FUZZY QUALITATIVE SIMULATION AND DIAGNOSIS OF CONTINUOUS DYNAMIC SYSTEMS

QIANG SHEN, B.Sc., M.Sc.

Thesis submitted for the degree of

Doctor of Philosophy

at

Heriot-Watt University,

Department of Electrical and Electronic Engineering,

Edinburgh, United Kingdom

September, 1991

This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognise that the copyright rests with its author and that no quotation from the thesis and no information derived from it may be published without the prior written consent of the author or the University (as may be appropriate).

Table of Contents

Table of Contents i
List of Figures v
Acknowledgements vii
Abstract viii
Abbreviations and Symbols ix

CHAPTER 1: INTRODUCTION 1
1.1 Preamble 1
1.2 Overview 8

CHAPTER 2: THE DEVELOPMENT OF QUALITATIVE REASONING ABOUT PHYSICAL SYSTEMS 13
2.1 Introduction 13
2.2 Three Classical Approaches to Qualitative Reasoning 14
2.3 Recent Approaches to Qualitative Reasoning 19
2.3.1 Non-Standard Analysis Based Approaches 21
2.3.2 Other Related and Important Approaches 27
2.4 Summary 30

CHAPTER 3: MOTIVATIONS FOR USING FUZZY SETS AND A REVIEW OF THE RELEVANT THEORY 32
3.1 Introduction 32
3.2 Motivations for the Use of Fuzzy Sets 33
3.3 A Review of the Relevant Aspects of the Theory 37
3.4 Summary 41

CHAPTER 4: FUZZY QUANTITY SPACE AND MODELLING PRIMITIVES 43
4.1 Introduction 43
4.2 Fuzzy Quantity Space 44
4.2.1 General Description 44
4.2.1.1 Desirable Properties and Basic Definition 44
4.2.1.2 Comparison with Crisp Quantity Spaces 46
4.2.2 Four-Tuple Parametric Representation 49
4.3 Fuzzy Modelling Primitives 51
4.3.1 Algebraic Constraints 52

4.3.1.1 Operations 52
4.3.1.2 Examples 56
4.3.2 Derivative Constraints 61
4.3.3 Function Relational Constraints 62
4.3.3.1 Operations 63
4.3.3.2 Modelling Example 65
4.4 Summary 69

CHAPTER 5: FUZZY QUALITATIVE SIMULATION
OF CONTINUOUS DYNAMIC SYSTEMS 71
5.1 Introduction 71
5.2 State Description 74
5.2.1 Representation of a Fuzzy Qualitative State 74
5.2.2 Computational Method for Persistence Time 76
5.3 State Transitions 79
5.3.1 State Transition Rules 81
5.3.2 Arrival Times for State Transitions 83
5.3.2.1 Theoretical Preparations 84
5.3.2.2 Computational Method for Arrival Time 87
5.4 Filtering Techniques 90
5.4.1 Constraint Filtering 90
5.4.1.1 Constraint Filtering Criteria 91
5.4.1.2 Using the Waltz Algorithm for Constraint Filtering 92
5.4.2 Temporal Filtering 93
5.4.2.1 Absolute Time Indices 94
5.4.2.2 Temporal Filtering Criterion 95
5.4.3 Global Filtering 97
5.5 FuSim Algorithm Structure 100
5.6 Examples 101
5.6.1 Example 1: Two Coupled Tanks 103
5.6.2 Example 2: A Mass on a Spring 111
5.7 Summary 122

CHAPTER 6: BACKGROUND AND ARCHITECTURE
OF MODEL-BASED DIAGNOSTIC SYSTEMS 124
6.1 Introduction 124
6.2 Motivations for Constructing Model-Based Diagnostic Systems 125
6.3 Static and Dynamic Models 127
6.4 Model Based Diagnosis of Static Systems 130
6.4.1 Normal-Behaviour Model Based Approaches 130

6.4.2 Normal and Fault Models Based Approaches 136
6.5 Model Based Diagnosis of Dynamic Systems 138
6.6 Architecture of Model Based Diagnostic Systems 143
6.6.1 The Architecture 144
6.6.1.1 Predictor 145
6.6.1.2 Discrepancy Generator 146
6.6.1.3 Candidate Proposer 147
6.6.1.4 Diagnostic Strategist 148
6.6.2 Instantiations of the Architecture 151
6.6.2.1 GDE+ 151
6.6.2.2 Mimic 153
6.7 Summary 155

CHAPTER 7: SYNCHRONOUS ITERATIVE DIAGNOSIS OF CONTINUOUS DYNAMIC SYSTEMS 157
7.1 Introduction 157
7.2 Synchronous Tracking 158
7.3 Iterative Search 161
7.3.1 Basic Method and Properties of Iterative Search 162
7.3.2 Modelling Dimensions 167
7.4 Discrepancy Measurement 170
7.5 The Synchronous Iterative Diagnostic System: SID 173
7.5.1 Predictor 175
7.5.1.1 Data Interpreter 175
7.5.1.2 Behaviour Simulator 176
7.5.2 Discrepancy Generator 176
7.5.3 Candidate Proposer 180
7.5.3.1 Candidate Generator 181
7.5.3.2 Candidate Ranker 185
7.5.4 Diagnostic Strategist 186
7.5.4.1 Performance Evaluator 186
7.5.4.2 Supervisor 187
7.6 Example 189
7.6.1 Review of Normal-Behaviour Model and the Behaviour 189
7.6.2 Diagnosing Faults against Relation Strength Dimension 190
7.7 Summary 194

CHAPTER 8: CONCLUSION 195
8.1 Summary of the Thesis 195
8.2 Future Work 199
8.2.1 Suggestions for Work on FuSim 200

8.2.2 Suggestions for Work on SID 203

APPENDIX A: LIST OF PUBLICATIONS 206

APPENDIX B: SOFTWARE ARCHITECTURAL DESIGN OF FUSIM 208

APPENDIX C: SOFTWARE ARCHITECTURAL DESIGN OF SID 211

REFERENCES 216

List of Figures

Figure 2.1 Comparison between the *QSim* and *HR-QSIM*'s Quantity Spaces
Figure 2.2 Strict and Heuristic Interpretations of *O[M]* Relations
Figure 3.1 Approximately Four
Figure 3.2 A Normal Convex Fuzzy Number
Figure 3.3 The α-Cuts of 'Approximately Four'
Figure 4.1 Comparison between Fuzzy and Crisp Quantity Spaces
Figure 4.2 Parametric Representation of a Normal Convex Fuzzy Number
Figure 4.3 A Fuzzy Qualitative Space
Figure 4.4 The Utilisation of the Approximation Principle
Figure 4.5 A Fuzzy Qualitative Space
Figure 4.6 Two Coupled Tanks
Figure 4.7 Two Characteristic Curves of the Orifice
Figure 4.8 Two Subsets of Fuzzy Qualitative Values
Figure 5.1 Input and Output of the FuSim Algorithm
Figure 5.2 A Functional Overview of the FuSim Algorithm
Figure 5.3 The Persistence Time of an Interval-Valued State
Figure 5.4 The Width of a Fuzzy Qualitative Value
Figure 5.5 A Fuzzy Quantity Space
Figure 5.6 Four Kinds of Transitions
Figure 5.7 The Crossing-Point of Two Fuzzy Numbers
Figure 5.8 Three Types of α Distance
Figure 5.9 The Arrival Time for an M-Transition
Figure 5.10 Interface Window of the Fuzzy Simulation of Two Coupled Tanks
Figure 5.11 Fuzzy Quantity Space Used in the Coupled-Tanks Example
Figure 5.12 A Step-ahead Simulation of Two Coupled Tanks
Figure 5.13 A Mass on A Spring
Figure 5.14 A Cycle of the Behaviour of 'a Mass on a Spring' System
Figure 5.15 Generation and Filtering of the First 1/4 Period of Behaviour
Figure 5.16 Behaviours of Modified Models
Figure 6.1 'Open Loop' Structure of Dependency-Recording Based Systems
Figure 6.2 The Hypothesise-and-Match Cycle of Mimic
Figure 6.3 A Primitive Architecture for MBDS
Figure 7.1 Tracking a Model
Figure 7.2 Candidate Generation by Iterative Search

Figure 7.3 A Classification of Acceptable and Unacceptable Behaviours
Figure 7.4 Structure of the SID System
Figure 7.5 Discrepancy Generation
Figure 7.6 False Positive and False Negative Diagnoses
Figure 7.7 Monitoring and Diagnosis

Acknowledgements

I should like to express my thanks to my supervisor, Professor R. R. Leitch, for his advice and encouragement throughout the course of my research.

Much of the work reported in this thesis has been financially supported by the State Committee of Education of China and the British Council by whom the author was awarded the Sino-British Friendship Scholarship when the corresponding award was first established. A small part of this research has been undertaken within ESPRIT II Project P5143, named ARTIST and funded by the Commission of the European Communities.

It is with much love that I thank my father, Buchao Shen, and my mother, Sumin Zhou, for their encouragement and support throughout my education. I wish to acknowledge an incalculable debt of time to my beloved wife, Changjing Shang, and my son, Yuan Shen. The immense help and patience of Changjing has made the writing of this thesis possible.

Finally, I should like to thank my colleagues in the Knowledge Based Systems Group of the Intelligent Automation Laboratory for providing such a nutritious research environment.

Abstract

The theory of Fuzzy Sets and the development of Qualitative Reasoning have had similar motivations: coping with complexity in reasoning about the behaviour of physical systems. The first part of this thesis presents a synthesis of these techniques to provide a fuzzy qualitative simulation algorithm, named *FuSim*, that offers significant advantages over existing qualitative simulation methods. It allows a more detailed description of system variables, through an arbitrary, but finite, discretisation of the quantity space which, in turn, allows a more detailed description of functional relationships in that both strength and sign information can be represented. It also enables ordering information on rates of change to be used to compute temporal information on the state and the possible state transitions. These result in a considerable reduction in the number of spurious behaviours generated during the simulation and in a mechanism for producing temporal durations associated with each qualitative state. Both of these properties are crucial to practical utilisation of qualitative simulation.

In the second part of the thesis, a synchronous iterative diagnostic framework for continuous dynamic systems, called *SID*, is described that utilises *FuSim* to model and simulate the physical system to be diagnosed. In particular, techniques for the synchronous tracking of the evolution of the system between equilibria are established. A discrepancy metric that allows for the continuous degradation from normal to faulty behaviour to be monitored is given. Furthermore, a method for identifying faults through iterative search against modelling dimensions is proposed. This provides explicit feedback from detected discrepancies to model adjustments, thereby reducing the sensitivity to modelling errors and allowing approximate fault models to be used.

Abbreviations and Symbols

AI	Artificial Intelligence
ARTIST	Advanced Reasoning Tool for Model-Based Diagnosis of Industrial Systems
ATMS	Assumption-Based Truth Maintenance System
FuSim	Fuzzy Qualitative Simulation
MBDDS	Model-Based Diagnosis of Dynamic Systems
MBDS	Model-Based Diagnostic Systems
NSA	Non-Standard Analysis
OMR	Order-of-Magnitude Reasoning
QR	Qualitative Reasoning about Physical Systems
SID	Synchronous Iterative Diagnosis
OBS (.)	The observation from physical systems
PRED (.)	The prediction from models
Sup	The least upper bound operator
max, min	The maximum and minimum operators
t	The absolute time index
Φ	The empty set
ΔT_a	The arrival time
ΔT_p	The persistence time
$(\Delta(.,.))_\alpha$	The α distance metric
$\|.\|$	The absolute value operator
$<_\alpha$, $>_\alpha$	The α ordering relational signs
$\in$, $\bar{\in}$	The belonging and non-belonging signs
$\mu(.)$	The membership function
A_α	The α-cut of a fuzzy set A
$M^+(.,.)$	The monotonic increasing operator
$M^-(.,.)$	The monotonic decreasing operator
Q_F	The fuzzy quantity space
$W(.)$	The width operator
$d(.,.)$	Distance metric
$m(.)$	Mapping operator

CHAPTER 1

INTRODUCTION

1.1. Preamble

The ability to qualitatively reason about the behaviour of physical systems is important to understanding and interacting with the real world for both human beings and intelligent machines. Accordingly, studies of methods for performing such reasoning have become an important subject of research in the Artificial Intelligence and Cognitive Science communities. The essential goal of Qualitative Reasoning about the Behaviour of Physical Systems, often simply termed Qualitative Reasoning and sometimes indistinctly referred to as Qualitative Modelling or Qualitative Physics, is to capture both the common-sense of the person on the street and the tacit knowledge underlying the quantitative knowledge used by engineers and scientists [37, 117].

The key to Qualitative Reasoning is to find ways to represent continuous properties of physical systems by sets of finitely discretised symbols. A system state in a qualitative description typically corresponds to many states in a traditional (numerical) representation, allowing more abstract descriptions of state such that each of the numerically represented states is distinguished by having the same 'meaningful behaviour pattern' occurring in them. To perform reasoning about the corresponding qualitative behaviours based on such abstracted descriptions of physical quantities appropriate inference mechanisms are required to cope with partial or incomplete information. This forms the central issue of Qualitative Reasoning.

The last decade has seen significant progress towards the development of formal methods for qualitative reasoning about the behaviour of physical systems.

Approaches to Qualitative Reasoning result in a set of eclectic techniques for generating a qualitative description of the system behaviour from a description of system structure and certain initial operating conditions. Such reasoners offer the prospect of coping with the inherent complexity found in today's technological systems and of providing the basis for automated solutions to an extended range of application problems. In particular, Qualitative Reasoning forms the heart of Model-Based Reasoning techniques for automated diagnosis, control, design, and industrial training systems. By the use of knowledge of system structure and behaviour, the model-based approach overcomes the commonly recognised problem of rule-based expert systems that are restricted to empirical associations. However, a number of important technical problems remain to be solved in order to make qualitative techniques viable for full-scale industrial applications [66].

At present these limitations fall into two non-distinct classes:

1) Qualitative ambiguity;

2) Lack of temporal information.

The first occurs because of the inherent ambiguity of qualitative calculi [101, 107]. When performing the derivation of the qualitative behaviour of a physical system, i.e., the *qualitative simulation* of the system, this appears as the generation of numerous spurious behaviours which tend to obscure the real behaviour, making the results impracticable except for relatively small and (over) simplified problems. Reducing these spurious behaviours is currently the main occupation of this particular research area. Two approaches are being followed: the development of filtering techniques that utilise either system theoretic properties, e.g. the non-intersecting constraint [57, 102] or additional knowledge from some other source, e.g. experimental, and the development of more descriptive qualitative representations that allow more

detailed information on the system to be incorporated, thereby reducing the ambiguity at source. In principle, the ambiguity cannot be eliminated due to the inherent essence of qualitative representation, therefore, a combination of these methods will be required.

The second limitation occurs because current techniques have no explicit information on temporal durations but, at most, provide a logical ordering in the evolution of system state [95, 123]. Again, two approaches to this problem are being developed. The first attaches an explicit temporal duration to each qualitative state, with the temporal reasoning, i.e., inferences about time [1], being made independently of behavioural generation [123]. Constraint Propagation methods [14, 110] are used separately to propagate both the qualitative values and their durations. The second approach extends the representation of the qualitative state to allow ordering information on the relative rates of change of variables [15, 113, 116]. This allows a temporal duration within which a qualitative state remains, called the persistence time, to be calculated, although still representing time in a weak form.

This thesis presents a *Fuzzy Qualitative Simulation* technique that utilises the theory of *Fuzzy Sets* [126] to give an arbitrary, but finite, discretisation of the representation of system variables. Such a method allows both strength and sign information on the functional relationships holding against two or multi-variables to be represented, resulting in a considerable reduction of the inherent ambiguity of qualitative calculi. Also, this simulation system allows ordering information on the rates of change of variables to be modelled, producing a measure of how long the system remains within a particular (qualitative) state and/or when a state transition occurs. Thus, an ordering of the evolution of system states *and* the associated temporal durations are obtained. The temporal information so produced further helps to

significantly limit qualitative ambiguity and, more importantly, makes qualitative simulation much more suitable for use within application systems where information about time durations is crucial to effective automated reasoning.

The use of graded membership to represent impreciseness within Fuzzy Sets allows the subjective element of common-sense knowledge to be incorporated within a formal algorithm for generating behaviour from a structural model, thereby, combining theoretical knowledge [66], in the form of modelling constraints, and empirical knowledge, in the interpretation of system values. Furthermore, this allows a rigorous approach to capturing uncertainty and provides a unifying framework from which other recently developed techniques can be generated as specialisations [93, 97]. In so doing, the fuzzy qualitative simulation technique combines the previously disparate fields of Qualitative Reasoning and Fuzzy Sets, both of which were introduced to cope with the complexities found in modelling and reasoning about the behaviour of physical systems.

The development of the fuzzy qualitative simulation itself is significant in that it overcomes or reduces many difficulties existing in current approaches to Qualitative Reasoning. However, the real advantages of the fuzzy simulation appear when it is utilised in Model-Based Reasoning tasks such as the automation of diagnosis and control. For this purpose, the thesis further presents investigations into the area of Model-Based Diagnosis, resulting in a novel technique for diagnosing continuous dynamic systems by the use of fuzzy qualitative simulation.

To determine why a physical system has not worked correctly compared with its design intention, it is useful to know how it was supposed to work in the first place. It is this simple observation that underlies the considerable interest in the development of model-based diagnostic systems [18]. The basic paradigm of Model-Based Diag-

nosis can be best viewed as the interaction of the observations from a physical system and the predictions from a model of the system. Observations indicate what a physical system is actually doing whilst predictions indicate what the system is intended to do. Such an approach to diagnosis is, therefore, essentially dependent upon the use of an explicit structural model of the physical system to be diagnosed to predict the system behaviour. Whenever there exists a difference, or a discrepancy, between observations and predictions, it is presumed that the system is not working correctly and certain faults have occurred within the system. Such discrepancies provide crucial information to the characteristics of the possible faults. The ultimate task of Model-Based Diagnosis is then to identify which of the components of the system could have failed and led to all the discrepancies detected. Diagnostic techniques developed in this way provide a number of important advantages over the early rule-based diagnostic methods that, although might be effective and efficient for familiar situations, suffered from the problems of incompleteness and inflexibility.

Like Qualitative Reasoning, Model-Based Diagnosis has also tremendously gained in significance within the last decade. In fact, such techniques have been successfully applied to diagnosing discrete static systems [26, 82]. However, two essential limitations exist that prevent these techniques from being used to cope with failures occurring in continuous dynamic systems:

1) Requiring an 'exact' definition of system models (including fault models) and an 'accurate' distinction between normal condition and fault behaviour;

2) Representing physical systems by algebraic models.

Current model-based techniques [26, 27, 82, 109] isolate fault candidates by following a fixed dependency tree back from conflict predictions to each component that has contributed to those predictions. This can be very effective in some situations

where 'exact' system models are available and, also, when faulty behaviour is clearly defined by strictly indicating whether there is a fault in the system or not, as in the domain of digital circuits. Unfortunately, in most physical systems where variables change continuously and, therefore, modelling errors usually exist, clear definitions of fault behaviours are difficult to give, such a fixed-dependency based reasoning mechanism may suffer from high risks of returning wrong diagnoses or become intractable. In this sense, these techniques cannot be effectively applied to situations where the system behaviour continuously degrades from acceptable to unacceptable.

Disregarding this difficulty, present model-based diagnostic systems make a common basic diagnostic assumption that the system model of normal behaviour is described by a set of algebraic equations, thereby neglecting energy storage within the physical system. Hence, no transient behaviour is represented and, therefore, no early detection of incipient faults is possible. To remove this crucial limitation, dynamic models of physical systems are required and, further, techniques for synchronously tracking the evolving behaviour of the dynamic systems must be established. Two approaches are being investigated in an attempt to tackle this problem. One simply extends the existing diagnostic systems based on static models to time-dependent observations, but is still fundamentally limited to systems in equilibrium [44]. Also, such modifications are unable to handle continuous degradation of fault behaviours. The other approach models systems by (qualitative) differential equations, thereby representing the dynamic characteristics of the system [33]. This is an important breakthrough in developing model-based diagnostic systems using dynamic models. Nevertheless, existing methods rely upon *a priori* knowledge of faults and do not present a principled way for 'tracking' model predictions against observations over time [94, 95].

This thesis adopts the second approach indicated above and provides significant developments that make model-based diagnostic systems for continuous dynamic systems practicable. To formalise the discussion, a generic architecture for Model-Based Diagnosis is provided that incorporates current approaches to diagnosing, including monitoring, both static and dynamic systems. A particular instantiation of this architecture is then proposed that supports a reliable diagnosis of faults by synchronously tracking the evolution of actual behaviour of the system being diagnosed. Techniques for monitoring and detecting the continuous distortion of the model prediction from the actual behaviour are also presented. Most importantly, departing from existing approaches, a radical reasoning mechanism is established that utilises quantified 'feedback' from discrepancies generated to the fault candidates returned by the diagnostic process. Diagnosis is performed by means of an iterative search within the space of possible model variations to identify those fault models that can minimise the discrepancies, thereby reducing the sensitivity of the diagnostic process to modelling errors and removing the need for fault models being supplied *a priori*. However, if fault knowledge is available, it can be exploited to focus and, therefore, speed up the search. Throughout the development of this diagnostic system, the employment of the fuzzy qualitative simulation offers crucial efficiency and effectiveness in making the task feasible.

Substantial parts of the work presented in this thesis have been published in the academic literature (see Appendix A).

1.2. Overview

The thesis is structurally organised as follows:

Chapter 2 This chapter provides an overview of the development of Qualitative Reasoning. A brief review of three pioneering approaches is first presented, showing the basic motivations and design choices for establishing such reasoners. Recent improvements and extensions are then discussed. In particular, influential techniques based on the theory of non-standard analysis are examined in detail as they relate to the developments reported in the thesis. On the whole, the chapter points out problems and limitations existing in current techniques for qualitative reasoning and gives a hint for possible solutions.

Chapter 3 This chapter serves as a motivation for the use of Fuzzy Sets within Qualitative Reasoning and also a concise review of relevant aspects of the fuzzy set theory. The ability of fuzzy sets to ease the representation of common-sense knowledge and to reason with uncertainty is argued. This sets the intuitive as well as theoretical basis for the work presented in chapters 4 and 5.

Chapter 4 This chapter describes the use of fuzzy sets to represent system variables and presents the basic modelling primitives that are allowed through such representation. The definition of a fuzzy quantity space is given that attains the desirable properties for representing variables in a qualitative way, namely, finiteness, granularity, and coverage. The capability of such a quantity space to considerably resolve the common difficulty in determining the boundaries between qualitative values is shown. To reduce computational complexity, potentially involved in the operations

on fuzzy qualitative values, a 4-tuple parametric representation of the qualitative values is then introduced, resulting in a uniform expression of 'sets' and 'values'. Three classes of operations (forming the modelling primitives): algebraic, derivative, and function relational, that may be performed within fuzzy quantity spaces are presented. The advantage that, when reasoning with order-of-magnitude information, the arithmetic operations allowed can help the automation of such reasoning, is exhibited. In particular, the flexibility that the semi-quantitative description of the functional relationships has to enable strength as well as sign information to be captured and represented is shown, significantly extending the capability of qualitative modelling.

Chapter 5 This chapter establishes the fuzzy qualitative simulation technique based on the representation of the fuzzy quantity space and fuzzy constraints provided in the last chapter. A brief functional overview of the technique is presented first, showing the structure of the resulting fuzzy qualitative simulation algorithm, called *FuSim*. The basic concept of qualitative states is then given with both magnitude and rate-of-change of a variable taking values from their corresponding sub-sets of a (fuzzy) quantity space, leading to an explicit representation of the persistence time of a variable within a particular state. Rules for calculating such times are provided. Next, four different types of possible state transitions are discussed based on continuity and differentiability assumptions. The description of the arrival time that a variable takes to transition from one state to another is also presented. After defining an important distance metric between fuzzy qualitative values, rules for determining

arrival times are shown. To minimise spurious behaviour, three filtering methods are developed, namely, constraint filtering, temporal filtering, and global filtering. In particular, the extended function relational constraints and the temporal filter resulting from the use of persistence and arrival times substantially restrict the existence of ambiguous states. The *FuSim* algorithm is now constructed, resulting in a general qualitative simulation tool that can model physical systems with uncertainty and generate the system behaviour in terms of a state sequence with associated temporal durations. At the end of this chapter, examples of using the *FuSim* algorithm are presented.

Chapter 6 This chapter presents the background and architecture of Model-Based Diagnostic Systems. The significance in developing techniques for model-based diagnosis is indicated by illustrating the advantages that such systems can have over traditional rule-based diagnostic systems. To classify the existing model-based approaches and clarify terminology confusion in the literature, basic definitions of static and dynamic systems are provided. A number of typical model-based techniques are then criticised under a classification of these approaches in response to whether they are established to handle static or dynamic physical systems. Based on this, a generic architecture is proposed which views a model-based diagnostic system as a composition of four sub-systems, namely, Predictor, Discrepancy Generator, Candidate Proposer, and Diagnostic Strategist. The generality of this architecture is instantiated by reconstructing two particular diagnostic systems chosen from both static and dynamic classes.

Chapter 7 This chapter develops a model-based diagnostic system for continuous dynamic physical systems: the synchronous iterative diagnostic system named *SID*, based on the use of fuzzy qualitative simulation. Three important features for diagnosing continuous dynamic systems, extracted from the development of *SID*, are first presented: an automatic 'tracking' method for matching the model predictions with observations, enabling discrepancy detection to be performed in a synchronous way without resorting to heuristics; a technique for identifying faults through an iterative search within the model space, supported by a clear visualisation of possible model variations against four characterised modelling dimensions; and a general description of discrepancy metrics for detecting degradation of continuous behaviour. After this, the diagnostic system is constructed with respect to the generic architecture given in the last chapter. In particular, it is argued that, combined with the synchronous tracking technique, the use of fuzzy qualitative simulation for behaviour-predicting provides reliable diagnoses. A detailed discrepancy measurement mechanism is provided by exploiting the fuzzy representation of qualitative states. Rules for searching fault models against modelling dimensions are presented, allowing the use of fault models (if available) to focus the search. Techniques for obtaining fault knowledge are also discussed. Candidates returned are ranked with respect to common-sense knowledge about likelihood of possible faults. Basic functionalities of the diagnostic strategist are pointed out and possible ways to realise these are proposed. Finally, experimental results are given.

Chapter 8 This chapter concludes the thesis by summarising the work that has been achieved herein and indicating limitations of the techniques. It also serves to recommend future work on both the fuzzy qualitative simulation and synchronous iterative diagnosis.

In short, this thesis consists of two significant contributions to the field of Artificial Intelligence in Engineering: a fuzzy qualitative simulation technique for representing and reasoning about physical systems and a synchronous iterative diagnostic method that utilises the former for model-based diagnosis of continuous dynamic systems.

CHAPTER 2

THE DEVELOPMENT OF QUALITATIVE REASONING ABOUT PHYSICAL SYSTEMS

2.1. Introduction

The development of qualitative reasoning about physical systems is rooted in the important, and essential, ability of offering a formal way to represent our knowledge about understanding and interacting with the physical world, without requiring traditional numerical or quantitative techniques. In particular, four basic motivations for developing qualitative reasoners can be identified as presented in the following:

1) To provide inference mechanisms based on the modelling paradigms that accord closely with our common sense intuition of the operation of physical systems, allowing experience and intuition about the physical systems to be represented and utilised as well as the tacit knowledge underlying those physical laws originally described in a quantitative way;

2) To provide a description for physical systems where quantitative methods are ineffective due to lack of knowledge of the structure and behaviour of the systems, enabling reasoning with partial, uncertain or incomplete information;

3) To provide a simpler representation of physical systems whose numerical models already exist and may be well-posed, but the computation involved for obtaining system behaviour from such models is too complex or even prohibitive, thereby reducing computational complexity and increasing economical efficiency;

4) To provide modelling methods that express a system model in a declarative for-

mat such that the same description can be used for a number of different purposes, supporting and automating various effective explanation facilities.

It is obvious that these motivations are not independent of each other and any individual research programme or application domain may exhibit a mixture of the above. In the early eighties, three pioneering approaches to *Q*ualitative *R*easoning (*QR*), i.e., [25], [35], and [50], have been developed that reflect some of these motivations. Since then and, in particular, after the special volume of *Artificial Intelligence* journal, entitled *Qualitative Reasoning about Physical Systems* [4] was published, *QR* has been established as an important research area within Artificial Intelligence. Although *QR* is still comparatively young, much interest has been attracted and significant contributions have been made to consolidate and improve the existing techniques and also to investigate new approaches.

This chapter first briefly outlines the three representative approaches to *QR*, overviewing the advances in *QR* and exposing the remaining problems. The major discussion in the chapter is then dedicated to a critical review of recently developed qualitative reasoners that use a less abstract representation of system variables in an attempt to reduce the essential qualitative ambiguities at source.

2.2. Three Classical Approaches to Qualitative Reasoning

Early approaches to *QR*, typically represented by [25], [35], and [50] have been inspirational in initiating a fundamental review of the way for representing physical systems and of the techniques for generating the corresponding behaviour. Although conflicting terminologies exist in these approaches, as pointed out in [59], they can be classified along four common design choices, namely, *ontology*, *quantity space*, *modelling primitives*, and *simulation algorithm*, as concisely defined in the following:

1) *Ontology:* the fundamental approach adopted to assert structure onto the physical world such that it can be decomposed into simpler, more easily understandable units. This essentially involves describing basic 'elements' from which a complete system model can be composed and the properties of such a model can be inferred.

2) *Quantity space:* the term used to describe the way that system variables are represented. It reflects the distinctive feature of *QR* in that the variables take their values from an underlying finite value set that represents a symbolic or qualitative description with associated semantics. This is therefore one of the key aspects of *QR* and much of the current work is directed at the properties and utility of various quantity spaces (see discussion in the next section).

3) *Modelling primitives:* the set of allowable basic descriptors, used to construct system models. These are intimately linked to the particular ontology chosen, as they affect the realisation of an adopted ontology, though the mapping from ontology to modelling primitives need not be one-to-one but may be one-to-many.

4) *Simulation algorithm:* the method used to describe the technique for generating the new values of system variables when the system is subjected to an external perturbation or producing a description of the system behaviour (over time) as a consequence of a set of initial influences applied to the system. Being another key aspect of *QR*, the simulation algorithms, with associated quantity spaces, have had by far the most development (also refer to the next section).

The basic theme of the discussion below about the three classical approaches to *QR* follows these common design choices. The most highly developed of the three is the *Qualitative Simulation (QSim)* technique [50, 52]. This is based on a *constraint-*

centred ontology, i.e., describing a system by means of a collection of mathematical constraints over system variables. Such constraints can be seen as an abstract description of a set of conventional differential equations which represent a physical system based on physical theories. Its quantity space is an alternating sequence of real numbers and ordinary intervals, including the real number zero and positive and negative infinity. *QSim* defines its primitives in terms of arithmetic, derivative, and monotonic functional constraints to allow ordinary differential equations to be degenerated to the corresponding qualitative differential equations. The simulation algorithm takes as input a set of variables, a set of constraints relating the variables, and a set of initial qualitative values for the variables. By using so-called possible transition rules derived from the ordinary intermediate and mean value theorems [70], a set of possible next states is determined. Then, this set of possible states is further restricted by checking the constraint consistency. That is to say, those possible values which do not satisfy the relationships between the variables are eliminated. In addition, global filtering methods may be employed to further reduce spurious behaviours. As output, *QSim* produces a tree of states with each path standing for a possible behaviour of the physical system being simulated.

An introductory overview of the *QSim* algorithm with its important limitations has been previously presented in [51]. We shall, therefore, not discuss this in further detail. However, it is necessary to point out that the constraint-centred ontology used in *QSim* will be adopted to develop the *QR* method presented later on. Hence, the general structure of our method will appear similar to that of the *QSim* algorithm, but with a number of significant extensions being developed.

The second approach, and the earliest, establishes a *component-centred* qualitative reasoner, named *Envision* [25]. This is accomplished by directly viewing a phy-

sical system as a composition of a number of identifiable components instantiated from a library of generic components in the application domain. It adopts the three symbolic values (−, 0, +) as its quantity space. The fundamental primitives are 'devices' such as pipes, springs, resistors, and transistors. They are represented by the qualitative differential equations describing the behaviour of the corresponding components, usually transformed by mapping the conventional physical equation which the component is consistent with onto one or a set of three valued symbolic algebraic equations. The simulation algorithm starts by making a change to the value of one of the system variables which originally are at an equilibrium state. Then, each particular operating condition of the system, strangely termed 'state', is determined by a process of constraint propagation [110]. The transitions between these possible states are also inferred based on the continuity assumption such that variables must change continuously over a transition. The generated possible transitions are filtered by checking the limit consistency and the constraint consistency. Here, the limit consistency simply requires that transitions from + or − to 0 should not happen together with transitions from 0 to either + or −. The process of constraint propagation, transition analysis and filtering is repeated until all new equilibrium states are attained such that a complete description of all possible interstate behaviours of the system are obtained. However, it is worth indicating that the term 'simulation' algorithm is herein such used, in contrast to the classical definition of a simulation, that the possible states are created without giving the temporal order in which they may occur in the real behaviour. Actually, only equilibrium states are produced. The intermediate states are part of the computational process and *not* a description of the transient of the physical system.

The third typical approach, called *Qualitative Process Theory* (*QPT*) [35], uses

the notion of a process directly or indirectly acting on some objects to modify the states of other objects within the interesting physical situation, thereby providing a *process-centred* ontology of the real world. The quantity space, the term first employed in this approach and now commonly used by other *QR* techniques, consists of a finite number of physically meaningful and important symbolic values, like the height of the top of a tank or the threshold voltage of a transistor, with the real number zero being always included. The basic primitive within *QPT* is the individual view that describes certain portions of a physical system in terms of objects and relationships between the objects, where the relationships may be ordered or monotonic propositions. A qualitative description of the system behaviour is generated by four deductions, i.e., instantiating the processes and views which might occur in a physical situation, selecting process and view instances which are mutually consistent as well as consistent with the known facts about the situation, resolving these structures in an attempt to find the signs of the derivatives of all the variables (i.e., the changing directions of these variables), and determining possible changes of the process and view instances themselves (resulting from the changes in the variables) by relating these derivative signs to the quantity conditions that govern the structures. This approach is the first one that provides a methodology for representing and reasoning with common sense knowledge, though it is the least formal amongst the three approaches.

No matter which ontology a particular approach adopts, common to these techniques, however, is the derivation of an abstract description of the behaviour of physical systems. In other words, they are all aimed at qualitatively simulating the behaviour of physical systems or generating the qualitative behaviour of the system from a description of its structure and the given initial conditions.

It is important to notice that abstract descriptions of system structure, based on particular quantity spaces, make it possible to have more concise representations of behaviour. However, the generation of the behaviour from such descriptions tends not to produce a unique solution [37]. This, of course, is to be expected, as the information required to produce a unique description has been eliminated in the intentional use of such abstraction. Qualitative models, therefore, usually produce ambiguous descriptions of behaviour, resulting in *qualitative ambiguity* as this is often termed in the area of *QR* research. Nevertheless, such ambiguous behaviour can still contain useful information for some application tasks. For example, if it is required to predict whether the current condition of a system can lead to a critical or faulty situation, it may be sufficient to show that none of the possible behaviours leads to a critical state without necessarily knowing which of these behaviours is the 'actual' behaviour of the system. In this way, a task can be satisfied even with incomplete descriptions. Whereas in traditional numerical methods, complete information must be available, and it needs to be precisely and uniquely characterised before *any* inference can be made.

2.3. Recent Approaches to Qualitative Reasoning

It is clear that one research strategy for developing *QR* techniques is to search for a qualitative mathematics capable of yielding significant results from a minimum of information. However, in practice there is usually some quantitative knowledge about the system being modelled and, in some cases, such knowledge may even be very important for depicting the system behaviour, though rarely enough to specify a complete numerical calculation. Early methods for the qualitative description of a physical system are too restrictive to use this knowledge because of their fundamental limitations in the representations of the physical structure as follows:

1) A weak representation of functional dependencies between variables; typically, this captures sign information but has no measure of strength of dependency;

2) Only sign information on the rate-of-change of the variables is represented, lacking ordering information amongst the rates of change;

3) No explicit representation of time.

It is, therefore, desirable to develop *QR* techniques which permit a more detailed description of quantities and functional relationships than existing qualitative reasoners, and yet requires neither the precision of real numbers nor exact and complex relations between variables so as to allow effective and efficient designs to be provided. Also, a technique which can qualitatively generate a physical system's behaviour in terms of its state sequence with associated temporal durations will significantly enhance the viability for real application of such approaches.

Recently, important work has been reported that either represents system variables using a less abstract quantity space or integrates quantitative and qualitative information. Less spurious behaviours are therefore obtained, though the reduction is, of course, determined by the kind of tasks to which these techniques are applied. This section presents a review of the representatives of such work with two subsections, one focusing on those qualitative reasoners that utilise the theory of *Non-Standard Analysis (NSA)* [83, 84] as their common mathematical underpinning, the other concentrating on some of the remaining typical approaches. Although there are many existing *QR* techniques that are not covered by this section, the work reviewed herein is directly linked to the particular *QR* technique to be developed within the thesis. A good collection of recent important contributions to *QR*, however, can be found in [117]. A substantial part of this review has been reported in [93, 97].

2.3.1. Non-Standard Analysis Based Approaches

Several techniques have been established for *'order of magnitude reasoning' (OMR)* based on *NSA*. This supports a calculus reflecting common sense knowledge about the effects of different orders of magnitude in the representation of system variables such that lower level or 'second order' effects in models can be neglected. When used as a self-contained reasoning system *OMR* produces a consistent framework for describing caricatures or exaggerated version of a physical system. As such it is useful for explaining limiting cases. Difficulty comes, however, when comparing the behaviour of such models with classical real-valued descriptions. In these cases, as discussed in the following, non-intuitive behaviour can result.

The pioneering research on *OMR* was presented in [80], a technique named *FOG*. It is concerned with reasoning with relative magnitudes, i.e., with the analysis of physical systems in which one quantity is much greater than another, or in the comparison of two systems of the same structure, but which have corresponding quantities of very different magnitudes. A set of inference rules defined on the basis of *NSA* have been developed through the following relational operators:

$A\ N_e\ B$ indicates A is negligible in comparison with B;

$A\ V_o\ B$ indicates A is close to B;

$A\ C_o\ B$ indicates A has the same sign and order of magnitude as B.

By the use of these rules, which are correct under a strict *NSA* interpretation, some secondary effects of physical systems can be eliminated, whilst maintaining the main properties of the systems. However, *FOG*'s algebraic operations in *NSA* result in the search for limits of the real numbers when *FOG* is intended to cope with practical problems. Unfortunately, it is impossible to exactly define a set, say, (a, b), to approach the set of infinitesimals in the field of real numbers, such that once

$A \in (a, b)$, $B \overline{\in} (a, b)$, and $A, B \in R$, we could consider $A \ N_e \ B$. Although *FOG* does not specify the size of such intervals, it is argued in [80] that they exist and that purely symbolic qualitative reasoning can be performed. However, this is not the case. In fact, without losing generality, three sets below can be used as the key intervals [80]:

$$I_1 = (-a, a), \quad I_2 = (1-a, 1+a), \quad I_3 = (b, c),$$

where $1 \gg a > 0$, $a < b < 1-a$, $c > 1+a$, and $a, b, c \in R$. If it is given that $A/C \in I_1$, $AB/CD \in I_2$, and $C \neq 0$, then, through ordinary interval algebraic operations [76], it follows that

$$D/B = (A/C) / (AB/CD) \in I_1 / I_2 = (-\frac{a}{1-a}, \frac{a}{1-a})$$

$$= (-\frac{a}{1-a}, -a] \cup (-a, a) \cup [a, \frac{a}{1-a}).$$

It is, therefore, not guaranteed that $D/B \in (-a, a)$, resulting in the invalidation of *FOG* rule R_{27}, i.e., if $AB \ V_o \ CD$, $A \ N_e \ C$, and $Sign(C) \neq 0$ then $D \ N_e \ B$, which was used to justify *FOG* in terms of *NSA* and is, of course, correct under the *NSA* interpretation. Further, it is possible to obtain the result that $D/B \in I_3$ since $I_3 \cap [a, \frac{a}{1-a})$ is not necessarily equal to the empty set. This would mean that, if AB is close to CD and A is negligible in relation to C with $C \neq 0$, then D might have the same sign and order of magnitude as B. This is inconsistent with a common-sense interpretation. Also, the set of *FOG*'s inference rules looks somewhat arbitrary and, therefore, *FOG* has to rely on other control techniques to govern the application of the rules. In addition, the method is developed for algebraic equations only and does not allow reasoning about the behaviour of dynamic systems.

An extended technique able to reason about the qualitative behaviour of dynamic

systems was presented in [15], called *order of magnitude reasoning in qualitative differential equations* and abbreviated to *CHEPACHET*. It makes use of the information provided by orders of magnitude in solving qualitative constraints by representing quantities and derivatives in terms of distinguished subsets of the *non-standard real line* [84], for instance,

$$(-MEDIUM) + MEDIUM = X,$$

$$X \in \{-MEDIUM, -SMALL, \{0\}, SMALL, MEDIUM\}.$$

Also, it formulates a number of rules governing the behaviour of system variables over orders of magnitude and, in particular, provides a mechanism for calculating the duration of state changes as infinite or infinitesimal in time. This later contribution was an important insight and provided the basis for calculating durations from rates of change and magnitudes of quantities. Nevertheless, the previously discussed boundary problem, namely the difficulty in interpreting underlying real-valued variables with respect to the qualitative representation used, also exists within the description of the quantities and derivatives of *CHEPACHET*. Take two so-called disjoint qualitative sets $SMALL = \{x \mid x > 0, x \ll 1\}$ and $MEDIUM = \{x \mid x > 0, x \sim 1\}$ [15] as an example. In general, it is not realistic to draw an exact boundary between these two. Another problem with this approach is that it allows only a fixed quantity space of orders of magnitudes of the system variables, and, therefore, the algebraic operation system has to be changed whenever it is used to incorporate different sets of orders of magnitude.

A similar approach to *CHEPACHET*, called *HR-QSIM*, was reported in [114], which is more expressive in the magnitude values of system variables than *CHEPACHET*. It extends the quantity space of the *QSim* algorithm to include infinitesimal and infinite numbers, although its qualitative derivatives can only have

four possible orders of magnitude. For instance, *QSim*'s qualitative values 0, (0, p), p (see figure 2.1.a) are represented within *HR-QSIM* as shown in figure 2.1.b., where, e.g., (*halo*, 0, +) denotes a set of positive infinitesimals and <0, p> indicates the set of real (non-infinitesimal) numbers between values 0 and p. *HR-QSIM* was developed as the main inference algorithm within a comparative analysis task, named *Exaggeration* [113, 116]. The purpose of the simulation performed within this framework is to predict changes in behaviour resulting from extreme, i.e, not physically possible, changes in parameter and/or variable values. For such a task *HR-QSIM* is clearly very useful. However, techniques developed within *HR-QSIM* are not generally applicable to qualitatively simulating actual physical systems. In fact, *HR-QSIM* would also suffer from the boundary problem if used for non-exaggerative reasoning, resulting in non-intuitive interpretations of the set of behaviours in a particular application domain. This largely restricts *HR-QSIM*'s capability to serve as an otherwise powerful *QR* technique.

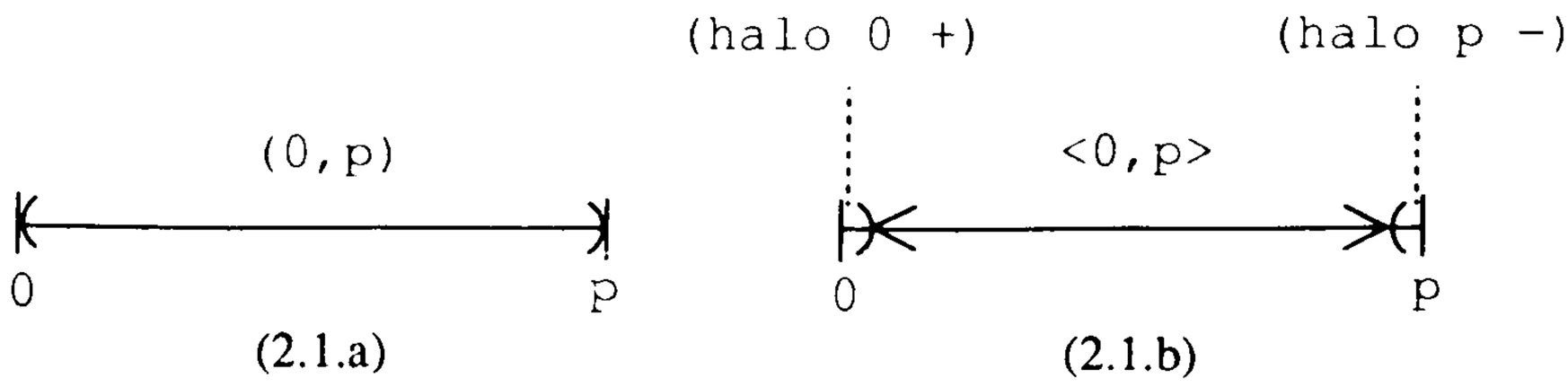


Fig. 2.1. *Comparison between the QSim and HR-QSIM's Quantity Spaces*

In *HR-QSIM*'s quantity space, for instance, the interval $(0, +\infty)$ could be represented as (*Halo* 0 +) and $<0, +\infty>$. Denoting X_1 = (*Halo* 0 +) and $X_2 = <0, +\infty>$, for any $x_1 \in X_1, x_2 \in X_2$, it follows that x_1 is a positive infinitesimal and x_2 is a finite real number, according to the notation of the quantity space. However, for a potential application in real situations where variables take values only

from the real line, we have to use, say, a set of small real numbers to approximately approach the set of infinitesimals. The degree of which a real number belongs to such a set usually cannot be described by either full membership or absolute non-membership. In fact, without losing generality, there are equivalent expressions for X_1 and X_2, we may denote them as $X_1 = (0, a)$ with a being any infinitesimal and $X_2 = (b, +\infty)$ with b being a finite real number. Once we try to choose a small number to approach the least upper bound of the set of infinitesimals the boundary problem would be incurred. The same problem would also arise when trying to determine the largest lower bound of set X_2. This is exactly the situation that requires gradual membership and no excluded-middle law; both of which are fundamental motivations for the development of fuzzy mathematics [126].

Both *CHEPACHET* and *HR-QSIM*, however, provide an important extension to the traditional three-value representation of the qualitative derivative of a variable, i.e., $\{+, 0, -\}$ or $\{inc, std, dec\}$ [4], though still in a very weak form. This is a crucial advantage that allows ordering information on the rates of change of the system variables to be used to calculate a temporal duration associated with each qualitative state. Henceforth, powerful temporal filtering methods can be developed that eliminate many of spurious behaviours, which cannot be ruled out by conventional filtering methods. Such a significant advantage enables considerable progress towards effective qualitative reasoning about continuous dynamic systems. Unfortunately, important limitations exist with both temporal approaches. For *CHEPACHET*, which intends to reason about actual physical systems, transition analysis, i.e., the analysis of the relationship between two adjacent states, ignores the fact that when dealing with the behaviour of continuous variables, with adjacent system states being represented by 'crisp' intervals on the real line, there is no time cost for a state to transition to

another. This results in the potentially powerful temporal filtering methods to be ill-conceived and inapplicable, although its temporal topology [15] is correct under a strict *NSA* interpretation. This is because such methods are essentially based on the use of the variance of a variable that governs the time taken to transition between adjacent states. To cope with exaggerative reasoning the transition analysis in [113, 114, 116] fortunately avoids this problem, since *HR-QSIM* is intended to consider an extreme behaviour that is qualitatively distinct in a crisp sense and continuity is an unintuitive property when applied to exaggerated variables. Notice that, to perform temporal filtering in *HR-QSIM*, instead of using the variable's variance, an alternative expression is utilised, called arrival time (informally, the time taken for a variable to reach its next state after leaving from its previous one); however, the essence and intuition behind this are the same as in *CHEPACHET*. Hence, *HR-QSIM*'s temporal filtering techniques cannot be directly utilised for simulating physical systems other than exaggerated ones without encountering the boundary problem, even though its temporal filters are very useful for exaggerative tasks.

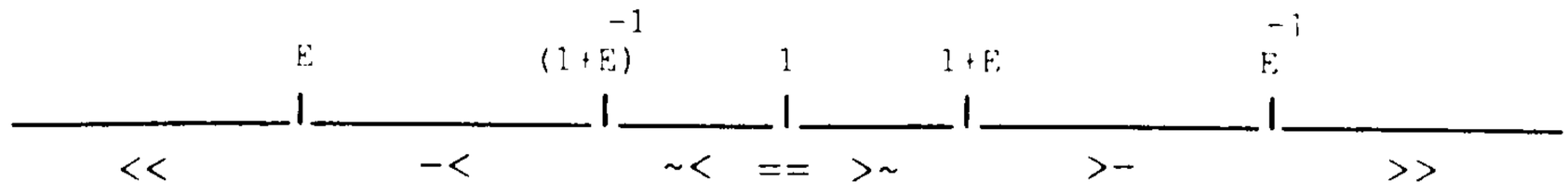
It is worth pointing out that, in addition to the above discussed problem, another fundamental limitation of *HR-QSIM* is that it does not indicate how to calculate the arrival time at transitions where the predecessor and successor states of a variable have different derivatives. However, there are many situations in which *HR-QSIM* has to cope with such transitions. For example, from the transition tables in [116], for a *CO*-transition from (l_i, std), the following states are two of the possible next states: $((Halo\ l_i +), (inc, negl))$, $((Halo\ l_i -), (dec, negl))$, both having different derivatives from *std*. Also, for an *NS*-transition from $((Halo\ 0 +), (inc, negl))$, both $((Halo\ 0 +), (inc, fin))$ and $(<0, inf>, (inc, fin))$ can be the successors. Unfortunately, *HR-QSIM* does not address the computational method for the arrival times

associated with these transitions, remaining more spurious behaviours than otherwise could have been filtered. In the above, a *CO*-transition indicates a variable transitions from a closed set to an open set and an *NS*-transition does the event when a variable transitions from one state to another such that its qualitative value changes from a non-standard open value to a standard open value or its qualitative derivative changes from a negligible rate to a finite rate [114, 116].

2.3.2. Other Related and Important Approaches

Recent developments are certainly not restricted to those based on *NSA*, there are other different approaches towards more viable techniques for *QR* than traditional ones. Nevertheless, the following discussion only concentrates on the discipline that is more related to our work later.

An important approach to reasoning with orders of magnitude and approximate relations, inspired by *FOG* and named *O[M]*, has been reported in [73, 74]. Rather than based on *NSA*, this technique is dependent upon seven primitive relations holding amongst system variables and/or variables' values (so-called quantities collectively): 'much smaller than', 'moderately smaller than', 'slightly smaller than', 'exactly equal to', 'slightly larger than', 'moderately larger than', and 'much larger than', denoted by $<<$, $\prec$, $\sim<$, $=$, $>\sim$, $\succ$, and $>>$, respectively. Each of these relations is interpreted in terms of the location of the quotient of the two compared quantities within some interval, with all such intervals (which are disjoint) being defined with respect to a unique (accuracy) parameter chosen according to domain knowledge (see figure 2.2.a). Thereby, *O[M]* provides a concrete semantics for the relations and allows 'exact' inferences to be performed.



(2.2.a)

(2.2.b)

Fig. 2.2. *Strict and Heuristic Interpretations of O[M] Relations*

Whilst such an interpretation is accurate when the 'accuracy' parameter can be specified, however, as pointed out in [73], it is too strict compared to human reasoning. In other words, it is very difficult to determine the unique boundary between two disjoint intervals used to describe two different relations. Realising this, a heuristic interpretation is further established in [73, 74] by replacing the boundary points of the intervals with (crisp) regions as shown in figure 2.2.b. This technique allows more aggressive and human-like inferences, nevertheless, it does not fundamentally remove the boundary problem. Actually, it can be seen from figure 2.2.b that, this approach shifts the difficulty in choosing the boundary between the two intervals indicating two different relations to that between an interval and a region which substitutes the (point) boundary. In so doing, *O[M]*, as with *FOG*, does not address the gradual nature of the introduced relations, i.e., the extent to which a quantity is much smaller or moderate larger for instance, in comparison with another one is often a matter of degree. In addition, *O[M]* remains a reasoning method for coping with static systems only.

There has also been considerable work relevant to the integration of quantitative and qualitative knowledge [3, 36, 53, 86, 99, 124] to reduce ambiguities, though each

has utilised a different methodology to handle the quantitative information. For example, the method reported in [53] uses incomplete quantitative information to augment the qualitative descriptions such that, by propagating upper and lower numerical bounds across constraints, it provides useful information (in terms of ranges) about variables whose values may not be known. However, this method applies quantitative knowledge to individual complete qualitative behaviours from the output of qualitative simulation, rather than interleaving quantitative and qualitative processing at the local propagation stage, so an entire tree of qualitative behaviours has to be generated before some sub-trees could be eliminated and this largely decreases the efficiency of the overall simulation.

Another important work is that presented in [124], where traditional three-valued symbolic algebra, i.e., the algebraic system ($\{-, 0, +\}$, $\dot{+}$, $\dot{-}$, $\dot{\times}$, $\dot{/}$), is merged with the ordinary real algebra. As such quantities may first be operated on by conventional arithmetic operators, while the results of these operations are then abstracted to the traditional symbolic values and classical symbolic operations are further evaluated. This produces a result that is less ambiguous than that produced by the traditional qualitative operations alone. However, the basic idea behind this technique seems to resort to using pure numerical information first, provided by real algebra, otherwise, such an algebra system would be exactly the same as the one used in [25], though the latter does not cope with the algebraic analysis of multiplication and division. Since the real algebra is utilised as a sub-algebra of the whole qualitative algebraic system this is somehow in conflict with one of the fundamental requirements of *QR*: identifying appropriate abstractions of the physical world by quantising the real line into a finite set of distinctions. This approach essentially consists of coupling a numerical reasoner and a qualitative reasoner together.

An interesting variation in the development of *QR* techniques is to provide the constraints in a causally ordered form. The resulting simulation algorithm is termed the *constructive* algorithm [119, 120] that utilises unidirectional constraints to represent the fixed structure of the system model with the perspective of system causality remaining unchanged throughout the simulation process. Such causally ordered constraints may be automatically generated from a given set of bi-directionally represented equations [48]. Within this constructive algorithm, values are propagated through constraints only when required. It allows the separation of the algebraic and derivative constraints into instantaneous propagation and infinitesimal or finite durations respectively. However, qualitative ambiguity also exists in this case, it appears as uncertainty in the ordering of transitions between states. To tackle this problem, the notion of differential planes is introduced, allowing the system model to be repeatedly differentiated so that higher-order derivative information can be utilised. Nevertheless, this will encounter a difficulty in determining how many such planes to be included in the model for each particular physical system during the simulation process.

2.4. Summary

This chapter briefly overviews three pioneering approaches to *QR* in terms of the four basic design choices they followed, namely ontology, quantity space, modelling primitives, and simulation algorithm. Whilst not resulting in effective qualitative reasoners, these early contributions to *QR* have inspired a lot of further approaches that greatly improve and extend the prospects. More recent developments, in particular, those based on *NSA* are presented as a central part of the chapter to show that *QR* has experienced a rapid growth within the last five years. However, important limitations still remain as also discussed in this part of the chapter since the research area of *QR*

is still embryonic and much needs to be done. It is clear, that an extension to current *QR* methods, to make them capable of capturing and using both sign and strength information on variable values and functional dependencies and also capable of extracting temporal information, based on well-developed mathematical techniques, would greatly enhance the effectiveness of *QR* approaches for those potential applications where such information is available. Chapters 4 and 5 will be dedicated to developing such a qualitative reasoner based on the theory of Fuzzy Sets. The next chapter will overview the motivations for the adoption of Fuzzy Sets and the theory relevant to the subsequent development.

CHAPTER 3

MOTIVATIONS FOR USING FUZZY SETS AND A REVIEW OF THE RELEVANT THEORY

3.1. Introduction

It is clear from the last chapter that the basic goal of *QR* is to develop Artificial Intelligence methods for qualitatively representing and reasoning about the behaviour of physical systems. An alternative, and distinct, approach to coping with modelling complex physical systems and reasoning with imprecise or uncertain information in a non-numerical way has also been independently developed based on the use of Fuzzy Sets [128]. Actually, the theory of fuzzy sets has as its main aim the formulation and solution of problems that are too complex or ill-defined to be susceptible to analysis by conventional quantitative techniques.

This chapter reveals the basic motivations for adopting fuzzy sets in *QR* by exhibiting the important advantage that fuzzy sets can offer for easing the representation of physical quantities. It presents the essential aspects of the theory of fuzzy sets, mainly in terms of mathematical definitions, providing the theoretical foundation for developing the fuzzy qualitative simulation technique presented later. Although definitional, the description of the basics of the theory also serves as a formal way to clarify how uncertain knowledge may be expressed using fuzzy sets and how such knowledge can be reasoned with by manipulating or composing the graded memberships of the associated fuzzy sets. This reinforces the original intuition for combining Fuzzy Sets and Qualitative Reasoning.

