Understanding Natural Language about Multiple Eventualities and Continuous Eventualities

Sheila Rock

Ph.D.
University of Edinburgh
1996

Understanding Natural Language about Multiple Eventualities and Continuous Eventualities

Sheila Rock

Ph.D.
University of Edinburgh
1996

Declaration

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

Sheila Rock
Edinburgh
January 16, 1997

Abstract

The main task that this thesis deals with is the provision of a comprehensive analysis covering a meaningful subset of English and developing a computational implementation that is able to show understanding of this language subset, in part via limited visualisation.

There is a well accepted analogy that says that eventualities exist in time, in ways that are similar to how objects exist in space. This analogy is used as a framework to investigate in detail those activities that are the eventuality analogue of plural and mass objects—multiple instances of an activity, or continuous occurrence of an activity respectively. These are called extended activities.

We examine the ways in which natural language is used to describe these kinds of activities, and discuss ways in which the meanings of such language can be represented. We concentrate on language that is in the form of instructions, and discuss the special relationship between instructions and activities.

Using the idea that some of our understanding of language comes from the context within which the language is being understood, we identify those parts of language about extended activities that are independent of context and indicate the places where context would play a part. Focusing on the context-independent part, the development of a grammar that can be used in an understanding system demonstrates that it is feasible to interpret important aspects of such language computationally. Further, the system includes a semi-graphical visualisation component that depicts in space the internal structure of the extended activity in time.

The work in this thesis relies on the notion that language about extended activities is playing a role analogous to that of object quantification. That is, instead of the more common view that such language is playing the role of event modification, we take the view that it plays the role of event quantification. This notion has been introduced by Moltmann, and taking this approach allows the identification and representation of meanings that would otherwise be omitted. Further, incorporating this into the computational framework is feasible, and an established approach to object quantification is used to implement event quantification.

Acknowledgements

The biggest debt of gratitude I owe is to my supervisor, Chris Mellish. I have received academic guidance, insights, encouragement and support from him without which I am sure I could never have completed this thesis. My second supervisors—first Robert Dale and later Elisabet Engdahl—have also been supportive through difficult times. Elisabet's approach to work and keeping going is what kept me going; Robert's approach to work and keeping going is somewhat scary, but his approach to discovery and his writing style have always been an inspiration.

Thanks too to Lex Holt, for constant support and friendship, and for talking to me about and helping me work through various topics that have been relevant to this research. And thanks—I think—for the LaTeX and Prolog encouragement and advice.

I have during the course of my research had the good fortune to receive advice, direction and discussion, at various times, from Marc Moens, Bonnie Webber, Martin Mellor, Khee Yin How, Sheila Glasbey. They probably don't remember this, but I do. Thanks also to Diana Bental, for reading drafts, and for maintaining a sense of humour; Neil Caithness for supporting my plan to do this in the first place, and continuing that support even though it has changed our lives completely; Ian Sanders for reading drafts, and for looking after my cats when I abandoned them to pursue this research.

I am also indebted to Richard Cooper and Steve Draper, and the Computing Science Department at Glasgow University, for giving me a home in which to finish writing.

I acknowledge with gratitude the partial funding I received from the South African Foundation for Research and Development, and the emergency award I received from the British Federation of Women Graduates.

Thanks to my examiners Marc Moens and Ian Pratt for valuable comments and useful suggestions—and for clearly reading this thesis so thoroughly!

Finally, I owe so much to my mother and my father—although they are no longer with me, their sentiment that there is nothing that a woman cannot do always will be.

Glossary

The terms described in this glossary are with respect to their particular use in this thesis.

ACTIVITY ENTITY This term is used when referring to a particular instance or instances of activity.

ACTIVITY An activity is an eventuality that is NON-STATIVE; it is something that happens, rather than a state of being.

AGENT The protagonist of execution of an instruction.

ASPECT For some authors this refers to the eventuality category that is suggested by the language describing a situation; for others it refers to the speaker's viewpoint when describing a situation.

ATELIC An atelic eventuality is one without a well defined ending point.

ATOMIC An activity is described as atomic if, at the level of granularity that is being employed, it consists of no sub-activity.

BNF An abbreviation of Backus-Naur Form.

BOUNDED – JACKENDOFF Jackendoff's[22] use of this term is to indicate individual objects or completed events.

BOUNDED Zemach's [57] use of this term, and the one that I adopt, refers to a notion of whether a temporal or spatial entity has parts.

CARDINALITY Applied to multiple instances of activity, this refers to the number of instances.

COERCION A term introduced by Moens & Steedman[28] that refers to the notion that eventualities that are seen as being from one category may be coerced to being seen as from another, depending on the context of use.

COMPLEX The term complex is used in this thesis to cover activity structures where the underlying sub-activities do not necessarily have any similarity.

COMPOUND A compound activity is a structured activity that consists of two or more non-similar activities. These activities are assumed to occur consecutively.

CONCURRENT A concurrent activity consists of two extended activities that both occur at the same time.

CONJUNCTIVE An extended activity is described as conjunctive if its extent is expressed in two different ways, such as '*for two minutes or until it thickens*'.

CONTENT This is used to refer to the 'substance' of an extended activity; the actual basic core activity is the content of a structured activity. In the structured activity described by '*Stir the soup for ten minutes*', the content is $\ll$*stir the soup*$\gg$.

CONTEXT This is used with regard to instruction execution, and refers to the particular set of circumstances under which an instruction is executed. The context includes all the objects that are involved in carrying out the instruction. Unless otherwise indicated, context as used in this thesis refers to the execution context—the set of actual circumstances—in which an activity takes place. When linguistic context is intended, the phrase linguistic context is used explicitly.

CONTIGUOUS This term is used by Zemach[57] to refer to activity that is 'unbroken' or 'uninterrupted'; its use is similar to that of continuous in describing activity.

CONTINUANTS IN SPACE Zemach's [57] term for entities that are bounded in time and continuous in space. These are what we conventionally see as events or processes.

CONTINUANTS IN TIME Zemach's [57] term for entities that are bounded in space and continuous in time. These are what we conventionally see as objects.

CONTINUOUS–ZEMACH Zemach's[57] use of the term continuous refers to an entity that does not have parts, with respect to the dimension—temporal or spatial—that is being considered.

CONTINUOUS Applied to an eventuality, this means that the activity has the property of extending in time, and within that time, from our perspective, there are no times when that activity is not occurring; activity of this sort is usually atelic. Zemach [57] calls such eventualities CONTIGUOUS; there is no time between any parts of such an eventuality that do not contain part of that eventuality.

CORE ACTIVITY When we have an extended activity, the core activity is the most basic activity described by the language, without any of the structuring.

COUNT-LIKE This is used to refer to activity that is the eventuality analogue of count in objects; it describes activity that is discrete.

DISCRETE I use the term discrete to refer to a core activity entity that is analogous to a singular object; it is an activity that is telic.

DISTINCT Activities are distinct if they are either temporally distinct or spatially distinct.

ELLIPSIS This refers to a syntactic item that has been omitted from a sentence; usually it is expected that the elided item is assumed to be understood.

EPISODE An episode is essentially what Hwang & Shubert [21] would refer to an activity as.

EVENTUALITY An occurrence in the world; a possible 'happening'. In the terminology of Zemach [57], this is a CONTINUANT IN SPACE.

EXECUTABLE SEMANTICS Particularly for the semantics of instructions, this refers to the semantics once execution context has been incorporated.

EXECUTION This refers to the actual carrying out of the activity described by an instruction, or simply the carrying out of an activity.

EXTENDED ACTIVITY An extended activity is an activity entity that is made up of sub-activities; it is structured and its internal structure conforms to that described by the term extended.

EXTENDED I use the term extended strictly to refer to the general case of activity that is complex in the very particular way that is of interest to my thesis. That is, it is a term that covers continuously executed activity, or repeated activity instances, or multiple instances of activity. It is a word that covers the eventuality analogue of plural and mass.

EXTENT when applied to extended activity, refers to the amount of such activity. For example, in '*Stir the soup for ten minutes*', the extent here is ≪*ten minutes worth*≫.

FREQUENCY Applied to multiple instances of activity, this refers to how often they occur.

GRANULARITY This refers to the level of detail at which an entity is being considered.

GROUPING This refers to the way in which entities are seen, conceptually, with regard to their structure and sub-structure.

HAPPENING In this thesis, the term happening (nominal) refers to an eventuality that is an activity with particular properties. These are that the activity—with respect to the particular level of granularity at which it is being considered—does not itself contain any structuring, although it may form part of another structuring.

INSISTED This is a term that refers to the way language is used to indicate extended activity structure. It is used when the extended structure is explicitly stated in the language, such as '*Do that twice*'.

INSTANTANEOUS Something happening within a moment; without delay. This is used to refer to an eventuality that happens in a time smaller than the granularity at which we are working. It is important to be clear that this does not refer to an eventuality taking *no* time to happen; it means that it happens in an extremely small amount of time.

INSTRUCTION An unlocated description of activity; a template or recipe for activity. An instruction is a statement about the activity (or activities) that an agent must perform, usually in order to bring about required or desired changes in the state of the world.

INTERNAL STRUCTURE This refers to the nature of the sub-structure of a complex or extended activity.

ITERABLE A verb is classified as iterable if it is possible for it to be used in sentences where it describes a continuous activity ('*Stir the soup for ten minutes*') as well as in sentences where it describes a discrete activity ('*Stir the soup five times*').

LF This is an abbreviation of Logical Form.

MASS-LIKE This is used to refer to activity that is the eventuality analogue of mass in objects.

MASS The term mass is used in this thesis to refer to some continuously occurring similar activity.

MEASURE ADVERBIAL This is an adverbial—which may be a prepositional phrase, or and adverbial phrase, or even a sentenceBAR—that describes an *amount* of activity. Examples are '*throughout the world*', '*for half an hour*', '*twice*'.

MEREOLOGY This is concern with the relationship of parts and wholes.

MULTIPLE INSTANCES This term is used to describe eventualities in a way analogous to that of using plural in describing objects. It describes the notion of there being more than one occurrence of some activity entity.

NEUTRAL SEMANTICS This is the semantics that can be extracted from the language, before reference to the context in which the language is to be applied. It is 'neutral to the context'.

NON-CONTINUANTS Zemach's [57] term for entities that are bounded in both time and space—he gives black holes as an example.

NON-STATIVE An eventuality that is not a state of being, but rather is something that happens. Activities are non-statives.

OBJECT A tangible entity in the world; a 'thing'. In the terminology of Zemach [57], this is a CONTINUANT IN TIME.

PATH EQUATIONS These determine potential unification of feature structures.

PATH This defines a path in a feature structure, and gives access to the value of that feature.

PHENOMENON This term is used in its philosophical sense – to denote an object of experience (in this case, and eventuality) as it appears, as distinguished from it as a thing-in-itself.

PROTRACTED I use the word PROTRACTED to mean continuous (contiguous, according to Zemach actually) over a period – mainly in the way of saying that "he slept for an hour" is a protracted event. So, it is used in a similar way to the word 'mass-like'.

PURE CONTINUANTS Zemach's [57] term for entities that are continuous—not bounded—in both time and space—he gives types and masses as examples.

QUANTIFIED TERM This refers to the representation of an entity that is seen as describing a quantity; in this thesis, quantified terms can come from describing the quantity of an eventuality, or from describing the quantity of an object.

REPEATED Applied to an eventuality, this means that the eventuality occurs more than once, and these occurrences follow each other in time.

SET The term set is used in this thesis to refer to a collection of similar activities.

SIMILAR Activities are similar if the language and perspective indicate that we should view them in the same way.

SITUATION Some authors [56, 45, 53, 18] use the term SITUATION to describe what are called eventualities, and distinguish between situation types. However, this term—situation—has also been used [50] to refer the same thing that some authors call a state.

