solution approximates an asymptotic relationship and, although simple, may suffice for some modelling purposes. The solution method essentially involves mapping the qualitative support set of variable A onto a set of quantitative values ({low, medium, high} $\rightarrow$ {5, 8, 10}) such that each element in the quantitative value set represents the average or approximate numerical change in X caused by the corresponding (qualitative) value of A over the time-step t + 1. The method is an example of a value change correspondence between the magnitude of change in one variable (here, X) and the value of another variable (here, A).

If a quantitative state variable is influenced by more than one non-quantitative variable this solution method can be extended by mapping every combination of influencing variable values onto a single quantitative value set. The resulting quantitative value for the combination of values at any given time-step can then be applied to update the state variable value using a standard algebraic operator. Each combination of values for the set of variables $Q_1 \ldots Q_n$ may have a different effect on the value of X_{t+1} and so each combination may, in theory, require a separate rule for updating of X. Given *n* possible variables each measured using a support set with *m* elements gives a number of possible rules equal to the number of possible combinations:

$$C = m_1 \times m_2 \ldots \times m_n$$

Where,

C = the number of different combinations of values for all influencing non-quantitative variables – equal to the number of rule required to specify how to update the state variable.

m_i = the number of elements in the support set of variable Q_i.

This may lead to a large number of rules being required to update quantitative state variables that are dependent on non-quantitative influences (e.g. 3 non-quantitative influencing variables, each with 3 element support sets = 3 x 3 x 3 = 27 rules). However, it should be noted that the number of possible combinations of values may be significantly reduced by the nature of the real-world relationship between the quantities being modelled, by the resolution of the data and by purpose of the model concerned. Not all possible combinations may cause the dependent variable to change significantly. Combinations that are not significantly different in effect on X may be lumped together in a single rule.

5.2.2.3 Updating State Variables with Non-Quantitative Support Sets

Here we are dealing with discretely valued variables, so the model developer should be aware of the rate of change problem and therefore specify rules that represent changes in value in such a way as to control the rate at which change occurs. As was illustrated in Chapter 4 with the RBCLM1 system, reasoning about change in non-quantitative state variables using D-Delta, T-Delta and T-In is a useful way of reasoning about how and when such variables may change value. The same principles can be applied to updating non-quantitative RBCLM2 state variables.

Intermediate variables can be used to represent the direction of change (D-Delta) of each state variable whilst time counter state variables can be used to represent the time required to change state (T-Delta) and time in state (T-In) variables. Just as with the RBCLM1, when T-In $\geq$ T-Delta the change specified by D-Delta can be implemented as the state variable value update. Values of influencing variables can be used to determine D-Delta and calculate and update T-Delta, whilst T-In can be updated based on when the state variable concerned changes value or if the conditions (influence values) change.

Alternatively rules specifying value change correspondences made conditional on a minimum length of time that a variable must be at a particular value (T-Delta $_{min}$) along with that variable's current time-in-state (T-In) can be used. Using this method T-Delta $_{min}$ represents the absolute smallest amount length of time possible for the state variable to change one discrete value i.e. its fastest possible rate of change. Unlike the RBCLM1-style T-Delta method, under this method T-Delta $_{min}$ is not calculated during a run, rather it is represented as a constant (if the state variable is an environmental property) or an attribute (if the state variable is a vegetation property). For example, in the situation where a non-quantitative state variable is only dependent upon quantitative influences then rules such as the following may be used:

IF ($0 \leq A < 100$ **AND** T-In (X) $\geq$ T-Delta $_{min}$ (X)) **THEN** (X will change from medium to low **OR** from high to medium)
IF ($200 \leq A < 300$ **AND** T-In (X) $\geq$ T-Delta $_{min}$ (X)) **THEN** (X will change from low to medium **OR** from medium to high)

ELSE X will not change value

These rules specify a value change correspondence between intervals or ranges of quantitative values and a non-quantitative state variable value change. The use of T-Delta $_{min}$ prevents overly fast dynamics from occurring but does not specify the actual rate of change and so cannot handle changes in rate (i.e. changes in T-Delta). However this may constrain dynamics sufficiently for some modelling problems. The method can be extended, using the same principles as detailed in the previous section, to cover the effects of multiple quantitative variables. More rules could be included such that each value change correspondence is between a combination of quantitative variable interval values and a change in value for the non-quantitative variable concerned.

If a non-quantitative state variable is dependent only on other non-quantitative state variables the notion of the value change correspondence incorporating T-Delta $_{min}$ can also be applied. In this case the correspondence would be between single or multiple discrete values and a change in value in the state variable concerned. Similarly if a non-quantitative variable is dependent on a mixture of both quantitatively and non-quantitatively measured variables, value change correspondences and T-Delta $_{min}$ can also be used. Here though the correspondences will be between combinations of single or multiple discrete values and single or interval quantitative values and particular discrete value changes.

5.2.2.4 Convenience Predicates for Update Rules

State variable values can be made dependent upon the values of other variables through using various RBCLM2 'convenience' (i.e. non-compulsory) predicates (detailed fully later):

- `constant`:

  Used to test or retrieve the value of named constants so they can be used as conditions or as part of update equations / rules.
- `determine_disturbance`:

  Used to determine if a named disturbance event of a particular value is occurring during the current time-step.
- `env_property`:

  Used to test or retrieve the value of named intermediate variable environmental properties so they can be used as conditions or as part of update equations / rules.
- `exogenous_variable`:

Used to test or retrieve the value of named exogenous variables so they can be used as conditions or as part of update equations / rules.

- `extract_value:`

  Used to retrieve the value of named state variables from their list data structures (e.g. Veg_details) so they can be used as conditions or as part of update equations / rules. The state variables can be vegetation or environmental properties or time counters.

- `time_counter:`

  Used to test or retrieve the value of named intermediate variable time counters so they can be used as conditions or as part of update equations / rules.

- `veg_attribute:`

  Used to test or retrieve the value of named vegetation attribute so they can be used as conditions or as part of update equations / rules.

- `veg_property:`

  Used to test or retrieve the value of named intermediate variable vegetation properties so they can be used as conditions or as part of update equations / rules.

Subsequent sections will detail the structure, meaning and use of these predicates within the RBCLM2 system. Modellers are not obliged to use convenience predicates but are encouraged to do so to ensure RBCLM2 model compatibility, KB re-use and to facilitate understanding.

5.2.2.5 KR Predicates

Figures 34 - 37 detail the structure of the update predicates and how they can be used to represent fragments of ecological knowledge about how different quantities change.

- `veg_state_update:`

  Used to determine changes in vegetation state. Figure 34 details a rule that uses a value change correspondence and a time counter check.

- `env_property_update:`

  Figure 35 details the predicate and how it can be used to update a qualitative variable, using a directed value change correspondence.

- `veg_property_update:`

  Figure 36 details the predicate and how it can be used to both update a qualitative variable using a value change correspondence and a quantitative variable using an algebraic equation embedded within the rule.

- `time_counter_update`:

  Figure 37 details a time counter update rule. Time counters are updated using equations embedded within rules. They are typically either incremented by 1 or reset.

`veg_state_update(Site,Env_details,Veg_details,Time_details,State,State_next).`

Example:

```
veg_state_update(Site,_,Veg_details,Time_details,halophilous_g
rass,juniperus_phoenicia):-
      series(Site,1),
      extract_value(canopy_height,Veg_details,Can),
      veg_type_property(transition_height,
      halophilous_grass,Trans_h),
      Can>=Trans_h,
      extract_value(time_in_state,Time_details,T_in),
      veg_type_property(minimum_time_in_state,
      halophilous_grass,Mt_in),
      T_in>=Mt_in.
```

which can be read as,
IF a site belongs to vegetation series 1 **AND** the canopy height of the site's vegetation is greater than or equal to the transition canopy height (transition_height) for halophilous grassland **AND** the site's vegetation has been in the state halophilous grassland for a period of time greater than or equal to minimum-time-in-state (MT-in) for halophilous grassland **THEN** the vegetation will change from halophilous grassland to *Juniperus phoenicia* shrubland.

Figure 34 RBCLM2 `veg_state_update` predicate

`env_property_update(Site,State,Env_details,Veg_details,Time_details,Name,Value1,Value2).`

Example:

```
env_property_update(Site,State,Env_details,Veg_details,Time_de
tails,soil_fertility,medium,low):-
      extract_value(canopy_height,Veg_details,Can),
      Can<30,
      determine_disturbance1(Site,State,Env_details,Veg_detail
      s,Time_details,grazing,_),
      constant(Site,slope_angle,Slo),
      (Slo==mild;
      Slo==medium;
      Slo==steep).
```

which can be read as,
IF the canopy height of the vegetation is less than 30 mm **AND** there is a grazing disturbance event of any severity during the current time-step **AND** the slope angle of the site is mild **OR** medium **OR** steep **THEN** the soil fertility at that site will change from medium to low.

Figure 35 RBCLM2 `env_property_update` predicate

```
veg_property_update(Site,State,Env_details,Veg_details,Time_deta
ils,Name,Value1,Value2).
```

Example 1:

```
veg_property_update(Site,State,Env_details,Veg_details,Time_deta
ils,canopy_height,Value1,zero):-
        (Value1 == medium;
        Value1 == low),
        determine_disturbance(Site,State,Env_details,Veg_details,T
        ime_details,fire,severe).
```

which can be read as,
IF there is a severe fire disturbance event and the vegetation canopy height is equal to medium **OR** low **THEN** the canopy height will be reduced to zero.

Example 2:

```
veg_property_update(Site,State,Env_details,Veg_details,Time_deta
ils,canopy_height,Value1,Value2):-
        veg_property(actual_growth_rate,Site,State,Env_details,Veg
        _details,Time_details,AGro),
        Value2 is (Value1 + AGro).
```

which can be read as,
The next value of canopy height (Value2) equals the current value (Value1) plus the value of the actual growth rate (Agro).

Figure 36 RBCLM2 `veg_property_update` predicate (example 1 uses a qualitative property, example 2 uses a quantitative property)

```
time_counter_update(Site,State,State_next,Env_details,Veg_details
,Time_details,Name,Value1,Value2).
```

Example:

```
time_counter_update(Site,State,State_next,_,_,_,time_in_state,T_i
n,T_in_next):-
        State==State_next,
        T_in_next is T_in + 1.
```

Which can be read as,
IF the vegetation at a site is going to be in the same state during the next time-step as it is during the current time-step **THEN** the next value of time-in-state (T-in-next) should be equal to the current value (T-in) + 1.

Figure 37 RBCLM2 `time_counter_update` predicate

5.2.3 Knowledge-Base: Intermediate Variables Rule-Base

5.2.3.1 Usage

Intermediate variables are those vegetation and environmental properties and time counters whose current value is not directly dependent on their previous value(s). Their values are inferred when necessary from the value of other variables (state, intermediate, exogenous and constants) through the application of intermediate variable determination rules, which are defined and stored in the intermediate variables rule-base. Intermediate variables may be used to represent amounts or directions of change (signs or next values). They may be used as conditions in update rules and other intermediate variable determination rules. As conditions, intermediate variable predicates may be used in two ways; either to check whether an intermediate variable has a particular value (e.g. a *severe* fire event may be required for vegetation to change from its present type to bare ground) or to find the current value of an intermediate variable (e.g. to calculate the value of a property such as palatability, as in the predicate `disturbance_level` below).

The RBCLM2 does not enforce any strict rules regarding how intermediate variables should be calculated. Rules for determining intermediate variable values should be based on available ecological knowledge and reflect model purpose. Again, however, various standard structures and methods emerged during the research that led to the development of the RBCLM2. These are detailed below.

5.2.3.2 Determining the Values of Intermediate Variables with Quantitative Support Sets

Just as with updating state variables, three basic scenarios can occur with respect to determining the value of quantitatively measured intermediate variables:

1. The quantitative variable is wholly dependent upon other quantitative variables.
2. The quantitative variable is partially dependent on quantitative and partially dependent on non-quantitative variables.
3. The quantitative variable is wholly dependent on non-quantitative variables.

In the first case the variable value should be calculated using an algebraic equation embedded within a rule. The formulation of such expressions will most likely be a product of empirical data analysis and / or some estimation work using standard methods.

In the second case algebraic equations, similar in form to those detailed in the section on updating quantitative state variables may be utilised. For example:

IF soil fertility = low **THEN** absolute growth rate = 0.1 x biomass

IF soil fertility = medium **THEN** absolute growth rate = 0.1 x biomass +1

IF soil fertility = high **THEN** absolute growth rate = 0.1 x biomass + 4

Again, the method can be extended to apply to more than one non-quantitative influence by using larger rule sets with each non-quantitative influencing variable's support set mapped onto either separate quantitative value sets or onto a single set in combination with other non-quantitative influences.

In the third case, the notion of the value correspondence may be applied. For example:

IF biomass = low **THEN** absolute growth rate = 10
IF biomass = medium **THEN** absolute growth rate = 20
IF biomass = high **THEN** absolute growth rate = 5

The dependency of a quantitative intermediate variable on multiple non-quantitative variables can also be captured by having each value correspondence can represent the correspondence between combinations of discrete values and a single quantitative value.

One note - the model developer should be aware that in making quantitative variables dependent on non-quantitative variables may mean that the quantitative variable's value may change quite suddenly, in jumps, rather than smoothly. Whether sharp changes in value matter to model behaviour depends on various factors including model purpose, structure and resolution.

5.2.3.3 Determining the Values of Intermediate Variables with Non-Quantitative Support Sets

Again, three scenarios exist for determining intermediate variable values:

- The non-quantitative variable is wholly dependent upon quantitative variables.
- The non-quantitative variable is partially dependent on quantitative and partially dependent on non-quantitative variables.
- The non-quantitative variable is wholly dependent on other non-quantitative variables.

For all 3 cases correspondences between combinations of influencing variable values and values for the intermediate variable can be used. Multiple variable dependencies can be handled through directed value correspondences between combinations of influencing variable values and value of the intermediate variable being determined. Rate of change does not matter in intermediate variable calculation.

5.2.3.4 Convenience Predicates for Intermediate Variable Rules

The same set of convenience predicates detailed in section 5.2.2.4 can be used as conditions in intermediate variable determination rules.

5.2.3.5 KR Predicates

If a model does not use any intermediate variables then it will have an empty intermediate variables rule-base component. This will not affect the operation of the RBCLM2 system.

The predicates used for the inference of intermediate variable values are not hardwired into the RBCLM2 in the same way as those for updating state variables. With state variables, the model developer is obliged to use the predefined predicates as they are directly called by the reasoning system. With intermediate variables, which are only ever accessed by update rules and not directly by the reasoning system, it is up to the modeller to decide whether to use the predefined 'convenience' predicates or to design new predicates more suited to the model concerned. The use of 'convenience' predicates is however advised to ensure compatibility between RBCLM2 models and to facilitate ease of understanding. There are four convenience predicates provided to represent intermediate variable determination rules:

- `disturbance_level`:

  Used to determine the values of disturbance events occurring during the current time-step. Disturbances are usually caused by exogenous variables or discrete events.

- `time_counter`:

  Used to determine the value of time counter intermediate variables. These generally represent a critical length of time important in determining some other variable value e.g. the rule in Figure 39 determines the value for a time counter, `critical_soil_development_time`, that is used as a condition in an `env_property_update` rule to act as the T-Delta $_{min}$ value for the state variable `soil_fertility`.

- `env_property`:

  Used to determine the value of non-disturbance causing environmental property intermediate variables.

- `veg_property`:

  Used to determine the value of vegetation property intermediate variables.

```
disturbance_level(Site,State,Env_details,Veg_details,Time_detai
ls,Name,Value).
```

Example:

```
disturbance_level(Site,State,_,Veg_details,_,grazing_animals,me
dium):-
       exogenous_variable(Site,grazing_animals,high),
       extract_value(canopy_height,Veg_details,Can),
       Can>0,
       veg_property(Site,State,Env_details,Veg_details,Time_deta
       ils,palatability,Pal),
       veg_property(Site,State,Env_details,Veg_details,Time_deta
       ils,actual_penetrability,APen),
       Pal==low,
       Apen==low.
```

which can be read as,
IF there are a high number of grazing animals at a site **AND** the canopy height of the site's vegetation is greater than 0 mm **AND** the site's vegetation has low palatability **AND** the site's vegetation has low penetrability **THEN** the severity of the grazing animals disturbance will be medium.

Figure 38 RBCLM2 `disturbance_level` predicate

```
time_counter(Site,State,State_next,Env_details,Veg_details,Time_
details,Name,Value).
```

Example:

```
time_counter(Site,State,State_next,_,_,_,critical_soil_developme
nt_time,1825):-
       veg_attribute(soil_development_rate,State,quick).
```

which can be read as,
IF the soil development rate under the present vegetation state is quick **THEN** the critical soil development time is 1825 days.

Figure 39 RBCLM2 `time_counter` predicate

```
env_property(Site,State,Env_details,Veg_details,Time_details,Nam
e,Value).
```

Example:

```
env_property(Site,_,_,_,_,runoff,low):-
      constant(Site,water_capacity,high),
      extract_value(soil_moisture,Env_details,Smoi),
      {Smoi == low;
      Smoi == medium}.
```

which can be read as,
IF the water holding capacity of a site's soil is high **AND** the soil moisture content of that site is low **OR** medium **THEN** the amount of runoff will be low.

Figure 40 RBCLM2 env_property predicate

```
veg_property(Site,State,Env_details,Veg_details,Time_details,Nam
e,Value).
```

Example 1:

```
veg_property(Site,State,Env_details,_,Time_details,canopy_height
_ddelta,incr):-
      extract_value(soil_moisture,Env_details,Smoi),
      {Smoi == medium;
      Smoi == high},
      exogenous_variable(Site,temperature,Temp),
      {Temp == medium;
      Temp == high}.
```

which can be read as,
IF the soil moisture level of a site is medium **OR** high **AND** the temperature of that site is medium **OR** high **THEN** the direction of change for the canopy height (canopy_height_ddelta) will be positive (incr).

Figure 41 RBCLM2 `veg_property` predicate

Note that in Figures 38 and 39 the intermediate variables represent magnitudes of amounts but in Figure 41 the intermediate variable represents the sign of a derivative (a direction of change).

5.2.4 Knowledge-Base: Vegetation Database

The Vegetation Database is used to store predicates representing knowledge about vegetation type attributes. As with intermediate variables, there is no obligation to use vegetation type attributes in any given RBCLM2 model. Even if they are used the modeller need not use the convenience predicate provided – `veg_attribute` (Figure 42). However, it is advised, for the reasons already given, that the convenience predicate be used. If a model does not use vegetation type attributes then there will be an empty vegetation database KB component.

```
veg_attribute(Name,State,Value).
```

Example:

```
veg_attribute(basic_growth_rate,thermophilous_grass,55).
```

which can be read as,
The vegetation type thermophilous grassland has a basic growth rate of 55 mm/month.

Figure 42 RBCLM2 `veg_attribute` predicate

5.2.5 Knowledge-Base: Variable Database

When being used within an external simulation environment there may be a need to convert between support sets when transferring variable values between the environment and the RBCLM2. The variables database aids integration between the RBCLM2 system and external simulation environments by containing declarative knowledge about variable names and support sets. For example, if a simulation environment cannot use text strings to display variable values, to display the values of qualitative or linguistically measured variables one option could be to convert the qualitative values to integer values. For example:

$$\{\text{low, medium, high}\} \rightarrow \{1, 2, 3\}$$

If non-quantitative initialisation data is to be set by an external simulation environment that stores the data in a numerical form then conversion will be necessary. This can be done for qualitative variables by defining a mapping such as:

$$\{\text{low, medium, high}\} \rightarrow \{(0 \leq x < 100), (100 \leq x < 600), (600 \leq x < 1000)\}$$

The Variable Database component contains predicates that define such mappings. They are not important to the basic operation of the RBCLM2 however and so will not be detailed here.

5.2.6 Simulation Datafile: Input Variables and Initialisation

5.2.6.1 Introduction

The simulation data component of the RBCLM2 system is responsible for providing the initial state variable values, the values for constants (other than vegetation attributes) and, every time-step, values for both exogenous variables and discrete events. The internal representation and use of these variables during a single time-step is common to all RBCLM2 models, operating either as stand-alone or from within an external simulation environments. However, each simulation mode may use a different method to specify simulation conditions for a run and to get the data specified into the simulation engine and reasoning system for processing each time-step.

Under stand-alone operation for example, the simulation datafile must be specified and coded in Prolog before a run. This requires that the time-steps to which each value of each variable apply must be specified beforehand in the simulation datafile. To do so, knowledge about time must be represented along with variable names and values.

Under simulation environment operation it may not be necessary for the model developer to specify and code simulation conditions before each run. Rather, exogenous variable and discrete event data may be inputted one time-step at a time from the external environment as the simulation proceeds and held as asserted predicate clauses within the RBCLM2 system. Having the simulation environment responsible for setting simulation conditions provides a way of dynamically linking RBCLM2 model input to the output of other models and vice versa.

The following section will describe the KR scheme for stand-alone operation. The details of communication between an external simulation environment and the RBCLM2 simulation engine for the purpose of initialisation and condition specification are not relevant for gaining an understanding of the RBCLM2 and so will not be entered into here.

5.2.6.2 Representing Simulation Conditions under Stand-Alone Operation

During initialisation the stand-alone simulation engine accesses the information contained within the simulation datafile on state and exogenous variable initial values and constants. Then, at the start of every time-step it accesses the simulation datafile to get the values for all relevant exogenous variables and discrete events for that time-step. To allow the user to specify these values the RBCLM2 provides several predicates. These predicates are called by the RBCLM2 simulation engine and reasoning system so they are not convenience, their use is mandatory:

- `constant:`

  Used to represent knowledge on properties that are to remain constant throughout a simulation except for vegetation attributes.

- `discrete_event`:

  Used to represent the type and timing of discrete event triggers.

- `exogenous_variable`:

  Used to represent knowledge on the type, timing and value of exogenous variables.

- `init_env_fact`:

  Used to represent knowledge about the initial value of environmental property state variables.

- `init_time_fact`:

  Used to represent knowledge about the initial value of time counter state variables.

- `init_veg_fact`:

  Used to represent knowledge about the initial value of vegetation property state variables.

- `init_veg_state`:

  Used to represent knowledge about the initial value of the state variable, vegetation state.

`constant(Site,Name,Value).`

Example:

```
constant(1,slope_angle,medium).
```

which can be read as,
At site 1 the value of the constant slope angle is medium.

Figure 43 RBCLM2 `constant` predicate

`discrete_event(Site,T,Name).`

Example:

```
discrete_event(1,10,fire).
```

which can be read as,
At simulation time-step 10, a fire trigger will occur at site 1.

Figure 44 RBCLM2 `discrete_event` predicate

```
exogenous_variable(Site,T,Name,Value).
```

Example 1:

```
exogenous_variable (1,_,wind_speed,medium).
```

which can be read as,
The wind speed at site 1 is medium regardless of the simulation time (notice the '_' used to show that the time argument, T, is to be ignored).

Example 2:

```
exogenous_variable(1,T,rainfall,low):-
      T<150.
```

which can be read as,
IF the current simulation time, T, is less than 150 **THEN** the rainfall at site 1 will be low.

Figure 45 RBCLM2 `exogenous_variable` predicate

```
init_env_fact(Site,Name,Value).
```

Example:

```
init_env_fact(1,soil_fertility,low).
```

which can be read as,
The initial value of the environmental property state variable soil fertility for site 1 is low.

Figure 46 RBCLM2 `init_env_fact` predicate

```
init_time_fact(Site,Name,Value).
```

Example:

```
init_time_fact(10,soil_development_time,100).
```

which can be read as,
The initial value of the environmental time counter state variable soil development time at site 10 is 100.

Figure 47 RBCLM2 `init_time_fact` predicate

```
init_veg_fact(Site,Name,Value).
```

Example:

```
init_veg_fact(2,canopy_height,medium).
```

which can be read as,
The initial value of the vegetation property state variable canopy height at site 2 is medium.

Figure 48 RBCLM2 `init_veg_fact` predicate

```
init_veg_state(Site,State).
```

Example:

```
init_veg_state(1,mesophilous_grass).
```

which can be read as,
The initial vegetation state for site 1 is mesophilous grassland.

Figure 49 RBCLM2 `init_veg_state` predicate

5.3 Reasoning System and Simulation Engine

5.3.1 Introduction and Operation

The RBCLM1 system enforced a particular method for reasoning about change in vegetation state – the use of D-Delta, T-Delta and T-In. To enforce this particular method the RBCLM1 reasoning system contained various rules for updating the variables T-Delta and T-In along with methods for querying rules that determine D-Delta.

The RBCLM2 system takes a different approach. With the RBCLM2 only the broad 'structural' ontology (vegetation properties etc.) is enforced – the methods for reasoning about change in state variables are not hardwired to provide greater representational flexibility. Ecological knowledge about vegetation attributes and properties is often available in various forms, so the RBCLM2 has been designed to be able to represent, reason and simulate with as many different types of knowledge as possible. This means that the RBCLM2 reasoning system does not enforce a particular method for reasoning about change in variable values, rather it is essentially a computational update algorithm to control the order in which different state variables are updated.

The reasoning system cannot drive simulations – this is the job of the simulation engine into which the reasoning system is embedded. The alternative simulation engines developed for use with the RBCLM2 vary in the exact details of their operation but all carry out approximately the same functions. Figure 50 details the RBCLM2 system operation flowchart under the stand-alone simulation engine (SASE). This differs from operation using the ModMED simulation engine (MMSE) and the MODULUS simulation engine (MOSE) in two main ways:

1. SASE run initialisation is based upon reading variable values from a simulation datafile that specifies simulation conditions for the whole run. MMSE and MOSE run initialisation relies on data being set via various predicates that can be queried by the external simulation engine
2. SASE keeps track of simulation time (T), incrementing T by 1 once per update / processing loop. MMSE and MOSE do not keep track of simulation time – this is done instead by the external simulation environment being used (i.e. Landlord and Geonamica respectively). When fired by the environment in which they are embedded, both MMSE and MOSE only go through one update / processing sequence. Results are then retrieved by the embedding environment and the RBCLM2 remains idle until prompted again at the next time-step.

The basic simulation functions (shown as dashed line boxes containing processing steps in Figure 50) are the same across the 3 simulation engines although the content of each function differs.

The RBCLM2 stand-alone simulation engine can only simulate one site at a time although, as will be discussed, spatial information can be represented and used in updating and determining variable values. When embedded within other environments multiple sites can be simulated if the environment is set up to do so. To start a stand-alone simulation run the user must specify the site to be simulated and the length of the run (T-Final). This information feeds directly into the SASE, which then initialises the site in question by accessing the simulation datafile and asserting initial state variables to the Prolog program before starting the simulation at T = 1. Time-units are not fixed, although a month is typical and one time-step equals one time-unit in length.

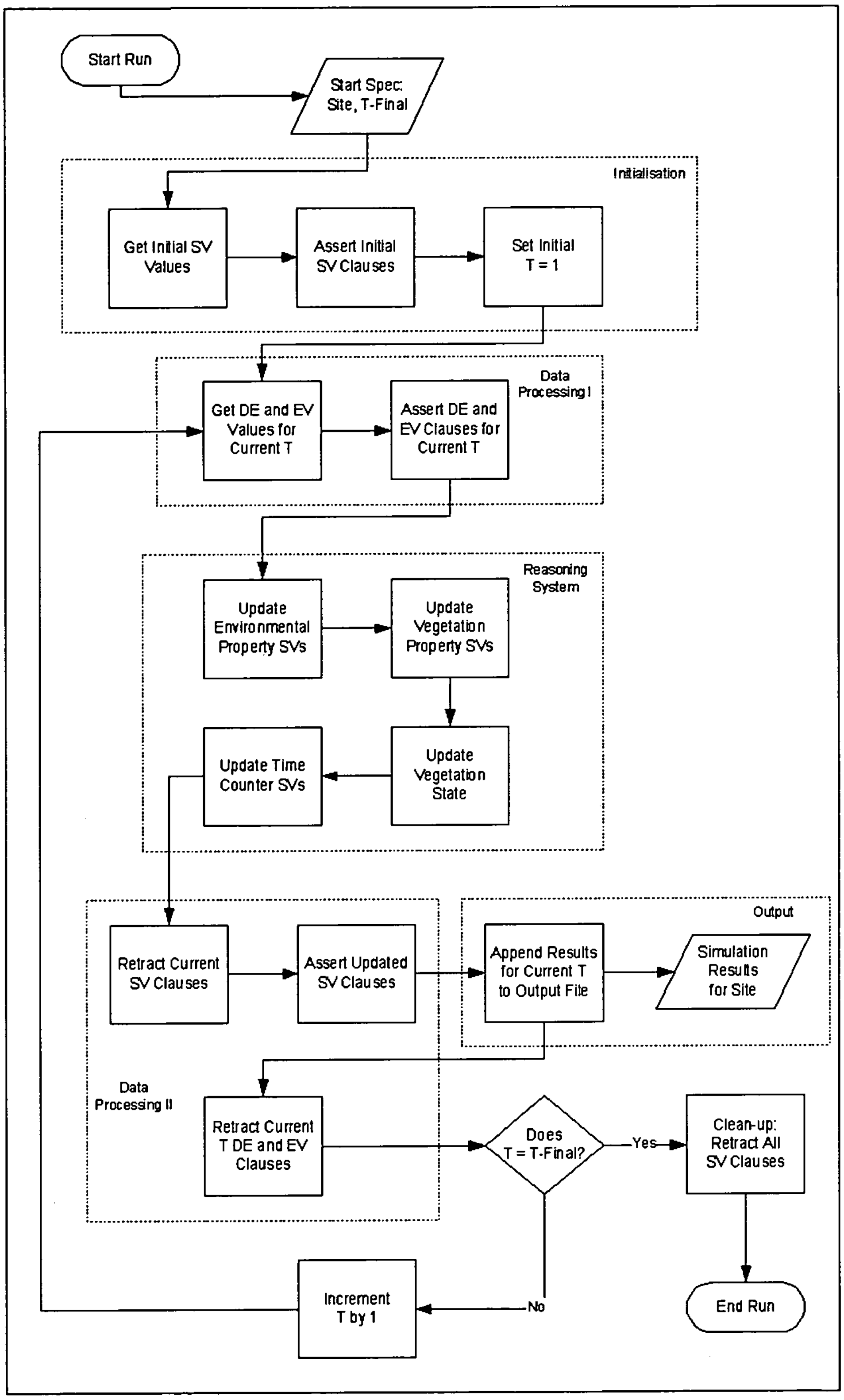

Start Run
Start Spec: Site, T-Final
Initialisation
Get Initial SV Values
Assert Initial SV Clauses
Set Initial T = 1
Data Processing I
Get DE and EV Values for Current T
Assert DE and EV Clauses for Current T
Reasoning System
Update Environmental Property SVs
Update Vegetation Property SVs
Update Time Counter SVs
Update Vegetation State
Output
Retract Current SV Clauses
Assert Updated SV Clauses
Append Results for Current T to Output File
Simulation Results for Site
Data Processing II
Retract Current T DE and EV Clauses
Does T = T-Final?
Yes
Clean-up: Retract All SV Clauses
Increment T by 1
No
End Run

Figure 50 RBCLM2 system stand-alone simulation engine operation flowchart (DE = discrete event, EV = exogenous variable, SV = state variable, T = simulation time, T-Final = simulation time at which run should stop, dashed line boxes represent simulation functions)

Data processing I occurs once per time-step and involves the SASE accessing the simulation datafile to get the values of all exogenous variable and discrete event variables for the current T. These values are then asserted as dynamic facts. At this point SASE uses the reasoning system to update model state variables in the order shown in Figure 21. During this update process intermediate variables are determined 'on the fly' as required. As discussed in section 6.1 this does not violate the 'calculate rate, update state' principle. The knowledge represented within the update and determination rules is used for all of this calculation and the responsibility for sound calculation rests there.

Data processing II involves retracting old state variable value facts and replacing them with clauses that contain the updated values resulting from application of the reasoning system's update algorithm to the KB. In addition, data processing II involves retracting all discrete event and exogenous variable facts asserted during the current time-step's data processing I stage. This prepares the way for the next time-step to begin.

Output from an RBCLM2 model run under the SASE goes to a file, appended to once per time-step. Output includes the site label, the current T, all state variable names and values and the name and value of the disturbance event for the current T.

Once data processing II and output stages have been completed the simulation either terminates, if the current T is equal to the desired simulation end-time or the processing loop is entered once more with the next T set equal to T + 1.

5.3.2 Reasoning System Predicates: Disturbance Event Handling

The RBCLM2 reasoning system component also contains a number of built-in predicates for various tasks other than variable updating. One such task is disturbance event handling.

Some state variable update or intermediate variable determination rules may specify as a condition that a particular type and value of disturbance event must be in effect at a site for the rule to succeed. However it is possible that there may be more than one factor (discrete event or exogenous variable) capable of causing a disturbance in effect at a site during a single time-step. Just like the RBCLM1, the RBCLM2 can only reason about the effect(s) of one disturbance event per time-step at each site. However due to the finer time-scale of the RBCLM2 this should present less of a problem – more than one disturbance is possible per year with the RBCLM2. To determine which disturbance occurs, the reasoning system uses a priority system identical to that of the RBCLM1. If there are 2 or more possible disturbance events for the same site during the same time-step then the one with the lowest priority value will exert an effect.

To represent the priority of different disturbance events the predicate `priority` is used (Figure 51).

```
priority(Name,Priority_value).
```

Example:

```
priority (fire,2).
```

which can be read as,
The disturbance priority of a fire is 2.

Figure 51 RBCLM2 `priority` predicate

The predicate `priority` is stored in the intermediate variables rule-base.

5.3.3 Reasoning System Predicates: Convenience Predicates

The reasoning system provides two built-in convenience predicates that may be used as conditions in state variable update or intermediate variable determination rules:

- `determine_disturbance`:

  Used as a condition where a particular type (and if desired, value) of disturbance event is required for a particular value update or calculation to apply. Note that this differs from the intermediate variable predicate disturbance_level detailed previously. Used as a rule condition by specifying a particular disturbance event (e.g. fire) and, if desired, a particular value for that disturbance event as well (e.g. medium). Figure 57 shows a general method for applying this predicate.

- `extract_value`:

  Used to extract the current value of a specified state variable from the list that details it (e.g. the value of canopy_height from Veg_details). can be used if an update or determination rule is dependent upon the value of a state variable. The predicate can be used to either obtain the variable's value or to test whether a particular value for a particular variable is currently true. For the first use the name of the state variable of interest and the list that contains its details must be specified (see Figure 58). For the second use, the name, value and containing list must be specified. The condition will succeed if the value is true and fail if not.

```
determine_disturbance(Site,State,Env_details,Veg_details,Time_d
etails,Name,Value).
```

Example:

```
predicate(argument1...n):-
      determine_disturbance(Site,State,Env_details,Veg_details,
      Time_details,erosion, severe),
      ...
```

which can be read as,
IF there is a severe erosion event in effect during the current time-step at the Site of interest **AND** all other conditions (denoted by ...) are satisfied **THEN** the consequences of predicate will hold.

Figure 52 RBCLM2 `determine_disturbance` predicate

```
extract_value(Name,List,Value).
```

Example:

```
predicate(Argument1...n):-
      ...
      extract_value(canopy_height,Veg_details,Can),
      ...
```

which can be read as,
The value of canopy_height from the list Veg_details is Can.

Figure 53 RBCLM2 `extract_value` predicate

5.4 Example Application – Capri Vegetation Model

5.4.1 Introduction

The RBCLM2 system was initially developed for a prototype model specification by Dr. Peter Csontos for the vegetation dynamics of the Island of Capri, Italy. Collaboration with various ModMED ecologists, notably Dr. Colin Legg and Dr. Peter Csontos, developed the model further using informal knowledge acquisition sessions.

This section will describe the ecological rules that formed the prototype model specification and illustrate how the available knowledge was suited to representation using the RBCLM2. In addition the RBCLM2 Capri Vegetation Model will be described. Documentation on model structure will be provided along with update rule sets for the three state variables used – vegetation state, canopy height and soil fertility. A full listing of all the rules used to determine intermediate variable values will not be given due to space limitations. The results of 2 model experiments will be documented – the first looking at the effects of increasing fire frequency with no grazing pressure, the second looking at the combined effects of increasing fire frequency and steady grazing pressure. The conditions are, as much as possible, the same as for the RBCLM1 model experiments, to facilitate comparison between the two systems.

In addition to the Capri model the RBCLM2 system has been used to model natural vegetation dynamics at a regional (50 km x 50 km) scale as part of the MODULUS project (Engelen *et al.* 2000). There, a single model was developed then parameterised for two separate regions – the Argolid in Greece and the Marina Baixa in Spain.

5.4.2 Community-Level Attributes, Properties and Dynamics: Available Knowledge

The prototype Capri Vegetation Model specification was developed by the ModMED ecologist Dr. Peter Csontos and consisted of an identified set of variables and legal values for those variables along with a set of rules that defined the conditions under which variables would assume different values. To illustrate available community-level attribute, property and dynamics knowledge and the correspondence between available knowledge and the RBCLM2 KR scheme some knowledge fragments from the prototype specification will be detailed here.

Knowledge concerning vegetation states was expressed in the prototype using classification rules. For example:

> **IF** distance from sea > 100m **AND** exposition = north **OR** exposition = plateau **AND** 700mm < canopy height ≤ 1500mm **THEN** vegetation is low macchia.
>
> **IF** distance from sea > 100m **AND** exposition = north **OR** exposition = plateau **AND** 1500mm < canopy height ≤ 4000mm **THEN** vegetation is high macchia.

These rules can be split apart and used to define a set of vegetation series rules and a set of state transition rules. For example:

> **IF** distance from sea > 100m **AND** exposition = north **OR** exposition = plateau **THEN** vegetation series = 2.

Regarding vegetation transitions, vegetation states from the same series were linked by transitions based on height. The concept of minimum time in state (T-Delta $_{min}$) was added later to control the rate of change. The specified classification rules were used as the basis for the transition rules:

> **IF** canopy height > 1500mm **AND** vegetation state $_t$ = low macchia **THEN** vegetation state $_{t+1}$ = high macchia.

The prototype model also contained various rules concerning the values of vegetation and environmental properties. Some of this knowledge was in the form of constraints on values:

> **IF** slope angle = steep **THEN** soil fertility cannot equal high.

This type of knowledge can be used to constrain how a state variable can change. For example:

> **IF** soil fertility $_t$ = medium **AND** D-Delta = inc **AND** T-In $\geq$ T-Delta $_{min}$ **AND** slope angle $\neq$ high **THEN** soil fertility $_{t+1}$ = high.

Other rules in the prototype specified the conditions under which different vegetation and environmental property values would occur. For example:

> **IF** the number of days with temperature = high and rainfall = low $\geq$ 10 **THEN** fuel continuity = high.

Rule-based knowledge fragments like that above can be easily represented using intermediate variable determination rules.

5.4.3 Structure and Value Knowledge

Figure 54 details the structure of the RBCLM2 Island of Capri Vegetation Model in the form of a directed graph. Table 35 details all the variables used in the model. The time-unit is 1 month.