3.2. Motivations for the Use of Fuzzy Sets

Fuzzy set theory deals with a subset of a universe of discourse, where the transition between full membership of a set and no membership is gradual rather than abrupt. Such subsets -- called *fuzzy sets* -- arise, for instance, when descriptions of ambiguity, vagueness, and ambivalence in the mathematical models of physical systems are needed. In the real world, the attributes of the system variables often emerge from an elusive vagueness or fuzziness, a re-adjustment to context, or an effect of human impreciseness. The use of the 'soft' boundaries of fuzzy sets, i.e., the graded memberships, allows subjective knowledge to be utilised in defining these attributes. With the accumulation of knowledge the subjectively assigned memberships can, of course, be modified [28]. Even in some cases where precise models are available, fuzziness may be a concomitant of complexity.

Advantages resulting from the adoption of fuzzy sets ease the requirement for encoding the knowledge about physical systems. As a simple example, a fuzzy set '*approximately 4*', within the real line R, can be depicted by figure 3.1, where $\mu_{Four}(x)$ denotes the degree of membership that an $x, x \in R$, belongs to the set 'approximately 4'. It is natural to assert that 4 belongs to this set with a full membership $\mu_{Four}(4) = 1$ and that 3 and 6 belong to it with memberships, say, 0.83 and 0.31, respectively; whereas 10 does not belong to the set, i.e., $\mu_{Four}(10) = 0$. Considering another example, although the labels *small*, *medium*, and *large* have an intuitive appeal, if we try to use these labels to represent some values of physical variables by interpreting them with crisp intervals, such as $small = \{x \mid x > 0, x \ll 1\}$ and $medium = \{x \mid x > 0, x \sim 1\}$ as in [15], it would then result in a non-intuitive representation. As indicated in section 2.3.1, drawing an exact boundary between these sets is not practical. Actually, when encoding a certain real number there will

exist difficulty in deciding which of the two sets the number should definitely belong to. It may well be the case that we can only say this number belongs to the *small* set with possibility A and to the *medium* one with possibility B. Recalling the discussion in the last chapter, the common basis of three major approaches to order-of-magnitude reasoning (namely, *FOG*, *CHEPACHET*, and *HR-QSIM*) is the theory of non standard analysis, which is based on classical set theory, whilst it appears that fuzzy concepts have been intuitively adopted within these techniques (including *O[M]*). Thus, if they are intended to perform qualitative reasoning about physical systems other than on exaggerated models problems would be incurred as analysed previously. To overcome such problems we propose to utilise fuzzy set theory as the mathematical basis to develop our *QR* technique.

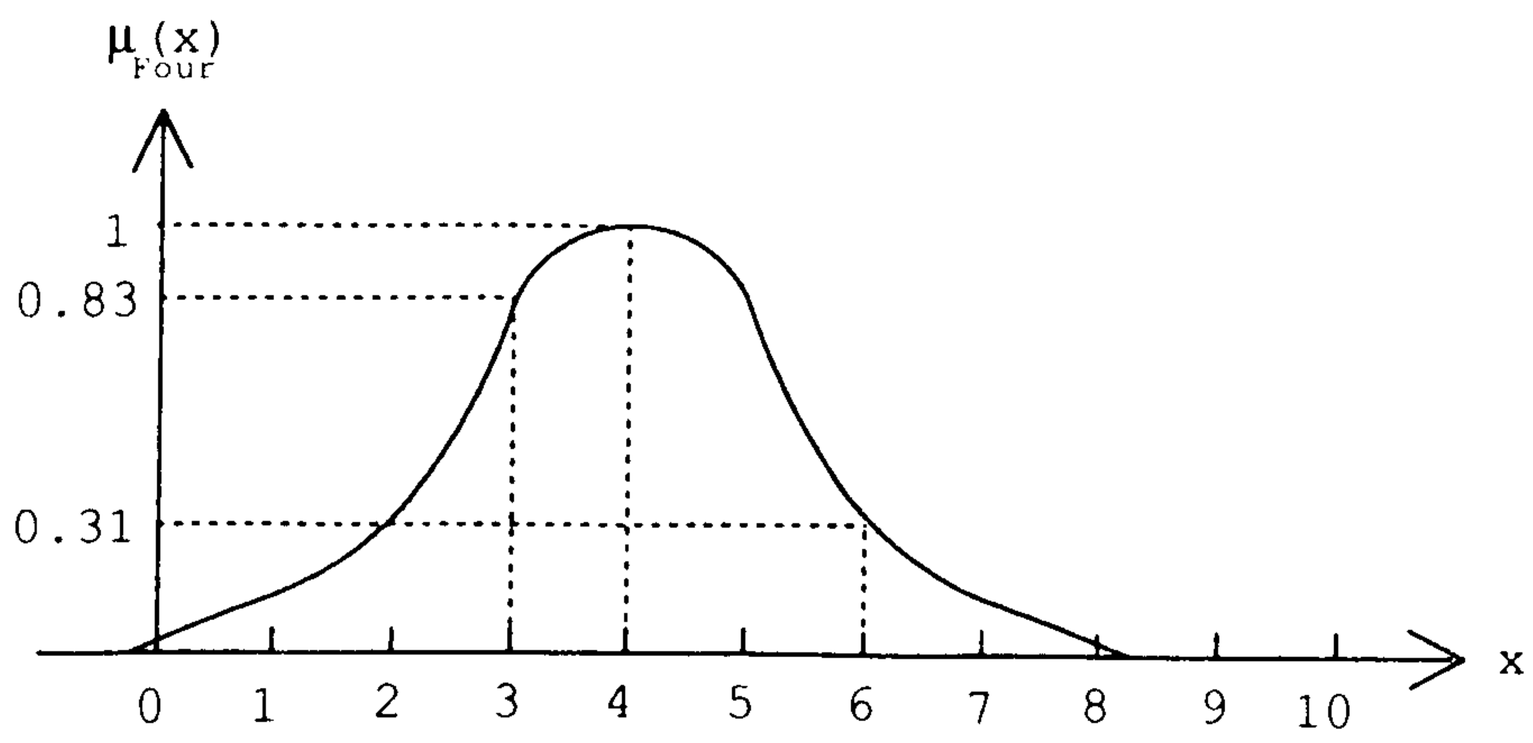


Fig. 3.1. *Approximately Four*

It is important, however, to notice that the concept of a fuzzy set is not merely a disguised form of subjective probability, although both the theory of fuzzy sets and the theory of probability are developed to attain realistic solutions to problems in decision analysis under uncertainty. In essence, fuzzy set theory is aimed at dealing with sources of uncertainty or impreciseness that are inherently vague and nonstatistical in nature. For example, the proposition *'X is a large number,'* in which *large number* is a label of a fuzzy subset of nonnegative integers, defines a *possibility*

distribution rather than the *probability distribution* of X. This implies that, if the degree to which an integer I fits our subjective perception of *large number* is $\mu(I)$, then the possibility that X may take I as its value is numerically equal to $\mu(I)$. Consequently, such a proposition conveys no information about the probability distribution of the values of X. Thus, probability theory does not provide an adequate tool for the analysis of problems in which the available information, like this proposition, is incomplete, imprecise, or unreliable. Attempts to utilise probability theory in qualitative reasoners have been made [30]. However, as argued above this does not conform to one of the basic motivations of *QR*: that of capturing and representing knowledge expressed in a vague form.

Since the middle of the seventies, fuzzy mathematics has been increasingly applied to many different fields of information processing, such as pattern recognition, signal processing, and process control; examples can be found in [45, 46, 56, 72, 85, 88, 89]. It is also being incorporated within systems based on Artificial Intelligence techniques, and in particular, within Knowledge Based Systems, as a method of representing and handling knowledge [7, 132, 133]. These applications demonstrate that fuzzy set theory provides a conceptual framework for the solution of imprecisely formulated problems.

Considering the fundamental motivation for developing *QR* techniques and the introduction of fuzzy set theory, it can be seen that both cases are rooted in a common mainstay: handling physical world qualitatively instead of quantitatively and explicitly coping with the essential uncertainty or incompleteness of practical models. We assert that Qualitative Reasoning based on current qualitative modelling techniques and Approximate Reasoning [129, 130, 131] based on fuzzy set theory are two strands of Artificial Intelligence research that are essential to developing and capturing

common-sense reasoning. Unfortunately, these two approaches have been independently developed by separate communities, and very little cross-fertilisation has occurred [12, 13, 32] (see relevant discussion in [90, 93, 97]). Therefore, it is of great interest and potential benefit to attempt to synthesise these two strands. Adopting this, the following two chapters present a semi-quantitative extension to qualitative modelling by means of fuzzy mathematics. Through this extension, it can be shown that the limitations existing in current *QR* approaches as pointed out earlier may be significantly reduced or resolved, and that fuzzy set theory can serve as a common mathematical tool for developing a formal foundation for the qualitative simulation of complex systems.

In general, the fuzzy *QR* method to be developed is, at first sight, a constraint-centred approach like *QSim* [52], it views the structure of a physical system as a set of abstract equations derived from physics. This results in an explicit representation of physical systems. It is because this method adopts the same modelling ontology as *QSim* that it is called *Fuzzy Qualitative Simulation* and abbreviated to *FuSim*. Conversely, Fuzzy Sets are concerned with representing common sense knowledge, for example, empirical judgements from the designers or operators of the system. Thus, *FuSim* can be considered as an integrated system that utilises an explicit model based on scientific presumptions *and* empirical judgements using fuzzy logic to combine the merits of both approaches.

As a particular but important application, *FuSim* will serve as the behaviour simulator within the model-based diagnostic system to be established in the chapters 6 and 7. With such an application, the adoption of Fuzzy Sets allows appropriate measures to be readily incorporated for detecting discrepancy between real observations from the physical system being diagnosed and the predictions from the model, thereby

easing the development of the diagnostic system.

3.3. A Review of the Relevant Aspects of the Theory

This section presents a brief review of the relevant aspects of fuzzy mathematics which forms the basis of fuzzy qualitative simulation. A more extensive treatment of fuzzy set theory can be found in [31, 49, 125].

Let X be a classical set of objects, called the universe, whose generic elements are denoted x. Then a *fuzzy set* is a set of pairs

$$A = \{(x, \mu_A(x)) \mid \mu_A(x) \in [0, 1], x \in X\},$$

where $\mu_A(x)$ is called the grade of membership of x in A, or sometimes, the membership distribution [49, 129]. Clearly, the closer the value of $\mu_A(x)$ is to 1, the more x belongs to A. When $\mu_A(x)$ is restricted to the values 0 and 1, the fuzzy set A degenerates to an ordinary set and its membership distribution becomes the characteristic function of the classical set. If X is the real line R, A is called a *fuzzy number*. Furthermore, if A satisfies the two conditions below

(1) $x \in X, \mu_A(x) = 1;$

(2) $x_1, x_2 \in X, \quad \lambda \in [0, 1],$

$$\mu_A(\lambda x_1 + (1-\lambda)x_2) \geq \min(\mu_A(x_1), \mu_A(x_2)).$$

A is a *normal and convex fuzzy number* [31, 129]. Actually, the fuzzy set 'approximately 4', given in the previous section, is such a fuzzy number, as illustrated in figure 3.2.

When we want to exhibit an element $x \in X$ that *typically* belongs to a fuzzy set A, we may demand its membership value to be greater than some threshold $\alpha \in [0, 1]$. The ordinary set of such elements is called the α–*cut* of A and denoted A_α [31]:

$$A_\alpha = \{x \in X \mid \mu_A(x) \geq \alpha\}.$$

For example, two characteristic functions of the α-cuts of 'approximately 4' are shown in figure 3.3 (in thicker lines), where α is equal to 0.5 and 0.9, respectively.

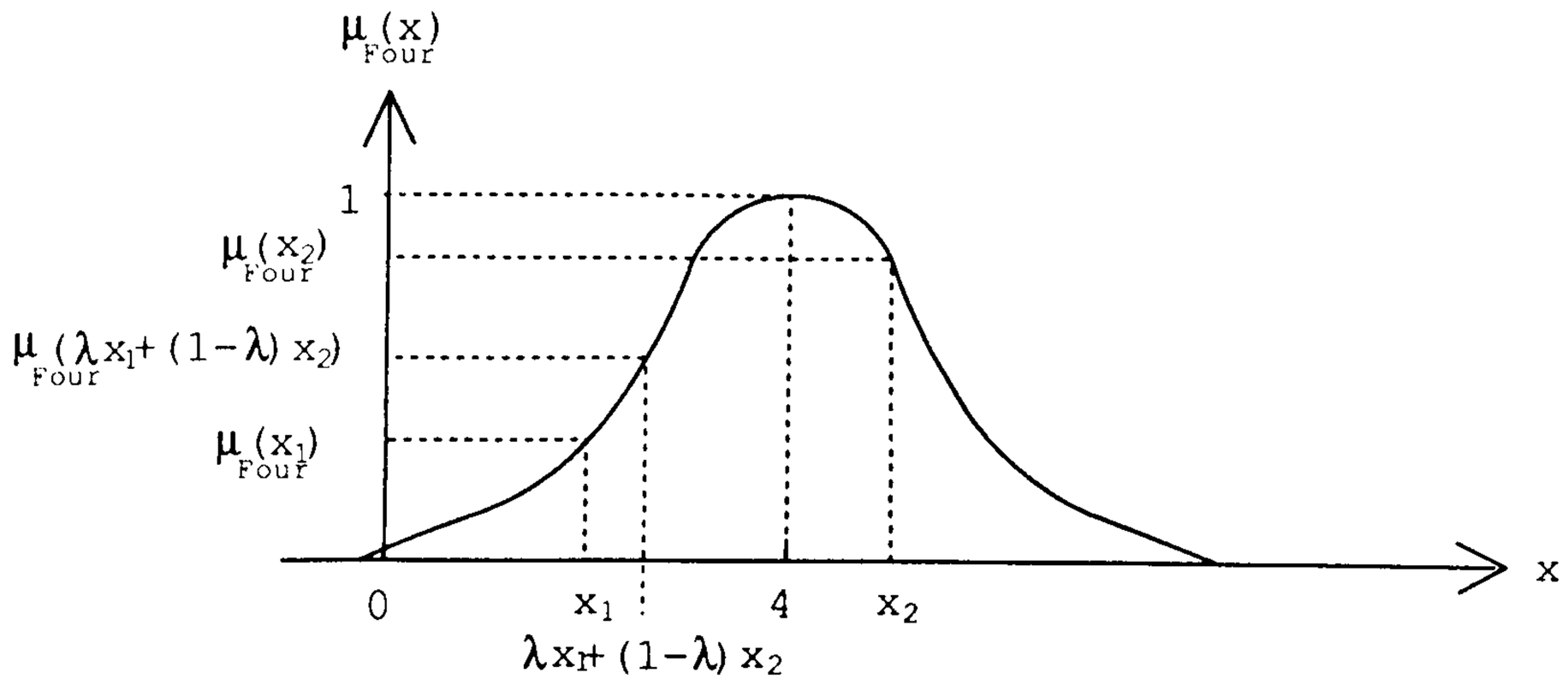


Fig. 3.2. *A Normal Convex Fuzzy Number*

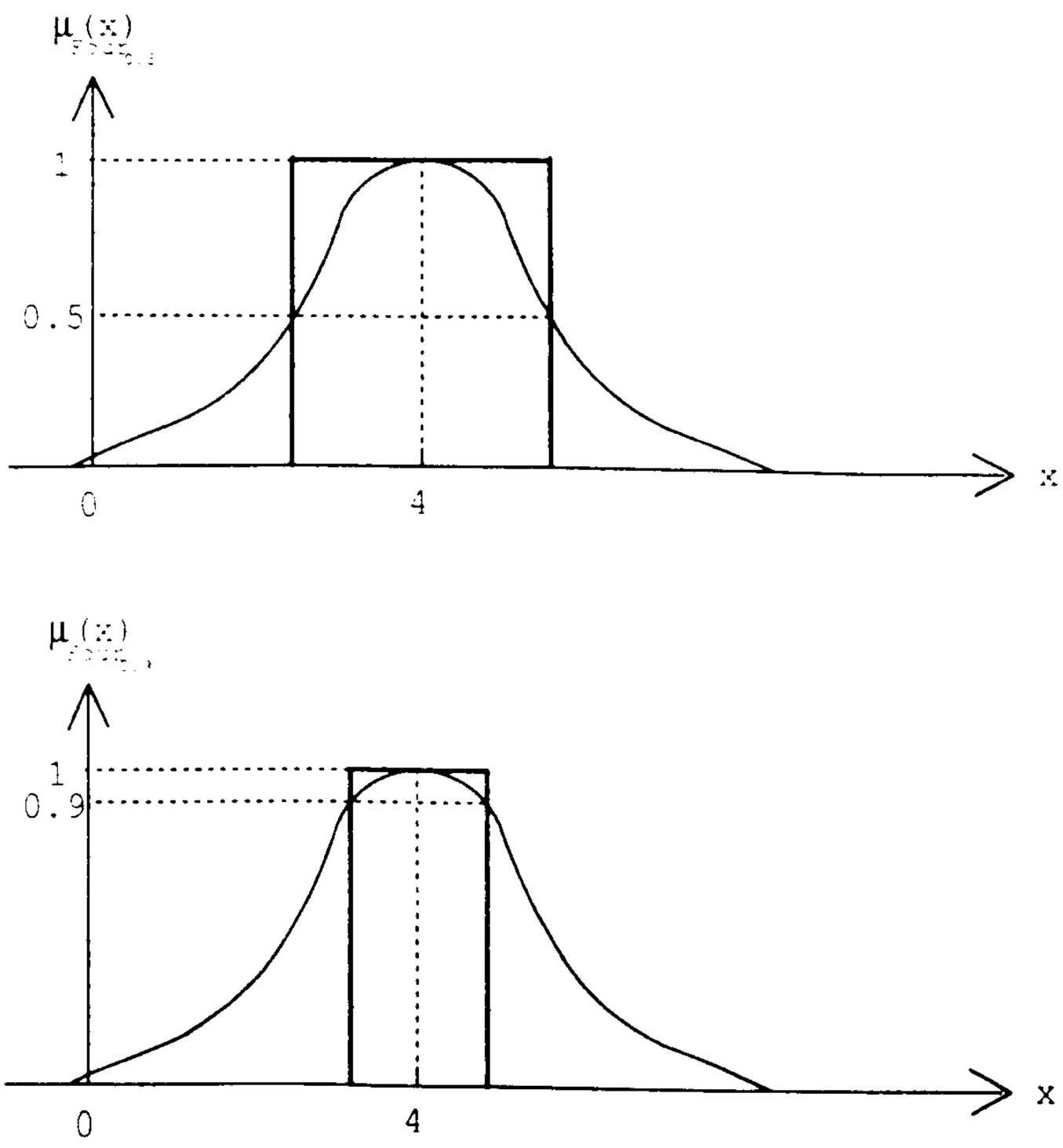


Fig. 3.3. *The α-Cuts of 'Approximately Four'*

Using the concept of α-cuts, a fuzzy set A may be decomposed into its associated α-cut A_α through the *resolution identity* [49]:

$$A = \int_0^1 \alpha A_\alpha,$$

where αA_α is the product of a scalar α with the set A_α and $\int_0^1$ is the union of the A_α, with α ranging from 0 to 1. The resolution indentity can be viewed as the result of combining those elements in A that fall into the same level set.

One of the most basic ideas of fuzzy set theory, which provides a general extension of classical mathematical concepts to fuzzy environments, is the *extension principle*. This can be stated as follows [129]:

> If an n-ary function f maps the Cartesian product $X_1 \times X_2 \times \cdots \times X_n$ onto a universe Y such that $y = f(x_1, x_2, ..., x_n)$, and $A_1, A_2, ..., A_n$ are n fuzzy sets in $X_1, X_2, \ldots, X_n$, respectively, characterised by membership functions $\{\mu_{A_i}(x_i), i=1, 2, ..., n\}$, a fuzzy set on Y can then be induced such that

$$\mu_B(y) = \begin{cases} \underset{\substack{x_1, \ldots, x_n \\ y=f(x_1, \ldots, x_n)}}{Sup} \min(\mu_{A_1}(x_1), ..., \mu_{A_n}(x_n)), & f^{-1}(Y) \neq \Phi; \\ 0, \quad f^{-1}(Y) = \Phi, \end{cases}$$

where Φ is the empty set.

The extension principle can be used to define a set of algebraic operations, O, on a family of fuzzy sets U in a universe X such as addition and multiplication. If U is finite, the algebraic system $<U, O>$ may not be closed. In order to maintain U unchanged after algebraic operations as often required, an *approximation principle* [5, 129, 130] is used such that for an n-ary operator $o \in O$, and n fuzzy sets

$A_1, A_2, ..., A_n \in U$, if $\hat{A} = o\ (A_1, A_2, ..., A_n)$ and

$$d\ (\mu_{\hat{A}}(x), \mu_A(x)) = \min_{B \in U} d\ (\mu_{\hat{A}}(x), \mu_B(x)), \qquad A \in U,$$

then the approximation of $\hat{A}$ is A. Where $d(.,.)$ is any distance which satisfies the axioms of a metric [87].

It is clear that the selection of a distance metric plays a critical role in the application of the approximation principle. However, no matter which kind of distance metric is used, it is usually not efficient to directly calculate the distance between the membership distributions. In order to save computation time and memory storage, a metric based on appropriate features of the membership distributions is usually used. Two important features of a fuzzy set are its *power* and *centre*. The power of a fuzzy set A is defined as the integral of the membership distribution $\mu_A(x)$, i.e.:

$$Power\,(A) = \int_R \mu_A(x)\ dx.$$

The centre of A is taken as the central element among all those elements whose degree of membership within A are equal to the maximum value of the overall membership distribution.

Another crucial concept in fuzzy set theory is the *fuzzy relation*, which is a generalisation of the normal crisp relation. An n-ary fuzzy relation in $X_1 \times X_2 \times \cdots \times X_n$ is, in fact, a fuzzy set on $X_1 \times X_2 \times \cdots \times X_n$. Fuzzy relations can be composed, and this composition is closely related to the extension principle. For example, if R is a relation from X_1 to X_2 (or, equivalently, a relation in $X_1 \times X_2$), and S is a relation from X_2 to X_3, then the *composition* of R and S is a fuzzy relation from X_1 to X_3 denoted by $R \circ S$ and defined by [31, 127]

$$\mu_{R \circ S}(x_1, x_3) = \sup_{x_2 \in X_2} \min\ (\mu_R(x_1, x_2), \mu_S(x_2, x_3)), \qquad (x_1, x_3) \in X_1 \times X_3.$$

Fuzzy relations and fuzzy relation composition form the basis of Approximate (or Fuzzy) Reasoning [31, 129, 130, 131]. Informally, such reasoning executes an inference process by which a possibly imprecise conclusion is deduced from a collection of imprecise premises. For example, a rule like

$$\text{if } x \text{ is } A_i, \text{ then if } y \text{ is } B_i, \text{ then } z \text{ is } C_i,$$

which governs the relationship between the premises A_i, B_i and the conclusion C_i, can be translated into a fuzzy relation

$$\mu_{R_i}(x, y, z) = \min\,(\mu_{A_i}(x), \mu_{B_i}(y), \mu_{C_i}(z)),$$

where A_i is a fuzzy set on the universe of a variable x; similarly, B_i and C_i are fuzzy sets, but not on the same universe.

When a set of such rules is available, a synthesised R can be obtained by composing each single fuzzy relation into

$$\mu_R(x, y, z) = \max_{i \in \{1,2,\ldots,n\}} \min\,(\mu_{A_i}(x), \mu_{B_i}(y), \mu_{C_i}(z)).$$

If the premises are known, e.g., variables x and y in the premises take values A' and B', respectively, then the value of the conclusion variable z is obtained by applying the compositional rule of inference [31, 129, 130]:

$$C' = (A' \times B') \circ R.$$

In the above, only a few basic aspects of the theory of fuzzy sets have been briefly outlined, however, they are necessary and fundamental for the development of the *FuSim* algorithm.

3.4. Summary

Since its birth, fuzzy set theory has been one of the major mathematical tools for representing uncertainty [125]. This chapter presents the basic motivations for the use

of fuzzy sets in developing a fuzzy qualitative simulation algorithm. As such, the approach gains the potential benefits from synthesising two originally separated and distinct approaches to handling complexity in modelling and reasoning about physical systems. The basic property of fuzzy sets, i.e., the graded memberships, significantly reduces (if not completely removes) the difficulty in describing or interpreting the values of physical variables. The possibility distribution, directly associated with a fuzzy set, remarkably differs from traditional probability distributions, allowing non-statistical but subjective information to be modelled. A concise theoretical review of the basic aspects of fuzzy mathematics is provided within the chapter. The establishment of the fuzzy qualitative simulation algorithm will be presented in the following two chapters

CHAPTER 4

FUZZY QUANTITY SPACE AND MODELLING PRIMITIVES

4.1. Introduction

To qualitatively simulate the behaviour of physical systems, basic conventions for describing system variables and relationships between the variables have to be established first. For this purpose, as pointed out in chapter 2, two corresponding design choices exist, i.e., the quantity space and the modelling primitives. Clearly, different choices result in different reasoning techniques. Having discussed the motivations for utilising Fuzzy Sets within *QR* in the last chapter, this chapter describes the fuzzy representation of physical quantities and operations allowed amongst such quantities. This leads to the development of an associated reasoning technique that provides a semi-quantitative extension to traditional qualitative simulation methods.

In response to these two design choices, the major contents of this chapter fall into the following two sections. Section 4.2 provides the definition of a fuzzy quantity space and discusses the properties of such quantity spaces relevant to performing qualitative simulation. Section 4.3 provides three different classes of qualitative operators that form the fundamental modelling primitives of fuzzy qualitative models, namely, the *algebraic, derivative*, and *function relational* operators. The way to use such modelling primitives and the resulting advantages of using them are demonstrated. A considerable part of the work described in this chapter has been reported in [90, 91, 92, 96, 98].

4.2. Fuzzy Quantity Space

The choice of representation of physical quantities plays a critical role in Qualitative Modelling. In fact, amongst the four different design choices for modelling and reasoning about physical systems (see section 2.2), the most important one is to decide on the form of representation of the system variables. All *QR* techniques describe quantities with a small set of symbols, the so-called quantity space whose elements are referred to as *qualitative values*, which are abstracted from the underlying field from which the variables of the physical system take values. This section presents the description of a particular type of quantity spaces within which a qualitative value is a normal and convex fuzzy number, forming the very basis for later development of the fuzzy qualitative simulation technique.

4.2.1. General Description

To provide a firm basis for a reasoning technique the quantity space employed must exhibit some basic characteristics. Three essential properties are therefore identified. The definition of the fuzzy representation of system variables is then given that holds these desired properties. A comparison between the fuzzy and non-fuzzy representations is made at the end of this sub-section, showing the flexibility of the former in interpreting physical quantities.

4.2.1.1. Desirable Properties and Basic Definition

As stated earlier, the introduction of quantity spaces reflects the distinctive feature of *QR* in that system variables take their values from a set of symbolically or qualitatively described values with associated semantics. Intuitively, this set of qualitative values should be *finite* with an adequate varying *granularity* and *cover* the whole field of interest. From this, three important properties that a useful quantity

space should possess can be drawn. They are called *finiteness*, *granularity*, and *coverage*, respectively, and expressed as follows:

Finiteness: each variable has a finite number of associated qualitative values, though under the different modelling conditions the number of the values used may differ from each other.

Granularity: if $x_1, x_2 \in R$ characterise 'similar things' or stand for 'similar properties' of a variable x, then the relevant qualitative values of x_1 and x_2 are equal to each other.

Coverage: all numerical values that variables may take are mapped onto their associated qualitative values with respect to the assumed granularity such that the complete set of the qualitative values covers the underlying field of interest.

Having described the obvious characteristics that a quantity space should have, we now exploit fuzzy numbers to establish a new class of qualitative values, enabling a semi-quantitative extension to the quantity representation of both magnitude and derivative of a system variable used in conventional *QR* techniques. The following is the definition of a fuzzy quantity space:

> A *fuzzy qualitative value* of a system variable is a fuzzy number chosen from a subset of normal convex fuzzy numbers. This subset is generated by an arbitrary but finite discretisation of the underlying numeric range of the variable. A set consisting of all the elements of such subsets, for all the variables in the system, is called a *fuzzy quantity space* and written as Q_F. The real number zero is required to belong to Q_F.

It is important to notice that, with this definition, a system variable takes values from a subset of a quantity space Q_F. This subset can be rather different from the

other subsets of the Q_F from which other system variables take values. Moreover, the magnitude and the rate of change of a variable can also have different sets of qualitative values. From one aspect, this allows a flexible representation of knowledge about systems since, if necessary, we can model a physical system with different detailed abstractions of its variables with respect to the extent of which we know about the variables.

As the fuzzy quantity space Q_F is generated by a finite discretisation of the underlying range of each variable of a system being modelled the Q_F will, of course, have the desirable properties of finiteness and coverage, as long as the system contains a finite number of variables. Granularity in the Q_F is obtained by the arbitrariness of the discretisation of the numeric ranges of system variables that are assumed to be of interest. Hence, we can translate a subset of a numeric range to one qualitative value according to what is needed in a particular modelling process, such that the extensions of a single qualitative intention may be rather different. Any Q_F that is formed according to the above definition will therefore inherently reflect these three characteristics.

4.2.1.2. Comparison with Crisp Quantity Spaces

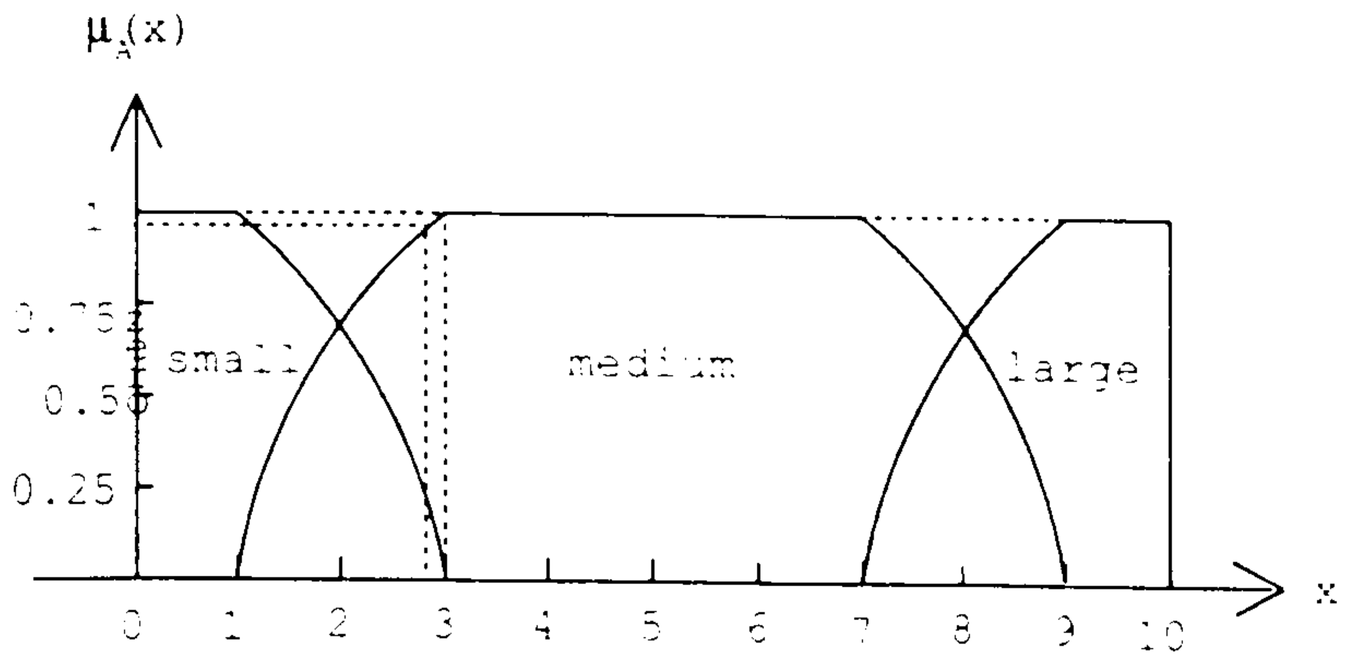
It is clear that all qualitative reasoners developed so far have utilised finite quantity spaces and their qualitative values can cover the underlying interesting fields. They also attempt to reflect different detailed abstractions of the description of system variables. For example, the simplest quantity space $\{+, 0, -\}$ in *Envision* [25], the seven-valued quantity space $\{-inf, -fin, -negl, 0, negl, fin, inf\}$ in *OMR* approaches (where, *inf* indicates infinite values, fin, finite values, and *negl*, negligible values) [15, 80, 114], and a group of real-valued 'landmarks' and open (crisp) intervals in *QSim*, all these quantity spaces reflect such objectives. This, of course, is

expected since finiteness, coverage, and granularity are the three fundamental requirements for designing a meaningful abstract representation of the variables.

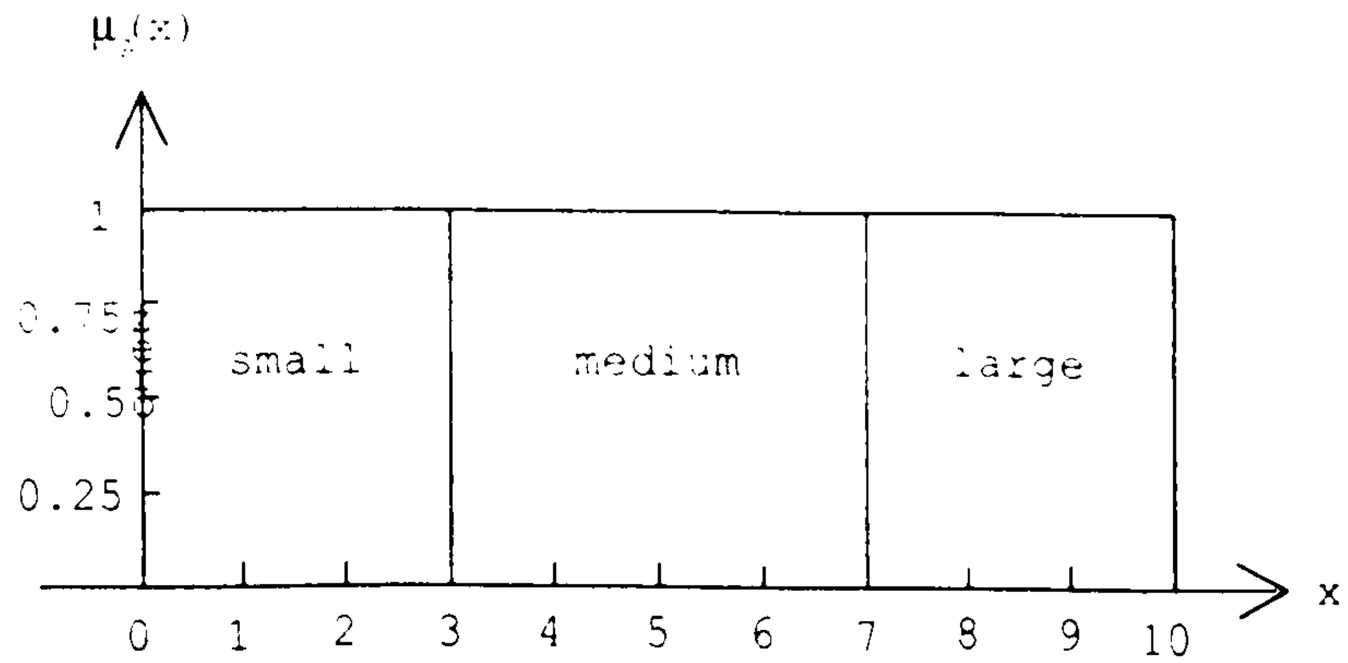
When considering granularity, however, the adoption of fuzzy sets offers a direct distinct advantage over the traditional crisp representations of the qualitative values. In fact, if we intend to describe the qualitative values of system variables only in terms of the crisp subsets of the underlying real range of the variables, the mapping from the real range to a quantity space will result in the search for the limits of the real numbers served as the boundaries between (disjointly) adjacent qualitative values within the quantity space. As analysed in chapter 2, this usually incurs severe difficulties in determining such limits. The fuzzy representation of qualitative values is more general than ordinary (crisp) interval representations, since it can represent not only the information stated by a well-determined real interval but also the knowledge embedded in the soft boundaries of the interval. Thus, fuzzy quantity space removes, or largely weakens (if not completely resolving), the boundary problem, achieved through the description of a gradual rather than an abrupt change in the degree of membership of which a physical quantity is mapped onto a particular qualitative value. It is, therefore, closer to our common sense intuition of the description of a qualitative value, especially when such descriptions are presented in linguistic terms [129, 130].

Let us investigate a simple example. By fuzzy representation, the underlying real range [0, 10], from which a physical variable takes values, can be mapped onto a set of qualitative values, say, {*zero*, *small*, *medium*, *large*}, as shown in figure 4.1.a, where each qualitative value A, actually a normal convex fuzzy number, has an associated linguistic term so that it corresponds to the perceived meaning. Within this quantity space, 3 belongs to *medium* with a strength or membership equal to 1.0 and

2.8 belongs to *medium* with a membership 0.93. Notice, however, that 2.8 also belongs to the set *small* with a membership 0.4. This non-exclusivity of values is an important aspect of fuzzy sets and, again, is important in capturing our common sense intuition for representing aggregated quantities, easing the interpretation of the underlying real values. However, using the crisp intervals, with the characteristic functions being $\mu_B(x)$, $B \in \{zero, small, medium, large\}$, as shown in figure 4.1.b, 3 belongs to [3, 7] named *medium* while 2.8 does not but fully belongs to the interval defined as *small*. Clearly, such a crisp representation would often result in non-intuitive interpretations.



(4.1.a)



(4.1.b)

Fig. 4.1. *Comparison between Fuzzy and Crisp Quantity Spaces*

In common with conventional qualitative simulation, three classes of operations

can be performed, i.e., algebraic, derivative, and function relational, on the fuzzy qualitative values within the fuzzy quantity space Q_F. However, the semi-quantitative description of quantity presented by the fuzzy numbers allows a much more flexible method for capturing functional information, allowing strength as well as sign information to be represented if indeed such information is available (and necessary). These three kinds of operation form the basis of the modelling primitives to be shown in the next section.

4.2.2. Four-Tuple Parametric Representation

The above definition of a fuzzy quantity space is given in a general form. Operations performed within such a quantity space, consisting of normal and convex fuzzy numbers with arbitrary forms of distribution, however, usually entail various types of computational difficulties. As a matter of fact, operations on fuzzy qualitative values are based upon the extension principle outlined in the previous chapter. This principle is invoked every time a particular operation is executed and requires extensive calculation. Also, the computational implementation of the calculation with arbitrary membership distributions of fuzzy numbers can only be done in a discrete domain obtained by sampling the original continuous distribution. The use of the extension principle with sampled membership distributions generates a considerable increase in the discrete samples of the result, and furthermore, only some of the resulting samples are correct. In order to reduce the computational complexity, more efficient ways to characterise fuzzy numbers must be exploited.

A number of techniques for solving this problem have been developed [7, 31, 49]. This usually involves a parametric approximation of the membership function. In particular, the membership distribution of a normal convex fuzzy number can be represented by the 4-tuple, $[a, b, \alpha, \beta]$, as shown in figure 4.2, and defined as

$$\mu_A(x) = \begin{cases} 0 & x < a - \alpha \\ \alpha^{-1}(x - a + \alpha) & x \in [a - \alpha, a] \\ 1 & x \in [a, b] \\ \beta^{-1}(b + \beta - x) & x \in [b, b + \beta] \\ 0 & x > b + \beta. \end{cases}$$

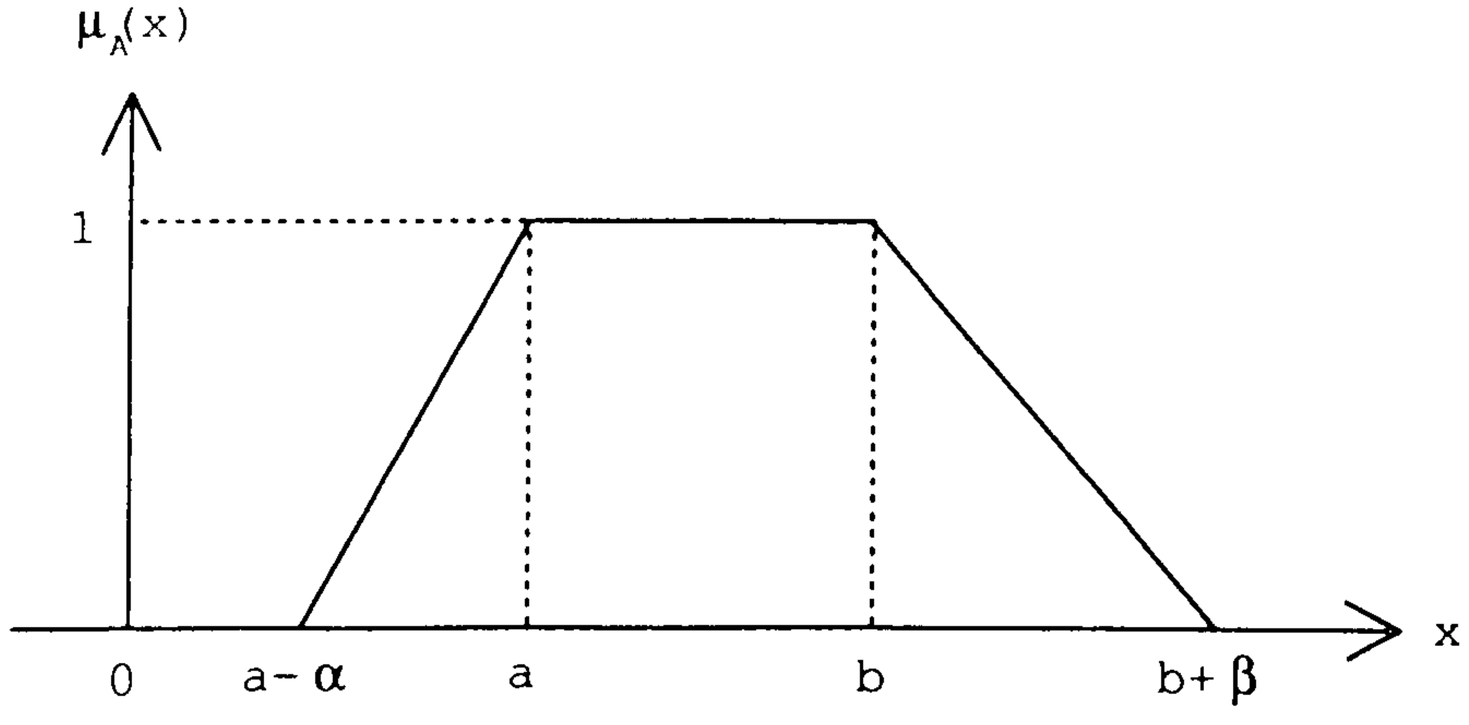


Fig. 4.2. *Parametric Representation of a Normal Convex Fuzzy Number*

The arithmetic operations on this kind of fuzzy numbers are well-developed [7] and easy to implement. We adopt such a representation to form the fuzzy quantity space for the fuzzy qualitative simulation. That is, from now on, by a fuzzy qualitative value we mean a 4-tuple parameterised fuzzy number. Apparently, using a specific quantity space containing only 4-tuple qualitative values does not affect the general definition of the fuzzy quantity space presented in the last sub-section or violate the three desirable properties.

A fuzzy quantity space formed in this way makes it possible to build a bridge between 'sets' and 'values' because this representation allows a real number, a real interval, a strict fuzzy number, and a fuzzy interval to be uniformly described. A strict fuzzy number is a fuzzy set whose nucleus (informally, those elements whose degrees of membership within the set are equal to 1) is a single real number and a

fuzzy interval is that whose nucleus is a real interval. Thus, the qualitative category representation and the ordinal representation can be combined in a natural way. For instance, the real number 4 can be denoted by a real interval [4, 4], which, in turn, can be represented by a 4-tuple fuzzy number [4, 4, 0, 0], whilst this fuzzy number is a special fuzzy subset of the real line. Similarly, the real interval [3.8, 4] can be represented by the fuzzy description [3.8, 4, 0, 0], and the strict fuzzy number '*approximately 4*' may be expressed by [4, 4, 3, 3]. In addition, the 4-tuple representation offers a great convenience in supporting the development of crucial techniques for detecting inconsistencies between the actual behaviour of a physical system and the simulated one from the system model within the model-based diagnostic system to be discussed in chapter 7. Also, the generality of the 4-tuple representation allows efficient modification of the level of abstraction of the quantity space.

4.3. Fuzzy Modelling Primitives

Our fuzzy simulation method adopts a constraint-centred ontology in that the model of a physical system is derived either from an underlying ordinary differential equation representation or from direct application of first order energy storage mechanisms [118]. We view each mathematical relationship holding against some system variables as a constraint that restricts the possible values that the variables may take. With respect to the quantity space defined in the preceding section, the design choice on modelling primitives, i.e., the constraints allowed, can now be specified.

To ease the discussion, an expression of the form

$$Q(z) = f(Q(x), Q(y)), \qquad Q(x), Q(y), Q(z) \in Q_F,$$

is called a *relevant constraint* on the system variables x, y, and z; where x and y are

called *constraining variables* which may be identical to each other and z is called a *constrained variable*. Both constraining and constrained variables take values from the fuzzy quantity space Q_F, and these values are denoted by $A(x)$, $B(y)$, and $C(z)$, respectively. As already mentioned, there are three classes of constraints possible for building a system model, namely, the algebraic, the derivative, and the function relational constraints; each of which is described in the following sections. Further discussions about modelling a physical system will be addressed in section 7.3.2, where *modelling dimensions* for building up a space of multiple models of the system are identified.

4.3.1. Algebraic Constraints

Within the general notation of constraints shown above, when f is an algebraic operator, the resulting constraints are termed arithmetic ones. The basic arithmetic operations on the fuzzy qualitative values in Q_F are those in the set of fuzzy numbers. The following presents the computational methods for such calculi with simple examples.

4.3.1.1. Operations

The formulae for the associated arithmetic operations are listed in Table 4.1, where $m = [a, b, \tau, \beta]$, $n = [c, d, \gamma, \delta]$, and $<_0$ is the partial order $<_\alpha$ when $\alpha = 0$. This partial order $<_\alpha$, within the set of 4-tuple fuzzy numbers, is defined such that for $A, B \in Q_F, A \neq B$, we say A is α-less than B, $A <_\alpha B$, if and only if $a < b$, $a \in A_\alpha, b \in B_\alpha$ with A_α and B_α being the α-cuts of A and B, respectively.

Formulae for Arithmetic Operations with Fuzzy Numbers		
Operation	Result	Conditions
$-n$	$(-d, -c, \delta, \gamma)$	all n
$\frac{1}{n}$	$\left[\frac{1}{d}, \frac{1}{c}, \frac{\delta}{d}(d+\delta), \frac{\gamma}{c}(c-\gamma)\right]$	$n >_0 0$, $n <_0 0$
$m + n$	$(a+c, b+d, \tau+\gamma, \beta+\delta)$	all m, n
$m - n$	$(a-d, b-c, \tau+\delta, \beta+\gamma)$	all m, n
$m \times n$	$(ac, bd, a\gamma+c\tau-\tau\gamma, b\delta+d\beta+\beta\delta)$	$m >_0 0$, $n >_0 0$
	$(ad, bc, d\tau-a\delta+\tau\delta, -b\gamma+c\beta-\beta\gamma)$	$m <_0 0$, $n >_0 0$
	$(bc, ad, b\gamma-c\beta+\beta\gamma, -d\tau+a\delta-\tau\delta)$	$m >_0 0$, $n <_0 0$
	$(bd, ac, -b\delta-d\beta-\beta\delta, -a\gamma-c\tau+\tau\gamma)$	$m <_0 0$, $n <_0 0$

Table 4.1.

It is important to realise that, for different fuzzy quantity spaces constructed by different subsets of the 4-tuple parametric fuzzy numbers the fundamental arithmetic operations performed within these spaces are of an identical form. That is to say, the basic algebraic operation system is fixed for any particular fuzzy qualitative simulation and, henceforth, does not depend on a particular quantity space used in the simulation. This, therefore, removes an important limitation existing in the *OMR* techniques, e.g., the *CHEPACHET* method (see section 2.3.1), where once the number of orders of magnitude changes the associated algebraic operation system has to be totally redefined.

As with the crisp quantity spaces of other qualitative simulation techniques there exists no inverse under the addition and multiplication operations within Q_F [75]. In

general, this presents difficulties in solving even simple fuzzy equations. This is the reason why we distinguish the constrained variable and the constraining variables in a relevant constraint. By constrained variable, we do not mean that there exists a causality between the variable and the constraining variables in the constraint, but rather that the directionality of solving a fuzzy equation is fixed. In fact, if we know the values of the constraining variables, we can obtain the value which the constrained variable should take by performing the operation on the values of the constraining variables. However, we cannot, in general, find an unknown value of a constraining variable by solving the equation as we can do in solving numerical equations (e.g., via transposition) due to the lack of inverse operations. Fortunately, by restricting a quantity space Q_F to be finite and closed, we can use the constraints to test the consistency of the sets of possible solutions. This is sufficient for achieving our present objective since, within the fuzzy simulation, constraints are used to filter the given sets of possible values that variables may take rather than to seek unknown values for some variables.

In order to ensure that a quantity space Q_F remains closed after certain operation has been applied to its elements, the approximation principle is utilised. This is required because of the fact that, when designing a quantity space, the assignment of the membership distributions of the qualitative values is often made by subjectively discretising the underlying real-valued range with respect to the desired properties of granularity and coverage. The resulting value from performing a particular calculation on the constraining variables, thus, may not be exactly equal to one of those values designed for the constrained variable. To describe the constrained variable in terms of predefined qualitative values, this principle is then necessary to be applied to that calculated value. This will become clearer when we present some examples later.

When using the approximation principle, distance measures play a central role in determining the degree of closeness between two fuzzy sets each of which belongs to a different subset of the same universe of discourse. For the present interest, the universe of discourse is the complete set of 4-tuple parametric fuzzy numbers. Let Q_F be one of these two subsets and the collection of all the possible results from the operations on the elements among Q_F be denoted by $\hat{Q}_F$. Clearly, the $\hat{Q}_F$ will also be a subset of the 4-tuple parametric fuzzy numbers as Q_F. Hence, we can choose the following as a distance measure [5]:

$$d(A, \hat{A}) = [(Power(A) - Power(\hat{A}))^2 + (Centre(A) - Centre(\hat{A}))^2]^{\frac{1}{2}},$$
$$A \in Q_F, \qquad \hat{A} \in \hat{Q}_F,$$

where, for 4-tuple parametric fuzzy numbers,

$$Power([a, b, \alpha, \beta]) = \frac{1}{2}[2(b - a) + \alpha + \beta],$$
$$Centre([a, b, \alpha, \beta]) = \frac{1}{2}[a + b],$$

which can be easily verified with respect to their definitions given in the previous chapter. The common coefficient, 1/2, on the right hand side of the above two expressions can be omitted when substituting these expressions into the distance expression.

Once we have a desirable distance measure, the approximation of a fuzzy number $\hat{A}$, $\hat{A} \in \hat{Q}_F$, to a qualitative value A in Q_F can be determined by choosing A such that the distance between $\hat{A}$ and A is the smallest among all the distances between the fuzzy number $\hat{A}$ and all the elements in Q_F that intersect with it. To reduce unnecessary calculation we first check if $\hat{A}$ intersects with any A. If so, the distance between $\hat{A}$ and A is calculated, otherwise A cannot be an approximation to

$\hat{A}$. In the case where there are more than one value in Q_F which have the same shortest distance from $\hat{A}$, all such values are treated as the approximation results of the original calculation.

It is important to indicate that, to avoid missing possible values that a variable may take, the most conservative attitude has been taken in finding the distances between a calculated value and the predefined values. That is, once a value defined in the quantity space intersects with the calculated their distance is evaluated. In normal cases only the value that results in shortest distance is considered. If two distance measures are similar, the associated values in the quantity space will both be treated as approximations. This has a significant implication in the simulation of the system behaviour. In fact, when there is a difficulty in distinguishing which of the generated behaviours reflects the actual behaviour of the system, we would rather leave them both as possible behaviours than delete one that might be correct even though it has a slightly larger distance. This assures that the simulated qualitative behaviours do include the underlying real behaviour.

4.3.1.2. Examples

Simple examples can illustrate how arithmetic constraints serve to limit a set of the possible values that a variable may take. Suppose that a system has three variables, x, y, z, and that they satisfy

$$Q(z) = Q(x) + Q(y).$$

For simplicity, it is assumed that these variables are normalised to have the same underlying numeric range [–1, 1] and take values from a fuzzy quantity space, Q_F = {*–large*, *–medium*, *–small*, *zero*, *small*, *medium*, *large*}, where each qualitative value is defined by a 4-tuple parametric, normal and convex, fuzzy number

$[a, b, \alpha, \beta]$ as follows, and shown in figure 4.3.

$$zero = [0, 0, 0, 0],$$

$$small = [0, 0.1, 0, 0.1],$$

$$medium = [0.3, 0.7, 0.2, 0.2],$$

$$large = [0.9, 1, 0.1, 0],$$

and $-[a, b, \alpha, \beta] = [-b, -a, \beta, \alpha]$.

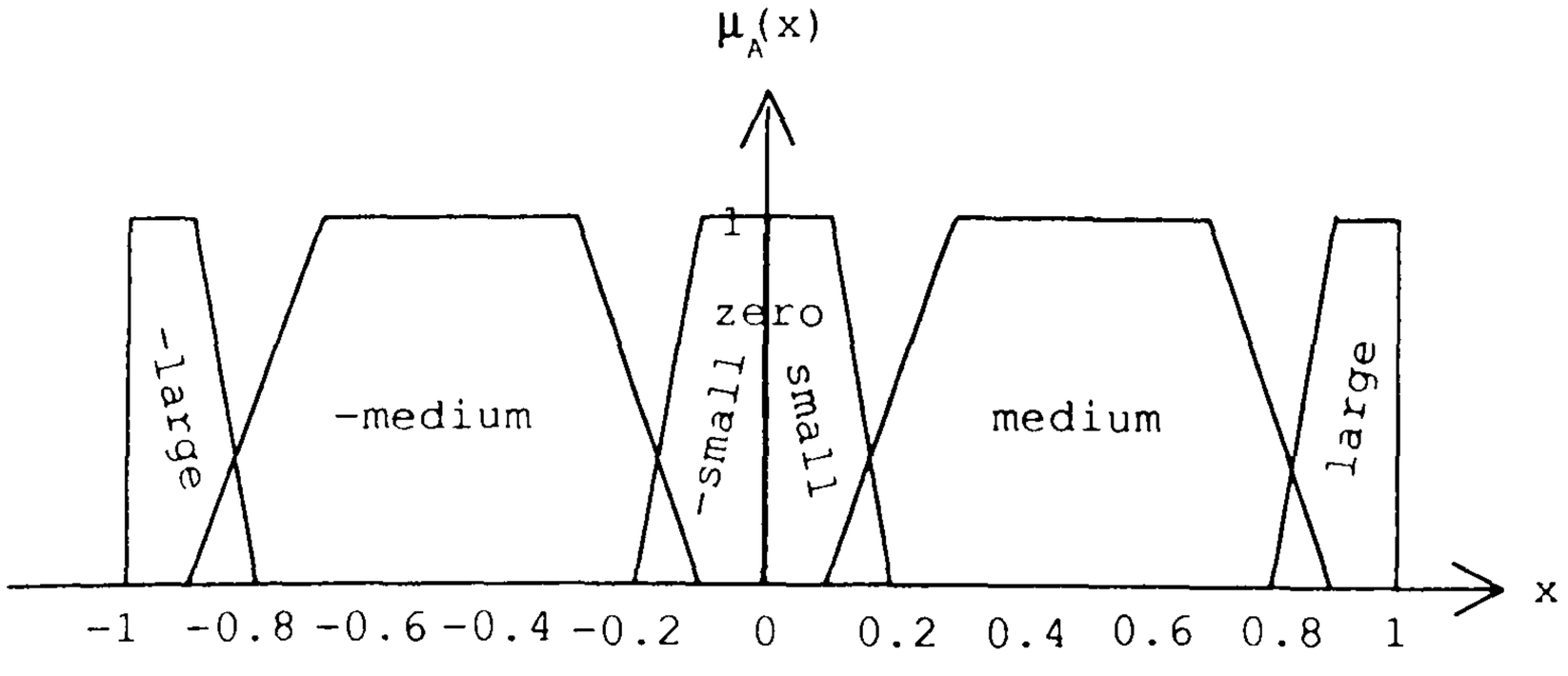


Fig. 4.3. *A Fuzzy Quantity Space*

If we know, at the beginning, that the values of the constraining variables x and y are $Q(x) = small$ and $Q(y) = large$, but the constrained variable z might be any value $Q(z)$ in the Q_F, then, through the relevant constraint, we shall have

$$Q(z) = [0, 0.1, 0, 0.1] + [0.9, 1, 0.1, 0] = [0.9, 1.1, 0.1, 0.1].$$

Since $[0.9, 1.1, 0.1, 0.1]$ does not belong to Q_F, the approximation principle is used to ensure Q_F remains closed. First of all, by checking if an element in Q_F intersects with $\hat{A} = [0.9, 1.1, 0.1, 0.1]$, the following set is generated,

$$_{\hat{A}}Q_F = \{[0.3, 0.7, 0.2, 0.2], [0.9, 1, 0.1, 0]\}.$$

Then, distances between $\hat{A}$ and A, $A \in {}_{\hat{A}}Q_F$, are evaluated and the results are:

$$D = \{1.17, 0.32\}.$$

Thus, the approximation of [0.9, 1.1, 0.1, 0.1] is [0.9, 1, 0.1, 0], based on the smaller distance 0.32. This results in

$$Q(z) = [0.9, 1, 0.1, 0] = large.$$

From the result above, the constrained variable z, which originally might be anything in the Q_F, now takes a single value by propagating the qualitative values of the constraining variables, x and y, under the addition constraint. Since the three variables have the same underlying numeric range, the result,

$$small + large = large,$$

is generated, and is well-suited to our common sense calculus. Importantly, this sets up a firmer basis for performing order of magnitude reasoning. In fact, when reasoning with order of magnitude information, conclusions can be obtained by propagating qualitative values through algebraic constraints, thereby avoiding problems of being obliged to introduce arbitrary limitations on the chaining of the inference rules.