SPATIALLY DISTINCT Two instances of an activity are spatially distinct if they occur, or are perceived to occur, in different spatial locations. For example, in '*A telephone rang in each room of the hotel*', all the ≪*telephone ringing*≫ activities can be perceived as spatially distinct.

STATE This refers to an eventuality that is not an activity, but is a 'state-of-being'.

STRUCTURE This refers to the way sub-parts of an activity are connected with each other and the overall activity.

STRUCTURINGS The term structurings is used to refer to ways in which sub-activities form an overall activity.

SUGGESTED Referring to the way language is used to indicate extended activity structure, this describes extended structure that is only implied by the language, such as '*Peel all of the oranges*'.

TELIC An telic eventuality is one that has a well defined ending point.

TEMPORAL EXTENT This refers to the size in time that an activity occupies; how long it occurs for.

TEMPORAL LOCATION This refers to where in time an activity is located; when it occurs.

TEMPORAL MEASURE ADVERBIAL This is a measure adverbial describing a temporal entity. For example, while '*throughout the world*' is a *spatial* measure adverbial, '*throughout the day*' and '*for ten minutes*' are *temporal* measure adverbials.

TEMPORALLY DISTINCT Two instances of activity are temporally distinct if there is, from our perspective, any amount of time

TOPOLOGY This is concern with structure and sub-structure, and the way in which sub-parts are arranged with respect to each other.

UNCOMMITTED I use this term when referring to a happening whose type has not been set, because it comes from a verb that is neither discrete nor continuous, but instead is iterable. The happening remains uncommitted until is is absorbed into a structuring that defines its type.

Notational conventions

In this thesis, the following notational conventions are used:

The typeface TERM is for anything that is a term that is specially meaningful in the thesis; whose precise meaning is important. Definitions of all of these are found in the Glossary.

When natural language examples appear in text, they will be as follows: '*This is a natural language example*'. However, in numbered examples, they appear in normal typeface.

Emphasis or stress is indicated *in this manner*.

Semantic glosses appear in the form ≪*semantic gloss*≫.

This is used for the converse of semantic glosses — ≫*language*≪ that expresses some semantic entity.

In feature structure descriptions,
features are noted thus: [**feature=**],
and feature values thus: [=**value**].

In natural language examples:
A sentence that is grammatical, but is semantically odd in the given context is marked #.
A sentence that is ungrammatical is marked *.
A sentence whose well-formedness is questionable is marked ?.

In semantic examples, anything that is not well-formed is marked **.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis investigates the problem of analysing activity structure described by natural language. Natural language utterances about EVENTUALITIES[1] often include language that describes REPEATED or CONTINUOUS execution of some ACTIVITY. An eventuality is taken here to be the same as that noted by Bach[4], and others since: something that exists in time. An activity is then a kind of eventuality; loosely, the set of activities is a subset of the set of eventualities. In particular an activity is an eventuality that is NON-STATIVE; something happening rather than being in some state. An activity can occur more than once, or the same activity can occur continuously over a period of time.

Analysing activity STRUCTURE then involves establishing what kind of basic activity is being described, and determining the repeated or continuous execution of the basic activity that makes up the overall activity.

Consider the following examples:

(1.1) e_1: Kim went to the library.

(1.2) e_2: Kim went to the library three times.

In (1.1) we have a simple utterance about an eventuality—that of Kim going to the library. e_1 represents the activity that this utterance describes.

(1.2) is an utterance describing Kim going to the library three times. This entire

[1] The precise meanings of these terms and others used in this thesis are also contained in the glossary.

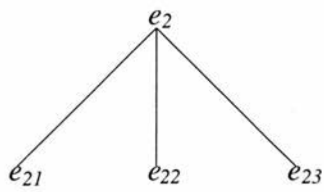


Figure 1.1: The activity structure of '*Kim went to the library three times*'.

activity has been represented by e_2. However, further analysis of the utterance allows us to determine that the activity is structured, and in fact consists of three sub-activities, each sub-activity being one instance of Kim going to the library. If we call these three sub-activities e_{21}, e_{22} and e_{23}, the activity structure[2] of e_2 is then that it is made of e_{21}, e_{22} and e_{23} occurring consecutively. The activity structure of e_2 is shown in Figure 1.1.

This kind of structural analysis of natural language utterances about eventualities is one which has received some treatment in various work, such as that of How[18], Hwang & Schubert[21], Moltmann[29], Karlin[23], Brée & Pratt[36, 37]. However, the research has been within the context of some other interest—this will be discussed in detail in Chapter 2. The main task that this thesis deals with, then, is the provision of a comprehensive analysis covering a meaningful subset of natural language—those parts describing the structure of eventualities that consist of repeated or continuous execution of some activity—and developing a computational implementation that is able to understand this language subset.

The thesis focuses primarily on texts that give instructions. There is an important connection between INSTRUCTIONS and ACTIVITY that is investigated in this thesis. Activity is something that exists in the world, and occurs at a particular time and place. An instruction is an unlocated description of activity, where the protagonist of the activity is the reader or agent understanding the instruction. It is reasonable to say that an instruction is a recipe, or template, for activity.

Also of interest is the notion that the execution of an instruction *in a* CONTEXT has

[2] We are not concerned with the activity content here, which is the business of Kim going to the library, but the structure, which is that there is this thing that happens—Kim goes to the library in this case—and it happens three times.

some input into the process of understanding instructions. Here, context refers to the particular set of circumstances under which an instruction is executed. That is, the execution context, rather than the linguistic context, is of concern. The execution context includes all the objects that are involved in carrying out the instruction. This encompasses things like tools and utensils, as well as the agent executing the instruction.

Let us imagine a situation where some cooking is to be done—we have a cook and some ingredients and some kitchen utensils and some appliances. These form the context for the cooking—the instruction execution—to be done. Let us further imagine two settings, which are described in **Context A** and **Context B**

Context A: a cook, three saucepans, a gas ring, water

Context B: a cook, three saucepans, a 4-ring hob, water

It is possible to do some analysis of an instruction out of context; however, the actual context in which the activity described by the instruction takes place may affect the understanding of the instruction. Executing the instruction in (1.3), taken from Floyd[15], in **Context A** will be different from executing it in **Context B**. Figure 1.2 represents possible activity structures for each context.

(1.3) Half-fill three saucepans with water and bring to the boil.

In **Context A**, we can imagine the cook half-filling the three saucepans with water and bringing each to the boil on the gas ring. These boiling activities would have to be done consecutively; however we do not know whether they would be immediately consecutive or whether some time might elapse between each. In **Context B** however, the context will allow the simultaneous boiling of all three saucepans of water. The instruction itself does not specify whether the simultaneous option or the consecutive option should be chosen[3].

Identifying those aspects of language—about activity that is repeatedly or continuously executed—whose understanding is influenced by context, and distinguishing

[3] Nor does it specify explicitly, for either of the contexts, whether the filling of the saucepans with water should be done before any of the boiling is executed. This is a different issue from that of resources available in the execution context however.

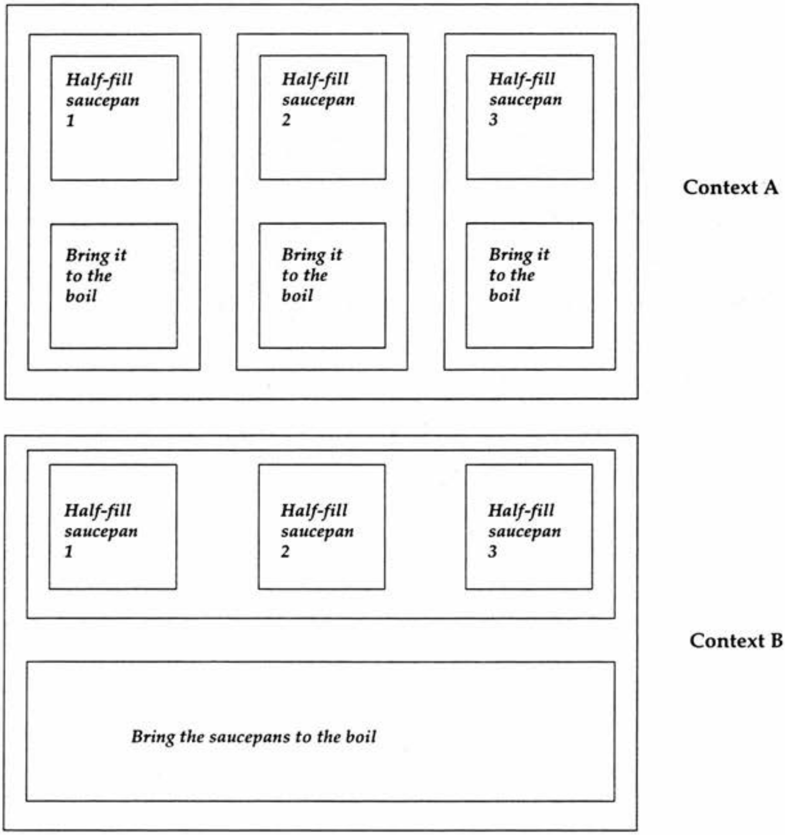


Figure 1.2: Possible activity structures for '*Half-fill three saucepans with water and bring to the boil*'.

them from the context-neutral aspects, is an important part of the work described in this thesis.

The rest of this chapter is concerned with setting the scene for those issues with which the thesis will deal.

- In the next section, the phenomena that are being considered are described more precisely.
- This is followed, in Section 1.2 by expressions of the aims and contributions of the thesis.
- Section 1.3 clarifies the overall methodology that has been adopted.
- In section 1.4 the reader is introduced to some of the concepts and philosophical arguments that are further developed in the thesis.
- This is followed by some introductory remarks about temporal issues, as pertinent to the issues in the thesis.
- In Sections 1.6 and 1.7 the choice of application domain and the role of the implementation are discussed.
- Finally, a brief discussion of what is considered to be outwith the remit of this thesis is presented.

1.1 The phenomena under investigation

At the crux of this endeavour is the wish to identify activity structure—that is, relationships between sub-activities—signalled by an instruction.

It is common to divide the space of what we can talk about into two kinds of things—objects and eventualities. These can occur in a variety of conceptual groupings and forms. For example, with objects, we can talk about plural count objects and mass objects. In this thesis, we are concerned with language that is about the eventuality analogues of these—multiple instances of activity and continuously executed activity.

It is also the case that there is an important relationship between activity and instructions. Instructions themselves are linguistic entities, that can be understood. Distinct from this, however, is the fact that instructions can be executed—that is, they can be translated into activity[4]. On the one hand, we have semantic information about what an instruction 'in the raw' means—semantics that can be extracted from the linguistic entity that is the instruction. On the other hand, there is semantic information that we obtain from the context in which the instruction itself is executed. In terms of activity, this is in the situation in which the activity—described by the instruction—occurs.

These then are the principal issues with which this thesis deals, and the following questions are addressed:

- What are these real-world phenomena of repeated activity and continuously executed activity?
- What linguistic constructions are used to describe them?
- What can we say about the interaction between the real-world phenomena, the linguistic constructions that we use to describe them, and the context in which their actual execution takes place?

1.2 The contributions of the thesis

This thesis has been produced in response to a number of interrelated needs in the area of natural language understanding. In this section, the principal aims of the work are stated briefly. Then the contributions made by the thesis are discussed, with regard to fulfilling these aims.

1.2.1 The principal aims of the thesis

The principal aim in pursuing this research is the provision of a first unified description of language that is about multiple instances of activity, or continuous execution of activity.

[4] This distinction is exemplified in computer programming language translation in the distinction between compile-time issues and run-time issues.

Sub-goals that support this aim are the following:

1. Exploiting the analogy between objects and eventualities in order to achieve the understanding of such language;
2. Extending work in the understanding of instructional texts[5];
3. Providing a computational implementation that embodies the principal aim described above.