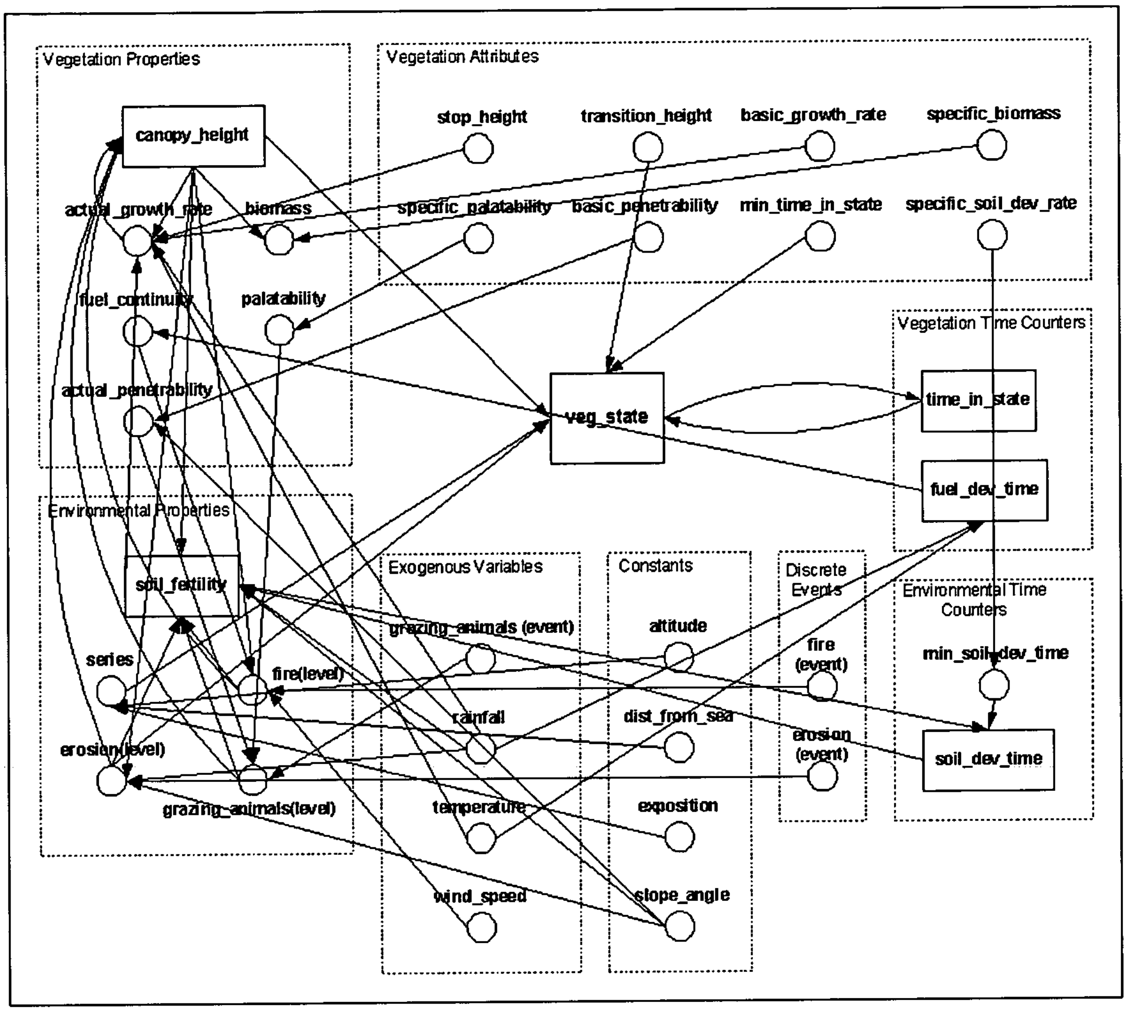


Figure 54 RBCLM2 Capri Vegetation Model structure (rectangles represent state variables, circles represent all other variables types, arrows represent influences)

Type	Variable Name	Description	Support Set and Units	Quantitative Mapping *(if applicable)*
VS	veg_state	Vegetation state / type	9 element unordered linguistic – see Fig. 28	n/a
VPSV	canopy_height	Average height of a site's vegetation canopy	Quantitative - $\Re$ (m)	n/a
VPIV	biomass	Approximate biomass of vegetation at a site	Quantitative - $\Re$ (kg / unit area)	n/a
VPIV	actual_growth_rate	The average canopy height growth rate per time-unit taking into account site conditions	Quantitative - $\Re$ (cm / month)	n/a
VPIV	fuel_ continuity	Approximately how continuous the vegetation cover is with respect to ignition and fire spread	2 element ordered linguistic {low, high}	n/a
VPIV	palatability	How palatable the vegetation is to grazing animals	2 element ordered linguistic {low, high}	n/a
VPIV	actual_penetrability	How easy it is for grazing animals to gain access to the vegetation to graze taking into account site conditions	2 element ordered linguistic {low, medium}	n/a
VTCSV	time_in_state	How long a site's vegetation has been in its current state	Quantitative – integer (months)	n/a
VTCSV	fuel_dev_time	How long the vegetation has been under low rainfall, high temperature environmental conditions	Quantitative – integer (months)	n/a
VAtt	basic_growth_rate	The canopy height growth rate for each vegetation type under optimal site conditions	Quantitative - $\Re$ (cm / month)	n/a
VAtt	basic_penetrability	How easy it is for grazing animals to gain access to the vegetation to graze	2 element ordered linguistic {low, high}	n/a
VAtt	min_time_in_state	The minimum time that vegetation must stay in a state before changing – a minimum 'maturation' time	Quantitative – integer (months)	n/a
VAtt	specific_palatability	The basic palatability of each vegetation type to grazing animals	2 element ordered linguistic {low, high}	n/a
VAtt	specific_soil_dev_time	A measure of how quickly soil develops under different vegetation types	2 element ordered linguistic {slow, quick}	n/a
VAtt	specific_biomass	Approximately how much biomass each unit of vegetation height equates to per 100 m^2 (10 m x 10 m patch)	Quantitative - $\Re$ (g / m / 100 m^2)	n/a
VAtt	stop_height	The maximum canopy height for vegetation type	Quantitative - $\Re$ (cm)	n/a
VAtt	transition_height	Used to determine when vegetation changes state. Suppose there are three vegetation types, A (a	Quantitative - $\Re$ (cm)	n/a

Type	Variable Name	Description	Support Set and Units	Quantitative Mapping *(if applicable)*
		grassland), B (a shrubland) and C (a forest type). A's transition_height = 1m, B's transition_height = 2m. A may change state into B if its canopy_height is ≥ 1 m. B can change state to A, if its canopy_height is ≤ 1 m or change state to C if its canopy_height ≥ 2 m. C may change state to B if its canopy_height ≤ 2m.		
EPSV	soil_fertility	An approximate measure of how fertile a site's soil is	3 element ordered linguistic {low, medium, high}	n/a
EPIV	series	The type of site as characterised by altitude, distance_from_sea and exposition. Constrains how a site's vegetation may change	3 element ordered linguistic {1, 2, 3}	n/a
EPIV	erosion(level)	Erosion event severity of impact	2 element ordered linguistic {mild, severe}	n/a
EPIV	fire(level)	Fire event severity of impact	2 element qualitative {mild, sever}	In terms of canopy_height reduction caused: {50%, 100%}
EPIV	grazing_animals(level)	Grazing disturbance severity of impact	4 element qualitative {none, low, medium, high}	In terms of canopy_height reduction per month (cm): {0, 1.5, 4.5, 9}
ETCSV	soil_dev_time	A time in state counter for soil fertility	Quantitative – integer (months)	n/a
ETCIV	min_soil_dev_time	The length of time that it will take, under the current vegetation cover for soil_fertility to increase by one value	Quantitative – integer (months)	n/a
DE	fire(event)	Fire trigger	Binary linguistic {yes, no}	n/a
DE	erosion(event)	Erosion event	Binary linguistic {yes, no}	n/a
EV	grazing_animals(event)	Number of grazing animals (sheep equivalents) on a site	4 element qualitative {none, low, medium, high}	Not defined
EV	rainfall	The amount of precipitation received by a site for the current time-step	3 element qualitative {low, medium, high}	Not defined
EV	temperature	The average temperature for a site for the current time-step	3 element qualitative {low, medium, high}	Not defined
EV	wind_speed	The average wind speed for a site for the current time-step	3 element qualitative {low, medium, high}	Not defined
C	altitude	The average altitude of a site (m	3 element qualitative	{(0-100), (100-

Type	Variable Name	Description	Support Set and Units	Quantitative Mapping *(if applicable)*
		a.s.l.)	{low, medium, high}	600), (>600)}
C	dist_from_sea	The average horizontal distance of a site from the sea	3 element qualitative {very_near, near, far}	{(0-20), (20-100), (>100)}
C	exposition	The average direction that a site faces in terms of slope direction and slope angle	3 element qualitative {north, south, plateau}	Uses the same mapping as the RBCLM1 example. See Chapter 5, section 5.4.2.2
C	slope_angle	The average angle of slope over a site	4 element qualitative {mild, medium, steep, extreme}	In terms of degrees from the horizontal: {(0-10), (10-40), (40-70), (70-90)}

Table 35 RBCLM2 Capri Vegetation Model variable details (VS = vegetation state, VPSV = vegetation property state variable, VPIV = vegetation property intermediate variable, VTCSV = vegetation time counter state variable, VTCIV= vegetation time counter intermediate variable, VAtt = vegetation attribute, EPSV = environmental property state variable, EPIV = environmental property intermediate variable, ETCSV = environmental time counter state variable, ETCIV = environmental time counter intermediate variable, DE = discrete event, EV = exogenous variable, C = constant)

The Capri Vegetation Model was based upon a mixture of available knowledge – quantitative, qualitative and linguistic. To reflect available knowledge change in vegetation type was modelled based upon canopy height, time-in-state and minimum time in state (i.e. T-Delta $_{min}$). Canopy height was modelled quantitatively. Various other variables were also measured using quantitative scales as the available knowledge was in this form. Where qualitative support sets were used the principle of minimum variation Salles (1997) was followed as much as possible. Accordingly, most non-quantitative variables are measured using 2 or 3 element support sets.

5.4.4 Relationship Knowledge

Figure 55 depicts the possible states and state transitions that may occur in the RBCLM2 Capri Model. Table 36 provides details on the conditions that must be met for each transition to occur.

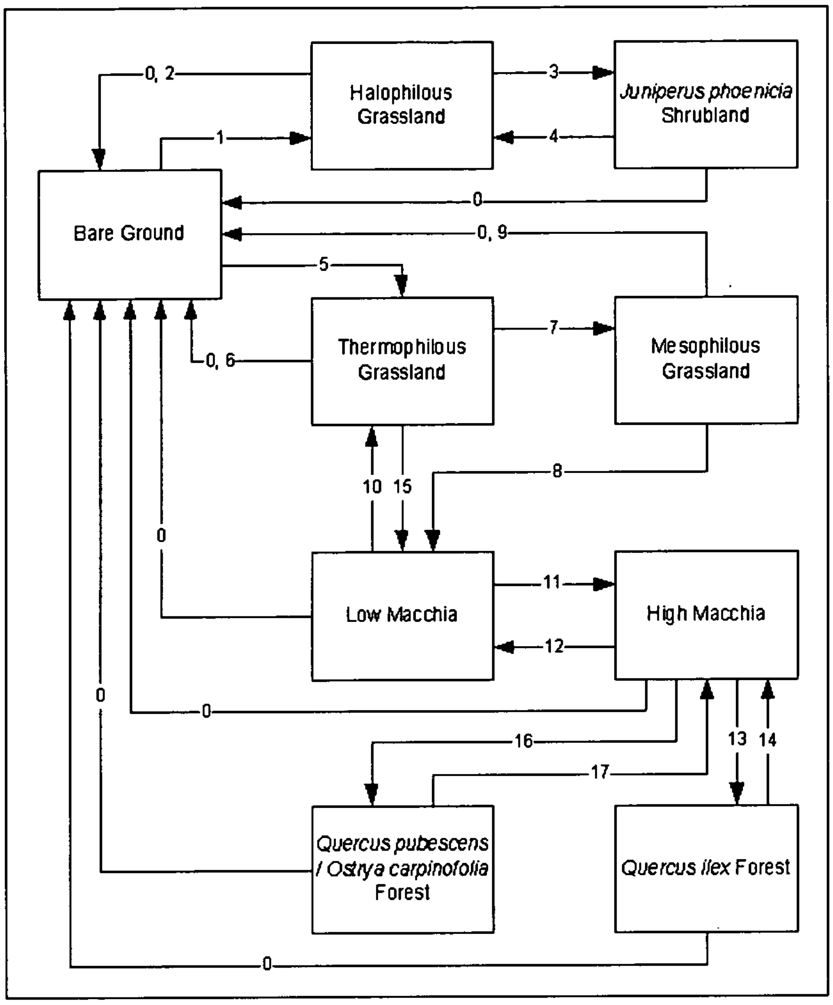


Figure 55 RBCLM2 Capri Vegetation Model vegetation state transition diagram (rectangular nodes represent vegetation states, arcs represent transitions under specific sets of conditions, each represented by a number)

The first part of determining how and when vegetation changes state is the determination of a site's series. The rules for determining series are as follows:

IF ('distance from sea' is equal to 'very near' **OR** 'near') **AND** ('altitude' is equal to 'low' **OR** 'medium') **THEN** the 'series' is 1.

IF 'distance from sea' is equal to 'far' **AND** ('exposition' is equal to 'north' **OR** 'plateau') **THEN** the 'series' is 2.

IF ('distance from sea' is equal to 'very_near' **OR** 'near') **AND** 'altitude' is equal to 'high' **AND** ('exposition' is equal to 'north' **OR** 'plateau') **THEN** the 'series' is 2.

IF 'distance from sea' is equal to 'far' **AND** 'exposition' is equal to 'south' **THEN** the 'series' is 3.

IF ('distance from sea' is equal to 'very near' **OR** 'near') **AND** ('altitude' is equal to 'high' **OR** 'south') **THEN** the 'series' is 3.

Number	Preceeding State	Succeeding State	Conditions
0	*Any except Bare Ground*	Bare Ground	E = severe
1	Bare Ground	Halophilous Grassland	S = 1, CH ≥ 30, TIN ≥ 12
2	Halophilous Grassland	Bare Ground	S = 1, CH ≤ 30
3	Halophilous Grassland	*Juniperus phoenicia* Shrubland	S = 1, CH = 500, TIN ≥ 60
4	*Juniperus phoenicia* Shrubland	Halophilous Grassland	S = 1, CH ≤ 500
5	Bare Ground	Thermophilous Grassland	(S = 2; S = 3), CH ≥ 30, TIN ≥ 12
6	Thermophilous Grassland	Bare Ground	(S = 2; S = 3), CH ≤ 30
7	Thermophilous Grassland	Mesophilous Grassland	S = 2, TIN ≥ 36
8	Mesophilous Grassland	Low Macchia	S = 2, CH ≥ 700, TIN ≥ 96
9	Mesophilous Grassland	Bare Ground	S = 2, CH ≤ 30
10	Low Macchia	Thermophilous Grassöamd	(S = 2; S = 3), CH ≤ 700
11	Low Macchia	High Macchia	(S = 2; S = 3), CH ≥ 1500, TIN ≥ 60
12	High Macchia	Low Macchia	(S = 2; S = 3), CH ≤ 1500
13	High Macchia	*Quercus ilex* Forest	S = 2, CH ≥ 4000, TIN ≥ 480
14	*Quercus ilex* Forest	High Macchia	S = 2, CH ≤ 4000
15	Thermophilous Grassland	Low Macchia	S = 3, CH ≥ 700, TIN ≥ 36
16	High Macchia	*Quercus pubescens / Ostray carpinofolia* Forest	S = 3, CH ≥ 4000, TIN ≥ 480
17	*Quercus pubescens / Ostray carpinofolia* Forest	High Macchia	S = 3, CH ≤ 4000

Table 36 RBCLM2 Island of Capri Vegetation Model vegetation state transition details (';' means logical 'or', ',' means logical 'and', CH = canopy_height (cm), E = erosion(level), S = sere, TIN = time_in_state (months))

Vegetation state transitions are modelled as being a direct function of four factors – present state, series, canopy height and time in state, although not all influence every transition. Other influences (rainfall, grazing etc.) affect transitions indirectly through influencing vegetation properties like actual growth rate or by determining the series to which a site belongs.

The basic reasoning applied to updating vegetation state can be summarized in two main rules – one for 'progressive' change and one for 'regressive' change. For progressive change between two states, A and B, where state B has a higher maximum height than state A:

> **IF** the canopy height of vegetation in state A is ≥ the transition height for state A **AND** the site's vegetation has been in state A for a length of time (time in state) ≥ the minimum time in state for state A to change **THEN** the transition from A to B will occur.

This rule reflects one way of reasoning about how vegetation changes – using height as a proxy and a time in state counter to control rate of change. In progressive vegetation change the dominant species that go to make up the succeeding state are typically taller and more woody. An approximate value for the canopy height at which it is reasonable to say 'most of the species present are now from the succeeding state' is used to partly determine when the change from one state to the next. This value is represented using the transition height vegetation attribute. The use of a minimum time in state counter represents the approximate minimum time (i.e. under optimal conditions) that vegetation in one state could develop into the next.

For regressive change between A and B, where B has a higher maximum height than A:

> **IF** the canopy height of vegetation in state B is ≤ the transition height of state A **THEN** the transition from B to A will occur.

This rule reflects the fact that regressive change is often caused by destruction of vegetation (e.g. by fire or grazing) rather than through a developmental process. The change in state can be viewed as having occurred because the dominant species of one state (e.g. B) have been removed, lowering the average canopy height and opening up gaps in the cover. The relative dominance of other species will increase as a result. The processes behind regressive change in such situations certainly require some amount of time to alter the vegetation (e.g. the action of grazers over a period of months) but do not require a minimum time as required in progressive change. The removal of dominant species can occur over an hour, a day, a week, a month etc. The logic behind these two rules formulations is reflected in the Capri state transition rules (see Table 36).

The RBCLM2 Capri Model also models the dynamics of other state variables – canopy height, time in state, soil fertility and soil development time. The Capri model rules will now be detailed.

Canopy height is a quantitatively measured state variable:

$$\text{canopy\_height}_{t+1} = f(\text{canopy\_height}_t, \text{erosion(level)}, \text{fire(level)}, \text{grazing\_animals(level)}, \text{actual\_growth\_rate})$$

A set of update rules using a mixture of value change correspondences and algebraic equations are used to update canopy height. The update rules involve determining whether disturbance events of specified values are currently in operation, calculating the effect of disturbances and determining the value of actual growth rate, a process that involves determining a chain of intermediate variables upon which actual growth rate is dependent. The basic rule set used:

IF there is an erosion disturbance of level = severe **THEN** canopy height at time t +1 will be 0, irrespective of its current value.

IF there is a fire disturbance of level = x **THEN** canopy height at time t +1 will be reduced by an amount equal to the reduction factor associated with fire level x

where,

fire(level) = mild $\rightarrow$ reduction factor = 50% of current value

fire(level) = severe $\rightarrow$ reduction factor = 100% of current value

IF there is a grazing animals disturbance of level x **THEN** canopy height at time t +1 will be equal to the canopy height at time t + actual growth rate – the grazing modifier associated with grazing animals disturbance level x such that the minimum value for canopy height is $\geq$ 0 cm.

where,

grazing_animals(level) = none $\rightarrow$ modifier = 0

grazing_animals(level) = low $\rightarrow$ modifier = 1.5 cm

grazing_animals(level) = medium $\rightarrow$ modifier = 4.5 cm

grazing_animals(level) = high $\rightarrow$ modifier = 9 cm

IF there is no disturbance **THEN** canopy height at time t +1 will be equal to canopy height at time t + actual growth rate

It should be noted that the effects of fire are modelled only in a very coarse-grained way. No attempt has been made to differentiate crown fires from ground-level fires.

The vegetation time counter state variable, time in state, is updated using two simple rules:

IF the vegetation state at time t + 1 ≠ the vegetation state at time T **THEN** time in state will be set equal to 1, irrespective of its current value.

IF the vegetation state at time t + 1 = the vegetation state at time T **THEN** time in state will be incremented by 1.

The environmental property state variable soil fertility is valued using a 3 element qualitative support set and updated using a set of rules. The ways in which the variable can change value can be viewed as state, or value, transitions in a way very similar to that of vegetation state. Figure 56 details the possible values and transitions that can occur. Table 37 details the transition conditions.

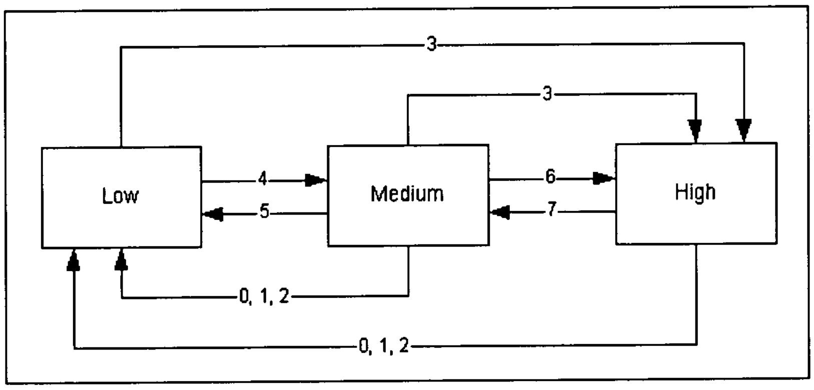


Figure 56 RBCLM2 Capri Vegetation Model soil fertility variable value transition diagram (nodes represent variable values, arcs represent value transitions under specific, numbered conditions)

Number	Preceeding Value	Succeeding Value	Conditions
0	*Any value*	Low	E = severe
1	*Any value*	Low	F = severe, R = high
2	*Any value*	Low	CH < 30, R = high
3	*Any value*	High	F = severe, R = low
4	Low	Medium	SDT ≥ CSDT, (SLO = mild; SLO = medium; SLO = steep)
5	Medium	Low	CH < 30, GA = *any value*, (SLO = mild; SLO = medium; SLO = steep)
6	Medium	High	SDT ≥ CSDT, (SLO = mild; SLO = steep)
7	High	Medium	CH < 30, GA = *any value*, (SLO = mild; SLO = medium)

Table 37 Figure 57 RBCLM2 Capri Vegetation Model soil fertility value transition details (CH = canopy_height, CSDT = critical_soil_development_time, E = erosion(level), F = fire(level), GA = grazing_animals(level), R = rainfall, SDT = soil_development_time SLO = slope_angle)

Soil fertility, a rough gauge of soil nutrient quality with respect to plant growth is modelled as:

soil_fertility $_{t+1}$ = f(soil_fertility $_t$, erosion(level), fire(level), grazing_animals(level), rainfall, canopy_height $_t$, soil_dev_time, min_soil_dev_time, slope_angle)

Basically, soil fertility is modelled as increasing under vegetation cover, but at rates specific to each vegetation state to reflect differences in litter quality and decomposition rates. Free from disturbance soil fertility will increase according to the following rule:

IF the soil development time (a time in state counter for soil fertility) ≥ the minimum soil development time (to increase by one value) for the present vegetation state **THEN** the value of soil fertility at time t +1 will be one value higher than at time t, until the maximum value is reached.

The maximum value is constrained by slope angle as highly fertile soils are modelled as not developing on extreme angle slopes (70-90^0). Disturbance factors like erosion, fire and trampling caused by grazing can reduce soil fertility. Severe fire disturbance is modelled as increasing soil fertility, but only under low rainfall conditions as high precipitation will wash away the nutrients that are freed following a fire event. If none of the rule conditions are met the value of soil fertility at time t+1 will be the same as the value of soil fertility at time t.

The environmental time counter state variable soil development time acts as a time in state variable for soil fertility to control the rate of increase of the variable. The update rule set for this counter is very simple and akin to that of the time in state variable.

> **IF** soil fertility at time t +1 ≠ soil fertility at time t **THEN** soil development time at time t +1 will be set to 1, irrespective of its value at time t.
>
> **IF** soil fertility at time t +1 = soil fertility at time t **THEN** soil development time at time t +1 will be incremented by 1, irrespective of its value at time t.

5.4.5 Model Results

As with the corresponding section in Chapter 5 the aim here is to demonstrate and assess the utility of the RBCLM2 for modelling vegetation dynamics. The results from a simulation experiment for a single site belonging to sere 2 are detailed below.

The experiment included a baseline run (disturbance free), plus a run for no fire with low grazing then a series of runs maintaining constant low grazing pressure with increasing fire frequency. Temperature and rainfall followed a typically Mediterranean annual climatic pattern (high summer temperature with low precipitation and relatively cold, wet winters) and each run was for a total of 1200 time-steps or 100 years (1 time-step = 1 month). Results for the 3 main state variables of interest, vegetation state, canopy height and soil fertility are detailed. Table 38 details the conditions for each run of the experiment.

	Run Number				
	1	2	3	4	5
Initial Values					
Vegetation State	Bare Ground	Bare Ground	Bare Ground	Bare Ground	Bare Ground
Canopy Height	10	10	10	10	10
Time in State	0	0	0	0	0
Soil Fertility	Low	Low	Low	Low	Low
SDT [*]	0	0	0	0	0
TSF [**]	0	0	0	0	0
FDT [+]	0	0	0	0	0
Constants					
Altitude	Medium	Medium	Medium	Medium	Medium
DFS [++]	Far	Far	Far	Far	Far
Exposition	North	North	North	North	North
Slope Angle	Medium	Medium	Medium	Medium	Medium
DEs					
Fire	None	Yes – *once per 120 time-steps from time-step 6 onwards (10 years)*	Yes – *once per 120 time-steps from time-step 6 onwards (10 years)*	Yes – *once per 60 time-steps from time-step 6 onwards (5 years)*	Yes – *once per 12 time-steps from time-step 6 onwards (every year)*
EVs					
Grazing	None	None	Low	Low	Low
Rainfall	Low: *Months 3 – 10 of every year (1 year = 12 time-steps)* Medium: *Months 1, 2, 11 & 12 of every year*	Low: *Months 3 – 10 of every year (1 year = 12 time-steps)* Medium: *Months 1, 2, 11 & 12 of every year*	Low: *Months 3 – 10 of every year (1 year = 12 time-steps)* Medium: *Months 1, 2, 11 & 12 of every year*	Low: *Months 3 – 10 of every year (1 year = 12 time-steps)* Medium: *Months 1, 2, 11 & 12 of every year*	Low: *Months 3 – 10 of every year (1 year = 12 time-steps)* Medium: *Months 1, 2, 11 & 12 of every year*
Temperature	Low: *Months 1 and 12 of every year* Medium: *Months 2, 3, 10 & 11 of every year* High: *Months 4 – 9 of every year*	Low: *Months 1 and 12 of every year* Medium: *Months 2, 3, 10 & 11 of every year* High: *Months 4 – 9 of every year*	Low: *Months 1 and 12 of every year* Medium: *Months 2, 3, 10 & 11 of every year* High: *Months 4 – 9 of every year*	Low: *Months 1 and 12 of every year* Medium: *Months 2, 3, 10 & 11 of every year* High: *Months 4 – 9 of every year*	Low: *Months 1 and 12 of every year* Medium: *Months 2, 3, 10 & 11 of every year* High: *Months 4 – 9 of every year*
Wind Speed	Medium	Medium	Medium	Medium	Medium

Table 38 Simulation Conditions for RBCLM2 Island of Capri Vegetation Model Experiment ([*] SDT = soil development time, [**] TSF = time since fire, [+] FDT = fuel development time, [++] DFS = distance from sea, DEs = discrete events, EVs = exogenous variables)

The results figures (58 – 62) present the discrete values for vegetation state and soil fertility using the following coding scheme:

- Vegetation state:

  1 = Bare Ground, 2 = Halophilous Grassland, 3 = Thermophilous Grassland, 4 = Mesophilous Grassland, 5 = Low Macchia, 6 = *Juniperus phoenicia* Shrubland, 7 = High Macchia, 8 = *Quercus ilex* Forest, 9 = *Quercus pubescens / Ostrya carpinofolia* Forest.
- Soil fertility:

  1 = low, 2 = medium, 3 = high.

Run 1, the baseline run simulates typical long-term Mediterranean vegetation dynamics (see Chapter 2). Starting in a bare ground state, the vegetation gradually develops in height and stature towards a *Quercus ilex* forest type (number 8), which is reached in just under 60 years. The development of the vegetation type can be traced by looking at the canopy height and growth rate (slope of canopy height results graph). The growth rate over the 100-year period varied, changing roughly when the vegetation state changed, reflecting differences in the basic growth rate of each type. The overall rate of growth slowed as the vegetation matured towards forest cover. At various points canopy height leveled off with a zero growth rate. This can be seen for example between years 30 and 60 when the canopy height stayed steady at over 4000 cm. At around 20 years the vegetation had changed from low macchia (number 5) to high macchia (number 7) due to its canopy height being greater than the transition height for low macchia and it having been in the low macchia state for a sufficiently long period of time. In doing so the vegetation continued to grow towards the maximum height for the high macchia state. Once it had reached this value (4400 cm in the model) the vegetation stopped growing but had not yet been in the high macchia state for a period of time sufficient to permit the transition to *Quercus ilex* forest. Consequently, the vegetation remained at the same average canopy height whilst maturing into the forest state. Soil fertility rose quite rapidly from low to high during the first fifteen or so years of the run.

Run 2 showed qualitatively identical patterns to run 1 in all three state variables but under conditions of no grazing and fire with an MRI = 10. The model did not produce the dynamic expected – that of vegetation degradation. It has been noted that a fire MRI = 20-30 would be expected not to cause degradation (dynamics akin to those shown) (Keeley 1986). Fire MRI = 10 has been noted as being sufficient to prevent shrubland from developing into forest in Mediterranean vegetation (Le Houerou 1981).

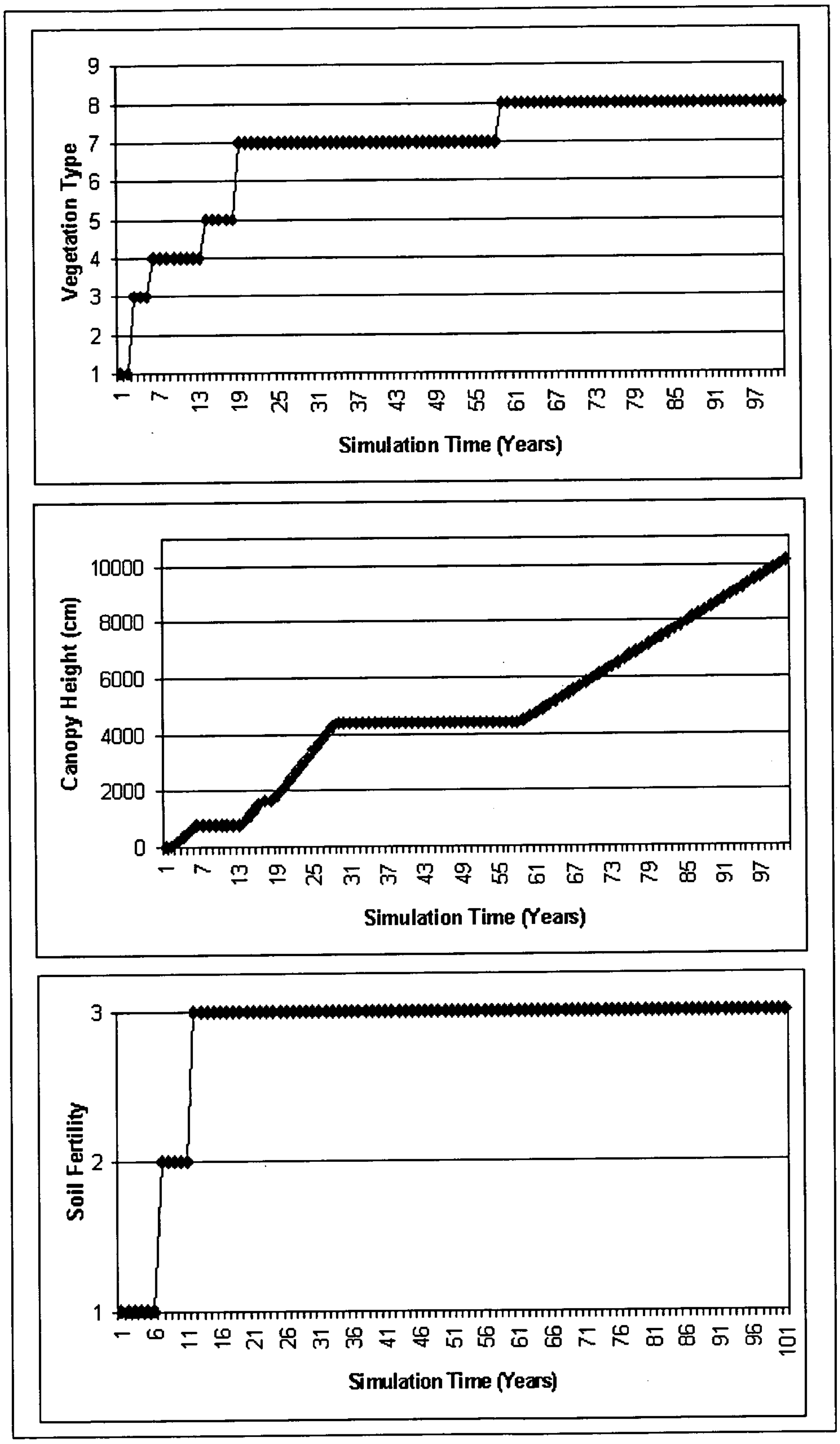


Figure 58 RBCLM2 Capri Vegetation Model run 1 results – no fire, no grazing

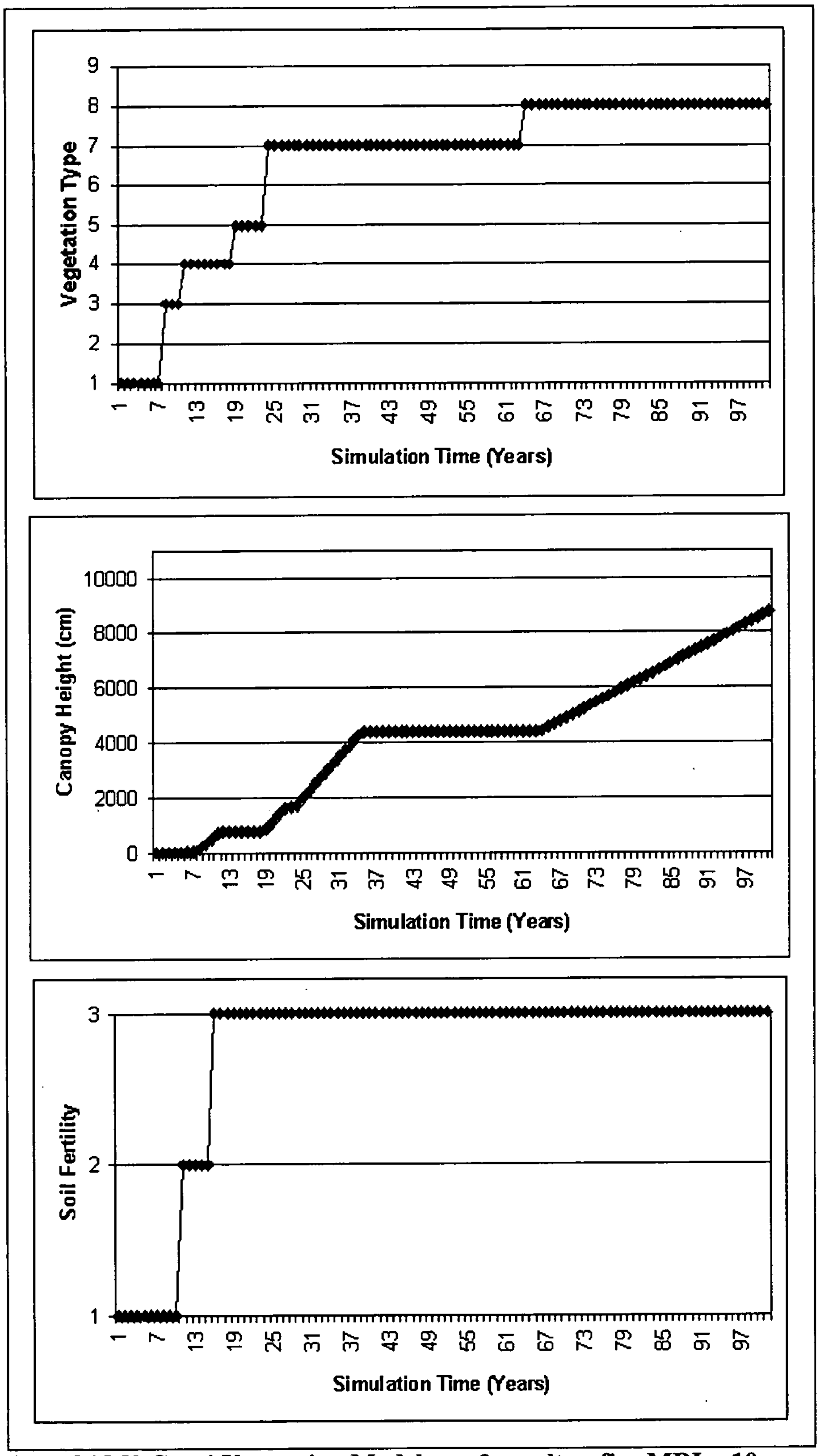


Figure 59 RBCLM2 Capri Vegetation Model run 2 results – fire MRI = 10, no grazing

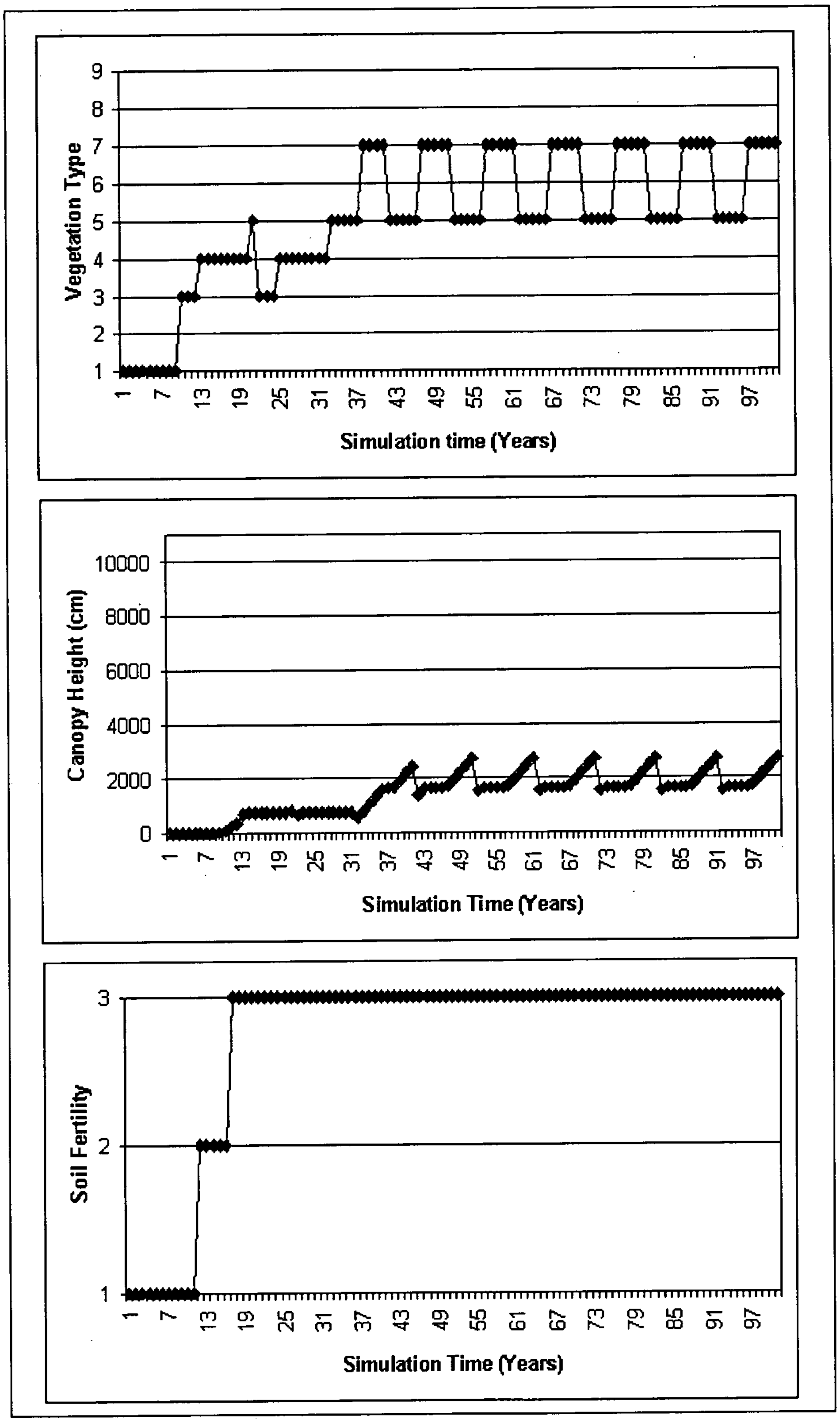


Figure 60 RBCLM2 Capri Vegetation Model run 3 results – fire MRI = 10, low grazing

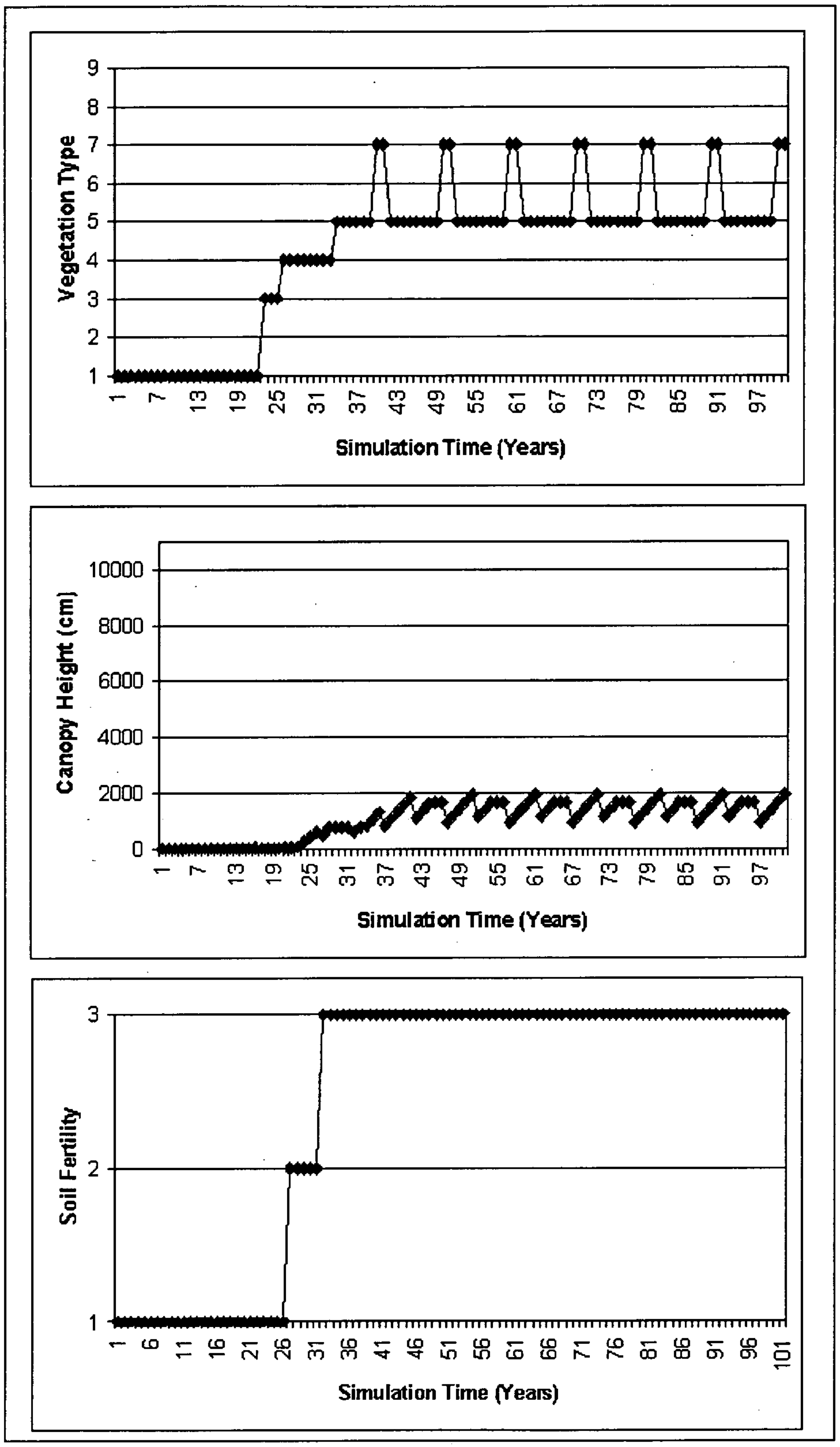


Figure 61 RBCLM2 Capri Vegetation Model run 4 results – fire MRI = 5, low grazing

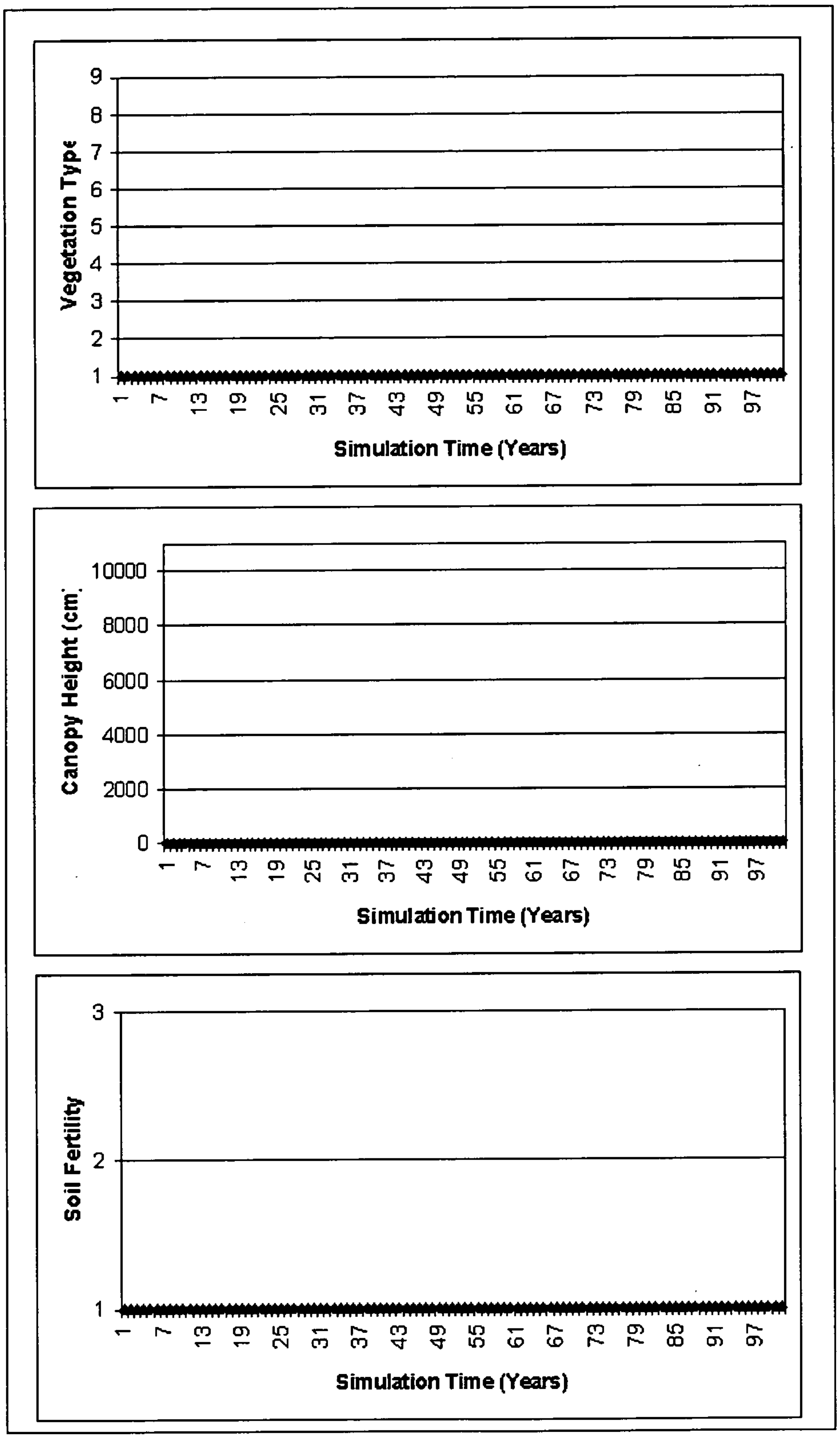


Figure 62 RBCLM2 Capri Vegetation Model run 5 results – fire MRI = 1, low grazing

Run 3 involved conditions of low grazing and a fire regime with a MRI of 10 years. The results were qualitatively different from those of runs 1 and 2. Vegetation never developed into a forest state, the end-point being a stable oscillation between low and high macchia maintained by fire events every ten years. The vegetation developed into mesophilous grassland then briefly transitioned into low macchia before a fire event at around year 20 caused the vegetation to regress to thermophilous grassland. By year 25 the vegetation had returned to mesophilous grassland and by year 35 or so the vegetation had developed into low macchia. Development continued until high macchia was reached at just before year 40. The fire at just after year 40 appears to have degraded the vegetation into a low macchia state from where the development-disturbance caused oscillation between the two macchia states begins. During this time period canopy height never grows above around 2500 cm, most of time staying beneath 2000 cm, presumably due to the combined effects of a 10 year MRI fire regime and constant low intensity grazing pressure. The stop height of low macchia is modeled as being 1650 cm. Soil fertility rise from low to high over the first 19 or so years of the simulation.