If the variable x takes a qualitative value $-small$ instead of $small$, then

$$Q(z) = -small + large = [0.8, 1, 0.2, 0].$$

As with the case where $Q(z) = small + large$, a set ${}_{\hat{A}}Q_F$ is generated, by collecting all the elements in the Q_F which intersect with $\hat{A} = [0.8, 1, 0.2, 0]$, such that

$${}_{\hat{A}}Q_F = \{medium, large\}.$$

The distances between $\hat{A}$ and A, $A \in {}_{\hat{A}}Q_F$, are:

$$D = \{1, 0.32\}.$$

Hence,

$$-small + large = large.$$

Actually, compared with the predefined quantity space, it is clear that the best approx-

imation of the computed value [0.8, 1, 0.2, 0] is [0.9, 1, 0.1, 0]. This is illustrated in figure 4.4, where the thicker line stands for [0.8, 1, 0.2, 0]. This simple instance further demonstrates that the ambiguity with regard to conventional sign algebra is significantly reduced with an extended quantity space and the desirable objective in performing common sense calculus has been attained. Also, from one aspect, it conforms to the fact that the problem of order of magnitude reasoning can be automatically solved by propagating qualitative values through fuzzy algebraic constraints.

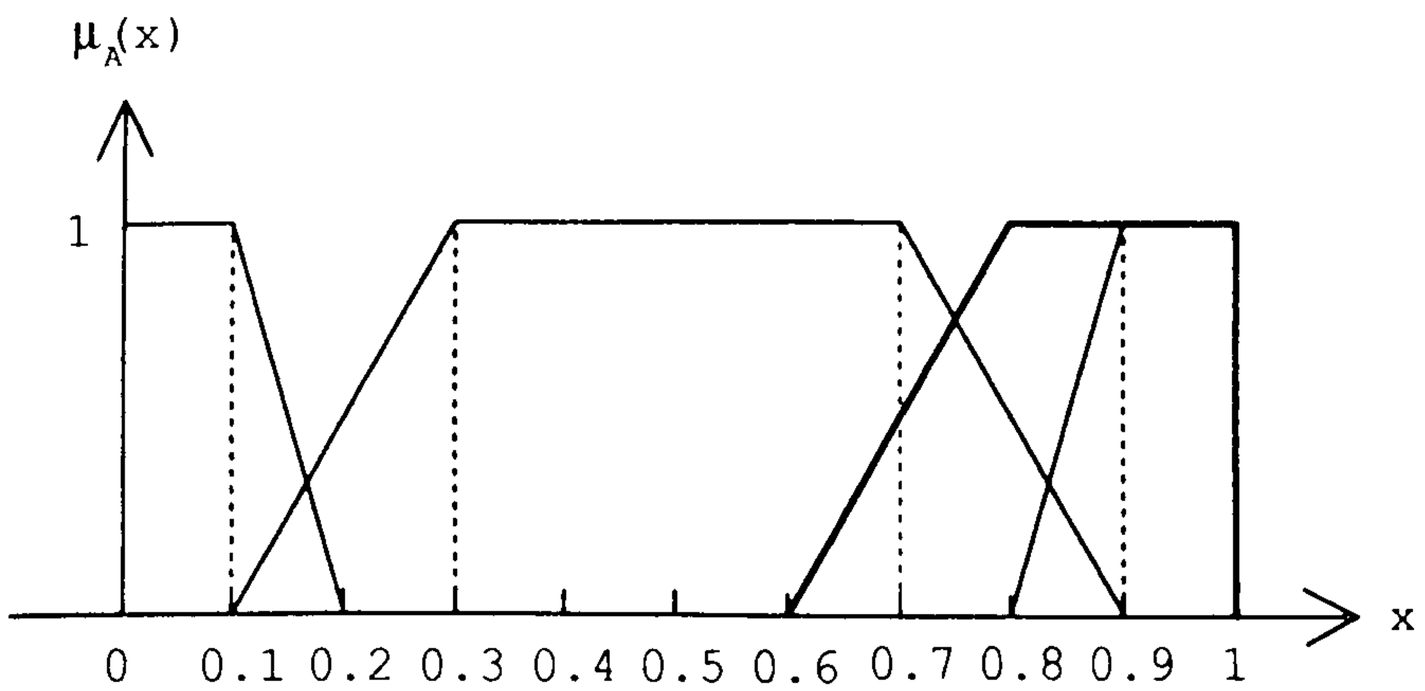


Fig. 4.4. *The Utilisation of the Approximation Principle*

It is apparent that, in general, for different quantity spaces a (identical) constraint imposes different restrictions on the values that the constrained and constraining variables may take. For example, let us consider the same arithmetic constraint used above and a different quantity space shown in figure 4.5. Starting with the same given condition, namely, $Q(x) = -small$ and $Q(y) = large$, we have

$$Q(z) = -small + large = [0.7, 1, 0.3, 0].$$

The set ${}_{\hat{A}}Q_F$ whose elements intersect with $\hat{A} = [0.7, 1, 0.3, 0]$ and belong to Q_F can then be produced as the following:

$${}_{\hat{A}}Q_F = \{small, medium, large\}.$$

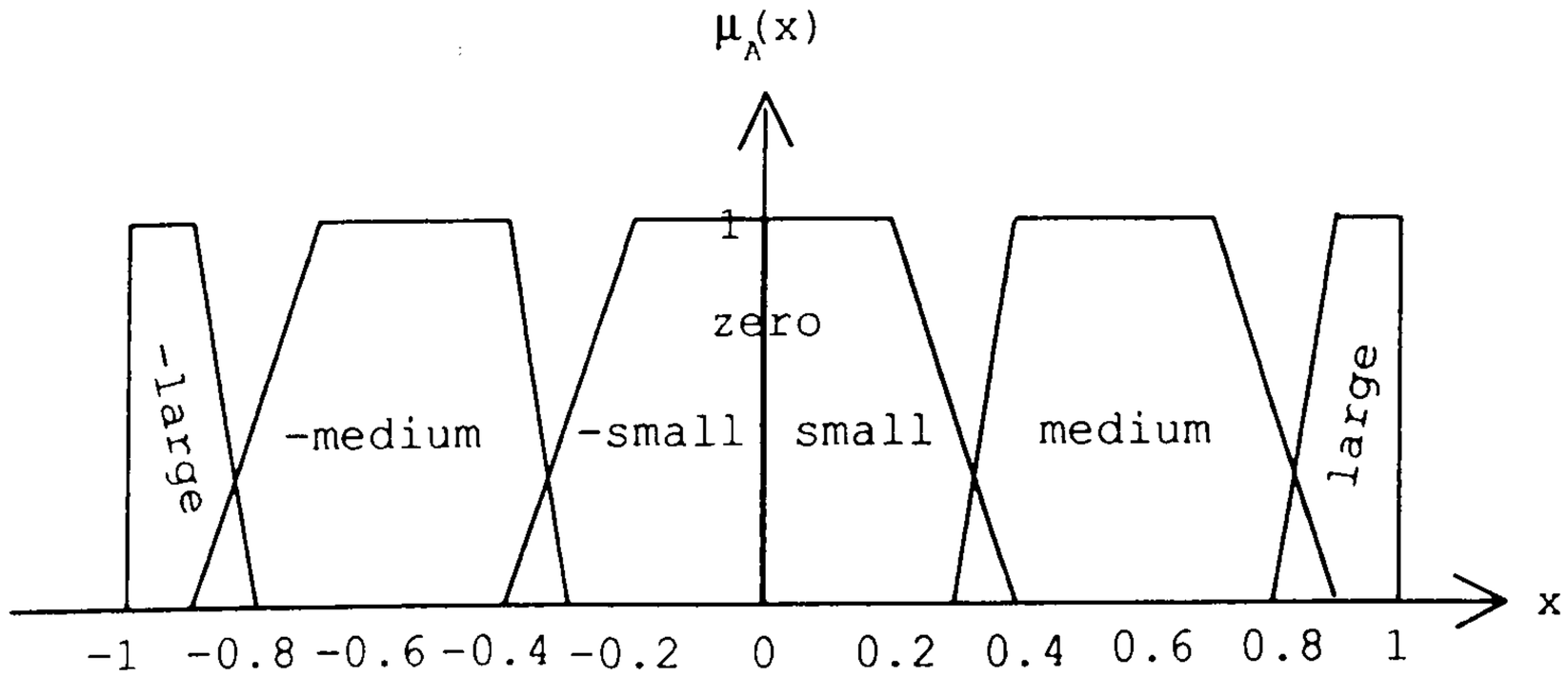


Fig. 4.5. *A Fuzzy Quantity Space*

Also, the corresponding set of the distances between $\hat{A}$ and A, $A \in {}_{\hat{A}}Q_F$, is:

$$D = \{1.53, 0.6, 0.63\}.$$

Thus, in response to the shortest distance, a result different from that previously obtained is found such that

$$-small + large = medium.$$

This is not surprising because the quantity space utilised herein differs from that employed earlier. Although the same linguistic terms have been used to describe the qualitative values in both quantity spaces, their respective underlying numerical ranges are actually rather different.

It is worth emphasising again that, if some distances computed appear similar to each other and are much smaller than the others, their associated qualitative values in the Q_F can be treated equally during a simulation process when required. In which case, those qualitative values with slightly larger distances will not be filtered by the relevant constraints in checking the consistency of a set of values assigned to the variables within the constraints. Relating to the example above, this means that, if we could not tell the difference between the two distances 0.6 and 0.63, the qualitative

values associated with these distances, namely, *medium* and *large*, can both be seen as the result of the calculation *–small + large*. Even in this case, the qualitative ambiguity embedded in the sign algebra has also been considerably reduced.

For efficiency reasons, when considering a specific application where the fuzzy quantity space is fixed, the approximation principle can be utilised off-line to compile an operation table. As a consequence, what is needed on-line is actually an entry-matching in the table when the algebraic calculation is met. This significantly reduces the computation time of the simulation. Notice that, as pointed out before, when constructing such a table the fundamental algebraic operation system remains unchanged. What is different for each particular application is only the description of the fuzzy quantity space itself.

4.3.2. Derivative Constraints

As with any simulation language for dynamic systems a differential operator is essential for determining the transient behaviour of the system. Within the simulation of a physical system, it provides a memory operation that accounts for energy storage in the system. Without them, a model developed is restricted to static systems.

In the present case, a derivative constraint is of the form

$$Q(y) = deriv\ Q(x), \qquad Q(x), Q(y) \in Q_F .$$

Since we shall cope with qualitative magnitudes and qualitative rates of change in the same way within the fuzzy simulation, the above constraint simply requires that the qualitative value of the magnitude of the constrained variable y must be the same as that of the rate of change of the constraining variable x. Although the derivative constraints do not actually perform differential operations as in numerical simulation, they restrict the directions and rates of change that the relevant system variables may have.

Using the same representation for both magnitudes and rates of change has important implications and advantages over the simulation systems that use only sign information for the derivatives, for example *QSim*. Within the fuzzy qualitative model of a dynamic physical system the derivative of a variable is designed to be able to take any value on a prespecified subset of the Q_F, thereby, allowing ordering information on the rates of change to be described. However, it is necessary to emphasise that, although both the magnitude and the rate-of-change of a variable follow the same representation (in terms of 4-tuple fuzzy numbers) they may take values from rather different subsets of a quantity space. The resulting ordering information is to be utilised in the fuzzy simulation algorithm to derive temporal durations of system states and the possible transitions between the states and, also, to develop an effective temporal filtering technique. Further, as discussed in chapter 7, such generated temporal durations are crucial to the effective use of qualitative simulation in model-based diagnosis of dynamic systems.

4.3.3. Function Relational Constraints

Current qualitative simulation techniques model functional dependencies as monotonically increasing or decreasing functions, such that the sign of the rate of change of a variable is the same as, or contrary to, the sign of the rate of change of another variable. These functions allow partial knowledge of the relationship between variables to be represented. However, for many applications, such functional relationships are often too weak as they do not allow any information on the strength of a relationship that is known to be utilised. Although so-called corresponding values, i.e., a set of real numbers which are respectively taken by a set of variables linked by a single constraint at the same time point, are used to further depict relationships amongst the variables, such knowledge is very specific and has to be expressed in a

numerated way. It is important, therefore, to be able to represent function relational constraints in a more detailed form to significantly restrict the possible values of the variables.

The use of a fuzzy quantity space allows qualitative function constraints to be represented as fuzzy relations. This enables partial numerical information to be utilised within functional dependencies, thereby providing stronger descriptions of the influence of a variable on another than the simple monotonic operator when such knowledge is available, but without requiring a full analytic function. The operational mechanism on fuzzy relations is presented below with the capability of modelling semi-quantitative information on the functional relationships being demonstrated using an example.

4.3.3.1. Operations

As indicated in section 3.3, a functional relation between a constrained variable and constraining variables may be viewed as a set of logical rules. When this relation is only qualitatively known, fuzzy logical rules can be stated to implement Approximate Reasoning [31, 129, 130, 131]. An example of such a rule is: ' *if pressure error is positive large or positive medium, and if change in pressure error is negative small, then heat input change is negative medium,*' where *positive large, positive medium,* and *negative small,* the values of the constraining variables (pressure error and change in pressure error), and *negative medium,* the value of the constrained variable (heat input change), are all fuzzy qualitative values of a fuzzy quantity space used for modelling a particular system. Such rules are of the form:

$$\text{if } x \text{ is } A_i, \text{ then if } y \text{ is } B_i, \text{ then } z \text{ is } C_i, \qquad A_i, B_i, C_i \in Q_F,$$

which is a conditional proposition and, by translating it through the min operation, is

equivalent to the following fuzzy relation:

$$\mu_{L_i}(x, y, z) = \min\,(\mu_{A_i}(x), \mu_{B_i}(y), \mu_{C_i}(z)).$$

When a set of n fuzzy rules is available, the resulting relation L is the union of the n elementary fuzzy relations L_i, $i = 1, 2, \ldots, n$:

$$\mu_L(x, y, z) = \max_{i \in \{1,2,\ldots,n\}} \min\,(\mu_{A_i}(x), \mu_{B_i}(y), \mu_{C_i}(z)).$$

This method of translation and aggregation of the rules is intuitively justified as follows: given the two consistent and nonredundant rules,

$$\text{if } x \textit{ is } A, \textit{ then } y \textit{ is } B,$$

$$\text{if } x \textit{ is } \bar{A}, \textit{ then } y \textit{ is unrestricted};$$

where $\bar{A}$ means the negation of A and is defined by $\mu_{\bar{A}}(x) = 1 - \mu_A(x)$, we have

$$\mu_L(x, y) = \max\,[1 - \mu_A(x), \min\,(\mu_A(x), \mu_B(y))],$$

which is exactly the semantic truth function for the logical implication:

$$A \rightarrow B : \bar{A}\ V\ (A\ \Lambda\ B).$$

Within this implication, V and Λ denote the logic disjunction and conjunction, respectively, and $\rightarrow$ indicates the implication connective [129].

If the constraining variables x and y take fuzzy values A' and B', respectively, the fuzzy value C' of the constrained variable z is obtained by applying the compositional rule of inference [31, 129, 130]

$$C' = (A' \times B') \circ L,$$

or

$$\mu_{C'}(z) = \max_{x,y} \min\,(\mu_{A'}(x), \mu_{B'}(y), \mu_L(x, y, z)).$$

In general, the approximation principle is again needed to ensure that the fuzzy quan-

tity space Q_F remains closed. This kind of qualitative functional relationship reflects one of the distinct advantages of our fuzzy qualitative simulation. It allows constraining variables to be mapped onto constrained variable in a semi-quantitative way, making effective use, if needed, of as much information about the functional dependencies as is available.

By this method, a fuzzy functional constraint can significantly extend the unbounded representation of an ordinary qualitative functional constraint, $M^+(.,.)$ or $M^-(.,.)$, representing strength information on a functional dependency. In [53], a different representational framework has been reported based on a similar motivation, which resorts to two pure numeric linear functional relationships, so-called the upper and lower envelopes, to bound the (qualitative) monotonic functional dependency. However, this can only cope with a relation between two variables, whereas, a fuzzy relation can be used to denote a relation held among any finite number of variables, though, in the present application, we do not use a relation to deal with more than three variables. More importantly, being one of the three kinds of qualitative constraints amongst system variables, fuzzy relations are local to each simulation step in the course of a fuzzy qualitative simulation. On the contrary, the method used in [53] is applied to an individual complete qualitative behaviour obtained from the output of an accomplished qualitative simulation, this largely decreases the efficiency of the overall simulation process.

4.3.3.2. Modelling Example

As an example, consider a basic experimental apparatus consisting of two hold-up tanks which are coupled by an orifice, as depicted in figure 4.6.

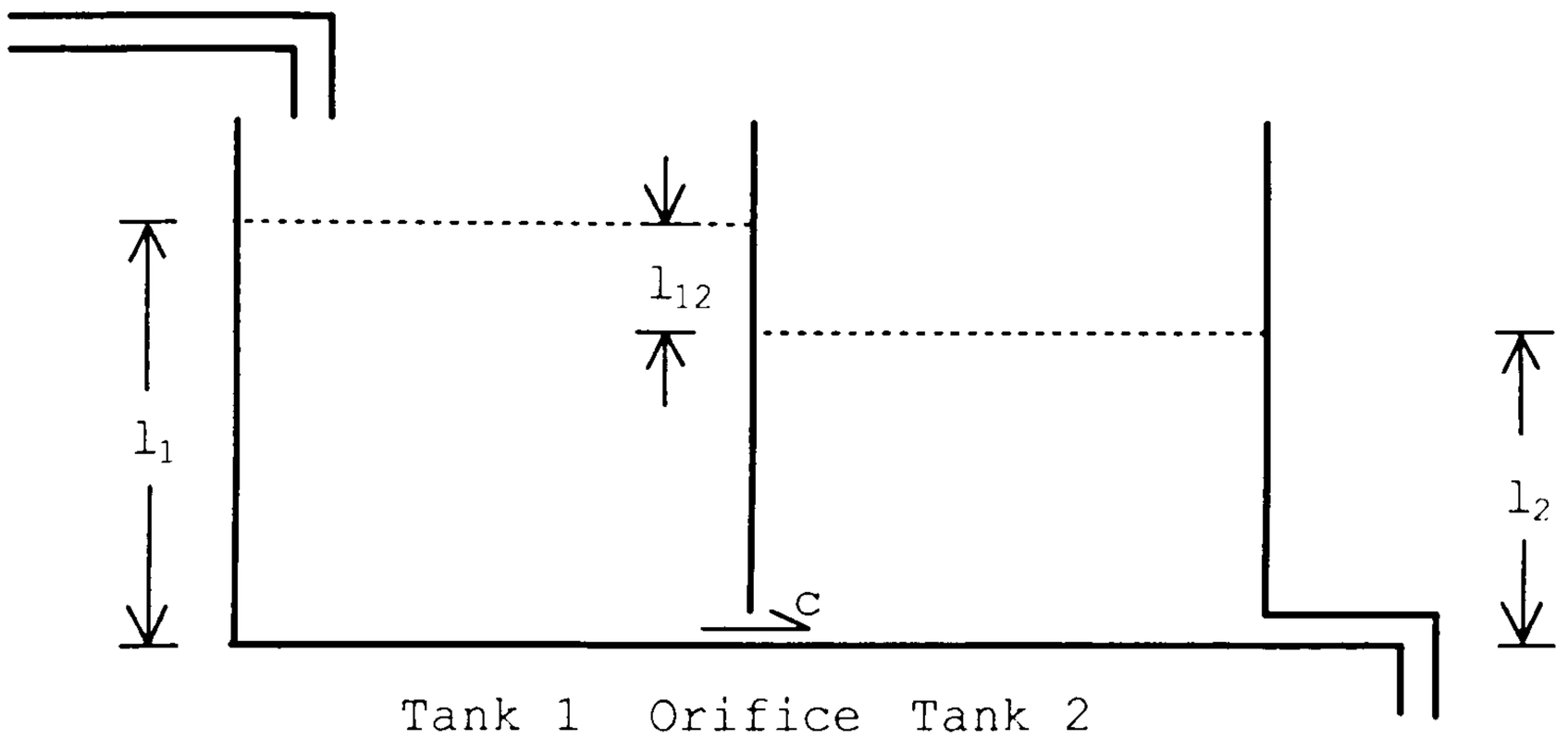


Fig. 4.6. *Two Coupled Tanks*

The characteristic equation for such an orifice is of the form

$$c = c_d a \sqrt{2g\ l_{12}},$$

where c_d is the discharge coefficient for the orifice, a is the cross-sectional area of the orifice, g is the gravitational constant, l_{12} is the level difference of the fluids in the two tanks, and c is the flow rate of fluid from tank 1 to tank 2. Clearly, this model describes a non-linear functional dependency between c and l_{12}. Also, the discharge coefficient c_d is subject to change because the coefficient is, in general, a function of the level difference l_{12} and the shape and length of the orifice. It is often difficult to get a precise numerical description for the characteristic equation, particularly in industrial settings. Actually, this is one of the basic reasons to investigate qualitative modelling for this experimental apparatus. Nevertheless, some typical characteristic curves can be obtained from experiments under assumed conditions, e.g., a sharp orifice, as shown in figure 4.7.

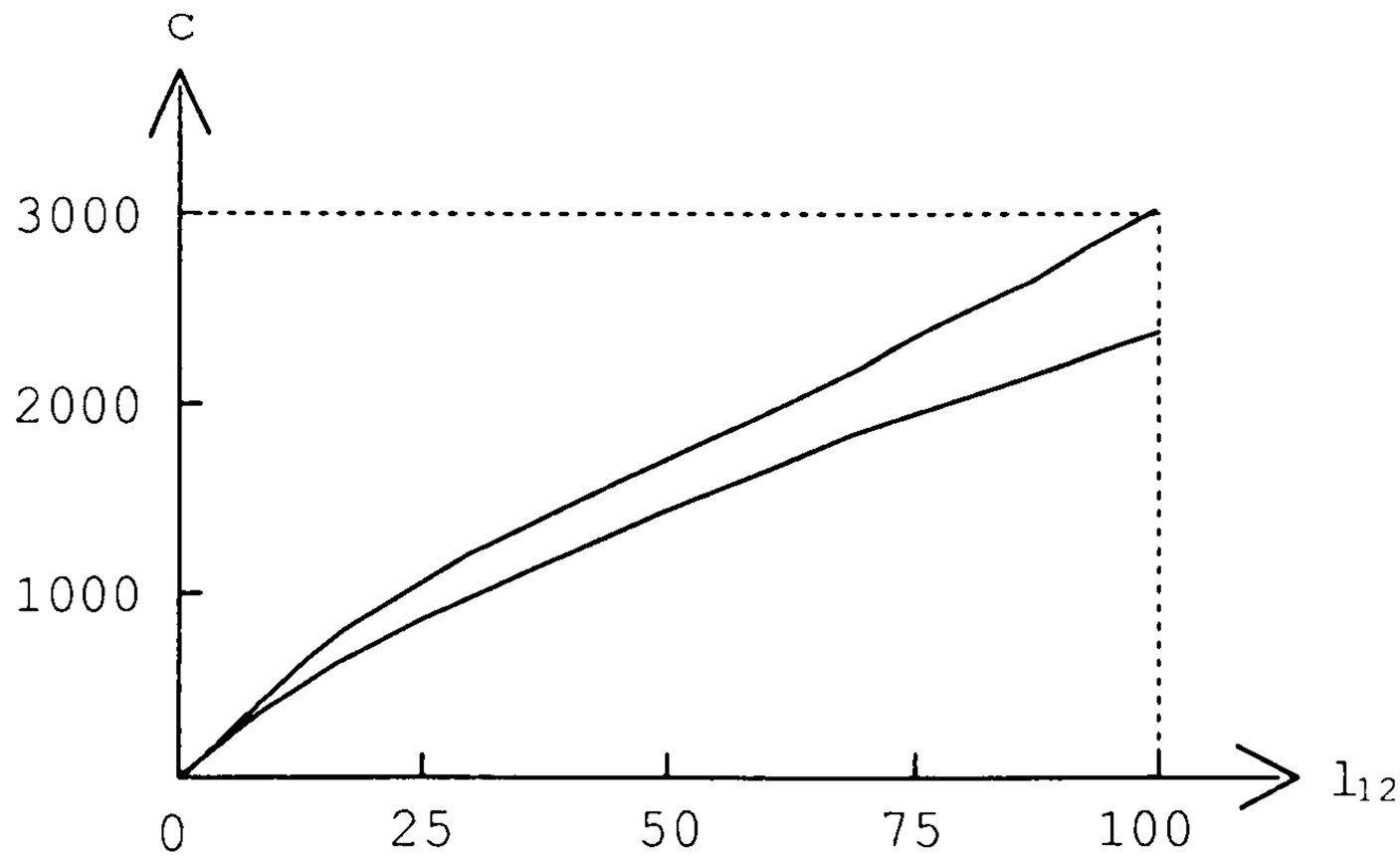


Fig. 4.7. *Two Characteristic Curves of the Orifice*

In conventional qualitative simulation systems, say, *QSim*, information on the characteristic equation for the orifice and the experiment curves can only be reflected by a monotonically increasing function $M_0^+(c, l_{12})$, or alternatively, modelled by

$$sign\left(\frac{dc}{dt}\right) = sign\left(\frac{dl_{12}}{dt}\right)$$

with a corresponding value (0, 0). However, if we use a fuzzy relevant constraint to qualitatively represent the functional dependency between c and l_{12}, on the basis of a fuzzy quantity space in which two subsets of qualitative values for the variables c and l_{12} are defined as in figure 4.8, we can have a set of logical rules reflecting the relationship such as

if l_{12} is *zero* then c is *zero*,

if l_{12} is *medium* then c is either *medium* or *large*,

if l_{12} is *large* then c is either *large* or *top* (*maximum*), etc.

These rules can be interpreted as a fuzzy relation according to the approach given in the previous sub-section.

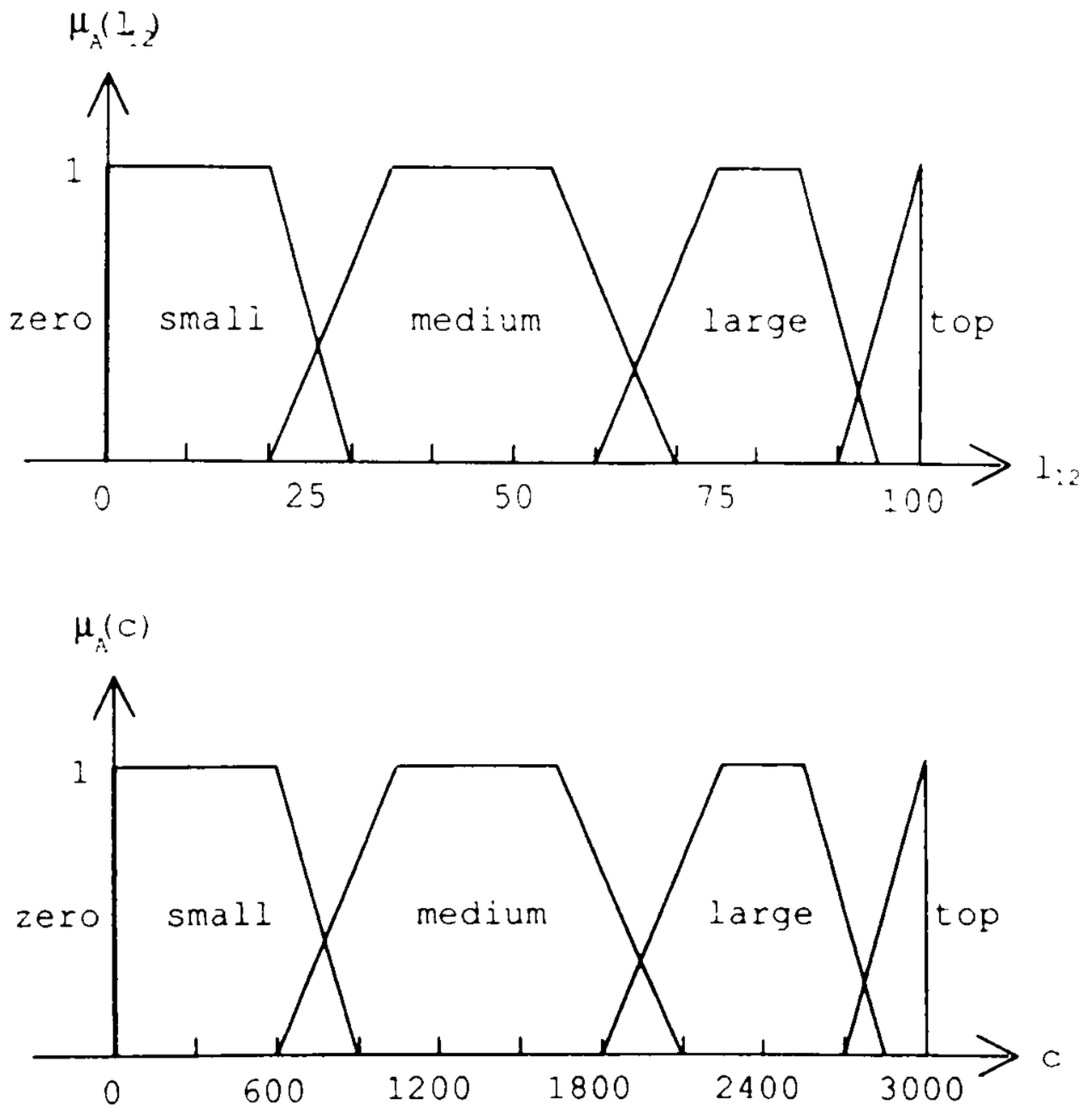


Fig. 4.8. *Two Subsets of Fuzzy Qualitative Values*

However, since in this example we model the fuzzy quantity space by the linguistic terms which are used in the rules, the fuzzy relation degenerates to a conventional binary relation, within the ranges of the magnitudes of l_{12} and c:

$$\begin{bmatrix} c \sim l_{12} & zero & small & medium & large & top \\ zero & 1 & 0 & 0 & 0 & 0 \\ small & 0 & 1 & 0 & 0 & 0 \\ medium & 0 & 0 & 1 & 1 & 0 \\ large & 0 & 0 & 1 & 1 & 1 \\ top & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Henceforth, when computing the fuzzy qualitative simulation, we can use this relation to rule out the impossible combinations of the qualitative values of the variables c and l_{12} by matching them with the matrix. For instance, a value pair

(*medium*, *small*) of (c, l_{12}) cannot co-exist but the pair (*medium*, *medium*) is possible with respect to this particular constraint.

More generally, if the above set of logical rules are translated and aggregated by performing the min and max operations on the qualitative values of the related variables, a particular combination of the values of the variables can then be checked by treating the compositional rule of inference as a constraint against the variables. Nevertheless, to act as a function relational constraint over the qualitative values within the fuzzy simulation, the degenerated form of the constraint is normally sufficient. This is because these qualitative values belong to the subsets of the fuzzy quantity space and, therefore, are bound to be known and modelled in the relation as those within the value set {*zero*, *small*, *medium*, *large*, *top*} of the variable c or l_{12} in the example (by chance l_{12} has the same value set as c here). In fact, those values to be checked by a function relational constraint result from only two possible ways: one is from applying the 'possible state transition rules' to the current qualitative values of the constrained and/or constraining variables or; another is from manipulating other constraint operations which share certain variable(s) with this constraint within the quantity space with the 'approximation principle' being used afterwards. Informally, the state transition rules simply indicate that a variable may possibly take which new qualitative values from the quantity space. This will become clear in the next chapter.

4.4. Summary

Within this chapter, the description of a fuzzy quantity space is given with the presentation of different operations that such a space supports. Three basic properties, finiteness, granularity, and coverage that every qualitative representation of physical quantities desires are pointed out. The fuzzy quantity spaces defined herein possess

these properties with a distinct advantage in easing the interpretations of the underlying real values into appropriately aggregated qualitative values. To avoid computational complexity the 4-tuple parametric representation of the fuzzy qualitative values is adopted which expresses both 'sets' and 'values' in an identical form. Three modelling primitives for developing a fuzzy model, i.e, the algebraic, derivative, and function relational constraints are defined. Simple examples are used to show the direct benefits from such fuzzy representation of the constraints over system variables: *OMR* may be automatically performed by propagating qualitative values through the arithmetic constraints and traditional monotonic relationships can be extended to cope with both sign and strength information by means of the function relational constraints. Based on such specified fuzzy quantity space and the fuzzy modelling primitives, the next chapter presents a fuzzy simulation algorithm that generates a qualitative description of the behaviour of continuous dynamic systems.

CHAPTER 5

FUZZY QUALITATIVE SIMULATION OF CONTINUOUS DYNAMIC SYSTEMS

5.1. Introduction

This chapter develops a fuzzy qualitative simulation algorithm based on the fuzzy quantity space and fuzzy modelling primitives described in the last chapter. The algorithm adopts the general approach to simulation taken by *QSim* but is significantly extended by the quantity space employed and the benefits that this representation offers. As already stated, the resulting *fuzzy qualitative simulation* algorithm is named *FuSim*.

Within the chapter, the notion of a qualitative state of a system variable is given first, in terms of a pair of qualitative magnitude and qualitative derivative, and then the transition between two qualitative states of a variable is discussed. Next, three filtering methods are described to restrict the value sets of the variables, namely, constraint filtering, temporal filtering, and global filtering. After this, an overall outline of the execution of the *FuSim* algorithm is shown and the experimental results of simulating two second-order systems are presented. A major part of this chapter has been reported in [90, 91, 92, 96, 98].

Like other qualitative simulation methods, fuzzy qualitative simulation requires making the fundamental assumption that the variables of the physical system being simulated are continuously differentiable functions, usually of time. We call this basic assumption the *continuity and differentiability assumption*. Based on this, *FuSim* takes as input a set of system variables with an associated fuzzy quantity

space, a set of constraints relating the variables (acting as the system model), and a set of initial values for the variables, and produces a tree of states with each path representing a possible behaviour of the system as output (see figure 5.1). This, of course, is similar to *QSim*, however, *FuSim* considerably reduces the set of spurious behaviours through the use of the fuzzy quantity space and associated filters. Furthermore, with a degree of freedom, *FuSim* generates a sequence of temporal durations coupled with the system's behaviour and, thus, gives an estimate of how long the system remains within a particular state and/or when a state transition occurs. This temporal information is extremely useful for practical applications and, in particular, is essential for the model-based diagnostic method to be established in chapter 7. The functional overview of the basics of the fuzzy qualitative simulation algorithm is depicted in figure 5.2 with each detailed function being explained in the following.

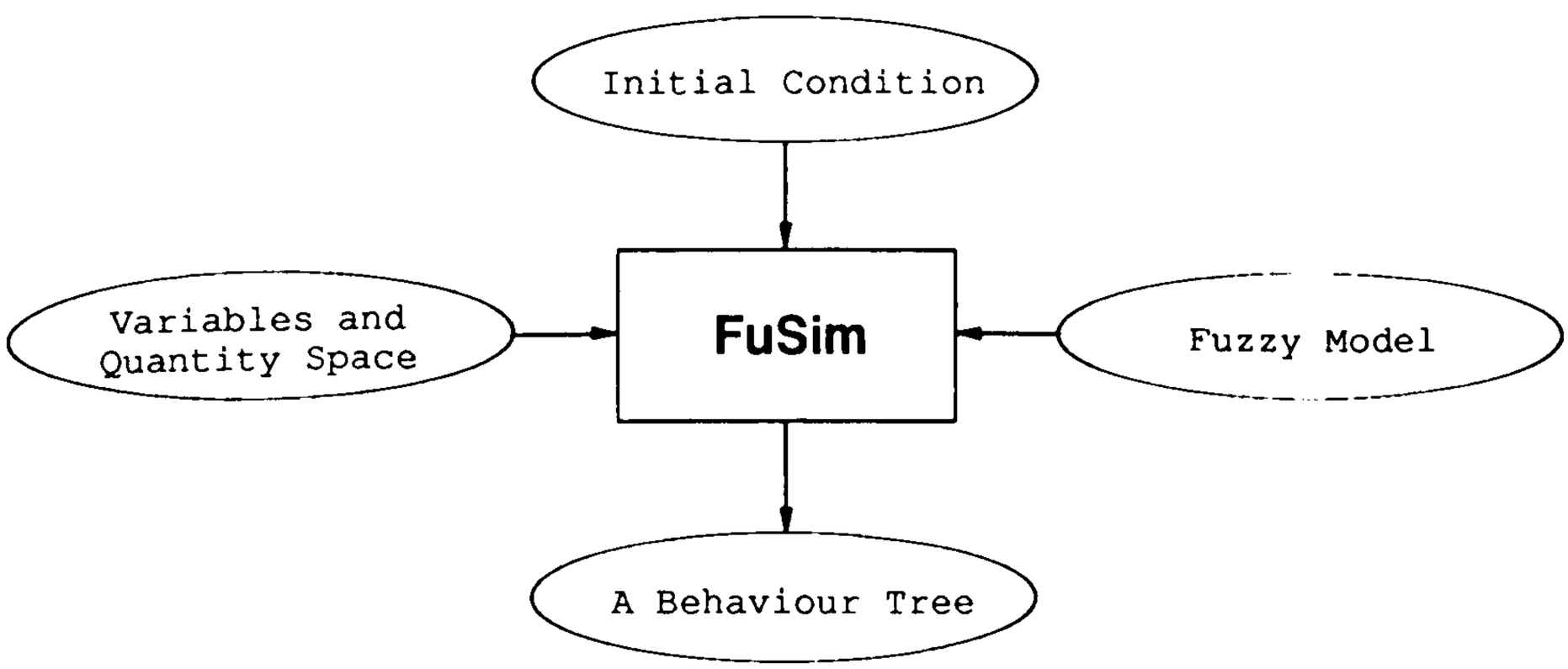


Fig. 5.1. *Input and Output of the FuSim Algorithm*

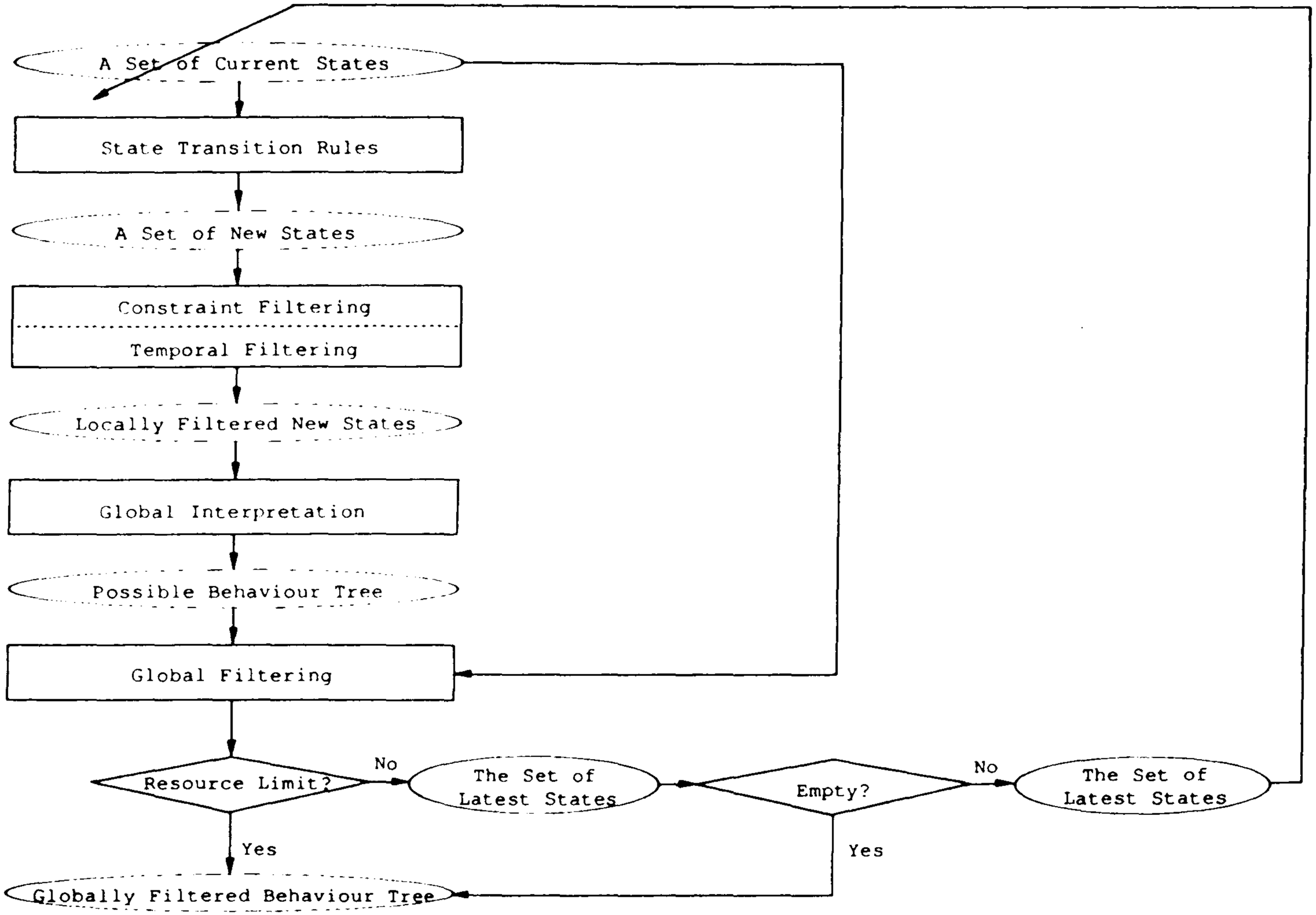


Fig. 5.2. *A Functional Overview of the FuSim Algorithm*

5.2. State Description

A physical system, in general, is characterised by a number of real-valued variables which vary continuously over time. To understand and reason about such a system, the concept of *state* is crucial. Actually, it is a set of different states at different time points or intervals that construct a system behaviour upon which we describe the system. As defined in *Modern Control Systems* [29], a system state possesses a knowledge of the system at some instant of time or during certain temporal duration (an instant or a duration is hereafter indiscriminately called a time scale) such that it obviates the need for any information about past behaviour of the system for predicting the future of the system. Clearly, the state of a system is determined if information on all the system variables is provided. Such information about a particular variable is therefore termed as the state of this variable at the time when the information is presented. By this, a system state is a conjunction of the states of all the system variables within the time scale shared by these variables. Henceforth, the following discussion will concentrate on the description of the states that system variables may take. For this purpose, the notion of the fuzzy quantity space, as a collection of qualitative values, is used in this section to define the *fuzzy qualitative state* of a system variable at a given time (interval).

5.2.1. Representation of a Fuzzy Qualitative State

For a given system variable x, the fuzzy qualitative state of x within a time scale ΔT_p, $QS(x, \Delta T_p)$, is a pair $<A, B>$, $A, B \in Q_F$, where A denotes the fuzzy magnitude and B the fuzzy rate of change of x, and Q_F is the fuzzy quantity space adopted for representing the system variables. Following the terminology in [113, 116] ΔT_p is called the *persistence time* of x in this fuzzy qualitative state.

It is important to point out that, within the above definition, although both A and B take values from Q_F it is not required that they obtain values from the same subset of the Q_F. In general, for the representation of the states of different variables the associated magnitudes can also have different value sets from each other and, of course, so can their rates of change. This offers an important flexibility in describing the states of variables and, henceforth, the entire system state with respect to the granularity required and/or the available knowledge of the variables.

A significant feature embedded in the description of a fuzzy qualitative state is the persistence time that records the amount of time a variable remains within a given qualitative state. Such a persistence time is inherently determined by the extent of the fuzzy magnitude and the fuzzy rate of change. Notice that the rate of change of a variable takes its values from a subset of Q_F rather than the set of three symbolic values $\{+, 0, -\}$. As such it permits ordering information on the rates of change and hence a measure of its variation with time. Figure 5.3 shows a simplified case in which each of the variables x, y, and z has the same interval-valued magnitude A, but their derivatives are real numbers B_x, B_y, and B_z with $B_x < B_y < B_z$, respectively. The durations for x, y, and z to persist within A can be depicted and the relation $\Delta T_x > \Delta T_y > \Delta T_z$ holds among them. This clearly shows the importance of having ordering information on rates of change to be able to determine a persistence time for a given state. Unfortunately, such a graphical representation is not possible when $A, B_x, B_y, B_z \in Q_F$.

It is worth indicating that, within conventional qualitative simulation techniques, the derivatives of system variables take values only from $\{+, 0, -\}$ and, therefore, no distinction between the rates of change (e.g., increasing rapidly or very slowly) can be made. A state sequence produced by such simulators describes the temporal relations

amongst the states only in a logical order, without giving an explicit measurement of the time scale within which a particular state remains. Although some temporal information may be deduced by exploiting those relationships that identify some of the variables to be the derivatives of others, there lacks a general mechanism to automatically perform such inferences. It might not be too difficult to implement this if the system has a single input and a single output and can be modelled in terms of a set of state variables with each being the derivative of another except for the output, i.e., phase variable representation. In such a case, the rate of change of each system variable will take values from the value set of the variable with one order higher derivative. This approach has been adopted within [3] in an attempt to produce temporal durations from rates of change information.

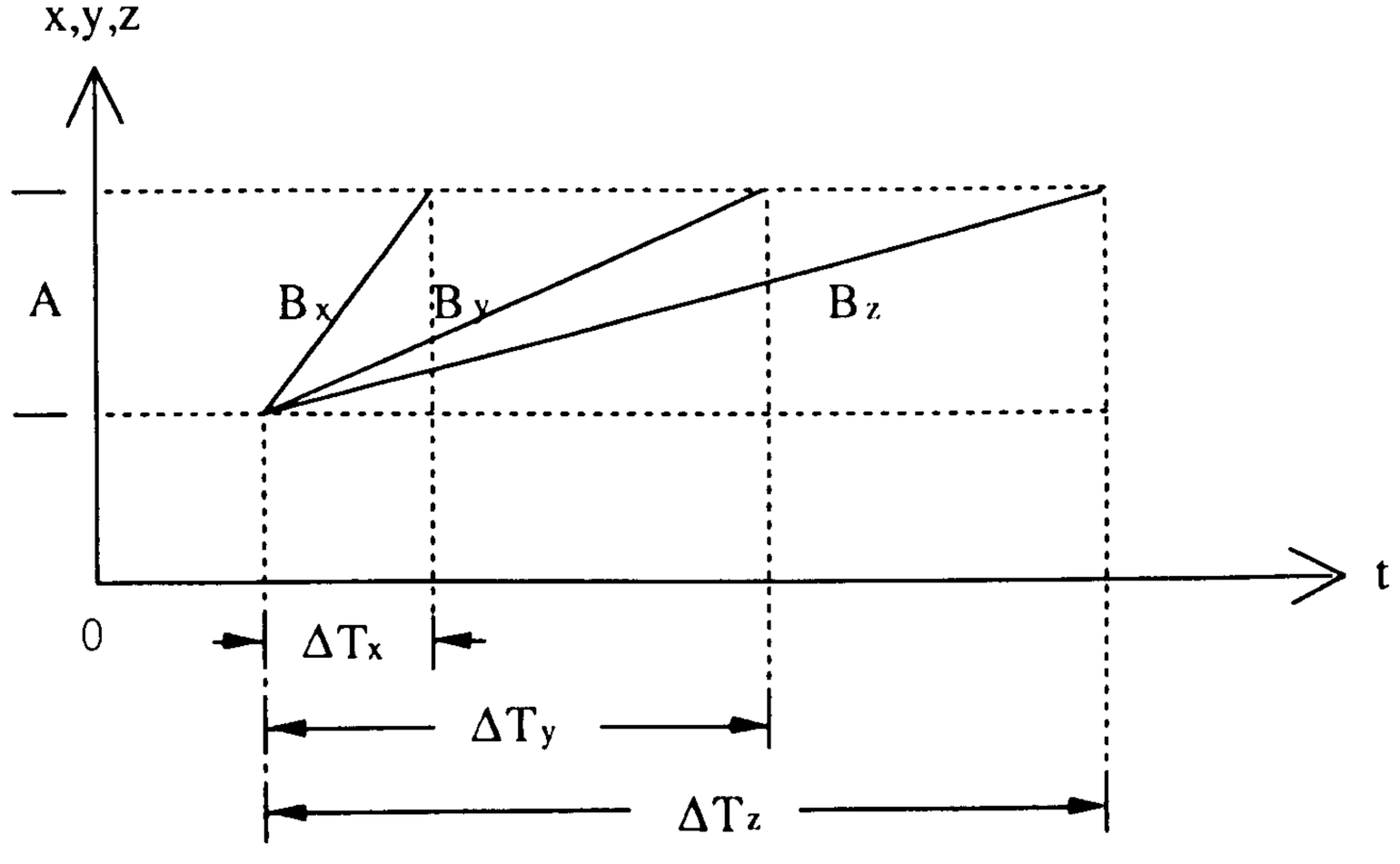


Fig. 5.3. *The Persistence Time of an Interval-Valued State*

5.2.2. Computational Method for Persistence Time

In principle, the persistence time of a fuzzy qualitative state should also be a fuzzy number since it expresses the time within which a fuzzy state exists [112]. However, using the *resolution identity* a (crisp) interval, within which a persistence

time may lie, can be calculated from the fuzzy magnitude A and fuzzy rate of change B by the following rules, where $W(.)$ expresses a plausible representation of the width of a fuzzy qualitative value and B_α denotes the α–cut of B with α being a degree of freedom:

(1) If $0 \Subset B_\alpha$, then $\Delta T_p \in \alpha \left[\frac{W(A)}{|B|}\right]_\alpha$, where $|B| = \begin{cases} B, & B >_\alpha 0, \\ -B, & B <_\alpha 0. \end{cases}$

(2) If $0 \in B_\alpha$, then ΔT_p is not well-determined, any length of time may elapse.

We use the length of the α-cut of a fuzzy number as a measure of its width, as shown in figure 5.4. It can be easily derived that the α-cut, A_α, of a qualitative value $A = [p_1, p_2, p_3, p_4]$ is the following

$$[p_1 + p_3(\alpha - 1), p_2 + p_4(1 - \alpha), 0, 0],$$

namely, a crisp interval

$$[p_1 + p_3(\alpha - 1), p_2 + p_4(1 - \alpha)].$$

We call this expression for finding the α-cut of a qualitative value the α-*cut formula*.

Based on this, we have,

$$W(A) = length(A_\alpha) = p_2 - p_1 + (1 - \alpha)(p_3 + p_4).$$

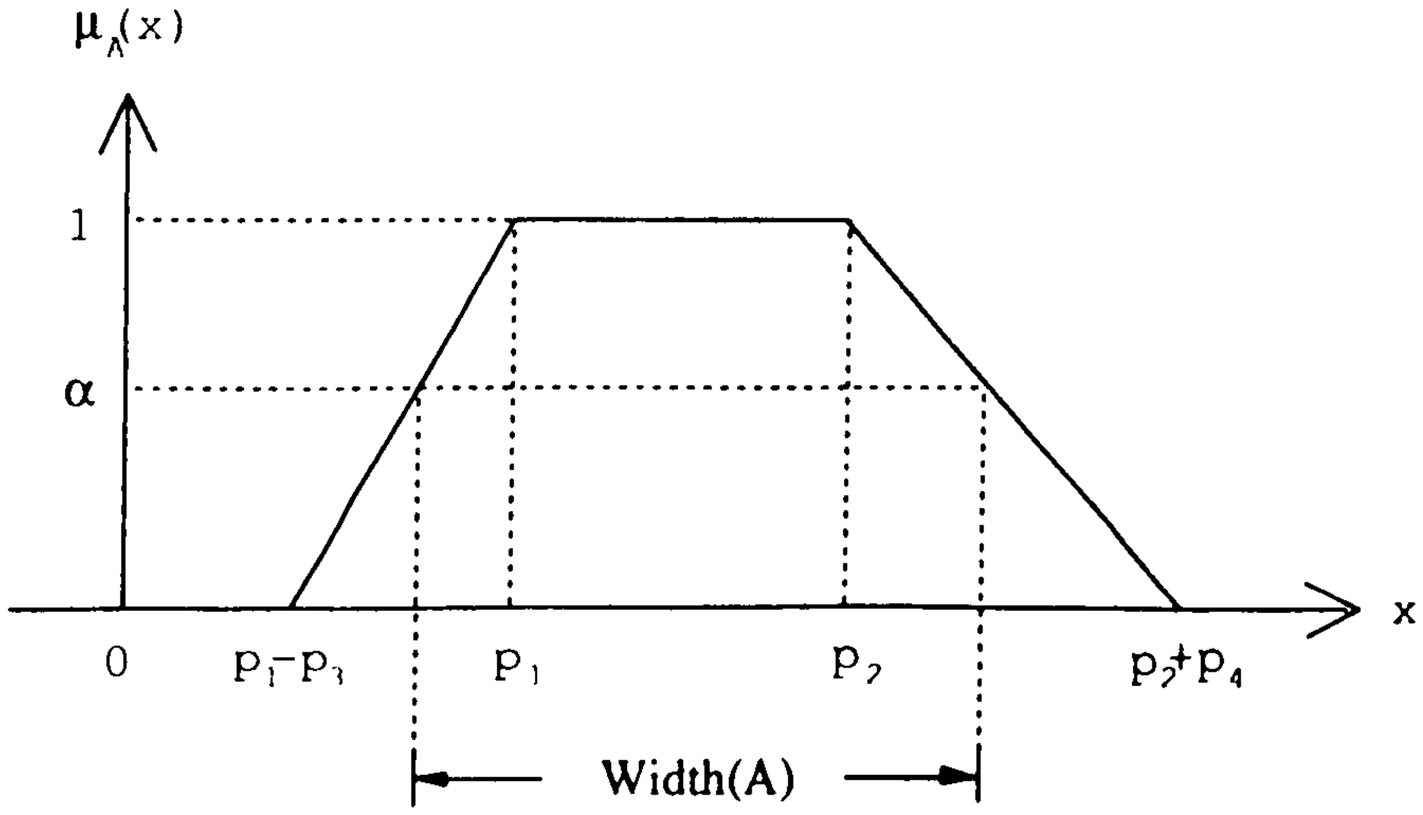


Fig. 5.4. *The Width of a Fuzzy Qualitative Value*

In the course of a particular simulation the value of the α used to calculate the width of fuzzy qualitative values is assumed to remain constant. This is intuitively justified by the requirement of treating all the qualitative values within a fuzzy quantity space equally. Therefore, the coefficient α in rule (1), introduced by the application of the resolution identity, can be omitted, and this rule is simplified to:

$$\text{If } 0 \Subset B_\alpha, \text{ then } \Delta T_p \in \frac{p_2 - p_1 + (1-\alpha)(p_3 + p_4)}{|B|_\alpha},$$

where, if $B = [q_1, q_2, q_3, q_4]$, the analytic form of the $|B|_\alpha$ is decided by

$$|B|_\alpha = \begin{cases} [q_1 + q_3(\alpha - 1), q_2 + q_4(1-\alpha)], & B >_\alpha 0, \\ [-q_2 + q_4(\alpha - 1), -q_1 + q_3(1-\alpha)], & B <_\alpha 0. \end{cases}$$

Within this equation, the upper expression is obtained by directly applying the α-cut formula to B and the lower one is obtained due to the fact that $-B = [-q_2, -q_1, q_4, q_3]$ and, again, by using the α-cut formula.

Suppose that we have a quantity space $Q_F = \{zero, small, medium, large\}$, described by figure 5.5, and choose $\alpha = 0.5$ to calculate the width of qualitative values. If we know a variable x stays in the state $<small, medium>$, and variable y in state $<medium, zero>$; then, from the first rule, the persistence time of x within its current state lies in

$$\left[\frac{W(small)}{medium}\right]_\alpha = \frac{0.25}{[0.35, 0.75]} = [0.33, 0.71].$$

For variable y, since its rate of change is *zero* it will, of course, remain within its current state forever unless there is some relevant constraint to force it to change. This is reflected by the second rule for computing the persistence time.