1.2.2 The research contributions made by the thesis

The contributions that this thesis makes are on various levels.

The principal worth of the thesis lies in its attempt to consolidate various bodies of research that relate to a particular problem in natural language understanding—that of understanding descriptions of particular kinds of complex activity. The main aim of the work is thus to examine

- multiple instances of activity and activity that occurs continuously—loosely, plural and mass in the context of eventualities
- how we talk about activity of this sort

and to provide a coherent account of the interaction of these two.

Although there has been some work on aspects of this, there has been no *unified* account. For example, there has been work on adverbial modifiers, such as that of Hitzeman[16] and Karlin[23]—these form only a part of the language that we use to express repeatedly or continuously executed activity. There has also been work on temporal prepositions, and their associated adverbials, by Pratt & Brée[36, 37]—again, this only accounts in part for how we can talk about this kind of activity. A unified account of language about this activity does not exist[6].

[5] I clarify that my use of the words 'instructional texts' refers to text that is in the form of instructions, rather than texts that are used for instructional, or educational, purposes.

[6] I do note however, that although the work of these authors covers only some of the language about this kind of activity, it also in many cases has coverage outside of the area on which I focus in this thesis.

To that end, then, this thesis provides a coherent account of how we employ language to describe activity that is complex because it consists of repeated subactivity or extended execution. This account is embodied in a computational implementation, that is able to take language fragments that describe activity, determine the internal activity structure that is described by the language, and produce a graphical representation of that structure.

At a more detailed level, there are some other important themes that contribute to the originality of the thesis.

- First, the thesis uses the claims made by various authors, including Mayo[26], Zemach[57] and Jackendoff[22], for viewing objects and eventualities as mutual analogues with respect to space and time.

  Much has been written on the analogies that can be drawn between objects and eventualities. A common view is that objects exist in space as eventualities exist in time. There has also been much written about how we talk about plural and mass in the context of objects. We can conceptualise repeated and continuous activity as the eventuality analogues in time, of plural count and mass objects in space.

  In computational linguistics, a lot of work has been done in understanding natural language about objects. For example, quantifiers have usually been treated as they apply to objects; the same is true of anaphora. This thesis takes the stance that it is useful for this work, done on objects, to be applied to language about eventualities. This approach is utilised in both theoretical and computational arenas. All of the analysis of activity structure is done with respect to the analogies with object structure; the synthesis of an account of how we can understand language about this kind of structured activity also refers to the analogies. In addition, in the computational implementation that demonstrates this analysis, existing algorithms for understanding language about objects have been used, with success, to understand language about eventualities. This is done in particular by providing a computational treatment of quantification of activity that is the eventuality equivalent of Hobbs & Shieber's well accepted computational treatment of quantification of objects[17].

  The thesis therefore supports—in a novel manner—the validity of the claims about the analogy, as well as identifying areas where the analogy is more subtle.

- Secondly, because the thesis is concerned with activity, instructional texts are a major focus. It has been noted, by Webber[55] and Chapman[8] among others, that natural language instructions are an under-researched area in general. Recently, though, aspects of the topic have been receiving attention. Examples of such work include understanding temporal ordering described by instructions (How[18]), analysing purpose clauses in instructions (Di-Eugenio[13]), and others. However, there are aspects of instruction understanding that have not been fully researched. This thesis aims to extend our coverage of instruction understanding to include instructions about repeated and continuously executed activity, thereby contributing significantly to research about instructions.

- Thirdly, the analysis is done with regard to instructions being executed in a context. The relation between activity and context is also an area of research that is receiving attention. For the particular area of concern—language about activity that is structured as described above—the thesis provides some insight into the role that execution context plays in the understanding process. In doing so, the broader contribution of enhancing our general model of instruction execution in context is made.

- Finally, this thesis is situated within the field of computational linguistics. It is therefore appropriate to embody some of the theory that is developed in a computational system. What is provided is thus a computational system that can take language fragments that are about repeatedly or continuously executing some activity, and produce a visual representation of this execution structure.

1.3 A note about methodology

The work in this thesis has been approached from a particular viewpoint. This viewpoint is that there is a phenomenon in the world—that of activity occurring repeatedly or continuously—that people talk about using various linguistic constructions. In the thesis, we first look at the fact that activity can be viewed in this structured way, and examine what kind of structuring can be done and what other kind of structuring it is similar to; this is where the notion of analogies between objects and eventualities comes in. Then we look at the ways we have of talking about this structuring; how these activity phenomena are captured using language.

More generally, the approach taken in this thesis is one where some phenomenon is delimited and we examine ways in which language (and then language processing) interacts with the phenomenon. The approach is akin to that taken by How[18], and by Singh & Singh[47], for example.

This is in contrast to the approach that is often taken in computational linguistics research, which is to choose some linguistic phenomenon first—say adverbials as in the work of Karlin[23] and Hitzeman[16] or prepositions, as done by Pratt & Brée[36]—and then examine what we are able to talk about, and how we are able to express things, using language of this sort.

It is not my intention to claim that either approach is a better one; I only wish to point out that they are different. Clarifying the approach taken in this particular thesis will, I believe, assist the reader in following some of the thesis structure and argument in subsequent chapters.

In the context of the approach taken, then, the study presented here is strongly language bound. The analysis, and hence the language understanding, goes as far as that suggested linguistically. We look for repetition, or extended execution, or multiple instances of activity, that is given only by understanding the language. So, in (1.4), although we know that reading a book involves repeated turning of pages, say, we do *not* identify this page turning as a repeated activity. However, we do identify that in both (1.4a) and (1.4b) it is possible to identify repeated activity—that of there being two instances of the activity of ≪*a book being read by Kim*≫[7].

(1.4) a. Kim read the book twice.

b. Kim read two books.

[7] This notation is used throughout the thesis to indicate a semantic gloss; it can be seen as representing 'what we might understand by'.

Clarifying this is important—because the phenomenon we are investigating can be viewed at a number of levels, we need to be aware of the role language is playing. It is reasonable to assume that language provides the level at which a reader is to conceptualise activity; this approach is one that is often taken in computational linguistics research. We return to discussion of these issues, and how they pertain in particular to activity that is conceptualised as repeated or continuous, in Chapters 3 and 4.

1.4 Some starting points

This thesis is in large part a drawing together of concepts from a variety of arenas. The purpose of this section is to introduce to the reader some of the theoretical and linguistic issues that are elaborated and utilised in later chapters.

1.4.1 The application of this work

It is useful at this point to clarify the orientation and motivation behind this work in terms of its application, as this informs some of the decisions taken and also influences the relative significance given to different issues.

I have imagined a robot being instructed to perform cooking tasks; a robot that is capable of similar physical ability to that of a human cook. The main application of this work is to understand the repetition and continuous activity that is described in instructions, to be able to present this to a robot for execution. The form of this presentation in the thesis is in fact as visualisations which are simple graphs, which are described in Chapter 7. However, keeping in mind the envisioned application—ultimately of being able to instruct some robot-like agent to perform repeated or continuous activity, and enabling the understanding of the nature of such repetition or continuity as signalled in the language itself—will be useful for reading the rest of the thesis.

1.4.2 Objects and eventualities

A common view of the world is that there are two kinds of basic thing in it that we can talk about—objects, which are tangible and exist physically in space, and eventualities, which are things that happen and exist in time. Certainly it is the case that an object also has some temporal aspect, and an eventuality may happen in some place. However, these are less important aspects of the entities concerned.

It has also been noted that there are strong parallels that can be drawn between our views of objects and our views of eventualities. Mayo[26] claims that events are the ontological reverse of material objects with respect to time and place. Zemach[57] argues that we do not need to have both ontologies—that either ontology is in fact sufficient in itself for talking about entities in the world. In the next chapter, we will examine in more detail both of these claims, and also look at other authors proposals—Bach[4] for example—for ontologies of events. In the rest of the thesis, then, we make use of the analogy in accounting for language about eventualities; we also identify places where the analogy breaks down.

1.4.3 Plural and mass

The concepts of plural and mass are familiar ones when we think and talk about objects. We all have experience of plural count objects, such as '*six apples*', and mass objects, such as '*water*'. Every object can be viewed as being either count or mass[8]. We distinguish between the two by saying that for something to be mass, a part of that something must also be that something. So, if we take water, a part of water is still water; it is mass. If we take six apples, a part of that is no longer six apples. They are count, and plural in this case. It is also the case that when we delimit a mass, we make it a count object. So, when water becomes a glass of water, it passes the tests for count objects.

Less familiar, though equally plausible, is the notion of plural and mass with respect to eventualities. We can think of plural eventualities, such as '*jumping three times*', and activity that is mass-like[9], such as '*sleeping*'. Many of the tests

[8] Sometimes the same linguistic item could represent either—we can talk about '*an apple*', meaning ≪*a single apple*≫ or '*some apple*', meaning ≪*some amount of a substance that is called apple*≫.

[9] I hesitate to use the term 'mass activity' because it conjures images of a large number of people

and concepts that pertain to plural count and mass in objects apply to repeated and continuous eventualities. For example, part of sleeping is still sleeping; part of jumping three times is not jumping three times. Parallels have been noted by various authors—Jackendoff[22], Bach[4], Talmy[51]—but this is still an area of research and discussion. It is argued in later chapters that some of these views need to be extended, though the basic concepts of plural and mass applied to eventualities are crucial.

1.4.4 Continuous execution, repetition and multiple instances of activity

Repeated activity and continuous activity are the main focuses of interest in this study. We have the simple view that repeated activity is the eventuality analogue of plural count, and that continuous activity is the eventuality analogue of mass.

With plural count in objects, when we have some number of objects, what this really means is that we have a number of instances of the object. If we have say '*four chairs*', we have ≪*four instances*≫ of objects that are ≪*chairs*≫. Turning to activity, repeated activity is then a number of instances of an eventuality. With '*jumping three times*', we have ≪*three instances*≫ of activities that are ≪*jumping*≫.

It is important to point out, however, that with the repeated activities, they happen *consecutively*. That is, they happen one after the other; this is a common view of the meaning of repetition.

It is also plausible, though, to imagine multiple instances of activity that do not occur consecutively. An example of this would be ≪*the film "Gone with the Wind" being shown a number of times*≫. Nothing in this insists that the showings could not happen simultaneously; it is possible that they all happen at the same time in different cinemas. Here we also have something that can be conceived of as the eventuality analogue of plural count: multiple instances of an activity. Thus, repeated activity is only one kind—a subset in fact—of 'plural' or multiple instances of an activity.

This distinction—between multiple occurrences of activity that occur consecutively

engaging in some activity, rather than activity that is homogeneous in nature.

and those that need not—is one which has not been noted in previous work that exploits the object-eventuality analogue, and is one which is significant in this thesis.

The term EXTENDED will be used to describe activity that is structured in this way that is analogous to plural and mass in objects. Thus, we would say that the sentences in (1.5) describe extended activities.

(1.5) a. Jo yawned a few times.

b. Lee slept peacefully for three hours.

1.5 Temporal aspects

Time is an essential aspect of activity—activity occurs in time. Moreover, time is especially relevant to activity whose execution is extended or repeated. The discussion that follows relies on a conventional view of time—that time is a single directional[10] axis. We have a concept of 'now', and concepts of 'before now' and 'after now'; other temporal relations and considerations follow from these basic starting points.

We first look at temporal considerations with respect to eventualities themselves, and then go on to discuss temporal considerations as they pertain to language.

1.5.1 Time and eventualities

The main focus of this thesis is eventualities, and eventualities are strongly related to time. Eventualities occur in time, and part of what is unique to a particular event are its temporal properties. For any event that actually exists, or occurs, there is time associated with it. This may be the time at or during which it occurs—the TEMPORAL LOCATION—or the time it occupies—the TEMPORAL EXTENT. These two kinds of time give us different information, although they are closely connected to each other and to the concept of universal time. For the discussion presented in this thesis, it is important to distinguish between them, and to be

[10] Mayo[26] argues for omitting the directional component when considering time; this is discussed in Chapter 2.

clear which aspects of eventuality analysis are influenced by each kind of temporal information.