Run 4 shows qualitatively similar results for the 3 state variables to run 3. The vegetation never develops into forest cover, the most mature stage reached being high macchia. The vegetation reaches a low macchia state by around year 35. It transitions into high macchia at around year 40 then quite rapidly degrades into low macchia after around 2 years. This marks the start of the final behavioural pattern for the model – a stable oscillation between low and high macchia with approximately 8 years in the low macchia state, then 2 years in high macchia before returning to low macchia for a further 8 years and so on. As this occurs canopy height fluctuates in a complicated but regular manner between 1000 and 2000 cm. Soil fertility rises from low to high but takes longer than in runs 1, 2 and 3 – approximately 30 years.

Under constant low grazing intensity and a fire regime with an MRI of 1 year (run 5) vegetation is prevented from developing at all. Vegetation state never changes from bare ground. The results graphs for all three state variables are essentially flat.

5.5 Discussion

5.5.1 Modelling Vegetation Dynamics – RBCLM2 System Utility

The results from the simulation experiment demonstrate that the RBCLM2 system (using the Capri Model) is capable of replicating the types of dynamic that would be expected under certain conditions but not others. Vegetation develops in a classic grassland – shrub – forest sequence under disturbance free and infrequent (MRI = 10) fire conditions with differences in time taken to reach forested state simulated. As mentioned this is fine and expected for the disturbance free run conditions but is qualitatively different from the dynamics expected under fire of MRI = 10. Fire of this or higher frequency would be expected to cause degradation in vegetation and prevent development to forest, marking a need for the sensitivity of the model's vegetation types to fire to be investigated as they are not sufficiently responsive. However the combined action of fire and grazing cause progressively more severe degradation as fire MRI decreases as would be expected in Mediterranean vegetation under some conditions. Overall the runs, although not perfect, demonstrate that is possible to coherently reason with a set of community-level knowledge fragments to model vegetation.

With respect to the dynamics characterisation of Burrows (1990), the RBCLM2 System is capable of replicating types 1, 2 3, 5 and 6:

- *Colonization and sequential replacement (Burrows dynamic type 1):*

  All 5 simulation runs demonstrate colonization from bare ground and the effects of sequential replacement in terms of vegetation state changes and differences in growth rate.
- *Direct replacement following the disturbance of established vegetation (Burrows dynamic type 2):*

  Run 2 demonstrates direct replacement following disturbance.
- *Cyclic replacement in response to endogenous or exogenous influences (Burrows dynamic type 3):*

  Both runs 3 and 4 demonstrate cyclical species replacement in terms of cyclical state dynamics, canopy height fluctuations and growth rate variations.
- *Fluctuating replacements in response to exogenous influences (Burrows dynamic type 4):*

  From the simulation results it is not clear how well the RBCLM2 can simulate fluctuating replacements. This is due to the annual resolution used to present results. There is no *a priori* reason to think that it cannot handle such changes. A monthly time-step and the seasonal variation in exogenous variables used in the Capri model are sufficient to capture seasonal fluctuations. They will most likely be picked up in sensitive variables like canopy height.
- *Vegetation maintained by frequent or continual disturbance (Burrows dynamic type 5):*

Run 5 demonstrates vegetation maintained in a bare ground state by the combined action of fire and grazing.

- *Vegetation in relative equilibrium for a time (Burrows dynamic type 6)*

  All runs demonstrate vegetation in states of equilibrium for varying periods of time. There is no *a priori* reason to suspect that rule-bases could not be designed or model runs undertaken that produce this behaviour more often.

- *Long-term, gradual change in response to autogenic or allogenic influences (Burrows dynamic type 7):*

  As with the RBCLM1 the use of a predefined set of states and transitions puts limits on the RBCLM2. The lack of a mechanistic basis means that it cannot model the effects of novel conditions on species mixtures.

The representation of vegetation structure in the RBCLM2 is more detailed compared to that of the RBCLM1 system and therefore capable of more accurately modelling dynamics. Differences in behaviour between different instances of the same vegetation type can be handled through vegetation properties. This partially solves the first problem with RBCLM1 as noted in Chapter 4.

However, each instance of each vegetation type, although capable of being in a slightly different overall state (where state means a combination of property values and type) will still possess the same value for its vegetation attributes as other instances of the same type. This means that the underlying functional characteristics and behaviours will be the same across all instances of each vegetation type, despite differences in property values. This feature may have potentially important consequences for model behaviour.

Ecologically, the value of vegetation attributes depends on the species composition of vegetation, not just on the vegetation type. Different species behave in different ways and as their abundance changes, so overall vegetation behaviour will also change. Each vegetation type may contain a range of different species compositions and therefore a range of values for each functional attribute. Differences in species composition within vegetation types are modelled in the RBCLM2 by means of vegetation properties such as canopy height. In an RBCLM2 model, as canopy height reduces but vegetation type stays constant, this can be interpreted as meaning that the average height is decreasing across the patch of vegetation, and also that species of smaller stature are increasing in abundance and changing the composition. Using the RBCLM2 system ontology even though the vegetation structure is changing, the values of the attributes used to model it remain constant until a state transition occurs. This may introduce error into RBCLM2 model behaviour.

At the species-level, the observed behaviour of a population will vary as a function of both species level properties (e.g. biomass, height etc.) and the species' functional attributes (life –history characteristics and morphology etc.) amongst other things. The attributes of a vegetation state in the RBCLM2 represent the aggregate functional characteristics of all species (and individuals) present in a community type. These characteristics will vary as species composition varies. The attributes of species populations do not represent the aggregated characteristics of all the individuals involved – they represent characteristics 'hard-wired' to each species and to each individual within each species. They will not change significantly irrespective of what state the species population is in over the time-scales we are looking at. The use of species-level attributes may provide a more accurate means of modelling vegetation dynamics than the RBCLM2. The next chapter will explore a species-level ontology with the aim of improving accuracy.

The RBCLM2 uses the same disturbance handling system as the RBCLM1. In the RBCLM1 this system could potentially lead to erroneous results. However, the RBCLM2 system can operate with a higher temporal resolution (monthly time-unit is normal for a model) due to it using a more detailed structural representation of vegetation (state, properties and attributes) where finer-scale changes can be represented based upon finer temporal resolution knowledge. The higher temporal resolution means that the use of the 'priority-of-effect' disturbance handling system is more likely to be sound. The monthly time-unit means that even if two disturbance events are scheduled to happen in the same time-step, the fact that the disturbance system can only reason about and handle the effects of one disturbance should not introduce error into the system. For example, if a fire event and grazing both occur during the same month then the RBCLM2 system will, assuming fire has priority of effect, only reason about the effects of the fire. The effects of the grazing will be ignored. This is a reasonable way to handle this combination of disturbances. In real vegetation, if grazing had occurred before the fire event, the effects of the grazing would be over-ridden by the effects of fire. If fire occurred before grazing then the amount of re-growth and possible damage that could be inflicted by post-fire grazing would most likely be insignificant on the scale of whole community dynamics.

Last, with the RBCLM2, a site's environment can be represented and affected by the state of the vegetation present, with the result that potentially important ecological processes can be modelled. The inclusion of an expanded environmental component in the RBCLM2 system ontology solves the last of the ontological problems faced by the RBCLM1 system.

5.5.2 Knowledge Representation

The RBCLM2 KR scheme is flexible, permitting the representation of a wide range of knowledge about how real-world quantities change over time in response to each other and are related to one another. The KR scheme enforces little hard-wired structure and as such allows knowledge to be represented in forms as near to the format in which it is available as possible. This allows RBCLM2 models to have a strong structural correspondence to available knowledge, potentially facilitating understanding and interpretation of models operation and results.

The KR scheme places little constraint on legal variable values and on model structure. Mixed support set relationships can be handled with no extra effort. The Capri model demonstrates that using knowledge fragments based on correspondences between variable values (either values or changes in values) is a useful means of representing and reasoning about change, not only of quantitative variables but of non-quantitative variables as well. Ecological knowledge is often incomplete and approximate. Rough correspondence between different variable values is often the only form in which knowledge is available.

However, the RBCLM2 KR scheme may be too flexible, insufficiently constraining the knowledge that can be represented and the formats for doing so. Beyond the ontology, there is little that must be adhered to, creating an open and flexible modeling system with little guidance to the user on how to model. Some possible errors and problems may be of a 'debugging' nature i.e. the RBCLM2, in enforcing few representational constraints, leaves the model developer with more opportunity to make coding mistakes that may be rectified with varying degrees of ease during verification. These mistakes can be regarded as model implementation 'bugs'. More seriously however, the lack of representational guidance may lead to more fundamental design errors in how variable values are reasoned with, particularly how they are updated. Discretely measured variables require careful handling to ensure that they change value at an appropriate rate in an appropriate direction. One method for doing so is to use temporal reasoning based upon D-Delta, T-Delta and T-In for each discretely measured state variable, as was used in the RBCLM1 system. Other methods can be employed but attention needs to be paid to their ability to accurately capture real dynamics. This is not a flaw in the system as such, but it does mean that extra onus is placed on the model developer. To reduce the possibility for error and provide a common language for representing knowledge about updating and determining the values of non-quantitative variables it can be argued that the enforcement of a stricter method would be beneficial. The notions of value change and value correspondences along with an extended version of the temporal reasoning system employed in the RBCLM1 are a good candidate for such a scheme. This will be explored in detail during the next Chapter.

The RBCLM2 system makes some progress towards separating out knowledge about model structure, variable support sets and relationships through the use of the variables database. However, this component was designed to meet the needs of simulation environments. It was not designed as a solution to the problems of separating model specification from model operation. Knowledge about model structure is still integrated into the rules that contain knowledge about how to update and determine variable values and knowledge about variable support sets is still largely not represented separately in the RBCLM2. To ensure that models are self-contained specifications that can be interrogated for different tasks and that both models and RBCLM2 system can be easily revised and updated there is a need for further separation of knowledge.

As with the RBCLM1, the RBCLM2 KR scheme is not explicitly spatial but influences that are determined by spatial processes such seed flow into and out of sites can be easily included in an RBCLM2 model and RBCLM2 models can be linked up to other models within a spatial environment. For example, in the MODULUS RBCLM2 Natural Vegetation Model, state transitions were made dependent upon the amount of 'seeds' of different vegetation types present at a site (Engelen *et al.* 2000). The seed flow out of and into each site (vegetation type and amount) was determined by an external model and relevant exogenous variable facts asserted dynamically through the MOSE. The exogenous variable facts for seed were then used as conditions for state transition, effectively spatialising the model.

5.5.3 Reasoning and Simulation

The RBCLM2's main purpose is to simulate vegetation type dynamics at the community-level through reasoning about community-level vegetation and environmental attributes and properties. Vegetation state, as it is measured using sets of discrete values, quite naturally changes in discrete jumps when the conditions for a state transition are met. In the Capri model, various quantitatively measured variables such as the intermediate variable 'actual growth rate' and the state variable 'canopy height' are directly or indirectly dependent on vegetation state. This means that when vegetation state changes, the value of these variables can change quite dramatically, jumping from one value to another.

Whether such discrete jumps matter for model behaviour depends on the purpose of the model and the variables concerned. In some cases there may be legitimate concern over the accuracy (with respect to real world dynamics) of discretely jumping quantitative variable values. There may be a need for some models to include a smoothing function for quantitative variables dependent on non-quantitative variables to prevent sudden jumps in value but it is not clear that this is a core requirement for all, or indeed any, RBCLM2 models. Smoothing functions were not included in the RBCLM2 system for reasons of parsimony. It remains to be seen whether there is a definite need for them.

The method employed by the RBCLM2 system for calculating the value of intermediate variables is computationally inefficient. Intermediate variables are calculated as and when they are required. This means that, in any given time-step each intermediate variable may be calculated many times, an obvious piece of redundant computation. The effect of this inefficiency was felt in the run-time taken by the RBCLM2 component of the MODULUS system, where the time taken by the RBCLM2 model was far greater than for other models when spatial and temporal resolution and model complexity are taken into account (Engelen *et al.* 2000). A better approach would be to calculate each intermediate variable once per time-step and to use the stored value as and when necessary for other calculations. This method will be explored in the next Chapter.

Chapter 6 Modelling Vegetation Dynamics III – A Rule-Based Species Attributes and Properties Approach

6.1 System Basics

6.1.1 Introduction

The RBCLM2 System, although able to represent vegetation structure in greater detail than the RBCLM1, still treats all occurrences of each vegetation type as functionally identical. This may not be adequate with potentially important consequences for the soundness of any predictions made. As species composition and structure varies within a particular instance of a particular vegetation type, the way in which vegetation responds to influencing factors may also change. One main option exists to resolve this difficulty – to abandon the imposition of top-down regularities in vegetation behaviour by disaggregating the representation of vegetation structure.

In addition the RBCLM1 and RBCLM2 systems contain various problems relating to representation and reasoning. Neither system cleanly separates model specification from model computation. The RBCLM1 enforces a temporal reasoning system for modelling change in a linguistic variable, vegetation state. This reasoning system is capable of addressing both the direction and rate of change problems and coherently simulating discrete variable dynamics. The RBCLM2, although capable of representing knowledge in such a way as to emulate this system does not enforce a strict method for doing so. This is a weak point. The Rule-Based Community-Level Modelling 3 (RBCLM3) System aims to address the ontological issues concerned with the functional response of vegetation, to extend the RBCLM1's temporal reasoning system to cover other non-quantitative variable and to better separate knowledge representation from reasoning.

6.1.2 System Ontology and Model Structure

Ecological systems are increasingly being viewed from a bottom-up, complex systems perspective (Costanza *et al.* 1993, Judson 1994, Olson and Sequeira 1995). With respect to vegetation this is achieved through concentrating study on the species or population-level (Peet and Christensen 1980, Smith and Huston 1989, Pickett and Kolasa 1989, Burrows 1990, Glenn-Lewin and van der Maarel 1992, Peet 1992, Pickett *et al.* 1994). The processes and mechanisms that determine vegetation structure operate at the species- and individual-levels and a large but incomplete body of knowledge has been accumulated regarding the behaviours of different species and species types under different conditions. From this perspective vegetation structure and dynamics result from the dynamics of species populations. Species-level approaches to understanding vegetation do not explicitly proscribe what the structure or dynamics of vegetation will be for any situation. Instead, vegetation structure and dynamics are treated as 'emergent' properties produced by the interaction of lower-level system components – the species populations.

One way of viewing vegetation at the species-level is through the classification provided by phylogenetic taxonomy. Different species may interact with each other in predictable ways to create predictable vegetation structures and dynamics. If so, perhaps vegetation can be usefully described and understood purely in terms of interacting populations of taxonomically defined populations. Currently however there is no single agreed upon theory detailing how plant species interact to form vegetation structure (Pickett and Kolasa 1989, Smith and Huston 1989).

Alternatively it may be better to view species as fulfilling functional roles within vegetation and to look at how vegetation is structured and changes in terms of its functional components. To do so requires that species be characterised by biological function rather than by phylogeny.

It has been argued by many authors that a functional classification and description of species is essential to the development of our understanding of vegetation structure and dynamics (Botkin 1974, Noble and Slatyer 1980, Grime 1985, Smith *et al.* 1993, Weiher and Keddy 1995, Noble and Gitay 1996, Gillison and Carpenter 1997, Gitay and Noble 1997, Lavorel *et al.* 1997, Lavorel *et al.* 1999, McIntyre *et al.* 1999, Weiher *et al.* 1999). There are quite simply too many species to include in a comprehensive theory of vegetation. Reducing the complexity is necessary.

The essence of the functional approach involves using biological characteristics (also called traits or attributes) that fulfil important functional roles in vegetation processes to describe and classify species or groups of species. The identified functional roles and attributes and the relative and absolute sizes of populations of species exhibiting particular attributes or syndromes of attributes can then be used as a basis for examining, describing and predicting vegetation.

A distinction must be made here concerning species attributes and species properties. Species attributes are regarded here as functional responses to influencing factors. For example, Bond and van Wilgen (1996) identify an important species attribute for post-fire vegetation dynamics - vegetative response to fire disturbance. They distinguish two possible 'values' for this attribute – an ability to vegetatively sprout and an inability to vegetatively sprout. Some species can sprout and therefore survive fires whereas other species cannot sprout and are therefore killed by fires. All species have a particular vegetative response to fire, an attribute 'value' that is genetically determined and does not change over time. If a species is capable of post-fire vegetative sprouting then it will always be capable of doing so although it is possible that attribute values may vary with life-stage e.g. juveniles of a particular species may behave differently from adult plants.

Species properties, like community properties, on the other hand are quantities that change over time and which describe the current state of a species population e.g. biomass. The values of such quantities, although dependent on attributes, are neither necessarily constant nor species-specific.

The RBCLM3 System uses attributes and properties to model how species populations change over time in response to each other and influencing environmental factors. Community-level dynamics can then be handled by using a classification system to derive vegetation type from the species-level state. This provides the RBCLM3 with a means of modelling vegetation from the bottom up and of addressing the RBCLM1 and RBCLM2 problems of treating all occurrences of each vegetation type as functionally identical.

The RBCLM3 represents the world as a community (analogous to RBCLM1 and RBCLM2 'sites' they represent both biotic and abiotic elements *sensu* the 'community-level' of Heathfield 2001), an object that contains sets of variables to represent environmental properties and also a set of species objects each of which contain sets of variables to represent species attributes and properties (see Figure 63). Each community object can contain from zero to as many as is computationally feasible species objects. A variable to represent classified vegetation type can be placed in the community object but its inclusion is not mandatory for an RBCLM3 model.

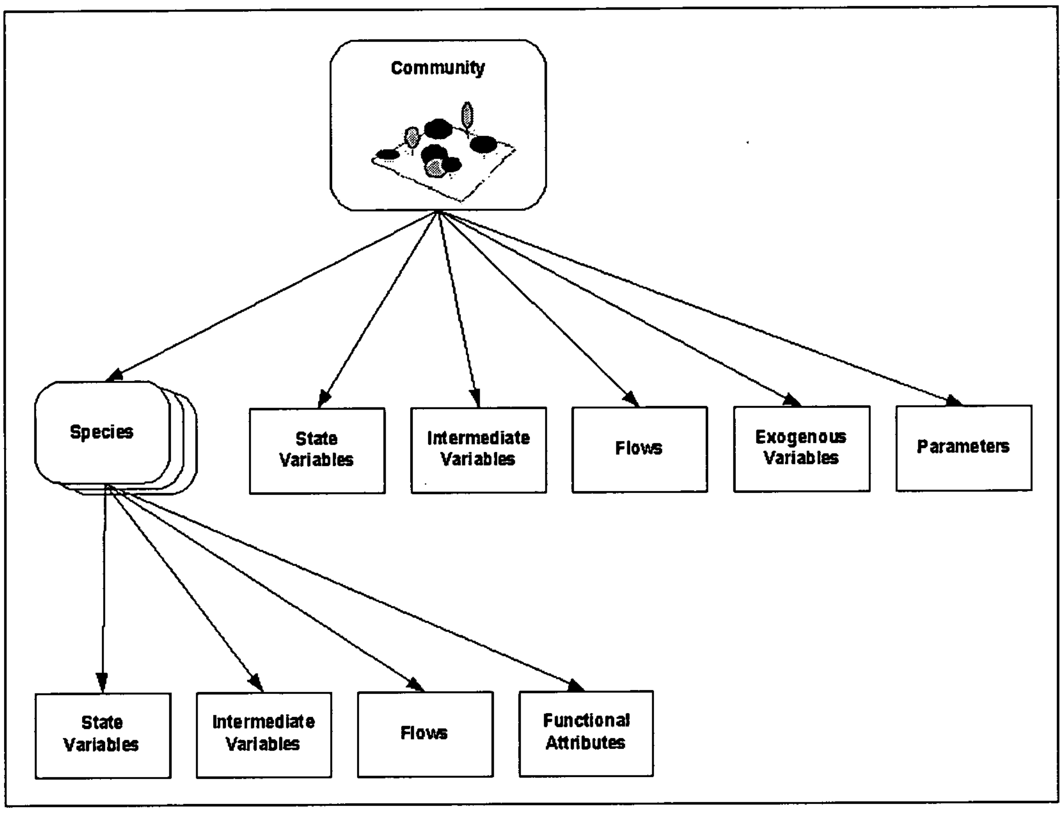


Figure 63 Informal RBCLM3 system ontology (rounded rectangles represent objects, multiple objects are represented by multiple overlapping rounded rectangles, normal rectangles represent variables or sets of variables, arcs represent 'has a' relationships)

Each community object possesses a set of variables that are grouped into 5 types:

- *State variables* (e.g. soil moisture).
- *Intermediate variables* (e.g. vegetation type, community flammability).
- *Flows:*

  The RBCLM3 was designed based on the ontology employed by compartment-flow or System Dynamics modelling (Hannon and Ruth 1994, Ford 1999, Deaton and Winebrake 2000) as it is relatively common ontology to use in ecology being well-suited to representing the storage and flow of matter – the so-called 'stuff' paradigm (Pickett *et al.* 1994). As such the ontological construct, the flow is used. Flows represent the absolute rate of change of state variables and although they can be considered to be a type of intermediate variable they are given special significance under the System Dynamics ontology - flows are the *only* variables permitted to directly influence the value of state variables. Intermediate variables that do not directly influence the values of state variables (although they may through effect propagation along a chain of influences) are not classified as flows but simply as intermediate variables instead.

Typical examples of flows could be inflow of soil water from precipitation or outflow of soil water from lateral flow.

- *Exogenous variables* (e.g. precipitation).
- *Parameters* (e.g. altitude, slope angle).

Each species object possesses a set of variables that can be grouped into 4 types:

- *State variables* (e.g. canopy height, biomass).
- *Intermediate variables* (e.g. amount of water available to a particular species).
- *Flows* (e.g. growth rate, biomass removal by fire).
- *Functional attributes* (e.g. palatability, reproductive mode).

One of the major differences between the RBCLM2 and RBCLM3 systems is with respect to their classification of variable types. The RBCLM2 KR scheme split variables into environment and vegetation then further by computational type (state, intermediate etc.). The RBCLM3 abandons the ontological distinction (vegetation or environment) to categorise purely by computational type. This is more generic and efficient for simulation – a state variable requires the same type of updating regardless of whether it is in a species or community object.

Just like the RBCLM1 and RBCLM2, the RBCLM3 is a modelling system providing a set of representational constructs and a reasoning system with which to build and run simulation models of vegetation dynamics. The RBCLM3 provides a unified temporal reasoning system like that of the RBCLM1 to model change in the value of *all* non-quantitative state variables. This is unlike either of the previous two systems and ensures that modelling with non-quantitative knowledge is performed coherently in a standard manner across all RBCLM3 models.

Within a single RBCLM3 model all species object instances possess the same set of variables although there is no enforcement of variables across different models. The same applies to community objects – no particular variables must be included in any given model. Beyond the constraints of the ontology and variable types described above the model developer is free to choose how to model, creating a flexible and general system for species-level modelling.

The RBCLM3 uses a type hierarchy (detailed later) to classify variables by type (state variable, flow etc. and whether they are scalars or arrays), classify the support sets used by model variables by type (linguistic, qualitative etc.) and define the species objects to be used. A set of factual statements is used to represent model structure (the variables included in the model, the objects to which they belong and the causal influences between them) and to define variable support sets. A set of factual statements and 'if ... then' rules is used to represent knowledge concerning the relationships between species (or species types), other species and environmental influences.

Object-specific rules are used to represent knowledge about how intermediate variable values are calculated whilst another set of object-specific rules detail how state variables change value. If a state variable is quantitative then knowledge concerning how it may change value is represented using a single rule. If a state variable is qualitative or linguistic then knowledge concerning how it may change value is represented using two sets of rules – one that represents knowledge concerning direction of change (D-Delta) and one that represents knowledge concerning the time required to change value under current conditions (T-Delta). In addition a set of factual statements are used to define correspondences between non-quantitative variable values within objects as necessary.

6.1.3 Simulation Overview

Before a simulation run commences the specified model structure is checked for correctness with respect to variable dependencies. This step should reduce model design error. If the model is found to be non-circular the run proceeds first by determining the calculation order of all the flows and intermediate variables involved. The calculation orders are stored and used every time-step to control the order in which the intermediate variables are calculated for efficiency reasons. The model objects (community plus the species that it contains) are then initialised and the simulation run begins. These steps will be described later on.

The RBCLM3 does not have a pre-defined length time-unit. A typical time-unit is 1 month and a single time-step is always equal to a single time-unit. At the start of each time-step during a simulation the values of flows and intermediate variables are calculated in order. The calculated values are then stored for use during the time-step in updating the values of state variables. Quantitative state variables are represented using single values and updated using one set of rules. Non-quantitative state variables are represented using a tuple of variables representing value, D-Delta, T-Delta and T-In, and updated using several sets of rules. One set of rules is applied to determine D-Delta (if the variable is qualitative then this can be either increasing, decreasing or steady and if the variable is linguistic then D-Delta equals the value that the variable is changing towards) whilst another is applied to determining T-Delta (under current conditions). Once these have been determined the current value for T-Delta is compared with the current value for T-In. If T-In ≥ T-Delta then the state variable value is updated and if T-In < T-Delta the value remains the same. If the variable is qualitative updating proceeds by changing the value of the variable to that of the support set element to the right of the present one (if the variable's D-Delta is increasing) or to the left (if the variable's D-Delta is decreasing). For example, if a qualitative variable value is low, increasing and T-In ≥ T-Delta, and its support set is {low, medium, high}, then its value will change to the support element to the right of low i.e. medium. If the variable is linguistic updating proceeds by changing the value of the variable to that indicated by the value of D-Delta.

In addition to state variables the RBCLM3 can represent and reason with quantitative and non-quantitative value and relationship knowledge concerning any other type of variable (flows, intermediate variables, parameters, functional attributes, exogenous variables).

6.1.4 System Architecture

The System is composed of 4 logical modules as shown in Figure 64. All modules are hardwired and must contain appropriate information for an RBCLM3 model to run.

6.1.4.1 Reasoning System

The core of the RBCLM3. It is responsible for:

- Checking model dependency circularity.
- Determining flow and intermediate variable calculation order.
- Initialising objects and variables.
- Controlling variable calculation and updating during simulation runs.
- Reasoning about change in quantitatively and non-quantitatively valued state variables.
- Keeping track of simulation time.
- Storing state variable values between time-steps.
- Producing output.

It contains range of built-in predicates for use in reasoning with quantitative and non-quantitative variables. Some of these constitute the RBCLM3 temporal reasoning system and as such are always used. Others provide functionality that can be used as and when appropriate to represent and reason with particular pieces of knowledge.

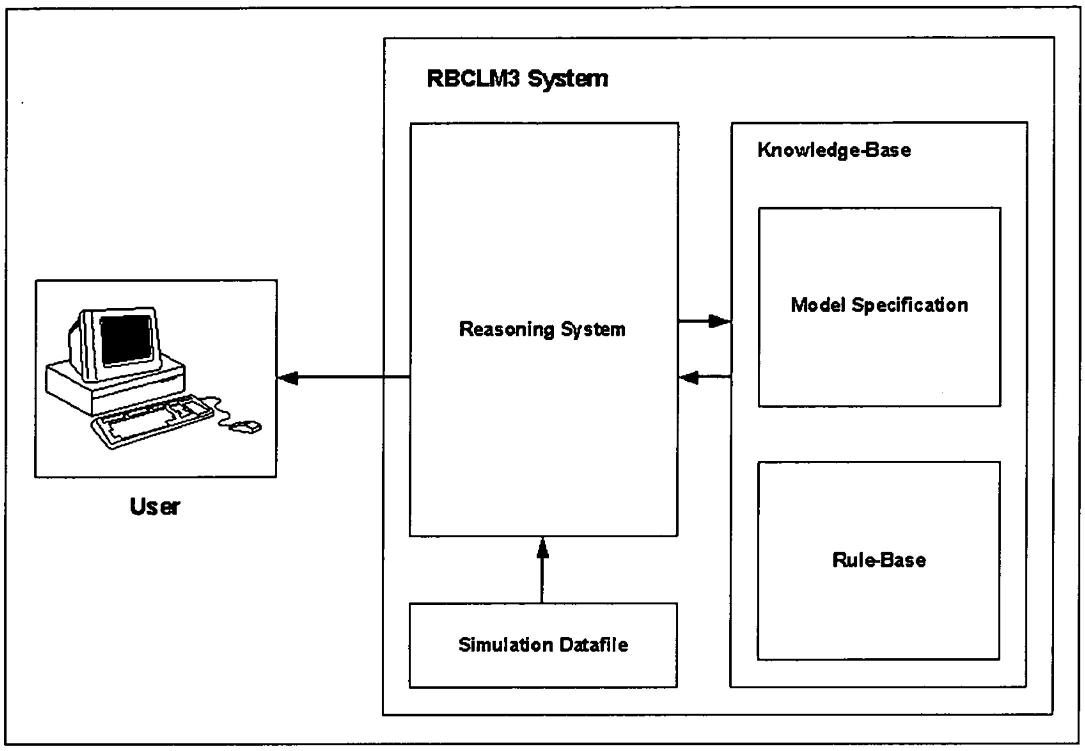


Figure 64 RBCLM3 System architecture and operation (arcs represent information flows resulting from predicate queries)

6.1.4.2 Knowledge-Base

Each KB details the structure, value and relationship knowledge that constitutes an RBCLM3 model of vegetation dynamics. The KB component is interchangeable to permit different parameterisations (same structure but different rule formulations) of the same model to be explored and also to allow different models (different structures) to be explored.

Model Specification

The model specification component contains a declarative representation of an RBCLM3 model's structure and value knowledge. The names of variables, their support sets, how they are related to other variables and to which objects they belong is specified. In addition the RBCLM3 type hierarchy for the model in question is specified in this component. The type hierarchy sorts the model variables into variable and value types, the support sets to be used by the model variables into types and details the objects to be used in the model e.g. the names of the species.

Rule-Base

The rule-base contains a declarative representation of relationships between variable values in the form of rules, organised into sets by type of variable and by the object and possibly instance to which they apply. Rules for T-Delta, D-Delta and intermediate variable values are contained here.

6.1.4.3 Simulation Datafile

Like the corresponding RBCLM1 and RBCLM2 components the simulation datafile provides initial values for state variables (for all objects), values for all model constants (parameters and functional attributes) and values for all exogenous variables for the course of a simulation run.

6.2 Knowledge Representation

6.2.1 Knowledge Representation Scheme – Introduction and Notation

The next 3 sections detail the RBCLM3 KR scheme predicates and how they can be used for representing available ecological knowledge regarding species-level dynamics. Table 39 details the argument notation used across these predicates. The notation is sometimes used in slightly varied forms for different purposes but the meaning should still be obvious. Additional notation, if not immediately obvious in meaning, will be detailed as necessary.

Argument Name	**Typical Value**	**Prolog Term Classification when Instantiated**	**Description**
`CommNo`	`1`	Constant (number)	Community object instance number.
`Destination`	`fire_sev`	Constant (atom)	The dependent variable in a functional relationship between variables.
`Entity`	`biomass`	Constant	A particular model entity e.g. named variable, support set etc.
`FlowList`	`[(precip, community, 1, low), (water_use, community, 1, medium)]`	Data structure (list)	A list of flows influencing a state variable – both names and values.
`InfValList`	`[(flamma, community, 1, high)]`	Data structure (list)	A list of influencing variable names and values.
`ObjectInst`	`(grass,1)`	Constant (number) or constant (atom) or compound term	Object instance identifier, usually employed for denoting species object instances and the community objects to which they belong.

Argument Name	Typical Value	Prolog Term Classification when Instantiated	Description
`ObjectName`	`species`	Constant (atom)	Object name – either 'community' or 'species'.
`Sort`	`model_entity`	Constant (atom)	A particular sort or type of objects.
`Source`	`flammability`	Constant (atom)	An independent variable in a functional relationship between variables.
`SupportSet`	`[zero, low, medium, high]`	Data structure (list)	The support set of a variable.
`TCurr`	`12`	Constant (number)	Current simulation time
`ValueTuple`	`[medium, 1, decr, 8]`	Data structure (list)	The tuple of values that are used to represent state variables – value, T-In, D-Delta and T-Delta.
`VarName`	`Biomass`	Constant (atom)	The name of a variable.
`VarVal, Val` *or* `Value`	`medium`	Constant (atom) or constant (number)	The value of a variable.

Table 39 RBCLM3 KR scheme argument notation

6.2.2 Knowledge-Base: Model Specification

The RBCLM3 maintains a clear distinction between the different types of knowledge that comprise a model – structural knowledge, value knowledge, relationships knowledge and reasoning. The Model Specification component serves as a vehicle for the representation of model structure, variable value types and classification knowledge regarding different sorts of model object. Within each Model Specification component the following knowledge is specified:

- A type hierarchy applied to the model concerned.
- A list of model variables by object.
- Details of the model variable dependency structure.
- Details regarding model variable support sets.

6.2.2.1 Type Hierarchy

The type hierarchy used by the RBCLM3 is shown in Figure 65. Classifying different elements of RBCLM3 models into types and sub-types permitted the Reasoning System to be designed in a more generic manner. Rather than specifying how to reason with many specific variables or support sets, different model variables, support sets etc. can be classified into different types. Reasoning can then be carried out on the basis of the types. This is particularly of use in situations where the representational and reasoning requirements for different types of things in a model are dissimilar. In the RBCLM1 there were only a few state variables (vegetation state, T-Delta and T-In) so no type hierarchy was required. In the RBCLM2 the representation and reasoning was largely left to the model developer. As discussed this can lead to model development errors. By 'automating' various reasoning tasks according to type in the RBCLM3 the chances of error should be reduced and the task of model design facilitated.

The type hierarchy used was developed based on the requirements that:

- Different types of variable require different types of reasoning e.g. state variables must be updated and their values stored between time-steps whereas intermediate variables only require calculation once per time-step.
- Variables with different types of support set require different means of representation and reasoning e.g. non-quantitative state variables are represented using a name plus a 4 value tuple composed of current value, T-In, D-Delta and T-Delta.
- Checking for circularity in variable dependency need only look at intermediate variables and flows. The RBCLM1 and RBCLM2 did not check model structure for correctness. Due to the potentially greater complexity of an RBCLM3 model (many multiple species) the need for some means of checking structural correctness was felt necessary to reduce error.
- Scalar variables only have one value to calculate or update whereas array variables can have many. Due to the RBCLM3 modelling with multiple species objects there is a need for array variables to be used to gather together values for variables across all species.
- For each community there may be many species objects to reason about but not vice versa.

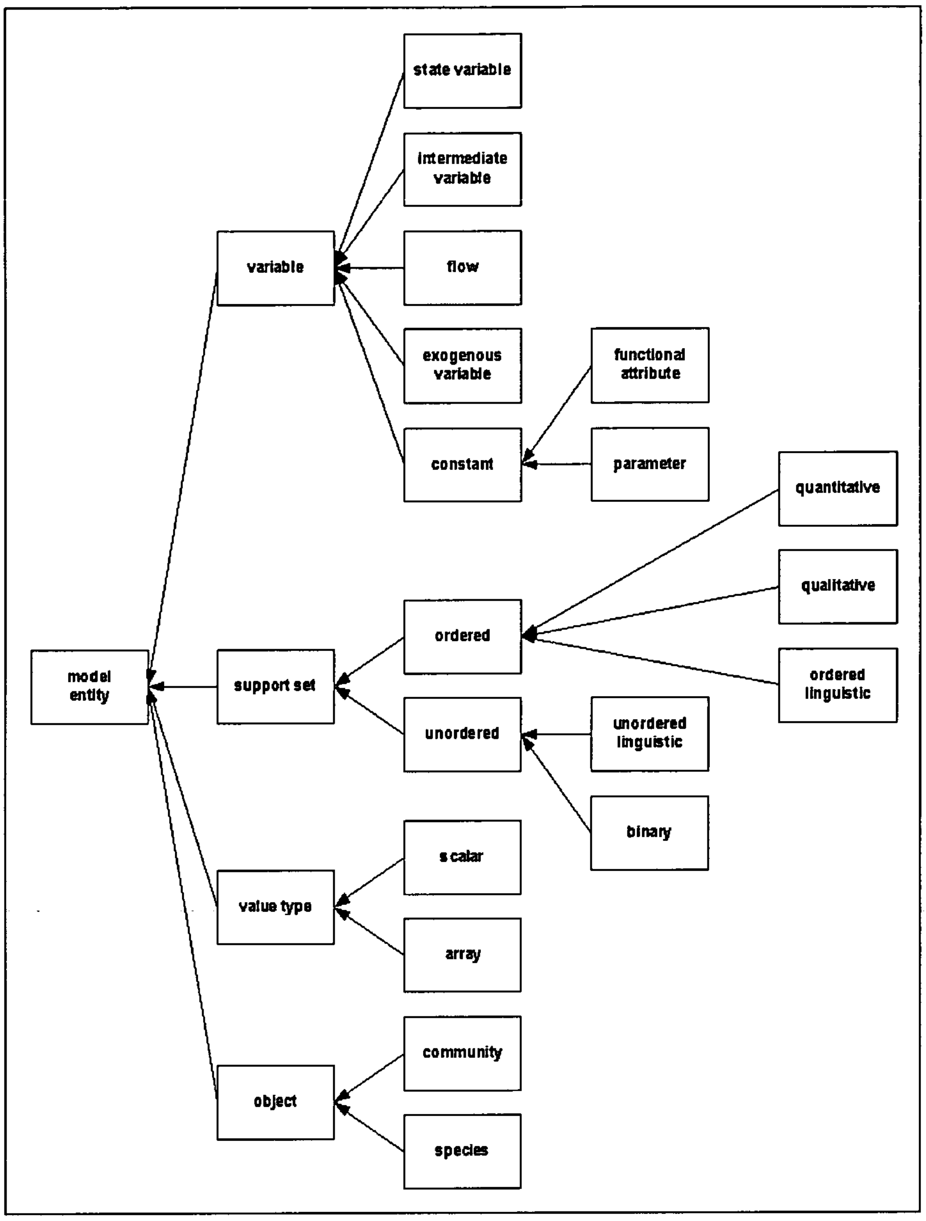


Figure 65 RBCLM3 type hierarchy (nodes represent types of model entity, arcs represent 'is a type of' relations)

In addition, it was decided that various features such as model structure interrogation and automatic documentation generation may be useful to add in the future. Designing the RBCLM3 around a type hierarchy potentially reduces the amount of re-engineering that may be required.

For any given model, the model developer must 'expand' or instantiate the type hierarchy depicted by providing details on the model entities that belong to each type. The types to be instantiated are the nodes on the extreme right-hand side of each branch in the type hierarchy (Figure 65). There is no need to instantiate every type of model entity with a set of entities from a given model. For example, if a model includes no arrays then there will be no entities contained within the array type of value type. The model developer must use the predicate detailed in Figure 66 to instantiate types.

`typeof(Type,Entity).`

Example 1:

```
typeof(flow,growth).
```

which can be read as,
'growth' is a type of 'flow'.

Example 2:

```
typeof(qualitative,[low,medium,high]).
```

which can be read as,
[low, medium, high] is a type of qualitative support set.

Figure 66 RBCLM3 `typeof` predicate (based on the `subobj` predicate of Robertson *et al.* 1991)

6.2.2.2 Model Structure

KR Scheme

Model structure can be thought of as the variables being used and the dependencies between them. In the RBCLM1 and RBCLM2 there was no separate and explicit representation of model structure. Instead dependencies were expressed as the conditions of rules used to calculate and update variables. The RBCLM3 separates out structure to facilitate model design (separation makes it easier to ensure the desired structure is used) and tasks like circular dependency checking. Four compulsory predicates are provided to specify model structure (see Figures 67 – 70):

- `has_var:`
  To represent knowledge concerning the variables possessed by each object.
- `infl:`
  To represent knowledge concerning dependencies between all variable types except state variables, which can only be influenced by flows (see below). Influences from state

variable to other variables should be represented using `infl` but influences from flows to state variables should be represented using `inflow` and `outflow` (below).