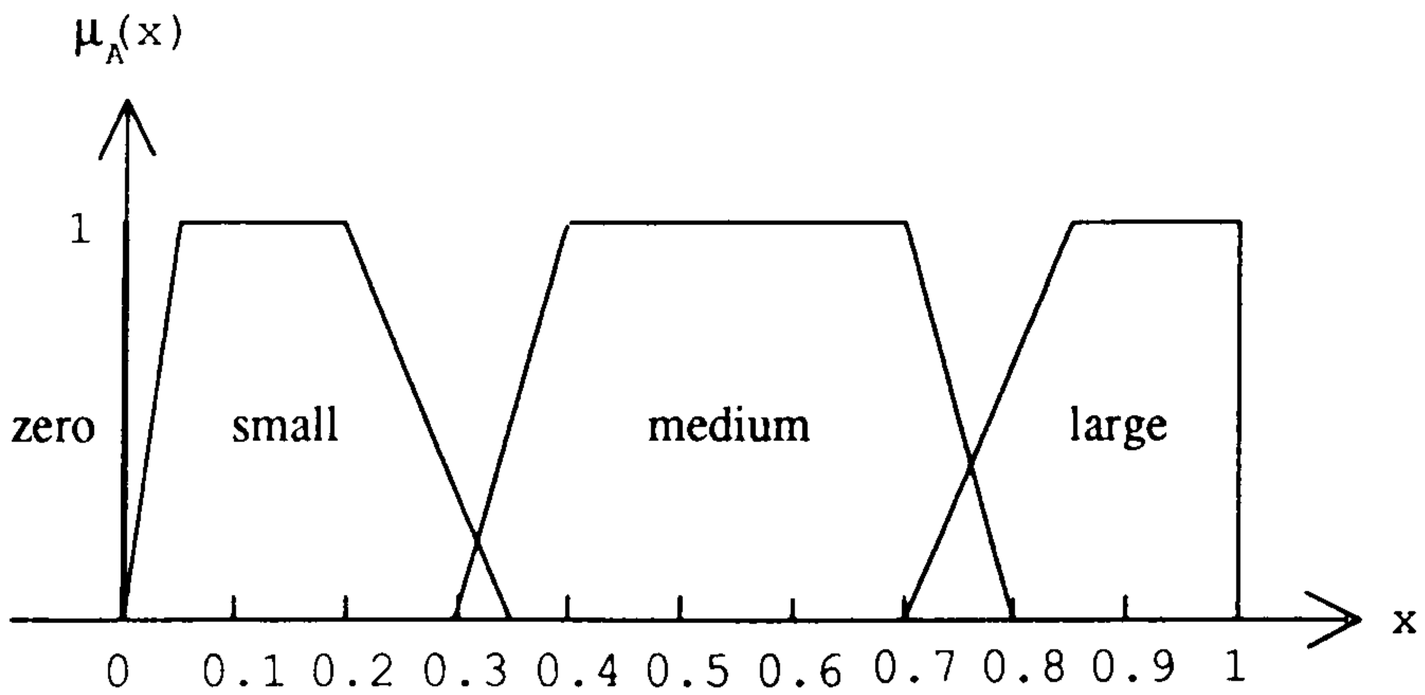


Fig. 5.5. *A Fuzzy Quantity Space*

Clearly, the persistence time obtained in this way presents a description of the amount of time within which a variable may remain in a particular state, although usually giving only a possible range. This is a major distinguishing aspect of *FuSim*, compared with traditional *QR* approaches.

5.3. State Transitions

The behaviour of a continuous dynamic system is reflected by a sequence of states that transition from one to another with respect to the temporal ordering amongst them. Having described the representational form of the fuzzy qualitative states, this section is dedicated to a discussion about the possible transitions between different states or, more concisely, about *state transitions*. By a state transition we mean that a system variable x changes from state $QS(x, \Delta T_{p_1}) = \langle A_1, B_1 \rangle$ to $QS(x, \Delta T_{p_2}) = \langle A_2, B_2 \rangle$, $A_1, A_2, B_1, B_2 \in Q_F$. Notice that, more generally speaking, A_1 and A_2 should belong to the value set of the magnitude of x or to a subset of Q_F, while B_1 and B_2 belong to that of the rate-of-change of x.

Four kinds of transitions are possible:

1) $A_1 = A_2, B_1 = B_2$ --- *Null-transition*;

2) $A_1 \neq A_2, B_1 = B_2$ --- *M-transition*;

3) $A_1 = A_2, B_1 \neq B_2$ --- *R-transition*;

4) $A_1 \neq A_2, B_1 \neq B_2$ --- *MR-transition*.

These transitions can be symbolically shown as in figure 5.6, where a solid circle indicates the fuzzy qualitative value of the magnitude of a variable, while an arrow indicates the fuzzy qualitative value of the rate of change of a variable, implying the changing direction of the variable. It is worth pointing out, however, that when the qualitative magnitude of a variable takes a real number as its value an R-transition is not possible. With these definitions on the possible state transitions, basic rules which govern such transitions are presented in the following sub-section and the time taken by each possible transition, an important characteristics resulting from the fuzzy state representation, is discussed in section 5.3.2.

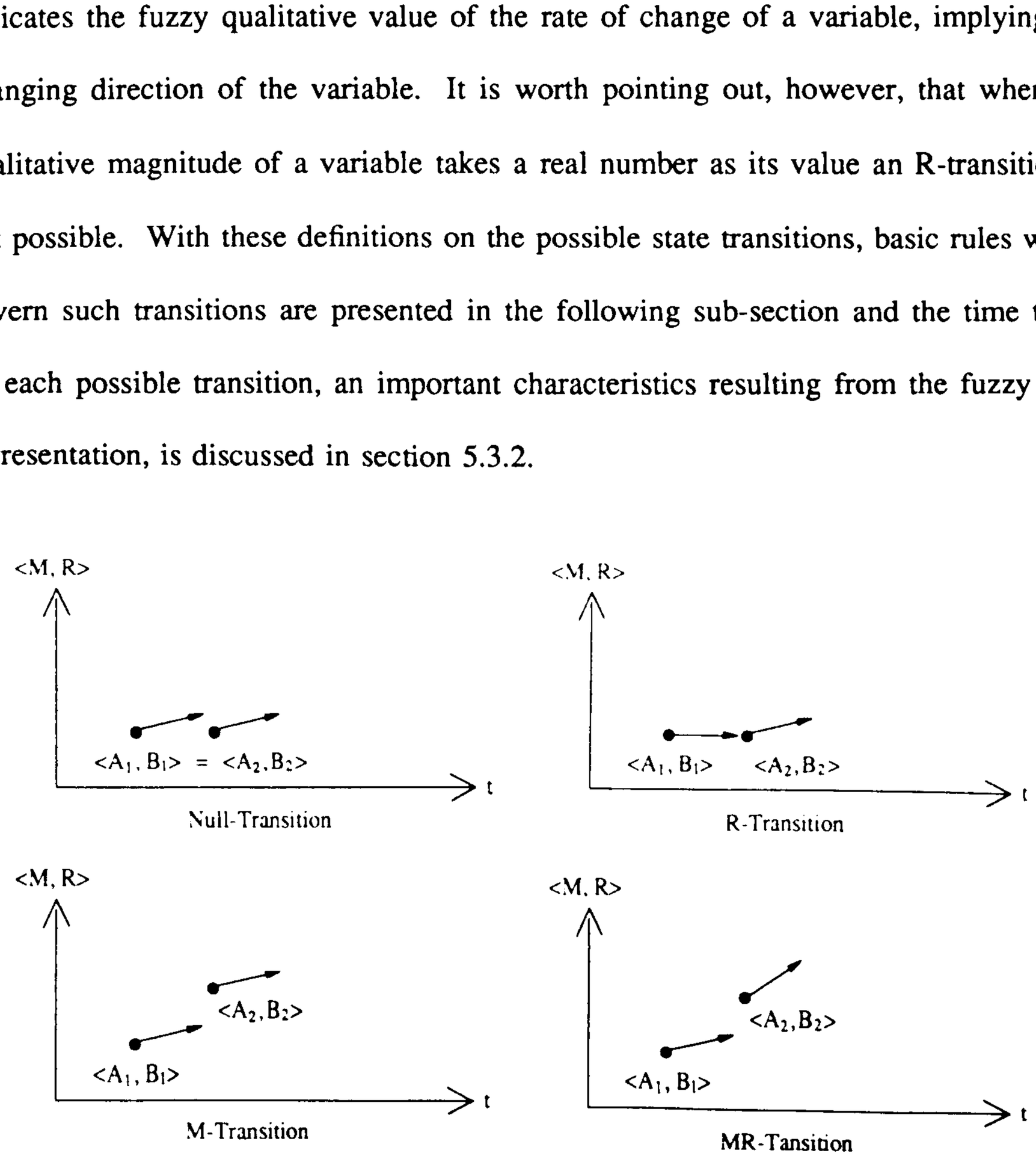


Fig. 5.6. *Four Kinds of Transitions*

5.3.1. State Transition Rules

On the basis of the continuity and differentiability assumption, possible state transitions, say, for a variable x to change from $QS(x, \Delta T_{p_1}) = <A_1, B_1>$ to $QS(x, \Delta T_{p_2}) = <A_2, B_2>$, can be represented by a set of rules. In common with other qualitative simulation approaches [15, 52, 114] these are called the *possible state transition rules*. Within these rules, '$<_\alpha$' is the partial order defined in chapter 4, but now, holding within the value set of the magnitude or rate-of-change of every variable rather than the complete set of all the 4-tuple parametric fuzzy numbers. We call A and B the α *adjacent qualitative values* within a particular value set if and only if there does not exist a C, which belongs to the subset such that $A <_\alpha C <_\alpha B$ if $A <_\alpha B$.

The following are the possible state transition rules, where A_1 and A_2 (or B_1 and B_2) are the α adjacent qualitative values of each other:

(1) If $B_1 >_\alpha 0$ ($B_1 <_\alpha 0$), then

if $A_1 \in R$, then $A_2 >_\alpha A_1$ ($A_2 <_\alpha A_1$),

else $A_2 \geq_\alpha A_1$ ($A_2 \leq_\alpha A_1$).

(2) If $B_1 = 0$, then

if $A_1 \in R$, then

if $A_2 >_\alpha A_1$ ($A_2 <_\alpha A_1$, or $A_2 = A_1$), then $B_2 >_\alpha 0$ ($B_2 <_\alpha 0$, or $B_2 = 0$);

else

if $A_2 \geq_\alpha A_1$ ($A_2 \leq_\alpha A_1$), then $B_2 >_\alpha 0$ ($B_2 <_\alpha 0$),

if $A_2 = A_1$, then $B_2 \in \{0, X, Y\}$, where X and Y are the α adjacent

qualitative values of 0.

It is straightforward to check the validation of the above rules. Let us take the case $B_1 >_\alpha 0$ in rule (1) as an example to illustrate this. When $A_1 \in R$, from the direct use of the standard intermediate value and mean value theorems we know that, for any $a_2 \in (A_2)_0$, $a_2 > A_1$ (i.e., $A_2 >_0 A_1$), therefore $A_2 >_\alpha A_1$, $\alpha \in [0, 1]$. If A_1 is not a real number but either a real interval or a strict fuzzy number or a fuzzy interval, for any underlying real-valued variable that takes values from $(A_1)_\alpha$, when it increases it will either still belong to $(A_1)_\alpha$ or go into the immediate next real interval whose elements are larger than the upper bound of the $(A_1)_\alpha$. Thus, if the α-cut of the magnitude value within the adjacent state of A_1 is $(A_2)_\alpha$, for any $a_2 \in (A_2)_\alpha$, a_2 is, of course, either belonging to $(A_1)_\alpha$ or larger than any $a_1 \in (A_1)_\alpha$. If not, or $A_1 >_\alpha A_2$, for any $a_1 \in (A_1)_\alpha$ and $a_2 \in (A_2)_\alpha$ this would result in $a_1 > a_2$ which indicates that the underlying variable is decreasing instead of increasing, thereby leading to a conflict. From this, it follows that $a_2 \geq a_1$ or $A_2 \geq {}_\alpha A_1$. Other cases within the possible transition rules can be similarly verified.

Clearly, these possible state transition rules play the same role in fuzzy qualitative simulation as the so-called *P* and *I* transition table does in *QSim*, although no new 'landmarks' will be generated in the former case. In fact, without considering new landmark generation, if a fuzzy quantity space degenerates to an *QSim*'s quantity space, i.e., a space whose elements are some special real numbers (landmarks) and the real crisp intervals adjacent to these landmarks, these rules (with $\alpha = 0$) can be easily reorganised into the *P* and *I* transition table in *QSim* [52]. For instance, let $l_j = A_1 \in R$ and $B_1 = 0 = std$, then, with respect to rule (2), we have

1) $A_2 >_0 A_1,\ B_2 >_0 0,$

2) $A_2 <_0 A_1,\ B_2 <_0 0,$

3) $A_2 = A_1,\ B_2 = 0.$

In case 1), for any $a_2 \in A_2$, $a_2 > A_1 = l_j$, therefore, $A_2 = (l_j, l_{j+1})$ (l_{j+1} is the landmark just larger than l_j); whilst $B_2 >_0 0$ simply means that $B_2 = inc$ if B_2 can only take values from $\{inc, std, dec\}$. This is exactly the *P2* transition of *QSim*, namely,

$$<l_j, std> \rightarrow <(l_j, l_{j+1}), inc>.$$

Similarly, case 2) indicates the following *P3* transition, where l_{j-1} is the landmark just smaller than l_j:

$$<l_j, std> \rightarrow <(l_{j-1}, l_j), dec>.$$

Finally, case 3) shows the *P1* or *I1* transition:

$$<l_j, std> \rightarrow <l_j, std>.$$

Based on the above possible state transition rules, a set of transitions from one given qualitative state description to its possible immediate successors (or next states) can be produced. Repeating such a process by treating the newly produced state as the given one and then generating further successor states from it, a sequence of possible future states can be obtained for each system variable, resulting in the generation of the possible system behaviour(s).

5.3.2. Arrival Times for State Transitions

An important notion associated with each possible state transition is the time that such transitions take. In fact, the time scale that a variable takes to transition from one qualitative state to another has been identified and called the *arrival time* in [113, 116]. We use this terminology within this thesis and denote an arrival time by ΔT_a. although, as pointed out in chapter 2, when describing states of continuous variables by means of real (crisp) intervals the corresponding arrival times must equal zero.

It is important, herein, to recognise the difference between the persistence time and the arrival time. As defined, a persistence time indicates the duration within which a state is described and is, therefore, local to only one pair of the magnitude and rate-of-change values of a variable. An arrival time shows the temporal interval it takes for a variable to reach its next state from the present one, thereby reflecting the influence of two (adjacent) states. Such a time scale results from the no excluded-middle property of a fuzzy quantity space, or equivalently the gradual boundaries of the fuzzy qualitative values. Otherwise, if the boundary between two adjacent states could be described by a (crisp) real number the arrival time would vanish, remaining only two non-fuzzy persistence times.

For simplicity, the following only discusses the arrival times taken by transitions between two states $<A_1, B_1>$ and $<A_2, B_2>$ where A_1 and A_2 satisfying that $A_2 \geq_\alpha A_1$, $\alpha \in [0, 1]$. As for the transitions between $<A_1, B_1>$ and $<A_2, B_2>$ with $A_2 \leq_\alpha A_1$, the corresponding discussion and the resulting computational method can easily be obtained from symmetry.

5.3.2.1. Theoretical Preparations

As with the persistence time, the arrival time, in *FuSim*, should also be a fuzzy number. Consequently, to determine the arrival time we use a similar method to that previously used for calculating the persistence time.

However, before presenting the computational method, basic notions for expressing it are required. First, it is necessary to find a key point, called the *crossing-point*, which is shared by the membership distribution of the current qualitative magnitude A_1 of a variable and that of the qualitative magnitude A_2 in the next state of the variable. For a Null-transition or an R-transition, there does not exist such a crossing-

point since $A_1 = A_2$. For an M-transition or MR-transition, where $A_1 = [p_1, p_2, p_3, p_4]$ and $A_2 = [q_1, q_2, q_3, q_4]$, a crossing-point (u, v) is defined as shown in figure 5.7. The value of u is the underlying real point where the membership distribution of A_1 intersects with the distribution of A_2 and v is the degree of membership of u within A_1 or A_2:

$$crossing\text{-}point(u, v): \begin{cases} u = \dfrac{1}{q_3 + p_4}(q_1 \times p_4 + p_2 \times q_3), \\ v = -\dfrac{1}{q_3 + p_4}(q_1 - p_2 - q_3 - p_4). \end{cases}$$

Due to the fact that all the fuzzy qualitative values are convex fuzzy numbers and reflect desired granularity any two adjacent qualitative values only share one crossing-point. Within this definition, the membership degree v of the crossing-point (u, v) is called the *crossing degree* and written as *c-degree* hereafter. Clearly, when a quantity space is fixed for a particular application, the set of all possible crossing-points can be predetermined.

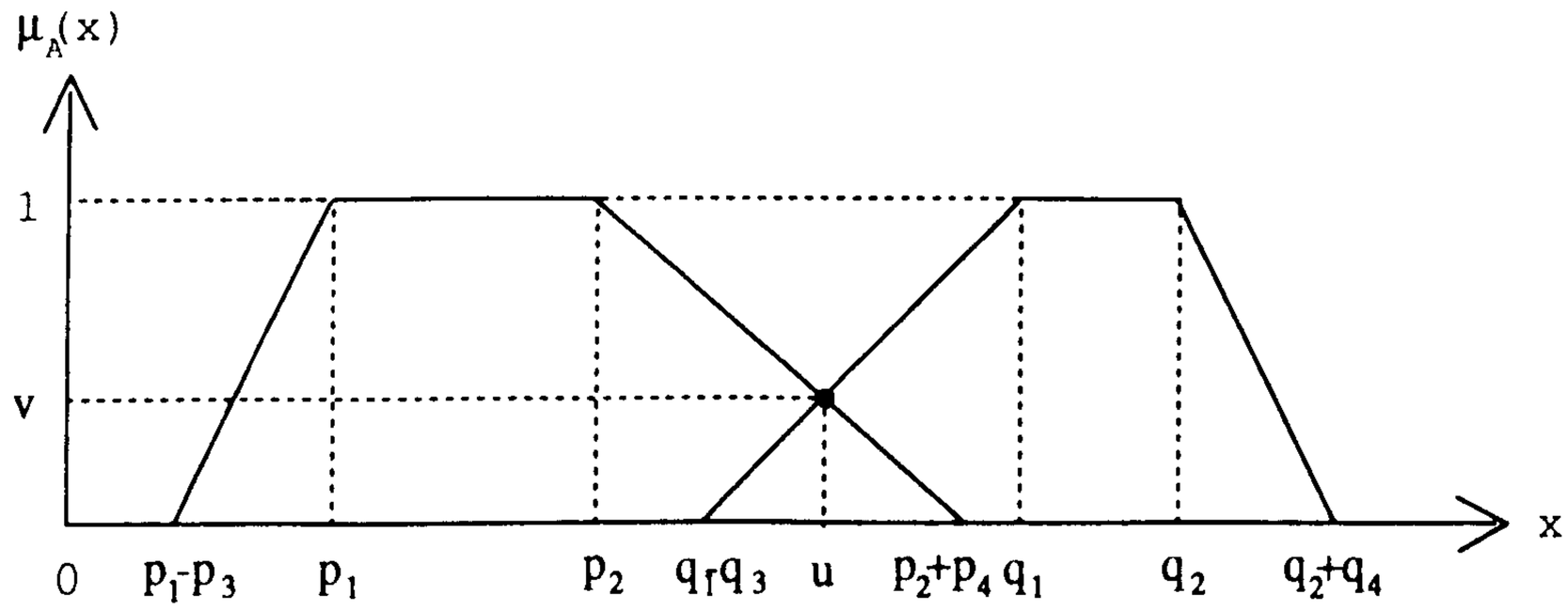
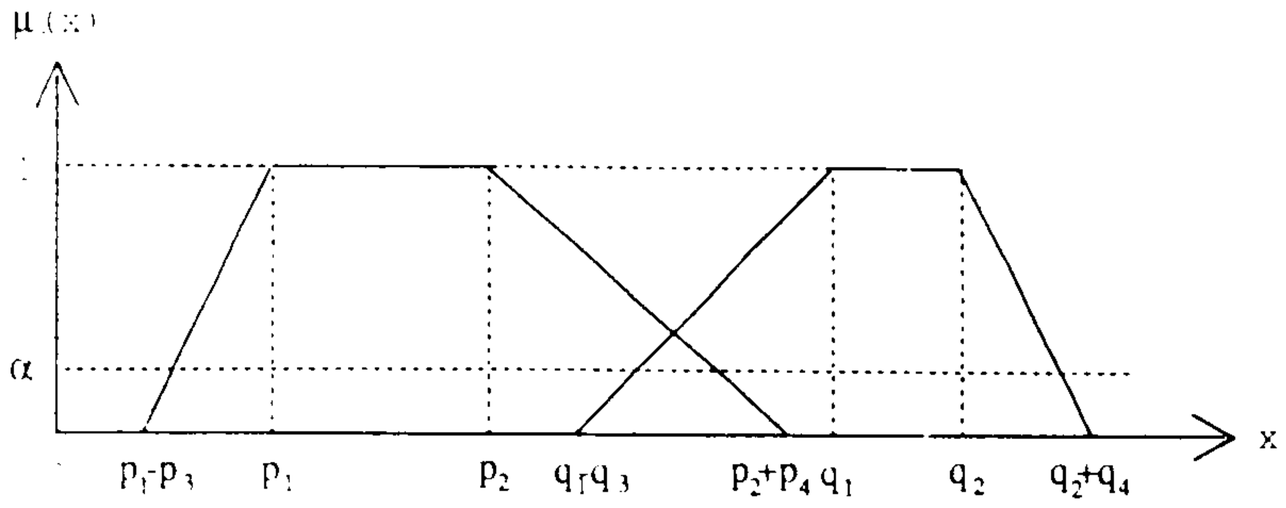


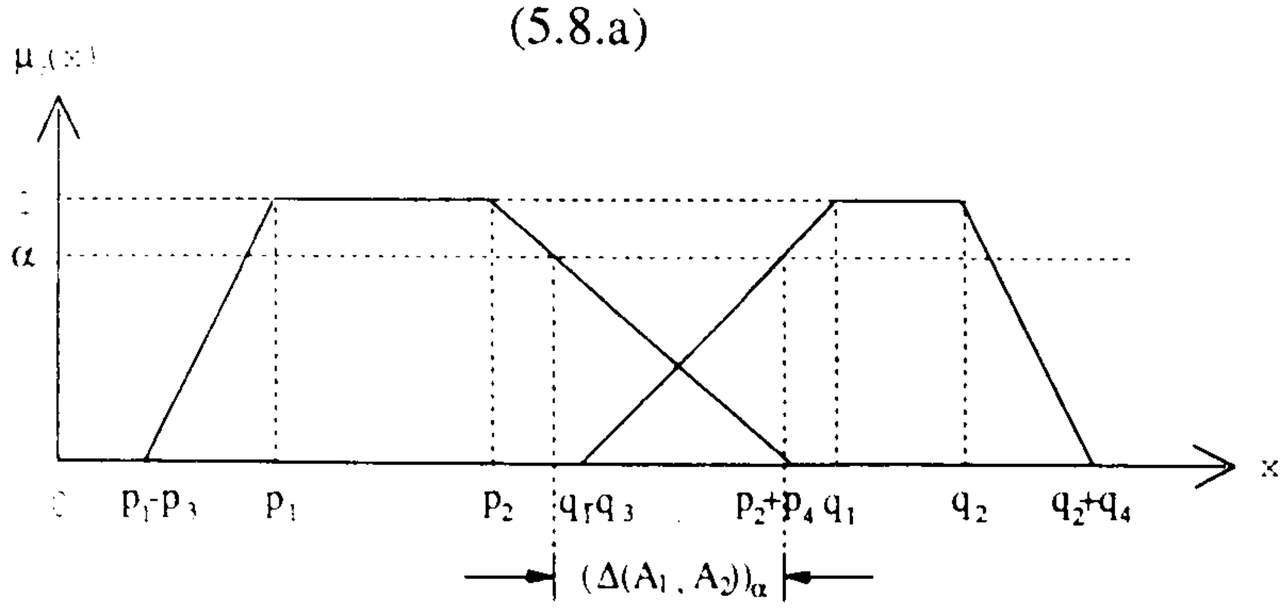
Fig. 5.7. *The Crossing-Point of Two Fuzzy Numbers*

Another important concept is the so-called α *distance metric* between two fuzzy numbers. An α distance, denoted by $(\Delta(A_1, A_2))_\alpha$, is defined as the shortest distance

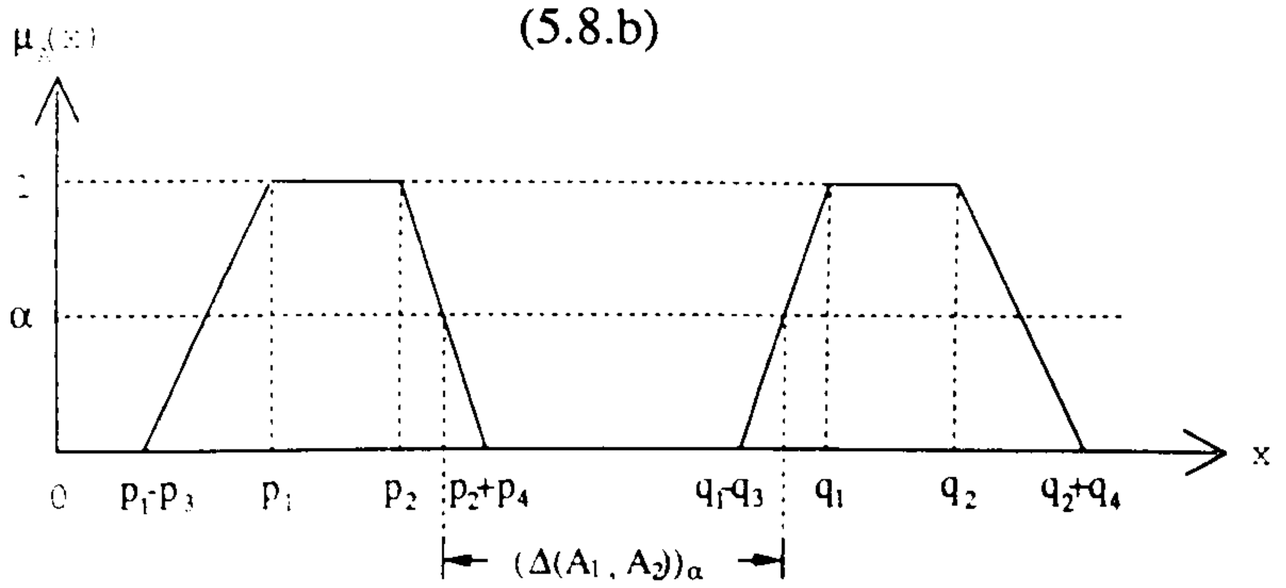
between any two underlying real numbers, each of which belongs to a different fuzzy number with the highest degree of membership being α. Informally, the α distance between any two fuzzy numbers is the shortest distance between the α-cuts of these two fuzzy numbers. For the 4-tuple parametric representation of fuzzy qualitative values, three different situations may exist that determine three different types of α distance, as depicted in figure 5.8.



(5.8.a)



(5.8.b)



(5.8.c)

Fig. 5.8. *Three Types of* α *Distance*

These α distances are listed as follows, where $A_1 = [p_1, p_2, p_3, p_4]$ and $A_2 = [q_1, q_2, q_3, q_4]$:

Type 1 $(\Delta(A_1, A_2))_\alpha = 0$ (see figure 5.8.a).

Type 2 $(\Delta(A_1, A_2))_\alpha = q_1 - p_2 + (\alpha - 1)(q_3 + p_4)$ (see figure 5.8.b).

Type 3 $(\Delta(A_1, A_2))_\alpha = q_1 - p_2 + (\alpha - 1)(q_3 + p_4)$ (see figure 5.8.c).

Notice that, symbolically, both type 2 and type 3 have the same α distance in representation. However, they have rather different implications since the former indicates a situation where A_1 intersects A_2, whilst the latter shows that the two fuzzy numbers do not intersect. When considering the coverage property of a quantity space, if the underlying real-valued variables continuously take values from an interest field, the latter situation implies that there is at least one qualitative value existing between A_1 and A_2 and, therefore, A_1 and A_2 cannot be the adjacent qualitative values of each other. Although, at present, only types 1 and 2 are of interest to us, the third type is listed for completeness of the presentation and, in particular, in preparation for the use of it in chapter 7.

Based on the notions of the crossing-point and the α distance metric, two general rules for determining the arrival times can now be constructed.

5.3.2.2. Computational Method for Arrival Time

The way to calculate arrival times is, of course, dependent upon what kind of transition is currently under consideration. In general, it can be described by two different rules, one for transitions where the magnitude of a variable is not changing and the other for when it is. It is not surprising that the computational method for arrival time is more complex than that for persistence times, since it deals with the time related to two adjacent states whilst a persistence time is only considered within

one particular state.

The following are the computational rules:

(1) For a Null-transition or an R-transition, i.e., $<A, B_1>$ to $<A, B_2>$ with $B_1 = B_2$ or $B_1 \neq B_2$ respectively, $\Delta T_a = 0$.

(2) For an M-transition or an MR-transition, if the value of α used in the possible transition rules, is less than or equal to the c-degree of the crossing-point between two adjacent magnitudes $A_1 = [p_1, p_2, p_3, p_4]$ and $A_2 = [q_1, q_2, q_3, q_4]$, then $\Delta T_a = 0$; otherwise, use the following sub-rules to find the arrival time:

(2-1) For an M-transition, $<A_1, B>$ to $<A_2, B>$ and $A_1 \neq A_2$, the rules to calculate ΔT_a are

(i) If $0 \Subset B_\alpha$, then $\Delta T_a \in \dfrac{(\Delta(A_1, A_2))_\alpha}{|B|_\alpha}$,

where $|B|_\alpha = \begin{cases} [q_1 + q_3(\alpha - 1), q_2 + q_4(1 - \alpha)], & B >_\alpha 0, \\ [-q_2 + q_4(\alpha - 1), -q_1 + q_3(1 - \alpha)], & B <_\alpha 0; \end{cases}$

and $(\Delta(A_1, A_2))_\alpha = q_1 - p_2 + (\alpha - 1)(q_3 + p_4)$.

(ii) If $0 \in B_\alpha$, then ΔT_a is not well-determined, any length of time may elapse.

(2-2) For an MR-transition, $<A_1, B_1>$ to $<A_2, B_2>$, and $A_1 \neq A_2$, $B_1 \neq B_2$, let $B = B_2 - B_1$, then, the rules to calculate ΔT_a are the same as those in case (2-1).

Formally speaking, in each case (i) of the sub-rules (2-1) and (2-2), ΔT_a should be written as

$$\Delta T_a \in \alpha \left\{ \frac{(\Delta(A_1, A_2))_\alpha}{|B|} \right\}_\alpha .$$

However, for the same reason as given in the explanation of the relevant rule for calculating the persistence time, we use a simplified formula to determine the arrival time. By substituting $(\Delta(A_1, A_2))_\alpha = q_1 - p_2 + (\alpha - 1)(q_3 + p_4)$ into case (i) we have

$$\Delta T_a \in \frac{q_1 - p_2 + (\alpha - 1)(q_3 + p_4)}{|B|_\alpha}.$$

As a simple example, figure 5.9 shows the arrival time for a variable x to change from $QS(x, \Delta T_{p_1}) = <A_1, B>$ to $QS(x, \Delta T_{p_2}) = <A_2, B>$, (M-transition), together with two persistence times for x to exist within these two states, respectively.

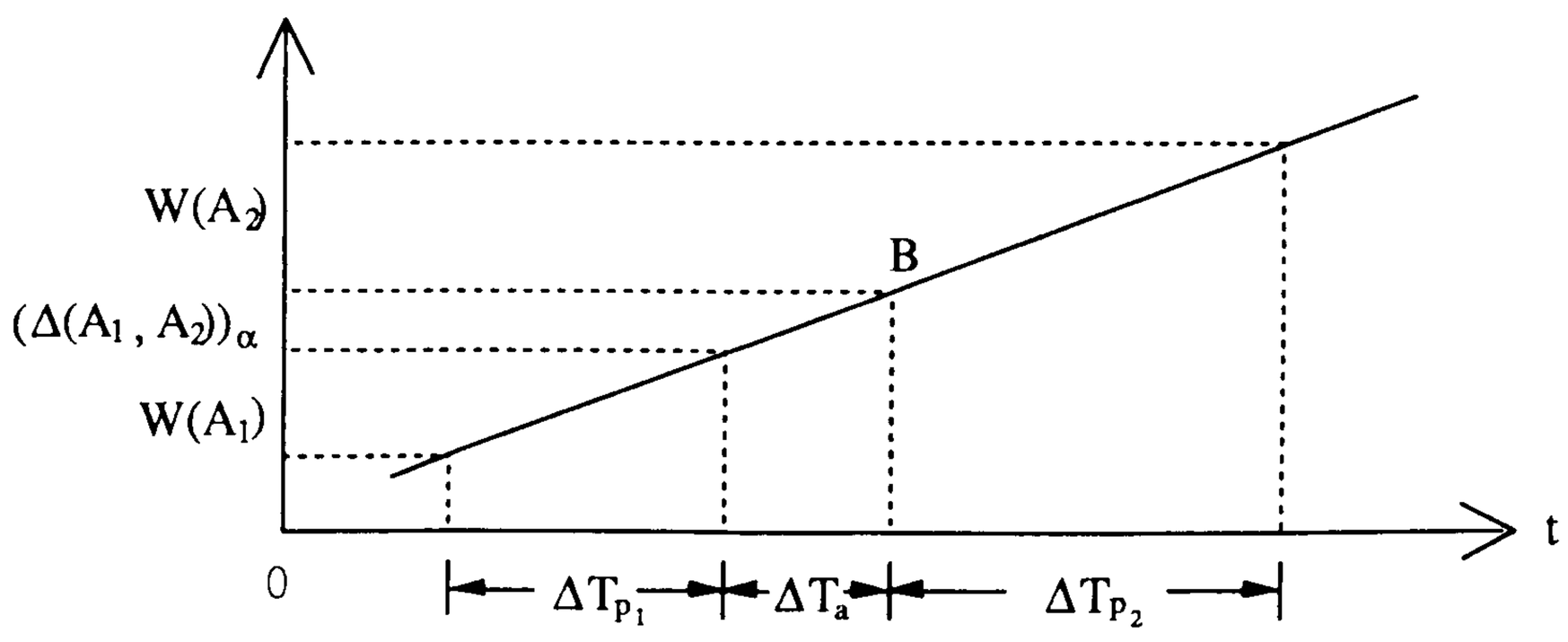


Fig. 5.9. *The Arrival Time for an M-Transition*

Within this figure, $A_1 = [p_1, p_2, p_3, p_4]$, $A_2 = [q_1, q_2, q_3, q_4]$, $B \in R$, $B > 0$, and

$$W(A_1) = p_2 - p_1 + (1 - \alpha)(p_3 + p_4),$$

$$W(A_2) = q_2 - q_1 + (1 - \alpha)(q_3 + q_4),$$

$$(\Delta(A_1, A_2))_\alpha = q_1 - p_2 + (\alpha - 1)(q_3 + p_4).$$

Obviously, if $p_1 = p_2 = p$, $q_1 = q_2 = q$, $p_3 = p_4 = q_3 = q_4 = 0$, i.e., the fuzzy quali-

tative values degenerate into real numbers, then $\Delta T_{p1} = \Delta T_{p2} = 0$; whilst ΔT_a now represents the sampling interval between real-valued states $<p, B>$ and $<q, B>$ in a discrete-time model.

It is important to remember that in the real case where the states are represented by adjacent crisp intervals there is no arrival time cost for a state to transition when treating variables as continuous functions of time. Actually, arrival times result from the 'soft' boundary representation of fuzzy qualitative values. The computational method for the arrival time reflects this fact. For example, if a variable transitions from $<A_1, B_1>$ to $<A_2, B_2>$, where $A_1 = [p_1, p_2]$, $A_2 = [q_1, q_2]$, $p_1 < p_2 = q_1 < q_2$, and B_1, B_2 are crisp intervals, $0 \in B_1 \cup B_2$; then, $(\Delta(A_1, A_2))_\alpha = 0$. Thus, $\Delta T_a = 0$.

5.4. Filtering Techniques

Possible state transition rules determine a set of plausible successor states from a given initial state. Basically, this is only decided upon the use of continuity of system variables. Like *QSim*, *FuSim* applies both local and global filtering methods to further restrict the set of next states. In fact, further restrictions on these possible successor states can be imposed by utilising the constraint relationships between the variables -- called *constraint filtering*, and information on the persistence and arrival times -- called *temporal filtering*. In addition, other knowledge about the system being modelled may be used to produce so-called *global filtering* methods. This section describes these three different filtering techniques.

5.4.1. Constraint Filtering

As *FuSim* adopts a constraint-centred approach, the *fuzzy constraint filtering* methods are essentially similar to those in *QSim*. Actually, for each argument of each

constraint, its associated set of possible qualitative values, generated from the use of the possible state transition rules, is filtered for consistency with the definition of the constraints and the consistency between constraints which share the argument. However, the semi-quantitatively extended function relational constraints greatly restrict the possible values that a constrained argument can have as shown in the example given in section 5.6. This is, of course, expected since more detailed information than monotonic functional dependencies has been utilised. The result of the fuzzy constraint filtering is a reduced set of possible successor states associated with each variable.

5.4.1.1. Constraint Filtering Criteria

In general, for a given relevant constraint, $Q(z) = f(Q(x), Q(y))$, where variable y may be the same as variable x, each argument in the constraint will have a set of possible qualitative values as either the qualitative magnitudes or the qualitative derivatives of a variable in the next state. The actual form of the constraint determines whether or not an element of this set of qualitative values is indeed the magnitude value (or the derivative value) of the associated variable at the successor state. Let the three sets of qualitative values for $Q(x)$, $Q(y)$, and $Q(z)$ be denoted by S_x, S_y, and S_z, respectively. Then, a *constraint consistency filtering criterion* local to the constraint simply says:

$$\text{For any } (Q_0(x), Q_0(y), Q_0(z)) \in S_x \times S_y \times S_z,$$

$$\text{if } \{Q_0(z)\} \cap \{f(Q_0(x), Q_0(y))\} = \Phi,$$

$$\text{filter } (Q_0(x), Q_0(y), Q_0(z)) \text{ from } S_x \times S_y \times S_z.$$

Within this criterion, $f(Q_0(x), Q_0(y)) \in Q_F$, indicating the values obtained from using the approximation principle to the result of a direct operation, f, on $Q_0(x)$ and

$Q_0(y)$; and $S_x \times S_y \times S_z$ is the Cartesian product of S_x, S_y, and S_z.

After each constraint within the system model has been checked with respect to the filtering criterion above, the constraint is, as before, associated with sets of the qualitative values of its arguments $Q(x)$, $Q(y)$, and $Q(z)$. If there are two or more relevant constraints sharing an argument, then the sets of the qualitative values of this argument in these constraints must be identical to each other. We call this rule the *pairwise consistency filtering criterion.* For example, if two constraints which share a common argument, say, the qualitative magnitude of a variable z, are $Q(u) = f(Q(y), Q(z))$ and $Q(z) = deriv(Q(x))$, and if the sets of possible qualitative values of the argument in the two constraints are S_{z_1} and S_{z_2}, respectively; then, the filtered set of the possible qualitative magnitudes of z in the next state is $S_{z_1} \cap S_{z_2}$.

5.4.1.2. Using the Waltz Algorithm for Constraint Filtering

It is important to notice that, once the set of possible qualitative values of an argument within a constraint has been checked for either constraint or pairwise consistency and some spurious values are indeed removed from this set, all the other constraints which share an argument with this constraint should then be rechecked. For this reason, the Waltz filtering algorithm, first presented in [112], is used for efficiency within *FuSim.* The algorithm entails an operation, called *refine*, on each relevant constraint and each argument of the constraint within the system model iteratively until the operation produces no more changes. An efficient implementation of the Waltz algorithm can be found in [14].

To be concise, let $C : C(Q(x_1), Q(x_2), Q(x_3))$ be a relevant constraint over three arguments: $Q(x_i)$, $i = 1, 2, 3$, though $Q(x_i)$ may be equal to $Q(x_j)$, $i \neq j$.

And, let S_i be the set of qualitative values for the argument $Q(x_i)$. The refine operation in the present application is then defined by the following, where $C(A_1, A_2, A_3)$ indicates that A_1, A_2, and A_3 are consistent with the constraint C:

$$refine(C, Q(x_i)) = \{A_i \in S_i \mid (A_j \in S_j, j = 1, 2, 3, j \neq i); C(A_1, A_2, A_3)\}.$$

Another opportunity that the Waltz algorithm offers can be captured by noticing that, in the case when this algorithm halts by assigning some argument the empty set, the current sets of the values for all the arguments being checked must have been inconsistent. It is therefore convenient to utilise it to check if any given initial state of the system itself is valid against the system model before starting the fuzzy simulation. This avoids the unnecessary computation that otherwise has to be encountered when an incorrect initial state is presented to the simulation system.

A possible transition of a variable survives the constraint filtering if both the qualitative magnitude and the qualitative derivative of its next state remain after the filtering operation. Thereby, for each variable, the result of the fuzzy constraint filtering is a reduced set of its possible next states.

5.4.2. Temporal Filtering

In this sub-section, the estimates of the persistence time and the arrival time are used to develop an effective temporal filtering method to further eliminate spurious successor states. We first describe the method for achieving a sequence of temporal labels that is directly attached to the sequence of simulated states, reflecting the evolution of a continuous dynamic system over time. Then, the filtering criterion is presented and the fundamental reason that such temporal filters can remove a large number of spurious behaviours is discussed.

5.4.2.1. Absolute Time Indices

It is shown in section 5.3 that, given two adjacent states of a variable, the persistence times within both states and the arrival time for the transition from the first state to the second can be calculated. It should be clear, however, that the time scales so computed are temporal intervals, indicating either how long the variable remains within a particular state or how long it takes to transition to the next state, and not the total time taken from the time label of the initial state of the variable. Nevertheless, once a sequence of states is generated, a sequence of the pairs of persistence time and arrival time is also produced. Based on this, the absolute time associated with a newly generated state is determinable.

Without losing generality, let the state sequence of a variable x be denoted by $\{X(n) \mid n = 0, 1, 2, \cdots\}$ with the initial state being $X(0)$ and the initial absolute time being $t_0' = 0$. Consider the simplest case first, where the sequence only consists of two states $X(0)$ and $X(1)$. From the rules for computing persistence and arrival times, the persistence times of $X(0)$ and $X(1)$ and the arrival time for $X(0)$ to reach $X(1)$ can be obtained and written as $\Delta T_{p_0}(x)$, $\Delta T_{p_1}(x)$, and $\Delta T_{a_1}(x)$, respectively. Thus, the time index attached to the state $X(0)$ is equal to

$$t_0 = t_0' + \Delta T_{p_0}(x) = \Delta T_{p_0}(x);$$

and the absolute time of the state $X(1)$ will then be

$$t_1 = \Delta T_{p_0}(x) + \Delta T_{a_1}(x) + \Delta T_{p_1}(x).$$

That is, this state appears when the time index becomes t_1. It is worth pointing out that, here, due to the fuzzy state representation, both the persistence times and the arrival time are given in terms of (estimated) possible ranges that the corresponding times fall within. Henceforth, the absolute time so found, t_1, is described by a *possi-*

ble range that it may lie in.

In general, for a new state $X(N+1)$, $N = 0, 1, \cdots$, its associated absolute time can be obtained in a similar way to the determination of t_1 shown above. Actually, suppose that the arrival time from state $X(N)$ to $X(N+1)$ is $\Delta T_{a_{N+1}}(x)$ and the persistence time of $X(N+1)$ is $\Delta T_{p_{N+1}}(x)$, then the time index for the state $X(N+1)$ is the following:

$$t_{N+1} = \sum_{n=0}^{N} \Delta T_{p_n}(x) + \sum_{n=1}^{N} \Delta T_{a_n}(x) + \Delta T_{a_{N+1}}(x) + \Delta T_{p_{N+1}}(x),$$

where $\sum_{n=0}^{N} \Delta T_{p_n}(x) + \sum_{n=1}^{N} \Delta T_{a_n}(x)$ is the time label of the state $X(N)$.

Of course, with the same reason as that for t_1, the t_{N+1} is also described in the form of a range rather than a real-valued time point. In so doing, when *FuSim* generates a state sequence it also presents a sequence of temporal indices associated with the states. This highlights an advantage of our simulation method over previous qualitative simulation techniques, since the conventional methods, e.g. *QSim*, only allow a temporal ordering in the states but no indication of the temporal extent. Noticing that such time indices are derived from the persistence and arrival times, for simplicity, when we describe the behaviour of a system we do not distinguish the terms between 'a state sequence with associated temporal durations' and 'a state sequence with associated absolute time indices' within the thesis, unless explicitly required.

5.4.2.2. Temporal Filtering Criterion

It is clear from the above discussion that, before a variable enters the current state, it has spent a certain period of time persisting in the last state plus a duration to reach this state, except in the case where the current state is the initial state. This is generally true for all the system variables. Having this observation and noticing that

a system state consists of all the individual states of the variables at a particular time index, a general filtering criterion, termed *temporal filtering rule* can be established and stated as follows:

> For any two system variables, x and y, if their persistence times within the present state, $\Delta T_p(x)$ and $\Delta T_p(y)$, lie in $[p_{1x}, p_{2x}]$ and $[p_{1y}, p_{2y}]$, and the arrival times, $\Delta T_a(x)$ and $\Delta T_a(y)$, for them to transition to a possible successor state lie in $[a_{1x}, a_{2x}]$ and $[a_{1y}, a_{2y}]$ respectively, then, unless x and y are independent of each other, they must satisfy the following temporal constraint:
>
> If $[p_{1x}, p_{2x}] = [p_{1y}, p_{2y}]$ (or $[a_{1x}, a_{2x}] = [a_{1y}, a_{2y}]$),
>
> then $[a_{1x}, a_{2x}] \cap [a_{1y}, a_{2y}] \neq \Phi$ (or $[p_{1x}, p_{2x}] \cap [p_{1y}, p_{2y}] \neq \Phi$);
>
> else $[p_{1x}, p_{2x}] + [a_{1x}, a_{2x}] \cap [p_{1y}, p_{2y}] + [a_{1y}, a_{2y}] \neq \Phi$.

In this filtering rule, a system variable x being independent of another variable y means that, in the structural model of the system, there is neither any relevant constraint between x and y nor any constraints associating x and y through other system variables to force x to change together with y, or that, although there is a given constraint related x and y, x is controlled by an external influence of the system such that it will not be affected by the internal variable y.

This temporal filter is rather powerful because it utilises ordering information on the rates of change of system variables held in the fuzzy quantity space. Actually, the order relationships among fuzzy qualitative rates of change reflect additional knowledge about the higher-order (≥ 2) derivatives of the variables. The fact that higher-order derivative information can be used to eliminate certain impossible behaviours has previously been recognised [24, 54, 121, 122]. For instance, as

presented in [54], the *curvature constraints*, namely the second-order derivative constraints, can be utilised to further limit the possible transitions when the first-order derivative of a variable is zero. Such constraints may be automatically derived by algebraically manipulating the qualitative model of a system, This is done, however, on the basis of the assumption about the smoothness of partially known functional relationships. In *FuSim*, the information on the second-order derivatives of system variables is naturally embedded in the partial order relations inside the fuzzy quantity space. As a matter of fact, information on the differences in strength amongst rates of change is reflected in the computational methods for the persistence and arrival times. The basic representation of the qualitative derivative within conventional qualitative simulation algorithms is different in that it is restricted to the three values {−, 0, +} [4]. In which case, no qualification on the rate of change (e.g., increasing slowly) is possible and hence no calculations of the temporal durations may be made. Then, other methods such as the *curvature constraints* have to be used. From this aspect, the temporal filtering in *FuSim* is one of the most important extensions to *QR* techniques.

5.4.3. Global Filtering

After constraint and temporal filtering, complete state descriptions are generated. However, the set of next states may still contain a number of spurious states and, therefore, be non-unique. Actually, a complete state description is only a mathematically plausible successor to the current state of the system, i.e., an assignment of a possible transition to each variable in the system without conflicting with the constraint and temporal restrictions. In order to eliminate as many spurious behaviours as possible, *global filtering* techniques are then applied. That is to say, some of these behaviours may be eliminated by a knowledge of system theoretic properties of the

real behaviour or other, often heuristic, information from external sources. In general, such knowledge is not local to each possible transition between two adjacent states of a particular variable but presented to the entire system behaviour. This is the very reason that the corresponding filtering methods are named global filters.

Currently implemented global filters in *FuSim* are those that check for *no-change* and *repeating*. A no-change means that a new system state is identical to its immediate predecessor and, therefore, can be deleted in the simulation. By a repeating state we mean the new state is identical to a state of its predecessors but not the immediate previous one, or identical to one of the other successors of its predecessors, which is not in the behaviour branch ended with it. When such a state is met the behaviour is marked as repeating and no further next state is generated from it.

It has been noticed that a powerful global filter for all second-order systems, called the non-intersection constraint has been independently reported in [57] and [102]. It is a general constraint based on the requirement that a trajectory of a system variable in a phase space cannot intersect itself unless the trajectory is a closed curve. This principle is deduced from the existence and uniqueness theorems [8] for simultaneous first-order differential equations. In *FuSim*, however, a variable x takes its qualitative values and qualitative derivatives from a fuzzy quantity space and the excluded-middle law is generally invalid inside this space. As such the non-intersection constraint cannot be directly applied in a plane formed on the basis of fuzzy values, called a fuzzy phase plane, for a second-order system. Nevertheless, by taking advantage of the membership distribution of a fuzzy qualitative value, which describes the degree of possibility that a variable takes an underlying real number value, we can deduce the *degree of possibility* that an underlying real trajectory intersects itself [6]. However, having considered that, when dealing with second-order

systems, *FuSim* has already been able to significantly reduce the number of spurious behaviours that conventional qualitative simulation techniques may produce without using the non-intersection principle, and that the deduction about such degrees of possibilities would encounter considerable computational complexity, we will realise this additional global filter only when it appears to be necessary.

Apart from the non-intersection constraint, there exists another global filter named the energy constraint [38]. Based on reasoning about energy, this filter is able to eliminate an important source of spurious behaviours, e.g., the steady and increasing oscillations for a damped second-order system. As claimed in [38], the basic principle of this technique is applicable to any higher-order systems. Unfortunately, a number of significant limitations exist within this filter. Firstly, it essentially depends upon the decomposition of the highest derivative term in the system model into conservation and non-conservation parts. This is not trivial for slightly complex systems, even for the example given in [38] that is used for demonstrating such constraints. Actually, as indicated in [38], it is rather difficult to algebraically manipulate a given set of qualitative differential equations that serves as the system model so as to decompose the highest derivative term. Secondly, even if the decomposition is feasible, the consistency of a particular behaviour with respect to an energy constraint is checked by using the classical three-valued qualitative (sign) addition constraint. The addition constraint itself, however, suffers from the severe problem of qualitative ambiguity [101]. Seeing this, we also leave this global filter unimplemented within *FuSim*.

In general, we conjecture that the extra information derived from the fuzzy quantity space:

1) ordering information on rates of change, and

2) strength information on functional dependencies,

eliminates many spurious behaviours at source, and, therefore, obviates the need for extensive global filtering beyond the no-change and repeating filters when considering second-order systems. So far, our empirical results have justified this assertion.

5.5. FuSim Algorithm Structure

As with *QSim*, the *FuSim* algorithm starts with a description of a set of variables with an associated quantity space, a known structural model (in terms of a set of fuzzy constraints), and an initial state of a physical system and produces the set of possible behaviours of the system by generating and filtering the set of possible transitions from one qualitative state description to its successors (see figure 5.1). A crucial advantage of *FuSim* over other approaches to qualitative simulation is that it also generates a sequence of temporal durations associated with each possible sequence of system states or each possible behaviour of the system.

To complete the presentation of the fuzzy qualitative simulation technique, this section presents the skeleton of the *FuSim* algorithm as follows, where steps 2 and 3 are combined by the Waltz filtering algorithm for efficiency reasons. Within the algorithm, constraint filters are followed by a temporal filter. This is so arranged to avoid unnecessary computation on temporal information required by the temporal filter. This algorithm structure can be better understood with reference to its functional overview shown in figure 5.2.

Step 1 From the current system state, determine a set of possible transitions for each variable through the possible state transition rules, a set of qualitative magnitude and qualitative derivative pairs is then attained; meanwhile, compute the persistence times of the variables at the current state and find the

absolute time index associated with the state.

Step 2 For each constraint, filter the qualitative value sets of all its arguments so that they satisfy the constraint.

Step 3 Filter the qualitative value set of each argument for consistency between constraints which share the argument.

Step 4 Compute the arrival times of the variables; then, use the temporal filtering rule to eliminate the qualitative value set for each variable which is in conflict with the rule.

Step 5 Generate all possible global interpretations from the value sets of all variables and use the simple global filters (no-change and repeating) to further eliminate these sets; then, mark each remaining interpretation successor of the current state.

Step 6 Repeat steps 1-5 until no more changes occur or a resource limit is exceeded.

The above algorithm has been implemented in the Quintus Prolog language [11] on a SUN 3 workstation with 16M bytes of RAM, the human-computer interface is realised by using the Prowindows graphics extension. The software architecture of the *FuSim* algorithm is given in Appendix B.

5.6. Examples

In this section, two examples are given to illustrate the operation and advantages of *FuSim*. For this purpose, the section is divided into two subsections, one demonstrating the fuzzy qualitative simulation of a system consisting of two coupled tanks, and the other showing the fuzzy qualitative simulation of a system composed of a mass on a spring. Both systems are commonly used to exhibit the capability and/or

problems of a particular reasoner in the research area of qualitative reasoning. These examples demonstrate that *FuSim* has a number of significant advantages over previous qualitative simulation techniques. In particular, it permits a more detailed description of quantities and functional relationships than existing qualitative reasoners, and yet requires neither the precision of real numbers nor the exact and complex relationships amongst variables. For each of the systems used in the examples, *FuSim* produces a unique behaviour, in terms of a sequence of states associated with temporal durations, without resorting to other supplementary filtering methods and, hence, without requiring added knowledge from other sources.

The results given in the following are those obtained by directly running the *FuSim* programme written in Prolog [11]. Within these experimental simulations, unless explicitly stated, the coefficient α, required by the use of the resolution identity, is chosen to be 0.5.

It is important, however, to notice that the choice of the α value is made almost arbitrarily. Actually, the only restriction over choosing such a coefficient is that it must be larger than the maximum membership value amongst those membership values of all the crossing-points within the quantity space used. This is due to the fact that, if there is a crossing-point whose membership of belonging to two certain adjacent qualitative values is larger than the α, then, two originally intended adjacent values degrade to a single value with respect to this coefficient α. In which case, two otherwise distinguishable states, with each taking a different qualitative value from those two adjacent values, now become a single state. Of course, different choices of the coefficient α will, in general, result in different persistence and arrival times computed during a simulation process. Fortunately, this only affects the precision of such estimations of the time scales without altering the fundamental simulation procedures

of the *FuSim* algorithm. Indeed, the α value particularly chosen within a simulation process is used to exhibit the underlying real numbers that belong to some fuzzy number with the membership being at least this value. The simple reason for us to choose 0.5 as its value within these simulations is that, in the field of *Fuzzy Sets and Systems*, it is common to use this value, reflecting the cases where the worst uncertain knowledge is present. At the end of this section, we shall give a simulation result from utilising a different α ($\alpha = 0.7$) to show the robustness of the *FuSim* algorithm with respect to different choices of this coefficient.

5.6.1. Example 1: Two Coupled Tanks

Consider the basic coupled tank apparatus, as depicted in the first sub-window within figure 5.10 (This figure is a screendump of the major interface window, automatically produced at the first stage of the fuzzy qualitative simulation of the two coupled tank system). The system consists of eight variables:

l_1, l_2 -- the levels of tank 1 and tank 2,

l_{12} -- the difference between the levels of tank 1 and tank 2,

i, o, c -- the input flow rate, the output flow rate, and the cross-flow rate, respectively, and

n_1, n_2 -- the inner flow rates of tank 1 and tank 2.