The temporal aspects are not always explicit when we talk about or think about events; however whenever activity occurs there *are* these temporal components. Let us think of some abstract eventuality, say that of ≪*the showing of the film "Gone with the Wind"*≫. It is easy to imagine some temporal extent for this eventuality: a few hours, or however long it takes to show the film.

It is tempting, though, to see the eventuality as not really having a time location component—that is, not being attached to the universal time axis. But, as soon as the eventuality exists—becomes something that happens, or has happened, or will happen—described by ≪*the showing of the film "Gone with the Wind"*≫ happens, the event of that showing has an associated temporal location component. It is the case that there may be a number of showings—each one that occurs has a temporal location. It is true that the temporal locations of two showings, say, may be the same. The showings are distinct events though, and each has its own associated temporal location. The question of spatial locations for these showings is not of concern here. Eventualities always have a temporal component; they may or may not—such as in the case of the eventuality of ≪*Kim liking swimming*≫—have a spatial component.

Some eventualities are seen as INSTANTANEOUS. It is important to be clear however, that this does not mean that there is no temporal extent component. Rather, it means that the size of the temporal component is extremely small. Similarly, some eventualities may not have a finite or definite temporal extent. This does not mean that the temporal component is qualitatively different. The eventuality described by ≪*Kim sleeping*≫, when it actually occurs, has a temporal location and a temporal extent. Perhaps that temporal extent is known, perhaps—at the current time—it is not known. Both possibilities need to be accounted for.

1.5.2 Time and language about eventualities

Language that is about eventualities incorporates temporal information in a variety of ways. Tense is a mechanism for situating eventualities to some degree. So, tense gives us a way of using language to include temporal information. The kind of temporal information that tense gives us is that about when in time eventualities occur; where on the time axis an eventuality is situated. There is much research concerning issues of tense; however this is not of interest in this thesis.

With instructions, the temporal aspects are interestingly different to those of other descriptive language. Instructions are descriptions of activities, which are given existence—and thereby temporal situation—when the instruction is executed. While it is sometimes the case that instructions include directives about explicit temporal location, the instruction itself does not contain a link to a temporal location. It contains a link to an activity, and the activity is linked to temporal location *when* the activity occurs; when the instruction is executed. This is illustrated with the following examples:

(1.6) Go to the meeting place at 4pm on Sunday.

(1.7) Put the wet clothes in the dryer.

In (1.6), only when the instruction is executed, and the activity occurs, do we get a temporal location. That is, the instruction itself does not have location. If it doesn't get executed, then *nothing* happens at 4pm on Sunday. Similarly, in (1.7), each time the wet clothes are put into the dryer—each time the activity happens—there is an associated temporal location. The location is the temporal location of the *activity*, not of the *instruction*.

We have also seen, in Section 1.4.4, that with multiple instances of activity, we can distinguish between those that occur necessarily consecutively and those that do not. This distinction is often expressed by the language—for example using phrases like '*three times*' expresses consecutive execution. In later chapters the role language plays in signalling the temporal structure of repeated sub-activity is discussed in detail.

1.6 The application domain

We have already seen that this thesis deals only with language that is about activity that is repeated or continuously executed. The thesis also focuses on instructional texts—because they are intimately connected with activity, and because they display properties that are useful in the analysis presented. Even with this narrowed focus, the breadth of natural language means that it is useful to concentrate on a particular domain of application—cookery recipes is the one chosen for this thesis.

The analysis that follows in the rest of this thesis focuses on, but is not exclusively limited, to cookery recipes. Where appropriate, instructions from other application domains are used to illustrate particular points; also non-instructional language is sometimes included to demonstrate that the theory is applicable to a wider domain. The computational implementation, however, deals exclusively with cookery recipes. This is a necessary and reasonable restriction for a computational system.

1.6.1 Instructional texts

Instructions were chosen for a number of reasons. It has already been indicated that they have a special interaction with activity—they are primarily *about* activity.

Instructions exhibit a limited amount of tense information; tense is generally absent from them. This is extremely useful in this thesis, as it is concerned with temporal issues that are *not* about tense. In addition they also exhibit many of the phenomena that are of concern—repetition of activity; extended execution of some activity; combinations of activity.

Distinguishing between the things that we get from language itself and the things we get from execution context is an important concern in this work. Instructions are an ideal base from which to investigate these issues.

1.6.2 Cookery recipes

The cooking recipe domain is interesting for a number of reasons. It is a limited domain, yet it exhibits some complex linguistic phenomena. Recipes have been chosen as a domain of investigation by a number of researchers—among them Karlin[23], Dale[12] and How[18].

It is also a domain where there is a large amount of data available. In particular, recipes contain many examples of language about the issues of interest to this study—repeated and extended execution of activity. Recipe books are plentiful, and have existed for long enough for there to be some conventions that are both useful and interesting. For example, it is conventional to list ingredients and then provide the cooking instructions. This separation is a useful one in practical terms; it also introduces elliptical language that is challenging.

1.7 The role of the implementation

It is appropriate at this point to discuss the inclusion of the computational implementation in this thesis. A computational system has been developed that is able to take in language from the application domain—cooking recipes—and identify from it activity structures that are being described. The activity structures are represented textually, in the form of feature structures[11].

These representations are then visualised, in the form of structure diagrams. The decision to use diagrams to demonstrate activity structure was motivated by two important factors. First, it is a further exploitation of the object-eventuality analogy. It uses objects that contain sub-objects and their relative physical location, in order to demonstrate eventualities that contain subeventualities and their relative temporal location. A second factor is that it is often the case that using a different medium of representation—in this case pictorial, rather than textual—allows easier representation of concepts that may be difficult to represent in the original representation.

An example is shown in Figure 1.3; Chapters 5, 6 and 7 describe the implementation and the form of the feature structures and subsequent diagrams in detail, while worked examples are shown in Chapter 8.

The implementation is included to demonstrate the consistency and the computational feasibility of the theory developed. The work is situated in the field of computational linguistics, and the theory developed is oriented towards being able to provide a computational system for language understanding. Though the system I have developed is by no means a comprehensive one for all language about the phenomena under investigation, it does cover a substantial part of the account that this thesis provides, and is an effective demonstration of the plausibility of

[11] Feature structure representations are described in Section 5.1.

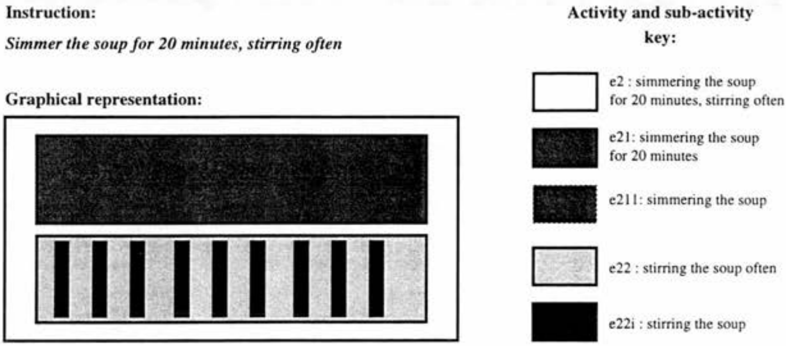


Figure 1.3: Example of language fragment and corresponding visualisation.

the account as a computational analysis.

1.8 Issues beyond the scope of the thesis

Because of its orientation towards consolidation, the work in this thesis touches on a number of areas. This section is concerned with delimiting the bounds of the thesis and stating the limits of its extent.

First, the primary issue of concern is activity *structure*. Activity content is not of interest, other than where it intimately affects structure. So, in the activity described in (1.4a), reproduced here, we do not analyse what it means for Kim to read the book—we are not interested that this activity involves Kim doing something that is ≪*reading*≫, that this ≪*reading*≫ is done to something that is a ≪*book*≫.

(1.4) a. Kim read the book twice.

b. Kim read two books.

The only concern is to determine that ≪*something happens*≫ and ≪*it happens two times*≫. With (1.4b), again we are not concerned with what it means to ≪*read a book*≫, but we *are* concerned with what it means to be ≪*two books*≫, as this may mean that there is sub-activity to be identified. Of course, there will be some aspects of the ≪*something happens*≫ that we need to know about in order to analyse the extent of its execution. We may need to know what kind of basic activity we have, in order to determine what extent of execution is possible. However, this is all done with respect to the structure of the overall activity.

Secondly, analysis of tense is not within the remit of this thesis. Aspectual information is used but this is distinct from tense. I do not present any analysis of how tense interacts with the internal temporal structure of activity; nor are any examples that contain tense information dealt with. Both of these are seen as interesting issues for future research, however.

Graphical representations are utilised, to demonstrate activity structure. These representations are extremely simple, and their limited usefulness is primarily in demonstrating eventuality structure via the use of objects. In Chapter 7, some of the problems and difficulties with graphical representations are indicated—that there are problems is well accepted—and particular shortcomings in the context of representing repeated or extended activity are identified. There is no claim to have developed a visual representation system that is free of or deals with these problems; however the graphical representations are useful in the sense described above.

As indicated in Section 1.3, the analysis is limited to understanding that can be obtained from the language. This means that no world modelling, planning or reasoning about activity content is included. We do indicate the appropriate places for such concepts to be included, but the inclusion is outwith the scope of this thesis.

The computational implementation is limited to deal with a recipe subset—this restriction is a reasonable one; its motivation is described in Section 1.6. At a more detailed level, the implementation also does not include resolution of anaphoric references to activity. This is discussed in the thesis, and clear indications are given of where such implementation would fit in. The same is true of context related information.

In the final chapter, some of the limitations mentioned here are discussed in the context of the entire thesis.

1.9 The structure of the rest of the thesis

The rest of this thesis is structured as follows:

- Chapter 2 reviews previous work that is pertinent to the thesis. This includes work on understanding instructions, understanding repetition, the analogues mentioned above, etc.
- The next four chapters are the central chapters of the thesis, and are linked in various ways. However each chapter has a clear principal function.
  - In Chapter 3 the phenomenon of activity—in particular the phenomenon of activity that gets repeated or continuously executed—is discussed in detail. That is, the focus is on the 'real world thing' that the natural language of interest talks about.
  - In Chapter 4 then, the linguistic issues are discussed—we look at the natural language that is used to describe the real world thing.
  - Chapter 5 is about semantic representations for the kinds of linguistic items that are discussed in Chapter 4.
  - In Chapter 6 I discuss how the two fit together—that is, I discuss how we can go from language to semantic representations corresponding to that language. In this chapter then, the ways in which we can talk about repeated and continuously executed activity are mapped onto representations repeated and continuous activity. This is the business of finding the semantics of natural language utterances; what is described in the chapter are mechanisms for understanding language about repeated and continuous activity.
- Chapter 7 describes an overall view of the understanding process described in the thesis, and a computational realisation of the mechanisms proposed in Chapter 6. It also describes a simple visualisation system that has been implemented in order to demonstrate that the semantics produced are meaningful.
- Chapter 8 presents a representative sample of worked examples, to illustrate theoretical concepts and practical issues.

- The final chapter is a summary and tying up, including the identification of potential areas for further research.

Chapter 2

Background

This chapter is concerned with introducing and discussing existing research that is related to and influences the work of this thesis. As has been stated in the previous chapter, a large contribution that this thesis makes is the drawing together of concepts and approaches to understanding language about a particular phenomenon—that of events that occur repeatedly or continuously. The role of this chapter is thus to present and discuss these concepts and approaches as they occur in the literature.