- `inflow:`

  To represent dependencies between state variables and inflows, flows that add to state variables or represent positive components of the rate of change of state variables.

- `outflow:`

  To represent dependencies between state variables and outflows, flows that subtract from state variables or represent negative components of the rate of change of state variables.

`has_var(Object,VarName).`

Example:

```
has_var(species,max_biomass).
```

which can be read as,
The object species has a variable called 'max_biomass'.

Figure 67 RBCLM3 `has_var` predicate

`infl(Source,Destination).`

Example:

```
infl(biomass,mortality).
```

which can be read as,
The variable 'biomass' influences the value of the variable 'mortality'.

Figure 68 RBCLM3 `infl` predicate

`inflow(Source,Destination).`

Example:

```
inflow(growth,biomass).
```

which can be read as,
The variable 'growth' is an inflow for (and therefore an influence on) the state variable 'biomass'.

Figure 69 RBCLM3 `inflow` predicate

```
outflow(Source,Destination).
```

Example:

```
outflow(smoi_use,smoisture).
```

which can be read as,
The variable 'smoi_use' (soil moisture use) is an outflow for (and therefore an influence on) the state variable 'smoisture' (soil moisture level).

Figure 70 RBCLM3 `outflow` predicate

Visualisation of Model Structure

Although not a formal part of the RBCLM3 KR scheme, before representing knowledge using the predicates and structures detailed in Figures 67 – 70, it is useful to design models using a graphical representation. Modelling systems such as Simile and STELLA (High Performance Systems Inc.) utilise a graphical language and user interface to facilitate model development. The use of a graphical representation can help reduce structural error by clarifying the dependencies between variables. For most people it is easier to structure a complex model visually than it is by using a set of rule-based or mathematical constructs. This may involve anything from simple influence diagrams to the use of complex graphical notation. Once a graphical design has been drafted the process of turning this into a formal and computable model can begin.

RBCLM3 model KR elements can be directly mapped onto the visual language of Simile and, to a lesser extent, that of STELLA. The RBCLM3 ontology and KR scheme are essentially a form of object-oriented compartment-flow modelling. Figure 71 depicts an RBCLM3 model using the graphical notation employed by Simile.

In Figure 71 the rounded rectangular boxes represent objects. Where there are many such boxes stacked on top of each other this indicates multiple instances for that object. With respect to the RBCLM3, there are only two types of object – communities and species. Within each community there may be many species instances represented using the stacked object notation. Locating the species objects inside the community objects indicates that species belong to the community. Each species instance will be structurally identical so structure need only be depicted once. However this does not mean that each species object will behave according to the same rules.

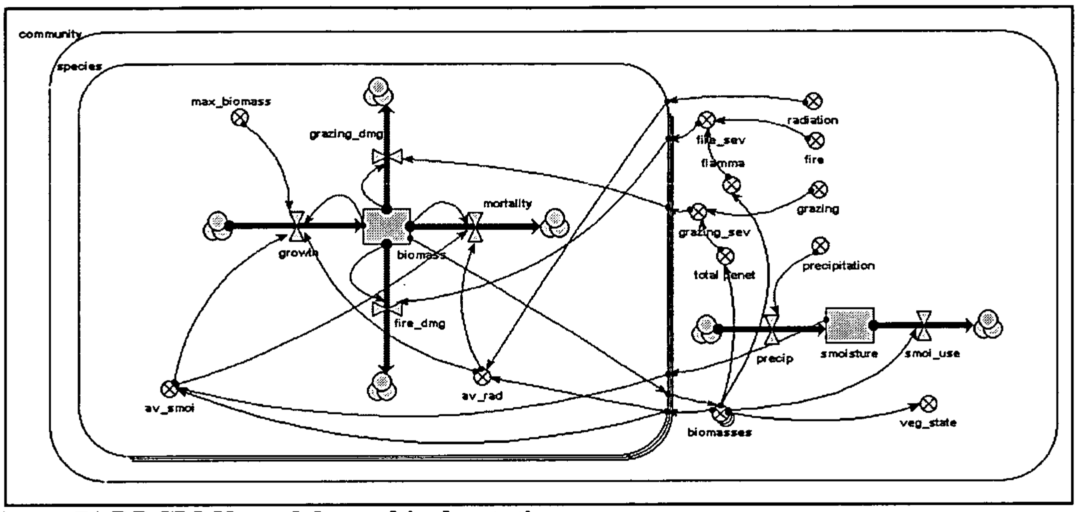


Figure 71 RBCLM3 model graphical notation

Model variables are represented using various symbols connected by influence arrows:

- *Crossed circle*:

  Used to represent intermediate variables, functional attributes, parameters and exogenous variables. If the symbol is not influenced then it is an exogenous variable, a parameter or a functional attribute. If it is influenced then it is an intermediate variable.
- *Thick arrow with 'cloud':*

  Used to represent a flow either into or out of a state variable. Only flows should directly connect to state variables as they are the only variables that can influence their values.
- *Rectangles with sharp edges:*

  Used to represent all state variables. Can influence the values of flows and intermediate variables but can only be influenced by flows.
- *Thin arrows:*

  Influence arrows. Used to represent causal relationships between variables with causality indicated by the direction of the arrowhead.

To design a model graphically the developer can use the graphical user interface provided by Simile. There is no formal link from Simile to RBCLM3 yet. In all cases the notation and rules detailed above and in Figure 71 should be adhered to. In doing so, the possibility of error during the formal process of rule construction and knowledge specification should be reduced.

6.2.2.3 Variable Support Sets

As well as structural knowledge the Model Specification component contains declarative knowledge concerning how model variables are valued. The support sets used to specify legal variable values should be represented using the predicate `value` (see Figure 72).

```
value(VarName,SupportSet).
```

Example:

```
value(growth,[zero,low,medium,high]).
```

which can be read as,
The variable 'growth' has the support set [zero,low,medium,high].

Figure 72 RBCLM3 `value` predicate

The explicit and separate representation of support sets was needed to permit the Reasoning System to utilise a general method for simulating change in non-quantitative state variables. This method will be described later.

6.2.3 Knowledge-Base: Rule-Base

The Rule-Base component specified the relationship knowledge for a model. This knowledge is first of all split into that concerned with calculating flows and intermediate variables and that concerned with updating state variables. Within these divisions the knowledge is then further split by value type – scalar or array. The need for the two value types is discussed below.

6.2.3.1 The Intermediate Variable and Flow Value Calculation Rule-Base

A Definition of Flows and Intermediate Variables

The RBCLM3 distinguishes between intermediate variables and flows. Both flows and intermediate variables are defined as variables whose values at some time t are not directly dependent upon their values at some previous time $t - 1$. Flows are distinguished however, for reasons of representational clarity within the RBCLM3, as being the only variables that directly influence the values of state variables. With respect to differential calculus, a flow can be viewed as any of the major positive or negative terms on the right-hand side of a differential equation.

To a certain extent the difference between flows and intermediate variables is purely cosmetic. However, distinguishing between flows and intermediate variables provides a useful way of improving the clarity of knowledge representation and model structure by making the developer explicitly think and represent influences on state variables in an separate way from other intermediate variables. This can draw attention to the special role that flows play in updating the value of state variables as opposed to only being intermediate steps in a chain of calculations. In addition, with respect to graphical representation of model structure, the notational separation of flows and intermediate variables can improve diagrammatic clarity (see Figure 71). To maintain consistency this notational separation should also be reflected in knowledge representation.

Calculating Scalar Intermediate Variables and Flows

If intermediate variables or flows only have a single value, that is if they are scalar variables, then knowledge concerning how their values can be calculated algebraically (if they are valued using a quantitative support set) or how they correspond to (combinations of) other variable values should be represented using the predicate `calc_scalar` (see Figure 73). This predicate can be used to do the same job as the predicates `veg_property`, `env_property`, `disturbance_level` and `time_counter` in the RBCLM2. Representing knowledge using a computational categorisation of variables makes the RBCLM3 KR scheme simpler and easier to understand.

The RBCLM3 does not impose any particular way of calculating intermediate variables or flows beyond the use of the predicate `calc_scalar`. Rules should be based on available ecological knowledge and reflect model purpose. As with the RBCLM2, various standard structures and methods can be employed, particularly value correspondences. The same scenarios and methods for handling them detailed in section 5.2.3 also apply to the RBCLM3 although they should all be represented using the one predicate, `calc_scalar`.

Both examples of `calc_scalar` shown in Figure 73 illustrate the use of value correspondences for representing non-quantitative relationship knowledge. Example 1 details a community object variable. Although the RBCLM3 reasoning system can only simulate one community (i.e. aspatial simulation) the reasoning system is designed such that it could be used by an external spatial simulation engine to simulate multiple communities. The argument 'CommNo' is included to facilitate this use of the RBCLM3 much like the 'Site' argument in RBCLM1 and RBCLM2.

```
calc_scalar(ObjectName,ObjectInst,VarName,InfValList,VarVal).
```

Example 1:

```
calc_scalar(community,CommNo,veg_state,InfValList,forest):-
    get_infdetails(biomasses,community,CommNo,InfValList,InfValL
    ist2),
    extract_arrayval((tree,CommNo),InfValList2,value_tuple(TBio,
    _,_,_)),
    gt_eqtl(biomass,TBio,medium).
```

which can be read as,
IF the value of tree biomass (TBio) within a community is greater than or equal to medium **THEN** the vegetation state (veg_state) of that community will be forest.

Example 2:

```
calc_scalar(species,(grass,CommNo),growth,InfValList,high):-
       get_infdetails(av_rad,species,(grass,CommNo),InfValList,AR
       Val),
       get_infdetails(av_smoi,species,(grass,CommNo),InfValList,A
       SMVal),
       ((gt_eqtl(av_rad,ARVal,medium),
       gt_eqtl(av_smoi,ASMVal,low));
       (gt_eqtl(av_rad,ARVal,low),
       gt_eqtl(av_smoi,ASMVal,medium))).
```

which can be read as,
IF (the value of radiation available to the species 'grass' within a community is greater than or equal to medium **AND** the value of soil moisture available to the species 'grass' within a community is greater than or equal to low) **OR** (the value of radiation available to the species 'grass' within a community is greater than or equal to low **AND** the value of soil moisture available to the species 'grass' within a community is greater than or equal to medium) **THEN** the growth of the species 'grass' in that community will be high.

Figure 73 RBCLM3 `calc_scalar` predicate

Example 2 details a species object variable. Species objects are identified using a Prolog compound term argument of the form:

(SpeciesName,CommNo)

This permits different species to be differentiated and modelled within a single community and also supports the extension of the RBCLM3 to the simulation of multiple communities.

Both examples use various convenience predicates as conditions. These convenience predicates exist to help represent and reason with ecological knowledge and can be used as conditions within rules for calculating intermediate variables or flows or for updating state variables. The predicates can be grouped into 2 types – those concerned with comparing variable values and those concerned with values of influencing variables.

Convenience predicates for comparing a variable's current value with its range of legal values:

- `eqt1:`
  Qualitative 'equal to' operator used to determine whether a qualitative variable is equal to a particular value from its own support set. Compare with `eqt2` detailed in Figure 74.
- `gt1:`
  Qualitative 'greater than' operator used to determine whether a qualitative variable value is greater than a particular element from its support set.
- `gt2:`
  Qualitative 'greater than' operator used to determine whether a qualitative variable value is greater than another qualitative variable value based on correspondences between variable values represented using `eqt2` (see Figure 74 below).
- `gt_eqt1:`
  Qualitative 'greater than or equal to' operator used to determine whether a qualitative variable value is greater than or equal to a particular value from its support set.
- `lt1:`
  Qualitative 'less than' operator used to determine whether a qualitative variable is less than a particular value from its support set.
- `lt2:`
  Qualitative 'less than' operator used to determine whether a qualitative variable value is less than another qualitative variable value based on correspondences between variable values represented using `eqt2` (see Figure 73 below).
- `lt_eqt1:`
  Qualitative 'less than or equal to' operator used to determine whether a qualitative variable is less than or equal to a particular value from its support set.
- `maximum:`
  Predicate that can be used to determine whether an ordered non-quantitative variable value is the maximum (i.e. right-most) according to its support set.
- `minimum:`
  Predicate that can be used to determine whether an ordered non-quantitative variable value is the minimum (i.e. left-most) according to its support set.

Convenience predicates concerned with the values of influencing variables:

- `extract_arrayval`

  Can be used to extract the value of a influencing variable stored as an element in an array.

- `get_infdetails`

  Can be used to extract the value of a named influence from a list of variables that influence the variable being calculated.

In addition to the convenience predicates above, a further predicate is provided for model developers to represent knowledge concerning qualitative equality relations between the values of different model variables:

- `eqt2`

  Qualitative 'equal to' operator used to represent correspondence between the values of two different qualitative variables using the same measurement units (see Figure 74). Correspondence here is used to mean 'equivalence in value'. `eqt2` is used as part of the reasoning involved in the built-in qualitative operator predicates `gt2` and `lt2`. The predicate can be useful if the values of two or more qualitative flows need to be compared. For example, biomass will increase if growth is greater than mortality. To establish the truth of this some means of representing how different qualitative values of each flow correspond to each other is required.

Note that `eqt2` does not specify the objects to which the variable value correspondence applies. The model developer must take care to use `eqt2` appropriately as a condition in rules for calculating or updating variable values. It would not be sensible for example to draw an equivalence between low growth rate for a tree species and low growth rate for a grass species.

`eqt2(VarName1,VarVal1,VarName2,VarVal2).`

Example:

```
eqt2(mortality,low,growth,low).
```

which can be read as,
The value low for the variable mortality represents the same amount of biomass as the value low for the variable growth does.

Figure 74 RBCLM3 `eqt2` predicate

Array Intermediate Variables

The RBCLM3 only has limited capability with respect to representing and manipulating array variables. Array variables in the RBCLM3 are designed to reside in the community object and gather together the values for species object variables. They are essentially constructed repositories for data under the current RBCLM3 formulation.

Why are they included in the system? The RBCLM2 did not use multiple objects – it had a 'flat' structure. An RBCLM3 model may contain multiple objects and it will more than likely be necessary for them to communicate with one another (e.g. to model competition between species). Array variables represent information that is only available at the level of the community (i.e. values for all species) and as such array variables should be placed within community objects via the `has_var` predicate. This could be done by giving each species direct access to the necessary variables from all species objects. However this kind of web-like influencing can easily become messy, error-prone and an inefficient way to represent and reason with knowledge. A more representationally and computationally efficient method of doing the same job is to gather all the information on each variable of interest into a single multi-valued variable, an array variable.

Array variables are constructed by gathering together the values of a variable across all species objects present in a community into a single data structure. To construct an array variable the model developer need only specify the following in the Model Specification component:

- The name of the array variable and that the variable is possessed by the community object using the predicate `has_var`.
- That the variable is of the value type 'array' by including a `typeof` clause `typeof(array,VarName)` where `VarName` should be set equal to the name of the array variable concerned.
- An influence from the species variable from whose values it should be constructed using the variable `infl`.

From this information, the RBCLM3 reasoning system will use a built-in predicate `build_array` to construct the array variable by putting all the species variable values into a single Prolog list structure that can then be used in rule conditions via the predicate `extract_arrayval` (to be detailed in section 6.3).

6.2.3.2 The State Variable Rule-Base

The RBCLM3 can represent state variables with different types of support set in different ways and this reflects upon the way in which their values are stored and updated. As with the intermediate variables and flows, the RBCLM3 does not impose the inclusion of any particular state variables nor does it impose the inclusion of any particular dependency structure. Essentially the model developer can include whatever variables are deemed appropriate and make their values depend on any other variables as deemed appropriate. This is similar to the RBCLM2 approach although the RBCLM3 does not distinguish between vegetation and environment state variables – a computational categorisation based on support set type is used instead. However unlike the RBCLM2, the RBCLM3 does enforce a specific method for representing knowledge and reasoning about change in non-quantitative state variable values. A single method is enforced to ensure that non-quantitative state variables are reasoned with coherently and to minimise the risks of model development error. This differs quite significantly from the RBCLM2 where, beyond the fact that a particular predicate must be used in KR, there were few constraints on the ways in which state variables could be updated. In any state variable update rule, regardless of support set, the same convenience predicates for conditions as detailed previously can be used (`ltl`, `get_infdetails` etc.).

Quantitative State Variables

Quantitative state variables are represented using a single value and a name. They are updated in a simple but very flexible manner. Few constraints are imposed upon the model developer when representing knowledge concerning how quantitative state variables change value except that, following the System ontology, state variables should only be influenced by inflows and outflows. As with the RBCLM2, various standard structures and methods can be employed, particularly value correspondences. The same scenarios and methods for handling them detailed in section 5.2.2 also apply to quantitative state variable knowledge representation in the RBCLM3 although they should all be represented using the one predicate, `update_var` (Figure 75)

Example 1 in Figure 75 details a quantitative state variable changing value depending on the value of two flows, an inflow and an outflow, both of which are valued using quantitative support sets. Example 2 details a quantitative state variable dependent on two flows, an inflow and an outflow, both of which are valued using qualitative support sets.

```
update_var(ObjectName,ObjectInst,VarName,FlowList,VarVal1,VarVa
l2).
```

Example 1:

```
update_var(species,(grass,1),height,FlowList,VarVal1,VarVal2):-
   get_infdetails(growth,species,(grass,1),FlowList,GVal),
   get_infdetails(mortality,species,(grass,1),FlowList,MVal),
   VarVal2 is VarVal1 + GVal - MVal.
```

which can be read as,
The value of the variable height for the grass species in community object 1 will be updated from its current value (VarVal1) to a new value (VarVal2) using the equation VarVal2 = VarVal1 + value for growth – the value for mortality.

Example 2:

```
update_var(species,(shrub,1),biomass,FlowList,VarVal1,VarVal2):
-
   get_infdetails(growth,species,(shrub,1),FlowList,GVal),
   get_infdetails(mortality,species,(shrub,1),FlowList,MVal),
   eqtl(growth,GVal,low),
   eqtl(mortality,MVal,medium),
   VarVal2 is VarVal1 + 5 - 15.
```

which can be read as,
IF the value of the growth flow for the shrub species in community object 1 **AND** the value of the mortality flow for the shrub species in community object 1 is medium **THEN** the value of the variable height for the grass species in community object 1 will be updated from its current value (VarVal1) to a new value (VarVal2) using the equation VarVal2 = VarVal1 – 10.

Figure 75 RBCLM3 `update_var` predicate

Note that the equation used to update biomass in this example could have been simplified with the inflow growth and the outflow mortality lumped together into a single net influence of minus 10 units / unit area on biomass. It is perfectly acceptable to represent the effect of a combination of non-quantitative flow values as a single numerical change in value.

Non-Quantitative State Variables: Background

For non-quantitative state variables the RBCLM3 enforces a specific method for representing knowledge and reasoning about change in the value. This approach differs significantly from the KR scheme and reasoning system employed by the RBCLM2.

A major problem in updating non-quantitative, and hence discretely valued, state variables is knowing when to change a state variable from one discrete value to another i.e. the *rate of change* problem. Techniques like envisionment involve abandoning the ability to predict what will happen at particular points in time. For the purposes of predictive modelling of vegetation dynamics this is not a useful option as discussed in Chapter 3. Being able to predict how vegetation will change over a specified time-period is more useful for theory and management. The temporal reasoning system employed by the RBCLM1 was shown to be of use in modelling with a linguistic state variable. The RBCLM2 does not enforce any particular method putting a greater onus on the model developer to ensure that non-quantitative state variables are handled coherently and soundly. This creates too high a risk of error. Instead the RBCLM3 employs a form of temporal reasoning. The method used is essentially the method used for reasoning about change in vegetation state in the RBCLM1 extended and generalised to reasoning about change in the value of any discretely valued state variable i.e. qualitative, ordered linguistic and unordered linguistic support sets.

To understand the KR scheme for non-quantitative state variables an understanding of the basic principles involved in the updating method is required. A full description of the reasoning method will be presented in section 6.3.

Each non-quantitative state variable is represented by a tuple of values stored as a Prolog list during simulation runs:

[VarVal, T-In, D-Delta, T-Delta]

where,

VarVal =	the current non-quantitative value of the state variable e.g. medium.
T-In =	the length of time (integer number of time-units) that the variable has been at its current value under the current direction of change (D-Delta) by the end of the current time-step.
D-Delta =	the current direction of change of the variable as determined by the values of the variable's influences.
T-Delta =	the time required to change value (integer number of time-units) based upon the current D-Delta and the value's of the variable's influences.

This differs from the RBCLM1 where D-Delta, T-Delta and T-In were stored and calculated as separate variables. This suited the RBCLM1 as there would only ever be 1 state variable per community. However, with the RBCLM3 there could be *n* species objects within each community, each modelled using many state variables. It is more computationally and representationally efficient to view each variable's value components (T-Delta, T-In etc.) as part of a single structure than as 4 separate variables.

For qualitative and linguistic variables D-Delta can take on one of four possible values (Table 40).

D-Delta Value	Interpretation for Qualitative Support Set Variables	Interpretation for Ordered Linguistic Support Set Variables
incr	The variable is increasing in value i.e. has a positive direction and rate of change	The variable is changing towards the support set value that is immediately to the right of its current value
std	The variable has no direction of change and a steady value	The variable has no direction of change and a steady value
decr	The variable is decreasing in value i.e. has a positive direction and rate of change	The variable is changing towards the support set value that is immediately to the left of its current value
zero	The variable is decreasing such that it will reach its lowest possible value within a single time-step	The variable is decreasing such that it will reach the left-most value in its support set within a single time-step

Table 40 D-Delta Values and Interpretation for Variables with Non-Quantitative Ordered Support Sets

The first three D-Delta values are included to represent gradual directional change and stability in variable values. The fourth D-Delta value is included to denote sudden catastrophic decrease in a variable value such as might occur to a species' biomass if a community is affected by a severe fire disturbance. It is usually accompanied by a T-Delta value of 0, although not necessarily so.

For unordered linguistic variables D-Delta can be any value from the variable's support set. At any given point in time D-Delta is set equal to the value that the state variable will take on next assuming that conditions stay constant. This use of D-Delta is exactly the same as in the RBCLM1 where a site's vegetation state, a variable valued using an unordered linguistic support set, was given a D-Delta value equal to the state towards which it was developing. It does not make sense to talk about a variable valued using an unordered set of linguistic values as increasing, decreasing or remaining steady. There are no ordering relations that hold between the elements of an unordered linguistic support set so changes in value must be reasoned about case by case in terms of current value and next value under different sets of conditions. Unordered linguistic variables can, by their very nature, change value in a non-linear (unordered) manner according to their specific dynamics. Change in value does not need to proceed from one element to an adjacent element.

Non-quantitative variables change value in a direction determined by the values of their influences. As well as determining D-Delta, the values of each non-quantitative variable's influences also determine the rate of change, how quickly it will take for the variable to actually reach the new value as indicated by D-Delta. Much like the RBCLM1, the RBCLM3 System employs the notion of time required to change value or T-Delta to handle rate of change in non-quantitative variables. T-Delta is valued using integers and the symbol 'n/a' if there is no direction of change. A T-Delta value of 0 denotes instant change in a variable value. This usually accompanies a D-Delta of zero. For different combinations of influence values under each possible D-Delta for a non-quantitative variable a T-Delta rule is used to represent how long it will take for the variable to change value by one support set element.

As a simulation run proceeds the value for T-Delta is determined and updated by repeated application of T-Delta rules along with an averaging algorithm employed by the reasoning system (see section 6.3). As long as a variable's D-Delta stays at the same value, its time in state (under current D-Delta) or T-In counter will increment by one every time-step. If D-Delta changes then T-In will be reset. When the state variable's T-In is greater than or equal to the current value for its T-Delta its value will change as indicated by its D-Delta value – either right or left along its support set one element (for qualitative or ordered linguistic variables) or to the support set element specified by the current D-Delta value (for unordered linguistic variables).

Non-Quantitative State Variables: Direction of Change Rule-Base

Updating non-quantitative state variables involves representing knowledge concerning how they change value under different conditions (D-Delta) and at what rate (T-Delta). Two sets of rules stored within the Rule-Base component of the KB are used for this purpose. The first set of rules are concerned with determining D-Delta under all possible combinations of influencing variable values. For qualitative and ordered linguistic variables there will be at least 3 D-Delta rules, one for each of the possible directions 'incr', 'std' and 'decr'. For unordered linguistic variables there is no minimum or maximum number of D-Delta rules – the number of rules to be used depends on the number of elements in the variable's support set and the number of ways in which the variable can change value.

Figure 76 details the predicate `d_delta_rule` that is used to represent D-Delta knowledge for all non-quantitative state variables. Examples are given for both community and species object state variables. When constructing `d_delta_rule` clauses the same convenience predicates detailed for intermediate variables and flows may be used as conditions e.g. `lt2`, `get_infdetails` etc.

Non-Quantitative State Variables: Time Required to Change Value Rule-Base

The second set of rules used to represent knowledge concerning how non-quantitative state variables change value are concerned with determining T-Delta. Knowledge concerning how long each variable takes to change value under every possible D-Delta value and combination of influencing variable values is represented using a set of T-Delta rules. Each T-Delta rule applies to a specified D-Delta value. There will be at least one T-Delta rule for each possible value of D-Delta although, where several combinations of influencing variable values apply to a single D-Delta value, but each combination results in a different rate of change (T-Delta), there may be more than one T-Delta rule for each D-Delta value.

Figure 77 details the predicate `t_delta_rule` that must be used to represent knowledge about non-quantitative variable T-Delta values. The examples shown follow on from Figure 76.

6.2.4 Simulation Datafile

6.2.4.1 Introduction

The model developer can use the Simulation Datafile to represent knowledge regarding the initial values of state variables, the values of any parameters or functional attributes to be used for a model and the values over time for any exogenous variables to be included. Mandatory-use predicates are provided for all these tasks.

6.2.4.2 State Variable Initialisation

Before a simulation run can begin each state variable must be initialised. The predicate `init_var_details` (see Figure 78) should be used to represent the initial values for any state variable regardless of object and support set.

d_delta_rule(ObjectName,ObjectInst,VarName,FlowList,Ddelta).

Example 1:

```
d_delta_rule(community,CommNo,smoisture,FlowList,incr):-
      get_infdetails(precip,community,CommNo,FlowList,PVal),
      get_infdetails(smoi_use,community,CommNo,FlowList,SMUVal),
      lt2(smoi_use,SMUVal,precip,PVal).
```

which can be read as,
IF the value of soil moisture use (smoi_use) is less than the value of precipitation (precip) within a given community object **THEN** the community state variable soil moisture (smoisture) will be increasing in value for that community object.

Example 2:

```
d_delta_rule(community,CommNo,smoisture,FlowList,decr):-
      get_infdetails(precip,community,CommNo,FlowList,PVal),
      get_infdetails(smoi_use,community,CommNo,FlowList,SMUVal),
      gt2(smoi_use,SMUVal,precip,PVal).
```

which can be read as,
IF the value of soil moisture use is greater than the value of precipitation within a given community object **THEN** the community state variable soil moisture will be decreasing in value for that community object.

Example 3:

```
d_delta_rule(species,(Inst,CommNo),biomass,FlowList,decr):-
      get_infdetails(growth,species,(Inst,CommNo),FlowList,GVal)
      ,
      get_infdetails(mort,species,(Inst,CommNo),FlowList,MVal),
      get_infdetails(g_dmg,species,(Inst,CommNo),FlowList,GDVal)
      ,
      get_infdetails(f_dmg,species,(Inst,CommNo),FlowList,FDVal)
      ,
      (gt2a(mort,MVal,growth,GVal);
      gt2a(g_dmg,GDVal,growth,GVal);
      eqt1(f_dmg,FDVal,low)).
```

which can be read as,
IF the value of a species mortality flow is greater than the value of its growth flow **OR** the value of that species' grazing damage flow is greater than the value of its growth flow **OR** the value of that species' fire damage flow is equal to 'low' **THEN** that species biomass will be decreasing.

Figure 76 RBCLM3 `d_delta_rule` predicate

`t_delta_rule(ObjectName,ObjectInst,VarName,DDelta,FlowList,TDelta).`

Example 1:

```
t_delta_rule(community,CommNo,smoisture,decr,FlowList,0):-
      get_infdetails(precip,community,CommNo,FlowList,PVal),
      get_inf_details(smoi_use,community,CommNo,FlowList,SMVal),
      eqtl(precip,PVal,zero),
      eqtl(smoi_use,SMVal,high).
```

which can be read as,
IF the value of precipitation (`precip`) is equal to zero **AND** the value of soil moisture use (`smoi_use`) is equal to high within a given community object **THEN** the community state variable soil moisture (`smoisture`) will have a T-Delta value of 0 for the D-Delta value `decr` (i.e. take 0 years to change one element to the left along its support set in value).

Example 2:

```
t_delta_rule(community,CommNo,smoisture,decr,FlowList,1):-
      get_infdetails(precip,community,CommNo,FlowList,PVal),
      get_inf_details(smoi_use,community,CommNo,FlowList,SMVal),
      (eqtl(precip,PVal,low);
      eqtl(precip,PVal,medium)),
      eqtl(smoi_use,SMVal,medium).
```

which can be read as,
IF (the value of precipitation is equal to low **OR** medium) **AND** the value of soil moisture use is equal to medium for a given community object **THEN** the community state variable soil moisture will have a T-Delta value of 1 for the D-Delta value `decr`.

Example 3:

```
t_delta_rule(species,(grass,CommNo),biomass,,decr,FlowList,4):-
      get_infdetails(gro,species,(grass,CommNo),FlowList,GVal),
      get_infdetails(mort,species,(grass,CommNo),FlowList,MVal),
      get_infdetails(gdmg,species,(grass,CommNo),FlowList,GDVal)
      ,
      eqtl(gro,GVal,high),
      ((gtl(mort,MVal,medium),
      gtl(gdmg,GDVal,low));
      (gtl(mort,MVal,low),
      gtl(gdmg,GDVal,medium))).
```

which can be read as,
IF a community's grass species' growth flow is equal to high **AND** (its mortality flow is greater than medium **AND** its grazing damage flow is greater than low) **OR** (its mortality flow is greater than low **AND** its grazing damage flow is greater than medium) **THEN** that grass species biomass will take 4 years to decrease by 1 value element (i.e. T-Delta = 1).

Figure 77 RBCLM3 `t_delta_rule` predicate

```
init_var_details(ObjectName,ObjectInst,VarName,ValueTuple).
```

Example 1:

```
init_var_details(community,1,smoisture,value_tuple(medium,3,std,
na)).
```

which can be read as,
The initial value for the variable soil moisture (`smoisture`) in community 1 is VarVal = medium, T-In = 3, D-Delta = std, T-Delta = n/a.

Example 2:

```
init_var_details(species,(grass,1),biomass,value_tiple(medium,2,
incr,10)).
```

which can be read as,
The initial value for the variable biomass for the species 'grass' in community 1 is VarVal = medium, T-In = 2, D-Delta = incr, T-Delta = 10.

Figure 78 RBCLM2 `init_var_details` predicate

Note how in Example 1 (above) the value of T-Delta is 'n/a' due to the fact that D-Delta = std. The variable is not changing value so the concept of T-Delta does not apply.

6.2.4.3 Parameterisation

RBCLM3 models have two types of constant – parameters and functional attributes. As both types of variable are computationally static a single predicate is provided by the RBCLM3 to specify their values for a simulation run (see Figure 79). Example 1 in Figure 79 depicts the specification of a parameter (community object constant) whilst example 2 depicts the specification of a functional attribute (species object constant). The distinction between attributes and parameters is therefore semantic.

6.2.4.4 Exogenous Variable Specification

In the RBCLM3 exogenous variables are always community object variables and must have their values specified for each time-step before a simulation run begins. This is achieved through the use of the predicate `var_details2` (see Figure 79).

The examples show how a repeating annual pattern in the exogenous variable radiation can be represented. The examples make use of a built-in RBCLM3 convenience predicate called `fmod`, which, in this case, divides the current time-step value (an integer) by 12. `fmod` only accepts integer results for the division and returns the integer remainder from the division. Given the fact that there are 12 months in a year, this creates a repeating pattern from 0 to

11, representing months 1 – 12 of the year (see Table 41). Exogenous variable values can be made conditional upon this monthly number, rather than explicitly upon time-step numbers. This reduces redundancy and makes it easier to specify exogenous variable values for a simulation run.

`var_details1(ObjectName,ObjectInst,VarName,VarVal).`

Example 1:

```
var_details1(community,1,slope_angle,steep).
```

which can be read as,
The value of the variable slope angle for community number 1 is ‘steep’.

Example 2:

```
var_details1(species,(shrub,1),max_biomass,high).
```

which can be read as,
The value of the variable maximum biomass for the species ‘shrub’ in community number 1 is high.

Figure 79 RBCLM2 `var_details1` predicate

`var_details2(community,CommNo,TCurr,VarName,VarVal).`

Example 1:

```
var_details2(community,CommNo,TCurr,radiation,medium):-
        fmod(TCurr,12,X),
        (X=<4;
        X>=7).
```

which can be read as,
The value of the exogenous variable radiation is medium outwith months 5 and 6 of every year (outwith June and July, *where* month 0 = January and 11 = December).

Example 2:

```
var_details2(community,CommNo,TCurr,radiation,high):-
        fmod(TCurr,12,X),
        (X==5;
        X==6).
```

which can be read as,
The value of the exogenous variable radiation is high for months 5 and 6 of every year (June and July, *where* month 0 = January and 11 = December).

Figure 80 RBCLM2 `var_details2` predicate

Time-Step (TCurr)	`fmod(TCurr,12)` Return Value	Explanation of fmod Return Value
0	0	0/12 = 0 integers with a remainder of 0
6	6	6/12 = 0 integers with a remainder of 6
11	11	11/12 = 0 integers with a remainder of 11
12	0	12/12 = 1 integer integers with a remainder of 0
18	6	18/12 = 1 integer integers with a remainder of 6
23	11	23/12 = 1 integer with a remainder of 11
24	0	24/12 = 2 integers with a remainder of 0

Table 41 Use of `fmod` to generate a 12 element repeating sequence from time-step values

6.3 Reasoning System

6.3.1 Introduction

The RBCLM3 Reasoning System component is responsible for reasoning with each KB to determine and update model variable values for different model objects, checking model structural correctness and executing time-driven simulations for a single community at a time. Section 6.1.2.2 provides a fuller list of responsibilities for the component. The RBCLM3 System enforces a particular method for reasoning about change in non-quantitative variables. As such it is more than a just an algorithm to control the order in which variables are calculated.

Figure 81 details the operation of the RBCLM3 Reasoning System when simulating a community. The Reasoning System controls simulation runs in 3 main stages:

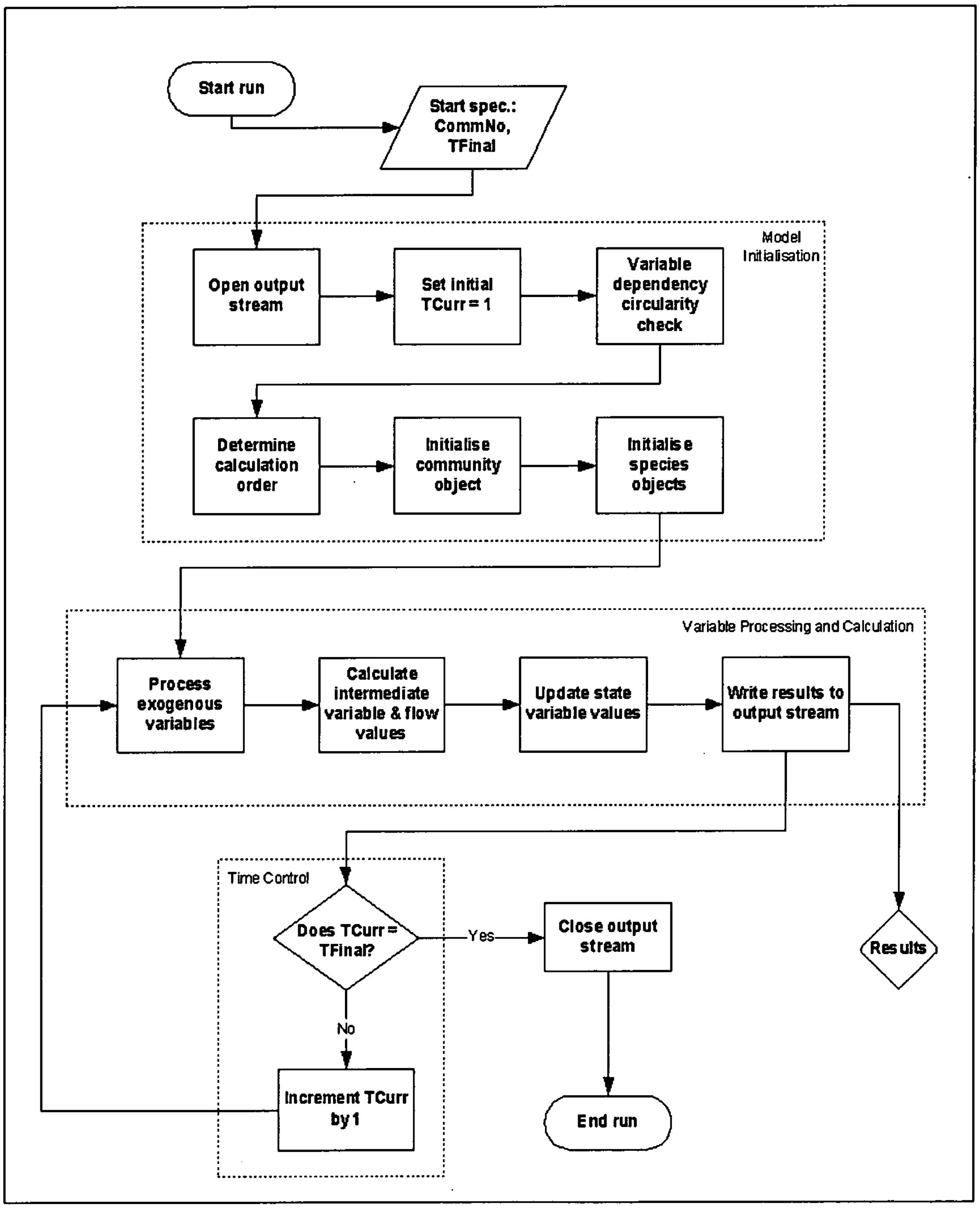
- *Model Initialisation*:

  Given the number of the community to simulate and the run-length as input, the RBCLM3 Reasoning System proceeds to initialise the model. This involves checking for model structural correctness, determining the order of calculation for model intermediate variables and flows and initialising state variables in the community and species objects to be simulated.
- *Variable Processing and Calculation:*

  Every time-step the Reasoning System is responsible for retrieving the values for exogenous variables from the Simulation Datafile, calculating intermediate variable and flow values in the order determined during initialisation and updating all model state variables.

- *Time Control:*

  This a simple step that involves incrementing the simulation clock time and determining when to terminate a simulation run.



Start run
Start spec.: CommNo, TFinal
Model Initialisation
Open output stream
Set initial TCurr = 1
Variable dependency circularity check
Determine calculation order
Initialise community object
Initialise species objects
Variable Processing and Calculation
Process exogenous variables
Calculate intermediate variable & flow values
Update state variable values
Write results to output stream
Time Control
Does TCurr = TFinal?
Yes
No
Close output stream
Results
Increment TCurr by 1
End run

Figure 81 RBCLM3 Reasoning System simulation operation

Simulation time-steps are set equal to model time-units, which, although not pre-defined (they depend on model purpose and available knowledge), will typically represent one month, possibly one year. RBCLM3 time-steps are always equal to 1 time-unit.

The RBCLM3 is designed to simulate one community object with as many species objects as desired and as is computationally feasible. As such, although a community represents an area of space, the RBCLM3 is effectively an aspatial modelling system. Community objects are given identification numbers to provide the opportunity to expand the system to spatial simulation.

The following two sections will detail the operation of an RBCLM3 simulation including how the System reasons with available knowledge to calculate variable values. The section after that will detail the built-in convenience predicates provided by the RBCLM3 for use as rule conditions.

6.3.2 Model Initialisation

6.3.2.1 Variable Dependencies – Circularity Checking

The RBCLM3 provides a means of constructing structurally complex models with many different variables spread over many different objects. To help ensure that models developed using the RBCLM3 are free from structural error and operate as intended, the Reasoning System carries out a check on model structural correctness before permitting a simulation run to proceed.

The general principle behind the updating of RBCLM3 models over time is 'calculate rate then update state'. This means that the all intermediate variables and flow values are calculated before any state variable is updated. This is the same as determining the right hand terms in a set of differential equations before integration. The need for a consistent mechanism for updating state variables was noted by Bugmann *et al.* (1996) who examined errors in behaviour caused by inconsistent methods for updating gap model state variables.

For intermediate variable and flow values to be calculated correctly requires that the dependencies between them are not circular i.e. making sure that any given flow or intermediate variable is not connected to itself by a chain of influences that does not contain a state variable (an integrating point). Before initialising a model the RBCLM3 Reasoning System checks the structure of a model for circular dependencies between intermediate variables and flows. It does this by constructing a list of all the intermediate variables and flows in a model then checking each one in turn for dependency circularity. If there is no circularity the model is deemed structurally correct and simulation can proceed. If circularity is detected an error message is produced detailing the circularity / circularities in structure and the run terminates. Appendix 2 details the algorithm used.

6.3.2.2 Calculation Order Determination

Once the structural correctness of a model has been verified the next stage of the initialisation process is to determine the order of calculation for all intermediate variables and flows. Intermediate variables and flow values can only be calculated once the values of their influences are known. If they are only influenced by state variables, exogenous variables, parameters or functional attributes (these variables always have known values) then there is no problem. Such intermediate variables and flows are classified as 'order 1' – they should be calculated first. Next come order 2 variables – those intermediate variables and flows that are dependent only on order 1 variables and known value variables. After that come order 3 variables and so on. When a simulation is running, the procedure for each time-step should involve the calculation of intermediate variables and flows from order 1 to order *n* before state variables are updated. The order in which these variables are calculated every time-step will not change over the course of a simulation so it makes for efficient computation to determine and store the calculation ordering once per run. Once determined and stored the calculation order list can be retrieved once per time-step and used to control when intermediate variables and flows are calculated. This is substantially different and more efficient from the 'on-the-fly' method employed by the RBCLM2. The RBCLM1 had no intermediate variables. Appendix 2 details the algorithm used.