Clearly, each system variable has its own underlying numeric range of values that it may take. For simplicity, however, we use a normalised range [–1, 1] to form the basis on which the fuzzy quantity space is discretised. The following fuzzy quantity space was adopted and denoted by Q_F:

$\{n_top, n_large, n_medium, n_small, zero, p_small, p_medium, p_large, p_top\}$,

with the qualitative values being represented by nine 4-tuple parametric fuzzy numbers as given below and shown in figure 5.11.

$$Q_F = \{[-1, -1, 0, 0.1], [-0.9, -0.75, 0.05, 0.15], [-0.6, -0.4, 0.1, 0.1],$$
$$[-0.25, -0.15, 0.1, 0.15], [0, 0, 0, 0], [0.15, 0.25, 0.15, 0.1],$$
$$[0.4, 0.6, 0.1, 0.1], [0.75, 0.9, 0.15, 0.05], [1, 1, 0.1, 0]\}.$$

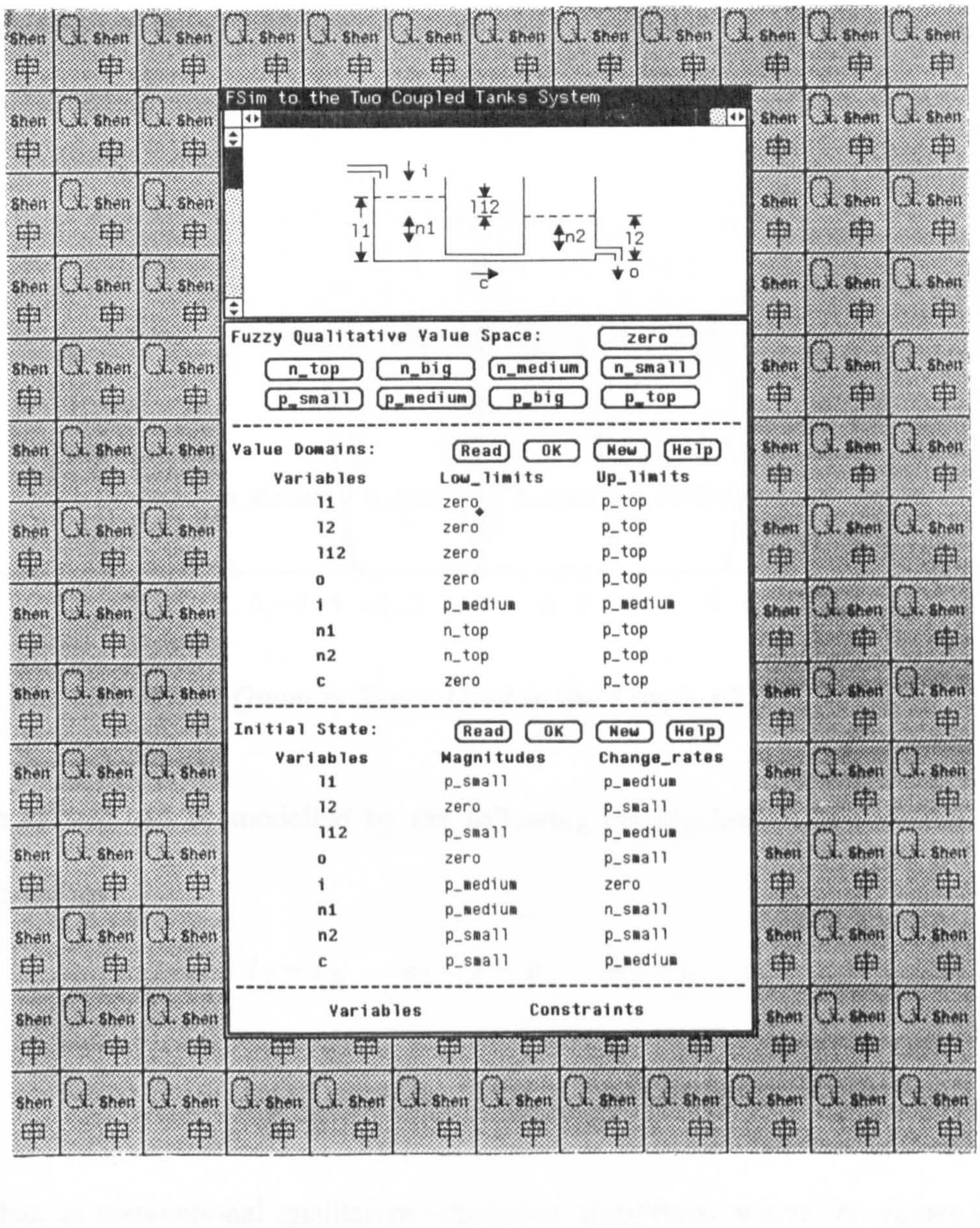


Fig. 5.10. *Major Interface Window of the Fuzzy Simulation of Two Coupled Tanks*

Each variable takes its own fuzzy qualitative values from a subset of Q_F. The ranges of the magnitudes of all system variables are shown in figure 5.10 under the heading

'Value Domains'. For example, variable l_1 takes qualitative values from

$$\{zero, p_small, p_medium, p_large, p_top\},$$

while the rate of change of variable n_1 may take qualitative values from the whole Q_F. If we fix the input flow rate i to be a given numeric value, say, $i = 0.5$, we may use *medium* to qualitatively describe it. This implies that, during the simulation, i can only take this single quantitative value that is qualitatively named *medium* and, therefore, its rate-of-change is always equal to *zero*.

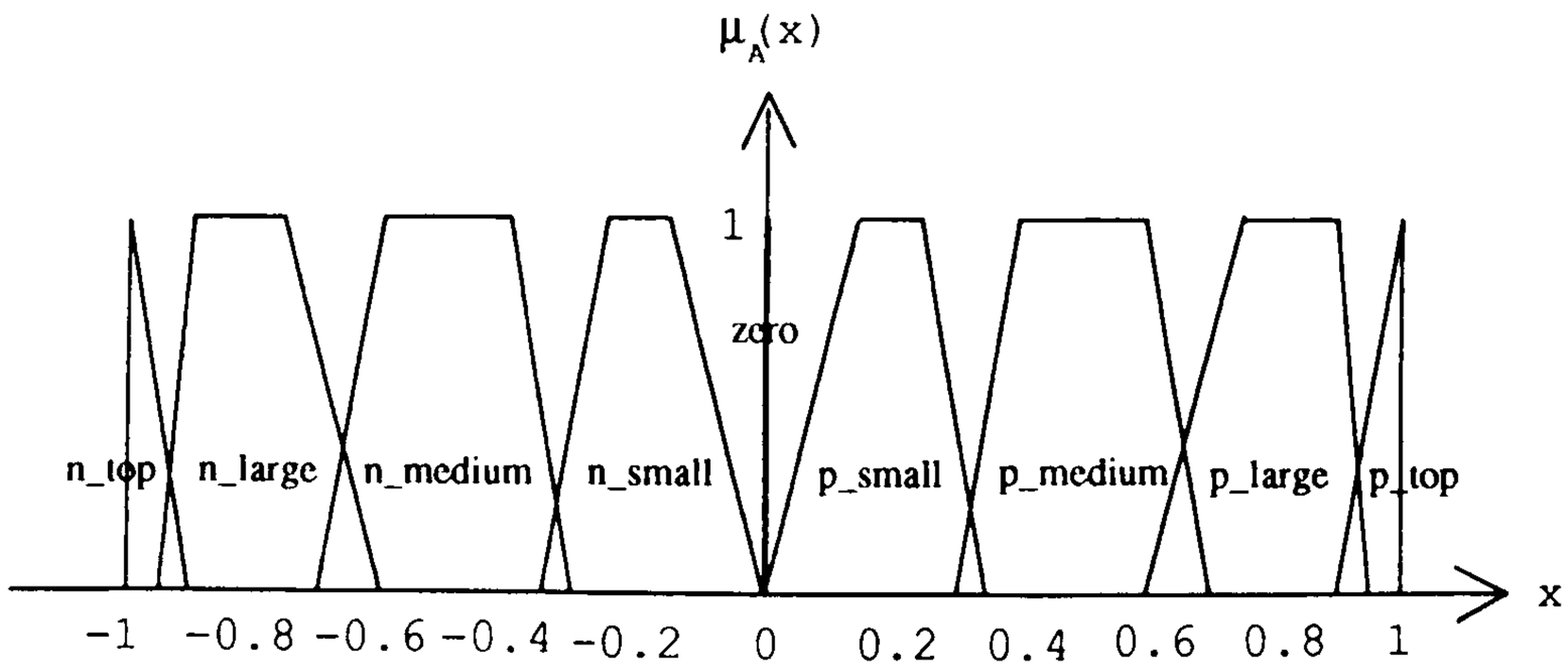


Fig. 5.11. *Fuzzy Quantity Space Used in the Coupled-Tanks Example*

The system can be modelled by the following six algebraic and derivative constraint equations:

$$l_{12} = l_1 - l_2, \quad n_1 = i - c, \quad n_2 = c - o,$$

$$n_1 = deriv\ l_1, \quad n_2 = deriv\ l_2,$$

$$deriv\ l_{12} = deriv\ l_1 - deriv\ l_2.$$

Notice that, in conventional qualitative simulation algorithms, where the values of the rates of change are modelled by three symbols {−, 0, +}, if the last algebraic constraint (amongst three rates of change) is used severe ambiguity will result [101, 107]. However, this problem is considerably reduced in *FuSim* as the rate of change of vari-

ables takes values from a more detailed quantity space rather than the simple three symbolic values.

To reflect the functional relationships between variables, in addition to the algebraic and derivative constraints, the functional dependency between the cross-flow rate c and the level difference l_{12} may be modelled, as shown in chapter 4, by

$$\begin{bmatrix} c \sim l_{12} & zero & p_small & p_medium & p_large & p_top \\ zero & 1 & 0 & 0 & 0 & 0 \\ p_small & 0 & 1 & 0 & 0 & 0 \\ p_medium & 0 & 0 & 1 & 1 & 0 \\ p_large & 0 & 0 & 1 & 1 & 1 \\ p_top & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Similarly, there exists another functional relationship between the output flow rate o and the level of tank 2, l_2:

$$\begin{bmatrix} o \sim l_2 & zero & p_small & p_medium & p_large & p_top \\ zero & 1 & 0 & 0 & 0 & 0 \\ p_small & 0 & 1 & 1 & 0 & 0 \\ p_medium & 0 & 0 & 1 & 0 & 0 \\ p_large & 0 & 0 & 1 & 1 & 0 \\ p_top & 0 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

It is clear that, the fuzzy relations obtained from the interpretation of the logic rules may contain some redundant information. However, the current implementation of *FuSim* does not check for this. Nevertheless, it will not affect the efficiency of the overall simulation because the translation from the rules to the relation matrices is done off-line.

The set of values under the heading 'Initial State' in figure 5.10 is taken as the initial condition that is presented to the system. Now that a description of the fuzzy model and an initial state of the coupled tank system have been given, the fuzzy qualitative simulation of the system is initiated and used to predict the possible behaviours

of the system. Here, we give some typical samples in one cycle of the simulation to show how *FuSim* works and produces a single next state without ambiguity.

From the possible state transition rules, each variable will have a respective set of possible next states in response to the initial condition as listed below:

l_1, l_{12}, c — (<*p_small* , *p_small*>, <*p_medium* , *p_small*>, <*p_small* , *p_medium*>, <*p_medium* , *p_medium*>, <*p_small* , *p_large*>, <*p_medium* , *p_large*>),

l_2, o — (<*zero* , *p_small*>, <*p_small* , *p_small*>, <*p_small* , *p_medium*>),

i — (<*p_medium* , *zero*>),

n_1 — (<*p_small* , *n_medium*>, <*p_small* , *n_small*>, <*p_small* , *zero*>, <*p_medium* , *n_medium*>, <*p_medium* , *n_small*>, <*p_medium* , *zero*>),

n_2 — (<*p_small* , *zero*>, <*p_small* , *p_small*>, <*p_small* , *p_medium*>, <*p_medium* , *zero*>, <*p_medium* , *p_small*>, <*p_medium* , *p_medium*>).

Thus, a number of possible next states are temporarily produced. Fortunately, many of these states are removed by the fuzzy constraint filter. For example, it is obvious that the state with variable l_1 being at <*p_small* , *p_large*> and n_1 at <*p_medium* , *n_small*> is a spurious one, since it invalidates the constraint $n_1 = deriv\ l_1$. Thereby, any conjunction (or combination) of the individual states of the variables, which includes <*p_small* , *p_large*> and <*p_medium* , *n_small*> as the states of variables l_1 and n_1 can be eliminated. This is efficiently accomplished by the Waltz algorithm. Likewise, state <*p_small* , *p_small*> of the variable c and <*p_medium* , *p_small*> of l_{12} cannot co-exist within a system state because they are inconsistent with the fuzzy relation $c \tilde{} l_{12}$. Although any combination, with <*p_small* , *p_small*> and <*p_medium* , *p_small*> being the possible successors of c and l_{12}, respectively, could also be eliminated by the temporal filter, matching the values of c and l_{12} with the degenerated fuzzy relation avoids unnecessary calculation

on temporal information.

It is important to notice that the qualitative state <*p_small* , *p_small*> of the variable c and <*p_medium* , *p_small*> of l_{12} survive all of the fuzzy algebraic and derivative constraint filtering, and that they do satisfy the condition that $sign(\ deriv\ \ c\ ) = sign(\ deriv\ \ l_{12}\ )$ from the initial state to the possible next state. That is to say, information provided by the monotonic increasing function $M^{+}(.,.)$ which is commonly used in other *QR* techniques cannot rule out this spurious case. This clearly shows the capability of fuzzy function relational constraint filtering.

As previously pointed out, the direct use of the constraint

$$deriv\ \ l_{12} = deriv\ \ l_1 - deriv\ \ l_2$$

in conventional qualitative simulation methods would result in qualitative ambiguity, though this constraint is well-known in the underlying differential equation representation. In *FuSim*, however, such a relevant constraint is valid and can be utilised to resolve ambiguities which would exist without applying it. Actually, the following pseudo successor state of the system,

l_1 :	<*p_medium* , *p_small*>	l_2 :	<*p_small* , *p_medium*>
l_{12} :	<*p_medium* , *p_large*>	o :	<*p_small* , *p_small*>
i :	<*p_medium* , *zero*>	n_1 :	<*p_small* , *n_small*>
n_2 :	<*p_medium* , *p_small*>	c :	<*p_medium* , *p_small*>,

generated from the initial state, survives all the constraints in the system model other than this constraint. Furthermore, even the temporal filter cannot eliminate the above state. Fortunately, this spurious state is removed since

$$deriv\ \ l_{12} = p_large \neq p_small - p_medium = deriv\ \ l_1 - deriv\ \ l_2,$$

which invalidates the constraint.

After constraint filtering, the following conjunction of the states of the system

variables remains as a possible next state of the initial one:

l_1 :	<*p_medium*, *p_medium*>	l_2 :	<*p_small*, *p_small*>
l_{12} :	<*p_small*, *p_small*>	o :	<*p_small*, *p_small*>
i :	<*p_medium*, *zero*>	n_1 :	<*p_medium*, *n_small*>
n_2 :	<*p_small*, *p_small*>	c :	<*p_small*, *p_small*>.

This state satisfies all the constraints on the system. However, we know that, on the one hand, within the initial state the persistence time of the variable l_1 takes value from [0.077, 0.143] and the arrival time for it to transition to the above state falls within [0.346, 0.643]; on the other hand, the variable n_1 persists within the initial state for a duration belonging to [1, 4] and takes no time to arrive at the new state (the arrival time cost by a Null-transition is *zero*). This is inconsistent with the temporal filtering rule since

$$[0.077, 0.143] + [0.346, 0.643] = [0.423, 0.786],$$

while,

$$[0.423, 0.786] \cap [1, 4] = \Phi.$$

Thus, this state is eliminated from the set of possible next states of the initial one. Alternatively, this can also be checked, and therefore ruled out, by comparing the time interval of the variable l_{12} against that of n_1. Actually, the persistence time of l_{12} at the initial state falls within [0.346, 0.643] and the arrival time for the R-transition that l_{12} takes is *zero*. Thereby, the total time interval taken by l_{12} does not intersect with that of the variable n_1 and hence the state is deleted.

The only successor state that passes all the filters is the following:

l_1 :	<*p_small*, *p_medium*>	l_2 :	<*p_small*, *p_small*>
l_{12} :	<*p_small*, *p_medium*>	o :	<*p_small*, *p_small*>
i :	<*p_medium*, *zero*>	n_1 :	<*p_medium*, *n_small*>
n_2 :	<*p_small*, *p_small*>	c :	<*p_small*, *p_medium*>.

Graphically, this immediate successor of the initial state is shown in figure 5.12 together with those spurious successor states generated by the possible state transition rules.

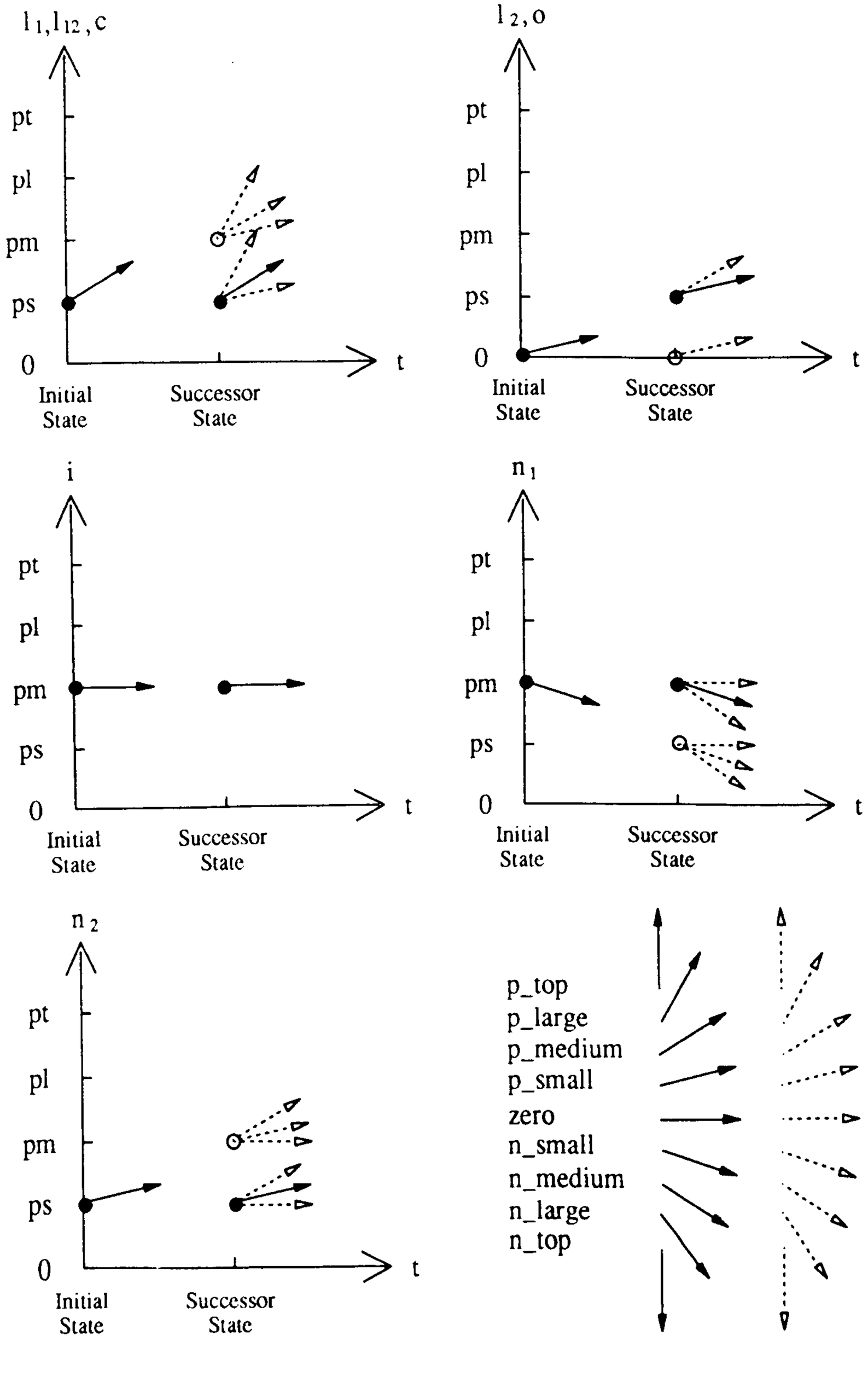


Fig. 5.12. *A Step-ahead Simulation of Two Coupled Tanks*

Within figure 5.12, the solid circles (representing the actual magnitudes) with the real arrows (representing the actual rates of change) denote the initial state and the above state, whilst both the empty circles (representing pseudo magnitudes) with the dotted arrows (representing pseudo rates of change) and the solid circles with the dotted arrows denote the spurious states. The detailed implications of the arrows are given beside the diagram for variable n_2. Also, for simplicity, within this figure, *pt*, *pl*, *pm*, and *ps* indicate the positive qualitative values of the magnitudes: *p_top*, *p_large*, *p_medium*, and *p_small*, respectively.

Clearly, this sub-section has concentrated on the ways that *FuSim* eliminates spurious behaviours. The following sub-section will pay more attention to the generation of a complete state sequence with associated temporal durations. In addition, the flexibility in modelling physical systems provided by the fuzzy representation will be shown as well as the ability that the associated simulation technique can considerably reduce the generation of spurious behaviours.

5.6.2. Example 2: A Mass on a Spring

Let us now consider another example: the system consisting of a mass hung on a spring, as shown in figure 5.13. This system includes three variables, each of which is associated with a normalised numerical range [−1, 1] from which the variables take values:

x -- the displacement of the mass from its rest point,

v -- the velocity of the mass, and

a -- the acceleration of the mass.

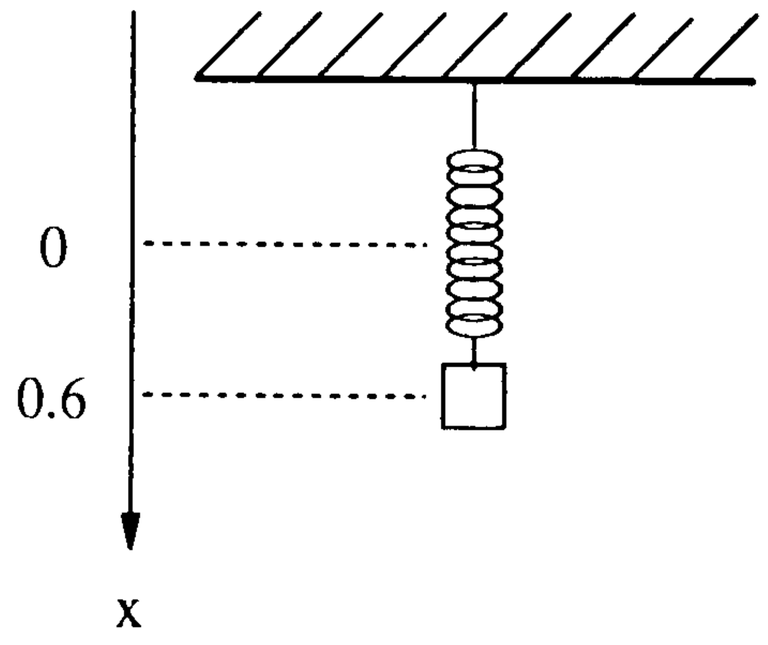


Fig. 5.13. *A Mass on A Spring*

The following set of 4-tuple parametric fuzzy numbers was chosen to form the fuzzy quantity space:

$$Q_F = \{[-1, -0.7, 0, 0.1], [-0.6, -0.6, 0, 0], [-0.5, -0.1, 0.1, 0.1],$$
$$[0, 0, 0, 0], [0.1, 0.5, 0.1, 0.1], [0.6, 0.6, 0, 0], [0.7, 1, 0.1, 0]\},$$

with respective names that corresponds to the perceived meaning:

$$Q_F = \{n_top, n_medium, n_small, zero, p_small, p_medium, p_top\}.$$

Notice that, here, the real number 0.6 is deliberately chosen to denote the concept *medium*, just to reflect the flexibility of the 4-tuple parametric representation of qualitative values within *FuSim*. For sake of notational simplicity, the Q_F is represented by

$$Q_F = \{-t, -0.6, -s, 0, s, 0.6, t\}.$$

The physical constraints in the system can be characterised as follows

$$deriv\ x = v,$$

$$deriv\ v = a,$$

$$\begin{bmatrix} a \tilde{} x & -t & -0.6 & -s & 0 & s & 0.6 & t \\ -t & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ -0.6 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ -s & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ s & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0.6 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ t & 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

The first and second equations establish the ordinary derivative relationships holding amongst the distance, velocity, and acceleration of the mass. The third functional relation between a and x is a weak, but stronger than monotonic operator, form of Hooke's law represented in a fuzzy relation.

Suppose that the initial states of the variables x, v, and a are <0.6, 0>, <0, –0.6>, and <–0.6, 0>, respectively. That is, the mass is moved away from the equilibrium point, $x = 0$, to $x = 0.6 > 0$, and then let go. The Waltz algorithm is used to check that such an initial state is indeed valid with respect to the above description of the fuzzy system model. The *FuSim* algorithm then takes this model and the initial condition as the input, and starts the simulation of the system. For this particular system, we are not concerned about the possible behaviour of the rate of change of the variable a. Thus, behaviours only different in the possible values of the rate of change of a can be treated as a single behaviour. By this, *FuSim* produces a unique behaviour of the system as follows, given in the form of the combination of the states of the system variables (x, v, a):

$$(<0.6, 0>, <0, -0.6>, <-0.6, 0>)$$

$$\rightarrow \quad (<s, -s>, <-s, -s>, <-s, s>)$$

$$\rightarrow \quad \left\{ \begin{array}{l} (<0, -0.6>, <-0.6, 0>, <0, 0>) \\ (<0, -0.6>, <-0.6, 0>, <0, s>) \\ (<0, -0.6>, <-0.6, 0>, <0, 0.6>) \end{array} \right\}$$

$$\rightarrow \quad (<-s\,,\ -s>,\ <-s\,,\ s>,\ <s\,,\ s>)$$

$$\rightarrow \quad (<-0.6,\ 0>,\ <0,\ 0.6>,\ <0.6,\ 0>)$$

$$\rightarrow \quad (<-s\,,\ s>,\ <s\,,\ s>,\ <s\,,\ -s>)$$

$$\rightarrow \quad \left\{\begin{array}{l}(<0,\ 0.6>,\ <0.6,\ 0>,\ <0,\ -0.6>)\\(<0,\ 0.6>,\ <0.6,\ 0>,\ <0,\ -s>)\\(<0,\ 0.6>,\ <0.6,\ 0>,\ <0,\ 0>)\end{array}\right\}$$

$$\rightarrow \quad (<s\,,\ s>,\ <s\,,\ -s>,\ <-s\,,\ -s>)$$

$$\rightarrow \quad (<0.6,\ 0>,\ <0,\ -0.6>,\ <-0.6,\ 0>),$$

and then, the above cycle repeats. This is shown graphically in figure 5.14 (one cycle only), where the solid circles denote the magnitudes and the arrows denote the rates of change.

It is very important to notice that, within this figure, the temporal points $(t_0, t_1, \ldots, t_8)$ satisfy the following expressions, calculated with respect to the computational method presented in section 5.4.2.1:

$$t_0 = 0, \quad t_1 \in [1, 11], \quad t_2 \in [1.09, 12], \quad t_3 \in [2.09, 23] \quad t_4 \in [2.18, 24],$$

$$t_5 \in [3.18, 35], \quad t_6 \in [3.27, 36], \quad t_7 \in [4.27, 47], \quad t_8 \in [4.36, 48].$$

Thus, with *FuSim*, we can determine the time, even though not in an exact form, when system state stays within a new state. In fact, a sequence, $\{\Delta T_{p_0}, \Delta T_{a_1}, \Delta T_{p_1}, \Delta T_{a_2}, \Delta T_{p_2}, \cdots\}$, of the persistence times and arrival times of the system states can be generated as follows:

$$\{0,\ [0.09, 1],\ [0.91, 10],\ [0.09, 1]\}_i, \qquad i = 1, 2, \cdots.$$

From this, we have that

$$\begin{aligned} t_1 &= t_0 + \Delta T_{a_1} + \Delta T_{p_1} \\ &\in [0, 0] + [0.09, 1] + [0.91, 10] = [1, 11], \end{aligned}$$

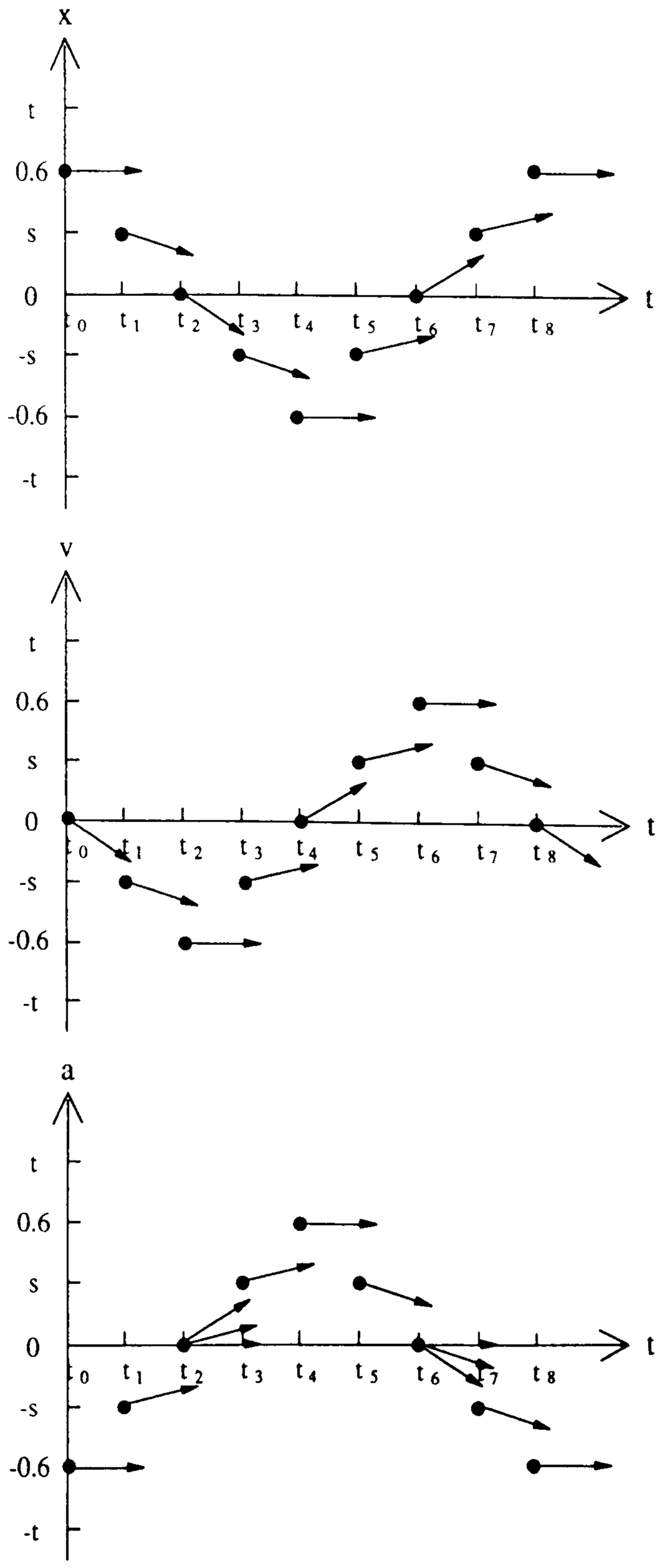


Fig. 5.14. *A Cycle of the Behaviour of 'a Mass on a Spring' System*

and that

$$t_2 = t_1 + \Delta T_{a_2} + \Delta T_{p_2}$$
$$\in [1, 11] + [0.09, 1] + [0, 0] = [1.09, 12].$$

Other time indices t_n, $n > 2$, are similarly determined.

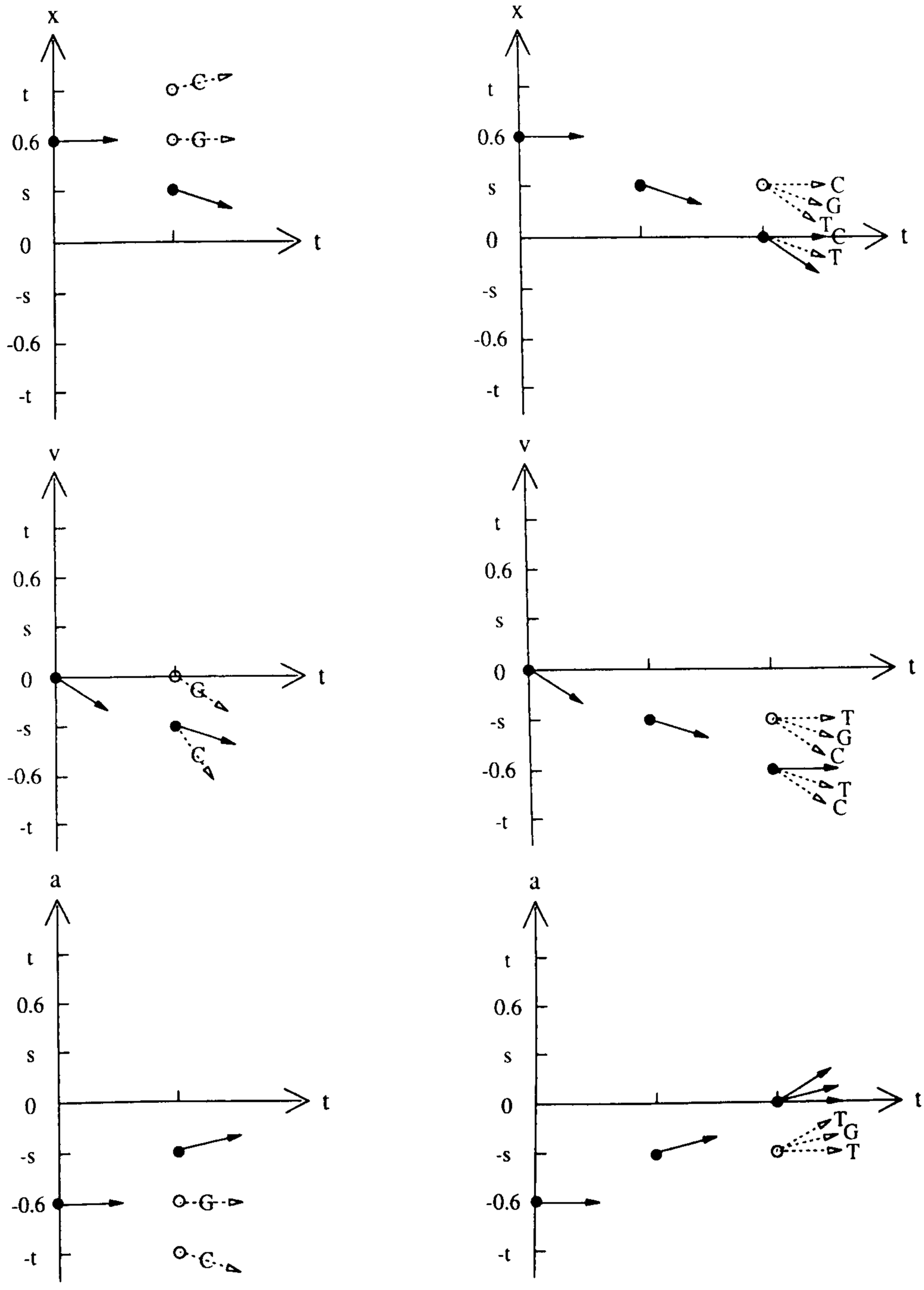
Consider two detailed samples for obtaining the above persistence and arrival times. Firstly, following the rule to calculate the arrival time, the time cost for the system to transition from the initial state to its successor ($<s, -s>$, $<-s, -s>$, $<-s, s>$) lies in the range [0.09, 1]. For instance, the variable a changes from $<-0.6, 0>$ to $<-s, s>$, taking the arrival time

$$\Delta T_{a_1}(v) \in \frac{-0.5 - (-0.6) + (0.5 - 1)(0.1 + 0)}{|[0.1, 0.5, 0.1, 0.1]|_{0.5}} = [0.09, 1].$$

Secondly, the persistence time for the system (x, v, a) to remain within state ($<s, -s>$, $<-s, -s>$, $<-s, s>$) lies in the range [0.91, 10]. For example, the variable x stays within the state $<s, -s>$ for the following duration, determined by the rule of calculating persistence times:

$$\Delta T_{p_1}(x) \in \frac{0.5 - 0.1 + (1 - 0.5)(0.1 + 0.1)}{|[-0.5, -0.1, 0.1, 0.1]|_{0.5}} = [0.91, 10].$$

As a complete example, figure 5.15 shows how *FuSim* produces the behaviour of the system from the initial state to its first two successors, i.e., the first 1/4 period of the complete cycle of the behaviour illustrated in figure 5.14. The one step-ahead behaviour from the initial state is depicted in figure 5.15.a and the two step-ahead simulation results are given in figure 5.15.b. Within these diagrams, those states marked with either of the signs *C*, *T*, and *G* denote the spurious states eliminated by either fuzzy constraint filtering or temporal filtering or simple global filtering (no-change and repeating), respectively. In addition, the solid or empty circles and the real or dotted arrows represent the same implications as those presented in figure 5.12.



(5.15.a) (5.15.b)

Fig. 5.15. *Generation and Filtering of the First* 1/4 *Period of Behaviour*

The elimination of the spurious states within figure 5.15.a is explained as follows. The pseudo new state $<t, s>$ of the variable x is filtered by the constraint: *deriv* $x = v$, since there does not exist a new state of the variable v that takes s as its

qualitative magnitude. Having removed the $<t, s>$ of x, the spurious successor state $<-t, -s>$ of the variable a can be deleted by the function relational constraint. It is important to notice that conventional monotonic decreasing constraint cannot eliminate this state (even if the corresponding value (0, 0) is known) due to the fact that, for variables (a, v), the pair $(<-t, -s>, <s, -s>)$ transitioned from (<–0.6, 0>, <0.6, 0>) satisfies $M_0^-(a, x)$. After this, the $<-s, -t>$ of v can be ruled out based on the pair-wise filtering criterion. The system state (<0.6, 0>, <0, –0.6>, <–0.6, 0>) for all the variables (x, v, a) is deleted by using the no-change filter and marked with G within the diagram for each variable. The only immediate successor state of the initial state is, therefore, the following:

$$(<s, -s>, <-s, -s>, <-s, s>).$$

By chance, no temporal filtering is required for removing these pseudo states at this stage.

Temporal filtering, however, plays a significant role in the reduction of the spurious transitions from the above first successor of the initial state. In fact, it can be seen from figure 5.15.b that, after using constraint filters, some spurious states still remain. Fortunately, most of them are eliminated by the use of temporal filters. For instance, the system state $(<0, -s>, <-s, 0>, <0, 0>)$ is a pseudo next state since, within the present state $(<s, -s>, <-s, -s>, <-s, s>)$, the persistence time of each variable lies in [0.91, 10] (see the detailed calculation given previously for example); whilst the arrival time for v to reach this state is 0 (an R-transition), but the time taken for x and a to transition is

$$\frac{0.1 - 0 + (0.5 - 1)(0.1 + 0)}{[0.1 + 0.1(0.5 - 1),\ 0.5 + 0.1(1 - 0.5)]} = [0.09,\ 1].$$

Hence, such a transition invalidates the temporal filtering rule and is deleted. As

shown in the figure, the remaining few pseudo states are removed by using two simple global filters. The only surviving successor states of the system state ($<s, -s>, <-s, -s>, <-s, s>$) that pass all three kinds of filters are the following:

$$(<0, -0.6>, <-0.6, 0>, <0, 0>),$$

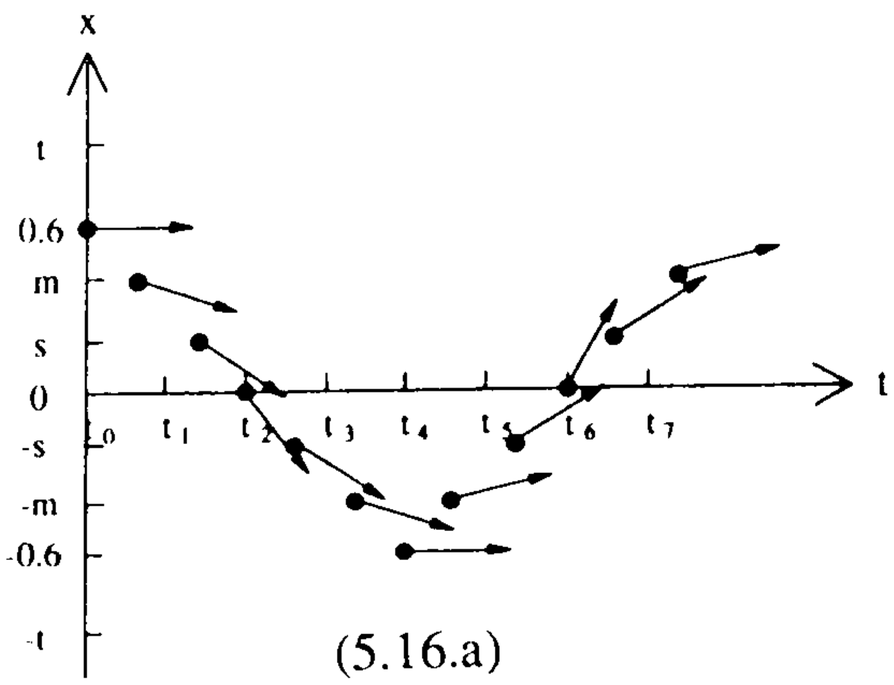
$$(<0, -0.6>, <-0.6, 0>, <0, s>),$$

$$(<0, -0.6>, <-0.6, 0>, <0, 0.6>).$$

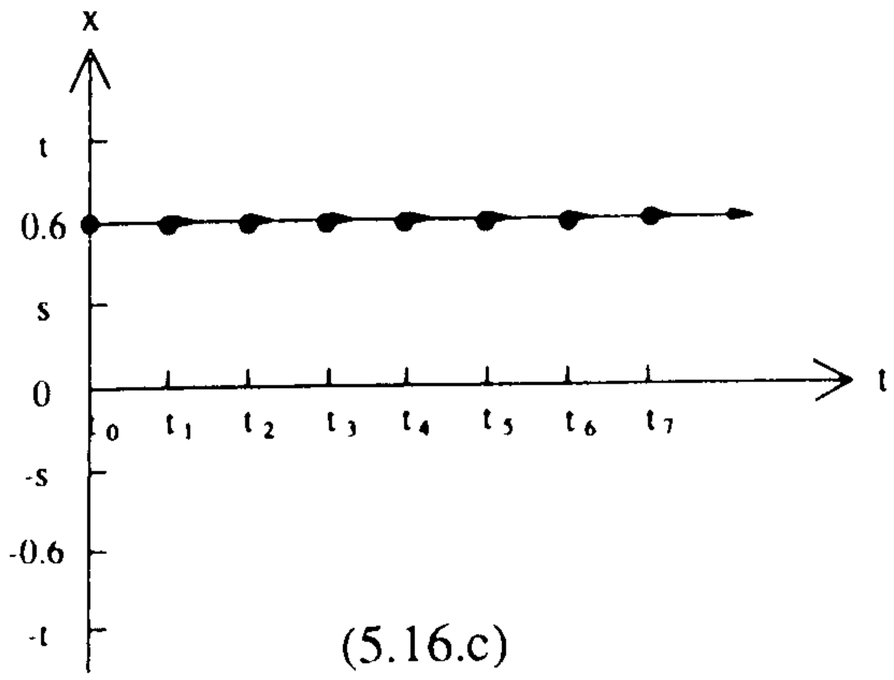
For this 'mass on a spring' system, we are not concerned about the rate of change of the variable a, these three 'different' states can, therefore, be treated as a single one. The path from the initial state, through ($<s, -s>, <-s, -s>, <-s, s>$), to one of these three states forms the first 1/4 period of the behaviour cycle given in figure 5.14. Other 3/4 period of the complete cycle was obtained in a similar way.

From the behaviour obtained through *FuSim*, it is clear that the mass will oscillate between the maximum amplitudes 0.6 and –0.6 forever, with a period belonging to the interval [4.36, 48] as computed previously.

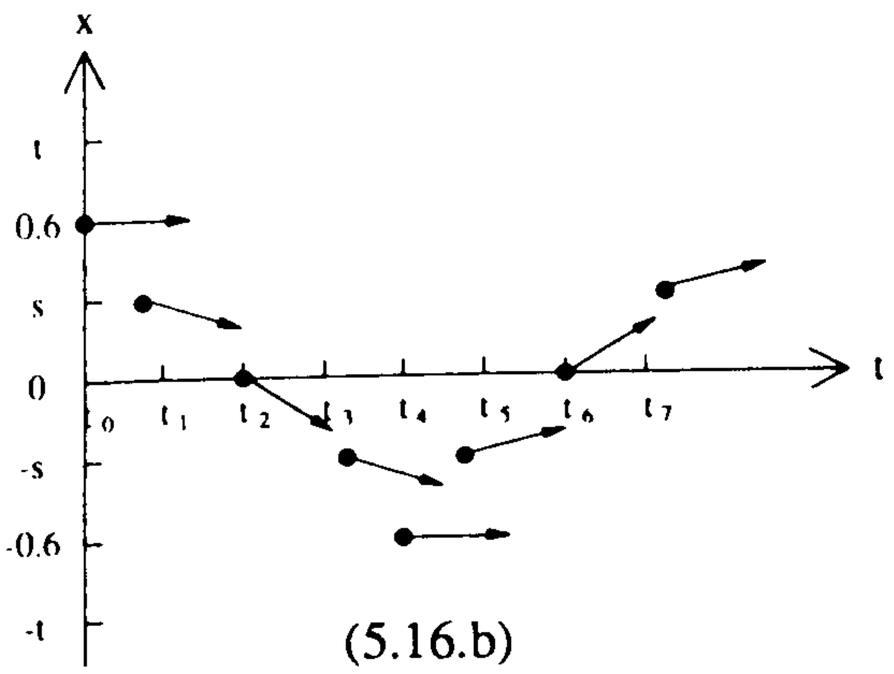
As pointed out before, we have identified different modelling dimensions (see section 7.3.2 for detailed definition) that may be modified in representing a fuzzy model. This is achieved, informally, through either of the following four channels: changing the cardinality of a quantity space, varying the membership distributions of the qualitative values in the space, enlarging or reducing the number of variables involved, and adjusting the strength relations between the variables. For the 'mass on a spring' system, a systematic investigation into these dimensions within the fuzzy qualitative simulation has been carried out [64] and the major results are presented below and illustrated in figure 5.16 (for the variable x only).



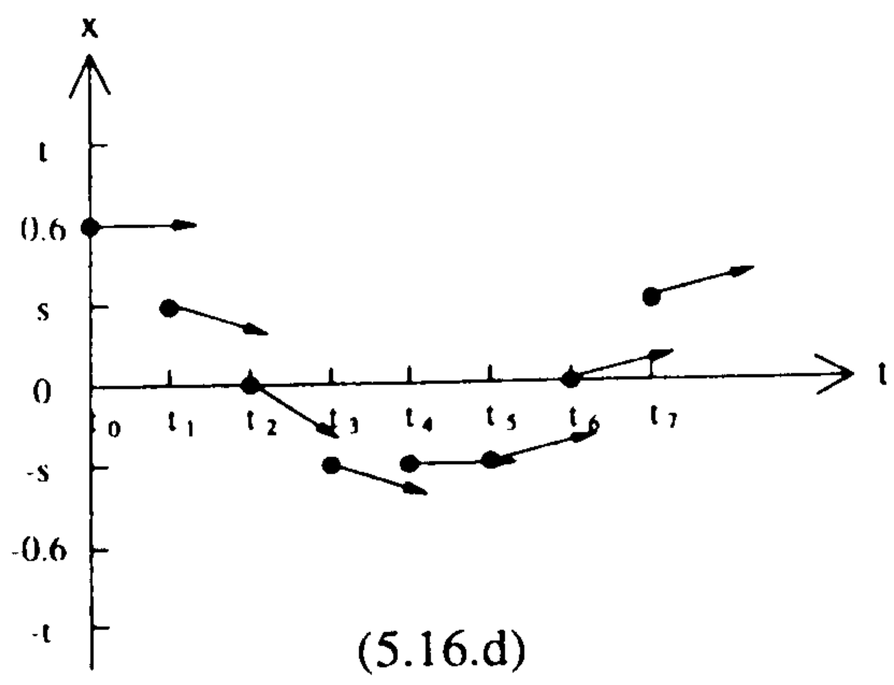
(5.16.a)



(5.16.c)



(5.16.b)



(5.16.d)

Fig. 5.16. *Behaviours of Modified Models*

When performing these simulations, the initial state of the system is assumed to be the same as that used previously. It can be seen from the figure that *FuSim* produces a unique but different behaviour in response to each different modification. Within these behaviours, the time indices used are those presented in figure 5.13 so as to ease the comparison between them. However, it is necessary to point out that the time labels computed with respect to each particular modification have their own possible ranges. That is, for example, the range of the absolute time of the second new state in figure 5.16.b need by no means be identical to t_2 but the two indices do intersect with each other. It is only due to the qualitative representation and the intersec-

tion that they happen to appear at the same place. Simple explanation of the simulation results is described as follows. Figure 5.16.a shows the behaviour obtained from using a denser cardinality of the quantity space ($\{-t, -0.6, -m, -s, 0, s, m, 0.6, t\}$) than that given earlier. Figure 5.16.b presents the result from modified membership distributions (notice the changes of particular time intervals while the period keeps qualitatively unchanged. Again, the period herein falls within a different time interval but intersect with that in figure 5.13). Figure 5.16.c gives the behaviour when the variables v and a vanish, indicating a mass-stuck condition. Figure 5.16.d depicts the resulting behaviour from introducing two more fuzzy relations between a and x and a and v, representing a *medium* friction condition. These results, from one aspect, show the flexibility that *FuSim* brings to system-modelling and the advantages to the Qualitative Simulation task in itself.

Finally, it is necessary to point out that all the simulation results presented above were obtained when the coefficient α was chosen to be 0.5. As indicated earlier, this does not affect the basic simulation procedures of the *FuSim* algorithm but only the temporal durations calculated from the fuzzy states. To demonstrate the robustness of *FuSim*, it is interesting to see what behaviour the algorithm will generate when using the same quantity space but a different α value. For this purpose, the *FuSim* algorithm was also used to simulate the 'mass on a spring' system with a coefficient α equaling 0.7. Starting with the same initial state as before, it generates, again, only one unique behaviour which has an identical appearance as that shown in figure 5.14. The graphical presentation of the simulated behaviour is, therefore, omitted herein. However, it should be noticed that, the possible ranges within which the absolute time indices associated with each qualitative state may fall, now, are narrower than those given in figure 5.14, as presented in the following:

$t_0 = 0$, $t_1 \in [1, 7.57]$, $t_2 \in [1.13, 8.57]$, $t_3 \in [2.13, 16.14]$ $t_4 \in [2.26, 17.14]$,

$t_5 \in [3.26, 24.71]$, $t_6 \in [3.39, 25.71]$, $t_7 \in [4.39, 33.28]$, $t_8 \in [4.52, 34.28]$.

This, of course, is expected since, in principle, both persistence time and arrival time are fuzzy numbers and hence are convex fuzzy sets, a higher α value certainly results in narrower (or, for crisp intervals, the same) α-cuts of such fuzzy sets. This example, together with those presented earlier, indicates that the sensitivity of the *FuSim* algorithm is not high to the choice of the coefficient α. It is suggested that, when dealing with a particular simulation task, this coefficient is best made with respect to the certainty about the quantity space and the model utilised for the simulation. In the extreme, its value may be subjectively assigned provided that it is larger than the maximum membership value of all the crossing-points within the quantity space.

5.7. Summary

Based on the representation of a fuzzy quantity space and that of fuzzy modelling primitives developed in the preceding chapter, this chapter establishes the fuzzy qualitative simulation algorithm, *FuSim*, which is able to generate a qualitative description of the behaviour of physical systems. This algorithm describes a state of a physical variable in terms of a pair of qualitative magnitude and qualitative rate-of-change with the persistence time of the state being indicated. Following the continuity and differentiability assumption, the algorithm automatically produces possible state transitions from a given state and, meanwhile, computes the (arrival) times taken for such transitions. To minimise theoretically unavoidable qualitative ambiguity, it employs three different filters over the possible transitions, i.e., the constraint, temporal, and global filters. In particular, the function relational constraint filtering and the temporal filtering can eliminate a large number of spurious behaviours. Distinct from other *QR* techniques, when *FuSim* generates a system behaviour it also generates

the temporal duration associated with each state. This provides a significant advance in exhibiting the evolution of system state, by showing not only the temporal ordering amongst different states but also the extent of the durations of these states.

To conclude the development of the fuzzy qualitative simulation technique, we should like to emphasise that the essential limitations of current approaches to qualitative reasoning about physical systems, as presented in chapter 1, have been to a great extent addressed and reduced in *FuSim*. In order to further evaluate the performance of the *FuSim* algorithm, it is utilised as the behaviour simulator within a novel model-based diagnostic system. This is presented in the following chapters.

CHAPTER 6

BACKGROUND AND ARCHITECTURE OF MODEL-BASED DIAGNOSTIC SYSTEMS

6.1. Introduction

The development of effective qualitative simulation techniques is intrinsically important, however, the real advantage comes when such techniques are used in application systems. A major potential application domain of Qualitative Simulation is to utilise a particular simulation algorithm to predict the behaviour of a physical system within *M*odel *B*ased *D*iagnostic *S*ystems (*MBDS*). Although well established using numerical approaches to system modelling [47, 55, 71, 77], model-based diagnosis is currently flourishing within the Artificial Intelligence community. The use of *AI* approaches to system modelling and reasoning with incomplete and uncertain knowledge has resulted in a number of general and powerful diagnostic systems [26, 27, 82, 109]. However, as yet there is no generic architecture available for this task. It is important, therefore, to design a basic framework that sets a common basis for various approaches, easing the comparisons amongst them and, more significantly, allowing possible future improvements to be developed within a particular technique to be utilised by other approaches.

From this viewpoint, this chapter first briefly reviews the motivations for developing *MBDS* by showing the advantages that such systems could have over the first generation (rule-based) diagnostic systems. Then, basic definitions of static and dynamic systems, together with a clarification of concepts like time-varying and time-dependent, are provided so as to classify different *MBDS* techniques in the

following discussion. Section 6.4 presents an overview of typical approaches to *MBDS* for static systems, while section 6.5 outlines an existing approach for dynamic systems and discusses significant limitations within this method. Based on these, a general architecture of *MBDS* is proposed and two particular instantiations of the architecture, chosen from both static and dynamic approaches, are concisely revisited to demonstrate its generality. This architecture, together with the *FuSim* algorithm established in the previous chapters, will be used in the next chapter to develop a novel *MBDS* technique for diagnosing continuous dynamic systems which overcomes many shortcomings of the existing techniques discussed in sections 6.4 and 6.5. The work presented in this chapter utilises some of the basic terminology employed in *ESPRIT* project (P5143), named *ARTIST*, to which the author has been contributing since the project was launched in November 1990 [40, 60].

6.2. Motivations for Constructing Model-Based Diagnostic Systems

It is well known that diagnosis has been one of the major applications of first generation knowledge-based systems that are mainly characterised by the rule-based paradigm. Such developed diagnostic systems may be very successful in finding faults when applied to physical systems from which extensive operational experience has been learnt and stored in a (large) data base. A rule-based system is therefore dedicated to applications where the basic diagnostic knowledge utilised is represented in terms of the empirical symptom-failure associations. In other words, the rule-based approaches to diagnosis are usually restricted to familiar systems with known symptoms and faults. Such approaches are usually anti-causal. The structural properties of the physical system under consideration are only implicit and hence, they are difficult to support or automate the development of a diagnostic system based on fundamental theoretical principles. However, for many actual physical systems, a large amount of

theoretical design knowledge is usually available and may be very useful for diagnosis. Unfortunately, the rule-based systems fail to provide a foundation for using such knowledge without experiencing particular faults or foreseeing potential failures in the physical systems to be diagnosed. In addition, the construction and maintenance of diagnostic systems using large data bases are usually rather expensive because of the inflexibility and informality of the underlying techniques [10, 16].

Having established these drawbacks of the first generation diagnostic systems, intensive investigation into the development of *MBDS* using *AI* techniques has been carried out since the middle seventies, with a considerable growth of interest within the past five years [9, 10, 17, 20, 26, 27, 42, 82, 109]. The central idea of the *MBDS* approach is the use of an explicit model of the structure of the physical system to be diagnosed. Such models can reflect the physical structure of the system and can be used to generate predictions of the behaviour that can be compared with the observations from the physical system or amongst themselves. A discrepancy between the observed and predicted behaviours, or that between the predictions driven by the observations, implies an incorrectly working or faulty system. The *MBDS* systems then isolate and/or identify changes to the model that can remove the discrepancies discovered and, therefore, establish the fault(s) in the physical system.

Significant properties of the *MBDS* approaches to diagnosis have been reported in [18, 106]. In fact, the *MBDS* systems contrast favourably with traditional rule-based ones with a number of advantages as listed in the following:

1) Allowing diagnosis to be performed independently of the detailed representation of a physical system, at most only depending on whether the system is a static or dynamic one;

2) Enabling diagnosing a whole class of physical systems, built upon a set of commonly used components within an application domain, rather than a particular system;

3) Making it possible to diagnose systems where little or even no operational experience is available;

4) Easing the redesign and maintenance of the diagnostic system, thereby considerably increasing economical efficiency.

Realising these, numerous model-based diagnostic systems have been built up, though almost all of these systems are restricted to static models of physical systems.

6.3. Static and Dynamic Models

It is clear that, within *MBDS*, system models play the central role. However, different diagnostic techniques may require different classes of system models. For a given physical system to be diagnosed, it is, therefore, essential to select a right model which represents those attributes of the system that a particular technique can utilise to accomplish the corresponding diagnostic task. Unfortunately, in the literature of *MBDS*, there lacks a clear classification of system models and exists a confusion in the terminology amongst the terms like dynamic, time-dependent, and time-varying. Thus, it is necessary to clarify the use of such terms and, in particular, to emphasise the difference between a static model and a dynamic one so that the discussion presented in the following sections can be better understood.