- In Section 2.1 issues pertaining to ontology are discussed. In particular, research about the object-eventuality analogy that is central to this thesis is summarised and work that discusses the notion of plural and mass in eventualities is critiqued. We also discuss some of the eventuality hierarchies that have been presented.

- Section 2.2 is about work on temporal issues that impinges on this thesis. This comes in two main parts—work on aspectual information (which sometimes but not always interacts with tense), and work on temporal analysis of eventualities.

- Section 2.3 describes work that has gone some way to analysing the kind of language of concern here. In particular, work that includes some analysis of temporal adverbials is discussed. This includes approaches that treat some temporal adverbials as quantifiers; we will see that such an approach has many advantages for the task tackled by this thesis.

- The final section is concerned with instruction understanding, instruction execution and visualisation.

2.1 Ontology: objects and eventualities

In this section I present previous work that has been concerned with the ontology of eventualities and with the ontology of objects. Section 2.1.1 is concerned with philosophical notions of eventualities and objects. Section 2.1.2 presents work on eventuality composition and the notions of plural and mass with regard to eventualities. In Section 2.1.3, two proposed eventuality hierarchies are discussed and the hierarchy that is adopted in this thesis is identified.

2.1.1 The object-eventuality analogy

Similarities in how we can look at objects and how we can look at eventualities have been noted in the literature for a long time. This has been with regard to studies of ontology as well as studies of language. The object-eventuality analogy is the central theoretical basis of the work presented in this thesis. It is not my intention to try to summarise all of the work on this analogy here. Rather, I describe in detail those views and work that contribute to the theoretical underpinning, and hence inform the decisions taken in this thesis.

Four ontologies

We begin by looking at a thesis proposed by Zemach[57], which forms the basis of a discussion of the kinds of entities in the world that we can talk about. In his 1979 paper, which is philosophical in nature, Zemach puts forward the view that when we talk about entities in the world, we use terms from a mix of ontologies, and that in fact it is possible to talk about the same entities using terms from any one ontology. The paper elaborates this view, and provides supporting argument for it.

The primary claim made by Zemach[57] is twofold:

1. when we talk about entities in the world, we use terms that designate entities

Space	Time	Ontological entities
Bounded	**Bounded**	events (non-continuants)
Bounded	**Continuous**	things (continuants in time)
Continuous	**Bounded**	processes (continuants in space)
Continuous	**Continuous**	types (pure continuants)

Table 2.1: A summary of Zemach's ontologies.

in four ontologies;

2. each one of these ontologies is itself complete and sufficient.

Zemach believes that we use terms from a mix of these ontologies for pragmatic and historical reasons, rather than for reasons of adequacy. He first goes about delineating these four ontologies, and then shows that each is complete and able to express all entities that any of them can.

The four ontologies that Zemach presents are distinguished by whether they are BOUNDED or are CONTINUOUS in space or time—this gives the four ontologies that I summarise in Table 2.1. By the term continuous, Zemach does not mean 'unbroken' or 'uninterrupted'—for this concept he uses the term CONTIGUOUS. Continuous is used in the sense of not having parts, while bounded means having parts. Boundedness or continuousness are defined with respect to dimensions—in the discussion that Zemach presents the dimensions are time and space.

The four ontologies are therefore NON-CONTINUANTS, CONTINUANTS IN TIME, CONTINUANTS IN SPACE and PURE CONTINUANTS—NC, CT, CS and PC respectively. Zemach discusses each of these ontologies, and then goes about showing that we can talk about anything in the world (he restricts his discussion to spatiotemporal entities and excludes classes, numbers, universals and gods) using terms from any one of them.

Non-continuants: NCs—which Zemach calls events, but these entities do not correspond with our usual concept of events being things that occur—are four dimensional entities; any chunk of filled space-time is an event. Zemach's events are noncontinuous, do not endure, but may have contiguous or non-contiguous parts. I will say no more about NCs as they are not pertinent to the work of this thesis; Zemach himself has very little to say about this ontology other than that it exists and its terms are sometimes used when talking about entities in the ontology of processes.

Continuants in time: The most commonly used ontology is the one of CTs—the ontology of 'things'; entities that are bounded in space. Any object we can think of can be regarded as a CT. '*The table*', '*that pencil*', '*my nephew Ariel*' are names of entities that can be regarded as things. Zemach notes that all of these can be recategorised (translated) and regarded as events—indeed according to his thesis they could also be regarded as CSs or PCs.

CTs or things are bounded in space, and continuous in time. So, with regard to the spatial component, a thing has spatial parts. It can be divided in parts and these parts are different parts of the thing. With respect to time, however, a thing is a continuant. At different times, the thing is *not* a different thing. My nephew Ariel has parts that are different parts of Ariel (perhaps arms, legs; perhaps blood, bones): spatially Ariel is bounded. But the Ariel I see today is the same entity as the Ariel I saw yesterday: with respect to time, Ariel is continuous[1]. To summarise, viewing something as a CT involves viewing it as defined with respect to its location in space, but not defined with respect to its location in time. Zemach gives the example of a pin: the definition of a pin—in the CT ontology—specifies that whatever is a pin must have a certain defined spatial existence, but it says nothing about what sort of career a pin should have. Thus two pin-entities are one and the same thing only if there is temporal distance between their respective locations.

[1] What about the fact that Ariel is older today, and may have grown a bit—is this still the same Ariel as yesterday's Ariel? I believe that what Zemach would say is that it is the way we view Ariel—and so yes it is the same Ariel. If we were viewing Ariel as a CS, then today's Ariel could be different. In fact, that is a crucial point—we can view entities in any of the four ways he identifies, and depending on which we have chosen, we will have some associated views about sameness and changing.

Continuants in space: The ontology of CSs is the ontology of processes; entities that are bounded in time. Zemach claims that the terms of this ontology are often confused with the terms of the ontology of events (NCs). 'The October Revolution', 'this noise', 'the Roosevelt era' are entities from the ontology of processes—bounded (defined) in time but continuous in space. So, as a process, 'World War II' can be at the same time in many places—continuous in space—but cannot be in the same place at many times—bounded in time. I point out that this is not to say that 'World War II' occurs only for an instant; it can occur for a large period of time, but any occurrence of 'World War II' that is outside of this period (say the period 1939 to 1945) has to be conceived of as a different instance of 'World War II'. Zemach exemplifies this further by suggesting some explosion happening—two people living in different streets hear the explosion, and they hear *the same* explosion. An explosion that occurs a day later is a different explosion.

Although Zemach distinguishes between continuousness and contiguousness, he does not discuss their interaction. So, he does not discuss how we might view these two explosions if they happened immediately after each other. Are they two different explosions or are they parts of the same explosion? I believe that this is not a problem; explosion A and explosion B may be regarded as either, but whichever configuration we choose (different explosions or the same explosion—say explosion C), if we are drawing on the CS ontology, the same boundedness and continuousness properties hold. Regarding explosion C as made of explosion A and B has to do with structure within ontology, and not with which ontology is being adopted. So even if explosions A and B occurred a day apart, if we considered explosion C to be defined by both of these explosions, explosion C is still bounded in time and continuous in space. Continuousness with respect to a dimension is to do with whether an entity endures in a dimension, rather than its actual location or contiguousness within that dimension.

Pure continuants PCs are what Zemach views as types—continuous (enduring) in both time and space. He argues that types are not categories, universals, classes or abstract entities, as has been variously suggested, but are material objects. When we talk of 'The African Antelope' or 'The Taxpayer' or 'The Person on the Street', we are using terms from the fourth ontology. Zemach demonstrates through examples why it is problematic to see these as abstract entities or classes;

he further says that it is probably possible to reduce these to named entities that are only things—this is after all the substance of his thesis—but believes that there is nothing to be gained from this enterprise (which he describes as "stamping out an ontology").

He then presents an important contention, which is that the terms we usually call mass nouns behave like names of PCs, and that they should be regarded as such. It has often been thought that we distinguish mass nouns by the fact that they name scattered individuals—'water' names the aqueous part of the universe, 'red[2]' names the red-looking part of it. 'The River Thames' is a spatially contiguous entity; it is not a mass noun. Zemach adds to this—for him it is the fact that whenever water is present, 'water' (and not a water-part) is present. This is not the case with the River Thames—different parts of it are parts of the River Thames. This concept is illustrated with examples that are CSs and examples that are CTs. I can point to Ariel's leg and say '*this is part of Ariel*'; however if I point to water, you do not expect me to say '*this is part of water*'. If I hear music, I do not hear part of music; if I listen to one rendition of the Moonlight Sonata, I do not say that I have heard only part of the Moonlight Sonata—that in order to hear it I have to hear all of its renditions.

This approach has important impact on the analysis presented in later parts of this thesis, where we are concerned with ways to view eventualities when they are mass-like. For example, Moens & Steedman[28] distinguish '*hearing music*' and '*hearing the Moonlight Sonata*' as describing entities from different aspectual categories. For Zemach, mass nouns are from an ontology different to the one of things; they are also from an ontology different to the one of processes. He would distinguish between 'The October Revolution' and 'The Moonlight Sonata' as coming from different ontologies; I believe that he would consider 'one particular rendition of the Moonlight Sonata'—or five renditions or four thousand—as coming from the same ontology as 'The October Revolution'. Moens & Steedman, on the other hand, don't distinguish between 'hearing the Moonlight Sonata' and '*hearing one rendition of the Moonlight Sonata*'

For this thesis, I exploit the analogy between things and processes (objects and eventualities is the terminology I shall use for these kinds of entities respectively)

[2] Although we also think of red as an adjective, 'the colour red' is a noun.

as discussed in the next section; however a clear view of the distinction between process and thing entities and mass-like entities is an important part of this.

The object event complementarity thesis

We now turn to earlier work—a paper by Mayo[26], published in 1961—that also discusses, from a philosophical viewpoint, ways in which we can view and understand the world. Mayo's particular concern is with spatial and temporal notions; the notions of object and event. The central part of Mayo's thesis is that there is a complementary relationship between objects and events; that they are mutual complements with respect to space and time.

This thesis relates to Zemach's[57][3] work in that it involves two of his ontologies—the ontology of CSs and the ontology of CTs. However, Mayo's intent is to try to show that events are exactly the ontological reverse of objects. So, Mayo's work concerns only these two ontologies, and is about demonstrating their mutual complementarity.

I have summarised Mayo's argument in Table 2.2. This argument relies on a notion that time and space are distinct dimensions, and that neither is ordered. That is, according to Mayo we need to attach less importance to the idea that time naturally progresses 'forward', and instead take a more abstract view that it extends in any direction—one of two since it is a single-dimensioned entity—in the way that space does. This is a somewhat unusual view; Mayo explicates it in terms of an analysis of what it is to 'move'.

According to Mayo[26, pg351–352],

> "To move through space or, properly, to move, is to be in different places at different times ... To be in different places at different times is to move. For objects, this is locomotion. To be at the same place at different times is not to move. For objects, this is to be at rest. To be at different places at the same time is impossible for objects; we have

[3] Mayo's paper was published fifteen years earlier than Zemach's. However, I have presented their work in the reverse order here because I feel that the ontology distinction should precede discussion of ontology correspondences. Zemach in fact makes some reference to Mayo's paper, and says that although the analogy is a good one, parts of the exposition are unclear and could be misleading.

	Object	Event
I	Limited extension and unlimited duration Cannot occupy the whole of space but could occupy the whole of time There must be room in space for many objects, which may or may not overlap temporally	Limited duration and unlimited extension Cannot occupy the whole of time but could occupy the whole of space There must be room in time for many events, which may or may not overlap spatially
II	Can at different times occupy the same space or different spaces	Can at different places occupy the same time or different times
III	Cannot be at different places at the same time	Cannot be at different times at the same place

Table 2.2: A summary of Mayo's object and event analogy.

> to say that there are two such objects, perhaps similar but numerically distinct."