6.3.2.3 Object Initialisation

Each community object may contain n_{cv} variables along with n_s species object instances each with n_{sv} variables. The `has_var` predicate in the KB Model Specification component details how many species objects there are, what they are called and which variables belong to species and which to community objects. All state variable values in all object instances need to be instantiated before a simulation run can commence. The RBCLM3 Reasoning System is responsible for this using its object initialisation algorithm.

Essentially, the Reasoning System initialises the community object being simulated by constructing a list of all the state variables that it contains based on the `has_var` facts. It then retrieves the values for these variables from Simulation Datafile `init_var_details` facts. The values for the variables are then stored in an object data structure. The object data structure is used as the means of internally maintaining state variable values for a model during a simulation. It was designed to be extensible, to pre-empt the possibility of making the RBCLM3 spatial with multiple communities each containing multiple species simulated at the same time. The form of the object data structure:

[(Object1, [(Inst1,[(Var1,Val1), ... (Var*n*,Val*n*)]), ... (Inst*n*)]), (Object2,[(Inst1,[(Var1,Val1), ... (Var*n*,Val*n*)]), ... (Inst*n*)]), ... (Object*n*)]

The data structure is capable of representing n_o objects, each with n_i instances each containing n_v variables represented using a tuple of name and value, where value will also be a tuple if the variable is non-quantitative.

After the community object has been initialised the Reasoning System compiles a list of the species contained within the community by searching the type hierarchy instantiated in the Model Specification component. It then constructs a list of species object state variables based upon `has_var` facts and retrieves their values for each species from `init_var_details` facts. These variable values are put into the object data structure as detailed above.

6.3.3 Variable Processing and Calculation

6.3.3.1 Exogenous Variable Processing

At the start of every time-step during a simulation the values for any exogenous variables must be instantiated. The Reasoning System does this by simply constructing a list of exogenous variables and retrieving their values from the database of `var_details2` facts and rules held in the Simulation Datafile. Some exogenous variable values may only apply to certain time-steps so the current time-step value is used as an argument (see Figure 80 and Table 41). The retrieved list of variables are then stored in a data structure for use during the time-steps processing. It should be noted that exogenous variable are always only community object variables. They are almost always climatic or other abiotic factors.

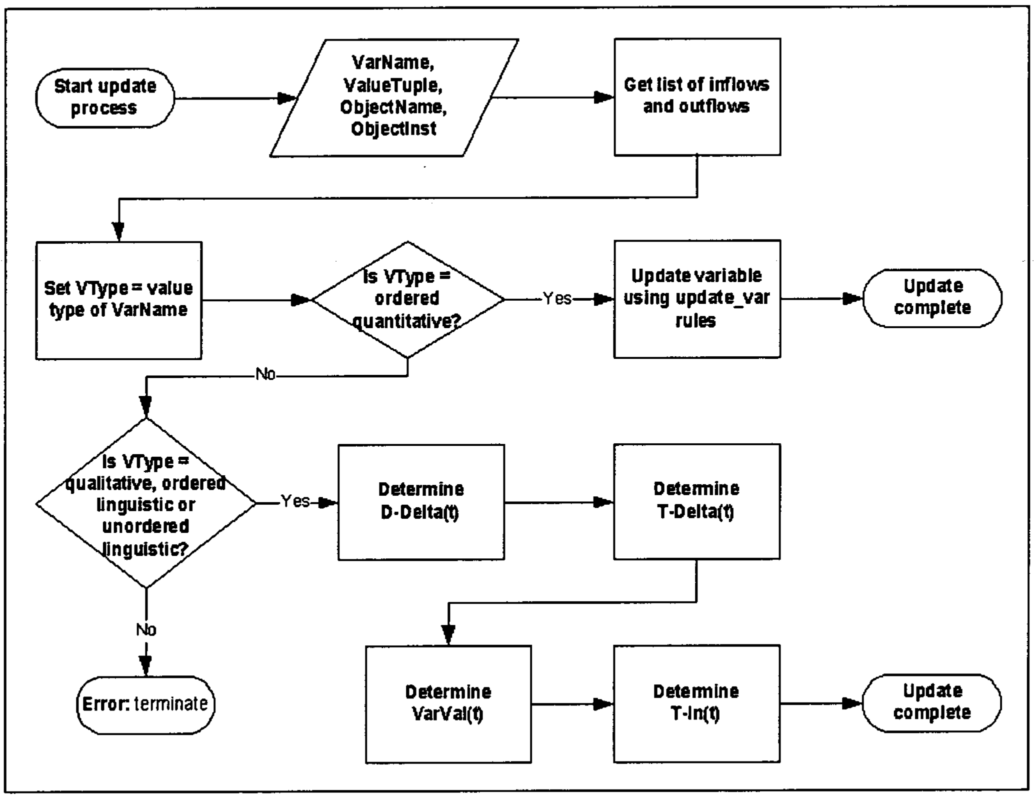
6.3.3.2 Intermediate Variable and Flow Value Calculation

Immediately after exogenous variable value processing every time-step, the Reasoning System calculates the values for all intermediate variables and flows. This step is the 'calculate rate' part of the pseudo-parallel 'calculate rate, update state' model update mechanism employed. The algorithm essentially involves recursing round the calculation order list (see section 6.3.2.2) calculating all intermediate variable and flows value in the order denoted by their calculation order.

For each community intermediate variable or flow only one value is calculated. For each species intermediate variable or flow the Reasoning System determines a value for each species object. The calculated variable values are stored in a data structure exactly the same as the object data structure detailed in section 6.3.2.3. At the end of the time-step the data structure is deleted and a new one constructed afresh during the next time-step.

6.3.3.3 State Variable Value Updating

Figure 82 details the process of updating a single state variable regardless of object. The algorithm is itself embedded within an algorithm that processes the object data structure detailed in section 6.3.2.3. The embedding control algorithm is responsible for ensuring that all state variables in all objects instances are updated and for updating the object data structure to reflect the changes in value that result.



Start update process
VarName, ValueTuple, ObjectName, ObjectInst
Get list of inflows and outflows
Set VType = value type of VarName
Is VType = ordered quantitative?
Yes
Update variable using update_var rules
Update complete
No
Is VType = qualitative, ordered linguistic or unordered linguistic?
Yes
Determine D-Delta(t)
Determine T-Delta(t)
No
Error: terminate
Determine VarVal(t)
Determine T-In(t)
Update complete

Figure 82 RBCLM3 state variable update algorithm

The single variable update process is quite simple but contains a number of sub-processes that will be detailed in the following two sections. Based upon the name of the variable to be updated, its value tuple and the object and instance to which it belongs, the Reasoning System constructs a list of influencing inflows and outflows based on `inflow` and `outflow` facts held in the RBCLM3 Model Specification component. It also determines the variable's support set type. If the support set type is quantitative then the variable's value will simply be updated by applying an `update_var` rule. If the value type is non-quantitative then, by implication, the variable's value type must be qualitative, ordered linguistic, unordered linguistic or binary. In this case the state variable is not just a name and a value but is represented by a value, T-In, D-Delta and T-Delta tuple. The method of reasoning about change in the value of such variables involves calculating and updating the values of D-Delta and T-Delta before reasoning about a change in value based on the values of T-In and T-Delta.

D-Delta Determination

D-Delta determines the way in which a non-quantitative variable is changing value. Figure 83 details the algorithm involved in determining a variable's D-Delta for a given time-step. The algorithm applies to all non-quantitative support set variables. It should be noted that D-Delta $_t$ is not dependent on D-Delta $_{t-1}$. It is effectively a type of intermediate variable.

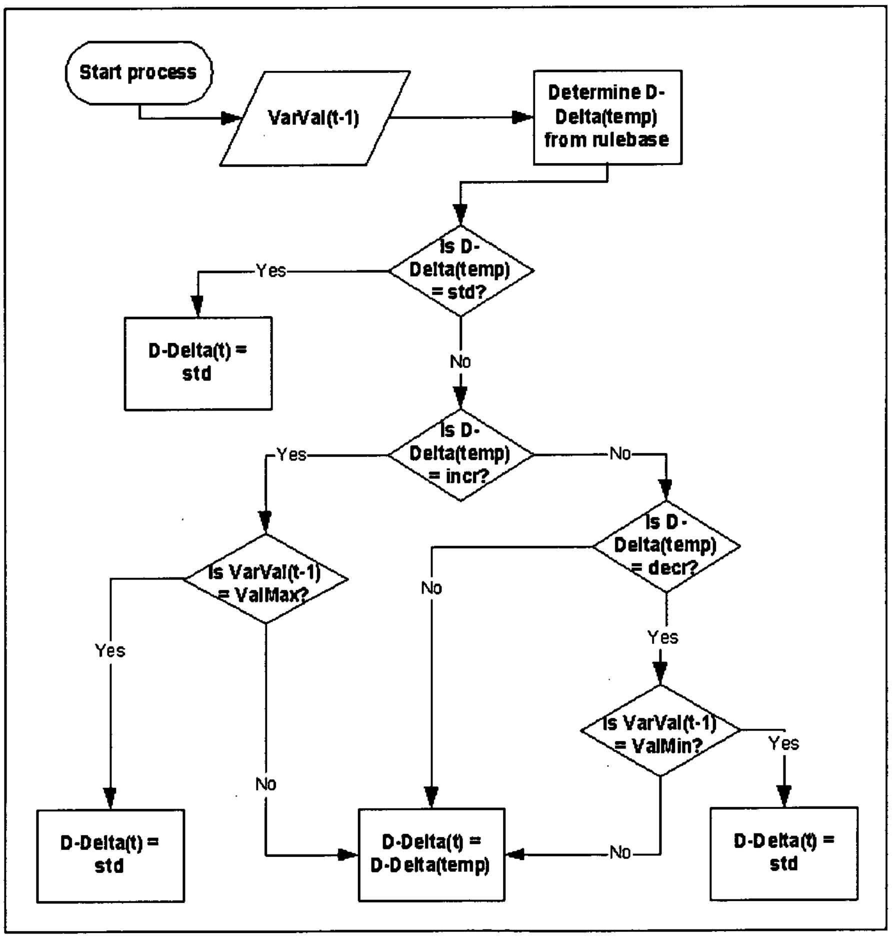


Figure 83 RBCLM3 D-Delta determination algorithm (see text for explanation)

The process first of all involves determining a temporary current value for D-Delta (D-Delta $_{temp}$) based on the `d_delta_rules` rule-base. If D-Delta $_{temp}$ is 'std' then D-Delta $_{t}$ (the actual current value for D-Delta) is set equal to 'std'. If not, extra rules are used. Note that only qualitative and ordered linguistic support set variables can have a D-Delta value of std.

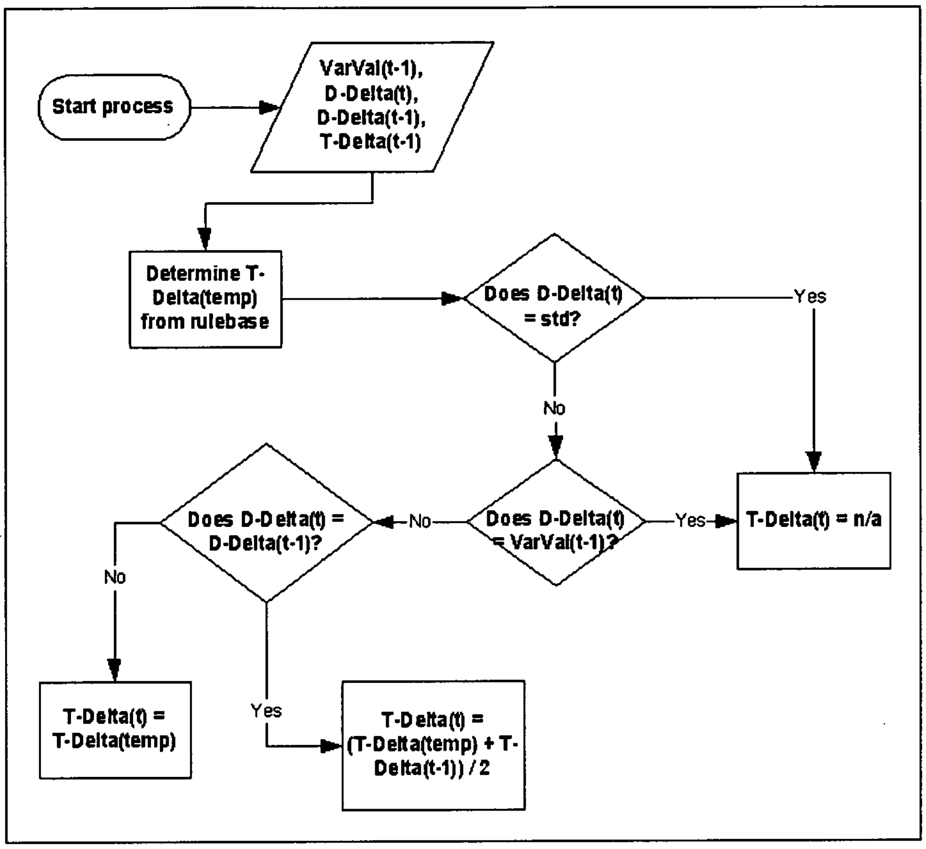
The next few decision points involve examining combinations of the previous time-step's value for the variable (VarVal $_{t-1}$) and D-Delta $_{temp}$. Two distinct points on qualitative and ordered linguistic support sets are distinguished to do this. The value element that is farthest right along the support set is termed the maximum value for the variable (ValMax). The value element that is farthest left along the support set is termed the minimum value for the variable (ValMin). Note that these concepts do not apply to variables valued using unordered linguistic support sets. If D-Delta $_{temp}$ is 'incr' and VarVal $_{t-1}$ is equal to the variable's maximum then D-Delta $_{t}$ is set equal to 'std' to represent the fact that the variable's value cannot go any higher. If D-Delta $_{temp}$ is 'decr' and VarVal $_{t-1}$ is equal to the variable's minimum then D-Delta $_{t}$ is set equal to 'std' to represent the fact that the variable's value cannot go any lower.

If neither of these cases is true then D-Delta $_{t}$ is set equal to D-Delta $_{temp}$. This will necessarily be the case for unordered linguistic variables for which the D-Delta values 'incr', 'std' and 'decr' do not apply. It will also be true if D-Delta is 'zero' or if qualitative and ordered linguistic variables are increasing in value but not at their maximum, or decreasing in value but not at their minimum.

T-Delta Determination

T-Delta is the RBCLM3 equivalent of numerical rate of change. It determines how quickly and exactly when changes in non-quantitative variable values will occur. The algorithm for updating a variable's T-Delta value is shown in Figure 84. T-Delta values are like state variables – they are updated and may be directly dependent on their previous value.

For a given variable a temporary value for T-Delta (T-Delta $_{temp}$) is first determined by application of `t_delta_rules`. If D-Delta $_{t}$ is equal to 'std' (qualitative and ordered linguistic variables only) or if D-Delta $_{t}$ is equal to the VarVal $_{t-1}$ (unordered linguistic variables) then T-Delta $_{t}$ is set equal to 'n/a'. This indicates that the concept of T-Delta is not applicable under current conditions as the variable concerned does not have a direction of change.



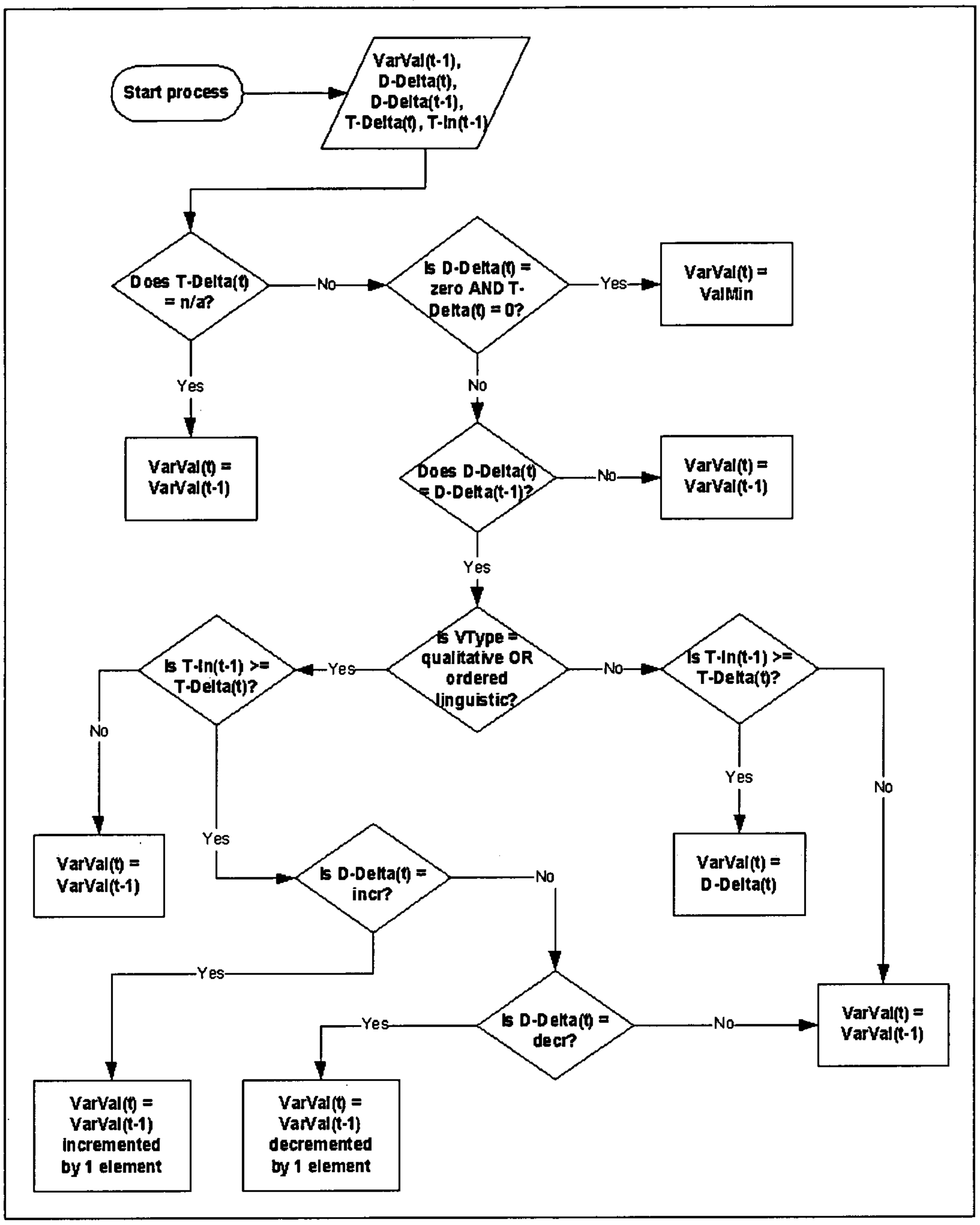
Start process
VarVal(t-1), D-Delta(t), D-Delta(t-1), T-Delta(t-1)
Determine T-Delta(temp) from rulebase
Does D-Delta(t) = std?
Yes
No
T-Delta(t) = n/a
Does D-Delta(t) = VarVal(t-1)?
Yes
No
Does D-Delta(t) = D-Delta(t-1)?
No
Yes
T-Delta(t) = T-Delta(temp)
T-Delta(t) = (T-Delta(temp) + T-Delta(t-1)) / 2

Figure 84 RBCLM3 T-Delta calculation algorithm (see text for explanation)

If T-Delta is not set equal to 'n/a' then the variable has a direction of change. If the direction of change has changed from that of the previous time-step (D-Delta $_t \neq$ D-Delta $_{t-1}$) then T-Delta $_t$ must be calculated afresh so T-Delta $_t$ is set equal to T-Delta $_{temp}$. If, however, the variable's direction of change for the current time-step is the same as for the previous time-step then T-Delta $_t$ is set equal to the average of T-Delta $_{t-1}$ and T-Delta $_{temp}$. This allows for the situation to occur where a variable's direction of change stays the same over successive time-steps but the rate of change alters.

Updating Variable Value

The purpose of calculating D-Delta and T-Delta is to determine when and how non-quantitative state variables change value over time. The algorithm in Figure 85 details how non-quantitative state variable values are updated based upon their previous value, T-In, D-Delta and T-Delta.



Start process
VarVal(t-1), D-Delta(t), D-Delta(t-1), T-Delta(t), T-In(t-1)
Does T-Delta(t) = n/a?
No
Yes
Is D-Delta(t) = zero AND T-Delta(t) = 0?
Yes
No
VarVal(t) = ValMin
VarVal(t) = VarVal(t-1)
Does D-Delta(t) = D-Delta(t-1)?
No
VarVal(t) = VarVal(t-1)
Yes
Is VType = qualitative OR ordered linguistic?
Yes
No
Is T-In(t-1) >= T-Delta(t)?
No
Yes
VarVal(t) = VarVal(t-1)
Is T-In(t-1) >= T-Delta(t)?
Yes
No
VarVal(t) = D-Delta(t)
Is D-Delta(t) = incr?
No
Yes
Is D-Delta(t) = decr?
Yes
No
VarVal(t) = VarVal(t-1)
VarVal(t) = VarVal(t-1) incremented by 1 element
VarVal(t) = VarVal(t-1) decremented by 1 element

Figure 85 RBCLM3 non-quantitative state variable value update algorithm

The algorithm uses the value of the variable concerned from the previous time-step, D-Delta $_t$ (just calculated), D-Delta $_{t-1}$, T-Delta $_t$ (just calculated) and T-In $_{t-1}$ in the process of updating value. This information is used first in three rules that determine how state variable value changes under special circumstances (see first three decision points in Figure 85). If the T-Delta is 'n/a' then, by inference, the variable has no direction of change and VarVal $_t$ is therefore set equal to VarVal $_{t-1}$. If D-Delta $_t$ is equal to 'zero' and T-Delta $_t$ is equal to 0 then conditions are such that an 'instantaneous' drop in the (qualitative or ordered linguistic) variable's value has occurred. When this occurs the variable value is set to the minimum value in its support set. If neither of these 'special case' rules apply a third rule is checked before the change in variable value is determined. The third rule is designed to help prevent non-quantitative variable values from changing too rapidly and at the wrong time-step. If a variable has changed direction of change (i.e. D-Delta $_t$ $\neq$ D-Delta $_{t-1}$) between the current and previous time-steps then it is not allowed to change value. A variable must have been changing value in a particular direction for at least one time-step before a value change is permitted to occur. Without this constraint non-quantitative state variables could change both D-Delta and value within the same time-step, potentially disrupting model behaviour.

If none of these rules apply the variable's support set is then tested. If the variable is valued using a qualitative or ordered linguistic support set then a series of further rules are tested. If the variable has not been at its current value under its current direction of change for long enough (i.e. if T-In $_{t-1}$ < T-Delta $_t$) then its value does not change. If however T-In $_{t-1}$ $\geq$ T-Delta $_t$ then the variable will change value in some way. Given that the variable will change value, if the variable's D-Delta $_t$ = 'incr' then the value of the variable will be incremented. This means that the variable's value will be set equal to the support set element immediately to the right of the element that represents the current value. If D-Delta $_t$ = 'decr' then the value of the variable will be decremented by one element i.e. set equal to the support set element immediately to the left of the one that represents the current value. As a catchall rule to handle exceptions, if the variable's T-In $_{t-1}$ $\geq$ T-Delta $_t$ but D-Delta $_t$ $\neq$ 'incr' or 'decr' then the variable value is set equal to the value during the previous time-step.

If the state variable is unordered linguistic the updating of value is much simpler. If the variable's T-In $_{t-1}$ $\geq$ T-Delta $_t$ then the variable has changed value and the new value is set equal to the value of D-Delta $_t$. If not the value remains the same (i.e. VarVal $_t$ is set equal to VarVal $_{t-1}$).

T-In Updating

T-In represents the time that a variable has spent at a particular value at the end of a time-step. The procedure for updating the T-In counter is simple. Three rules are used:

IF VarVal $_t$ $\neq$ VarVal $_{t-1}$ **THEN** T-In $_t$ = 1

IF D-Delta $_t$ $\neq$ D-Delta $_{t-1}$ **THEN** T-In$_t$ = 1

IF VarVal $_t$ = VarVal $_{t-1}$ **AND** D-Delta $_t$ = D-Delta $_{t-1}$ **THEN** T-In $_t$ = T-In $_{t-1}$ + 1

Basically, if a variable's value has changed its T-In is reset and if it hasn't changed value then its T-In value is incremented by 1. These rules ensure that, if a variable changes value, by the end of the time-step during which it has changed value, it will have a T-In value of 1. T-In is only updated after any changes in value have been determined.

6.3.4 Reasoning System: Convenience Predicates

6.3.4.1 Qualitative Operators

The Reasoning System provides a range of convenience predicates that can be used as conditions in rules for determining or updating variable values. One set of these predicates are concerned with providing functionality similar to algebraic order of magnitude operators like 'greater than' and 'less than or equal to' but for qualitative or ordered linguistic variables. These 'qualitative operators' have already been briefly described in section 6.2.3.1. Two of these operators are described in more detail below to further illustrate. It should be noted that these operators should only be applied to ordered support sets.

All the qualitative operators work on a similar basis. This involves determining the numbered position of specified variable values within ordered support sets with position number 1 taken as being the left-most element and *n* as the right-most in a support set of *n* elements. As the elements increase in magnitude from left to right along an ordered support set, the question as to whether a particular value is greater than another value from the same support set can be answered by comparing their numerical position values.

In some models it may prove useful to be able to compare non-quantitative values from different variables. For example, has a species biomass growth rate exceeded its senescence rate? To compare values between variables or support sets a set of value correspondences must be declared in the Rule-Base component using the `eqt2` predicate (see Figure 74). These correspondences should fully map the values from one support set (let's call it A) onto those from the other one concerned (let's call it B). Once mapped, each value from A can easily be given a value position with respect to support set B. Based upon value correspondences and the use of value positions variable values can therefore be compared with respect to order of magnitude across different support sets. The mapping of support sets onto each other is dependent on the nature of the variables and the granularity of the support sets concerned. There are no general guidelines.

eqt1 is a qualitative version of the 'equal to' operator. It is used to determine whether a qualitative variable is equal to a particular value from its support set. Figure 86 details the form of the operator and gives an example of its use where it is used in a calc_scalar rule.

```
eqt1(VarName,VarVal1,VarVal2).
```

Example:

```
calc_scalar(community,CommNo,smoi_use,InfValList1,zero):-
      get_infdetails(biomasses,community,CommNo,InfValList1,InfV
      alList2),
      extract_arrayval((grass,CommNo),InfValList2,value_tuple(GB
      io,_,_,_)),
      extract_arrayval((shrub,CommNo),InfValList2,value_tuple(SB
      io,_,_,_)),
      extract_arrayval((tree,CommNo,),InfValList2,value_tuple(TB
      io,_,_,_)),
      eqt1(biomass,GBio,zero),
      eqt1(biomass,SBio,zero),
      eqt1(biomass,TBio,zero).
```

Example:
IF the biomass of grass (GBio) = zero **AND** the biomass of shrub (SBio) = zero **AND** the biomass of tree (TBio) = zero **THEN** the value of soil moisture use (smoi_use) will equal zero.

Figure 86 RBCLM3 qualitative 'equal to' operator eqt1

gt1 is a qualitative version of the greater than operator and is used to determine whether a qualitative variable value is greater than a particular element from its own support set. The example in Figure 87 shows how the operator can be used as a condition to determine the value of fire damage for a generic shrub species.

gt2 is a qualitative 'greater than' operator used to determine whether a qualitative variable value is greater than another qualitative variable value based on correspondences between variable values represented using eqt2 (Figure 74). Figure 88 shows how the operator can be used as a condition to determine whether one of two flows of water (into and out of a community object water storage) is greater than the other based on a set of specified eqt2 value correspondences.

```
gt1(VarName,VarVal1,VarVal2).
```

Example:

```
calc_scalar(species,(shrub,CommNo),fire_dmg,InfValList1,high):-
      get_infdetails(fire_sev,community,CommNo,InfValList1,FSVa
      l),
      get_infdetails(biomass,species,(shrub,CommNo),InfValList1
      ,
      value_tuple(Bval,_,_,_)),
      eqtl(fire_sev,FSVal,severe),
      gt1(biomass,BVal,low).
```

which can be read as,
IF there a severe fire (`fire_sev` = `severe`) **AND** shrub biomass (`BVal`) > low **THEN** the shrub species will sustain high fire damage (`fire_dmg`).

Figure 87 RBCLM3 qualitative 'greater than' operator gt1

```
gt2(VarName1,VarVal1,VarName2,VarVal2).
```

Example:

```
d_delta_rule(community,CommNo,smoisture,FlowList,decr):-
      get_infdetails(precip,community,CommNo,FlowList,PVal),
      get_infdetails(smoi_use,community,CommNo,FlowList,SMUVal)
      ,
      gt2(smoi_use,SMUVal,precip,Pval).
```

which can be read as,
IF the value of the flow soil moisture use (`smoi_use`) > the value the of the flow precipitation **THEN** the state variable soil moisture will have a D-Delta value of decr.

Figure 88 RBCLM3 qualitative 'greater than' operator gt2

6.3.4.2 Variable Value Extraction Predicates

When the Reasoning System queries a rule in the Rule-Base to determine or update a variable value a list of the variable's influencing variable names and current values is passed in as a list of some sort – the structure varies depending on the type of variable. Not every rule will necessarily require the use of all influencing variable values, so some means of extracting the appropriate information to query conditions inside rule bodies is necessary. To do this the RBCLM3 provides two value extraction convenience predicates:

Figures 89 and 90 detail these predicates. They can be used as conditions in any rule – `calc_scalar`, `d_delta_rule`, `t_delta_rule` etc. Examples have been given in previous figures, to which the reader is referred.

```
get_infdetails(VarName,ObjectName,ObjectInst,InfValList,VarVa
l).

where,

ObjectInst = (SpeciesName,CommNo) OR CommNo
InfValList = [(VarName,ObjectName,ObjectInst,VarVal)|Rest]
```

Figure 89 RBCLM3 `get_infdetails` value extraction predicate

```
extract_arrayval(ObjectInst,InfValList,VarVal).

where,

ObjectInst = (SpeciesName1,CommNo1)
InfValList = [((SpeciesName2,CommNo2),VarVal))|Rest]
```

Figure 90 RBCLM3 `extract_arrayval` value extraction predicate

`get_infdetails` is used to obtain a variable (`VarName`) value (`VarVal`) belonging to a specified object (`ObjectName`) and instance (`ObjectInst`) from a list of influencing variable values (`InfValList`). The value, `VarVal` can then be used within the body of a rule in further conditions. `VarVal` may in fact be a list of values for an array variable. To obtain a particular value of the array variable (i.e. for a particular instance) `extract_arrayval` must then be used on the array `VarVal` result.

`extract_arrayval` is used to obtain a value from a list representing an array variable's value. As all the values within the list represent the same variable, there is no need to specify the variable name. Instead, only the species object instance need be specified (see Figure 90).

6.3.4.3 Miscellaneous Convenience Predicates

`fmod` is a convenience predicate that can be used to generate repeating numerical patterns. Section 6.2.4.4 details how `fmod` can be used to specify the time-steps over which particular exogenous variable values apply.. This section shall not repeat the detail there – it will suffice to add a specification of the predicate here (see Figure 91).

```
fmod(X,Y,Z):-
      Z is X mod Y.
```

Example of a query:

```
-? fmod(10,6,Z).
-? Z = 1.
```

Explanation:
10 divided by 6 is equal to 1 (integer) with a remainder of 4.

Figure 91 Specification and example of the RBCLM3 `fmod` predicate

6.4 Example Application – RBCLM3 'Proof of Concept' Model

To demonstrate that it is possible to simulate species-level dynamics using the rule-based temporal reasoning formalism employed by the RBCLM3 a 'proof of concept' model (PoCM) was developed.

6.4.1 Structure and Value Knowledge

The structure of the RBCLM3 PoCM is detailed in Figure 71. Each species object instance has 1 state variable, absolute biomass, with 1 inflow (growth) and 3 outflows (mortality, fire damage and grazing damage). The biomass attainable by each species' is limited by a maximum absolute biomass value represented by the functional attribute maximum biomass. The community object also has 1 state variable, soil moisture, which is influenced by 1 inflow (precipitation) and 1 outflow (soil moisture use by the plant species). Each species is influenced by each other species in terms of competition for the limited resources of soil water and light radiation. The community receives water through an exogenous variable called precipitation that feeds directly into soil moisture. The community also receives light radiation from an exogenous variable, radiation. Each species only receives a certain amount of each resource, given by the species property intermediate variables available radiation and available soil moisture, based upon the amount available (given by soil moisture and radiation values) and the biomass of all the species (given by the array variable biomasses). Each species has a different ability to capture resources. For example, as tree biomass increases the tree species will capture more radiation than the grass species due to its stature and growth form. So, the resources that are available to each species depend not only on the actual amount available but also on each species' own resource capture ability and the biomass and resource capture abilities of the other species. This allows the situation whereby a large biomass of grass may out-compete a small biomass of tree species for access to water and/or light resource to be handled.

As well as resource competition the RBCLM3 PoCM incorporates disturbance through the exogenous variables fire and grazing. Along with variables representing the flammability and penetrability of the community, these exogenous variables influence disturbance severity variables, which in turn exert the effects of disturbance on the species through effects on the biomass outflows fire damage and grazing damage. How disturbance affects each species is different and determined by species-specific rules. Note here that the RBCLM3 does not employ a special disturbance system. Variables representing disturbance effects are ordinary intermediate and exogenous variables.

Table 42 details the variables used in the model. To explore the utility of the RBCLM3 method for reasoning about change in non-quantitatively valued variables it was decided to use qualitative support sets for most model variables. The basic qualitative support set [zero,low,medium,high] was used in accordance with the principle of minimum variation (Salles and Bredeweg 1997). Zero was included as many model variables can, in reality, go to zero. However biomass was modelled with a minimum value of v_low (very low) to reflect that there is almost always greater than zero biomass present, from seed rain, roots, isolated individual plants etc. The value intervals low, medium and high were felt to be the minimum required to capture interesting quantity dynamics. Fire was modelled as a binary trigger that may result in a fire event of [none, low, high] severity. Grazing animals, valued qualitatively, can also result in a grazing disturbance of [none, low, high] severity. Vegetation type is valued using an unordered linguistic support set, [bare,grassland,shrubland,forest].

The time-unit (and time-step) of the RBCLM3 PoCM is 1 month.

Type	Name	Description	Support Set Type	Support Set
SV_c	smoisture	Soil moisture volume	Qualitative value	[zero,low,medium,high]
F_c	precip	Volume of soil water inflow from precipitation per time-step	Qualitative	[zero,low,medium,high]
F_c	smoi_use	Volume of soil water outflow by vegetation uptake per time-step	Qualitative	[zero,low,medium,high]
IV_c	biomasses	Array variable containing all the species biomass values	Array of qualitative valued variables	[v_low,low,medium,hi gh]
IV_c	fire_sev	Fire severity	Ordered linguistic	[none,mild,severe]
IV_c	flamma	Community flammability rating	Qualitative	[zero,low,medium,high]
IV_c	grazing_sev	Grazing severity	Ordered linguistic	[none,mild,severe]
IV_c	total_penet	Community penetrability based on all the species biomasses	Qualitative	[zero,low,medium,high]
IV_c	veg_state	Community vegetation type	Unordered linguistic	[bare,grassland,shrubla nd, forest]
EV	fire	Fire disturbance event trigger	Binary	[no,yes]
EV	grazing	Grazing animal numbers	Qualitative	[zero,low,medium,high]
EV	precipitation	Volume of precipitation per time-step	Qualitative	[zero,low,medium,high]
EV	radiation	Amount of PAR per time-step	Qualitative	[zero,low,medium,high]
SV_{sp}	biomass	Biomass of each species across whole community	Qualitative	[v_low,low,medium,hi gh]
F_{sp}	fire_dmg	Biomass outflow per time-step caused by fire damage	Qualitative	[zero,low,high]
F_{sp}	grazing_dm g	Biomass outflow per time-step caused by grazing damage	Qualitative	[zero,low,medium,high]
F_{sp}	growth	Biomass inflow per time-step due to natural growth	Qualitative	[zero,low,medium,high]
F_{sp}	mortality	Biomass outflow per time-step due to natural senescence	Qualitative	[zero,low,medium,high]
IV_{sp}	av_rad	Amount of PAR resource available to a species for a time-step	Qualitative	[zero,low,medium,high]
IV_{sp}	av_smoi	Amount of soil moisture resource available to a species for a time-step	Qualitative	[zero,low,medium,high]
FA	max_biomas s	Maximum absolute biomass for a species	Qualitative	[zero,low,medium,high]

Table 42 RBCLM3 Proof of Concept Model variable details (SVn = state variable, IV n = intermediate variable, F n = flow, EV = exogenous variable, P = parameter and FA = functional attribute, where n denotes the object to which the variable belongs; c = community object, $_{sp}$ = species object)

6.4.2 Relationship Knowledge

6.4.2.1 State Variables

Tables 43 and 44 detail the rules used to determine D-Delta and T-Delta for the community state variable, smoisture. As T-Delta values (in monthly time-units) are specific to particular directions of change as well as particular combinations of influencing variable values, Table 45 presents the T-Delta rules grouped by D-Delta value. In both tables, qualitative operators are represented by standard arithmetic symbols (=, > etc.) with a subscript 'q' to denote qualitative e.g. $=_q$ is a qualitative equal to operator, either `eqt1` or `eqt2` depending on whether the values being compared belong to the same variable and support set (`eqt1`) or different variables (`eqt2`). Each combination of influencing variable values that leads to a particular value of D-Delta or T-Delta are shown enclosed in standard round brackets. Within combinations, different variable values required are linked by a logical **AND**. If there are many combinations that can lead to the same value of D-Delta or T-Delta, each bracketed condition that applies is separated by a logical **OR** operator and placed on a separate line for clarity.

D-Delta Value	**Conditions**
decr	(smoi_use $>_q$ precip)
std	(smoi_use $=_q$ precip)
incr	(smoi_use $\leq_q$ precip)

Table 43 smoisture D-Delta rules

D-Delta : T-Delta	**Conditions**
decr : 0	(smoi_use $=_q$ high AND precip $=_q$ low)
decr : 1	(precip $=_q$ low) OR (smoi_use $=_q$ medium AND precip $=_q$ medium)
decr : 2	((precip $=_q$ low OR precip $=_q$ medium) AND smoi_use $=_q$ high) OR (precip $=_q$ low AND smoi_use $=_q$ medium) OR (precip $=_q$ zero AND smoi_use $=_q$ low)
incr : 0	(precip $=_q$ high AND smoi_use $=_q$ zero)
incr: 1	(precip $\geq_q$ medium AND smoi_use $=_q$ low)
incr: 2	Any set of conditions for D-Delta = incr not covered by incr: 0 and incr: 2

Table 44 smoisture T-Delta rules

Each species biomass variable is influenced by 4 flows – 1 inflow and 3 outflows. The same biomass D-Delta rules are used for all species. The PoCM assumes that each species responds identically to the net difference between biomass inflow and biomass outflow. For example, if there is no growth inflow but at least one non-zero outflow (fire_dmg, grazing_dmg or mortality) then, irrespective of species, there will be a net outflow of biomass, setting D-Delta equal 'decr'.

Within each species, the qualitative support set for each biomass flow (inflow or outflow) is assumed to map identically onto a numerical scale i.e. low growth represents the same range of amount of biomass as low mortality for a given species. Between species however the mapping will be different e.g. high growth for the tree species will be more than high growth for the grass species due to the density and sheer size of tree species compared to grass species.

T-Delta values are handled in a more species specific manner. For example, grass species can grow more rapidly than tree species so T-Delta values under the same conditions must be different. Each species is assumed to have a fast and a slow rate of change for both D-Delta = incr and D-Delta = decr. Consequently each species has two possible T-Delta values for each of these D-Delta values. The rules to determine T-Delta take the values of the biomass inflows and outflows as conditions. Although these biomass flows vary between species in the absolute biomass that they represent, they are assumed to represent the same relative values i.e. low grass growth represents the same proportional range of possible growth per time-step as for low shrub and tree species growth. It is therefore assumed that the same combination of flow values will produce the same rate of change for a given D-Delta value across all species i.e. mortality being greater than growth and a low fire damage flow will cause a rapid decline in biomass for all species. The actual T-Delta value inferred will however, be different to represent species-specific growth rates etc. Each species is also assumed to react to high fire damage in the same way – complete destruction of non-seed biomass represented by D-Delta = zero and T-Delta = 0.

It should be noted that, although the same rule sets are used to determine how biomass changes value for all 3 species in the PoCM, these rules refer only to the values of each species' biomass flows. Species-specific rules are used to determine the values of those flows. This ensures that their values reflect species-specific responses to environmental conditions.

Table 45 details the general rules for determining biomass D-Delta, irrespective of species. Table 45 details the T-Delta rules for all 3 species. The same notation is used as in Tables 43 and 44.

D-Delta Value	Conditions
zero	(fire_dmg $=_q$ high)
decr	(mortality $>_q$ growth AND fire_dmg $=_q$ low) OR (grazing_dmg $>_q$ growth AND fire_dmg $=_q$ low) OR (growth $=_q$ high AND mortality $>_q$ medium AND grazing_dmg $>_q$ low) OR (growth $=_q$ high AND mortality $>_q$ low AND grazing_dmg $>_q$ medium) OR (growth $=_q$ medium AND mortality $\geq_q$ medium AND grazing_dmg $\geq_q$ low) OR (growth $=_q$ medium AND mortality $\geq_q$ low AND grazing_dmg $\geq_q$ medium) OR (growth $=_q$ low AND mortality $\geq_q$ medium AND grazing_dmg $\geq_q$ low) OR (growth $=_q$ low AND mortality $\geq_q$ low AND grazing_dmg $\geq_q$ medium)
incr	(growth $=_q$ high AND mortality $\leq_q$ medium AND grazing_dmg $\leq_q$ medium) OR (growth $=_q$ medium AND mortality $\leq_q$ medium AND grazing_dmg $=_q$ zero) OR (growth $=_q$ medium AND mortality $=_q$ zero AND grazing_dmg $\leq_q$ medium) OR (growth $=_q$ low AND mortality $=_q$ zero AND grazing_dmg $\leq_q$ low) OR (growth $=_q$ low AND mortality $\leq_q$ low AND grazing_dmg $=_q$ zero)
std	Any combination of flow values not covered by the conditions for D-Delta = zero, incr or decr.