It is important to notice that, all model-based approaches to diagnosis fundamentally involve reasoning about the behaviour of physical systems. While such behaviour, in general, consists of two components reflecting two different responses due to the system's internal state and the exogenous inputs, respectively. These two

different responses can be utilised as a basic criterion to base a useful classification of system models, which is suitable for selecting models within *MBDS* with respect to the (approximated) behaviours of the actual physical systems to be diagnosed.

For most physical systems the inputs, outputs and internal variables are explicit functions of time, even if they are simply constant functions. In general, the present value of a system's output (behaviour) is a result of what has happened in the past as well as what is currently affecting it. This is because physical systems transfer energy between subsystems and energy cannot be transmitted instantaneously. All physical systems therefore store, or delay, a certain amount of the energy that are transferring. As such, all physical systems are known to be dynamic. However, in many cases the energy storage is minimal and inputs solely affect the system's response. In this case the system is said to be static, although this must be an approximation.

Concisely speaking, in a *dynamic model*, the outputs depend on both past and present values of the inputs. A dynamic model is said to possess memory, representing energy storage, because of this dependence on past history. Continuous dynamic systems are usually described by the integro and/or differential equations relating the chosen variables. The energy storage within such systems is represented by the current values of the integrals or by derivative operators via the rate of change of the system variable. In addition, dynamic systems with finite states can be expressed by models in terms of difference equations. A *static model* is one for which the present values of the outputs depend only on the present values of the inputs, and is said to be memoryless and instantaneous. In this case the operators relating the system variables are independent of time and can, therefore, only represent the behaviour at steady state. Continuous static models are normally represented by algebraic equa-

tions and static systems with finite states can be expressed by discrete algebraic equations. For instance, a series *RC* circuit is usually modelled by a continuous dynamic model to reflect the energy storage in the capacitor *C* and also the continuity in the charge or discharge of *C*. However, a digital full adder is often modelled by a static model since the output (the sum) of the adder is only a function of both present inputs.

Both dynamic and static models can be further sub-classified into two sub-groups by considering if the shifting of the time origin will affect a system's behaviour. If so, the model is known as *time-varying* one; otherwise, it is said to be *time-invariant*. For example, if, within the above *RC* circuit, the resistance *R* of the resistor varies significantly against the environment temperature (e.g., a thermistor), this circuit should then be modelled by a time-varying model due to the fact that the temperature is a function of time and, hence, so is the resistance *R*. On the contrary, this circuit exhibits time-invariant property if the variations in the parameter values of both *R* and *C* can be ignored.

Another important concept is the so-called *time-dependent* behaviour which states that the system behaviour is changing in response to a time-varying input. It must be clear that any system, whether dynamic or static and time-varying or time-invariant, may exhibit such time-dependent behaviour. For instance, when a circuit consists of only ideal resistors, a typical example for static time-invariant system, is presented by an *AC* current as input, the output of the system does change due to the change of the input at the same time. However, if a fixed value of current is applied to this system, the output will always be the same regardless of when the input is applied. It is crucial to recognise the difference between the time-dependent response of a static system and the transient response of a dynamic system. This is a source of

much confusion within the *MBDS* literature. For a time-invariant dynamic system, even if presented with a constant input, its response may appear rather different over the time axis. Nevertheless, no matter when such an input is presented, the response will, of course, always have an identical overall appearance because of assumed time-invariant characteristics.

Although definitional, the distinctions given above are important in that it is anticipated that the selection or possible combination of various *MBDS* techniques will depend upon whether dynamic or static models, with, perhaps, time-invariant or time-varying characteristics, of the physical system are to be used. This will become clear through the following discussion about the existing techniques for *MBDS*.

6.4. Model Based Diagnosis of Static Systems

In general, two fundamental classes of *MBDS* systems can be identified, determined by the basic models of the physical systems, static or dynamic, that a particular diagnostic technique employs. This section concentrates on the *MBDS* techniques developed for diagnosing physical systems modelled by algebraic equations, a class we call model-based diagnosis of static systems. The next section will discuss the other class, namely, model-based diagnosis of dynamic systems.

6.4.1. Normal-Behaviour Model Based Approaches

By normal-behaviour model based approaches to diagnosis we mean those based on models of the intended designed structures, with desirable behaviours, of the physical system to be diagnosed. There have been many such diagnostic systems built for dealing with situations where all discrepancies can be explained by a single component being faulty [10, 17, 19, 20, 42]. For example, by modelling the intended behaviour of each component as a constraint, and the complete system as a set of

interconnecting constraints, a *constraint suspension* technique (named *HT* [17, 19]) can determine if a component is faulty in the following way. It simulates a reduced constraint set, formed by temporarily removing the associated constraint of the suspected component, to test whether the behaviour of the reduced system matches the observations. If so, *HT* indicates that the removed component is the culprit. Otherwise, this component can be exonerated. However, since it propagates values through one component at a time by solving one equation with one unknown *HT* fails whenever problems as simple as solving two equations in two unknowns are encountered. Unfortunately, such situations commonly exist within even slightly complicated systems, e.g., systems having feed-forward or feedback.

An alternative approach, named *DART* [42], was developed by representing knowledge of system's structure and behaviour as logical propositions and inferring through a variation of resolution, so-called *resolution residue*, that works as a direct proof procedure rather than using a refutation technique. Such a general inference procedure allows *DART* to deduce which components are suspect and how to reduce the suspect set to a singleton. Actually, the resolution residue is a logically complete representation and inference mechanism, nevertheless, it usually incurs a significant computational cost. Furthermore, completeness of the inference procedure provides no guarantee for completeness of a predictor used to generate the model behaviour [26]. For instance, generating exact behaviour of a continuous (dynamic) system often requires solving non-linear (higher-order differential) equations, whilst there does not exist a general method to solve such equations. Completeness can only be attained to a certain degree of approximation.

Whilst *HT* and *DART*, together with other researches reported in [9, 10, 20], are important contributions to *MBDS*, they deal with single faults only. A technique,

called the *General Diagnostic Engine* (*GDE*) [26], has been established to extend these approaches by diagnosing systems with multiple failed components. Recognising the fact that the space of suspected components grows exponentially with the number of faults simultaneously under consideration, the *Assumption-based Truth Maintenance Systems (ATMS)* [21, 22] is utilised to record the dependencies of inference in order to determine conflicts effectively. Actually, *GDE* generates and maintains only a minimal suspected set (consisting of so-called *minimal candidates*), thereby saving a substantial amount of work. While the fundamental exponential problem has not been changed, its effect has been reduced, enabling *GDE* itself to handle systems larger than might otherwise have been possible. In addition, *GDE* also offers a degree of domain-independence since its diagnostic procedure is separated from the behaviour-predicting procedure. When coupled with a behaviour simulator, i.e., an inference engine, *GDE* exhibits powerful diagnostic ability and, therefore, is commonly believed to be the most influential work in the research area of model-based diagnosis.

A similar technique for diagnosing multiple faults by using first-order logic to represent systems was independently developed in [82], though, as pointed out there, many other logics can also be used as the representation language. This can be seen as a reconstruction of the covering set theory presented in [81]. For simplicity we call this approach *HIT* to acknowledge the *hitting sets* [82], since the method for computing minimal suspected set (called *diagnoses*) is based on the determination of minimal hitting sets. It is worth pointing out that the original *HIT* algorithm failed to accurately follow the description of how to compute the minimal hitting sets given in [82]. Fortunately, the problem thus resulted has been completely resolved [43]. Essentially, both *HIT* and *GDE* provide an equivalent theoretical framework for

MBDS. By understanding that a *conflict* is a set of components which cannot all be functioning correctly, the definitions of diagnosis and conflict in *HIT* are equivalent to those of minimal candidates and conflict in *GDE*, respectively. Further, the so-called theorem prover of *HIT* corresponds roughly to the inference engine of *GDE*. However, three major differences between them exist. First, *GDE* is derived from sound intuitions but lacks a formalisation of the theory, while *HIT* provides formal definitions and proofs. Second, *GDE* goes beyond *HIT* by providing a method for computing probabilistic information to identify appropriate system measurements to make next [23], though there is no reason that such measurement suggestions could not be supplemented or replaced by other knowledge sources [109]. Last, and the most important, *GDE* utilises *ATMS* to obtain great efficiency in candidate generation by only considering the minimal conflict sets, however, *HIT* does not address this technique since it does not require the minimal conflicts to be provided before computing the hitting sets. Nevertheless, there is no reason why an *ATMS* cannot be used within the *HIT* technique to increase its diagnostic efficiency.

Seen from a general point of view, the above approaches can be grouped into a unified technical framework that we can call the *dependency-recording* based approach to model-based diagnosis. The diagnostic task performed within this framework can be explained in the following. Based on the normal-behaviour model of the system to be diagnosed, an inference algorithm generates the predictions of the system's behaviour from obtained observations. Meanwhile, a specific recorder maintains the dependencies of these predictions in order to determine which basic assumptions used for modelling the system have resulted in any inconsistencies or conflicts. Within such approaches, the system's model is usually built by an object-based language [25, 100] in conjunction with a constraint propagator [110]. The constraint

system draws inferences from relations amongst system variables, without distinguishing the directionality of the relations when propagating the values of the variables. The dependency-recording can be realised by the *ATMS*, which is able to construct a dependency network with respect to the performed inferences and labels each present conclusion with a minimal set of assumptions that have enabled the inferences so far. Whenever a discrepancy is discovered, i.e. a variable has two different values at a time, a minimal set of conflicting assumptions will then be generated. From such conflict sets, a minimal set of diagnostic candidates (containing at least one element out of each conflict) are further determined through a compiled algorithm that, mathematically, is equivalent to computing the minimal hitting sets [82]. Such an approach, therefore, directly relates discrepancies and candidates in a fixed way, combining discrepancy detection (a decision-making procedure about inconsistencies between observations and predictions or amongst different predictions) with candidate generation (another decision-making procedure about possible faulty components that underlay the inconsistencies). It offers great efficiency to diagnose, in parallel, various potentially conflicting situations, including unforeseen ones.

The dependency-recording techniques, however, do not explicitly exploit the discrepancies detected. The information, for instance a measure of the differences between the generated discrepancies, which could otherwise be gained for improving diagnosis through explicit feedback from the discrepancy detection mechanism to the behaviour-predicting mechanism has been neglected. Such information can be very useful to propose potential faults by adjusting the system model and then evaluating the adjusted model(s) to check if the predictions from the model(s) match the observations such that the original discrepancies have been eliminated or to check if the discrepancies have at least been reduced. Unfortunately, it cannot be utilised by the

dependency-recording based diagnostic systems. As depicted by figure 6.1, due to the direct combination of discrepancy detection and candidate generation, such systems lack an explicit mechanism (shown in the dotted-line part) to automatically modify the system model that has resulted in the discrepancies. The diagnostic system building in this way, therefore, appears as an 'open loop' system. It should be pointed out that, due to the efficient and effective utilisation of the *ATMS*, a general sketch of dependency-recording based systems is very difficult to make. In fact, the separation of the Model and the Discrepancy Detector within figure 6.1 is conceptually made since, when implementing such a system, the discrepancy detection (or conflict recognition) is performed within the *ATMS* without using an independent discrepancy detector.

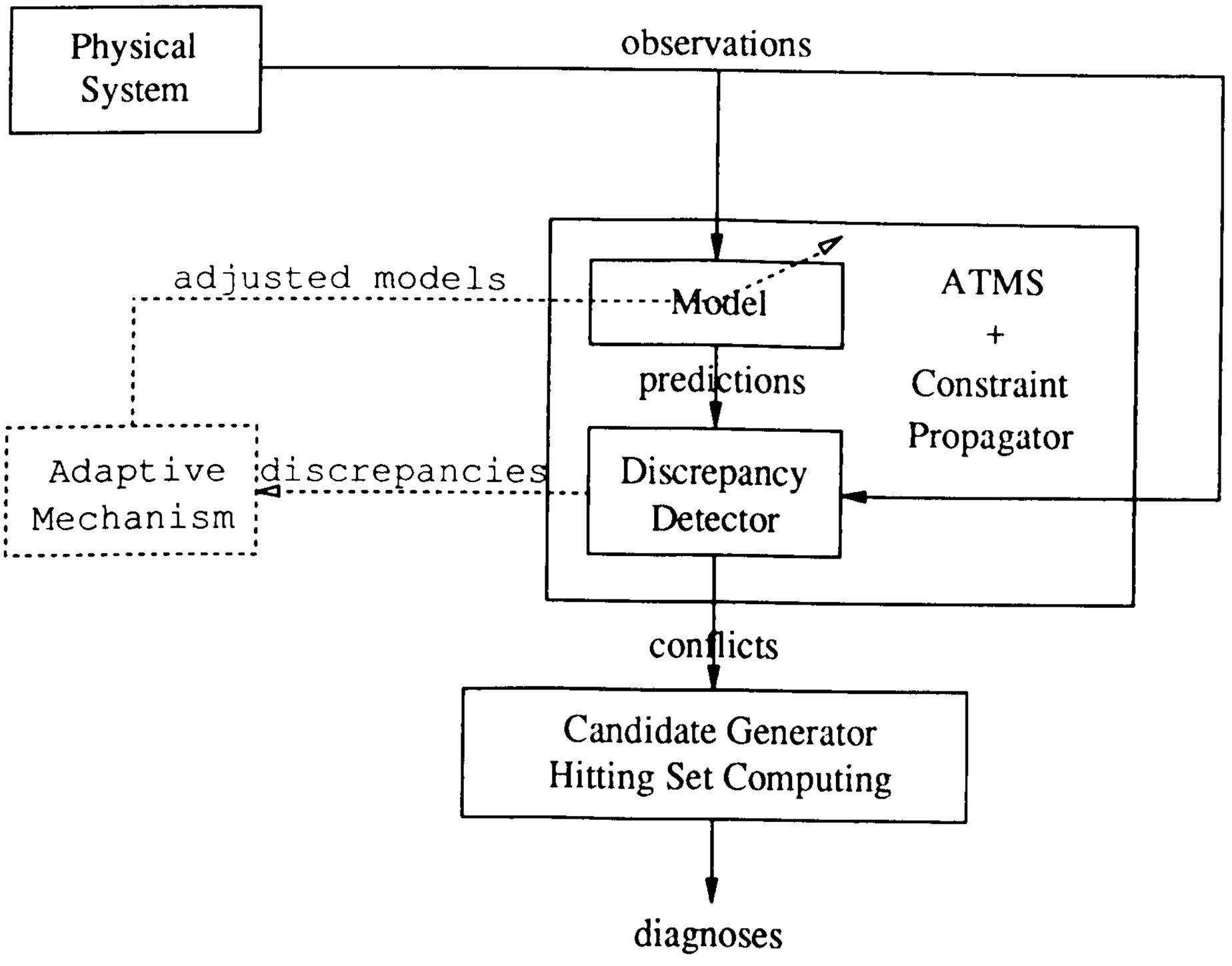


Fig. 6.1. *'Open Loop' Structure of Dependency-Recording Based Systems*

It is clear that, when the system to be diagnosed consists of only binary-valued components (e.g., digital circuits as commonly exampled in the literature of dependency-recording based diagnosis) it might be difficult or even not necessary to utilise such information on discrepancies. Nevertheless, without such feedback or the robustness property of self-adjusting of the system model, the dependency-recording techniques crucially require that exact system models be provided; imprecise models can result in wrong diagnoses. Of course, when accurate information on the system model is known or assumed the dependency recording techniques provide an extremely effective method for relating model assumptions to conflicts. It is because of these developments that the dependency-recording based approach represents a significant contribution to *MBDS* for static systems.

6.4.2. Normal and Fault Models Based Approaches

It is important to notice that the above explained diagnostic mechanism uses only a fixed model to predict the normal behaviour of a physical system. It determines a component (or a constraint) to be faulty if a retraction of its corresponding correctness assumption makes the predicted behaviour consistent with the observations. Knowledge about the behaviour models of known failures is not used. This may lead to implausible or even incorrect diagnoses because retracting an assumption does not necessarily imply that there exists a corresponding faulty behaviour in the physical system. Fortunately, the dependency-recording based approach to *MBDS* has been extended to exploit fault models (when available) so as to identify the actual mis-behaviour of the faulty components and, thus, limit the occurrence of implausible or wrong diagnoses. However, the introduction of fault models introduces computational complexity, thereby requiring various 'focusing' techniques to control the generation of candidates [103]. Clearly, the diagnostic strategies so used will therefore affect

both effectiveness and efficiency of such an extended method.

Recently, important research has been reported for developing diagnostic systems which use both normal-behaviour and fault models. Two representatives of such work are: *Sherlock* [27] and *GDE+* [109], extending *GDE* to derive additional diagnostic discriminations by using knowledge about faults. Both greatly enhance *GDE*'s capability of identifying faulty models as well as determining the faulty components. In fact, *Sherlock* views the central task of diagnosis as identifying the behavioural modes of all the components within a system, including whenever possible those for known failures. Meanwhile, it maintains the intuitions behind the normal-behaviour based approaches: to determine faulty components without necessarily knowing how they fail, by assigning each component with a unknown mode. However, each component now has multiple models and each model predicts a different behaviour which must be considered. There are, therefore, far more behaviours to reason about and the useful concepts of minimal conflicts and minimal candidates in *GDE* now becomes difficult to be handled (if not virtually meaningless) [27]. To avoid this problem, it resorts to probabilistic information about the likelihood of each mode of behaviour, introducing the most probable fault behaviours to components to restrict generating candidates and control the underlying predictor. Instead of using probabilistic information, *GDE+* utilises a resolution rule that depends on the completeness of fault models associated with a component, a strict assumption (although retractable) that a component fails in no other way than those specified in the fault models. Further detailed discussion about *GDE+* will be given in section 6.6.2.1, where it is employed as an instantiation of the general *MBDS* architecture to be established later.

6.5. Model Based Diagnosis of Dynamic Systems

Whilst important, the model-based diagnostic systems mentioned above have the crucial limitation that the system model is limited to algebraic models of the system considered to be in equilibrium. This is restrictive in that many fault conditions are only diagnosable during the transient between equilibria, since essential features of the system may depend on changing states and variables, as often the case with industrial process plants. Further, the transient behaviour can be used to provide an early indication of developing faults thereby allowing correct action to be taken before a disaster might happen. Essentially, for continuous dynamic systems, diagnosis must be performed based on their evolving behaviours during operating. Thus, the inference algorithm used for simulating system behaviour must be able to handle dynamic models, often represented by integro and/or differential equations. Introducing differential operation into system models automatically introduces time arguments into the representation of these models. This presents an important requirement to the system model such that it should have to be able to track the evolution of the behaviour of the physical system. Algebraic models cannot reflect this objective, though they can be used to simulate system behaviours in response to time-dependent inputs within a steady state [44]. As pointed out earlier, there exists a terminology confusion in the literature of *MBDS* such that the time-dependent responses of a static system are mistaken as the evolving responses of a dynamic system, thereby resulting in an unjustified claim that techniques developed for model-based diagnosis of static systems can be applied to diagnose dynamic systems. Also, the extremely useful dependency-recording technique in the diagnosis of static systems provides little diagnostic power in a system having feedback loops which is often the case in dynamic systems. The obvious problem is that tracing the dependencies recorded would usu-

ally return all the components within a feedback loop as suspect candidates because the output feeds back to the input of the loop. Thus, developing a technique for on-line *M*odel *B*ased *D*iagnosis of continuous *D*ynamic *S*ystems (abbreviated to *MBDDS* hereafter) will be an important contribution to the state-of-the-art in establishing model-based diagnostic systems.

Such work is only beginning. A pioneering work has been provided in [33], which is a technique (named *Mimic*) for monitoring dynamic systems based on fault models. *Mimic* utilises the Qualitative Simulation algorithm, *QSim* [52], as the behaviour simulator, with assistance of available quantitative information, to generate the dynamic behaviour of the system. By using a qualitative model to represent a system rather than a numerical one, an infinite number of infinitesimally close numerical behaviours is reduced to a small number of qualitatively distinct behaviours. Through such a representation, *Mimic* induces diagnostic knowledge off-line in the form of a decision tree by qualitatively simulating the known fault models. It accomplishes diagnosis in a hypothesise-and-match cycle on-line as illustrated by figure 6.2 [33], utilising the induced diagnostic knowledge but, essentially, based on model simulations.

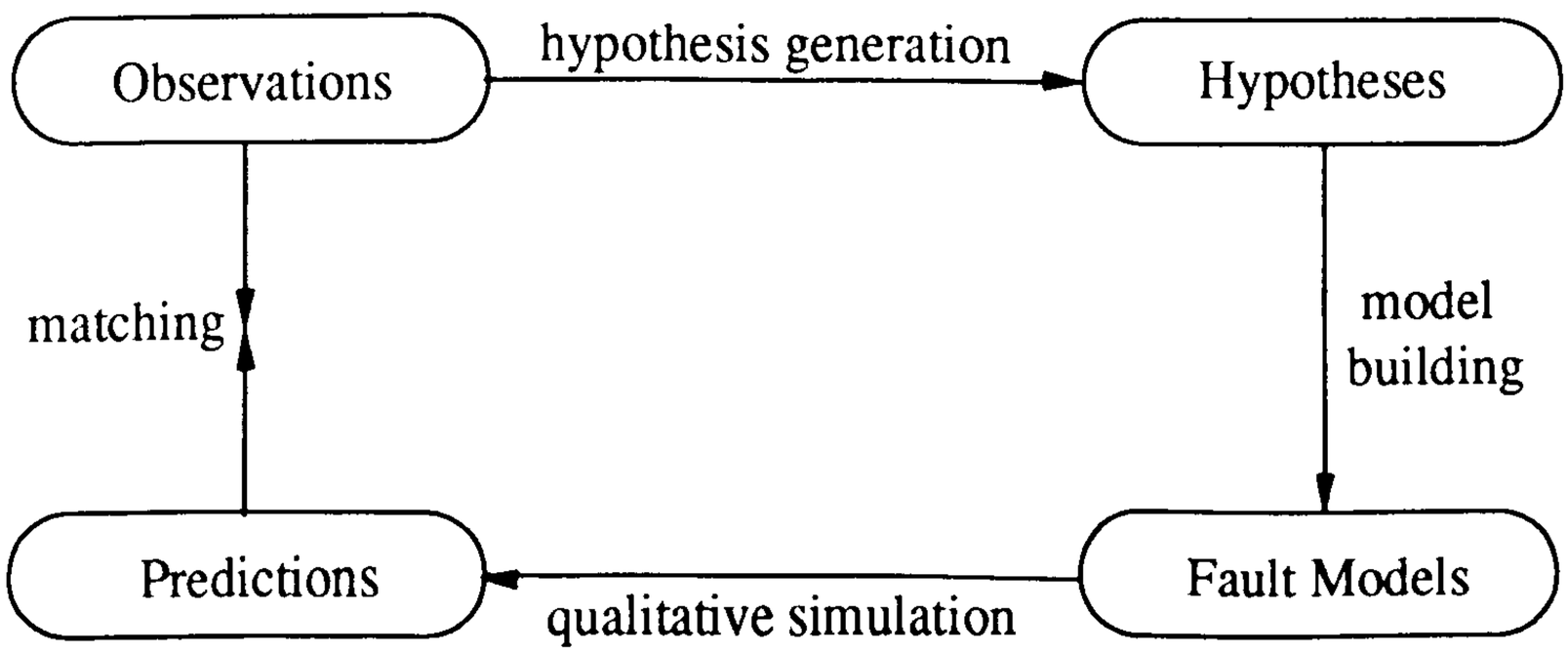


Fig. 6.2. *The Hypothesise-and-Match Cycle of Mimic*

To be concise, such a cycle can be explained as follows. Observations from the physical system evoke fault hypotheses through the decision tree, with each hypothesis corresponding to a particular fault model. *QSim* generates the predicted behaviour from the model of a possible fault, using as its initial condition the observation that evoked the associated hypothesis. If the predictions match with further observations *Mimic* returns the fault model being simulated as the one that reflects the physical system's current operating condition; otherwise, the model is exonerated from the suspect. In so doing, as claimed in [33], diagnosis can be refined as the physical system's dynamic behaviour is revealed over time. Furthermore, if requested, multiple-fault hypotheses may be created and tested incrementally.

From the way that *Mimic* performs diagnosis, it can be seen that *Mimic* combines rule-based and model-based approaches, the rule-based element providing fault hypotheses and the model-based element evaluating the hypotheses. It differs remarkably from static model-based approaches such as *GDE* and/or *Sherlock* in that these diagnostic systems are difficult to cope with dynamic systems from the first place no matter whether they make use of correct models only or together with fault models. More importantly, *Mimic* establishes, at least in principle, a feedback loop from the discrepancies detected between observed and expected system behaviours to the selection of possible faults which may have resulted in those discrepancies. Although in [33], observations were said to be used to initiate hypotheses instead of discrepancies, these observations are supposed to reflect the deviations of the model-based predictions from the real behaviour of the physical system. From this point of view, the underlying technique of *Mimic* is also similar to traditional approaches to system identification within the field of Model-Reference Adaptive Control [55], where numerical modelling and estimation methods are utilised for the determination of pos-

sible faults in the physical systems.

Being the first approach to on-line diagnosis of continuous dynamic systems using qualitative reasoning techniques, *Mimic* presents a significant contribution to developing *MBDDS*. However, a number of important difficulties in performing such diagnoses exist, as discussed in the following. First, only ordering of temporal information on the evolution of the physical system is provided from the model simulations by *QSim*, thereby making the 'tracking' of the real behaviour difficult. Other than using heuristic knowledge, *Mimic* cannot automatically guide and control the comparison between predictions and observations. As a particular implementation of the *MBDDS* systems, *Mimic* performs diagnosis dependent upon the changing behaviour of a physical system over time. The detection whether or not an observation is matched with a certain prediction can only be correctly done under the condition that both the observation and the prediction are made at the same time, or within the same system state. Unfortunately, it does not indicate a method to deal with this basic requirement. Actually, *QSim*, the behaviour simulator of *Mimic*, simply does not have such a representational ability to reflect such an objective. This causes problems about how *Mimic* realises tracking fault-model predictions against the observations, a central mechanism of dynamic diagnosis. Such problems have increasingly become noticed in the literature, actually, an on-going improving version of *QSim*, named *Q3* that attempts to integrate incomplete quantitative information into qualitative simulation [3], is being utilised to tackle these problems [34].

The second difficulty existing in *Mimic* is that, in addition to producing a qualitative description of the real behaviour, *QSim* may generate many spurious behaviours, again complicating the model tracking procedure, and, perhaps, making it intractable. Due to the difficulty in tracking and the ambiguity in simulation, *Mimic*

may produce many implausible and/or incorrect diagnoses. To reduce the possibility of wrong diagnoses, *Mimic* has to heavily resort to quantitative information apart from qualitative simulations, even for a simple example given in [33]. This will, sometimes, raise questions about when and how for a diagnostic system, built in a qualitative way, to seek such quantitative information. For instance, when an observation is qualitatively consistent with the normal-behaviour model of a physical system how can the diagnostic system know if this observation should be compared against a related quantitative range, as has to be done in that simple example presented in [33]. While, obviously, it is not intuitive to test if every qualitatively consistent observation matches with some underlying numerical range. Again, it is expected that the *Q3* may help to partially resolve this problem [34].

The third limitation is reflected by the fact that *Mimic* is based upon the assumption that all possible faults are explicitly modelled and have been simulated *a priori*. When this cannot be satisfied it only returns a message warning that the system currently being monitored is faulty, even if there exists a knowledge of possible faults in terms of potential model variations from the normal-behaviour model which have not yet been explicitly modelled and exhaustively simulated. In order to avoid this problem, under the assumption of single-fault, an proposal has recently been made to utilise the dependency-recording method to generate possible faults *only* from the structural model of normal behaviour (though, without showing how this may be implemented) [34]. However, an apparent restriction over this is that such a technique would encounter, as all the dependency-recording based methods are difficult to avoid, a severe difficulty in coping with faults occurring within feedback loops that most dynamic systems have and, therefore, significantly reduce the diagnostic power that original *Mimic* can have for such systems.

Although these limitations exist, *Mimic* offers an exciting prospect for *MBDDS*. Clearly, a diagnostic system which can overcome or reduce these difficulties whilst retaining the advantages of diagnosing dynamic systems based on a well-developed qualitative reasoning technique would be beneficial. The next chapter will contribute to this goal.

6.6. Architecture of Model Based Diagnostic Systems

From the brief review of existing approaches to *MBDS* given above, it can be seen that, although developed for coping with different physical systems (static or dynamic) and based upon different system models (normal-behaviour or fault or both), these approaches share a common fundamental concept. That is, diagnosis is accomplished by the use of an explicit model of a system's structure. Such a model-based diagnostic mechanism determines some components (or constraints over the components) of a physical system which account for the observed abnormalities, i.e., the discrepancies between the observed and designed (normal) behaviour of the system or other inconsistencies between the predictions themselves obtained from the model and driven by the observations.

However similar to each other these approaches appear, it is difficult to compare them functionally without a common structural basis upon which the diagnostic systems are built. This would prevent a particular technique established within a particular *MBDS* system from being used for a different diagnostic task. Thus, it is important to construct a generic architecture which can reflect the basic idea of *MBDS* and consists of functionally crucial sub-systems that can be specified with different interpretations for different purposes. Further, such an architecture should be flexible enough to allow possible future improvements to be investigated within the area of model-based diagnosis and also to allow, where appropriate, useful combinations of different

approaches. This section is dedicated to reaching such a goal.

6.6.1. The Architecture

We construct the architecture of model-based diagnostic systems with respect to the general functionalities that such a system should ideally have. However, to cope with a particular class of diagnostic problem the corresponding realisation of this architecture may or may not need a complete implementation of all such functionalities. At the most general level, an *MBDS* system can be considered as being composed of an interconnection of four fundamental modules: A *Predictor* module that utilises an explicit model of the system to be diagnosed and a mechanism for predicting the behaviour from the model; a *Discrepancy Generator* that compares the observations from the system with the predictions, or amongst the predictions themselves driven by the observations, and generates appropriate discrepancies; a *Candidate Proposer* that produces fault candidates from knowledge of the discrepancies and that of possible faults (though, fault knowledge may not be necessary for some particular situations); and a *Diagnostic Strategist* that contains the meta-knowledge necessary to control and co-ordinate the complete diagnostic process. It will become clear that these four basic modules functionally play different roles in the entire diagnostic process based on different kinds of knowledge provided for them, and that the essential parts of an *MBDS* system are the first three sub-systems, namely, the Predictor, the Discrepancy Generator, and the Candidate Proposer. Such a primitive architecture of *MBDS* can be depicted as shown in figure 6.3, with each functional module being defined in the following. It is worth mentioning, again, that some of the basic terminology employed within the *ARTIST* project is herein utilised to describe the development of this architecture.

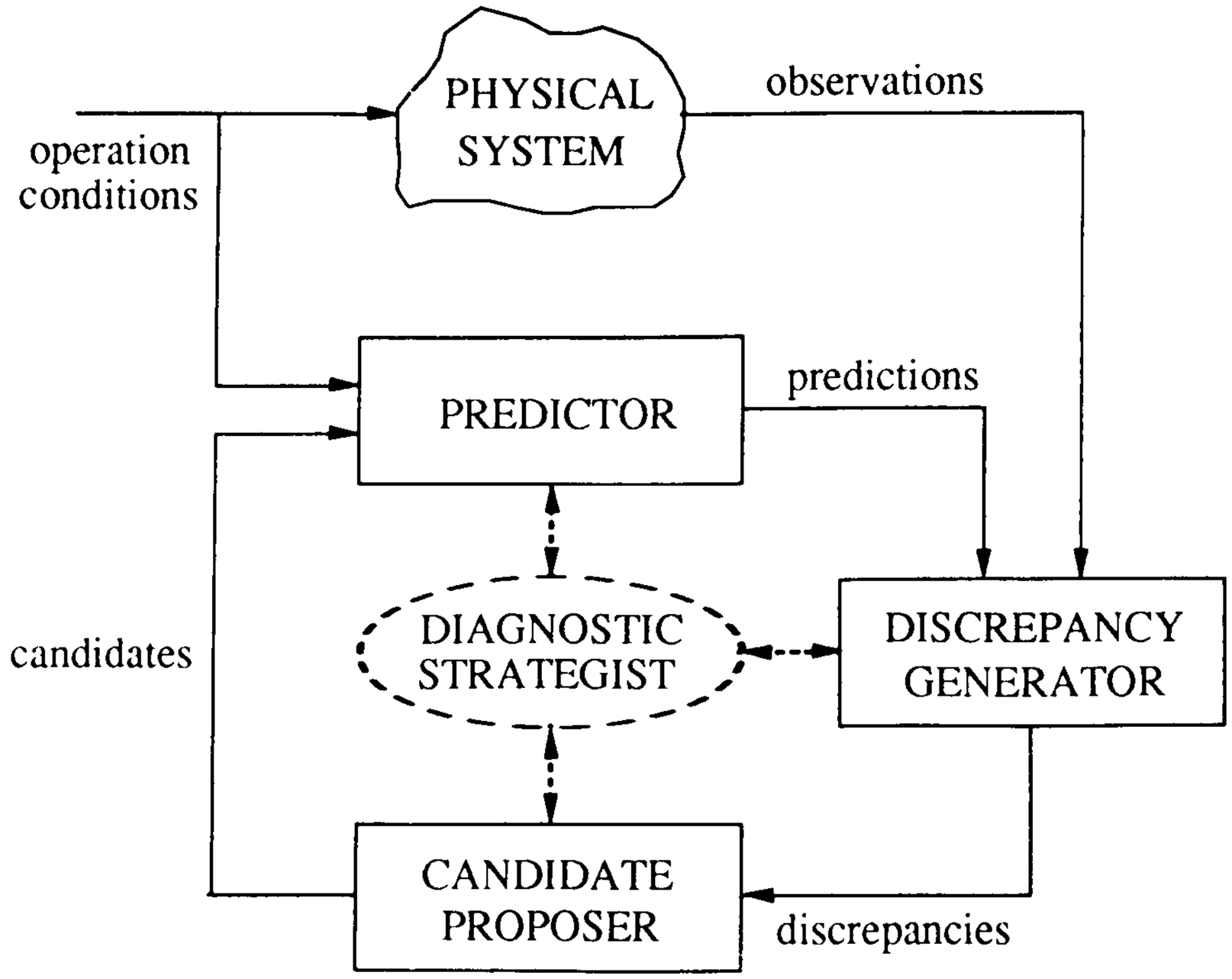


Fig. 6.3. *A Primitive Architecture for MBDS*

6.6.1.1. Predictor

The task of this module is to predict the behaviour of a physical system from observations based on an explicit model of the physical system. This module itself consists of two sub-modules: a *Data Interpreter* that translates the observations of the actual behaviour and other operating conditions of the physical system, obtained from a user or another computer system, into appropriate representation with respect to a particular diagnostic task and the system model used; and a *Behaviour Simulator*, the key sub-module of the Predictor, that derives a more complete description of behaviour from the (interpreted) observations and other known operating conditions based on the given model.

It is clear that the central issue herein is the system model which reflects knowledge about the physical system to be diagnosed. To perform model-based

diagnosis, in principle, both theoretical and empirical knowledge may be utilised in each module of the basic architecture, depending on how the term 'model' is interpreted. The theoretical knowledge, here, means that obtained from the formal application of scientific theories in terms of physical laws, while empirical indicates that the knowledge based on experimental evidence gained from experience. However, if a system model is entirely built upon empirical associations the resulting diagnostic system will degenerate to that belonging to the first generation knowledge-based diagnostic systems, thereby suffering from the same setbacks of rule-based systems as pointed out in section 6.2. This is why we emphasise that the system models within *MBDS* are represented by explicit structural relations amongst system variables, usually in the form of physical constraints driven from theoretical knowledge.

Another important issue is the predictive mechanism or the simulation algorithm used to generate the behaviour from the system model. Selecting which kind of simulation algorithm to generate predictions is very dependent upon the physical systems that are to be diagnosed. For instance, to perform static model-based diagnosis, the algorithm utilised *should be able to draw inferences* from relations amongst system variables without distinguishing the directionality of the relations when computing the values of the variables. For dynamic model-based diagnosis, a crucial requirement is that the simulation algorithm must enable the generation of an evolution of the system's behaviour over time.

6.6.1.2. Discrepancy Generator

The task of this module is straightforward, to detect inconsistencies between observed and predicted behaviours or between different predictions generated from observations and, then, return the discrepancies (if any) in an appropriate form suitable for the use of candidate generation.

It is worth pointing out that, in all research on *MBDS*, observations are treated as facts no matter whether they have been interpreted in terms of a certain representational form. However, as usually the case, observations may suffer from noisy measurements. Thus, a Discrepancy Generator used within a particular *MBDS* system should have certain robustness such that it can view two different measured values as the same when necessary. Otherwise, pseudo-discrepancies and thereby spurious candidates could result, though such effects might be reduced through data interpretation.

Of course, the selection of a robust discrepancy detection mechanism is very reliant upon knowledge of potential discrepancies and upon the satisfaction criteria that are utilised. Further, when dealing with continuous dynamic systems, a particular and important requirement of such a discrepancy generator is that it should be able to detect inconsistencies between observations and predictions over time (i.e., whenever the system being diagnosed changes its state the generator must 'track' the observations) and also provide a 'measure' of the degree of behavioural degradation.

6.6.1.3. Candidate Proposer

The task of this module is given discrepancies to propose the fault candidates and, based on this, to perform further steps to discriminate these candidates when necessary. This module can be decomposed into three sub-modules: a *Candidate Generator*, the key sub-module, that generates possible faults which may account for the current discrepancies; a *Candidate Evaluator* that discriminates and/or elaborates the candidates through certain refining evaluation process(es) such as extracting more information about the actual behaviour of the physical system and/or performing further reasoning with the model [108]; and a *Candidate Ranker* that ranks the candidates according to the possibility, probability, and/or common-sense likelihood (e.g. how unlikely that a large number of components may fail at one time in a non-

explosive situation) of which they could be the real culprits.

When emphasising the importance of theoretical knowledge within *MBDS*, as pointed out earlier, empirical knowledge should not be ignored. In particular, when maintaining the basic premise of model-based approach that utilises theoretical knowledge for system modelling, within the candidate generation task it is useful to exploit the efficiency of empirical knowledge, normally in the form of local associations which directly relate symptom with failures (e.g. a look-up table). Also, the Candidate Ranker can play a more active role in the complete candidate generation and evaluation processes when empirical knowledge is used if such knowledge is indeed available. This is because, for many physical systems, some particular faults are often known to be more likely to happen than others. This does not necessarily imply 'pure' empirical knowledge but usually the integration of both theoretical and empirical knowledge sources. A generic architecture constructed in this way, therefore, can support a model-based diagnostic system to merge the merits of model-based and rule-based approaches.

6.6.1.4. Diagnostic Strategist

The task of this module is to control the entire diagnostic process and coordinate the information exchange between different modules. It can be further viewed as the combination of two sub-modules: a *Performance Evaluator* that adjudges the performance of the system model used, the discrepancies generated, and the fault candidates proposed, subject to certain satisfication criteria and/or basic diagnostic goals; and a *Supervisor* that guides the information exchange and executes the control actions based on the evaluated diagnostic performance and other knowledge resources such as domain-dependent heuristics.

Although the crucial reasoning process in diagnosis only involves behaviour prediction, discrepancy detection, and candidate generation, the success of the diagnosis, i.e., the possible failures returned by this process are the real ones that the physical system is suffering from, is based on the varieties of knowledge available to the diagnostic system and the diagnostic strategies employed for deciding how to utilise such knowledge. The importance of the diagnostic strategies within the entire diagnostic process is reflected in the following major tasks of the Supervisor. Of course, some of these strategies may not be necessary for certain *MBDS* systems.

1) To control the Predictor by focusing attention, often termed the *focus of attention* [103], indicating which part of the system's structure, which models, what value range and cardinality of the quantity space in the representation of system variables should be currently considered, and when to begin and stop simulating the behaviours. The basic intuition behind this is that a physical system may be described by a number of different models, including fault models. However, using all these models for the behaviour-predicting at one time is computationally too expensive and often unnecessary. Furthermore, for those systems which exhibit a large degree of structural complexity their models may have to be simplified and/or decomposed before being utilised [106]. For dynamic and/or time varying systems, without tight temporal restriction on the simulation the prediction process may never terminate.

2) To show the Discrepancy Generator by specifying what satisfaction criteria in general and which kind of discrepance metric and what threshold in particular are currently being used, and, if necessary, when to start and terminate the comparisons between predicted and observed behaviours or amongst different predictions.

3) To control the Candidate Proposer by focusing suspicion, usually called the *focus of suspicion* [103], that narrows down the scope of interesting candidates or makes diagnostic assumptions like single-fault assumption or no-structure-change assumption (stating that the structure of the system will not change and faults only occur inside individual components); and to guide it by pointing out which next action to be taken to refine the suspected candidates, for example, to use fault hypotheses or more detailed representation of the system's structure. Without such control and guidance the number of possible candidates to be considered or returned might be far too large or even intractable.

As stated earlier, the Supervisor applies the control strategies to a diagnostic process based on evaluating the diagnostic performance currently achieved and other knowledge resources. The way to tell whether or not a particular diagnostic task has been 'optimally' accomplished is to determine the *performance indices* related to either the candidates generated thus far or the discrepancies generated thus far or both. Sometimes, the time restriction to completing a diagnosis also needs to be considered, in particular for the real-time applications of a diagnostic system. A performance index is a measure, obtained from using a particular *metric*, which indicates how well those particular candidates match the underlying real faults or how well and/or how fast those discrepancies are vanishing, with respect to specified satisfaction criteria. The selection of such metrics is dependent upon the concrete representation of the potential candidates and discrepancies that the diagnostic system can generate, and therefore, dependent upon the representational form of the model of the physical system and that of the values of system variables.

6.6.2. Instantiations of the Architecture

To demonstrate the generality of the architecture developed for *MBDS* in the last sub-section, this sub-section revisits two previously mentioned typical *MBDS* systems: *GDE+* (for diagnosing static systems) and *Mimic* (for diagnosing dynamic ones) via reconstructing them with respect to the architecture. It should be clear that the following reconstruction is mainly based on the techniques reported in [109] and [33], though a few modifications and extensions are added by considering their potential abilities to utilise techniques developed in other *MBDS* systems and by taking into account some updated improvements made to the two diagnostic systems. In particular, it is worth pointing out that, an interesting research intended to extend *GDE+* to a more general diagnostic mechanism is currently being undertaken [104, 105]. Some of its basic ideas, but by no means the complete, has been incorporated in the instantiation of *GDE+* as follows.

6.6.2.1. GDE+

Predictor

Data Interpretor: This sub-module maps the given real observations onto an appropriate representation, though it might be empty since, within the behaviour simulator used in [109], the observations are treated as facts in the original form of which they are provided.

Behaviour Simulator: A constraint propagator [110] which, given the model of a physical system represented by a set of algebraic constraints, performs local derivation (or propagation) of some variables' values from other given variables' values. Importantly, the dependency of each such propagation step is recorded by the *ATMS* [21, 22].

Discrepancy Generator

It identifies discrepancies simply based on whether a variable's observed value and predicted value or its predicted values obtained from using different constraints are identical. The important point here is that, if two values are detected as inconsistent, the generator installs (minimal) inconsistent assumption sets, upon which the predictions have been made, through the dependency network maintained by the *ATMS*. These inconsistent assumption sets are so-called the conflicts [26].

Candidate Proposer

Candidate Generator: A formal algorithm that computes all the minimal hitting sets [82] for the collection of conflict sets returned by the Discrepancy Generator. A minimal hitting set is a possible candidate.

Candidate Evaluator: It discriminates and/or elaborates possible candidates by using further measurements and/or behaviour refinement. The possible future measurements are proposed by a so-called 'one-step look-ahead' technique which enables discriminating the candidates with a minimised number of the measurements [39]. Behaviour refinement is done by exploiting knowledge about faults, using a resolution rule which assumes that if predictions made from all known fault models of a component (or a sub-system) are inconsistent with the current observation then the component (or the sub-system) does not fail.

Candidate Ranker: It ranks the possible candidates by assigning each candidate with the reverse of its cardinality (i.e. the number of the possible faults which would happen together) as weight; the larger a weight, the higher the rank and the more likely the associated possible fault to happen.

Diagnostic Strategist

Performance Evaluator: This sub-module can be treated as empty except for conveying the ranks of possible faults to the Supervisor. This is because, when the system model is given, it is not necessary to evaluate either candidates or conflicts if the Discrepancy Generator returns conflicts without distinguishing the differences amongst the underlying discrepancies and the candidates are directly related to conflicts in a fixed way.

Supervisor: It guides the Candidate Proposer to refine the highest ranking candidate(s) first by exploiting the related fault model(s), and commands the Proposer to discriminate between candidates by using additional measurements if these candidates are equally likely. Also, it makes the decision when to retract the basic assumption that only typical or known faults may occur and to consider unknown faults (e.g. when each possible candidate proposed consists of too many faulty components).

6.6.2.2. Mimic

Predictor

Data Interpretor: This sub-module maps the given real observations onto the corresponding qualitative values (landmarks or intervals between the landmarks) [52].

Behaviour Simulator: The *QSim* algorithm that models a physical system as a set of first-order qualitative differential equations and allows the system's dynamic behaviour to be generated given its initial state.

Discrepancy Generator

It detects discrepancies by using a so-called similarity function which computes the similarities between observations and predicted states. This is done in such a way that if the similarity computed is below a (given) threshold then a discrepancy is found, although the explicit forms of both the similarity function and the threshold are not presented in [33].

Candidate Proposer

Candidate Generator: A decision tree which directly relates the inconsistency resulting from the current observation with possible candidates (or fault hypotheses). The tree is provided off-line by an inductive learning technique [79] that captures diagnostic knowledge from qualitatively simulating all the possible fault models known.

Candidate Evaluator: This sub-module is empty due to the basic characteristics of the hypothesise-and-match cycle built in the diagnostic system. The candidates produced by the decision tree are automatically tested and debugged through multiple further measuring, predicting, and comparing processes that conceptually may be run in parallel. That is, for each of such processes, a partial behaviour of a particular fault model (one of the possible candidates) is predicted and compared against new observations. If matched, the candidate is confirmed to be a final diagnosis; otherwise discard this model from the set of possible candidates.

Candidate Ranker: It orders the possible candidates by likelihood, though which kind of likelihood is used within a particular diagnostic task is left open.

Performance Evaluator: It may evaluate the candidates proposed by recording the remaining discrepancies (associated with each of the suspect faults) when these candidates are returned: the smaller the recorded discrepancy, the higher the performance of a particular diagnosis. Also, it can evaluate the performance of the discrepancies themselves such that the smaller the similarity between the predicted and observed behaviours, the more the model used is distorted from the designed values. Of course, all these evaluations should be subject to temporal restrictions, i.e., they are valided only at the time or during the period when those comparisons are made.

Supervisor: It guides the Candidate Proposer to check the highest ranking candidate(s) first by testing the related fault model(s) against new observations over time, and controls the Discrepancy Generator to compare the qualitatively matched observations with the underlying quantitative range when necessary. However, the most important task of the Supervisor herein is to show how far the predicted behaviour should be made and which part of it should be compared with the current observation. Unfortunately, as pointed out earlier, how this can be done is not clear in [33].

6.7. Summary

Through showing the advantages of model-based approaches to diagnosis, the basic motivations for constructing *MBDS* systems are briefly presented. Based on a critical review of the developed *MBDS* techniques for diagnosing static or dynamic systems, two distinct approaches are identified, i.e., the *dependency recording* based approach and the *MBDDS* approach that utilises dynamic models of physical syste and explicitly establishes the information 'feedback' from the discrepancies detect

to the adjustment (or substitution in *Mimic*) of the system model that underlies these discrepancies. Further, a generic architecture for *MBDS* is proposed that consists of four interacting modules, namely, a Predictor, a Discrepancy Generator, a Candidate Proposer, and a Diagnostic Strategist. A clear distinction of the functionalities of these modules is made and two typical instantiations of the architecture are given, demonstrating the desirable generality that this architecture has for building different *MBDS* systems to perform different diagnostic tasks. In fact, this generic architecture has been used to establish the basic framework for *MBDS* systems within the *ARTIST* project [40]. In the next chapter, a model-based diagnostic system which conforms to this architecture will be constructed that utilises *FuSim* as the Behaviour Simulator within the Predictor for diagnosing continuous dynamic physical systems.

CHAPTER 7

SYNCHRONOUS ITERATIVE DIAGNOSIS OF CONTINUOUS DYNAMIC SYSTEMS

7.1. Introduction

Having developed a generic architecture for model-based diagnosis, this chapter investigates how the *FuSim* algorithm can be utilised within *MBDS* systems. This results in several innovations in the development of model-based diagnosis for continuous dynamic systems. In particular, a technique for the *synchronous tracking* of the evolution of a physical system between equilibria is developed. Further, and importantly, a method for *iteratively searching* through the space of possible model variations is proposed. Also, a *discrepancy measurement* mechanism is given that allows for the continuous degradation from normal to faulty behaviour to be monitored. While establishing a particular realisation of the architecture using *FuSim*, this chapter also opens a number of opportunities for future work.

The chapter is arranged as follows. First, the general descriptions of the *synchronous tracking*, the *iterative search*, and the *discrepancy measurement* are presented in sections 7.2, 7.3, and 7.4, respectively. Then, a particular implementation of such an *MBDDS* system based on *FuSim*, named *SID* for *S*ynchronised *I*terative *D*iagnosis, is constructed with respect to the generic architecture given in the last chapter. Next, the operation of *SID* is demonstrated by an application to a simple system under various fault conditions. Finally, the chapter is summarised. Substantial work presented in this chapter has been reported in [64, 65, 94, 95] and the *SID* diagnostic system is currently being further developed within the *ARTIST* project [40,

41, 60].

One important thing to mention is that, although this chapter utilises *FuSim* to develop an *MBDDS* system, the above-named three general techniques for *MBDDS* are identified and extracted. These techniques, we believe, are important contributions to constructing a new type of *MBDS* systems using *AI* modelling and reasoning techniques. Actually, it can be seen from the following sections that, there is no reason why other simulation tools that can generate the evolving behaviour of physical systems, with associated temporal durations, cannot be used for the same purpose, if there is no need to cope with uncertainty. This is why, herein, we describe the basic methodology established first.

7.2. Synchronous Tracking

The reliable diagnosis of faults on continuous dynamic systems requires that the model of a physical system synchronously tracks the evolution of the observations from the system. To achieve this, the behaviour simulator used within an *MBDDS* system must produce the durations (at least, certain estimates of the durations) associated with the respective states and also minimise the spurious behaviours generated. Furthermore, a technique which is able to compare predictions against observations over time must be provided. In fact, for coherent fault detection (or discrepancy detection) the comparison between the observed behaviour and the predicted behaviour must be made at the same system state, i.e., at the same absolute time (point or duration). This is because the system model used must operate *synchronously* with the natural evolution of the physical system. This brings a problem to the use of traditional qualitative simulation algorithms as the behaviour simulators, e.g. *QSim* in *Mimic*, because of the need to guide and control the comparison between the observations and the predictions. In this sense, temporal information becomes

essential to the maintenance of synchronous behaviour. Without this it is impossible, without resorting to heuristics, to control the evolution of the models, including spurious behaviours and also fault models. It is in this respect that *FuSim* provides important advantages. This will become clear in the discussion presented in section 7.5.2.

Suppose that a qualitative simulator is provided that satisfies the requirement of generating qualitative states with associated temporal durations. The rule used to guide and control the synchronous tracking of the system behaviour (for detecting discrepancy) can then be directly deduced from knowledge of the temporal durations of the qualitative states and is stated as follows:

> Treat the current observation, $OBS(t_0)$, as the initial state of the model and the time when the observation is made as the initial temporal point. From the next observation $OBS(t_1)$, generate the simulated behaviour from $OBS(t_0)$ until the temporal upper bound of a qualitative state meets or covers t_1. Then, compare this last generated state with $OBS(t_1)$.

Notice that, when more than one possible behaviour results from the qualitative simulator employed (due to the theoretically unavoidable ambiguities of qualitative simulation [101, 107]), the model being used remains as a validated system model provided that one of the last states generated thus far within these behaviours matches with the observation. Also, further simulation, when necessary, will continue from the behaviour which contains the matched state.

Illustratively, the way to track a model, or to compare predictions against observations, can be depicted by figure 7.1 and explained in the following. Without losing generality, suppose that the first two observations from the physical system are $OBS(t_0)$ and $OBS(t_1)$, the simulator then uses $OBS(t_0)$ as the initial state, $PRED(T_0)$, and starts running the simulation of the model. If it generates the possible next state,

say, $PRED(T_1)$, which matches $OBS(t_1)$ under the condition that $t_1 \subseteq T_1$, no discrepancy is detected and hence the discrepancy generator waits for another observation, $OBS(t_2)$, to be available. After this, the qualitative simulator continues predicting the successor state $PRED(T_2)$, however, this state's temporal information indicates that for any $t \in T_2$, $t < t_2$. Thus, it keeps making further predictions from $PRED(T_2)$ and results in both $PRED(T_2')$ and $PRED(T_2'')$ as possible next states. Now that $PRED(T_2'')$ does not match with $OBS(t_2)$, though $t_2 \subseteq T_2''$, no further predictions will be made following the branch beginning at the $PRED(T_2'')$ whatever later observations are obtained. However, since $PRED(T_2')$ matches $OBS(t_2)$ and $t_2 \subseteq T_2'$, the model being used remains as validated and the process of prediction and comparison recurs, starting from the matched prediction $PRED(T_2')$. Otherwise, if $PRED(T_2')$ also conflicts with $OBS(t_2)$, this model is discarded. Discarding a model, herein, simply implies that this model is inconsistent with the current operating condition of the physical system.

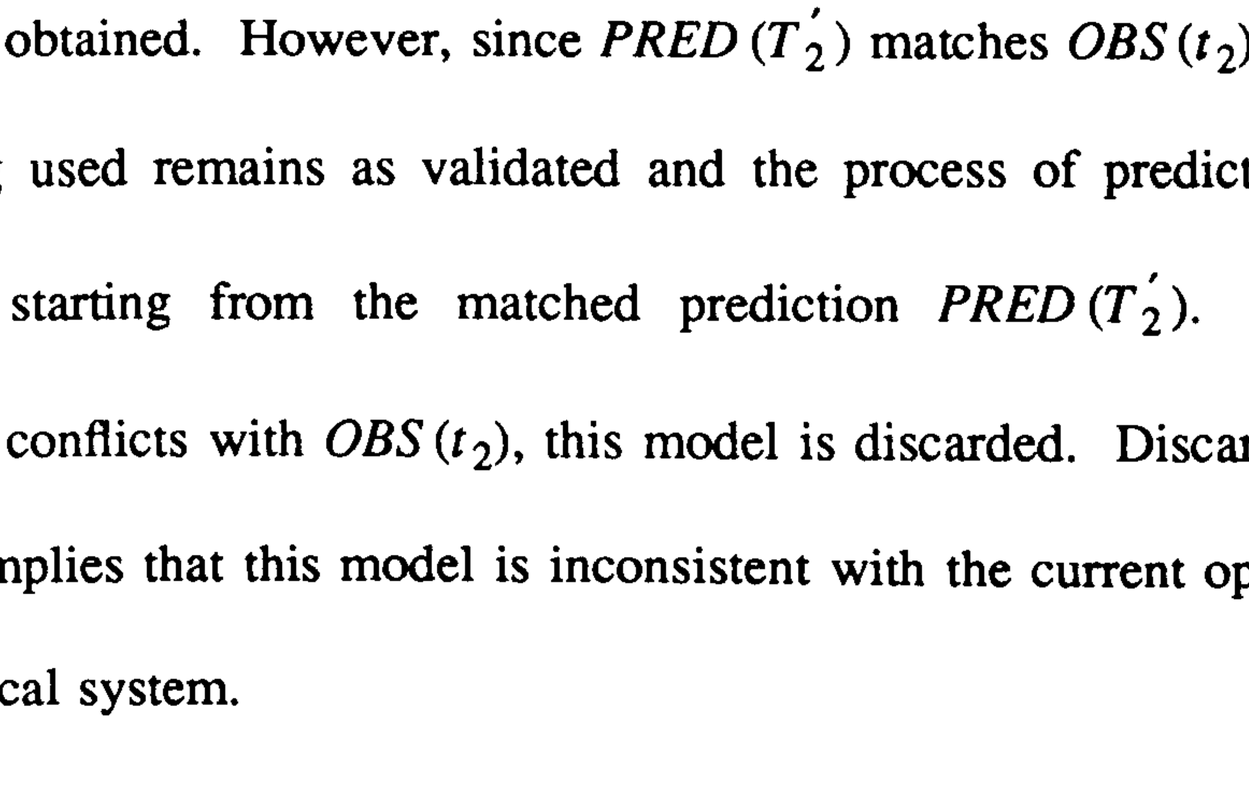


Fig. 7.1. *Tracking a Model*

The present synchronous tracking mechanism tracks a model without distinguishing if it is the normal-behaviour model, a modified model of the normal-behaviour one against a particular requirement (see discussions in the next section), or a known fault model. Therefore, if the model currently being tracked reflects the normal behaviour of the system, an on-line discrepancy generator built in this way actually performs system-monitoring with an identical structure. Once a discrepancy is detected the diagnostic process is activated and the model of normal behaviour is adjusted along potential model variation directions. The monitoring task then becomes a fault identification task. How to hypothesise the possible faults is, of course, the task of the Candidate Proposer for which the following iterative search technique is proposed.