He then gives the complementary analysis as applied to events, but uses the term propagation for the locomotion analogue of events.

> "To be at the same time in different places is, for an event ... to occur at those places. ... to be at different times in the same place is impossible for events. Again we should have to say that there are two such events, perhaps similar but numerically distinct."

Mayo then clarifies the notion 'moving through time':

> "To "move through through time" then, is either (i) just to move, which is the same as moving through space; or else it is (ii) a nonliteral way of referring to something else. This something else can only be *endurance*. To move through time is just to endure. The spatial complementary of endurance is extension."

This said, Mayo argues that we may expect to find a spatial complement for this nonliteral moving through time, and says that this is the notion of 'spreading through space'—I'd call the temporal analogue 'enduring through time'. The spreading is not directional—if for example we conceive of a desk, we do not have an image of where it begins, and in which direction it spreads. It is the case that we may be standing at one corner of it, and thus decide to consider that the desk begins at this corner, but this is only for convenience. For an event, say that of a lecture being given, it endures in some temporal existence. Mayo would say that it is only convention that makes us consider that the lecture begins at a particular time. He argues that as easily as we could take the middle of the desk and see it spreading both forwards and backwards in all three dimensions, we could take the middle of the lecture event, and see it as enduring ('into the distance') away from the perceiver in both temporal directions.

So, if we extend this view, it would seem that Mayo wants us not to see time as some moving axis that we experience by being, in a sense, on a conveyor belt. He does not address in much detail the distinction between on the one hand 'moving through space' and 'spreading through space' and on the other 'moving through time' and 'enduring through time'. It seems to me that moving through space naturally includes a change in time and place—indeed Mayo claims this too; spreading through space (in the sense that Mayo uses it, to mean spatial extent) does not carry an implication of a change in time or space.

Let us clarify with an example, using the second condition in the summary of Mayo's analogy (Table 2.2). This states that

II An object can at different times occupy the same space or different spaces.

An event can at different places occupy the same time or different times.

Let us consider an object, say a pin, and two times, say time A and time B. If the pin does not move—through space—then at times A and B it will occupy the same space. If it does move (locomotion is what Mayo would call this moving), then at times A and B it will occupy different spaces; yet it is the same object.

Now we will apply the same argument to an event, say a rendition of the Moonlight Sonata, and two places, say place Y and place Z. If the rendition of the Moonlight Sonata does not move—through time—then in places Y and Z it will occupy the same time. If it does move (and here I believe Mayo would call this propagation), then at places Y and Z it will occupy different times; *yet it is the same event.*

This seems very strange in a conventional view of the world: we would never say that the pin is a different pin if it turns up (tomorrow) in a different place. Yet, we would say that the rendition of the Moonlight Sonata that occurs today is different from the one that occurs tomorrow. But is it? How would we know if it is the same? In fact, how is it that we know tomorrow's pin is the same one as today's? It seems that it is only our view of time being a moving axis that makes us treat these entities in different ways; we have an image of that pin being 'carried through time, by the conveyorbelt that is time passing'; a thesis that Mayo rejects. For Mayo's thesis to be adopted, we would either need to say that at time B, the pin in a different place is (always) a different pin from the one we found at time A; or we would need to say that in place Z the occurrence of the Moonlight Sonata at different times is the same occurrence of the Moonlight Sonata as occurred in place Y. This is in conflict with Zemach's view of continuants; it is also in conflict with our intuitive view of objects and eventualities in space and time.

Mayo's thesis that objects and eventualities have a complementary relationship with respect to time and space is a useful one that will be exploited in the work presented here. The notion that time should not be viewed as a directional axis—and all the associated implications for objects in time and space—is, however, not one that I adopt; I use the conventional idea that time is directional.

2.1.2 Eventuality composition

Notions of mereology and topology

There is a body of work on the topic of MEREOLOGY which bears on the work here in that it is about structure. Again, I will not present all of this work; only enough to point out some of its relevance to analysing structure—the endeavour with which this thesis is primarily concerned.

Mereology pertains to a notion of parts, and is concerned with the ways in which entities are composed. It can be thought of as dealing with parts and wholes. This is in contrast to issues of TOPOLOGY where what is of interest is the ways in which entities relate to each other—and here the notions of boundaries and connectedness are relevant. I believe that structure, in the way we are interested in it for the purposes of this thesis, pertains to both mereology and topology.

Pianesi & Varzi[32] (and others before them) argue for an integration—but not a conflation—of mereological and topological notions when describing event structures. Although less explicit, the approach Jackendoff[22] takes is in line with this view. In the next section, Jackendoff's approach and its relevance for analysing plural and mass-like events is discussed, and some difficulties with it are highlighted. However, the basic premise that there are these two areas of concern with regard to analysing structure, is one that is used in the further development of the ideas in this thesis.

Pianesi & Varzi's approach is to use two primitives—a mereological relation of parthood and a topological notion of boundary—to characterise event structures. Their notion of separation plus integration of mereological and topological notions is one that I adopt as useful; in particular this is relevant in Section 3.2.

Plural and mass in eventualities

The analogy whereby eventualities are temporal entities in the way that objects are spatial entities (as for example described by Mayo[26] and discussed in Section 2.1.1) gives us a useful way of looking at eventualities. Many authors, such as Jackendoff[22] and Talmy[51], use the analogy to analyse repeated events as plural, and also look at what can be said about eventuality structure in terms of its relation to plural and mass in objects. In this section, some of this work is presented and one important limitation is identified.

In a paper called 'Parts and Boundaries'[22], Jackendoff also distinguishes—though not as formally—between mereological and topological notions. The principal aim of the paper is to describe a mechanism for determining conceptual structure, or meanings, from linguistic descriptions. Jackendoff proposes a two-part approach to doing this. He advocates the need for CONCEPTUAL WELL-FORMEDNESS RULES, which is a set of primitives and principles of combination that defines the well-formed expressions of conceptual structure, and a set of CORRESPONDENCE RULES, which determine the translation between conceptual structure and syntactic and morphological representation. This relies on the assumption that such a correspondence is possible; Jackendoff also says that because the syntactic side of the correspondence is language-particular, so will the correspondence rules be language-particular. So, he separates syntactic distinctions—between nouns and verbs, say—from conceptual distinctions—between objects and events, say, but stresses the importance of dealing with both these kinds of distinctions.

In his exposition, Jackendoff also leans heavily on the object-eventuality analogue. So, there is the distinction between conceptual structures and syntactic/morphological structures, and the analogy between objects and eventualities; it is this latter aspect that is relevant here. In particular, Jackendoff says [22, pg16]

(2.1) "the semantic value of repetition is identical to that of the plural, that is, it encodes the multiplicity of a number of tokens belonging to the same category."

He then goes about describing how in the case of objects, 'plural' maps an expression denoting an instance of a category onto an expression denoting a multiplicity of instances of that category. So, we have a syntactic/morphological distinction between single and plural forms in language, and a conceptual distinction between an instance of a category and a multiplicity of instances of that category, and 'plural' is the linguistic mechanism for this.

In the case of events, repetition does the same thing as plural, although the linguistic mechanism for this is a different one. Contrast '*apple*' becoming '*apples*', and '*the light flashed*' becoming '*the light flashed until dawn*' (both Jackendoff's examples). Jackendoff says that it is a morpheme that delivers the plural in the case of objects; however he claims there is a different mechanism that permits the repetitive interpretation in the flashing example.

In view of this, Jackendoff proposes that we should be looking for a system of conceptual encoding that cuts across the syntactic distinction between nouns and verbs as well as across the conceptual distinction between objects and events; he argues for a class of rules called 'rules of construal'.

We shall return to this later; for now though we focus on Jackendoff's use of the object-eventuality analogy in his view of plural and repetition. Jackendoff refers to the parallel—also pointed out by others such as Bach[4] and Talmy[51]—between the count/mass distinction in objects and the distinction between temporally bounded events and temporally unbounded processes.

For objects, Jackendoff gives the example that for a count noun, say '*apple*', one cannot divide its referent up and still get something named by the same count noun; the same is true for '*three apples*'. With a mass noun such as '*water*', part of water can also be called '*water*'.

Jackendoff applies these concepts to eventualities, using an example from Talmy[51]

(2.2) The light flashed until dawn.

According to Jackendoff, while there is a sense of something being repeated, the core sentence, '*the light flashed*', does not suggest repetition. Jackendoff continues by saying that it is also not the temporal expression (the modifier '*until dawn*') that contributes the sense of repetition, and uses as example

(2.3) Bill slept until dawn.

This does not express repeated acts of sleeping, in the way that (2.2) expresses repeated acts of flashing. For Jackendoff, the difference comes in regarding the internal structure of the interval over which the activity is taking place. Repetition is likened to the plural of actions, and Jackendoff uses the analogy to the count/mass distinction to account for the difference. One can divide up an event described by '*Bill slept*' and get smaller events describable by '*Bill slept*'. This is analogous to a mass noun. In contrast, and if we take Jackendoff's view that '*the light flashed*' is a single flashing event, this cannot be divided into smaller events describable as '*the light flashed*'. In Jackendoff's terminology, we have a single event, and this is like a count noun. Example (2.2) is a description of the activity analogue of a plural count noun.

I argue, however, that if we had first encountered example (2.2) without the modifier (as '*the light flashed*'), it is not certain that we would have chosen the reading of a single flash; in fact the sentence in that form is vague. This is made clearer when we look at (2.4).

(2.4) a. John knocked on the door.

b. John knocked on the door for a few minutes.

In (2.4a), it is easy to see a repeated knocking action, and the modifier in (2.4b) serves to distil the repetition rather than to introduce it.

I would add that if we admit that '*the light flashed*' **could** indicate repeated flashes, we would have the analogue of the ambiguity we sometimes get with plural count and mass nouns, such as '*rice*'. As Jackendoff points out, repetition is not simply lexically indicated. I maintain that verbs like '*tap, knock, flash*' are actually ambiguous with regard to repetition.

We now return to Jackendoff's argument for a broader system of conceptual encoding. He provides this in the form of two features, BOUNDED and INTERNAL STRUCTURE. Both of these features apply to both objects and events, thereby utilising the object-event analogy, and indeed cutting across both syntactic and conceptual distinctions as Jackendoff wants.

The feature BOUNDED, which encodes boundedness, is encoded as

$\underline{+b}$ for individual objects and completed events, and

$\underline{-b}$ for unbounded substances (such as bare mass nouns) and unbounded processes.

The feature INTERNAL STRUCTURE encodes plurality.

Aggregates (such as those entities expressed by plural nouns) are encoded as $\underline{+i}$ and

substances (usually expressed by mass nouns) are encoded as $\underline{-i}$.

So Jackendoff is proposing a four-way distinction which he exemplifies as follows:

- $+b - i$: individuals (a pig)
- $+b + i$: groups (a committee)
- $-b - i$: substance (water)
- $-b + i$: aggregates (buses, cattle)

In my view, a committee only has feature $+i$ because we know that committees are made of a number of people; I believe he would also put entities like some apples, or five people into such a category. Jackendoff also exemplifies his features with respect to events: '*John ran to the store*' is [+b,-i]; '*John slept*' is [-b,-i]; '*The light* *flashed until dawn*' is [+b,+i] and '*The light flashed continually*' is [-b,+i]. This feature system is then, for Jackendoff, one that cuts across conceptual categories.

Jackendoff then describes five functions that map between values of b and i, to capture the notions of plurality, composition, grinding and partitivity. The reader is referred to the original text for details of these; for the study here the four way categorisation that treats objects and events analogously (described above) is the issue of interest.