Table 45 General (all spp.) biomass D-Delta rules

D-Delta : T-Delta (G : S : T)	Conditions
zero : (0 : 0 : 0)	(fire_dmg $=_q$ high)
decr : (1 : 6 : 18)	(mortality $\geq_q$ growth AND fire_dmg $=_q$ low) OR (grazing_dmg $\geq_q$ growth AND fire_dmg $=_q$ low) OR (growth $=_q$ medium AND mortality $\geq_q$ medium AND grazing_dmg $\geq_q$ low) OR (growth $=_q$ medium AND mortality $\geq_q$ low AND grazing_dmg $\geq_q$ medium) OR (growth $=_q$ low AND mortality $\geq_q$ medium AND grazing_dmg $\geq_q$ low) OR (growth $=_q$ low AND mortality $\geq_q$ low AND grazing_dmg $\geq_q$ medium)
decr : (2 : 8 : 24)	(growth $=_q$ high AND mortality $>_q$ medium AND grazing_dmg $>_q$ low) OR (growth $=_q$ high AND mortality $>_q$ low AND grazing_dmg $>_q$ medium)
incr : (1 : 6 : 18)	(growth $=_q$ high AND mortality $\leq_q$ medium AND grazing_dmg $\leq_q$ medium) OR (growth $=_q$ low AND mortality $=_q$ zero AND grazing_dmg $\leq_q$ low) OR (growth $=_q$ low AND mortality $\leq_q$ low AND grazing_dmg $=_q$ zero)
incr : (2 : 8 : 24)	(growth $=_q$ medium AND mortality $\leq_q$ medium AND grazing_dmg $=_q$ zero) OR (growth $=_q$ medium AND mortality $=_q$ zero AND grazing_dmg $\leq_q$ medium)

Table 46 Species object biomass T-Delta rules (G = grass species, S = shrub species, T = tree species)

It should be noted that the rules for each T-Delta value conform to the conditions required to produce the D-Delta value to which they apply. This is essential to maintain consistency between D-Delta and T-Delta rules.

6.4.2.2 Flows

The community object state variable, smoisture, is affected by two flows – an inflow, precip, and an outflow, smoi_use. Tables 47 and 48 details the rules used to determine their values.

The inflow, precip is directly proportional to the value of the exogenous variable, precipitation. The outflow, smoi_use, varies in relation to the amount of biomass present in the community in such a way that the relative usage of different species is represented. If there is essentially no vegetation then smoi_use is held to be zero. For the flow to be low, the grass biomass can be medium or less whilst the shrub and tree biomasses must both be low or v_low. This reflects a greater water usage rate by the shrub and tree species compared to the grass species. A high flow rate occurs if all of the species have a medium or high biomass or if any one of the species has high biomass whilst both the other two have a biomass greater than or equal to low.

Flow Name	**Value**	**Conditions**
precip	zero	(precipitation $=_q$ zero)
	low	(precipitation $=_q$ low)
	medium	(precipitation $=_q$ medium)
	high	(precipitation $=_q$ high)
smoi_use	zero	(biomass$_g$ $=_q$ zero AND biomass$_s$ $=_q$ zero AND biomass$_t$ $=_q$ zero)
	low	(biomass$_g$ $\leq_q$ medium AND biomass$_s$ $\leq_q$ low AND biomass$_t$ $\leq_q$ low)
	medium	Any combination of biomass values not covered by smoi_use = zero, low or high
	high	(biomass$_g$ $\geq_q$ medium AND biomass$_s$ $\geq_q$ medium AND biomass$_t$ $\geq_q$ medium) OR (biomass$_g$ $=_q$ high AND biomass$_s$ $\geq_q$ low AND biomass$_t$ $\geq_q$ low) OR (biomass$_g$ $\geq_q$ low AND biomass$_s$ $=_q$ high AND biomass$_t$ $\geq_q$ low) OR (biomass$_g$ $\geq_q$ low AND biomass$_s$ $\geq_q$ low AND biomass$_t$ $=_q$ high)

Table 47 smoisture flow rules

The rules used to determine the values of each species biomass flows are detailed in tables 48 – 50. The subscripts g, s and t are used to denote that variables belong to the grass, shrub or tree species instance as appropriate.

For all species, growth is set equal to zero under three conditions: if biomass is equal to the absolute maximum biomass achievable for a species (this is medium for grass and high for both shrub and tree, valued using the same support set to reflect the fact that shrub and tree biomass, at carrying capacity, will be greater than grass biomass) or if either the soil moisture or radiation available to a species is zero (note that this does not necessarily mean that the soil moisture or radiation across the whole community object is zero, simply that the resources available to a particular species are zero). The rules for non-zero growth values vary across species to reflect their functional characteristics e.g. the grass species is modelled as being susceptible to low levels of available soil moisture or radiation whereas the shrub and tree species are modelled as being hardier and less likely to quickly respond to low resource levels. For all three species, medium growth is modelled as resulting from influencing variable values that do not result in zero, low or high growth. This approach to rule construction shall be discussed in section 7.5.

A similar approach was taken to modelling the other flows. The functional characteristics and behaviour of each species is captured in the differences between the rules leading to flow values. There is no explicit use of functional attributes variables in the rules. This was a conscious design decision. Representing functional characteristics within species-specific rules can be regarded as a type of 'implicit' functional attributes (IFA) modelling. An alternative design that could have been used would have been to design a set of general rules, one for each value of flow, and have those rules use species-specific functional attribute values (e.g. palatability of grass = high, palatability of tree = low) as conditions. This could be regarded as a type of 'explicit' functional attributes (EFA) modelling of species. Both types of modelling can be entered into with the RBCLM3. The PoCM was designed around the IFA approach as the knowledge available from the ecological literature was more immediately suited to this representation. As discussed at the start of this chapter, there is currently no agreement on how best to functionally characterise species or if a generally useful and meaningful functional characterisation or classification of species exists at all. If the EFA approach is to be taken, a range of important attributes must be identified, each species given a value for each attribute and rules developed to reason with the values of the attributes in order to determine and update other variable values.

Flow Name	Value	Conditions
growth	zero	(biomass$_g$ $=_q$ max_biomass$_g$) OR (av_rad$_g$ $=_q$ zero) OR (av_smoi$_g$ $=_q$ zero)
	low	(av_rad$_g$ $=_q$ low) OR (av_smoi$_g$ $=_q$ low)
	medium	Any combination of influencing variable values not covered by growth = zero, low or high
	high	(av_rad$_g$ $\geq_q$ medium AND av_smoi$_g$ $\geq_q$ low) OR (av_rad$_g$ $\geq_q$ low AND av_smoi$_g$ $\geq_q$ medium)
fire_dmg	zero	(fire_sev $=_q$ none) OR (biomass$_g$ $=_q$ v_low)
	low	Any combination of influencing variable values not covered by fire_dmg = zero, low or high
	high	(fire_sev $=_q$ mild AND biomass$_g$ $\geq_q$ medium) OR (fire_sev $=_q$ severe)
grazing_dmg	zero	(grazing_sev $=_q$ none) OR (biomass$_g$ $=_q$ v_low)
	low	(grazing_sev $=_q$ mild AND biomass$_g$ $\leq_q$ medium)
	medium	Any combination of influencing variable values not covered by grazing_dmg = zero, low or high
	high	(grazing_sev $=_q$ severe AND biomass$_g$ $\geq_q$ medium)
mortality	zero	(biomass$_g$ $=_q$ v_low)
	low	(av_rad$_g$ $=_q$ high AND av_smoi$_g$ $=_q$ high)
	medium	Any combination of influencing variable values not covered by mortality = zero, low or high
	high	(av_rad$_g$ $\leq_q$ low AND av_smoi$_g$ $\leq_q$ low)

Table 48 Grass species biomass flow rules

Flow Name	Value	Conditions
growth	zero	(biomass$_s$ $=_q$ max_biomass$_s$) OR (av_rad$_s$ $=_q$ zero) OR (av_smoi$_s$ $=_q$ zero)
	low	(av_rad$_s$ $=_q$ low AND av_smoi$_s$ $\leq_q$ medium) OR (av_rad$_s$ $\leq_q$ medium AND av_smoi$_s$ $=_q$ low)
	medium	Any combination of influencing variable values not covered by growth = zero, low or high
	high	(av_rad$_s$ $\geq_q$ medium AND av_smoi$_s$ $\geq_q$ medium)
fire_dmg	zero	(fire_sev $=_q$ none) OR (biomass$_s$ $=_q$ v_low)
	low	Any combination of influencing variable values not covered by fire_dmg = zero, low or high
	high	(fire_sev $=_q$ severe AND biomass$_s$ $>_q$ low)
grazing_dmg	zero	(grazing_sev $=_q$ none) OR (biomass$_s$ $=_q$ v_low)
	low	(grazing_sev $=_q$ mild AND biomass$_s$ $\leq_q$ medium)
	medium	Any combination of influencing variable values not covered by grazing_dmg = zero, low or high
	high	(grazing_sev $=_q$ severe AND biomass$_s$ $=_q$ high)
mortality	zero	(biomass$_s$ $=_q$ v_low)
	low	(av_rad$_s$ $=_q$ high AND av_smoi$_s$ $\geq_q$ medium) OR (av_rad$_s$ $\geq_q$ medium AND av_smoi$_s$ $=_q$ high)
	medium	Any combination of influencing variable values not covered by mortality = zero, low or high
	high	(av_rad$_s$ $=_q$ low AND av_smoi$_s$ $\leq_q$ low) OR (av_rad$_s$ $\leq_q$ low AND av_smoi$_s$ $=_q$ low)

Table 49 Shrub species biomass flow rules

Flow Name	Value	Conditions
growth	zero	(biomass$_s$ $=_q$ max_biomass$_g$) OR (av_rad$_s$ $=_q$ zero) OR (av_smoi$_s$ $=_q$ zero)
	low	(av_rad$_t$ $=_q$ low AND av_smoi$_t$ $\leq_q$ medium) OR (av_rad$_t$ $\leq_q$ medium AND av_smoi$_t$ $=_q$ low)
	medium	Any combination of influencing variable values not covered by growth = zero, low or high
	high	(av_rad$_t$ $=_q$ high AND av_smoi$_t$ $=_q$ high)
fire_dmg	zero	(fire_sev $=_q$ none) OR (biomass$_t$ $=_q$ v_low)
	low	Any combination of influencing variable values not covered by fire_dmg = zero, low or high
	high	(fire_sev $=_q$ severe AND biomass $=_q$ high)
grazing_dmg	zero	(grazing_sev $=_q$ none) OR (biomass$_t$ $=_q$ v_low)
	low	(grazing_sev $=_q$ mild AND biomass$_t$ $\leq_q$ medium)
	medium	Any combination of influencing variable values not covered by grazing_dmg = zero, low or high
	high	(grazing_sev $=_q$ severe AND biomass$_t$ $=_q$ high)
mortality	zero	(biomass$_t$ $=_q$ v_low)
	low	(av_rad$_t$ $\geq_q$ medium AND av_smoi$_s$ $\geq_q$ medium)
	medium	Any combination of influencing variable values not covered by mortality = zero, low or high
	high	(av_rad$_t$ $\leq_q$ low AND av_smoi$_t$ $\leq_q$ low)

Table 50 Tree species biomass flow rules

6.4.2.3 Intermediate Variables

The community object contains five scalar intermediate variables, four related to disturbance severity determination and one to classifying the species composition into a vegetation type, and one array intermediate variable, to gather all the species biomass values into a single variable.

Fire disturbance is modelled as resulting from an exogenous trigger variable – ‘fire’. If this is ‘yes’ for a given time-step then a fire may occur, depending on the vegetation present. Each community has a ‘flammability’ rating that is determined by the amount of each species’ biomass. The rules for determining flammability are detailed in Table 51.

Value	Conditions
low	(biomass$_g$ $\leq_q$ low AND biomass$_s$ $\leq_q$ low AND biomass$_t$ $\leq_q$ low)
medium	Any combination of influencing variable values not covered by flamma = low or high
high	(biomass$_g$ $=_q$ high) OR (biomass$_s$ $=_q$ high)

Table 51 Community flammability (flamma variable) rules

If there is a fire trigger event then community flammability decides how severe the fire will be. Fire severity is modelled using a three value support set – {none, mild, severe}. It should be noted that severity of impact on vegetation is modelled, not intensity of fire or disturbance. The rules used are detailed in Table 52.

Value	**Conditions**
none	(fire $=_q$ no)
mild	(fire $=_q$ yes AND flamma $=_q$ low) OR (fire $=_q$ yes AND flamma $=_q$ medium)
severe	(fire $=_q$ yes AND flamma $=_q$ high)

Table 52 Fire severity (fire_sev) rules

Grazing disturbance is handled in a similar way. An exogenous variable, grazing, representing the intensity of grazing pressure on a community object during a given time-step, is used to determine whether there is any grazing occurring in a given community. If there is non-zero grazing then the severity of effect is determined based upon the total penetrability of the community. Total penetrability represents the ease of access for animal foraging and browsing. It is directly dependent upon the biomass of all species present. The rules for total penetrability are detailed in Table 53 whilst the rules for grazing severity are detailed in Table 54. The rules incorporate the implicit effects of each species' growth form on total penetrability e.g. grass, even at a relatively high biomass level is easy to forage and browse, whereas shrubs, at high biomass are often dense, forming impenetrable thickets. Note that with respect to total penetrability, a value of 'zero' is the highest, the most impenetrable.

Value	**Conditions**
zero	($\text{biomass}_s =_q$ high AND $\text{biomass}_t =_q$ high)
low	($\text{biomass}_s =_q$ high AND $\text{biomass}_t \geq_q$ medium)
medium	Any combination of influencing variable values not covered by total_penet = zero, low or high
high	($\text{biomass}_s \leq_q$ low AND $\text{biomass}_t \leq_q$ low)

Table 53 Total penetrability (total_penet) rules

Value	Conditions
none	(grazing $=_q$ zero) OR (total_penet $=_q$ zero)
mild	Any combination of influencing variable values not covered by grazing_sev = none or severe
severe	(grazing $=_q$ high AND total_penet $\geq_q$ medium)

Table 54 Grazing severity (grazing_sev) rules

Vegetation state is an intermediate variable used to classify the vegetation (i.e. all the species together) of a site into a discrete type. There are three vegetation types distinguished – grassland, shrubland and forest. Table 55 details the rules used to determine (classify) the values.

Value	Conditions
grassland	($\text{biomass}_s \geq_q$ medium AND $\text{biomass}_t \leq_q$ low)
shrubland	Any combination of influencing variable values not covered by veg_type = grassland or forest
forest	($\text{biomass}_t \geq_q$ medium)

Table 55 Vegetation type (veg_type) rules

All the individual species biomass values are brought together into a single array variable, biomasses.

Each species object instance has two intermediate variables, available radiation (av_rad) and available soil moisture (av_smoi). Available radiation is influenced by radiation and the biomasses of all the species present. The rules for determining the value for each species are detailed in Tables 56 – 58 along with the rules for determining soil moisture use for each species.

Intermediate Variable	**Value**	**Conditions**
av_rad	zero	(radiation $=_q$ zero)
	low	($\text{biomass}_s \geq_q$ medium) OR ($\text{biomass}_t \geq_q$ medium) OR (radiation $=_q$ low)
	medium	Any combination of influencing variable values not covered by av_rad = zero, low or high
	high	($\text{biomass}_s \leq_q$ low AND $\text{biomass}_t \leq_q$ low AND radiation $=_q$ high)
av_smoi	zero	(smoisture $=_q$ zero)
	low	($\text{biomass}_s \geq_q$ medium) OR ($\text{biomass}_t \geq_q$ medium) OR (smoisture $=_q$ low)
	medium	Any combination of influencing variable values not covered by av_smoi = zero, low or high
	high	($\text{biomass}_s \leq_q$ low AND $\text{biomass}_t \leq_q$ low AND smoisture $=_q$ high)

Table 56 Grass species available radiation (av_rad) and available soil moisture (av_smoi) rules

Intermediate Variable	Value	Conditions
av_rad	zero	(radiation $=_q$ zero)
	low	(radiation $=_q$ low) **OR** (biomass$_t$ $\geq_q$ medium)
	medium	Any combination of influencing variable values not covered by av_rad = zero, low or high
	high	(radiation $=_q$ high **AND** biomass$_t$ $\leq_q$ low)
av_smoi	zero	(smoisture $=_q$ zero)
	low	(smoisture $=_q$ low) **OR** (biomass$_t$ $=_q$ high)
	medium	Any combination of influencing variable values not covered by av_smoi = zero, low or high
	high	(smoisture $=_q$ high **AND** biomass$_g$ $\leq_q$ low **AND** biomass$_t$ $\leq_q$ low)

Table 57 Shrub species available radiation (av_rad) and available soil moisture (av_smoi) rules

Intermediate Variable	Value	Conditions
av_rad	zero	(radiation $=_q$ zero)
	low	(radiation $=_q$ low) OR (biomass$_t$ $=_q$ high)
	medium	Any combination of influencing variable values not covered by av_rad = zero, low or high
	high	(radiation $=_q$ high AND biomass $\leq_q$ medium)
av_smoi	zero	(smoisture $=_q$ zero)
	low	(smoisture $=_q$ low) OR (biomass$_g$ $=_q$ high AND biomass$_s$ $=_q$ high)
	medium	Any combination of influencing variable values not covered by av_smoi = zero, low or high
	high	(smoisture $=_q$ high AND biomass$_g$ $\leq_q$ medium AND biomass$_s$ $\leq_q$ medium)

Table 58 Tree species available radiation (av_rad) and available soil moisture (av_smoi) rules

The rules to determine the value of available radiation for each species implicitly include the effects of various functional characteristics including growth form (if the species is tall of stature then it will have easier access to light) and leaf area and angle (how much shade does the species cast?). This is also true of the rules to determine available soil moisture, where functional characteristics including root structure and depth are used implicitly.

6.4.2.4 Between Variable Value Correspondences

The PoCM uses a set of `eqt2` value correspondences to facilitate the comparison of different flow values in rules for calculating variable values. Table 59 details the correspondences to be used. It should be noted that the correspondences between species object variables only apply within a single instance although this is not formally represented. For example, the assertion that a mortality flow of low is equal to a growth flow of low should not be interpreted as saying that a grass mortality flow of low represents the same amount of biomass as a tree growth flow of low. However it can be interpreted as meaning that a low grass mortality represents the same amount of biomass as a low grass growth flow. As discussed the flow values across different species object instances represent different absolute amounts of biomass.

Variable Name	Variable Value		Variable Value	Variable Name
smoi_use	zero	$=_q$	zero	precip
	low	$=_q$	low	
	medium	$=_q$	medium	
	high	$=_q$	high	
mortality	zero	$=_q$	zero	growth
	low	$=_q$	low	
	medium	$=_q$	medium	
	high	$=_q$	high	
grazing_dmg	zero	$=_q$	zero	growth
	low	$=_q$	low	
	medium	$=_q$	medium	
	high	$=_q$	high	
fire_dmg	zero	$=_q$	zero	growth
	low	$=_q$	low	
		$=_q$	medium	
	high	$=_q$	high	
biomass	v_low	$=_q$	v_low	max_biomass
	low	$=_q$	low	
	medium	$=_q$	medium	
	high	$=_q$	high	

Table 59 Qualitative (eqt2) value correspondences

6.4.3 Model Results

The aim of this section is to describe a set of simulation experiments carried out using the PoCM under different scenaria (see Table 60). The aim of the experiments, as with those performed with the RBCLM1 and RBCLM2, is to demonstrate the utility of the RBCLM3 System for modelling vegetation dynamics. The first experiment involved simulating the baseline or neutral conditions in a single run free from fire and grazing. From this a series of runs were undertaken to explore what the combined effects of fire and grazing would be in the PoCM. One run was performed under conditions of no fire and low grazing intensity. The next two runs were performed under conditions of low grazing and fire once every 3 years and low grazing and fire once every year respectively, to explore the interaction between grazing and fire frequency.

	Run Number			
	1	2	3	4
Initial State Variable Values				
smoisture	[medium,3,std,n/a]	[medium,3,std,n/a]	[medium,3,std,n/a]	[medium,3,std,n/a]
Grass biomass	[medium,2,incr,10]	[medium,2,incr,10]	[medium,2,incr,10]	[medium,2,incr,10]
Shrub biomass	[medium,1,incr,10]	[medium,1,incr,10]	[medium,1,incr,10]	[medium,1,incr,10]
Tree biomass	[low,1,incr,10]	[low,1,incr,10]	[low,1,incr,10]	[low,1,incr,10]
Exogenous Variable Values				
fire	no	no	yes - *once every 36 time-steps from t = 6 (once every 3 years)*	yes - *once every 12 time-steps from t = 6 (once every year)*
grazing	zero	low	low	low
precipitation	low: Months 5 & 6 of every year (1 year = 12 time-steps) medium: Months 3,4 & 7-9 high: *Months 1,2 & 10-12*	low: Months 5 & 6 of every year (1 year = 12 time-steps) medium: Months 3,4 & 7-9 high: *Months 1,2 & 10-12*	low: Months 5 & 6 of every year (1 year = 12 time-steps) medium: Months 3,4 & 7-9 high: *Months 1,2 & 10-12*	low: Months 5 & 6 of every year (1 year = 12 time-steps) medium: Months 3,4 & 7-9 high: *Months 1,2 & 10-12*
radiation	medium: Months 1-4 & 7-12 high: *Months 5 & 6*	medium: Months 1-4 & 7-12 high: *Months 5 & 6*	medium: Months 1-4 & 7-12 high: *Months 5 & 6*	medium: Months 1-4 & 7-12 high: *Months 5 & 6*
Run Length	1200 time-steps (100 years)	1200 time-steps (100 years)	1200 time-steps (100 years)	1200 time-steps (100 years)

Table 60 Run conditions for RBCLM3 PoCM simulation experiment

The run conditions were designed such that tree biomass was initialised lower than both grass and shrub species biomass. This option was chosen so that the tree species did not immediately out-compete the grass and shrub for light and/or water. It represents the situation where the vegetation in the community object is starting off as a mixed grass/shrub cover shrubland state.

6.4.3.1 Results

Figures 92-95 present the results of each run in the simulation experiment. The results shown are the values for the state variables (3 biomass variables and 1 soil moisture variable) at the end of every year over the course of each run (1200 time-steps). Figure 96 presents the vegetation type results of all the simulations in one diagram. Figures 97-100 present the detailed month by month results for the first 100 months of each run to further illustrate the RBCLM3 PoCM's behaviour.

The following coding scheme is used in the results Figures:

- Biomass:

  1 = v_low, 2 = low, 3 = medium, 4 = high.
- Soil moisture:

  1 = zero, 2 = low, 3 = medium, 4 = high.
- Vegetation type:

  1 = grassland, 2 = shrubland, 3 = forest.

Run 1 (Figure 92) simulates the increase in tree species biomass over time that is expected and true most undisturbed vegetation. Figure 95 details this change in species dominance as the change in vegetation state from initial mixed cover shrubland to predominately forest cover after an initial minor oscillation between shrubland and grassland. The biomass of the tree species reaches high after around 10 years before it settles into a cycle, varying between high and medium. This is most likely driven by the periodic cycles in both grass and shrub species biomass that result from the collective effect of all species biomasses on available soil moisture and radiation levels.

As the tree biomass increases the grass biomass immediately drops from medium to low within a year or two and, despite a brief recovery, drops even further to very low at around year 7. The grass species then begins to cycle between low and very low biomass levels for the remainder of the run.

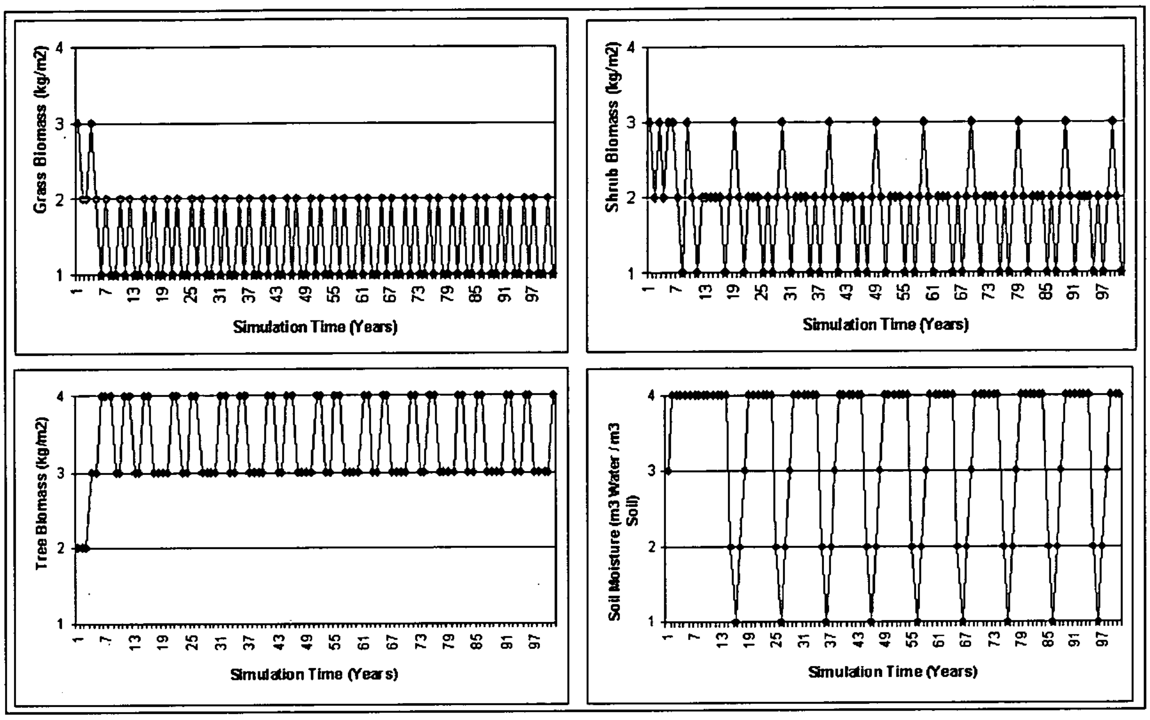


Figure 92 Annual results for RBCLM3 PoCM run 1 – no fire, no grazing

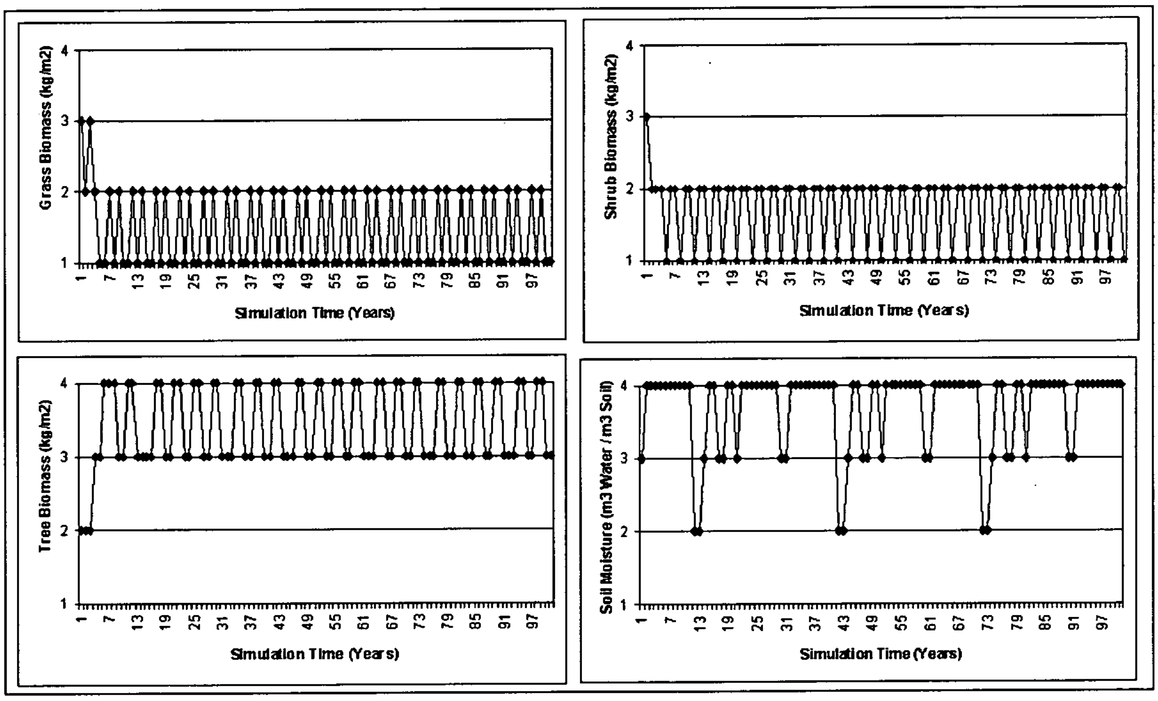


Figure 93 Annual results for RBCLM3 PoCM run 2 - no fire, low grazing

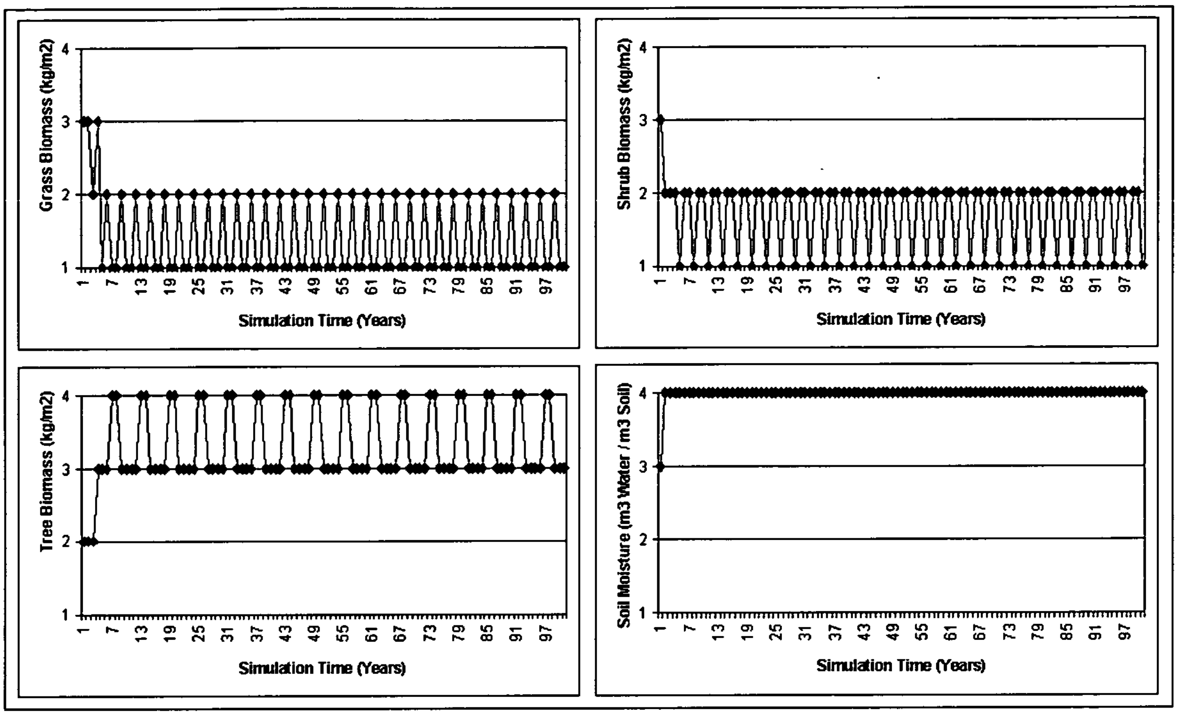


Figure 94 Annual results for RBCLM3 PoCM run 3 - fire once every 36 time-steps, low grazing

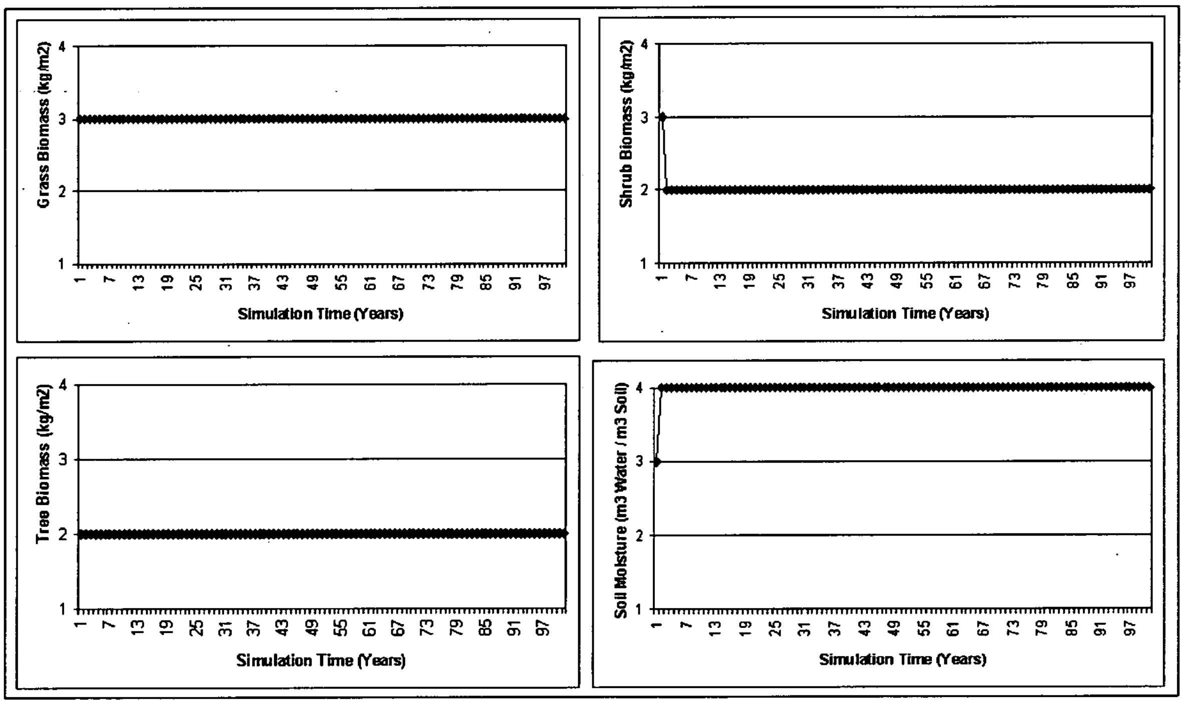


Figure 95 Annual results for RBCLM3 PoCM run 4 - fire once every 12 time-steps, low grazing

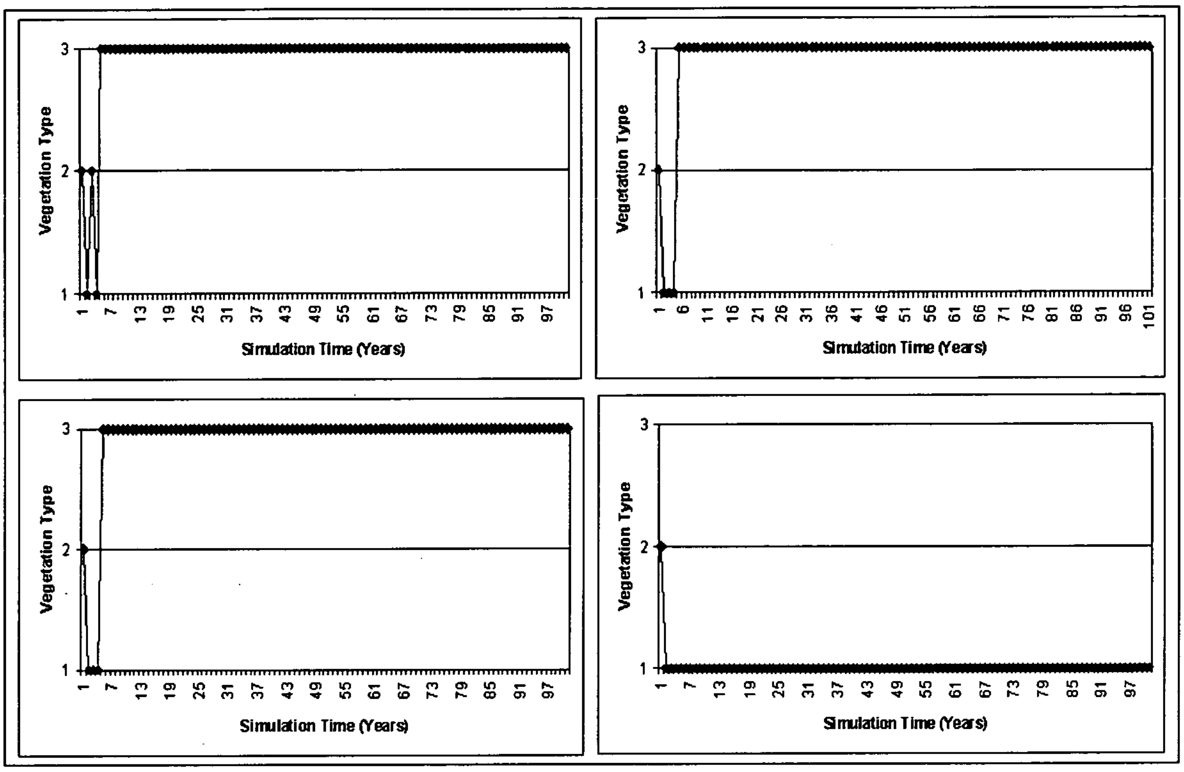


Figure 96 Annual vegetation type results for RBCLM3 PoCM (run 1 top left, run 2 top right, run 3 bottom left, run 4 bottom right)

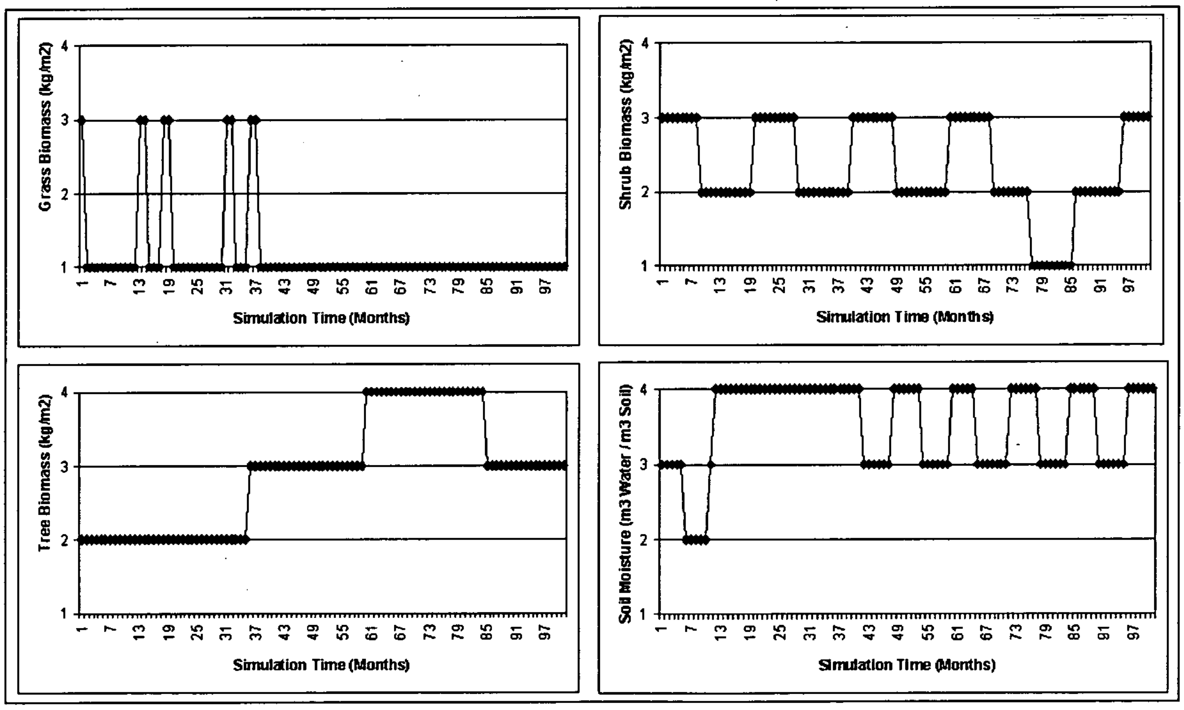


Figure 97 Monthly results for RBCLM3 PoCM run 1 – no fire, no grazing

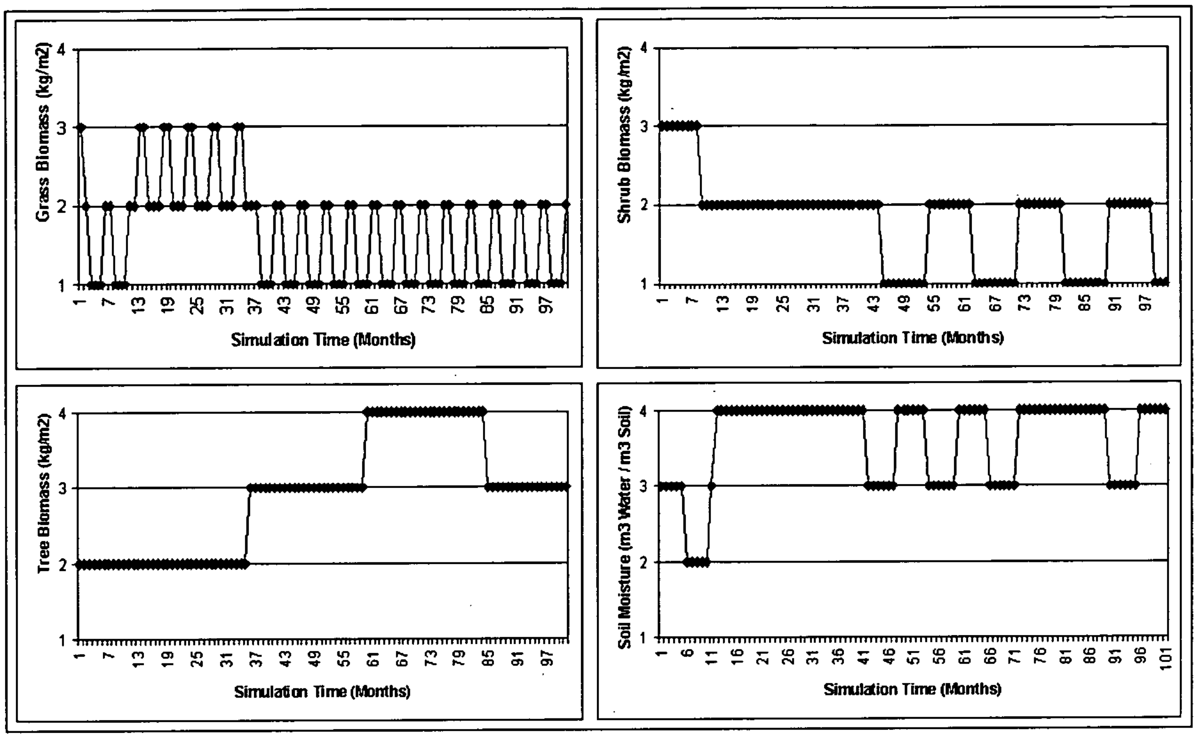


Figure 98 Monthly results for RBCLM3 PoCM run 2 – no fire, low grazing

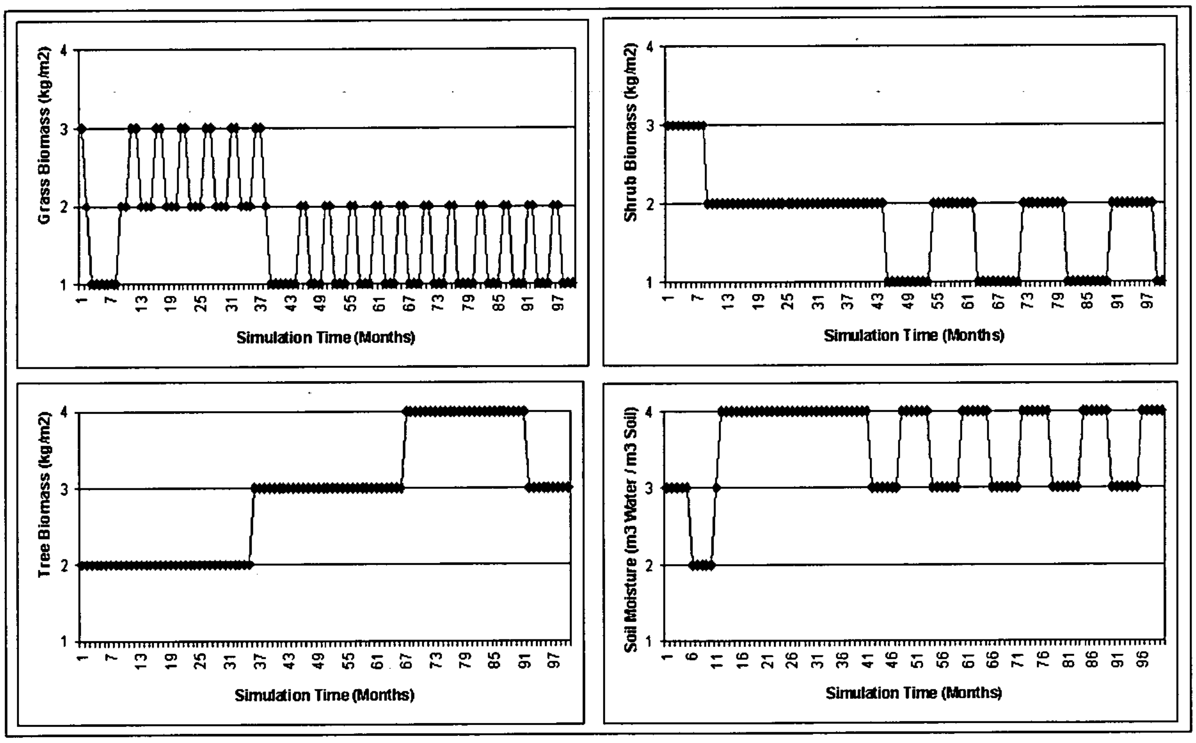


Figure 99 Monthly results for RBCLM3 PoCM run 3 – fire once every 36 time-steps, low grazing

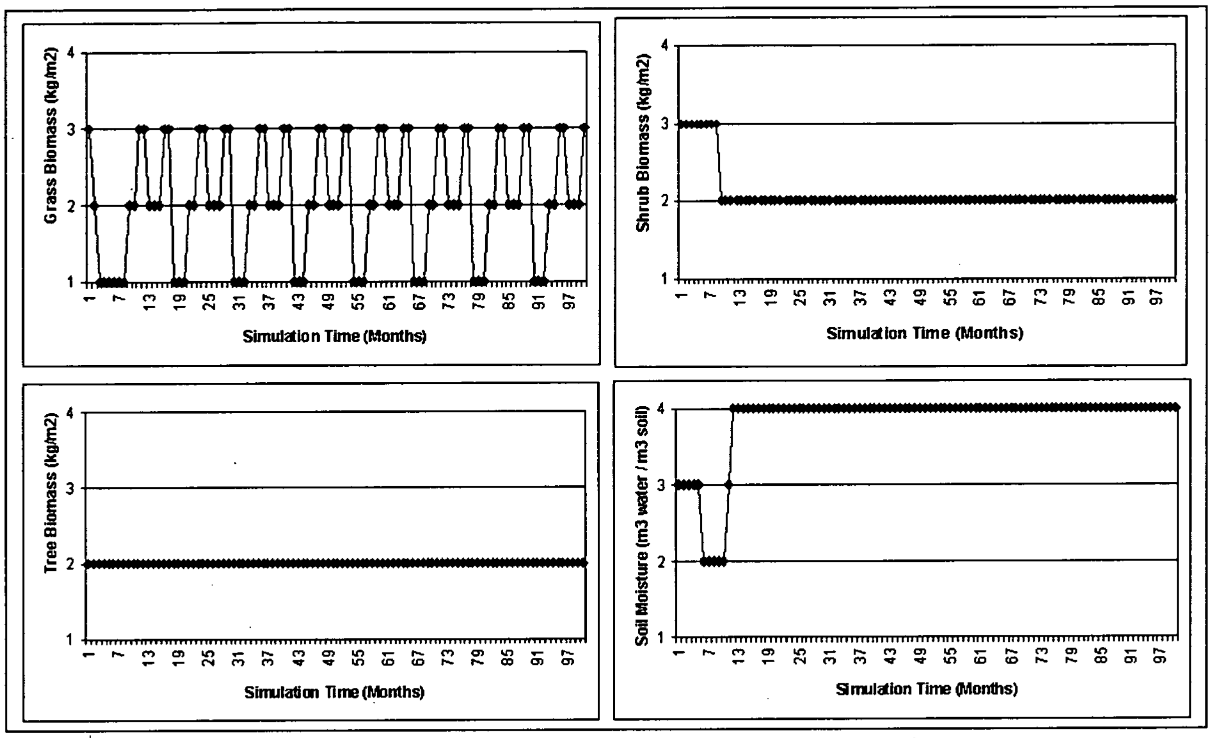


Figure 100 Monthly results for RBCLM3 PoCM run 4 - fire once every 12 time-steps, low grazing

The shrub biomass in run 1 cycles between medium and low levels for the first 7 or so years before dropping to very low, probably as a result of the increased resource competition resulting from high tree biomass. After this the shrub biomass increases to a medium level, the species possibly benefiting from the reduction in grass biomass levels. As the grass species recovers to a low biomass the shrub species suffers and it drops again to very low levels. It then recovers to low within a few years before settling into cyclical behaviour for the remained of the run.