7.3. Iterative Search

From the previous discussion about the dependency-recording based approaches to *MBDS*, it is clear that those techniques do not make explicit use of feedback from the discrepancies detected to the choice of faulty components in a normal-behaviour model or to the selection of fault models. Rather, they assume exact relationships between fault candidates and generated conflicts and use *assumption-based truth maintenance systems* (*ATMS*) [21, 22] to efficiently record such dependencies. For the purpose of identifying faults (instead of merely isolating faults), when only approximate fault knowledge is known (as in the case of a continuous fault space), the sensitivity of these approaches to errors can be high. An alternative approach, identified in the last chapter and specified herein, is to make the feedback from discrepancies to model variations explicit and to use this feedback to iteratively search for suitable fault models that minimise the discrepancies. This feedback structure reduces the sensitivity of the diagnostic process to modelling errors or to inaccuracies

in the fault models.

Such an approach to *MBDS* is based on an *iterative search* technique to be described in the following sub-section. At the most schematic level, diagnosis is performed as a refining process as follows. Initialised by observations, a behaviour simulator predicts the behaviour based on the system's model under normal conditions, and a discrepancy generator detects and generates the discrepancies between the predictions and the observations. The resulting discrepancies direct an iterative search of the space of possible (likely) model variations, using the most likely variation direction first, until a 'matching' fault model is obtained. To acknowledge the basic technique used the *MBDS* systems developed in this way will be termed the *iterative search* based diagnostic systems. Within such systems, the 'matching' decision is made such that a 'metric' of a discrepancy can be minimised (subject to certain satisfaction criteria) in a particular modelling dimension (or a particular direction of model variations). This, of course, requires that the possible model dimensions be clearly identified and characterised. Section 7.3.2 is, therefore, dedicated to defining four basic modelling dimensions, namely, *abstraction, commitment, resolution*, and *relation strength.*

7.3.1. Basic Method and Properties of Iterative Search

It is important to remember that the purpose of developing an iterative search method is to propose candidates and then to evaluate these candidates, based on the explicit utilisation of information on the discrepancies generated during the diagnostic process. As emphasised earlier, in general, such a process is automatically initiated by switching from a monitoring process (over the physical system to be diagnosed) once certain discrepancies are detected. This implies that the system model of intended normal behaviour is always available, a basic assumption made by all *MBDS*

approaches. Further, it is assumed that a model space of possible variations of the normal-behaviour model has been characterised by providing underlying modelling dimensions (see the next sub-section), though explicit models other than the normal-behaviour one within the model space are not required. Each dimension, combined with others, determines a sub-space of model variations. Based on this, the process of iterative search can be concisely described as follows.

Start with system-monitoring based on the normal-behaviour model of the physical system, when certain discrepancies are generated (determined by the adopted distance metric and often represented by their corresponding types and amount, see section 7.4) these discrepancies initiate the modifications and/or adjustments of the model within different sub-spaces of model variations characterised by the underlying modelling dimensions. To what degree such modifications or adjustments should be made is dependent upon the detailed properties of the discrepancies (e.g., which type the discrepancies belong to and how large they are). Each such modified (or adjusted) model will be treated as a candidate to be further evaluated as previously illustrated in the model-tracking process. The selection of which model dimensions to search and the assertion of possible fault models are under the control of the diagnostic strategist. If the current model under tracking is discarded, another model is assumed by further modifying or adjusting the current one unless the entire model space or those model sub-spaces required to search by the diagnostic strategist have been exhaustively searched. Certainly, the convergence, i.e, whether or not the iterative process of the search will terminate, is very dependent upon choosing the correct dimensions to search and also a suitable discrepancy metric used for testing the matching.

From this viewpoint, a candidate generator that implements such an iterative search method maps discrepancies onto particular fault hypotheses. However, impor-

tantly, the fault hypotheses herein do not necessarily imply those pre-defined off-line as in *Mimic* but usually obtained from the on-line search process through modifying (or adjusting) the normal-behaviour model of the physical system. In order to perform the mapping between the discrepancies and the candidates the candidate generator stores the modelling dimensions to be used to build system models and a set of rules, called the *search rules*, to guide the search against these dimensions. Of course, the concrete form of such rules may vary with respect to different distance metrics used and also the satisfaction criteria provided. This will be further discussed in section 5.7.3.1, where specific search rules are presented in response to the particular discrepancy detection mechanism adopted. In general, the skeleton of the candidate generation process by iterative search can be illustrated by figure 7.2.

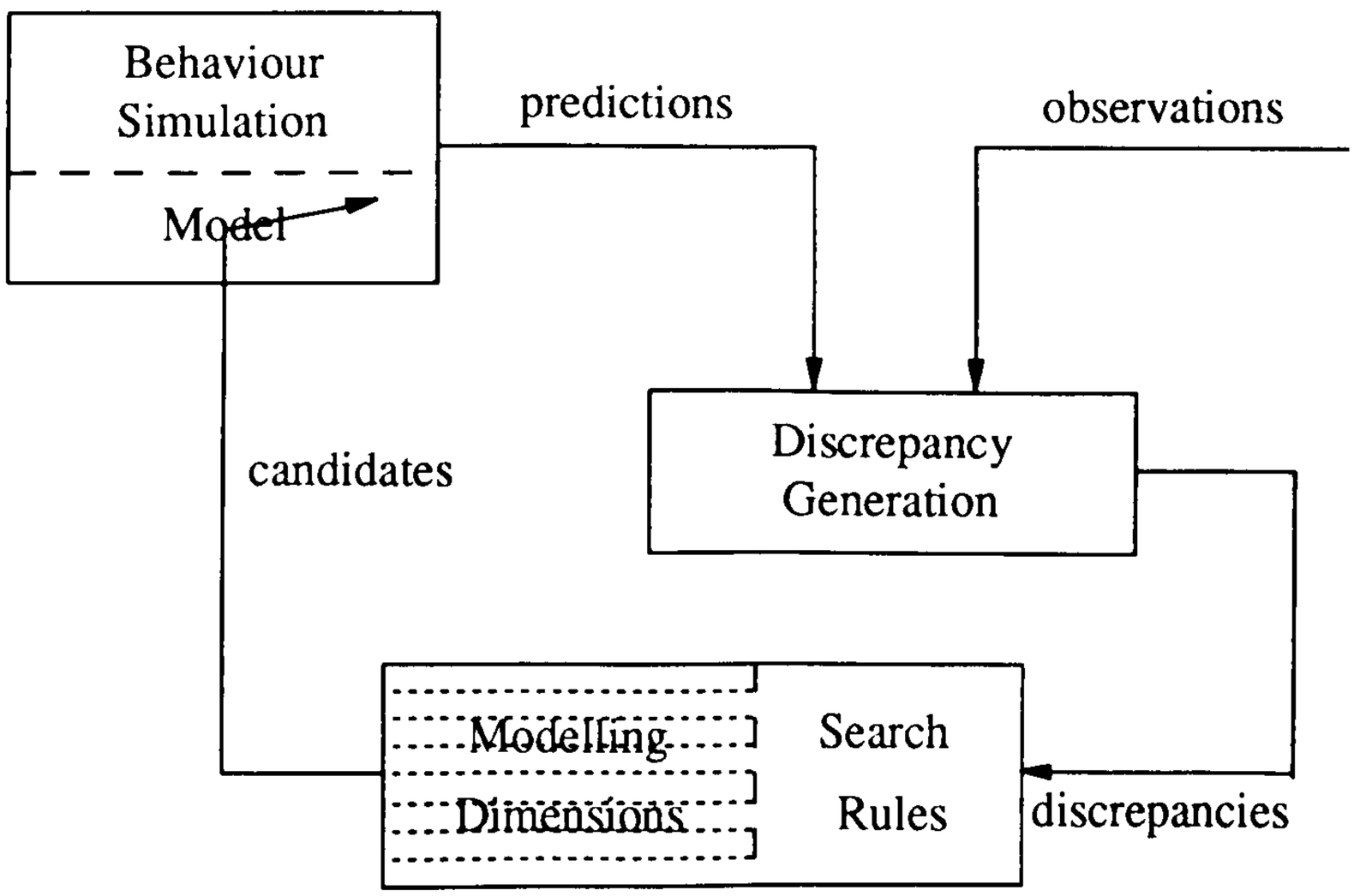


Fig. 7.2. *Candidate Generation by Iterative Search*

It is important to point out that fault models can also be used within the iterative search based approach, though not necessary to be presented to enable a diagnostic process. As a matter of fact, if fault models are known *a priori*, either from off-line

compilation or directly from empirical evidence (again, see discussion in section 7.5.3.1), then these may be effectively utilised to initialise the search at the corresponding points in the model space. This focuses the search, often termed *focus of suspicion*, at the most likely points and iteration starts from them, greatly reducing the search space and hence enhancing the efficiency of the search process. An interesting, and extreme, case is that the model space consists of only normal-behaviour model and known fault models, i.e., all faults are known *a priori*. Under this condition the diagnostic task performs a hypothesise-and-test cycle: the candidates are proposed by an off-line compiled look-up table which links discrepancies with fault hypotheses and are further evaluated through synchronously tracking against the evolution of the physical system. As a result, *Mimic* [33], based on the search in a known fault space, can be treated as a special case of the iterative search technique, though [33] does not explicitly indicate the underlying property of the iteration. However, it should be clear that fault models are not essential and search can be initialised anywhere in the model space.

An apparent and significant property of the iterative search based approach to *MBDS* is that the diagnostic process can start with an estimated system model and, thus, it does not require the absolute precision of the system model as in the dependency recording based approaches. Another important property of this approach is that it is very suitable for diagnosing dynamic systems, since such a diagnostic process can gain information from reasoning over time through the iteration. Also, this technique allows different knowledge to be utilised for candidate generation in *MBDS*, e.g., exploiting diagnostic knowledge in the way that *Mimic* does. In principle, the iterative search method can be applied in parallel, thereby, being able to diagnose multiple faults at a time. However, in contrast to the dependency recording approach

this method cannot guarantee to generate all the possible faults at any one time. In addition, due to the possibility of a local minimum (occurring when a diagnostic process terminates at a particular modelling dimension whilst better 'matching' between observations and predictions may be made if the search process is, otherwise, further performed against other dimensions), incorrect diagnoses may be produced. These properties will become clearer when we present the concrete diagnostic system established by using *FuSim*.

It is worth indicating that the iterative search method can be seen as a generalisation of the conventional numerical approaches to *MBDS*. Actually, although it appears to be new to the use of *AI* modelling techniques for diagnosis, it utilises the basic concepts of the traditional approaches to model reference reasoning in control and system identification [55, 71]. Methodologically, this is rather different from the dependency recording approach where, basically, a particular faulty component is determined if a retraction of its corresponding correctness assumption makes the predicted behaviour consistent with observations. In fact, the dependency recording based techniques accord with representing the converged system for all possible faults by presuming exact relationships between these faults and generated conflicts, thereby compiling an iterative feedback structure onto a fixed 'open-loop' system. Combined with the technique for synchronously tracking the evolution of physical systems, the *MBDS* system so developed is aimed at diagnosing continuous dynamic systems. However, we argue that, the generality of the underlying technique embedded in such a diagnostic system, i.e., iteratively searching the model space to identify faults, can also be utilised for diagnosing finite state and/or static systems.

7.3.2. Modelling Dimensions

As stated earlier, the convergence of iterative search based diagnostic process is very dependent upon choosing the appropriate directions of model variations to search. Also, model-based diagnosis is a sophisticated reasoning process, the complexity of system models must be taken into account so as to minimise the reasoning process. These require that the possible model variation directions be first identified and characterised. A degree of freedom of the model variations is hereafter called a *modelling dimension*. Four dimensions are defined such that the elements of each sub-space of models, characterised by the associated modelling dimension within the whole space of possible model variations, share some common properties. Considering the importance of identifying different modelling dimensions, it is worth mentioning that, although the present discussion focuses on supporting the iterative search diagnosis and is based on the *FuSim* algorithm, the analysis applies to all approaches to Qualitative Modelling [58, 67]. Furthermore, the establishment of these primitive modelling dimensions is also fundamental for other application systems, e.g., for an intelligent training system [62]. In general, such an approach underpins the current interest in work on multiple models [64, 108].

Abstraction

It is well known that the critical task in qualitative simulation is to decide on the form of the representation of system variables. In *FuSim*, the variables within a model take values from a fuzzy quantity space. The cardinality, with associated underlying semantics, of the quantity space is called the level of *abstraction* of the model. Thus, a sub-space of system models can be obtained by varying the level of abstraction utilised to represent the physical quantities. In fact, the term abstraction is herein used in order to

reflect the change in the quantity space in the sense that as the number of qualitative values which a variable may take decreases a more abstract description of this variable results. In other words, as the cardinality decreases the number of distinctions, or amount of detail, in representing the system variables also decreases. This immediately results in two important impacts on diagnosis. On the one hand, to reduce the complexity of system modelling and further that of the entire diagnostic process the most abstract system model is desirable to initiate search; on the other hand, a far too abstracted model may collapse behaviour distinctions necessary for distinguishing normal and faulty behaviours.

Commitment

For a particular system model, when the level of abstraction is chosen the description of the underlying range of the system variables on the quantity space may require the degree of certainty about such a description to be reflected. Within *FuSim*, different assignment of the membership functions for each qualitative value that a variable can take reflects different *possibility* that the variable should take the underlying real values. For a given quantity space, the particular form of the membership functions of the fuzzy qualitative values within this space is called the *commitment*. Clearly, modifying the form of commitment will result in a sub-space of models such that variables within a particular model take their qualitative values with different degrees of certainty. Commitment can be very useful within the diagnostic systems that have to cope with unreliable observations and/or ill-defined models. It is worth pointing out that, in addition to representing uncertainty by fuzzy sets, probabilistic distributions may also

be used to express commitment. However, apart from *FuSim*, as yet little work has appeared in representing uncertainty within Qualitative Modelling.

Resolution

Both abstraction and commitment are defined based on the assumption that the set of system variables that is of interest is fixed. However, in modelling real physical systems, the modeller must decide which of these variables to represent within the model and consequently which to ignore. The number of variables incorporated within a particular model is called the *resolution* of the model. The choice is very dependent upon the task and the precision to which the specification of that task is expressed. Modifying the resolution dimension, with respect to diagnosis, usually reflects structural changes due to physical effects, this typically involves the increase or reduction of the system order (number of variables). A subspace of models can, therefore, be attained by varying the resolution of the system model, i.e., from changing the representation of the given scope of the physical system with greater or less detail.

Relation Strength

Having established the resolution, abstraction, and commitment, the relationships between system variables can then be determined. Within *FuSim* these relationships are represented by either algebraic or derivative or functional constraints. In particular, functional dependencies are described by fuzzy relations holding against two or three variables, allowing both strength and sign information to be represented. Varying the level of the relational strength implies the modification of the 'gain' between the

related variables. Such a model variation direction is thus called the *relation strength* dimension. This results in a commonly used sub-space of models such as different dependency strengths in relations between velocity and distance caused by an increased friction or those between force and acceleration resulting from a reduced mass.

The conjunction of the sub-model spaces determined by the variations in each dimension distinguished above defines a space of models with a particular model being a point within this space depending upon the current choice on each modelling dimension. This visualisation of the model variations is important to determining an appropriate strategy in the search for unknown fault models.

7.4. Discrepancy Measurement

When dealing with diagnosis of continuous dynamic systems, it is important to notice that the inherent continuity of such systems presents a difficulty in defining when the behaviour represents a 'faulty' system. The behaviour of physical systems often continuously degrades from an acceptable set to an unacceptable set [63], implying that the space of fault models is also continuous. In fact, the design specification for a continuous dynamic system usually does not define a single (real-valued) correct behaviour but a set of behaviours that can satisfy the desired performance criteria. When the system is realised under the specification and works under the expected operating conditions it should produce an *acceptable behaviour* that broadly satisfies the requirements. However, under certain circumstances, behaviours that do not satisfy the performance specification of the system may exist, resulting in *unacceptable behaviours* that imply malfunctions or faults [63]. Even for a well engineered (or well designed and realised) physical system, unacceptable behaviours may still result from various sources. For instance, such behaviours may come from one or

more of the following: the parameter values drifting from their nominal ranges during the operational life-time of the system, the unexpected external disturbances exerted on the system, and the possible physical changes in the structure of the system.

Unfortunately, the exact boundary between such two sets of behaviours is, in general, not easy to identify due to the continuous degradation of the system behaviour. Unlike discrete systems where the definition of 'faulty behaviour' is straightforward: either the system is working correctly or not. For continuous systems, just what level of (behaviour) degradation determines a 'fault' condition can be very subjective. Figure 7.3 shows such a classification of acceptable or unacceptable system behaviours within which the shadow area represents the vague boundary between the two classes. It is, therefore, essential to have an appropriate *discrepancy measurement* method to detect the degradation such that the degree to which the system behaviour is degrading from the (ideal) acceptable set can be determined.

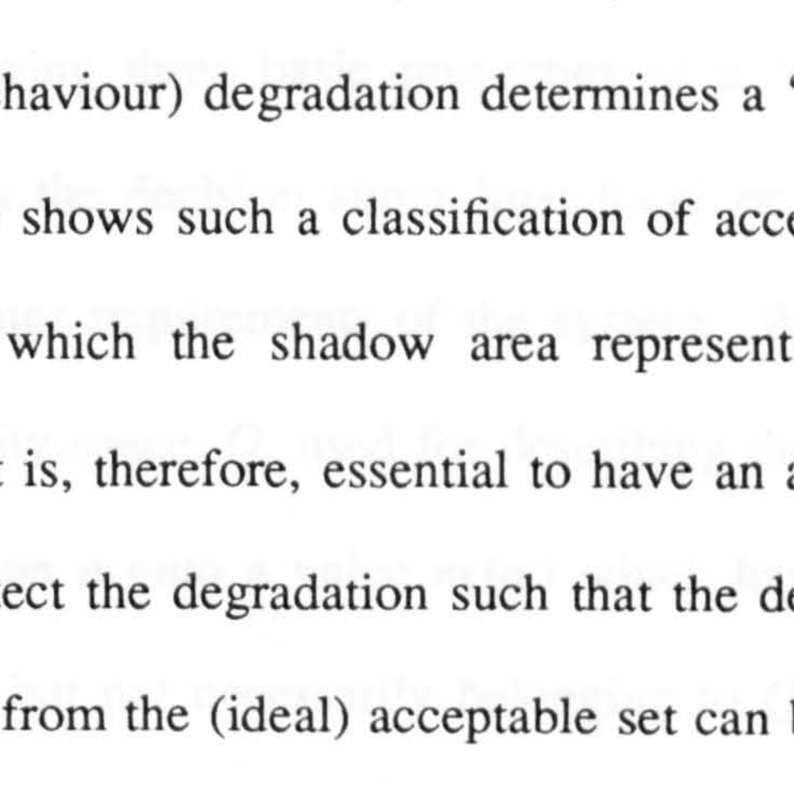


Fig. 7.3. *A Classification of Acceptable and Unacceptable Behaviours*

It is clear that the qualitative description of variable states provides a principled way to reduce the complexity in distinguishing infinitesimally close numerical behaviours. This eases the classification of acceptable and unacceptable behaviours.

However, qualitative representation does not completely remove the fundamental problem of the boundary vagueness between these two classes. In particular, a *discrepancy metric* is still required to detect how much difference between the intended (qualitative) model-based predictions that reflect unambiguous acceptable behaviours and the actual observations from the physical system. Of course, the representational form of such a metric is dependent upon the detailed representation of the physical quantities or the quantity space adopted by a particular qualitative simulator. Nevertheless, the following three basic properties of a 'metric' $d(.,.)$ must be held under the condition that the decision about 'matching' or 'unmatching' is made with respect to the performance requirements of the system. Within these properties, A and B belong to the quantity space, Q, used for describing the predictions and $m(.)$ maps a real-valued observation a onto a value $m(a)$ which has the same representation form as those within Q but not necessarily belonging to Q (in the sense of data interpretation). That is, to be able to compare predictions against observation (of course, at the same system state), a discrepancy metric such used must satisfy:

1) If A matches $m(a)$, $d(A, m(a)) = 0$.

2) If A does not match $m(a)$, $d(A, m(a)) > 0$.

3) In any case, $d(A, m(a)) = d(m(a), A)$,

   and $d(A, B) \leq d(A, m(a)) + d(B, m(a))$.

Apparently, this is an analogue of the axioms of a classical metric, though classical metrics are not bounded by any other performance criteria [87]. Various distance functions can be defined for this purpose and each reflects a different emphasis of the performance requirements of the system. When using *FuSim* as a behaviour simulator, a particular realisation of the distance function is the α distance defined in section 5.3.2.1. This will be discussed in detail in section 7.5.2.

It is worth indicating that the set of unacceptable behaviours may require further classifications to reflect different 'types' of such behaviours. For example, a set of unacceptable behaviours may be divided into two (or more) subsets such that it can be said that some behaviours slightly, whilst others largely, degrade from the acceptable behaviours, based on the use of different ranges of the discrepancy distances detected. Also, such a sub-classification may be done by grouping different unacceptable behaviours that might have been caused by similar underlying fault conditions. A deeper classification of system behaviours like these has an important impact on the development of useful strategies for organising diagnostic search. This will be exampled in the candidate generation process within the following *SID* system.

7.5. The Synchronous Iterative Diagnostic System: SID

This section presents the *SID* system, a realisation of the generic architecture shown in the last chapter for model-based diagnosis of continuous dynamic systems using the synchronised iterative search technique introduced in the previous sections. Within this diagnostic system, *FuSim* is utilised as the Behaviour Simulator. The resulting structure of the system is shown in figure 7.4. Being a particular implementation of the generic architecture, it is not surprising that, at the most schematic level, this figure appears similar to that used for describing the architecture (see figure 6.3). The corresponding software design of the *SID* system is given in Appendix C.

Before starting to describe each particular module within *SID*, it is important, however, to point out that the significant features of *FuSim* are essential to building an efficient and effective *MBDDS* system. As can be seen in the previous chapters, such useful attributes of *FuSim* include:

1) the temporal duration of the qualitative states is given;

2) stronger functional constraints can be utilised;

3) a substantial reduction in the spurious behaviours can be achieved as a product of 1) and 2); and

4) fuzzy sets allow the subjective element in system modelling to be incorporated and reasoned with in a formal way.

Built based on a Behaviour Simulator with these characteristics, the *SID* system allows faults that are developing, or are only observable, during the transient to be correctly identified. This will be verified throughout the following description of the diagnostic system and, also, by the example presented in section 7.6.

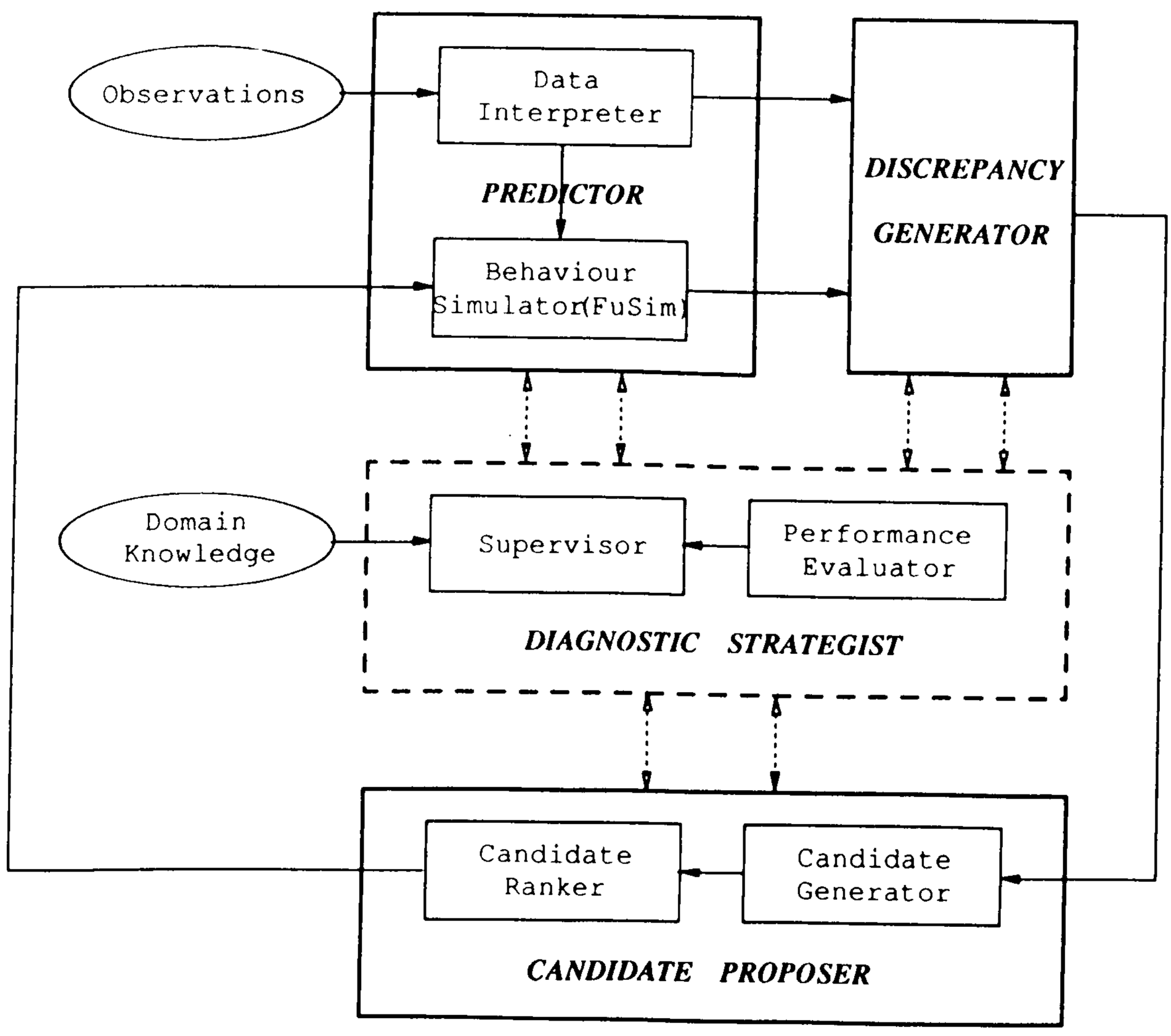


Fig. 7.4. *Structure of the SID System*

7.5.1. Predictor

As just mentioned, *FuSim* is utilised as the Behaviour Simulator within the Predictor. It is therefore straightforward to explain this module functionally in terms of the designed architecture as follows.

7.5.1.1. Data Interpreter

There are two potential uses of the real-valued inputs to and observations from the physical system within *SID*, one for detecting and generating discrepancies and the other for initialising (fuzzy) qualitative simulation. In response to these two different tasks, the Data Interpreter translates the inputs and observations into two different representation forms, though both are expressed by their associated 4-tuple normal and convex fuzzy numbers. For discrepancy generation, the Interpreter simply maps a given real number onto its corresponding (non-fuzzy) fuzzy number. That is to say, within *SID*, the mapping functor $m(.)$ given in section 7.4 is defined such that, for any $a \in R$, $m(a) = [a, a, 0, 0]$. For example, if two observed real values are 1 and 2, the interpreted observations will be [1, 1, 0, 0] and [2, 2, 0, 0], respectively. However, for simulation initialisation, the Interpreter maps a given real number onto its corresponding qualitative value with respect to the quantity space that the underlying variable takes values from by using the approximation principle. To be concise, the terms of inputs and observations to be mentioned in the following discussions will express those which have been interpreted by the Interpreter in connection with their intended application (generating discrepancies or initiating simulation), unless otherwise stated.

7.5.1.2. Behaviour Simulator

Based on the model (represented by means of a set of fuzzy constraints) and the interpreted operating conditions (inputs and observations) of the physical system, the Behaviour Simulator, i.e., the *FuSim* algorithm generates the predicted behaviour of the system. The behaviour so generated is described as a sequence of system states and an associated sequence of temporal durations. Generally speaking, different models and/or different operating conditions result in different behaviours. Which model is being simulated is determined by the Diagnostic Strategist via the focusing of attention and heavily dependent upon the outcome of the Candidate Proposer. Since *FuSim* was developed as a qualitative simulator in previous chapters, it is not necessary to re-describe in further detail how it predicts a system's behaviour.

7.5.2. Discrepancy Generator

As indicated earlier, diagnosing a physical system crucially relies upon the discrepancy detection between the observed and predicted behaviours of the system. The task of the Discrepancy Generator is, as reflected by its name, to decide on whether or not there is indeed a discrepancy between a prediction and an observation and, if so, to generate this discrepancy. Since both the predicted values and the interpreted observed values are represented by the 4-tuple fuzzy numbers, the prediction and the observation can generally be denoted as $P = [p_1, p_2, p_3, p_4]$ and $O = [o, o, 0, 0]$, respectively. The discrepancy generation is then realised by applying a set of rules to P and O, based on the satisfaction criterion indicating that once the membership value of the intersecting point of the P and O, or the crossing-degree (see section 5.3.2.1), is larger than a (given) threshold they are said to be matched with each other. These rules are deduced by the use of the α distance (which is defined in section 5.3.2.1 and satisfies the three basic properties of a discrepancy

'metric' shown in section 7.4) and listed in the following, where α reflects the desired degree of matching, i.e., the threshold which may be given by default or independently assigned by the Diagnostic Strategist on-line:

(1) If O intersects with P as illustrated by figures 7.5.a, 7.5.b, and 7.5.c, use the sub-rules below:

(1-1) If $o \in [p_1, p_2]$ (see figure 7.5.a), there is no inconsistency between P and O, P and O are matched (actually, the α distance between P and O equals *0*);

(1-2) If $o \in (p_2, p_2 + p_4(1-\alpha)]$ (see figure 7.5.b), there exists a minor inconsistency but P and O are still said to be matched based on the satisfaction criterion used, in the meantime, record the membership difference between them, i.e., $(o - p_2) / p_4$ (again, the α distance between the two equals *0*);

(1-3) If $o \in (p_2 + p_4(1-\alpha), p_2 + p_4]$ (see figure 7.5.c), there is an inconsistency between P and O and the discrepancy, $o - p_2 - p_4(1-\alpha)$, is recorded and regarded as a *type I* discrepancy (the amount of the discrepancy is the α distance between P and O).

(2) If O does not intersect with P as shown in figure 7.5.d, there is an inconsistency between P and O and the discrepancy, $o - p_2 - p_4(1-\alpha)$, is recorded and regarded as a *type II* discrepancy (also determined by the α distance).

Notice that, symbolically, both rule (1-3) and rule (2) have the same α distance in representation. However, they have rather different implications as indicated by the corresponding premises of rule (1) and rule (2). That is, within rule (1-3), the α distance is detected under the condition that O intersects with P, whilst this is not the case with rule (2). Also, rules (1-2), (1-3), and (2) give the method to detect and generate discrepancies when o is greater than p_2. For the other situation, where o is

smaller than p_1 the corresponding discrepancy generation rules are obtained by symmetry. The differences and/or discrepancies recorded with respect to the above rules are utilised for candidate ranking and may also be used to contribute to the performance evaluation of the diagnostic process (see later discussions in sections 7.5.3.2 and 7.5.4.1).

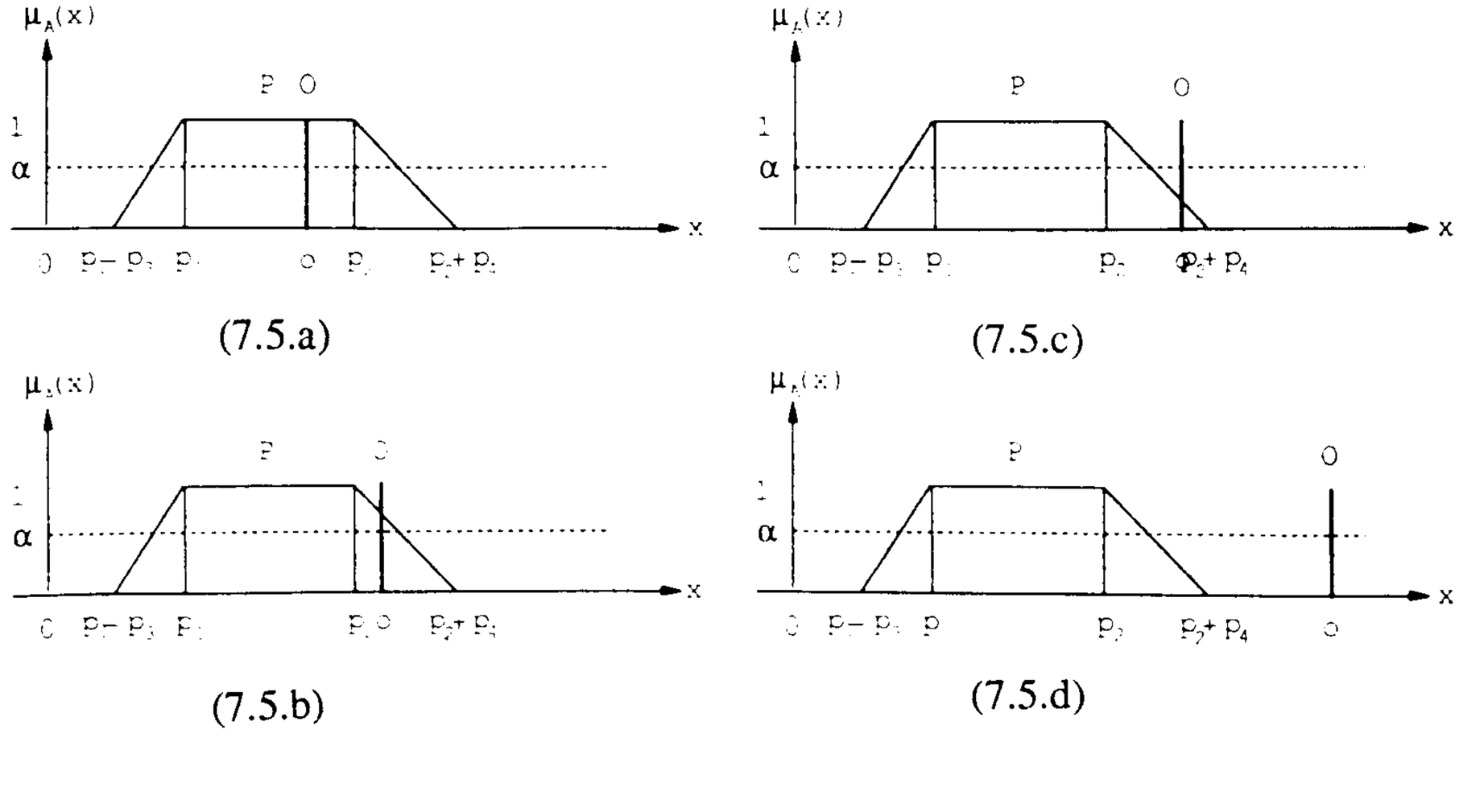


Fig. 7.5. *Discrepancy Generation*

It is clear from section 7.2 that the Discrepancy Generator accomplishes its task by synchronously comparing observations with predictions. This presents a problem in utilising a conventional qualitative simulation algorithm as the behaviour simulator within *MBDDS* systems due to the lack of explicit representation of temporal durations and qualitative ambiguity. Fortunately, *FuSim* is able to produce a temporal duration sequence associated with the state sequence and to substantially reduce the number of spurious behaviours generated, thereby enabling an effective and efficient discrepancy detection method to be developed with respect to the synchronous tracking technique previously described.

As a matter of fact, the integration of the capability of synchronising behaviour in *SID* (supported by *FuSim*'s explicit temporal descriptions) and the ability of reducing spurious behaviours in *FuSim* reduces the possibility of generating false positive and/or false negative diagnoses. By a false positive diagnosis, we mean a false alarm or a pseudo diagnosis that reflects an untrue operating condition of the physical system. By a false negative diagnosis, however, we mean that a particular fault has happened but has been missed by the diagnostic system. Usually, the false negative diagnoses are potentially more damaging. Without causing confusion, in later discussion false positive diagnoses and false negative diagnoses are simply called false positives and false negatives, respectively.

Regarding discrepancy detection, there are three basic sources that such false diagnoses can result from. The first is due to the wrong match made at different states of the predicted and observed behaviours or at different time (points or durations). The second is because of the accidental but, again, wrong match between a spurious prediction and an observation, though at the same time. Both of these result in false positives. The third, and the worst, results from failing to track a model until the appropriate time, thereby leading to a false negative if the model currently under tracking indeed reflects the actual faulty behaviour of the physical system. These three cases are respectively shown in figures 7.6.a, 7.6.b, and 7.6.c. Within *SID*, however, the opportunity of producing false negatives is eliminated and that of introducing false positives caused by the unreal matches at different times is reduced. This is because the Discrepancy Generator makes the comparison between a prediction and an observation only when the prediction is the last one generated with respect to the latest sampling time. Further, the false positives resulting from spurious behaviour matching can be considerably reduced because of the fact that *FuSim* itself generates,

if any, a limited number of spurious behaviours and that, again, the Discrepancy Generator only compares those predictions whose associated temporal durations match the sampling time of the current observation. Apart from the advantage of providing reliable discrepancies, it is also worth pointing out that, such a disrepancy detection mechanism can automatically guide the Predictor to generate only a definite and small number of predictions in response to each new sample, without requiring other control heuristics.

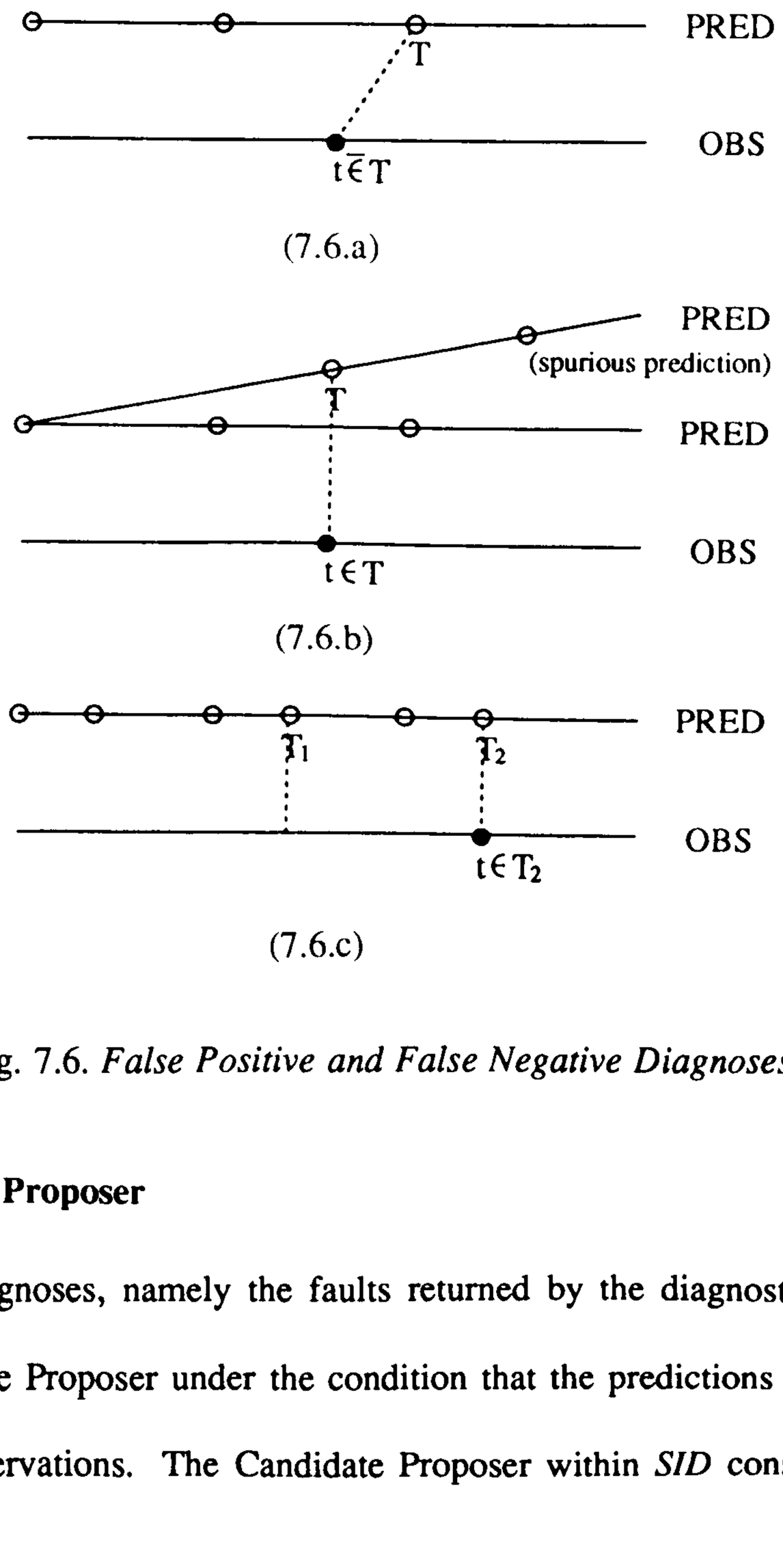


Fig. 7.6. *False Positive and False Negative Diagnoses*

7.5.3. Candidate Proposer

The final diagnoses, namely the faults returned by the diagnostic system, come from the Candidate Proposer under the condition that the predictions from the Predictor match the observations. The Candidate Proposer within *SID* consists of a Candi-

date Generator that, given discrepancies, generates possible faults and a Candidate Ranker which ranks amongst the candidates. As indicated in section 7.3, the Candidate Generator produces candidates through iteratively searching the model space. However, different iterative search based diagnostic systems may use different search rules. The first part of this sub-section is, therefore, dedicated to the description of the detailed search rules adopted by *SID* and, also, to the description of the basic technique for obtaining useful and explicit fault knowledge to increase search efficiency (though fault knowledge is not necessary for performing iterative search). The second part discusses the method presently used for candidate ranking which, from the efficiency point of view, has an important impact on candidate generation and hence on the entire diagnostic process.

7.5.3.1. Candidate Generator

It has been shown that the task of candidate generation is achieved by exploiting discrepancies through an iterative search process performed within defined model sub-spaces characterised by particular modelling dimensions. This requires that particular search rules be developed to map the discrepancies onto possible fault models and that basic modelling dimensions be characterised. The Candidate Generator in *SID* stores the four modelling dimensions within fuzzy qualitative simulation as defined in section 7.3.2. The search rules currently used reflect diagnostic heuristics and can be stated in the following, where modifying a model means to vary that model with respect to a particular dimension chosen by the Diagnostic Strategist:

Start with normal-behaviour model: if the discrepancy generated is type I modify this model; and if the discrepancy generated is type II choose the most likely fault model as the modified model, i.e., the candidate unless there do not exist any known faults. In case where no fault models exist,

the type II discrepancy also initiates the modification of the normal-behaviour model.

Start with modified model: if the discrepancy resulting from the modified model is less than that recorded previously, namely the α distance is shorter than the previous one, maintain this model as the candidate and substitute the previous discrepancy with the current one; if the discrepancy vanishes, i.e, the predictions obtained from this modified model match the observations, return the model as a final diagnosis; if the discrepancy is unchanged further modify this model; and if the discrepancy becomes larger than before discard this model as a possible candidate.

It is important, however, to notice that for simplicity the above rules are presented without indicating which direction of the modelling dimension is to be modified. In practice, without knowing fault models, the modifications of the normal-behaviour model will be performed against one direction first and, if no diagnosis is returned from this direction the search process will be re-done against the inverse direction, starting again with the normal-behaviour model. If the first modified model is that of a known fault, further modifications of it will be executed in both directions by viewing the fault model as an initial 'normal-behaviour' model, thereby searching in the same way as the situation where no fault models exist. It should also be noticed that, in principle, the candidate generation process governed by these search rules can be accomplished in parallel to generate multiple candidates. That is, the normal-behaviour model can be modified against multiple modelling dimensions at the same time. Nevertheless, whether or not this process is executed in parallel will not affect the basic diagnostic mechanism of *SID*, and hence, the following discussions are dedicated to diagnoses performed against one modelling dimension

only.

Clearly, the Candidate Generator generates candidates by making use of both the type and the amount of the discrepancies to guide the identification of faults. In fact, the candidate generation process is a systematic search process performed within the space of possible model variations and hence diagnosis can be refined over time. This avoids the need for explicit fault models and even allows the diagnostic process to start with an estimated system model of normal behaviour, since the precise model which reflects the actual operating condition of the physical system can eventually be automatically identified through the iteration. As such, the iterative search based diagnostic systems in general and *SID* in particular actually perform system identification within different model sub-spaces, by treating the physical system as the reference standard. It is in this sense that *SID* may be viewed as an analogue of traditional model reference approaches to adaptive control [55], within which the Behaviour Simulator, *FuSim*, is an analogue of traditional numerical simulators.

Although explicit knowledge about faults is not necessary, it can be effectively utilised to restrict the search space as reflected in the search rules if indeed such knowledge is available, a technique we term *fault-guided search*. This reflects an objective that, in real applications, fault models (if any) are constructed and/or provided for more severe or more obvious possible faults. If there is a type II discrepancy there is a heavy distortion of the physical system from the designed behaviour (or acceptable behaviours) and, thus, the most likely fault model should be tested first. We believe that using both fault models and normal-behaviour models in diagnosis, by distributing the fault models within the model space, exploiting these models to initiate fault hypotheses, and then, applying iterative search to refine or reject the hypotheses is beneficial to increasing the effectiveness and efficiency of the

diagnostic process.

As previously stated, an extreme case resulting from the candidate generation process is that, if each model sub-space consists of only normal-behaviour model and fault models, the Candidate Generator will degenerate into a look-up table such as the so-called Decision Tree within the *Mimic*-like *MBDDS* systems [33, 95, 96]. There are (at least) two ways to obtain such tables: either directly from the domain expertise as the first generation expert systems do or indirectly from simulating fault models. The following presents a formal method to show how the diagnostic knowledge embedded in such a look-up table can be induced from the results of simulating the fault models by *FuSim*. We use the basic technique reported in [33] to attain this. Essentially, each particular (foreseen) fault model of the physical system is defined in terms of a set of fuzzy qualitative constraints. Then, each such defined model is simulated using *FuSim*, starting from each possible initial state, to generate a complete behaviour tree for the model. A state is, herein, expressed by a fuzzy qualitative magnitude and a fuzzy qualitative rate of change. From each behaviour tree, training instances can be constructed by using the discrepancies between the states of the observable variables and those of the normal-behaviour model, generated from the same initial states, such that each instance is labelled with the fault model used for generating the tree. Finally, these training instances are compressed by an inductive learning programme [79] to form the desired decision table. Such a model-based learning technique provides good coverage of the available knowledge of faults [78]. Although this method may be rather time-consuming it only has to be done once and is performed off-line.

It should be noticed that, when making the off-line preparations to obtain the decision table for a particular physical system to be diagnosed, the comparison

between the behaviour of the normal-behaviour model of the system and the behaviour of a fault model is synchronously executed in a similar way to that described for tracking a model on-line as presented in section 7.2. Of course, the time used to guide and control the comparison is, now, not real-time but that resulting from artificially simulating the normal-behaviour model with the initial time point being zero.

7.5.3.2. Candidate Ranker

The Candidate Ranker ranks candidates on the basis of their associated discrepancies. This is required to reduce the number of candidates that would otherwise have to be considered simultaneously, so as to increase the efficiency and to limit the potentially existing computational explosion of the entire diagnostic process. Candidate ranking can be directly realised by using the recorded α distances (or discrepancies) since, obviously, the smaller the α distance measured, the more likely the associated candidate reflects the current operating condition of the physical system and, therefore, the higher the rank of the candidate should be. It efficiently reduces the search space of model variations. Without this, for example, a candidate returned by the candidate generation process based on a discrepancy of type I may well be evaluated (in the sense of tracking the candidate) at the same time or even before another resulting from the discrepancy of type II.

It is necessary, however, to point out that, evaluating the highest ranked candidate first does not guarantee generating a correct diagnosis when the predicted behaviour from the model matches the observed behaviour due to the false positive matching caused by the qualitative ambiguities. Unfortunately, false positive diagnosis is always a problem when a qualitative simulation algorithm is utilised as the behaviour simulator, even if the model space is exhaustively searched without using

the information on the rank of candidates. Nevertheless, as stated before, the number of false positives that may be generated by *SID* is significantly restricted by using *FuSim* as its Behaviour Simulator and by employing the synchronised discrepancy detection mechanism. Apart from ranking candidates, the information on the recorded membership differences, generated by the discrepancy generation rule (1-2), may be similarly used to rank amongst the returned diagnoses if necessary.

7.5.4. Diagnostic Strategist

The Diagnostic Strategist module controls the entire diagnostic process based on the on-line evaluation of the diagnostic performance, obtained by a sub-module called the Performance Evaluator, in connection with other knowledge resources such as domain-dependent heuristics. The control actions applied to the other modules of the diagnostic system are accomplished by another sub-module, named the (Diagnostic) Supervisor. Although important, the Diagnostic Strategist does not affect the central mechanism of iterative search based diagnosis since, essentially, such a diagnostic system merely involves the reasoning processes performed within the Predictor, the Discrepancy Generator, and the Candidate Proposer. Without a Diagnostic Strategist, in principle, the system should still be able to return diagnoses, though it may suffer from sever problem of computational explosion and/or very low efficiency in the search for fault models. Realising this, the section concentrates on the discussion about the major tasks of its two components but leave their concrete realisation as an important future work.

7.5.4.1. Performance Evaluator

As defined, a Performance Evaluator is used to assess the diagnostic performance. This is completed by evaluating the performance of the system model used,

the discrepancies generated, and the candidates proposed, subject to certain satisfaction criteria and/or basic diagnostic goals. The best performance is achieved if a diagnosis can reflect the real behaviour of the physical system using the highest abstraction, the lowest resolution, the simplest commitment, and the shortest time taken for diagnostic decision-making. The performance of the discrepancies are determined by how well those discrepancies are decreasing compared with the previously recorded ones. This is usually dependent upon the amount of the discrepancies reduced and also the rate of such reduction. The performance of a particular candidate may be reflected by the performance of the remaining discrepancies or by the number of possible faulty components within the returned fault model for a non-explosive situation. In general, such measures of performance are not independent of each other and, hence, a combined performance index may be utilised for the overall diagnostic performance.

7.5.4.2. Supervisor

The generic architecture presented in the last chapter emphasizes the control aspect of the diagnostic process by making such knowledge explicit within the Diagnostic Strategist module. Being the executor of the diagnostic strategies, the Supervisor guides and controls the diagnosis based on the evaluated performance and other knowledge resources outside of the *SID* system. The major tasks of the Supervisor in *SID* can be summarised as follows:

1) Decide on the levels of abstraction and commitment of the quantity space for system modelling and data interpretation;

2) Determine which model to be used as the system model for (normal) behaviour simulation;

3) Select the thresholds for discrepancy detection and generation;

4) Choose appropriate modelling dimensions to search for candidate generation;

5) Indicate particular interesting fault areas and fault types for bounding the search and/or change the next (starting) point of modifying a model within the model space for speeding up the diagnosis.

It is clear that these tasks are aimed at controlling different modules within *SID* with respect to the generic architecture. In fact, the results of the first two actions are applied to the Predictor, usually named the focus of attention, and those of the last two actions are presented to the Candidate Proposer, normally called the focus of suspicion. The outcome of the third action is obviously utilised by the Discrepancy Generator.

Creating a particular instantiation of this supervision sub-module may be facilitated by offering a language for expressing such control knowledge and/or control heuristics via a control algorithm. To be consistent with the theoretical basis upon which the Behaviour Simulator, *FuSim*, is built, it is natural to propose that, a fuzzy logic controller [68, 69, 72] may be employed to realise this. Using a fuzzy controller as the Supervisor will ease the information exchange between the Diagnostic Strategist and other modules, namely, the Predictor, the Discrepancy generator, and the Candidate Proposer. More importantly, a Supervisor constructed in this way will be able to merge the experience, heuristics, and intuition of domain experts in performing the control. However, to develop a concrete fuzzy controller for this purpose is, of course, beyond the scope of this thesis. Nevertheless, this approach seems promising and is the one which we are currently pursuing.

Instead of establishing an actual method to implement a fuzzy controller to perform the diagnostic supervision, we herein simply give a conceptual description of

such fuzzy controllers in general. As required, the Supervisor usually takes as inputs the performance indices of candidates and discrepancies and, perhaps, the performance indices of the rates of change of the discrepancies. Based on a set of linguistically described tactical knowledge and the applications of fuzzy implication and the compositional rule of inference, the supervising actions to be executed can be obtained (see section 3.3). Of course, basic control knowledge and/or control heuristics must be available when developing the diagnostic system. Such tactical knowledge is often expressed in the form of conditional statements like 'if *diagnostic performance*, then *control action(s)*'. In so doing, the Supervisor produces its outputs in terms of indicating a particular adjustment of the Discrepancy Generator and/or changing a particular focusing content applied to either the Predictor or the Candidate Proposer, thereby accomplishing its major tasks as previously listed.

7.6. Example

A simple example, of 'a mass on a spring', as depicted in figure 5.13 in Chapter 5, is given to demonstrate how *SID*, the synchronised iterative search based diagnostic system, determines faults using *FuSim*. For clarity and self-containability, the results of modelling and simulating this physical system under the frictionless condition are first briefly reviewed. Then, results of diagnosing faults against the relation strength dimension are presented.

7.6.1. Review of Normal-Behaviour Model and the Behaviour

Suppose that the normal-behaviour model assumes frictionless motion. Thus, as shown in section 5.6.2, the physical system can be characterised by two derivative and one functional constraints as follows

$$deriv\ x = v, \qquad deriv\ v = a,$$

$$\begin{bmatrix} a \tilde{} x & -b & -0.6 & -s & 0 & s & 0.6 & b \\ -b & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ -0.6 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ -s & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ s & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0.6 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ b & 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

The first and second equations establish the ordinary derivative relationships holding amongst the distance, velocity, and acceleration of the mass. The third functional relation between a and x is a weak, but stronger than monotonic operator, form of Hooke's law represented as a degenerated fuzzy relation; with each value corresponding to a perceived meaning, for instance, $-b$ denoting *negative big* and s indicating *positive small*.

With such a system model and assuming that the mass is moved away from the equilibrium point, $x = 0$, to $x = 0.6 > 0$, and then let go. *FuSim* produces a unique behaviour for each system variable, as shown in figure 5.14 (one cycle only), with the following durations associated with the fuzzy qualitative states:

$$t_0 = 0, \quad t_1 \in [1, 11], \quad t_2 \in [1.09, 12], \quad t_3 \in [2.09, 23] \quad t_4 \in [2.18, 24],$$

$$t_5 \in [3.18, 35], \quad t_6 \in [3.27, 36], \quad t_7 \in [4.27, 47], \quad t_8 \in [4.36, 48].$$

7.6.2. Diagnosing Faults against Relation Strength Dimension

Let us assume that the diagnostic system starts monitoring the physical system from its initial operating condition. For notational simplicity, the interpreted observations in *SID* are still represented in their original real-valued form in the following description of the diagnostic process. When an observation is obtained at $t_1 = 1.5$ from observable variables (x, v) such that $OBS(1.5) = (0.3, -0.4)$, where the observed

values are normalised with respect to the underlying numerical ranges of the variables. Based on the normal-behaviour model, *FuSim* predicts the immediate next state from the initial one, $(s, -s)$, which matches with the observation under the condition that the predicted state temporally covers the observation time (see the result shown in the preceding sub-section). This indicates that the physical system is performing normally. In the same way, two further observations, $OBS(2.6) = (0, -0.6)$ and $OBS(3.7) = (-0.35, -0.45)$, also guide the system model to make the next two predictions that match with the observations, providing synchronous tracking of the normal-behaviour model. Consider that the fourth observation is obtained such that $OBS(4.8) = (-0.45, 0)$, whose temporal information is used, as before, to guide *FuSim* to produce a predicted state. However, from the model of normal behaviour *FuSim* generates a unique state $(-0.6, 0)$ as the successor state of the previous state $(-s, -s)$ (matched with $OBS(3.7)$), thereby resulting in a type II discrepancy with the variable x (with respect to the discrepancy generation rule (2)).

Now that the discrepancy between the current observation and prediction has been detected and *FuSim* produces unique behaviour from the normal-behaviour model, this model is discarded and, hence, *SID* assumes that the physical system is 'faulty' and the original monitoring task becomes a diagnostic one. To be concise, assume that the Diagnostic Strategist only requires the Candidate Generator to search the model sub-space characterised by the relation strength dimension. Within this dimension, it is presumed that, there is a known fault model that reflects a mass-stuck condition, equivalent to a friction system with an infinite damping coefficient, and is described by

$$x = x_0, \qquad v = a = 0,$$

with x_0 being the displacement where the mass is stuck. Since the discrepancy generated is type II the candidate evokes this model as the possible fault, thereby beginning the fault-guided search. With the current state ($OBS(4.8)$) being $(-s, 0)$, which can, of course, be expressed directly by (–0.45, 0) if it is not intended to use any quantity space to qualitatively represent this simple model, the prediction $(-s, 0)$ is generated under the guidance of the temporal information obtained from the next observation $OBS(5.9) = (-0.25, 0.35)$. Clearly, this prediction does not match with the observation. Henceforth, the stuck model does not reflect the actual operating condition of this mass-on-a-spring system and can then be exonerated from being the suspect.