Other than this, there are two points about Jackendoff's approach to mention:

1. While plural events can be seen as a multiplicity of a number of events of the same category, this is *not* the same as repetition. That is to say, I disagree with the contention that semantically, repetition and plural are identical—cf. (2.1). This view does not allow for the possibility of a multiplicity of events, all of which happen at the same time. Take for instance the multiplicity of events referred to by (2.5). This is plural in the sense described by Jackendoff [+b, +i], yet is not covered simply by the concept of repetition.

   (2.5) In every room in the hotel, an alarm sounded at 5 am.

   A more detailed examination of the concept of multiplicity when applied to events, and its relation to notions of repetition, is called for; this will be addressed in later chapters of this thesis. In particular, notions of time and space will be included in the analysis.

2. While '*apples*' is generally considered to be a plural count noun, part of apples may still be able to be called '*apples*'. This is made clearer if we use the phrase '*some apples*'. Jackendoff is not very clear about how he deals with the difference between say '*some apples*' and '*five apples*'. I believe both of these would be [+b, +i], yet part of some apples can still be some apples; part of five apples is no longer five apples.

Finally, I note that Jackendoff's treatment of boundedness and partitivity is not in conflict with Zemach's view of mass nouns. For Zemach, mass nouns and mass verbs come from the same ontology as each other—they are what he calls pure continuants—and would be considered as distinct from either multiple events or objects; multiple instances of entities from the ontologies of processes and things respectively. These are entities that would be [-b,-i] in Jackendoff's classification. While I have some disagreement with Jackendoff on details of whether, for example, a particular sentence or a verb suggests repetition or not—and Jackendoff himself says that these details are still open to investigation for him—his feature categorisation is useful and, together with Zemach's view of mass, will form the basis of distinctions that I will make.

2.1.3 Eventuality hierarchies

In order to talk about those entities in the world that happen or exist in time, it is important to be explicit about the different kinds of such entities. Numerous authors have done this to some extent, and for a variety of purposes. In this section, I describe two such discussions—that presented by Bach[4] and that presented variously by Moens and Steedman[28, 50]; much work that has followed assumes distinctions very similar to one or the other of these approaches. The section ends with the introduction of some terms that are often used in distinguishing between kinds of eventualities.

Both approaches have started with the idea that what we must do is see what kinds of 'verbal expressions' we have, and then look at the range of things we can talk about using these. This approach is distinct from the more philosophical approaches (of Zemach and of Mayo, presented in Section 2.1.1) that focus on things in the world. That is, here we look at work that is about how language reflects the world; work that is about how we talk about the world rather than work about how the world is.

Bach's eventuality hierarchy

The first approach is that of Bach[4], and is based on Carlson's[7] but uses different terminology. Bach's distinctions are used by various authors including Dale[12] and How[18]; Figure 2.1 is a reproduction of Bach's hierarchy. This is a hierarchy of EVENTUALITIES—this term, which has now been adopted by many authors, is used in the rest of the thesis to describe any entity that happens or is a state of

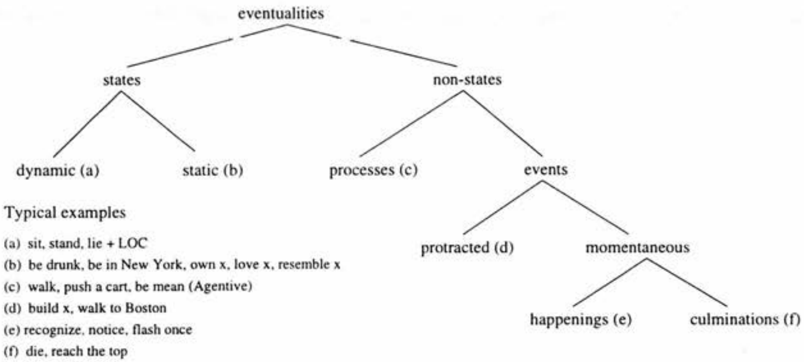


Figure 2.1: Bach's taxonomy of eventualities (from [4, pg6]).

being[4]. Bach further distinguishes between eventualities in terms of STATES and NON-STATES; non-states (which broadly are things that happen, rather than things that are states of being) are the entities of concern in this thesis.

Bach also distinguishes—in hierarchical fashion—between four kinds of non-states. These are PROCESSES, EVENTS which are protracted, and two kinds of momentaneous events—HAPPENINGS and CULMINATIONS.

Moens and Steedman's eventuality hierarchy

Moens and Steedman[28] have distinctions which are similar, but they do not present them in a hierarchy. Their distinctions are reproduced in Figure 2.2. The approach taken by them is to regard these distinctions as ones of ASPECTUAL CLASSIFICATION—in Section 2.2 we present a discussion of various views about what aspect really is. However, it is sufficient for now to accept that Moens & Steedman are distinguishing between different types of eventuality—they do not use this term, and they generally though not explicitly use the term situation to describe the entire class of such entities.

They distinguish between EVENTS—which are like the entities Bach refers to as

[4] Some authors[56, 45, 53, 18] use the term SITUATION to describe such entities, and distinguish between situation types. However, this term—situation—has also been used[50] to refer the same thing that some authors call a state.

	EVENTS		STATES
	atomic	extended	
+conseq	CULMINATION recognize, spot, win the race	CULMINATED PROCESS build a house, eat a sandwich	understand, love, know, resemble
-conseq	POINT hiccup, tap, wink	PROCESS run,swim,walk, play the piano	

Figure 2.2: Moens and Steedman's aspectual classifications (from [28, pg17]).

non-states—and STATES, but unlike Bach, they do no further distinction within the state category. In the event category, they also have four distinctions, which are not hierarchical, but are similar to those of Bach. The four categories come from two orthogonal two-way distinctions: whether the event is atomic or not, and whether the event has a consequent state or not. For Moens & Steedman[28], events such as '*recognise*' and '*wink*' are both atomic; the former has a consequent state and the latter does not. '*Build a house*' and '*play the piano*' are extended (non-atomic) with again the former having a consequent state—the house being built—and the latter not. The concept of ATOMIC relates to the granularity at which events are being perceived; for example, while a cough would usually be perceived as atomic—containing no smaller event entities, to a doctor it may be viewed as being non-atomic. In general, building a house would be seen as extended, or non-atomic; however there are circumstances where such a building event could be perceived as atomic. Dale[12] presents a similar granularity argument when discussing objects.

Using the two two-way distinctions described above, Moens & Steedman identify the categories of PROCESS, CULMINATION, CULMINATED PROCESS and POINT. Culminations and points are both atomic; culminations and culminated processes both have consequent states. They also include the concept of COERCION, where an entity from one category can be seen as coming from another when it is wrapped in language that causes an aspectual class shift. An example of this is where a culminated process (such as described by '*John played the piano for an hour*') becomes a point (say in '*John played the piano for an hour every day*'). In this case, I would say that the coercion is to do with granularity; it informs the perspective with which the entire event is viewed.

An overall perspective of eventuality hierarchies

Superficially, it seems as if the distinctions made by Bach and those made by Moens & Steedman are essentially the same. However, it is the case that if we look at the particular examples each uses to illustrate their distinctions, we see that what Bach sees as happenings ('*recognize, notice, flash once*') are variously categorised by Moens & Steedman as culminations ('*recognize*') and points (I believe they would categorize '*flash once*' as a point). Now, it is either the case that the distinctions are in fact conceptually different, or (as has been commented by various authors, including Moens & Steedman) the categories apply to eventualities and not abstract linguistic fragments; it could be that the envisioned context in each that is different, resulting in different classification of the use of the linguistic item.

If we accept the latter, then the correspondences between Bach's and Moens & Steedman's classifications are demonstrated in Figure 2.3, which also includes Vendler's[52] earlier terminology and distinctions. The cases where a distinction has not been made, or a category not identified, are indicated with a '—'. Moens & Steedman's classifications are based on those of Vendler, but using different terminology. There has been some overloading of terms—for instance, for Bach, processes and events are different kinds of non-stative eventualities, while for Moens & Steedman, a process is a kind of event.

It is interesting to compare Moens & Steedman's hierarchy to Jackendoff's[22] approach which was discussed in Section 2.1.2. Jackendoff adopts a Vendlerian[52] classification and takes the view that his processes (what he also calls 'unbounded directed situations') correspond to Vendler's activities, and his events ('bounded situations') are as Vendler's accomplishments. Jackendoff's rules of construal sometimes function in a similar way to Moens & Steedman's coercion. For Jackendoff, activities can come from accomplishments being iterated, as in '*flash repeatedly*'.

Bach	**Moens and Steedman**	**Vendler**
State	State	State
Non-state	Event	-
Culmination	Culmination	Achievement
Event : protracted	Culminated Process	Accomplishment
Happening	Point	-
Process	Process	Activity

Figure 2.3: Correspondences between different eventuality categorisations.

Jackendoff also includes three additional classes—inceptions (such as '*leave*', '*commence*' and '*start*'), duratives ('*stay*', '*keep*', '*persist*') and point-events (which are much like Moens & Steedman's points).

The terms TELIC and ATELIC are often also used when discussing eventualities; some authors also use telicity as a feature when distinguishing event types—How[18], Moltmann[29] are examples. Telic events are those in which the eventuality described has a well-defined culmination point; for atelic events this is not the case. Note that this is distinct from whether the event has a consequent state or not. (2.6a) describes an event that is telic; (2.6b) is atelic; in both cases there is a consequent state.

(2.6) a. Bring the soup to the boil.

b. Steam the fish.

This demonstration of correspondences and all the terminology is included in order to indicate where there has been confusion in the literature, and also to clarify the way in which it is used in this thesis.

The entities of interest, from the space of EVENTUALITIES, are those that Bach calls 'non-state', and Moens & Steedman call 'events'. I use the term ACTIVITY to refer to the general class of non-state eventualities. Within this, only the distinctions between 'processes' and 'events', as made by Bach, are of concern to me.

2.2 Time: temporal and aspectual concepts

Temporal issues are pertinent to this thesis in two ways. First, the thesis is about a particular kind of temporal analysis—that of internal event structure with regard to time—and so it is important to discuss how this fits in with more general issues in temporal analysis. Secondly, much has been written about the interaction between tense and aspect—aspect is important in determining the nature of an eventuality entity when trying to analyse its structure. We therefore need to examine views of aspect, and how this relates to the temporal structure of eventualities.

2.2.1 Temporal analysis

There is a range of issues of concern when we look at temporal analysis of eventualities; we have a host of mechanisms available for expressing different temporal relations between different kinds of eventualities. The only work I shall report on here is that where some reference has been made to eventualities that have the property of being extended or a multiplicity of some eventuality.

Temporal relations between situations[5]

How[18] describes a framework for temporal analysis of discourse that includes many assumptions that are relevant to the work of this thesis. His framework allows for the determination of the order (temporal) of the events and states described in a discourse. Two primary issues are of concern—the identification of situations (what are called eventualities in this thesis) and the identification of the temporal relations between these situations. How also examines the role of information about context in all of this.

One of the important contributions made by How is the proposal of an ontology for situations that includes the notions of composite and repetitive events. This comes from a discussion of the limitations of a Davidson-like representation of simple events. A Davidson-like representation of (2.7a) is shown in (2.7b).

[5] I note that I am here using the word 'situation' where I would usually use the word 'eventuality', only to be consistent with the terminology of the authors I am paraphrasing. I have already explained that these words have been both overloaded and used interchangeably in the literature.

(2.7) a. John bought the car from Mary.

b. $\exists e.\exists c$ (buying(e) $\wedge$ buyer$(e, john)$ $\wedge$ seller$(e, mary)$ $\wedge$ car(c) $\wedge$ patient(e, c))

On skolemising (2.7b), we have a unique constant, say $e1$, that represents the buying event. Information about the event $e1$ is asserted via the predicates *buyer*, *seller* and *patient*. The advantages of a Davidson-like representation are summarised (by Wilensky[56] and then by How[18]) as follows:

- A unique constant is given to an event, which allows for representation of subsequent references to the event. So, if we have a sentence '*This upset Jane*' following (2.7a), we are able to resolve '*this*' to refer to $e1$ as in (2.8).