Looking at the dynamics of the soil moisture state variable it can be seen that water levels initially rise to high, probably due to the fact that tree biomass had not risen and both shrub and grass biomass values had dropped. High water levels are maintained until around year 13 or 14 when they crash to zero over a few years. This can probably be attributed to the medium – high biomass levels of the tree species in combination with the other two species. After the crash the soil moisture then recovers to low, then medium and rises to high again over a period of around 5 years. High levels are maintained for a further 5 years or so before they crash and recover again in exactly the same manner. This cycle then continues for the remainder of the run.

In run 2 (Figure 93), under low grazing and no fire, similar behaviour occurs, with tree biomass gradually increasing over the first 7-10 years and the vegetation type turning to forest (Figure 96). Compared to run 1 the tree biomass shows similar cyclical behaviour, oscillating between high and medium levels, but is high for slightly longer over the whole run. This could be explained through grazing having a more detrimental effect on the grass and shrub species than the tree species. Indeed, although both the grass and shrub species demonstrate roughly the same behaviour as in run 1, the shrub species in particular has an overall lower biomass value over the whole run. This will most likely reduce the pressure on resources and hence the other species. Soil moisture levels behave similarly but never crash to zero, instead bottoming-out at low levels and cycling more frequently between high and medium levels.

Run 3 (Figure 94), with fire occurring at a return frequency of every 36 months from month 6 onwards and grazing occurring at low intensity, shows overall similar behaviour to both runs 1 and 2 in that tree biomass increases from low to high over the first 7 – 10 years of the run before settling into an cycle between medium and high levels. Both grass and shrub biomass levels drop from medium to low before settling into cyclical behaviour, oscillating between very low and low. Tree biomass is medium for a greater proportion of the run than in runs 1 and 2, continuing the trend observed between the first two runs' results whilst grass is very low for a greater proportion of the time than in runs 1 and 2. This shows some degree of combined fire and grazing effect although should be noted that the relatively high levels of tree biomass are unrealistic considering the frequency of fire disturbance. Shrub biomass is essentially the same as for run 2. Run 3's soil moisture levels rise almost immediately to high and remain at this level for the rest of the run. This is probably attributable to the reduction in both tree and grass biomass overall. Vegetation type (Figure 96) behaviour in run 3 is, unrealistically, the same as run 2.

In run 4 (Figure 95) the grass species remains constant at a medium biomass, dominating the vegetation, which immediately transitions into grassland type (Figure 96). The shrub species biomass drops immediately from medium to low and remains constant at this level for the remainder of the run. The tree biomass never rises above low. Soil moisture rises to high almost immediately, probably as a result of the overall low combined biomass of the community's vegetation and, in particular, the low shrub and tree biomasses.

It should be noted that the annual simulation run results only present some of the dynamics occurring in the PoCM. In a numerical simulation model running on a monthly time-step, results can be averaged for a year and, if presented annually, give a reasonably accurate picture of model behaviour. With non-quantitative variable values however it is not possible to average using general rules, if at all e.g. what is the average of low + low + medium biomass for grass? Therefore, to present results annually a single value for each variable for each year must be used. This choice may well distort some of the dynamics so, to further illustrate the capabilities of the RBCLM3 and provide a closer look at the dynamics of the PoCM, Figures 97-100 present the monthly results for biomass and soil moisture.

Looking at the monthly dynamics it can be seen that more is happening within the model than may be appreciated if only the annual results are examined. For example, in run 1 (Figure 100), the grass biomass can be seen to mostly be very low with short periodic increases to medium levels up until around month 40 (years 3-4). After that it remains at very low levels. The shrub biomass can be seen to regularly cycle between low and medium biomass levels until around month 75 (years 6-7) when it drops to very low. Soil moisture can be seen to not simply rise quickly to a high level but instead to drop to low before rising through medium again to high. From then soil moisture fluctuates between high and medium. Looking at the run 1 annual results soil moisture appears to be at a constant high level.

6.5 Discussion

6.5.1 Modelling Vegetation Dynamics - RBCLM3 System Utility

The PoCM results demonstrate that it possible to model vegetation dynamics using available knowledge concerning how various species and environmental quantities are related to each other and change over time. Based on observed general trends in vegetation structure and composition over time, the PoCM produces the expected transition to a forested state under undisturbed conditions (Begon *et al.* 1990, Burrows 1990). However the simulated dynamics under both fire regimes (runs 3 and 4) are unrealistic. Tree and shrub biomass should be suppressed and vegetation degraded by such frequent disturbances, particularly in combination with grazing damage.

The biomass dynamics of all three species are probably too rapid compared to real biomass dynamics. The same is also probably true of soil moisture dynamics. The effect of introducing low intensity grazing affects the dynamics of the grass and shrub species more than it does the tree species simulating preferential grazing of grasses and shrubs. The introduction of fire does have some effect on the dynamics of run 3 compared to run 2 but the difference between the two is small, indicating that the rules governing the effect of fire could be improved. The classification rules for vegetation type could also be improved as the forest type is reached too quickly compared to normal real-world times. The classification for forest type requires that biomass$_t$ $\geq_q$ medium. This might be better changed to biomass$_t$ $\geq_q$ high. In addition, the number of states could be increased to better describe the variation in composition. When interpreting results though, the reader should not just examine the annual data as much of the PoCM's behaviour is obscured. The monthly results must also be examined.

However, despite these criticisms it must be borne in mind that the PoCM was constructed using approximated generic species types. The model is not based on any specific set of species or empirical information. As a consequence, the behaviour of the PoCM, although reasonable in its overall form, shows some discrepancies with respect to real-world dynamics. The main point of the PoCM was to illustrate that it is possible to represent and reason with available ecological knowledge for the purpose of modelling vegetation. The results demonstrate that this is indeed possible.

With respect to the types of vegetation dynamic noted by Burrows (1990), the RBCLM3 System is capable of replicating all seven, making a strong case for the utility of the System:

- *Colonization and sequential replacement (Burrows dynamic type 1):*

  The RBCLM3 can handle species effectively decreasing in abundance, biomass etc. to the point where they are locally extinct. It can also handle locally extinct species re-growing after a suitable period of time or under the influence of an exogenous seed flow. As species biomasses change over a run, sequential replacement can be simulated.
- *Direct replacement following the disturbance of established vegetation (Burrows dynamic type 2):*

  Imagine that vegetation pre-fire consists of resprouter species like Pinus halepensis. Rules can be constructed for species such that vegetative resprouters grow more rapidly than seeder species, thereby ensuring the direct replacement and re-growth of pre-disturbance vegetation.
- *Cyclic replacement in response to endogenous or exogenous influences (Burrows dynamic type 3):*

Repeating patterns in exogenous variables can be specified and used as drivers for cyclical vegetation change. Rule sets can also be constructed that produce cyclical dynamics based upon endogenous species interactions (see PoCM annual results for an example).

- *Fluctuating replacements in response to exogenous influences (Burrows dynamic type 4):*

  As shown in run 4 monthly results grass biomass fluctuates following an annual seasonal pattern, probably in response to precipitation and radiation cycles. In doing so the grass species replaces the shrub and tree species in biomass terms periodically.
- *Vegetation maintained by frequent or continual disturbance (Burrows dynamic type 5):*

  Although not clear from the PoCM results, there is no reason to expect that rules governing species dynamics cannot be designed such that certain species are precluded or maintained at a particular abundance, biomass etc. levels by different disturbance factors and levels.
- *Vegetation in relative equilibrium for a time (Burrows dynamic type 6):*

  The PoCM annual results show that dynamic equilibrium between species can be achieved using the RBCLM3 System.
- *Long-term, gradual change in response to autogenic or allogenic influences (Burrows dynamic type 7):*

  This was not explored in the PoCM. However the effects of long-term, gradual change could be simulated by setting many species biomass, abundance etc. values to zero then letting these species grow only when conditions are right (e.g. temperature above a certain level etc.) whilst letting other species become locally extinct when conditions become less favourable. This provides a means of treating species as existing *in potentia* in a community but only 'switching them on' when conditions become suitable.

As discussed, the RBCLM3 PoCM was developed upon an 'implicit functional attributes' (IFA) basis. Other than maximum biomass, no variables were used to represent the values of functional attributes for the species concerned. This approach has been demonstrated to work with the PoCM and is useful to take if knowledge is available concerning how species or species types behave overall but not with respect to particular attributes and attribute values. However if, as with some functional attributes modelling work such as the Vital Attributes scheme (Noble and Slatyer 1980) explicit attributes can be identified and given values for different species and the dynamics of vegetation modelled based upon general rules applied to those attribute values, then an explicit functional attributes (EFA) approach could be taken with an RBCLM3 model. Figure 101 details a more EFA flavoured PoCM. The basic structure is the same but instead of the rules for determining species intermediate variables and flows being species-specific (i.e. different rules for each species, designed to implicitly reflect the properties of each species), they are general for all species but able to produce different values for different species based upon the values of influencing functional attributes (e.g. life_form or fire resistance, fresist). The RBCLM3 is capable of modelling with EFA knowledge.

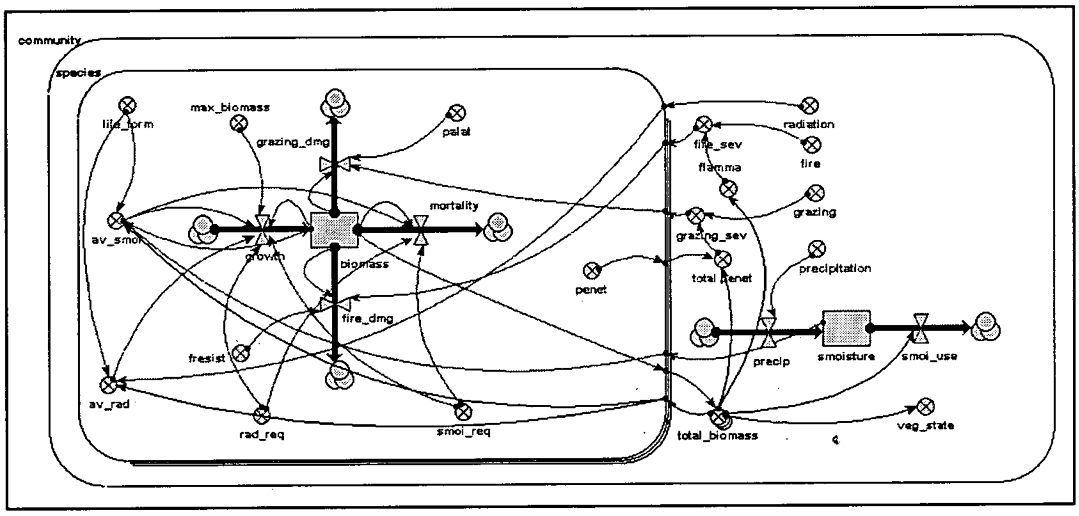


Figure 101 Explicit functional attributes version of the RBCLM3 Proof of Concept Model

6.5.2 Knowledge Representation

Current functional attributes modelling methods use specific variables with hard-wired support sets and variable specific methods for reasoning about change in value. They also dictate the set of variables (including functional attributes) that must be used to model vegetation irrespective of the knowledge that is available or appropriate.

From the idea that generalised representational and reasoning methods can be applied to modelling with non-quantitative value and relationship knowledge the compartment-flow modelling ontology was extended, embedded within a community and species object world ontology and integrated with a general method of reasoning about change in the values of non-quantitative variables. The resulting system, the RBCLM3 can therefore model species and community dynamics using any set of variables deemed appropriate. Functional attributes can be included explicitly and different functional attributes used depending on the model being constructed or functional attributes can be included implicitly. This gives the RBCLM3 a degree of flexibility and generality missing from other functional species modelling approaches.

The RBCLM3 KR scheme separates the representation of structural, value and relationship knowledge and treats variable by computational type. This means that aspects of model structure can be interrogated and changed with relative ease. There is less scope for error during model development and a higher level of guaranteed consistency between RBCLM3 models than RBCLM2 models. This marks a significant improvement in the clarity and utility of KR.

The RBCLM3 can only represent and reason with a fixed number of species. However, as discussed different species may be become locally extinct within a community if their biomass etc. reaches zero. Treating all species as existing *in potentia* is not really a problem for the RBCLM3 except in terms of computational efficiency.

6.5.3 Reasoning and Simulation

The RBCLM3 Reasoning System marks a significant improvement and increase in sophistication on the designs used for the RBCLM1 and RBCLM2. The RBCLM3 System is more akin to a general modelling environment (albeit that it is specific to community and species objects) like STELLA or Simile. The inclusion of circularity checking is a step towards ensuring that RBCLM3 models are verifiably doing what they are supposed to do and the inclusion of the calculation order and once per time-step calculation of intermediate variable and flow values a significant improvement in computational efficiency.

The reasoning method employed by the RBCLM3 to model change in the values of non-quantitative variables has been demonstrated to work effectively. It overcomes the lack of a single coherent system for doing so in the RBCLM2 and extends the basic temporal reasoning system used in the RBCLM1. The RBCLM3 Reasoning System offers a solution to both the direction of change and rate of change problems. In addition, the reasoning method extends the capabilities of the System Dynamics / compartment-flow ontology. Whether the method is useful for predicting actual vegetation dynamics remains to be seen but it is certainly capable of doing so.

Chapter 7 Rule-Based Modelling of Vegetation Dynamics: A Discussion

7.1 Reprise

The main aim of this thesis has been to develop methods capable of predictive modelling of vegetation dynamics using only the types of knowledge that are readily available. Such knowledge is characterised by its fragmented form and the often imprecise and value heterogeneous nature of the relationships between ecological quantities. Using a form of knowledge-based simulation the three ontologically distinct RBCLM systems developed demonstrate that vegetation dynamics can be predicted using only available knowledge fragments.

Underlying the development of the RBCLM systems is the separation of knowledge representation (model specification) and reasoning (simulation) and the use of declarative methods to express the fragments of ecological knowledge. By separating variable direction of change (D-Delta) and rate of change values and measuring rate of change in terms of the time required to change value (T-Delta), an effective temporal reasoning system has been developed to coherently simulate using mixtures of quantitative, qualitative and linguistic variables. Reasoning with separate rules to determine D-Delta and T-Delta values solves both the direction of change and rate of change problems and permits prediction of discrete state variable values over absolute time. Indeed the temporal reasoning system that is most developed in the RBCLM3 system could provide a more general means of modelling with non-quantitative values and relationships for use more broadly within ecology and other domains depending on the form of available knowledge.

The RBCLM3 system could be viewed as an RBCLM* system (i.e. more general) as it is capable of doing the modelling handled by the other two systems. The RBCLM3, although it is species-based uses a community 'object'. There is no reason in principle why vegetation state could not be modelled using a linguistic state variable located in the community object. Likewise, community-level vegetation attributes and properties could be represented as variables in the community object to model using the RBCLM2 ontology. By doing so the full RBCLM3 temporal reasoning system could be utilised for modelling in terms of any of the three RBCLM ontologies. A few technical changes to the RBCLM3 implementation and to the semantics of its ontology would be required however as the RBCLM3 was not designed to be an RBCLM*.

The approach of determining intermediate variable values, including D-Delta, using only known value correspondences and inequalities, and of updating non-quantitative state variable values based only on their current D-Delta and T-Delta values has shown that it is possible to generate meaningful predictions when modelling with incomplete knowledge without succumbing to the excessive branching problem. The facts used to specify model constants and the rules used to represent how variable values are related to each other can be viewed as a set of constraints. The deductive simulation engines used by the three RBCLMs can be viewed as using constraint propagation to predict changes given (starting) state over time. Although Kuipers (1994) notes that constraint propagation can be blocked by an insufficient set of constraints, the RBCLM approach demonstrates that by defining an appropriate set of constraints and using an appropriate system for reasoning with them, constraint propagation can work well as a means of simulating with incomplete knowledge. There appears to be no benefit from using available ecological knowledge within weak qualitative relations then subjecting those relations to exhaustive constraint satisfaction as is the approach taken in QR. When the form of relationships between quantities are only partially known, as they typically are in vegetation ecology, the view taken here is that it is better to deduce the consequences of what is definitely known in a systematic and tractable manner than attempt to enumerate the whole state space of possibilities. Although it may be true that the RBCLM approach requires more information to specify a model than a typical QR technique the results are more likely to be useful for science and management.

7.2 Handling Incomplete Knowledge

7.2.1 Imprecision

Available knowledge about vegetation can be regarded as incomplete. This means that it is imprecise and / or indeterminate, only partially characterising objects, quantities and relationships. The approach taken during this thesis was to represent imprecise value knowledge using discrete qualitative values representing intervals, points or linguistic terms. An alternative approach is taken by fuzzy set theory wherein imprecise knowledge is represented as a set of discrete values (representing the division of a continuous scale) and probabilistic membership values (see for example Bolloju 1996). Variables can then be valued as being members of more than one adjacent discrete value by having membership values for each one.

Describing qualitative variable values using fuzzy set memberships does not match the more discretely categorical form of available ecological knowledge. This is in agreement with authors from other domains such as Camara *et al.* (1987) (water resource sciences) and Kuipers (1994) (physical systems) that the fuzzy set representation does not suit intuitive domain understanding closely. Although representing and reasoning with fuzzy membership values may permit finer grained predictions to be made it is not clear that such extra precision is either useful or needed. The strictly categorical approach used in the RBCLMs mixed with quantitatively valued variables where appropriate appears adequate for the task of predicting vegetation dynamics.

7.2.2 Rule-Base Structure and Indeterminacy

Each RBCLM rule-base is composed of sets of clauses for different predicates in the form of factual or conditional statements. During RBCLM simulation the Prolog 'engine' follows a particular execution or search model when trying to determine whether the conditions of a rule are true or what the value of an argument is. Clauses for the predicate being queried are accessed and tested sequentially from the first to the last. Unless explicitly told to search for all solutions the Prolog engine will stop as soon as it has found a matching clause. Certain aspects of each RBCLM system's operation uses a 'find all solutions' method but in calculating a variable's value or determining how a value will change the RBCLM systems all use the standard Prolog engine search strategy. Rules (clauses) are searched sequentially until a matching one is found then searching stops.

This means that the ordering of rules and facts in a rule-base or database may have an effect on model behaviour. If more than one rule could apply under a given set of conditions (e.g. if more than two variable value determination rules could apply) the current implementation of the RBCLM simulation engines / reasoning systems will not find the second (or third etc.) solution.

To work around this feature of the RBCLM systems attention must therefore be paid to rule ordering. If there are 2 or more rules which could be used under the same conditions (argument values) then the one most likely to apply (strongest precedence) according to domain-specific knowledge should be put first in the rule-base then the second and so on. This technique should be followed for example in constructing D-Delta rule-bases in RBCLM3. D-Delta = zero is assumed to take precedence over all other possibilities (it represents sudden forced change to zero such as might happen to biomass after a fire) and so D-Delta = zero rules should be placed first in the D-Delta rule-base.

It is however possible that more than one rule may apply under a given set of conditions when determining a variable value in an RBCLM rule-base. Such cases are termed here 'genuine' indeterminacy to distinguish them from the 'forced' indeterminacy that occurs in QR techniques due to envisioning with weak relations. Three possible routes could be taken here to resolve the genuine indeterminacy problem:

1. *Enforce single solutions*

   Model developers could be advised to ensure all rules in all rule-bases must be mutually exclusive i.e. one rule for one set of conditions. Different rule definitions could then be tried for different runs to explore their effects on model behaviour. Done systematically this could prove to be a useful way of determining errors in rule-bases, revising knowledge and improving understanding. Changing rule definitions and / or orderings between runs can be viewed as being analogous to changing parameter values between runs in traditional numerical modelling. However it may be overly labour intensive to determine whether all rules are really mutually exclusive.
2. *Enforce single solutions but try different possibilities:*

   The RBCLM approach could be extended to find all solutions to each rule query and flag up the occurrences (time-step and variable) where more than one solution exists. The user could then try different rule orderings or rule definitions in different runs to explore the consequences of different rules being used. This is a similar solution to (1) but frees the user from having to ensure all rules are mutually exclusive before running and may provide a means of guiding the systematic testing of different rule definitions and orderings.
3. *Branching*:

   It would be possible to extend the RBCLM approach such that whenever determining a variable value all possible values (compatible rules) are searched for rather than just the first. If more than one solution could hold then the simulation could branch. This may suit uses whereby all possible solutions are desired. However the approach could end up producing excessively branched simulations and falling foul of the problems suffered by QR techniques.

Indeterminacy could also be handled by means of probabilistically modelling changes in discrete variable values then using Monte Carlo simulation techniques to determine behaviour (Manly 1991). The lack of data and difficulty in estimating discrete variable value changes is a well-recognised problem in Markov Chain-based ecological modelling and available ecological knowledge is not typically expressed probabilistically. In any case it is not clear that there is a need for probabilistic reasoning when modelling vegetation dynamics. Modelling using deterministic knowledge fragments appears to be sufficient although further testing is required, particularly concerning the development and validation of rule-based models in the three RBCLM systems. In addition the use of determinism in RBCLM modelling permits the consequences of available knowledge to be clearly deduced, followed and understood. Deterministic rule-based modelling, although perhaps not providing precise predictions, certainly appears capable of modelling the overall dynamics exhibited by vegetation.

7.3 Uses for Rule-Based Vegetation Models

'*It is better to obtain an imprecise or partial answer to the right question rather than a precise answer to the wrong question*' (Leitch *et al.* 1990). Taking this maxim as a guiding principle, rule-based models developed in an RBCLM system can be used for both advancing scientific understanding and for providing appropriately grained decision support. Deterministic ecological knowledge fragments may represent the informal result of abstracting from many empirical observations and experiments. Being able to formalise them in a rule-based system and deduce their consequences is a potentially valuable advance. Many fragments of knowledge when reasoned with together may produce unexpected and non-trivial conclusions. The production of unexpected outcomes from reasoning with individual knowledge fragments can be viewed as a process of knowledge-based emergence.

To support the process of theory development in vegetation ecology it is important to have tools that permit the consequences of new hypotheses to be deduced in the context of existing knowledge. For example, the ability to explore the corpus of knowledge that has been amassed in vegetation ecology over the past century could provide a means of relating species-level structure and function with the formation and dissolution of community patterns. If this topic could be studied at the landscape-level with multiple communities significant advances in understanding relationships between scales may be achievable.

Models are well suited to this task and can provide useful virtual experimentation platforms. The RBCLM systems, particularly the RBCLM3 (given present conclusions that a theory of vegetation dynamics, when developed, will be species-based), provide one such model-based tool. The contribution of the RBCLM approach is to permit the types of hypotheses and knowledge that ecologists have concerning, for example, the characteristics of species and species types, to be made operational for model-based investigations. Other ecological modelling techniques tend to require quantitative specification of relationships and parameterisation / initialisation data that is not available, potentially forcing model developers into the position of having to estimate or even guess relationships and values. The obvious danger here is that such models end up producing 'a precise answer to the wrong question'.

The utility of a model depends on how well it corresponds to reality and answers the problem for which it was designed to address. Corresponding to reality does not however mean being as precise as possible. Phenomena in the natural world occur at various spatial, temporal and organisational scales and as such are nested inside and contain many other phenomena. In a similar way, the causes of patterns can be nested into hierarchies – those acting from levels below and those acting from levels above phenomena of interest. Causes may take various forms – mechanisms, the direct interactions between lower level system components that result in certain patterns and structures, or constraints, behaviourally enabling or restricting system features of scales above and below that which the observed phenomenon is occurring. Causes can always be broken down into more and smaller level causes but, as both Holland (1998) and Pickett *et al.* (1994) point out, this form of reductionism is both futile and unnecessary – phenomena may be adequately explained and understood without resorting to the smallest possible scales. For example, galaxy formation is not explained by means of quantum physics. The degree to which a phenomenon has to be examined and explained in terms of lower level mechanisms and processes depends on the system concerned and the purpose of the explanation.

Theories should be internally consistent but not necessarily complete and there may be more than one theory held to be true about a particular phenomena given that each theory has a distinct domain (scale, aspect of phenomena explained). Viewing models as theories, it is perfectly acceptable to place limits on the application of a model without harming that model's credibility (Pickett *et al.* 1994).

When seeking to predict vegetation dynamics, it may be sufficient to model at a low resolution depending on the nature of the vegetation and the purpose of the model. With respect to model utility there can be important differences between models designed to increase scientific understanding (research models) and model designed to provide information to support management decisions (policy models) (Engelen *et al.* 2000). Both types of models are necessarily problem oriented. Research models are typically designed to address problems concerned with furthering our understanding of the processes and patterns that occur in the world. Policy models are typically designed to address problems concerned with understanding the effects of management actions. In vegetation ecology the resolution required to further our understanding of process and pattern is currently accepted as being the species-level. Species-level explanations are mechanistic and, as such, are likely to provide greater capability in accounting for causes and observed patterns (Pickett *et al.* 1994). Community-level approaches are phenomenological and therefore not likely to yield robust insight (Pickett *et al.* 1994). However, if applied under certain conditions, phenomenological models may provide a useful, quick and equally robust means of predicting the behaviour of phenomena (Holland 1998).

It is therefore likely that the RBCLM1 and RBCLM2 systems will not provide sufficient resolution to be useful for furthering our understanding of when, how and (crucially) why vegetation changes. They may however prove to be of use in providing coarse-grained predictions of community-level change to inform decision-makers about the likely consequences of management actions such burning, cutting etc. Indeed the RBCLM2 system has already been applied to providing integrated environmental decision support in the context of the MODULUS decision support system (Engelen *et al.* 2000).

The RBCLM3 system however has sufficient resolution that it could be used to further scientific understanding of how species-level function and behaviour relates to community-level pattern. Indeed, because single deterministic solutions are generated using the RBCLM approach it may be easier to systematically test different rule formulations to determine which models, which hypotheses, best correspond to empirical observations.

7.4 Modelling Systems Research

The separation of representation from reasoning utilised in the three RBCLM systems is in line with current thinking regarding the development of a standard declarative ecological modelling language as the best means of approaching the development of modelling systems (Muetzelfeldt 1999). The idea of declaratively specifying models separately from reasoning with them follows on from the notion of a model blueprint developed by Muetzelfeldt *et al.* (1989). Procedurally implemented programs mix declarative knowledge concerning model structure and relationships with control statements. This means that model structures, relationships and assumptions tend to be hard-wired, unclear and difficult to modify or update. It also means that models are difficult to re-use or combine leading to huge redundancies in model development effort. Separating out the description of the problem (the model representation) from the way in which it is reasoned with can solve these problems and provide the basis to running models along with performing other reasoning tasks such as structural comparison between models, model structure and relationship explanation / interrogation, automatic visualisation etc. (Muetzelfeldt *et al.* 1989, Robertson *et al.* 1991, Muetzelfeldt 1999). To facilitate this view of modelling the development of a standardised representational syntax for ecological models is required. The three RBCLM systems, particularly the RBCLM3, where temporal reasoning and separation of model description from reasoning is most developed, may contribute to this effort by identifying possible ways in which non-quantitative value and relationship knowledge can be used.

As they stand, the RBCLM systems constitute domain-specific languages and reasoning systems for modelling vegetation dynamics (Fall and Fall 2001). However it may be possible to use the temporal reasoning system of the RBCLM3 for extending the System Dynamics ontology to provide rule-based non-quantitative modelling capabilities. The RBCLM3 is already based on the ontology making extension potentially easier to achieve. By doing so the concepts underlying the RBCLM3 may be used more generally as part of the drive to develop a standard ecological modelling language.

7.5 What Next?

The main task to be tackled next in the development of rule-based modelling of vegetation dynamics using the RBCLM systems is the construction, verification and validation of vegetation models. This will provide more the substantial and compelling evidence that is needed regarding the utility of the RBCLM approach before it can confidently be applied for research or policy purposes. Available knowledge could be collected using knowledge acquisition exercises and literature search and analysis. There is a growing body of knowledge regarding the functional attributes of species and species types that could prove invaluable to this effort. In addition, because of the phytosociological tradition, ecologists studying the Mediterranean region have substantial amounts of state and property-based community-level knowledge that could be used to develop and test RBCLM1 and RBCLM2 models. Experiences within the ModMED II and ModMED III EU projects have shown this to be possible (McIntosh *et al.* 2001).

There are spatialisation possibilities with the RBCLM systems. Their logic-based KR schemes can be extended to include spatially distributed factors like seed rain – variables could be included for seed generation and variable value determination and update rules could be made conditional on the values of such factors. To reason about space RBCLM models could be incorporated into a spatial simulation environment. This has already been achieved using an RBCLM2 model that was integrated into the Geonamica environment as part of the Modulus DSS (Engelen *et al.* 2000). The RBCLM2 model was responsible for (amongst other things) determining the type of seeds produced by each community cell across a raster landscape. Geonamica, with the aid of tailor-made sub-model, was responsible for distributing the seed types across the landscape to each RBCLM2 community cell. The RBCLM2 model then used the seed types present in each cell in the rules to update vegetation type.

Currently the RBCLM systems cannot handle spatial reasoning / simulation operations so embedding in a simulation environment is the only option for spatial rule-based modelling. However it may be possible to extend their reasoning systems to control simulation over multiple communities. The RBCLM3 system object data structure has been designed such that it can contain state variable information for *n* communities, each containing *n* species. Robertson *et al.* (1995) detail how logic can be used to specify relations (e.g. adjacency, distance etc.) between spatial areas, nominally a grid, but the approach is extensible to non-uniform spaces. If the RBCLM3 reasoning system could be extended to control simulation of multiple communities along with a means of extracting information (variable values) from each community and giving these values to other communities based on rules referring to spatial relations then there would be no need for an external simulation environment. The present RBCLM3 reasoning system could in principle be extended to loop over multiple communities rather than a single community. The non-trivial component of this task would be to co-ordinate the collection of spatial information (e.g. seed production) and distribute this information across multiple communities appropriately (e.g. distance-based seed distribution) whilst maintaining correct variable calculation order in terms of 'calculate rate, update state'.

Another possible avenue worth exploring is the possibility that available knowledge regarding quantity T-Delta values under different conditions may be expressed as ranges of values rather than single values. For example, under low light it may known that species *i* takes between 4 and 6 years to grow from low to medium biomass. The RBCLM temporal reasoning system could be extended such that when the basic T-Delta value is being calculated a value is randomly picked from a given range rather than the same single value always being chosen. It is unclear whether such an extension is required or whether it would have a substantial impact on model behaviour. However it could prove to be an interesting way of more accurately capturing ecological knowledge if it is found to be of this form and of representing stochasticity in vegetation dynamics.

Last of all, the task of developing RBCLM system models is at present one that involves hand-coding in Prolog. The RBCLM system KR schemes provide a structure to follow but the task could be made easier. To facilitate RBCLM model construction a graphical-user interface (GUI) could be implemented for rule-base development / knowledge representation. Such a GUI could be used to improve the clarity of rule syntax and structure clarity through the use of a higher-level representation more akin to natural language. In addition a GUI could be designed to facilitate the execution of RBCLM models and for displaying results.

7.6 Concluding Statement

Model-based tools and studies play a crucial role in complex scientific domains like vegetation ecology. The ability to utilise as many different types of knowledge as possible in modelling can only be of benefit. Despite the fact the symbolic computation in the form of Prolog has been in existence since the 1970s the problem of utilising non-quantitative knowledge for predictive modelling of vegetation dynamics has only been poorly tackled in the past, if at all. It is hoped that the present study has provided a sound evaluation of the various ways in which such knowledge can be utilised and, through the development of the temporal reasoning system, identified an effective means of putting it to use in pure and applied modelling.

Bibliography

Aber,J.D. & Federer,C.A. (1992) *A generalized, lumped-parameter model of photosynthesis, evapotranspiration and net primary production in temperate and boreal forest ecosystems. Oecologia*, **92**, 463-474.

Arianoutsou,M. & Ne'eman,G. (2000) Post-Fire Regeneration of Natural *Pinus halepensis* forests in the East Mediterranean Basin.*IN:* Ne'eman, G. and Trabaud, L. (2000), *Ecology, Biogeography and Management of Pinus halepensis and Pinus brutia Forest Ecosystems in the Mediterranean Basin*, Backhuys Publishers, Leiden, Netherlands.

Armstrong,H.M. & Milne,J.A. (1995) The Effects of Grazing on Vegetation Species Composition.*IN:* Usher, M. B., Thompson, D. B. A., and Meater, A. J. (1995), *Heaths and Moorlands: Cultural Landscapes*, 1st, HMSO, Edinburgh.

Begon,M., Harper,J.L. & Townsend,C.R. (1990) *Ecology; Individuals, Populations and Communities.* Blackwell Science, Cambridge, MA.

Begon,M. & Morimer,M. (1986) *Population Ecology, A Unified Study of Animals and Plants.* Blackwell Scientific Publications, Oxford.

Blondel,J. & Aronson,J. (1999) *Biology and Wildlife of the Mediterranean Region.* Oxford University Press, Oxford, UK.

Bolloju,N. (1996) *Formulation of qualitative models using fuzzy logic. Decision Support Systems*, **17**, 275-298.

Bond,W.J. & van Wilgen,B.W. (1996) *Fire and Plants, Population and Community Biology Series 14.* Chapman and Hall, London.

Bonissone, P. P. and Valavanis, K. P. (1985), *A Comparative Study of Different Approaches to Qualitative Physics Theories, IN:* 1985), Second AI Applications Conference, IEEE.

Bossel,H. & Schäfer,H. (1989) *Generic Simulation Model of Forest Growth, Carbon and Nitrogen Dynamics, and Application to Tropical Acacia and European Spruce. Ecological Modelling*, **48**, 221-265.

Botkin, D. B. (1974), *Functional Groups of Organisms in Model Ecosystems, IN:* Levin, S. A. (7-1-0074), Ecosystem Analysis & Prediction, Proceedings of a SIAM-SIMS Conference, Society for Industrial & Applied Mathematics, Philadelphia.

Bredeweg,B. (1992) *Expertise in qualitative prediction of behaviour.* PhD thesis, University of Amsterdam.

Bugmann,H., Fischlin,A. & Kienast,F. (1996) *Model convergence and state variable update in forest gap models. Ecological Modelling*, **89**, 197-208.

Burrows,C.J. (1990) *Processes of Vegetation Change.* Unwin Hyman, London.

Camara,A.S., Pinheiro,M., Antunes,M.P. & Seixas,M.J. (1987) *A New Method for Qualitative Simulation of Water Resources Systems: 1. Theory. Water Resources Research,* **23**, 2015-2018.

Clancey,W.J. (1993) *The Knowledge Level Reinterpreted: Modelling Socio-Technical Systems. International Journal of Intelligent Systems,* **8**, 33-49.

Clements,F.E. (1904) *The development and structure of vegetation. Botanical Survey of Nebraska,* **3**, 1-175.

Clements,F.E. (1916) *Plant succession: an analysis of the development of vegetation. Publications of the Carnegie Institute,* **242**.

Clements,F.E. (1928) *Plant succession and indicators.* Wilson, New York.

Clocksin,W.F. (1997) *Clause and Effect; Prolog Programming for the Working Programmer.* Springer-Verlag, Berlin, Germany.

Cohn,A.G. (1989) *Approaches to Qualitative Reasoning. Artificial Intelligence Review,* **3**, 177-232.

Costanza,R., Wainger,L., Folke,C. & Masler,K. (1993) *Modeling Complex Ecological Economic Systems: Toward an Evolutionary, Dynamic Understanding of Humans and Nature. BioScience,* **93**.

Cowles,H.C. (1899a) *The ecological relations of the vegetation on the sand dunes of Lake Michigan Part 1. Botanical Gazette,* **27**, 95-117.

Cowles,H.C. (1899b) *The ecological relations of the vegetation on the sand dunes of Lake Michigan Part 2. Botanical Gazette,* **27**, 167-202.

Cowles,H.C. (1899c) *The ecological relations of the vegetation on the sand dunes of Lake Michigan Part 3. Botanical Gazette,* **27**, 281-308.

Cowles,H.C. (1899d) *The ecological relations of the vegetation on the sand dunes of Lake Michigan Part 4. Botanical Gazette,* **27**, 361-391.

Cowles,H.C. (1901) *The physiographic ecology of Chicago and vicinity. Botanical Gazette,* **31**, 73-108.

Cowles, H. C. (1910), *The fundamental causes of succession among plant associations,* 1909,

Cowles,H.C. (1911) *The causes of vegetative cycles. Botanical Gazette,* **51**, 161-183.

Cropper,W.P. & Carter Ewel,C. (1987) *A Regional Carbon Storage Simulation for Large-Scale Biomass Plantations. Ecological Modelling,* **36**, 171-180.

Dale,M.R.T. (1999) *Spatial Pattern Analysis in Plant Ecology.* Cambridge University Press, Cambridge, UK.

D'Ambrosio, B. (1987), *Extending the Mathematics in Qualitative Process Theory, IN:* 1987), Proceedings of AAAI-87, Morgan Kaufmann.