After eliminating the mass-stuck model the Candidate Generator tries to modify it against the current modelling dimension to find the diagnosis following the search rules. Against the relation strength dimension and, in particular, between non-friction and infinite friction, it is natural to assume that the system is currently experiencing a non-zero friction condition with the damping coefficient being *medium*, although in a real situation it may also necessary to consider cases with *small* or *big* damping coefficients. The friction condition can be modelled by five fuzzy constraints, namely,

$$deriv\ x = v, \qquad deriv\ v = a, \qquad a = a_1 + a_2,$$
$$a_1 \tilde{}\ x, \qquad a_2 \tilde{}\ v;$$

where additional variables a_1 and a_2 are introduced to include the effectiveness of the friction. Two fuzzy relations, $a_1 \tilde{}\ x$ and $a_2 \tilde{}\ v$ are modelled in a very similar way to the weak form of Hooke's law shown earlier. Starting with the state $(-s, 0)$ (with respect to the $OBS(4.8)$) and guided by the sampling time of $OBS(5.9)$, the prediction from this friction model is $(-s, s)$ which matches the new observation. The diagnos-

tic system thus returns the *medium* friction condition as the fault within the physical system. Alternatively, it may continue tracking this possible fault if it is necessary to confirm that the physical system is suffering from this fault. For instance, from two further made observations, $OBS(6.0) = (0, 0.45)$ and $OBS(7.1) = (0.2, 0.3)$, *FuSim* will generate two matched states based on the friction model: $(0, s)$ and (s, s).

The whole monitoring and diagnostic process explained above is shown in figure 7.7 (for the variable x), where the temporal points $(t_0, t_1, ..., t_7)$ satisfy those given in the previous sub-section. Finally, it is important to realise that, without using fault guided search or if the mass-stuck model was unknown, the possible underlying fault, the *medium* friction would still be identified, though it might be necessary to check the *small* friction condition first after shifting monitoring to diagnosis.

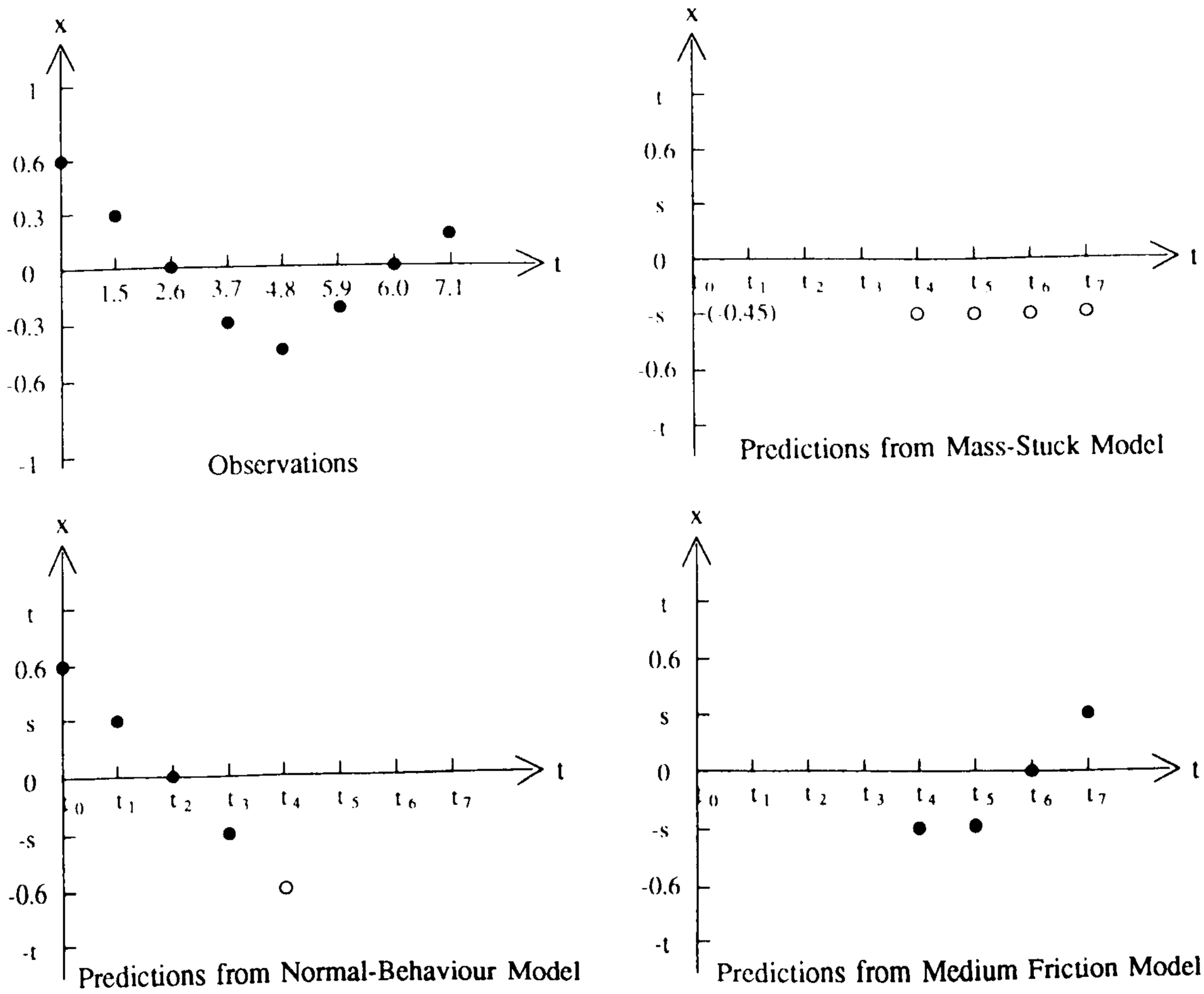


Fig. 7.7. *Monitoring and Diagnosis*

7.7. Summary

This chapter presents an approach to model-based diagnosis of continuous dynamic systems using *FuSim*. Although diagnosing continuous dynamic systems using *AI* techniques is just at its beginning, *FuSim* makes such a task feasible. The resulting diagnostic system, *SID*, complies with the architecture of *MBDDS* and, more importantly, presents a number of important contributions to *MBDDS*. Firstly, a technique for the synchronous tracking of the evolving behaviour of a dynamic physical system is developed, allowing the synchronous detection of discrepancies between the observations and predictions of the system. Supported by the temporal durations produced by *FuSim*, the synchronous tracking between predictions and observations eliminates the possibility of false negative diagnoses, while the reduction in spurious behaviours resulting from the use of *FuSim* reduces the possibility of false positive diagnoses, without the need for numerical information or heuristics. Secondly, a proposal is made for iteratively searching for fault models which produce predictions that match the observations, offering the prospect of utilising fault models, approximate fault models, and non-fault-models within the same framework. This technique differs from traditional dependency-recording based approaches to *MBDS* that are restricted to exact and static models of physical systems, enabling continuous dynamic physical systems having feedback loops to be diagnosed. Thirdly, the representation of fuzzy qualitative values enables the establishment of a discrepancy metric for detecting degradation of the behaviour of a continuous system from acceptable condition to unacceptable conditions and for matching predictions with observations. In addition to these developments in *MBDS*, this chapter also opens some significant future work. Most of which will be tackled within the *ARTIST* project (see discussion about future work in the next chapter).

CHAPTER 8

CONCLUSION

8.1. Summary of the Thesis

This thesis has developed two novel techniques within the field of Artificial Intelligence in Engineering: a fuzzy qualitative simulation algorithm, named *FuSim*, and a synchronous iterative diagnostic method, called *SID*. The fuzzy simulation technique underlying the *FuSim* algorithm provides a general theoretical framework for qualitatively modelling and reasoning about continuous dynamic systems with incomplete or imprecise knowledge. The principle of synchronised iterative search within a model space offers the prospect of monitoring and diagnosing continuous dynamic systems based on the description of system structure, without necessarily being supplied with comprehensive symptom-failure associations as in rule-based diagnostic systems. These two approaches are naturally combined through the adoption of *FuSim* as the key component, i.e., the behaviour simulator, within *SID*. The *SID* technique is now being incorporated within the *ARTIST* project (*ESPRIT II*, P5143) that is to establish an advanced reasoning tool for model-based diagnosis of industrial systems, to which the author is currently contributing.

The evolution of Qualitative Reasoning (*QR*) is at a critical state: the excitement and euphoria that surrounded the initial work has given way to realisation of the inadequacies of the basic techniques and to the consequent limitations for practical applications, whilst more recent extensions aimed at reducing the limitations remain at the experimental stage. In particular, the most evident of the recent proposals, made to extend the representation of physical quantities to allow more detailed description

of variables but without requiring the use of the real numbers, still encounters various difficulties, especially when interpreting the quantity space used with respect to the standard real number line. This is discussed within the review of the developments in *QR* presented in chapter 2.

As indicated in chapter 3, the theory of Fuzzy Sets shares a similar motivation to that of *QR*, handling complexity in modelling and reasoning about physical systems but has been developed by a different research community. With such an understanding, to ease the representation of uncertain knowledge and/or resolve the difficulty in interpreting qualitative values, we have utilised fuzzy sets defined on the real number line to describe the quantity space within *FuSim*. The work on fuzzy simulation therefore synthesizes two previously disparate research areas.

Basically, to simulate the behaviour of a physical system, the system is modelled by a set of fuzzy constraints that may include algebraic, derivative, and function relations, defined upon a fuzzy quantity space chosen with associated semantics. The state of a system variable is described by a pair of qualitative magnitude and qualitative derivative. For a given initial state, the possible state transition rules determine its plausible successors based on continuity. The temporal duration within which a state stays or takes to reach its next state, namely, the persistence time and arrival time respectively, is obtained following the corresponding computational method. The spurious states are eliminated by the use of constraint filters (implemented by the Waltz algorithm for efficiency), temporal filters, and simple global filters (checking for no-change and repeating states). Details about these have been extensively discussed in chapters 4 and 5.

In general, we have shown that the *FuSim* algorithm has four distinct advantages over existing methods as follows:

1) It allows a more detailed description of system variables, through an arbitrary, but finite, discretisation of the quantity space. Such a quantity space possesses the desired properties of finiteness, coverage, and granularity for qualitatively representing physical quantities. The use of graded membership within the fuzzy quantity space, albeit in a parameterised form, allows the subjective element of common-sense knowledge to be incorporated in the basic description of the quantity space.

2) It enables semi-quantitative information about the strength, as well as the sign, of functional dependencies to be represented and reasoned with in a formal way by the use of fuzzy function relational constraints. This is an important practical advantage, in that, imprecise and partial numerical information about functional relationships is often known, although not in enough detail to develop a well-posed numerical model. Current techniques do not make use of such knowledge or do so in an ad hoc manner.

3) It allows the rate-of-change to be defined on the full fuzzy quantity space, thereby relative rates of change can be represented and temporal durations of system states computed. This results in a powerful temporal filter and enables the behaviour of a physical system to be expressed by a state sequence associated with a temporal sequence that vividly reflects the evolution of the system states.

4) It considerably reduces the number of spurious behaviours that would otherwise exist due to the qualitative ambiguity. This is a direct benefit attained from the second and the third advantages. Because of this, as justified by experimental results, computationally very expensive global filters are not required to generate a unique behaviour for a second-order system except for the simple no-change

and repeating ones.

Although, during the last decade, a number of general and powerful Model-Based Diagnostic Systems (*MBDS*) have been constructed using *AI* modelling techniques, almost all of them are unable to cope with continuous dynamic systems. It is this observation that inspired the research to utilise qualitative simulation in general and *FuSim* in particular within *MBDS* for continuous dynamic systems. To formalise the discussion, we have introduced a generic architecture which can include different techniques for diagnosing either static or dynamic systems.

A number of significant advances in the development of model based diagnosis of continuous dynamic physical systems are proposed within the thesis:

1) A technique for the predictions from models to synchronously track the behaviour of dynamic systems, without the need for heuristics, has been presented.

2) A proposal for iteratively searching for fault models has been made that produce predictions which match the observations. This procedure seems to be very promising, offering the prospect of utilising fault models, approximate fault models, and no fault knowledge models within the same framework.

3) A mechanism for detecting continuous degradation of system behaviour and/or for matching predictions with observations has been described.

Such a synchronous iterative diagnostic method for *MBDDS* represents a radical departure from previous *AI* based methods for *MBDS* that use dependency recording. The fundamental principle of iterative search and model dimensions provides an entirely new approach to *MBDS* and is likely to see many subsequent developments. The proposed approach brings *MBDS* closer to conventional control system structures. As such, much can be gained from examining previous (numerically based) model-

based approaches for adopting within *MBDDS*.

Being a particular implementation of this synchronised iterative search method, *SID* performs diagnosis by specifying all the four major modules of the introduced generic architecture: a Predictor, with *FuSim* being utilised as the Behaviour Simulator, to generate predictions about the states of the physical system to be diagnosed; a Discrepancy Generator to detect and generate discrepancies between the predictions and the observations; a Candidate Proposer to produce possible faults existing in the system based on the utilisation of the discrepancies; and a Diagnostic Strategist to control the entire diagnostic process. In particular, combined with the synchronous tracking technique, the adoption of *FuSim* that produces temporal durations and less qualitative ambiguities minimises the possibility of the generation of wrong diagnoses. Benefited from representing physical quantities in terms of fuzzy qualitative values, the discrepancy generation mechanism of *SID* determines both the type and the amount of discrepancies that are crucial for initiating the search and ranking the possible faults. Diagnosis is refined over time by applying the iterative search technique to identifying fault models within the model space built upon (four) different modelling dimensions.

FuSim and *SID* have been applied to simulate and diagnose second-order systems and the results are given in the thesis. Their software architectural designs are presented in Appendices B and C, respectively. Most of the work presented in the thesis has been published as shown in Appendix A where a list of publications related to this thesis is given.

8.2. Future Work

For the future many developments are foreseen. This final section of the thesis presents suggestions for such work with respect to the two developments established

in the previous chapters.

As a common important future work for both *FuSim* and *SID*, it is worth indicating that, within the *ARTIST* project, a major task is to build a practical model-based diagnostic system for a crude oil refining distillation plant (operated by the *CEPSA* company, Spain). Such a plant, in particular, its distillation columns, presents an ill-defined and complex system structure and exhibits continuous dynamic behaviour [2]. It is, therefore, our intention to utilise the techniques developed within this thesis as the basis of a general tool to model and diagnose the plant, especially the distillation column. To start with, simplified dynamic models of a distillation column are required. Such models are currently being constructed and evaluated using the actual data coming from the refinery. Based on so developed models, the strength and weakness of the tool will be further identified and potentially necessary improvements and/or extensions will then be made. When this is accomplished, the tool is intended to be applied to handle more detailed parts of the distillation column and, perhaps, to deal with the full scale refining plant.

Regarding each of the two developments, the useful investigations foreseen are given in the following two sub-sections, respectively.

8.2.1. Suggestions for Work on FuSim

The currently implemented version of *FuSim* uses a predefined quantity space for each particular simulation task. An improvement will therefore be to extend this version, if necessary, to automatically generate or remove some qualitative values within a quantity space during a simulation process. This means that the quantity space used would have an 'adaptive' capability in adjusting the degree of its abstraction with respect to the new information obtained from the simulation. For example, if qualitative ambiguity always exists in the choice between two particular qualitative values

within certain period of simulation, there are then (at least) two alterative ways to handle this by manipulating the quantity space. One is to define a new qualitative value which covers the original two values, especially when dealing with rates of change. Actually, in the *QR* literature, a similar technique termed *chattering constraints* has been reported [54]. The other is to further divide the underlying numerical range into three or more qualitative values. Of course, there is a trade-off between the reduction in ambiguity and the effectiveness and efficiency of the resulting simulation. Also, whether to increase or to decrease the cardinality of a quantity space is dependent upon the characteristics of the system being simulated. This is the very reason that an 'adaptive' method is required. Since *FuSim* allows the algebraic operation system to remain unchanged whatever quantity space is used, this possible modification would not result in essential change of its filtering mechanism. The detailed representation of the degenerated fuzzy relations would, however, have to be modified accordingly. It is worth pointing out that there exists a conceptual difference between the automatic generation of 'landmarks' in *QSim* and the use of a denser quantity space in *FuSim*. Actually, due to the non excluded-middle property of the fuzzy representation, intuitively, it is not so meaningful, and not easy, to produce many single real numbers (landmarks) as new qualitative values during the simulation process, unless the underlying uncertainty has been reduced.

At its present state, *FuSim* lacks a way to check if there is any redundant information existing in the function relational constraints provided by the user. Although this will not affect the run-time performance of the algorithm, the off-line preparation for such constraints involves redundant computations. Further, if a quantity space is to be able to adaptively change its own elements during a simulation process as discussed above, this will then result in unnecessary on-line computation and, perhaps,

set up an obstacle to designing the 'adaptive' method. This work should be considered together with the above-mentioned extension.

Present experimental simulations have all been applied to second (or lower) order systems. The most interesting work in the future is to model and simulate physical systems with higher order. This would help to further check the ability of *FuSim* in generating less spurious behaviours. For such purposes, it may well be necessary and beneficial to create other global filters different from the two simple ones that have been implemented, but without significantly increasing the overall computational complexity. Certainly, this would be a hard piece of work.

As indicated before, an interesting variation in the development of Qualitative Simulation is to develop *constructive* simulation algorithms. As argued in [119, 120], such techniques are generally more efficient than *non-constructive* ones (for which *QSim*, *HR-QSIM*, and *FuSim* are examples), and they can reflect an intuitive sense of causality in the simulated behaviour of the physical system. This, of course, is useful for application tasks such as explanation and diagnosis. However, the existing constructive algorithms utilise two separated mechanisms to generate and maintain variable states and their associated temporal information. It would, therefore, be beneficial to be able to integrate the modelling and temporal reasoning techniques in *FuSim* into a constructive algorithm. In fact, an important sub-task within the *ARTIST* project is dedicated to this.

After all, we should like to consider applying *FuSim* to different automated reasoning tasks. Clearly, at the time of writing this thesis, it has been effectively utilised only within the model-based diagnostic system *SID*. However, realising that *FuSim* has been developed as a general, qualitative modelling and simulation tool capable of exhibiting the behaviour of continuous dynamic systems when uncertain knowledge is

presented, it would be very interesting to see how it works within other potential application systems such as intelligent control, intelligent design, and/or intelligent tutoring systems.

8.2.2. Suggestions for Work on SID

The present realisation of the *SID* framework utilises the *FuSim* algorithm as the Behaviour Simulator. However, the fuzzy aspects are not essential to the synchronous tracking or to the iterative search procedure. In the former case, what is required is a simulation tool that produces the estimate of the duration of a qualitative state; whereas, in the latter case, a well formed modelling language supporting adequate modelling dimensions is essential. Thus, it will be interesting to be able to substitute *FuSim* with another qualitative simulator which satisfies these two requirements. Actually, this forms another research sub-task within the *ARTIST* project that is related to the integration of constructive and non-constructive algorithms mentioned earlier. An important side-effect from doing so would be that this may involve substantial alteration of the existing discrepancy generation methods. In addition, an uncertainty measure is, of course, still necessary if it is desired to exploit the commitment dimension that, as yet, may not be readily available with other qualitative simulators.

Experiments so far have utilised a very simple Diagnostic Strategist with a performance evaluation equal to the discrepancy measurement and 'hard-wired' guidance for search actions. However, the generality of the diagnostic method proposed allows for more complex performance evaluation, including the rate of change of the discrepancies and the time taken to achieve an 'acceptable' diagnosis. This is critical for extending the existing implementation into the real-time domain. Based on such evaluated diagnostic performance and other domain knowledge available, useful

search actions can be deduced through using a particular inference mechanism. As pointed out earlier, we intend to realise this via a fuzzy logic controller that shares the common mathematical basis as the Behaviour Simulator, *FuSim*. In particular, the basic principle embedded in Self-Organising Fuzzy Logic Controllers is very attractive [68, 69]. It allows the control rules relating the diagnostic performance and domain heuristics with specific diagnostic actions to be updated in an adaptive way. That is, a Diagnostic Strategist so implemented may start with a small group of control rules, with the accumulation of knowledge during a diagnostic process, it is able to modify them and generate new control rules. Of course, this requires an extension to the Performance Evaluator such that the performance of the Diagnostic Superviser within the Strategist itself can be evaluated.

The *SID* diagnostic method is essentially dependent upon choosing the correct modelling dimensions to perform the search. Four dimensions defined in the thesis provides the basis for characterising a space of possible model variations. Nevertheless, detailed influence on the effectiveness and efficiency of the entire diagnostic process that such dimensions may exert has not yet been exhaustively investigated. Of course, this is related to the concrete realisation of the Diagnostic Strategist. It is expected that further systematic investigations into the model space will significantly promote the development of powerful diagnostic strategies. Furthermore, these investigations would benefit the discussion about the principled use of the multiple models of a system. In fact, the utilisation of multiple models requires a clear understanding of the process by which models can be developed from one another and the consequent relationship between them. This is because the goal of such utilisation is to start with the most abstract, least resolution, least commitment, and weakest relation strength and 'pilot' within the model space until a particular application task is satis-

factorily achieved. This approach is still in its infancy and model switching on one dimension only has been reported so far [115]. Research on this issue forms another key part of the *ARTIST* project.

In principle, *SID* should also be able to diagnose static systems, though this would make the synchronised tracking of the behaviour of such systems meaningless. However, in the case that accurate normal-behaviour or fault models are known the existing (dependency-recording based) approaches may be more effective, and avoid the need to search by iteratively adjusting models. Nevertheless, we believe that a combination of these approaches may prove to be very general and effective. This is, again, one of the major tasks to be tackled within the *ARTIST* project. An attractive idea is that, in order to diagnose a dynamic system effectively and efficiently, we may first view the system being described by a model with a low level of resolution and a high level of abstraction. At such levels, the system may be described by a static model within which, say, a real sub-system or an original internal feedback loop can be treated as a single 'component' and, therefore, the dependency-recording techniques are directly applicable. As a result, one or a few such 'components' may be produced as faulty ones. Then, the synchronous iterative diagnostic method is applied to pinpoint detailed faults by focusing on such sub-systems that reflect system dynamics. The feasibility of such developments is currently being further investigated.

Ultimately, it is with favourable expectation that the author looks forward to results from future experimental and application work of both the fuzzy qualitative simulation and the synchronous iterative diagnosis techniques.

APPENDIX A

LIST OF PUBLICATIONS

A.1. Papers Published

[1] R. R. Leitch, H. Freitag, Q. Shen, P. Struss, and G. Tornielli, ARTIST: A Methodological Approach to Specifying Model-Based Diagnostic Systems, in: G. Guida and A. Stefanini (Eds.), *Proceedings of European Conference on Industrial Applications of Knowledge-Based Diagnosis*, pp. 183-208, Milan, Italy, 1991.

[2] R. R. Leitch and Q. Shen, Finding Faults with Model Based Diagnosis, in: *Proceedings of the 2nd International Workshop on Principles of Diagnosis*, pp. 121-134, Milan, Italy, 1991.

[3] R. R. Leitch and Q. Shen, Model Dimensions within Fuzzy Qualitative Simulation, in: *Proceedings of the 13th IMACS World Congress on Computation and Applied Mathematics*, Vol. 3, pp. 1212-1213, Dublin, Ireland, 1991.

[4] Q. Shen and R. R. Leitch, Application Studies of Fuzzy Qualitative Simulation, to appear in: *IMACS Transactions*, 1992.

[5] Q. Shen and R. R. Leitch, Combining Qualitative Simulation and Fuzzy Sets, in: B. Faltings and P. Struss (Eds.), *Recent Advances in Qualitative Physics*, MIT Press, 1991.

[6] Q. Shen and R. R. Leitch, Synchronised Qualitative Simulation in Diagnosis, in: *Proceedings of the 5th International Workshop on Qualitative Reasoning about Physical Systems*, pp. 171-185, Austin, Texas, 1991.

[7] Q. Shen and R. R. Leitch, Diagnosing Continuous Dynamic Systems Using Qualitative Simulation, in: *Proceedings of the 3rd International Conference on Control*, Vol. 2, pp. 1000-1006, Edinburgh, Scotland, 1991.

[8] Q. Shen and R. R. Leitch, Extending the Quantity Space in Qualitative Reasoning, in: *Proceedings of the 1st European Workshop on Qualitative Reasoning about Physical Systems*, pp. 1-14, Genova, Italy, 1991.

[9] Q. Shen and R. R. Leitch, Integrating Common-sense and Qualitative Simulation by the Use of Fuzzy Sets, in: *Proceedings of the 4th International Workshop on Qualitative Reasoning about Physical Systems*, pp. 220-232, Lugano, Switzerland, 1990.

[10] Q. Shen and R. R. Leitch, A Semi-Quantitative Extension to Qualitative Simulation, in: *Proceedings of the 10th International Conference on Expert Systems and their Applications*, EC2, pp. 223-234, Avignon, France, 1990.

A.2. Papers under Review

[11] Q. Shen and R. R. Leitch, Extending Qualitative Simulation by the Use of Fuzzy Sets, submitted for conference publication, 1991.

[12] Q. Shen and R. R. Leitch, On Extending the Quantity Space in Qualitative Reasoning, submitted for journal publication, 1991.

[13] Q. Shen and R. R. Leitch, Fuzzy Qualitative Simulation, submitted for journal publication, 1990.

A.3. Deliverable Report

[14] H. Freitag, R. R. Leitch, Q. Shen, P. Struss, and G. Tornielli, Model-Based Diagnostic System Reference Architecture, *ESPRIT Project P5143*, 34pp, 1990.

APPENDIX B

SOFTWARE ARCHITECTURAL DESIGN OF FUSIM

This appendix presents the software architectural design of the *FuSim* algorithm with respect to the algorithm structure given in section 5.5 (see figure 5.2 for its functional overview). Simple explanatory notes are presented in braces. As indicated in section 5.5, this algorithm has been completely implemented in Quintus Prolog [11] and run on a SUN 3 workstation with 16M bytes of RAM. The implementation conforms to the following design.

```
initiate-fusim(set-of-variables,
                    quantity-space,
                    set-of-constraints,
                    initial-state,
                    resource-limit)

{performing initialisations of FuSim,
 including clearing various recorders used}


run-simulation()

    checked-current-state := run-waltz-filter(initial-state)
    if    checked-current-state = null
          return invalid-initial-state
          exit
    else record checked-current-state  {as the root of the behaviour tree}
          behaviour-tree := run-simulator(checked-current-state)
    stop
```

```
- run-simulator(checked-current-state)

    loop
        persistence-times := calculate-persistence-time(checked-current-state)
        possible-next-states := determine-state-transition(checked-current-state)
        constraint-filtered-next-states := run-waltz-filter(possible-next-states)
        arrival-times := calculate-arrival-time(checked-current-state,
                                                possible-next-states)
        locally-filtered-next-states := run-temporal-filter(persistence-times,
                                                arrival-times,
                                                constraint-filtered-next-states)
        possible-behaviour-tree := run-global-interpreter(
                                                locally-filtered-next-states)
        globally-filtered-next-states := run-global-filter(possible-behaviour-tree,
                                                globally-filtered-behaviour-tree)
        if  resource-limit = true  or  globally-filtered-next-states = null
            return  globally-filtered-behaviour-tree  {the final behaviour tree}
            exit
        else  for  globally-filtered-next-state  in  globally-filtered-next-states
                   do : checked-current-state := globally-filtered-next-state
        nil
    end-of-loop


- calculate-persistence-time(checked-current-state)

        {performing calculation on persistence times
         with respect to the corresponding computational method}
        arrival-times-recorded := get-recorded-arrival-times()
        {performing computation on the absolute time indices}
        record associated-time-indices(checked-current-state)
        return persistence-times

- determine-state-transition(checked-current-state)

        {determining possible state transitions
         with respect to the possible state transition rules}
        return possible-next-states

- run-waltz-filter(possible-next-states)

        {performing constraint filtering with respect to both
         constraint consistency and pairwise consistency filtering criteria}
        return constraint-filtered-next-states
```

```
- calculate-arrival-time(checked-current-state,
                         possible-next-states)

      {performing calculation on arrival times
       with respect to the corresponding computational method}
      record arrival-times {meanwhile, clearing previous records}
      return arrival-times

- run-temporal-filter(persistence-times,
                      arrival-times,
                      constraint-filtered-next-states)

      {performing temporal filtering
       with respect to the corresponding criterion}
      return temporally-filtered-next-states {locally filtered next states}

- run-global-interpreter(locally-filtered-next-states)

      behaviour-tree-recorded := get-recorded-behaviour-tree()
      append(locally-filtered-next-states,
             behaviour-tree-recorded)
      return possible-behaviour-tree

- run-global-filter(possible-behaviour-tree,
                    globally-filtered-behaviour-tree)

      {performing global filtering by checking
       either no-change or repeating behaviours}
      record globally-filtered-behaviour-tree {as recorded behaviour tree}
      return globally-filtered-next-states
```

APPENDIX C

SOFTWARE ARCHITECTURAL DESIGN OF SID

This appendix presents the software architectural design of the *SID* system. First, the top level design is provided with respect to the functional structure of the system shown in figure 7.4. Then, the design for each of the four major sub-systems is given. Simple explanatory notes are presented in braces. The key intuition for developing such a software architecture is to provide a firm basis for implementing *SID*. For this purpose, the design is given in an as general as possible way. Further detailed specifications than what is provided herein can be found in [41].

C.1. Top Level Design

```
initiate-sid(normal-behaviour-model,
             modelling-dimensions,
             search-rules,
             default-focus-attention,
             default-focus-suspicion,
             diagnostic-control-rules,
             goals,
             resources)

   initiate-predictor(normal-behaviour-model,
                      default-focus-attention)
   initiate-discrepancy-generator(default-focus-attention)
   initiate-candidate-proposer(modelling-dimensions,
                               search-rules,
                               ranking-strategy,
                               default-focus-suspicion)
   initiate-diagnostic-strategist(diagnostic-control-rules,
                                  goals,
                                  resources)
```

```
{performing initialisations of SID such that,
 at the first call to the loop below, the following holds:
  model := normal-behaviour-model,
  focus-of-attention := default-focus-attention,
  focus-of-suspicion := default-focus-suspicion,
  candidates := null,
  next-action := null}


run-diagnosis()
loop
    observations := get-observations()
    predictions := run-predictor(model,
                                 observations,
                                 focus-of-attention,
                                 directly-interpreted-observations)
    discrepancies := run-discrepancy-generator(predictions,
                               directly-interpreted-observations,
                               focus-of-attention)
    if discrepancies ≠ null
       do : run-diagnostic-strategist(discrepancies,
                                      candidates,
                                           next-action,
                                           focus-of-attention,
                                           focus-of-suspicion)
          if next-action = stop
             return  candidates  {the final diagnoses}
             exit
          else nil
          candidates := run-candidate-proposer(discrepancies,
                                               focus-of-suspicion,
                                               next-action)
    else return  candidates        {the final diagnoses}
          exit
end-of-loop
```

C.2. Predictor

```
run-predictor(model,
              observations,
              focus-of-attention,
              directly-interpreted-observations)
```

```
    interpreted-observations := run-data-interpretation(observations,
                                                        focus-of-attention,
                                                        directly-interpreted-observations)
    predictions := run-prediction(model,
                                  focus-of-attention,
                                  interpreted-observations)
    return predictions

- run-data-interpretation(observations,
                          focus-of-attention,
                          directly-interpreted-observations)
    for observation in observations
        do : interpreting(observation, focus-of-attention)
    map(observations,
        directly-interpreted-observations) {mapping a onto [a, a, 0, 0]}
    return interpreted-observations

- run-prediction(model,
                 focus-of-attention,
                 interpreted-observations)
    predictions := call-fusim(model,
                              focus-of-attention,
                              interpreted-observations)
    return predictions
```

C.3. Discrepancy Generator

```
run-discrepancy-generator(predictions,
                          directly-interpreted-observations,
                          focus-of-attention)

    for directly-interpreted-observation in directly-interpreted-observations
        do : corresponding-prediction := get-corresponding(prediction,
                                                           directly-interpreted-observation)
        type(corresponding-prediction,
             directly-interpreted-observation)
        distance(corresponding-prediction,
                 directly-interpreted-observation)
    return types-and-distances  {the discrepancies}
```

C.4. Candidate Proposer

```
run-candidate-proposer(discrepancies,
                       focus-of-suspicion,
                       next-action)

    candidates := run-candidate-generation(discrepancies,
                                           focus-of-suspicion,
                                           next-action)
    ranked-candidates := run-candidate-ranking(candidates)
    return ranked-candidates

- run-candidate-generation(discrepancies,
                           focus-of-suspicion,
                           next-action)

    case next-action = nil
         {performing search with respect to the search-rules initiated}
         return candidates
    case next-action = new-start-points
         {performing search with start points changed}
         return candidates
    case next-action = stop
         return diagnoses      {the final diagnoses}
    else nil

- run-candidate-ranking(candidates)

    for candidate in candidates
        do : determine-rank(candidate)
        {ranking the candidate with respect to the ranking-strategy initiated}
    sort-according-to-rank(candidates)
    return ranked-candidates
```

C.5. Diagnostic Strategist

run-diagnostic-strategist(discrepancies,
candidates,
next-action,
focus-of-attention,
focus-of-suspicion)

set-of-performances := run-performance-evaluator(discrepancies,
candidates)
next-action := determine-next-diagnostic-action(set-of-performances)
focus-of-attention := determine-focus-of-attention(set-of-performances)
focus-of-suspicion := determine-focus-of-suspicion(set-of-performances)

- run-performance-evaluator(discrepancies, candidates)

{depending upon the performance criteria used
and the goals and resources initiated}
return set-of-performances

- determine-next-diagnostic-action(set-of-performances)

{inferring based on the diagnostic-control-rules initiated}
return next-action

- determine-focus-of-suspicion(set-of-performances)

{inferring based on the diagnostic-control-rules initiated}
return focus-of-attention

- determine-focus-of-attention(set-of-performances)

{inferring based on the diagnostic-control-rules initiated}
return focus-of-suspicion

REFERENCES

[1] J. F. Allen, Maintaining Knowledge about Temporal Intervals, *Communications of the ACM*, Vol. 26, No. 11, pp. 832-843, 1983.

[2] J. Azcarate, G. M. Coghill, J. Pascual, and J. Reyes, Requirements of Demonstrator B, *IR4.1, ESPRIT Project P5143*, 1991.

[3] D. Berleant and B. Kuipers, Combined Qualitative and Numerical Simulation with *Q3*, in: *Proceedings of the 4th International Conference on Qualitative Physics*, pp. 140-152, Lugano, 1990.

[4] D. G. Bobrow (Ed.), *Qualitative Reasoning about Physical Systems*, North-Holland, 1984.

[5] P. P. Bonissone, *The Problem of Linguistic Approximation in System Analysis*, Ph.D. Thesis, University of California, Berkeley, 1979.

[6] P. P. Bonissone, Summarizing and Propagating Uncertain Information with Triangular Norms, *International Journal of Approximate Reasoning*, pp. 71-101, 1987.

[7] P. P. Bonissone and K. S. Decker, Selecting Uncertainty Calculi and Granularity: An Experiment in Trading-off Precision and Complexity, in: L. N. Kanak and J. F. Lemmer (Eds.), *Uncertainty in Artificial Intelligence*, pp. 217-247, North-Holland, 1986.

[8] W. E. Boyce and R. C. diPrima, *Elementary Differential Equations*, John Wiley & Sons, 1977.

[9] A. L. Brown, Qualitative Knowledge, Causal Reasoning, and the Localization of Failures, *Technical Report AI-TR-362, MIT*, 1976.

[10] J. S. Brown, R. R. Burton, and J. de Kleer, Pedagogical, Natural Language and Knowledge Engineering Techniques in SOPHIE I, II and III, in: D. Sleeman and J. S. Brown (Eds.), *Intelligent Tutorial Systems*, Academic Press, pp. 227-282, 1982.

[11] W. F. Clocksin and C. S. Mellish, *Programming in Prolog*, Springer-Verlag, 1981.

[12] B. D. D'Ambrosio, *Qualitative Process Theory Using Linguistic Variables*, Ph.D. Thesis, University of California, Berkeley, 1986.

[13] B. D. D'Ambrosio, Extending the Mathematics in Qualitative Process, in: *Proceedings of the 6th National Conference on Artificial Intelligence*, pp. 595-599, 1987.

[14] E. Davis, Constraint Propagation with Interval Labels, *Artificial Intelligence*, **32**, pp. 281-331, 1987.

[15] E. Davis, Order of Magnitude Reasoning in Qualitative Differential Equations, in: D. Weld and J. de Kleer (Eds.), *Readings in Qualitative Reasoning about Physical Systems*, Morgan Kaufmann, pp. 422-434, 1990.

[16] R. Davis, Expert Systems: Where Are We? and Where Do We Go from Here? *AI Magazine*, **1**, 1982.

[17] R. Davis, Diagnostic Reasoning Based on Structure and Behaviour, *Artificial Intelligence*, **24**, pp. 347-410, 1984.

[18] R. Davis and W. Hamscher, Model-Based Reasoning: Troubleshooting, in: *Exploring Artificial Intelligence*, Morgan Kaufmann, pp. 297-346, 1988.

[19] R. Davis, H. Shrobe, W. Hamscher, K. Wieckert, M. Shirley, and S. Polit, Diagnosis Based on Structure and Function, in: *Proceedings of the Second National Conference on Artificial Intelligence*, pp. 137-142, 1982.

[20] J. de Kleer, Local Methods of Localizing Faults in Electronic Circuits, *AI Memo 394, Artificial Intelligence Laboratory, MIT*, 1976.

[21] J. de Kleer, An Assumption-Based TMS, *Artificial Intelligence*, **28**, pp. 127-162, 1986.

[22] J. de Kleer, Extending the ATMS, *Artificial Intelligence*, **28**, pp. 163-196, 1986.

[23] J. de Kleer, Using Crude Probability Estimates to Guide Diagnosis, *Artificial Intelligence*, **45**, pp. 381-391, 1990.

[24] J. de Kleer and D. G. Bobrow, Qualitative Reasoning with High-Order Derivatives, in: *Proceedings of the 3rd National Conference on Artificial Intelligence*, pp. 86-91, 1984.

[25] J. de Kleer and J. S. Brown, A Qualitative Physics Based on Confluences, *Artificial Intelligence*, **24**, pp. 7-83, 1984.

[26] J. de Kleer and B. C. Williams, Diagnosing Multiple Faults, *Artificial Intelligence*, **32**, pp. 97-130, 1987.

[27] J. de Kleer and B. C. Williams, Diagnosis with Behavioural Modes, in: *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, Vol. 2, pp. 1324-1330, 1989.

[28] H. Dishkant, About Membership-Function Estimation, *Fuzzy Sets and Systems*, Vol. 5, No. 2, pp. 141-148, 1981.

[29] R. C. Dorf, *Modern Control Systems*, Addison-Wesley, 1989.

[30] J. Doyle and E. P. Sacks, Stochastic Analysis of Qualitative Dynamics, in: *Proceedings of the 3rd International Workshop on Qualitative Physics,* California, 1989.

[31] D. Dubois and H. Prade, *Fuzzy Sets and Systems: Theory and Applications,* Academic Press, 1980.

[32] D. Dubois & H. Prade, Order-of-Magnitude Reasoning with Fuzzy Relations, in: *Proceedings of IFAC Symposium on Advanced Information Processing in Automatic Control*, Nancy, 1989.

[33] D. Dvorak and B. Kuipers, Model-Based Monitoring of Dynamic Systems, in: *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, Vol. 2, pp. 1338-1343, 1989.

[34] D. Dvorak and B. Kuipers, Process Monitoring and Diagnosis: A Model-Based Approach, *IEEE EXPERT*, pp. 67-74, 1991.

[35] K. D. Forbus, Qualitative Process Theory, *Artificial Intelligence*, **24**, pp. 85-168, 1984.

[36] K. D. Forbus, Interpreting Measurements of Physical Systems, in: *Proceedings of the 5th National Conference on Artificial Intelligence*, pp. 113-117, 1986.

[37] K. D. Forbus, Qualitative Physics: Past, Present, and Future, in: *Exploring Artificial Intelligence*, Morgan Kaufmann, pp. 239-296, 1988.

[38] P. Fouche and B. Kuipers, Introducing Energy into Qualitative Simulation, in *Proceedings of the 1st European Conference on Qualitative Reasoning about Physical Systems*, pp. 28-44, 1991.

[39] H. Freitag, A Generic Measurement Proposer, in: *Proceedings of International Workshop on Expert Systems in Engineering*, 1990.

[40] H. Freitag, R. R. Leitch, Q. Shen, P. Struss, and G. Tornielli, Model-Based Diagnostic System Reference Architecture, *Deliverable Report D1.1*, *ESPRIT Project P5143*, 34pp, 1990.

[41] H. Freitag, Q. Shen, G. Tornielli, and M. Chantler, Specification of *ARTIST/I*, *Deliverable Report D2.1* (in preparation), *ESPRIT Project P5143*, 1991.

[42] M. R. Genesereth, The Use of Design Descriptions in Automated Diagnosis, *Artificial Intelligence*, **24**, pp. 411-436, 1984.

[43] R. Greiner, B. A. Smith, and R. W. Wilkerson, A Correction to the Algorithm in Reiter's Theory of Diagnosis, *Artificial Intelligence*, **41**, pp. 79-88, 1989.

[44] T. Guckenbiehl, Representing Structure and Function in the Framework of MODISC, *Technical Report FhG-IITB*, Karlsruhe, 1989.

[45] G. R. Guo and Q. Shen, Fuzzy Processing of Radar Signals, *Proceedings of International Conference on Radar*, Naking, 1986.

[46] G. R. Guo and Q. Shen, Fuzzy Algorithm for Error-Correcting Estimation of Radar Signals, *Electronics Letters*, Vol. 26, No. 16, pp. 1321-1324, 1990.

[47] R. Isermann, Process Fault Detection Based on Modelling and Estimation Methods - A Survey, *Automatica*, Vol. 20, No. 4, pp. 387-404, 1984.

[48] Y. Iwasaki, Causal Ordering in a Mixed Structure, in: *Proceedings of the 7th National Conference on Artificial Intelligence*, Vol. 1, pp. 313-318, 1988.

[49] A. Kandel, *Fuzzy Mathematical Techniques with Applications*, Addison-Wesley, 1987.

[50] B. J. Kuipers, Commonsense Reasoning about Causality: Deriving Behaviour from Structure, *Artificial Intelligence*, **24**, pp. 169-203, 1984.

[51] B. J. Kuipers, The Limits of Qualitative Simulation, in: *Proceedings of 9th International Joint Conference on Artificial Intelligence*, Vol. 1, pp. 128-136, Los Angeles, 1985.

[52] B. J. Kuipers, Qualitative Simulation, *Artificial Intelligence*, **29**, pp. 289-338, 1986.

[53] B. J. Kuipers and D. Berleant, Using Incomplete Quantitative Knowledge in Qualitative Reasoning, in: *Proceedings of the 7th National Conference on Artificial Intelligence*, pp. 324-329, 1988.

[54] B. J. Kuipers and C. Chiu, Taming Intractable Branching in Qualitative Simulation, in: *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, pp. 1079-1085, 1987.

[55] Y. Landau, *Adaptive Control: The Model Reference Approach*, Marcel Dekker, 1979.

[56] E. T. Lee, An Application of Fuzzy Sets to the Classification of Geometric Figures and Chromosome Images, *Information Sciences*, **10**, pp. 95-114, 1976.

[57] W. W. Lee and B. J. Kuipers, Non-Intersection of Trajectories in Qualitative Phase Space: A Global Constraint for Qualitative Simulation, in: *Proceedings of the 7th National Conference on Artificial Intelligence*, pp. 286-290, 1988.

[58] R. R. Leitch, The Modelling of Complex Dynamic Systems, *IEE Proceedings, Part D*, Vol. 134, No 3, pp. 245-250, 1987.

[59] R. R. Leitch, A Review of the Approaches to the Qualitative Modelling of Complex Systems, in: D. A. Linkens (Ed.), *Trends in Information Technology*, Peter Peregrinus, 1989.

[60] R. R. Leitch, H. Freitag, Q. Shen, P. Struss, and G. Tornielli, *ARTIST*: A Methodological Approach to Specifying Model-Based Diagnostic Systems, in: *Proceedings of European Conference on Industrial Applications of Knowledge-Based Diagnosis*, pp. 183-208, Milan, 1991.

[61] R. R. Leitch and M. Gallanti, Task Classification for Knowledge Based Systems in Industrial Automation, to appear in: *IEEE Transactions on Systems, Man and Cybernetics,* 1991.

[62] R. R. Leitch, P. V. S. Ponnapalli, J. A. Sime, and A. F. Slater, Intelligent Training Systems: A Status Report on *ESPRIT Project P2615*, in: *Proceedings of IEE Colloquium on Intelligent Tutoring Systems*, 1990.

[63] R. R. Leitch and H. C. Quek, A Behaviour Classification for Integrated Process Supervision, in : *Proceedings of the 3rd International Conference on Control*, Vo. 1, pp. 127-133, Edinburgh, 1991.

[64] R. R. Leitch and Q. Shen, Model Dimensions within Fuzzy Qualitative Simulation, in: *Proceedings of the 13th IMACS World Congress on Computation and Applied Mathematics*, Vol. 3, pp. 1212-1213, Dublin, 1991.

[65] R. R. Leitch and Q. Shen, Finding Faults with Model Based Diagnosis, in: *Proceedings of the 2nd International Workshop on Principles of Diagnosis*, pp. 121-134, Milan, 1991.

[66] R. R. Leitch and A. Stefanini, Task Dependent Tools for Intelligence Automation, *International Journal of Artificial Intelligence in Engineering*, pp. 126-143, 1989.

[67] R. R. Leitch, M. E. Wiegand, and H. C. Quek, Coping With Complexity in Physical System Modeling, *Artificial Intelligence Communications*, Vol. 3, No. 2, pp. 48-57, 1990.

[68] D. A. Linkens and M. F. Abbod, Self-Organizing Fuzzy Logic Control for Real-Time Processes, in *Proceedings of the 3rd International Conference on Control*, Vol. 2, pp. 971-976, 1991.

[69] D. A. Linkens and S. B. Hasnain, Self-Organizing Fuzzy Logic Control and its Application to Muscle Relaxant Anaesthesia, *IEE Proceedings, Part D*, Vol. 138, No. 3, pp. 274-284, 1991.

[70] L. Loomis, *Calculus*, Addison-Wesley, 1977.

[71] D. G. Luenberger, An Introduction to Observers, *IEEE Transactions on Automatic Control*, **16**, pp. 596-602, 1971.

[72] E. H. Mamdani, Advances in the Linguistic Synthesis of Fuzzy Controllers, *International Journal of Man-Machine Studies*, **8**, pp. 669-678, 1976.

[73] M. L. Mavrovouniotis & G. Stephanopoulos, Reasoning with Orders of Magnitude and Approximate Relations, in: *Proceedings of the 6th National Conference on Artificial Intelligence*, Seattle, pp. 626-630, 1987.

[74] M. L. Mavrovouniotis & G. Stephanopoulos, Order-of-Magnitude Reasoning with *O[M]*, *International Journal of Artificial Intelligence in Engineering*, Vol. 4, No. 3, pp. 106-114, 1989.

[75] M. Mizumoto and K. Tanaka, Some Properties of Fuzzy Numbers, in: M. M. Gupta, R. K. Ragade, and R. R. Yager (Eds.), *Advances in Fuzzy Set Theory and Applications*, pp. 153-164, North-Holland, 1979.

[76] R. E. Moore, *Interval Analysis*, Prentice-Hall, 1966.

[77] R. Patton, P. Frank, and R. Clark (Eds.), *Fault Diagnosis in Dynamic Systems: Theory and Applications*, Prentice Hall International Ltd, 1989.

[78] D. A. Pearce, The Induction of Fault Diagnosis Systems from Qualitative Models, in: *Proceedings of the 7th National Conference on Artificial Intelligence*, pp 353-357, 1988.

[79] J. R. Quinlan, Induction of Decision Trees, *Machine Learning*, **1**, pp. 81-106, 1986.

[80] O. Raiman, Order of Magnitude Reasoning, in: *Proceedings of the 5th National Conference on Artificial Intelligence*, pp. 100-104, 1986.

[81] J. A. Reggia, D. S. Nau, and P. Y. Wang, Diagnostic Expert Systems Based on a Set Covering Model, *International Journal of Man-Machine Studies*, **19**, pp. 437-460, 1983.

[82] R. Reiter, A Theory of Diagnosis from First Principle, *Artificial Intelligence*, **32**, pp. 57-95, 1987.

[83] A. Robert, *Nonstandard Analysis*, John Wiley & Sons, 1988.

[84] A. Robinson, *Non-Standard Analysis*, North-Holland, 1966.

[85] E. H. Ruspini, Recent Developments in Fuzzy Cluster Analysis and its Application, in: *Proceedings of International Congress on Applied System Research and Cybernetics*, 1980.

[86] E. Sacks, Hierarchical Reasoning about Inequalities, in: *Proceedings of the 6th National Conference on Artificial Intelligence*, pp. 649-654, 1987.

[87] G.S. Sebestyen, *Decision-Making Process in Pattern Recognition*, The MacMillan Company, 1962.

[88] Q. Shen, Fuzzy Image Smoothing, in: *Proceedings of the 10th International Conference on Pattern Recognition*, New Jersey, Vol. 2, pp. 74-78, 1990.

[89] Q. Shen, Fuzzy Intraframe Smoothing of a Noisy Image, *Electronics Letters*, Vol. 26, No. 13, pp. 908-910, 1990.

[90] Q. Shen and R. R. Leitch, A Semi-Quantitative Extension to Qualitative Simulation, in: *Proceedings of the 10th International Conference on Expert Systems and their Applications*, EC2, pp. 223-234, Avignon, 1990.

[91] Q. Shen and R. R. Leitch, Integrating Commonsense with Qualitative Simulation by the Use of Fuzzy Sets, in: *Proceedings of the 4th International Workshop on Qualitative Physics*, Lugano, pp. 220-232, 1990.

[92] Q. Shen and R. R. Leitch, Fuzzy Qualitative Simulation, submitted for journal publication, 1990.

[93] Q. Shen and R. R. Leitch, Extending the Quantity Space in Qualitative Reasoning, in: *Proceedings of the 1st European Workshop on Qualitative Reasoning about Physical Systems*, Milan, pp. 1-14, 1991.

[94] Q. Shen and R. R. Leitch, Diagnosing Continuous Dynamic Systems Using Qualitative Simulation, in: *Proceedings of the 3rd International Conference on Control*, Vol. 2, pp. 1000-1006, Edinburgh, 1991.

[95] Q. Shen and R. R. Leitch, Synchronised Qualitative Simulation in Diagnosis, in: *Proceedings of the 5th International Workshop on Qualitative Reasoning about Physical Systems*, pp. 171-185, Austin, 1991.

[96] Q. Shen and R. R. Leitch, Combining Qualitative Simulation and Fuzzy Sets, in: B. Faltings and P. Struss (Eds.), *Recent Advances in Qualitative Physics*, MIT Press, 1991.

[97] Q. Shen and R. R. Leitch, On Extending the Quantity Space in Qualitative Reasoning, submitted for journal publication, 1991.

[98] Q. Shen and R. R. Leitch, Extending Qualitative Simulation by the Use of Fuzzy Sets, submitted for conference publication, 1991.

[99] R. G. Simmons, "Common-sense" Arithmetic Reasoning, in: *Proceedings of the 5th National Conference on Artificial Intelligence,* pp. 118-124, 1986.

[100] P. Struss, Multiple Representation of Structure and Function, in: J. S. Gero (Ed.), *Expert Systems in Computer-Aided Design*, North-Holland, 1987.

[101] P. Struss, Mathematical Aspects of Qualitative Reasoning, *International Journal for Artificial Intelligence in Engineering,* Vol. 3, No. 3, pp. 156-169, 1988.

[102] P. Struss, Global Filters for Qualitative Behaviours, in: *Proceedings of the 7th National Conference on Artificial Intelligence,* pp. 275-279, 1988.

[103] P. Struss, A Framework for Model-Based Diagnosis, *Siemens Technical Report INF2 ARM-10-88*, 1988.

[104] P. Struss, Extensions to ATMS-Based Diagnosis, in: J. S. Gero (Ed.), *Artificial Intelligence in Engineering: Diagnosis and Learning*, 1988.

[105] P. Struss, Diagnosis as a Process, in: *Proceedings of International Workshop on Model-Based Diagnosis*, 1989.

[106] P. Struss, Model-Based Diagnosis - Progress and Problems, in: *Proceedings of the 3rd International GI Congress on Knowledge-Based Systems*, Munich, 1989.

[107] P. Struss, Problems of Interval-Based Qualitative Reasoning, in: D. Weld and J. de Kleer (Eds.), *Readings in Qualitative Reasoning about Physical Systems*, pp. 288-305, Morgan Kaufmann, 1990.

[108] P. Struss, A Theory of Model Simplification and Abstraction for Diagnosis, *Proceedings of the 5th International Workshop on Qualitative Reasoning about Physical Systems*, pp. 25-57, 1991.

[109] P. Struss and O. Dressler, Physical Negation - Integrating Fault Models into General Diagnostic Engine, in: *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, Vol. 2, pp. 1318-1323, 1989.

[110] G. J. Sussmen and G. L. Steele, Constraints - a Language for Expressing Almost-Hierarchical Descriptions, *Artificial Intelligence*, **14**, 1980.

[111] M. Vitek, Fuzzy Information and Fuzzy Time, in: E. Sanchez (Ed.), *Fuzzy Information, Knowledge Representation and Decision Analysis*, pp. 159-162, Pergamon Press, 1984.

[112] D. Waltz, Understanding Line Drawings of Scenes with Shadows, in: P. Winston (Ed.), *The Psychology of Computer Vision*, McGraw-Hill, 1975.

[113] D. Weld, Exaggeration, in: *Proceedings of the 7th National Conference on Artificial Intelligence*, pp. 291-295, 1988.

[114] D. Weld, *Theories of Comparative Analysis*, MIT Press, 1990.

[115] D. Weld, Automated Model Switching, in: *Proceedings of the 3rd International Workshop on Qualitative Physics*, 1989.

[116] D. Weld, Exaggeration, *Artificial Intelligence*, **43**, pp. 311-368, 1990.

[117] D. Weld and J. de Kleer, *Readings in Qualitative Reasoning about Physical Systems*, Morgan Kaufmann, 1990.

[118] P. E. Wellstead, *Introduction to Physical Systems Modelling*, Academic Press, 1979.

[119] M. E. Wiegand and R. R. Leitch, A "Predictive Engine" for the Qualitative Simulation of Continuous Dynamic Systems, in: *Proceedings of the 3rd International Workshop on Qualitative Physics*, 1989.

[120] M. E. Wiegand and R. R. Leitch, A Predictive Engine for the Qualitative Simulation of Dynamic Systems, in: *Proceedings of the 4th International Conference on Applications of Artificial Intelligence in Engineering*, pp. 141-150, 1989.

[121] B. C. Williams, Qualitative Analysis of MOS Circuits, *Artificial Intelligence*, **24**, pp. 281-346, 1984.

[122] B. C. Williams, The Use of Continuity in a Qualitative Physics, in: *Proceedings of the National Conference on Artificial Intelligence*, pp. 350-354, 1984.

[123] B. C. Williams, Doing Time: Putting Qualitative Reasoning on Firmer Ground, in: *Proceedings of the 5th National Conference on Artificial Intelligence*, pp. 105-112, 1986.

[124] B. C. Williams, *MINIMA*: A Symbolic Approach to Qualitative Algebraic Reasoning, in: *Proceedings of the 7th National Conference on Artificial Intelligence*, pp. 264-269, 1988.

[125] R. R. Yager, S. Ovchinnikov, R. M. Tong, and H. T. Nguyen (Eds.), *Fuzzy Sets and Applications: Selected Papers by L. A. Zadeh*, John Wiley & Sons, 1987.

[126] L. A. Zadeh, Fuzzy Sets, *Information and Control*, **8**, pp. 338-353, 1965.

[127] L. A. Zadeh, Similarity Relations and Fuzzy Orderings, *Information Sciences*, **3**, pp. 177-200, 1971.

[128] L. A. Zadeh, Outline of a New Approach to the Analysis of Complex Systems and Decision Processes, *IEEE Transactions on Systems, Man and Cybernetics*, **3**, pp. 28-44, 1973.

[129] L. A. Zadeh, The Concept of a Linguistic Variable and its Application to Approximate Reasoning, Parts 1 and 2, *Information Sciences*, **8**, pp. 199-249 and 301-357, 1975.

[130] L. A. Zadeh, The Concept of a Linguistic Variable and its Application to Approximate Reasoning, Part 3, *Information Sciences,* **9**, pp. 43-80, 1976.

[131] L. A. Zadeh, A Theory of Approximate Reasoning, in: J. Hayes, D. Michie, and L. I. Mikulich (Eds.), *Machine Intelligence,* Vol. 9, Halstead Press, pp. 149-194, 1979.

[132] L. A. Zadeh, The Role of Fuzzy Logic in the Management of Uncertainty in Expert Systems, *Fuzzy Sets and Systems,* **11**, pp. 199-227, 1983.

[133] L. A. Zadeh, A Theory of Commonsense Knowledge, in: H. J. Skala, S. Termini, and E. Trillas (Eds.), *Aspects of Vagueness*, pp. 257-296, 1984.