Davis,J.R., Nanninga,P.M., Hoare,J.R.L. & Press,A.J. (1989) *Transferring Scientific Knowledge to Natural Resource Managers Using Artificial Intelligence Concepts. Ecological Modelling*, **46**, 73-89.

Deaton,M.L. & Winebrake,J.J. (2000) *Dynamic Modeling of Environmental Systems.* Springer, New York.

de Kleer, J. and Brown, J. S. (1983), *The Origin, Form and Logic of Qualitative Physical Laws, IN:* Bundy, A. (8-8-1983), Proceedings 8th International Joint Conference on Artificial Intelligence (IJCAI-83).

de Kleer,J. & Brown,J.S. (1984) A Qualitative Physics Based on Confluences. *IN:* Weld, D. S. and de Kleer, J. (1990), *Readings in Qualitative Reasoning about Physical Systems*, 1, Morgan Kaufmann, San Mateo, CA.

Ellison,A.M. & Bedford,B.L. (1995) *Response of a wetland vascular plant community to disturbance: a simulation study. Ecological Applications*, **5**, 109-123.

Engelen, G., van der Meulen, M., Hahn, B., Uljee, I., Mulligan, M., Reaney, S., Oxley, T., Blatsou, C., Mata-Porras, M., Kahrimanis, S., Mazzoleni, S., Coppola, A., Winder, N. P., van der Leeuw, S., and McIntosh, B. S. (2000), *MODULUS: A Spatial Modelling Tool for Integrated Environmental Decision Making*, **EU FP4 Modulus Project (ENV4-CT97-0685)** Final Report, **EU DG XII , Brussels.**

Facelli,J.M. & Pickett,S.T.A. (1990) *Markovian Chains and the Role of Hostory in Succession. Trends in Ecology & Evolution*, **5**, 27-30.

Fall,A. & Fall,J. (2001) *A domain-specific language for models of landscape dynamics. Ecological Modelling*, **137**, 1-21.

Fedra,K. (1995) *Decision Support for Natural Resources Management: Models, GIS, and Expert Systems. AI Applications*, **9**, 3-19.

Forbus,K.D. (1984) Qualitative Process Theory. *IN:* Weld, D. S. and de Kleer, J. (1990), *Readings in Qualitative Reasoning about Physical Systems*, Morgan Kaufmann, San Mateo, CA.

Forbus,K.D. (1988) Qualitative Physics: Past, Present, and Future. *IN:* Weld, D. S. and de Kleer, J. (1990), *Readings in Qualitative Reasoning about Physical Systems*, Morgan Kaufmann, San Mateo, CA.

Forbus,K.D. (1990) The Qualitative Process Engine. *IN:* Weld, D. S. and de Kleer, J. (1990), *Readings in Qualitative Reasoning about Physical Systems*, Morgan Kaufmann, San Mateo, CA.

Ford,K.M., Bradshaw,J.M., Adams-Webber,J.R. & Agnew,N.M. (1993) *Knowledge Acquisition as a Constructive Modeling Activity. International Journal of Intelligent Systems*, **8**, 9-32.

Ford,A. (1999) *Modeling the Environment, An Introduction to System Dynamics Modeling of Environmental Systems.* Island Press, Washington, D.C.

Franklin,J., Syphard,A.D., Mladenoff,D.J., He,H.S. , Simons,D.K., Martin,R.P., Deutschman,D. & O'Leary,J.F. (2001) *Simulating the effects of different fire regimes on plant functional groups in Southern California. Ecological Modelling*, **142**, 261-283.

Gaines,B.R. (1993) *Modeling Practical Reasoning. International Journal of Intelligent Systems, 8, 51-70.*

Gillison,A.N. & Carpenter,G. (1997) *A generic plant functional attribute set and grammar for dynamic vegetation description and analysis. Functional Ecology*, **11**, 775-783.

Gitay,H. & Noble,I.R. (1997) What are functional types and how should we seek them? *IN:* Smith, T. M., Shugart, H. H., and Woodward, F. I. (1997), *Plant Functional Types, their relevance to ecosystem properties and global change*, Cambridge University Press, Cambridge, UK.

Glenn-Lewin,D.C., Peet,R.K. & Veblen,T.T. (1992) *Plant Succession: Theory and Prediction.* Chapman and Hall, London.

Glenn-Lewin,D.C. & van der Maarel,E. (1992) Patterns and processes of vegetation dynamics.*IN:* Glenn-Lewin, D. C., Peet, R. K., and Veblen, T. T. (1992), *Plant Succession: Theory and Prediction*, 1, Chapman and Hall, London.

Gordon,S.L. (1989) *Theory and Methods for Knowledge Acquisition. AI Applications*, **3**, 19-30.

Grime,J.P. (1985) Towards a Functional Description of Vegetation. *IN:* White, J. (1985), *The Population Structure of Vegetation*, 1st, Dr. W. Junk Publishers, Netherlands.

Guerrin,F. (1991) *Qualitative reasoning about an ecological process: interpretation in hydroecology. Ecological Modelling*, **59**, 165-201.

Guerrin,F. (1992) *Model-based Interpretation of Measurements, Analyses, and Observations of an Ecological Process. AI Applications*, **6**, 89-101.

Guerrin,F., Bousson,K., Steyer,J.-P. & Trave-Massuyes,L. (1994) *Qualitative Reasoning Methods for CELSS Modelling. Advances in Space Research*, **14**, 307-312.

Guerrin, F. (1995), *Dualistic Algebra for Qualitative Analysis, IN:* Bredeweg, B. (5-16-1995), Working Papers Ninth International Workshop on Qualitative Reasoning (QR'95), Dept. of Social Science Informatics, University of Amsterdam, Amsterdam, The Netherlands.

Guerrin, F., Dumas, J., Davaine, P., Beall, E., and Clement, O. (1997), *Qualitative modeling of the impact of the environment on early stages of salmon populations, IN:* Ironi, L. (6-3-1997), Proceedings of the 11th International Workshop on Qualitative Reasoning, Istituto di Analisi Numerica, CNR Pavia, Italy.

Hannon,B. & Ruth,M. (1994) *Dynamic Modeling*. Springer-Verlag, New York.

Heathfield,D. (2001), *Landlord – Landscape Modelling Environment*, ModMED: Modelling Mediterranean Ecosystem Dynamics, Final ModMED III Report (ENV4-CT97-0680), EUDGXII, Brussels, Belgium.

Heller,U. and Struss,P. (1996), *Transformation of Qualitative Dynamic Models - Application in Hydroecology, IN:* Qualitative Reasoning: The Tenth International Workshop, AAAI Press Technical Report WS-96-01, AAAI Press, Menlo Park CA.

Hobbs,R.J. (1997) Can we use plant functional types to describe and predict responses to environmental change? *IN:* Smith, T. M., Shugart, H. H., and Woodward, F. I. (1997), *Plant Functional Types, their relevance to ecosystem properties and global change*, 1st, Cambridge University Press, Cambridge.

Holland,J. (1998) *Emergence, from Chaos to Order.* Oxford University Press, Oxford, UK.

Hunt,J.E. & Cooke,D.E. (1994) *Qualitatively Modelling Photosynthesis. Applied Artificial Intelligence*, **8**, 307-332.

Isagi,Y. & Nakagoshi,N. (1990) *A Markov Approach for Describing Post-fire Succession of Vegetation. Ecological Research*, **5**, 163-171.

Johnson,A.R. (1996) Spatiotemporal Hierarchies in Ecological Theory and Modeling. *IN:* Goodchild, M. F., Steyaert, L. T., Parks, B. O., Johnston, C., Maidment, D., Crane, M., and Glendinning, S. (1996), *GIS and Environmental Modeling: progress and Research Issues*, GIS World Books, Fort Collins, Colorado.

Judson,O.P. (1994) *The rise of the individual-based model in ecology. Trends in Ecology & Evolution*, **9**, 9-14.

Keane, R. E., Long, D. G., Menakis, J. P., Hann, W. J., and Bevins, C. D. (1996), *Simulating Coarse-Scale Vegetation Dynamics Using the Columbia River Basin Succession Model - CRBSUM*, General Technical Report INT-GTR-340, USDA, Forest Service, Intermountain Research Station, Utah.

Keeley,J.E. (1986) Resilience of mediterranean shrub communities to fires. *IN:* Dell, B., Hopkins, A. J. M., and Lamont, B. B. (1986), *Resilience in Mediterranean-Type Ecosystems, Tasks for Vegetation Science 16*, 1st, Dr. W. Junk, The Hague.

Kitzmiller, C. T. (1988), *Simulation and AI: Coupling symbolic and numeric computing, IN:* Henson, T. (3), Proceedings of the SCS Multiconference on Artificial Intelligence and Simulation: The Diversity of Applications, SCS International, San Diego.

Kuipers,B. (1986) Qualitative Simulation. *IN:* Weld, D. S. and de Kleer, J. (1990), *Readings in Qualitative Reasoning about Physical Systems*, Morgan Kaufmann, San Mateo, CA.

Kuipers,B. (1994) *Qualitative Reasoning, Modeling and Simulation with Incomplete Knowledge.* The MIT Press, London, England.

Kurz,W.A., Beukema,S.J., Klenner,W., Greenough,J.A., Robinson,D.C.E., Sharpe,A.D. & Webb,T.M. (2000) *TELSA: The Tool for Exploratory Landscape Scenario Analyses. Computers and Electronics in Agriculture*, **27** , 227-242.

Lavorel,S., McIntyre,S. & Grigulis,K. (1999a) *Plant response to disturbance in a Mediterranean grassland: How many functional groups? Journal of Vegetation Science*, **10**, 661-672.

Lavorel,S., McIntyre,S., Landsberg,J. & Forbes,T.D.A. (1997) *Plant functional classifications: from general groups to specific groups based on response to disturbance. Trends in Ecology & Evolution*, **12**, 474-478.

Lee,K.N. (1993) *Compass and Gyroscope, Integrating Science and Politics for the Environment.* Island Press, Washington D.C.

Le Houérou,H.N. (1981) Impact of Man and his Animals on Mediterranean Vegetation. *IN:* di Castri, F., Goodall, D. W., and Specht, R. L. (1981), *Mediterranean-Type Shrublands; Ecosystems of the World 11*, Elsevier , Amsterdam.

Legg,C.J. (1980) *A Markovian Approach to the Study of Heath Vegetation Dynamics. Bulletin of Ecology*, **11**, 393-404.

Legg, C. L. (1995), *Markov and Transition Matrix Models.* Unpublished ModMED Project Report, The University of Edinburgh 1995.

Leitch,R.R., Wiegand,M.E. & Quek,H.C. (1990) *Coping with Complexity in Physical System Modelling. AICOM*, **3**, 48-57.

Levins,R. (1966) *The strategy of model building in population biology. American Scientist*, **54**, 421-431.

Loehle,C. (1987) *Philosophical tools: potential contributions to ecology, Oikos*, **51**, 97-104.

Logofet,D.O. & Lesnaya,E.V. (2000) *The mathematics of Markov models: what Markov chains can really predict in forest successions. Ecological Modelling*, **126**, 285-298.

Luger,G.F. & Stubblefield,W.A. (1993) *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. The Benjamin/Cummings Publishing Company, Redwood City, CA.

Luken,J.O. (1990) *Directing Ecological Succession*. Chapman & Hall, London.

Manly,B.F.J. (1991) *Randomization and Monte Carlo Methods in Biology*. Chapman and Hall , London, UK.

Mauchamp,A., Rambal,S. & Lepart,J. (1994) *Simulating the dynamics of a vegetation mosaic: a spatialized functional model. Ecological Modelling*, **71**, 107-130.

May,R.M. (1973) *Qualitative Stability in Model Ecosystems. Ecology*, **54**, 638-641.

Mazzoleni, S. and Legg, C. J. (2001), *ModMED: Modelling Mediterranean Ecosystem Dynamics*, ModMED III Final Report ENV4-CT97-0680, EU DGXII, Brussels, Belgium.

McGlade,J.M. (1999) Ecosystem analysis and the governance of natural resources. *IN:* McGlade, J.M. (1999), *Advanced Ecological Theory, Principles and Applications*, 1st, Blackwell Science Ltd., Oxford, UK.

McIntosh, B. S., Legg, C. J., Csontos, P., Arianoutsou, M., and Mazzoleni, S. (2001), *Rule-Based Modelling*, ModMED: Modelling Mediterranean Ecosystem Dynamics, Final Report ModMED III Project (ENV4-CT97-0680), EU DGXII, Brussels, Belgium.

McIntyre,S., Diaz,S., Lavorel,S. & Cramer,W. (1999) *Plant functional types and disturbance dynamics - Introduction. Journal of Vegetation Science*, **10**, 604-608.

McRoberts,R.E., Schmoldt,D. & Rauscher,H.M. (1991) *Enhancing the Scientific Process with Artificial Intelligence: Forest Science Applications. AI Applications*, **5**, 5-26.

Moore,R.C. (1982) The Role of Logic in Knowledge Representation and Commonsense Reasoning. *IN:* Brachman, R. J. and Levesque, H. J. (1985), *Readings in Knowledge Representation*, Morgan Kaufmann, Los Altos.

Moore,A.D. & Noble,I.R. (1990) *An Individualistic Model of Vegetation Stand Dynamics. Journal of Environmental Management*, **31**, 61-81.

Moore,A.D. & Noble,I.R. (1993) *Automatic model simplification: the generation of replacement sequences and their use in vegetation modelling. Ecological Modelling*, **70**, 137-157.

Mueller-Dombois,D. & Ellenberg,H. (1974) Succession, Climax and Stability. *IN: Aims and Methods of Vegetation Ecology*, John Wiley and Sons.

Muetzelfeldt,R.I. (1995) *A framework for a modular modelling approach for agroforestry. Agroforestry Systems*, **30**, 223-234.

Muetzelfeldt, R. I. *Workshop 9: Common Standards for Agro-Ecological Modelling*. Food and Forestry: Global Change and Global Challenges, GCTE Focus 3 Conference. 1999.

Muetzelfeldt,R.I., Robertson,D., Bundy,A. & Uschold,M. (1989) *The use of Prolog for improving the rigour and accessibility of ecological modelling. Ecological Modelling*, **46**, 9-34.

Nahal,I. (1981) The Mediterranean Climate from a Biological Viewpoint. *IN:* di Castri, F., Goodall, D. W., and Specht, R. L. (1981), *Ecosystems of the World II, Mediterranean-Type Shrublands*, 1, Elsevier, Oxford.

Nanninga, P. M. and Davis, J. R. (1985), *An Overview of Expert Systems*, CSIRO Technical Memorandum 85/2, CSIRO, Canberra.

Naveh, Z. (1990), *Fire in the Mediterranean - A Landscape Ecological Perspective, IN:* Goldammer, J. G. and Jenkins, M. J. (1989), Proceedings of the Third International Symposium on Fire Ecology, SPB Academic Publishing, The Hague.

Naveh,Z. & Liebermann,A.S. (1985) *Landscape Ecology*. Springer-Verlag, New York.

Noble, I. R. (1985), *Fire effects, vital attributes and expert systems*, CSIRO Technical Memorandum 85/2, CSIRO, Canberra.

Noble,I.R. (1987) *The role of expert systems in vegetation science. Vegetatio*, **69**, 115-121.

Noble,I.R. & Gitay,H. (1996) *A functional classification for predicting the dynamics of landscapes. Journal of Vegetation Science*, **7**, 329-336.

Noble, I. R. and Slatyer, R. O. (1977), *Post-fire Succession of Plants in Mediterranean Ecosystems, IN:* Mooney, H. A and Conrad, C. E. (Proceedings of the Symposium on Emvironmental Consequences of Fire and Fuel Management, USDA Forest Service.

Noble,I.R. & Slatyer,R.O. (1978) *The Effect of Disturbance on Plant Succession. Proceedings of the Ecological Society of Australia*, **10**, 135-145.

Noble,I.R. & Slatyer,R.O. (1980) *The use of vital attributes to predict successional changes in plant communities subject to recurrent disturbances. Vegetatio*, **43**, 5-21.

Odum,E.P. (1969) *The strategy of ecosystem development. Science*, **164**, 262-270.

Olson,R. & Sequeira,R.A. (1995) *Emergent computation and the modelling and management of ecological systems. Computers and Electronics in Agriculture*, **12**, 183-209.

Page,E.H. (1994) *Simulation Modeling Methodology: Principles and Etiology of Decision Support*. PhD Thesis, Virginia Polytechnic Institute and State University.

Pakeman,R.J., Hill,M.O. & Marrs,R.H. (1995) *Modelling Vegetation Succession After Bracken Control. Journal of Environmental Management*, **43**, 29-39.

Pausas,J. (1999) *Response of plant functional types to changes in the fire regime in Mediterranean ecosystems: A simulation approach. Journal of Vegetation Science*, **10**, 717-722.

Peet,R.K. (1992) Community structure and ecosystem function. *IN:* Glenn-Lewin, D. C., Peet, R. K., and Veblen, T. T. (1992), *Plant Succession: Theory and Prediction*, 1, Chapman and Hall, London.

Peet,R.K. & Christensen,N.L. (1980) *Succession: A Population Process. Vegetatio*, **43**, 131-140.

Pickett,S.T.A., Collins,S.L. & Armesto,J.J. (1987) *A hierarchical consideration of the causes and mechanisms of succession. Vegetatio*, **69**, 109-114.

Pickett,S.T.A. & Kolasa,J. (1989) *Structure of theory in vegetation science. Vegetatio*, **83**, 7-15.

Pickett,S.T.A., Kolasa,J. & Jones,C.G. (1994) *Ecological Understanding, The Nature of Theory and The Theory of Nature*. Academic Press, San Diego.

Plant,R.E. & Loomis,R.S. (1991) *Model-based Reasoning for Agricultural Expert Systems. AI Applications*, **5**, 17-28.

Quezel,P. (1981) The Study of Plant Groupings in the Countries Surrounding the Mediterranean: Some Methodological Aspects. *IN:* di Castri, F., Goodall, D. W., and Specht, R. L. (1981), *Mediterranean-Type Shrublands, Ecosystems of the World 11*, Elsevier, Amsterdam.

Quezel,P. (1981a) Floristic Composition and Phytosociological Structure of Sclerophyllous Matorral around the Mediterranean. *IN:* di Castri, F., Goodall, D. W., and Specht, R. L. (1981), *Mediterranean-Type Shrublands; Ecosystems of the World 11*, Elsevier, Amsterdam.

Rajagopalan,R. (1986) Qualitative modelling and simulation: A survey.*IN:* Kerckhoffs, E. J. H., Vansteenkiste, G. C., and Zeigler, B. P. (1986), *AI Applied to Simulation, Simulation Series Volume 18 Number 1*, First, The Society for Computer Simulation, San Diego, CA.

Ricklefs,R.E. (1990) *Ecology*. Freeman, New York.

Robertson,D., Bundy,A., Muetzelfeldt,R.I., Haggith,M. & Uschold,M. (1991) *Eco-Logic: Logic-Based Approaches to Ecological Modelling*. The MIT Press, London, England.

Robertson,D., Haggith,M., Kendon,G., Agusti,J. & Goldsborough,D. (1995) *Application of Logic Programming to Decision Support Systems in Ecology. AI Applications*, **9**, 23-38.

Ryan,M.G., Hunt,E.R., McMurtrie,R.E., Agren,G.I., Aber,J.D., Friend,A.D., Rastetter,E.B., Pulliam,W.M., Raison,R.J. & Linder,S. (1996) Comparing Models of Ecosystem Function for Temperate Confier Forests. I. Model Description and Validation. *IN:* Breymeyer, A. I., Hall, D. O., Melillo, J. M., and Agren, G. I. (1996), *Global Change Effects on Coniferous Forests and Grasslands*, 1, John Wiley and Sons,

Rykiel,E.J.Jr. (1989) *Artificial Intelligence and Expert Systems in Ecology and Natural Resource Management. Ecological Modelling*, **46**, 3-8.

Salles,P.S.B.A. (1997) *Qualitative Models in Ecology and their Use in Learning Environments*. PhD thesis, The University of Edinburgh.

Salles, P. S. B. A. and Bredeweg, B. (1997), *Building Qualitative Models in Ecology, IN:* Ironi, L. (1997), Proceedings of the 11th International Workshop on Qualitative Reasoning (QR'97), Publicazioni N.1036, Istituto di Analisi Numerica CNR, Pavia, Italy.

Salles, P. S. B. A., Muetzelfeldt, R. I., and Pain, H. (1996), *Qualitative Models in Ecology and their Use in Intelligent Tutoring Systems, IN:* Iwasaki, Y. and Farquhar, A. (1996), Proceedings of the 10th International Workshop on Qualitative Reasoning (QR'96), AAAI Technical Report WS-96-01, AAAI,

Schut,C. & Bredeweg,B. (1996) *An overview of approaches to qualitative model construction. The Knowledge Engineering Review*, **11**, 1-25.

Shannon,R.E., Mayer,R. & Adelsberger,H.H. (1985) *Expert Systems and Simulation. Simulation*, **44**, 275-284.

Shortliffe,E.H. (1976) *Computer-Based Medical Consultations: MYCIN*. American Elsevier / North-Holland, New York.

Silvertown,J.W. & Lovett Doust,J. (1993) *Introduction to Plant Population Biology*. Blackwell Scientific Publications Ltd, Oxford.

Smith,T. & Huston,M. (1989) *A theory of the spatial and temporal dynamics of plant communities. Vegetatio*, **83**, 49-69.

Starfield,A.M., Farm,B.P. & Taylor,R.H. (1989) *A rule-based ecological model for the management of an estuarine lake. Ecological Modelling*, **46**, 107-119.

Sterling,L. & Shapiro,E. (1994) *The Art of Prolog*. The MIT Press, Cambridge, MA.

Stillings,A., Feinstein,M.H., Garfield,J.L., Rissland,E.L., Rosenbaum,D.A., Weisler,S.E. & Baker-Ward,L. (1987) Artificial Intelligence: Knowledge Representation. *IN: Cognitive Science: An Introduction*, MIT Press, London.

Struss,P. (1988) Problems of Interval-Based Qualitative Reasoning - A Preliminary Report. *IN:* Fruectenicht, H. W. (1988), *Technische Expertensysteme: Wissenrepräsentation und Schlussfolgerungsverfahren*, R. Oldenbourg Verlag, Munich.

Tomaselli,R. (1981a) Main Physiognomic Types and Geographic Distribution of Shrub Systems Related to Mediterranean Climates. *IN:* di Castri, F., Goodall, D. W., and Specht, R. L. (1981), *Mediterranean-Type Shrublands. Ecosystems of the World 11*, Elsevier, Amsterdam.

Tomaselli,R. (1981b) Relations with other Ecosystems: Temperate Evergreen Forests, Mediterranean Coniferous Forests, Savannahs, Steepes and Desert Shrublands. *IN:* di Castri, F., Goodall, D. W., and Specht, R. L. (1981), *Ecosystems of the World III, Mediterranean-Type Shrublands*, 1, Elsevier, Oxford.

Trabaud,L. (1991) *Fire regimes and phytomass growth dynamics in a Quercus coccifera garrigue. Journal of Vegetation Science*, **2**, 307-314.

Trabaud,L. (1994) Postfire Plant Community Dynamics in the Mediterranean Basin. *IN:* Moreno, J. M. and Oechel, W. C. (1994) *The Role of Fire in Mediterranean-Type Ecosystems*, 1, Springer-Verlag, New York.

Uhrmacher,A.M., Cellier,F.E. & Frye,R.J. (1997) *Applying Fuzzy-based Inductive Reasoning to Analyse Qualitatively the Dynamic Behaviour of an Ecological System. AI Applications*, **11**, 1-10.

Usher,M.B. (1979) *Markovian Approaches to Ecological Succession. Journal of Animal Ecology*, **48**, 413-426.

Usher,M.B. (1981) *Modelling ecological succession, with particular reference to Markovian models. Vegetatio*, **46**, 11-18.

Usher,M.B. (1992) Statistical models of succession. *IN:* Glenn-Lewin, D. C., Peet, R. K., and Veblen, T. T. (1992), *Plant Succession, Theory and Prediction*, 1, Chapman and Hall, London.

van der Valk,A.G. (1992) Establishment, colonization and persistence. *IN:* Glenn-Lewin, D. C., Peet, R. K., and Veblen, T. T. (1992), *Plant Succession: Theory and Prediction*, 1, Chapman and Hall , London.

Walters,C. (1986) *Adaptive Management of Renewable Resources.* Fisheries Centre, University of British Columbia, Vancouver.

Weiher,E. & Keddy,P.A. (1995) *Assembly rules, null models, and trait dispersion: new questions from old patterns. Oikos*, **74**, 159-164.

Weiher,E., van der Werf,A. , Thompson,K., Roderick,M., Garnier,E. & Eriksson,O. (1999) *Challenging Theophrastus: A common core list of plant traits for functional ecology. Journal of Vegetation Science*, **10**, 609-620.

Westman,W.E. (1986) Resilience: concepts and measures. *IN:* Dell, B., Hopkins, A. J. M., and Lamont, B. B. (1986), *Resilience in Mediterranean-Type Ecosystems*, 1st, Dr W Junk, The Hague, Netherlands.

White,P.S. (1979) *Pattern, process and natural disturbance in vegetation. Botanical Review*, **45**, 229-299.

Wiens,J.A. (1989) *Spatial Scaling in Ecology. Functional Ecology*, **3**, 385-397.

Williams,B.C. (1991) *A theory of interactions: unifying qualitative and quantitative algebraic reasoning. Artificial Intelligence*, **51**, 39-94.

Williams,B.C. & de Kleer,J. (1991) *Qualitative Reasoning about physical systems: a return to roots. Artificial Intelligence*, **51**, 1-9.

Windon Jr.,D.R. & Massey,J.G. (1991) *Translating Knowledge from Expert to Expert System. AI Applications*, **5**, 33-39.

Appendix 1 – SIMAO Qualitative Algebra Modelling Evaluation

QS(x)def*n.m* Mapping Definitions

The definitions QS(*x*)def*n.m* for all numerically measurement quantities *x* are presented below.

QS(x)def1.1
$N_i = \{(0 < pp \leq 14), (14 < p \leq 28), (28 < m \leq 42), (42 < f \leq 56), (56 < ff \leq 70)\}$.
$r_i = \{(0 < pp \leq 0.2), (0.2 < p \leq 0.4), (0.4 < m \leq 0.6), (0.6 < f \leq 0.8), (0.8 < ff \leq 1.0)\}$.
$\propto_{j\,i} = QS(r_i)$.
$K_i = \{(0 < pp \leq 20), (20 < p \leq 40), (40 < m \leq 60), (60 < f \leq 80), (80 < ff \leq 100)\}$.
$N_j = QS(N_i)$.
$r_j = QS(r_i)$.
$\propto_{i\,j} = QS(r_i)$.
$K_j = QS(K_i)$.

QS(x)def1.2
$N_i = \{(0 < pp \leq 16), (16 < p \leq 32), (32 < m \leq 48), (48 < f \leq 64), (64 < ff \leq 80)\}$.
$r_i = \{(0 < pp \leq 0.2), (0.2 < p \leq 0.4), (0.4 < m \leq 0.6), (0.6 < f \leq 0.8), (0.8 < ff \leq 1.0)\}$.
$\propto_{j\,i} = QS(r_i)$.
$K_i = \{(0 < pp \leq 20), (20 < p \leq 40), (40 < m \leq 60), (60 < f \leq 80), (80 < ff \leq 100)\}$.
$N_j = QS(N_i)$.
$r_j = QS(r_i)$.
$\propto_{i\,j} = QS(r_i)$.
$K_j = QS(K_i)$.

QS(x)def2.1
$N_i = \{(0 < pp \leq 8), (8 < p \leq 24), (24 < m \leq 56), (56 < f \leq 72), (72 < ff \leq 80)\}$.
$r_i = \{(0 < pp \leq 0.2), (0.2 < p \leq 0.4), (0.4 < m \leq 0.6), (0.6 < f \leq 0.8), (0.8 < ff \leq 1.0)\}$.
$\propto_{j\,i} = QS(r_i)$.
$K_i = \{(0 < pp \leq 10), (10 < p \leq 30), (30 < m \leq 70), (70 < f \leq 90), (90 < ff \leq 100)\}$.
$N_j = QS(N_i)$.
$r_j = QS(r_i)$.
$\propto_{i\,j} = QS(r_i)$.
$K_j = QS(K_i)$.

QS(x)def2.2
$N_i = \{(0 < pp \leq 9), (9 < p \leq 27), (27 < m \leq 63), (63 < f \leq 81), (81 < ff \leq 90)\}$.
$r_i = \{(0 < pp \leq 0.2), (0.2 < p \leq 0.4), (0.4 < m \leq 0.6), (0.6 < f \leq 0.8), (0.8 < ff \leq 1.0)\}$.
$\propto_{j\,i} = QS(r_i)$.
$K_i = \{(0 < pp \leq 10), (10 < p \leq 30), (30 < m \leq 70), (70 < f \leq 90), (90 < ff \leq 100)\}$.
$N_j = QS(N_i)$.
$r_j = QS(r_i)$.
$\propto_{i\,j} = QS(r_i)$.
$K_j = QS(K_i)$.

SIMAO Operator Test Results

The SIMAO [+] operator

Associativity:

Equation 1, operation order 1:
- (10 + 50) + 30 = 60 + 30 = 90 ≡ ff
- (pp [+] m) [+] p = m [+] p = m

Equation 2, operation order 2:
- 10 + (50 + 30) = 10 + 80 = 90 ≡ ff
- pp [+] (m [+] p) = pp [+] m = m

Commutativity:

Equation 1, term order 1:
- 30 + 50 = 80 ≡ f
- p [+] m = m

Equation 1, term order 2:
- 50 + 30 = 80 ≡ f
- m [+] p = m

Equation 2, term order 1:
- 30 + 50 + 30 = 110 ≡ ff
- p [+] m [+] p = m

Equation 2, term order 2:
- 30 + 30 + 50 = 110 ≡ ff
- p [+] p [+] m = m

Equation 2, term order 3:
- 50 + 30 + 30 = 110 ≡ ff
- m [+] p [+] p = m

The SIMAO [-] operator

Associativity:

Equation 1, operation order 1:
- (50 – 10) – 30 = 40 – 30 = 10 ≡ pp
- (m [-] pp) [-] p = m [-] p = m

Equation 1, operation order 2:
- 50 – (10 – 30) = 50 – (- 20) = 70 ≡ f
- m [-] (pp [-] p) = m [-] pp = m

Equation 2, operation order 1:
- (30 – 50) – 10 = - 20 – 10 = - 30 ≡ pp
- (p [-] m) [-] pp = pp [-] pp = pp

Equation 2, operation order 2:
- 30 – (50 – 10) = 30 –40 = - 10 ≡ pp
- p [-] (m [-] pp) = p [-] m = pp

Equation 3, operation order 1:

- (90 – 30) – 10 = 60 – 10 = 50 ≡ m
- (ff [-] p) [-] pp = ff [-] pp = ff

Equation 3, operation order 2:

- 90 – (30 – 10) = 90 – 20 = 70 ≡ f
- ff [-] (p [-] pp) = ff [-] p = ff

Commutativity:

Equation 1, term order 1:

- 30 – 50 = - 20 ≡ pp
- p [-] m = pp

Equation 1, term order 2:

- 50 – 30 = 20 ≡ pp
- m [-] p = m

The SIMAO [*] operator

Associativity:

Equation 1, operation order 1:

- (10 x 30) x 70 = 21,000 ≡ ff
- (pp [*] p) [*] f = pp [*] f = p

Equation 1, operation order 2:

- 10 x (30 x 70) = 21,000 ≡ ff
- pp [*] (p [*] f) = pp [*] m = pp

Commutativity:

Equation 1, term order 1:

- 10 x 30 = 300 ≡ ff
- pp [*] p = pp

Equation 1, term order 2:

- 30 x 10 = 300 ≡ ff
- p [*] pp = pp

Equation 2, term order 1:

- 10 x 30 x 70 = 21,000 ≡ ff
- pp [*] p [*] f = p

Equation 2, term order 2:

- 70 x 10 x 30 = 21,000 ≡ ff
- f [*] pp [*] p = pp

Equation 2, term order 3:

- 30 x 70 x 10 = 21,000 ≡ ff
- p [*] f [*] pp = pp

Distributivity over addition:

Equation 1, operation order 1:

- 10 x (30 + 50) = 10 x 80 = 800 ≡ ff
- pp [*] (p [+] m) = pp [*] m = pp

Equation 1, operation order 2:

- (10 x 30) + (10 x 50) = 300 + 500 = 800 ≡ ff
- (pp [*] p) [+] (p [*] f) = pp [+] m = m

Equation 2, operation order 1:

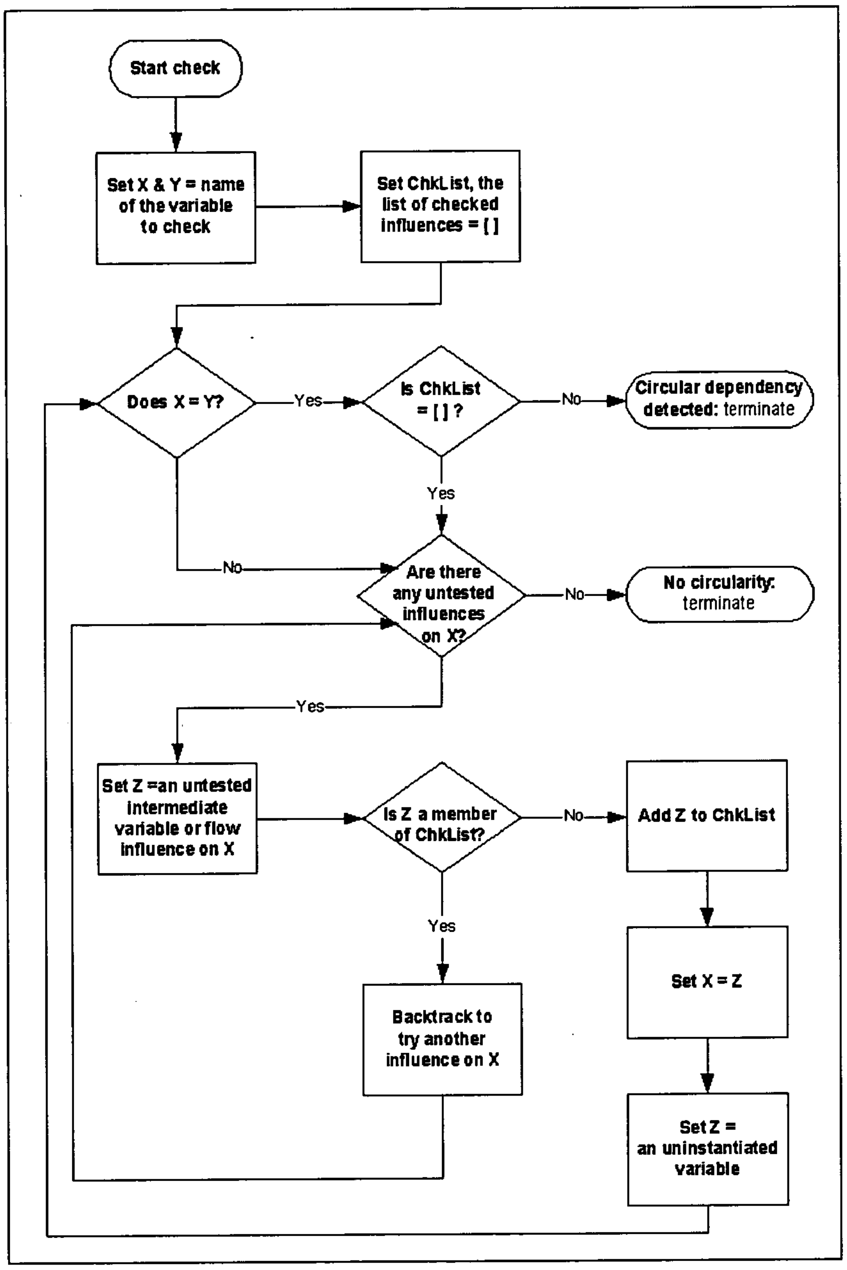
- 10 x (30 + 80) = 10 x 110 = 1100 ≡ ff
- pp [*] (p [+] f) = pp [*] f = p

Equation 2, operation order 2:

- (10 x 30) + (10 x 80) = 300 + 800 = 1100 ≡ ff
- (pp [*] p) [+] (pp [*] f) = pp [+] m = m

Appendix 2 – RBCLM3 Algorithms

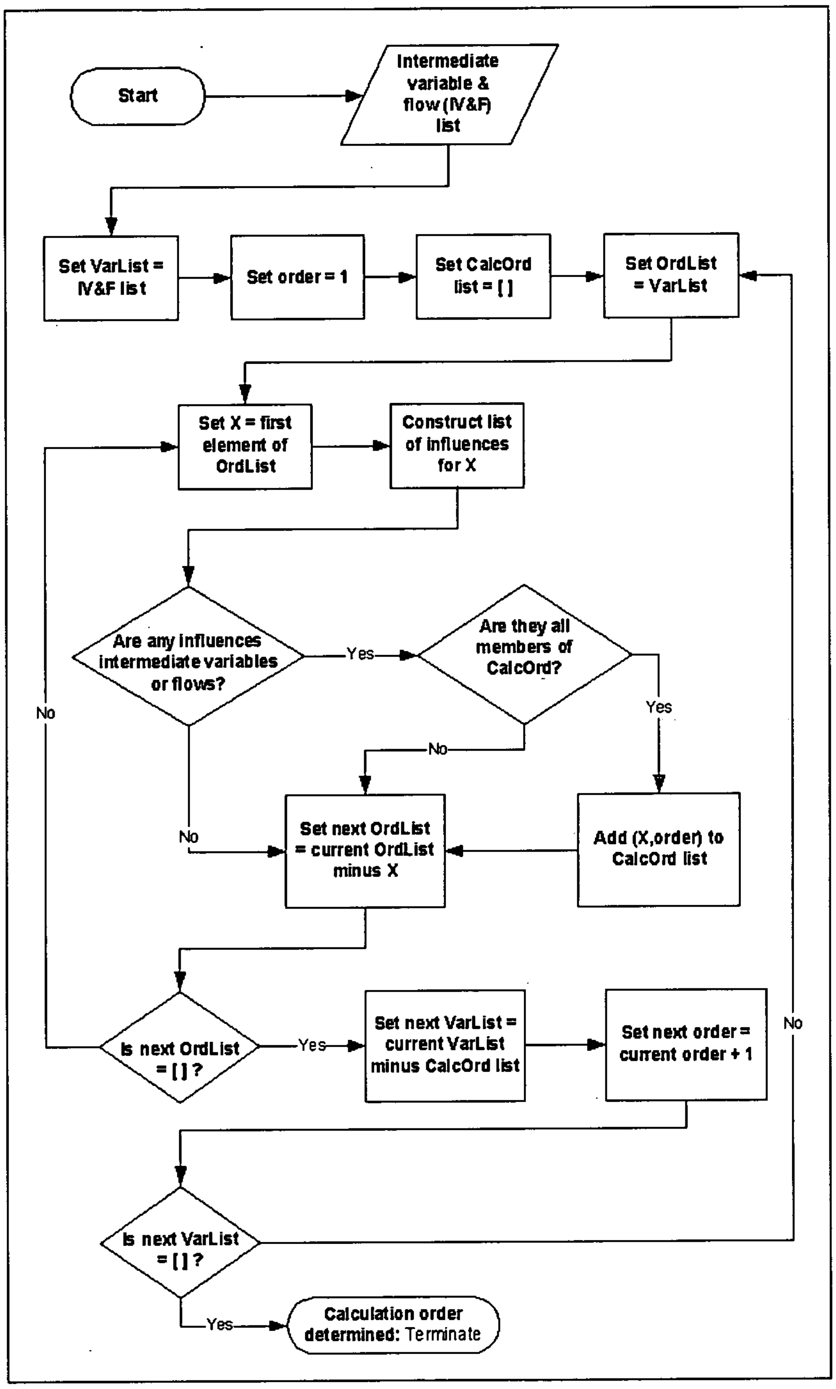
Circular dependency checking algorithm



Start check
Set X & Y = name of the variable to check
Set ChkList, the list of checked influences = [ ]
Does X = Y?
Yes
Is ChkList = [ ] ?
No
Circular dependency detected: terminate
Yes
No
Are there any untested influences on X?
No
No circularity: terminate
Yes
Set Z = an untested intermediate variable or flow influence on X
Is Z a member of ChkList?
No
Add Z to ChkList
Yes
Set X = Z
Backtrack to try another influence on X
Set Z = an uninstantiated variable

The circularity checking algorithm works by trying to find a route from a variable X to the same variable X. The algorithm is recursive and utilises the backtracking facilities provided by the SICStus Prolog interpreter. The algorithm essentially tries each influence on X in turn trying to trace the influence back to X. All possible influence paths onto X are tested to see if any lead back to X without going through state variables. To ensure that the path tracing does not get caught in an infinite loop, a list of previously visited influences is kept so that they are not tried twice (the list called ChkList).

Calculation order determination algorithm



Start
Intermediate variable & flow (IV&F) list
Set VarList = IV&F list
Set order = 1
Set CalcOrd list = [ ]
Set OrdList = VarList
Set X = first element of OrdList
Construct list of influences for X
Are any influences intermediate variables or flows?
Yes
Are they all members of CalcOrd?
Yes
No
No
Set next OrdList = current OrdList minus X
Add (X,order) to CalcOrd list
No
Is next OrdList = [ ] ?
Yes
Set next VarList = current VarList minus CalcOrd list
Set next order = current order + 1
No
Is next VarList = [ ] ?
Yes
Calculation order determined: Terminate

Basically the algorithm recurses round the list of intermediate variables and flows for a model one element at a time. The calculation order is set to 1 for the first recursion (i.e. to search for order 1 variables). For each element a list of influences is constructed based on the `infl` predicate used in the Model Specification component of the KB. If any of these influences are intermediate variables or flows that have not already been checked and assigned a calculation order then it means that the variable concerned is dependent on a variable whose value must be calculated beforehand. In this case the next element of the list is checked. If however it is found that either none of the variable's influences are intermediate variables or flows or, if they are, that they have already been assigned a calculation order, then a tuple is added to the calculation order list containing the variable name and its calculation order (in the first instance this is equal to 1). The calculation order list form:

[(VarName,Order),(VarName,Order) ...]

where,
Order = the calculation order for the variable, VarName.

Once the full list of intermediate variables and flows have been processed in this way the variables that have been put into the calculation order list (i.e. given a calculation order) are removed from the whole intermediate variable and flow list. The reduced list is then put through the same process for calculation order 2. As variables are found that are only dependent on variables with known values (i.e. those already assigned a calculation order) they are added to the calculation order list. When the recursion is complete for a second time, those variables that have been determined to be calculation order 2 are removed from the variable list and the process begins again for order 3. This is repeated until all intermediate variable and flow orders have been determined. The number of orders may vary from 1 to *n* depending on model complexity.