  (2.8) $\exists e$ (upsetting(e) $\wedge$ agent$(e, e1)$ $\wedge$ patient$(e1, jane)$)

- Events and objects are both treated as instances of types. For example, in the sentence in (2.7a), the buying event is represented in the same way as the car object. This analogous treatment is useful in a variety of ways. As illustration consider the the respective representations of '*the pen in the box*' and '*the pen broke in the box*', where the predicate *in* has an object argument in the first case, and an event argument in the second.

  (2.9) a. in$(pen1, box1)$

  b. breaking$(e1)$ $\wedge$ patient$(e1, pen1)$ $\wedge$ in$(e1, box1)$

- Verbal adjuncts and verbal complements can be treated in a uniform way. The primary advantage of this is that it is not necessary to specify in advance the number of arguments a verb predicate should have, or whether some arguments are optional.

As How goes on to point out, though, there are some limitations with the Davidsonian approach. In particular, while the Davidsonian representation is adequate for simple events, problems arise when there are more complex event structures. This is particularly relevant in the context of this thesis, as we are, after all, primarily concerned with event structure. How describes the following limitations:

- We often don't know how many events we have. For a sentence like (2.10a), it is possible to view this as the conjunction of two simple events and come up with the representation in (2.10b). However, for (2.11) and (2.12), neither can be represented using simple events.

  (2.10) a. Stir the soup twice.

  b. stirring($e1$) $\wedge$ stirring($e2$)

  (2.11) Stir the soup occasionally.

  (2.12) Stir the soup every 5 minutes.

- A second problem is the one of representing sentences that contain a progressive adjunct, such as '*Bring the soup to the boil, stirring occasionally*'. If we only have available the concept of simple events, we either have to treat the entire sentence as a simple event, or treat the situation as two simple events; one of boiling and one of stirring occasionally; neither is totally satisfactory. For this and the previous problem, How proposes the solution of allowing events to consist of sub-events.

- How also includes Schubert & Hwang's[43] example, where they show the problems of representing sentences that contain quantifiers such as (2.13) using simple events.

  (2.13) Everyone looked at Mary.

  This has two problems—we don't know how many people looked at Mary, and we also want a way of being able to refer to the entire event as well as referring to each event of one person looking at Mary.

In view of these limitations, How proposes a representation that covers the following three event kinds (which he says make up the category of Generalised Physical Events, following from an analogy with Dale's notion of Generalised Physical Object[12]):

simple events These may be telic or atelic, contain an action which an agent has to carry out for the event to occur, and involve participants.

repetitions This covers a simple notion of repetition, where what is included is an event to be repeated, and a number of repetitions.

composite events These are distinct from repetitions in that their sub-events are 'explicitly indicated'—for example (2.14) is a composite event in How's scheme. These sub-events are then the constituents of the composite event.

How also allows for the combination of repetitions and composite events, where the sub-event of a repetition may be a composite event, and one of the sub-events of a composite event may be a repetition. He gives the following examples

(2.14) Bring the soup to the boil, stirring occasionally.

(2.15) Flake the salmon discarding all the bones.

and says that (2.14) is a composite of the simple event ≪*bring the soup to the boil*≫ and the repetition ≪*stir occasionally*≫; (2.15) is a repetition of the composite event of ≪*flaking the salmon*≫—a simple event—and ≪*discarding all the bones*≫—a repetition.

How specifies the relations between the sub-events of composite events; he identifies four temporal relations and four non-temporal ones; these are summarised in Table 2.3. For some reason, he groups the none relation with the non-temporal ones; I feel it is more useful covering both groups.

For repetitions, How considers only the notions of cardinality and frequency. He does not consider the full extent of temporal relations between the sub-events of a repetition in the way that he does for the sub-events of a composite event.

The major advances of How's approach that are pertinent to the work of this thesis can be summarised as follows:

- The concept of sub-events is a useful one and is central to the philosophical stance I take.
- How's approach retains the advantages of Davidsonian representation; unlike Hwang & Schubert[20], he treats objects and events analogously, and also treats verbal adjuncts and verbal complements in the same manner.

Temporal relations	Non-Temporal Relations
1. Precedes and After	5. Enables and Enabled-By
2. Includes and Included-In	6. Generates and Generated-By
3. Meets and Met-By	7. Constituent and Constituent-In
4. Simultaneous	8. None

Table 2.3: How's relations between situations (from [18, pg120]).

- How's approach also overcomes one of the disadvantages of Davidson's representation —that of an inability to deal with inexplicitly described multiple events—in a clever way. Unlike Wilensky[56], whose solution is to include complex events as part of the event ontology, How sees complex events as a perspective from which an event can be viewed.

It is the case that How's work did not intend to fully account for internal event structure in the way that this thesis does; his account of repetition and of situations that occur over time intervals is therefore limited. He does not allow for a more general case of multiple events that are provided by repetition; nor does he allow for some ambiguity that arises when composite events are described.

The tense trees mechanism

In a variety of papers [20, 43, 21], Hwang and Schubert describe an approach to dealing with tense, aspect and time adverbials, that uses what they call tense trees. Their main focus in this work is dealing with tense and temporal ordering in discourse. They are concerned with the order in which events—described in discourse—actually occur in time, with respect to each other; this is not of primary importance to my concerns, which are instead with the internal mereology and topology of structured activities. They do however also include some attempts at dealing with temporal adverbials within their approach.

The tense trees mechanism they describe can be summarised as follows: tense (for them) is seen as a sentential operator, operating on the logical form of a sentence. Tense trees provide a contextual structure within which temporal operators and modifiers can be interpreted. The tense tree is built up by rules that are associated with the sentential operators as the LFs of the sentences in a narrative text get interpreted. The event (they use the term EPISODES) tokens extracted from the LFs are appended to the tree nodes whose positions mirror the structural positions of the corresponding tense operators in the sentence.

The approach is illustrated in the following example. In (2.16), we have three sentences, each followed by the LF corresponding to it. The LF's have operators, shown in italics, acting on them—the *decl* operator comes from the fact that the sentences are declarative.

(2.16) a. John went to hospital.
(*decl* (*past* (John go to hospital)))

b. He had fallen on a patch of ice
(*decl* (*past* (*perfect* (He fall on a patch of ice))))

c. and had twisted his ankle.
(*decl* (*past* (*perfect* (He twist his ankle))))

Each LF is interpreted by calling the rule associated with the leftmost operator; this recursively calls the rule associated with the operator to its right, until there are no more operators. A tense tree for the discourse is shown in Figure 2.4. $e1$, $e2$ and $e3$ are the episodes corresponding to *John go to hospital*, *He fall on a patch of ice* and *He twist his ankle* respectively. The tokens $u1$, $u2$ and $u3$ represent the utterance events introduced by the *decl* operator, and $r2$ and $r3$ are introduced by the *past* operator, and may be thought of as the reference time for the corresponding episodes.

The positions of nodes in the tree correspond to the structural positions of sentential operators:

- Nodes A and B come from the rule associated with the *decl* operator in (2.16a); $u1$ is also created and stored at node A by this rule. The *decl*

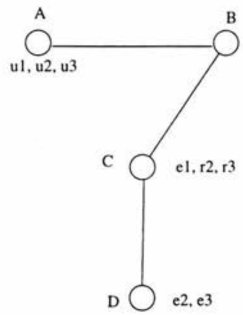


Figure 2.4: The tense tree corresponding to example (2.16).

operators in (2.16b) and (2.16c) respectively create $u2$ and $u3$ and store them at node A.

- Node C is created by the rule associated with the *past* operator in (2.16a); this rule also creates $e1$ and stores it at node C. The rules associated with the *past* operator in (2.16b) and (2.16c) respectively create and position $r2$ and $r3$.

- Node D comes from the rule associated with the *perfect* operator in (2.16b); this rule also creates $e2$ and stores it at node D. The rule associated with the *perfect* operator in (2.16c) creates $e3$ and stores it at node D.

Hwang & Schubert's approach has been criticised by How[18]. He demonstrates in particular that their method does not correctly deal with some discourses, because they use only tense information in determining which event pairs should be related. How shows that world knowledge and information from preceding text should also be used; the view of tense as a sentential operator is for How problematic—in embedded discourse, when a tense change does not provide enough information, this approach fails. In this thesis, however, I am not concerned with the temporal relations between different events in discourse.

Hwang & Schubert[21] also claim that their tense trees approach is appropriate for analysing temporal adverbials. In particular, they look at adverbials specifying repetition—which they say have received little attention—as well as adverbials of time-span and duration.

In their approach, sentences containing such adverbials describe complex events, and the adverbials modify the structure of the complex events. As in their analysis of tense, temporal adverbials are seen to operate at sentential level; it is also the case that all temporal adverbials are treated in a similar fashion. Thus, in the sentence '*Mary left yesterday*', the past tense operates on the sentence ≪*mary leave*≫, and ≪*yesterday*≫ operates on the sentence ≪*past(mary leave)*≫.

Their analysis is a two-stage process; thus for the above sentence they get (2.17) after scoping and (2.18) after deindexing the scoped form.

(2.17) (past((adv-e(during *Yesterday*)) [Mary leave]))

(2.18) ($\exists e1$: [$e1$ before $u1$]
[[[$e1$ during Yesterday] $\wedge$ [Mary leave]] ** $e1$])

This description is included to introduce the tense trees approach in general—as I noted, its primary purpose is for the analysis of tense and temporal ordering. However, there has been some discussion [21] of its use in dealing with temporal adverbials. In Section 2.3.4 I discuss in more detail the way in which their approach deals with temporal measure[6] adverbials in particular.

2.2.2 Aspect and event structure

In the area of aspect, there are two 'schools of thought' about what aspect is, resulting in two main computational treatments of aspect. In this section I summarise the basic tenets of these two different views with regard to activity structure.

Aspect as situation type

For authors such as Moens & Steedman[28], Hitzeman[16] and Passonneau[31], aspect (of a linguistic item, usually a clause) is the type of the situation underlying the linguistic item. They distinguish between situation types with respect to aspectual classification—and connect language to situation types. The situation (and therefore aspectual) distinctions of Moens & Steedman have been described in Section 2.1. They also propose a tripartite structure for activities—which as

[6] What is meant by 'measure adverbials' is discussed in Section 2.3.3.

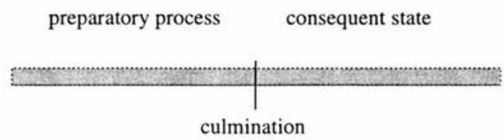


Table 2.4: Moens & Steedman's tripartite event structure (from [28, pg18]).

has been noted, they call events—which they relate to their aspectual categories. Passonneau adopts a similar structure. The event structure contains a preparatory process, which can lead up to a culmination; the culmination is an instantaneous thing which is followed by a consequent state. Their event structure, called a nucleus, is reproduced in Figure 2.4. For each of their aspectual categories (ACTIVITY, ACHIEVEMENT, etc.) they describe which parts of the nucleus are of concern.

Aspect as speaker's viewpoint

In contrast with the above, there are authors such as Singh & Singh[47] and Song[49], who believe that the structure of an event and its relationship to other events in a discourse depends on two things: its SITUATION TYPE and its ASPECT. They view situation types in a manner similar to the way Passonneau and others view aspect (as described above and in Section 2.1). However, they distinguish aspect as different from situation type—it is defined (from Smith[48]) as the viewpoint of a speaker towards a situation.

In this model then, situations are objective, but speakers choose an aspect according to what they wish to convey and what the language they are using allows them to say. Singh & Singh[47] mention the two usual aspect distinctions—PERFECTIVE and IMPERFECTIVE, with imperfective allowing for progressive or habitual—and introduce an additional one—the NEUTRAL PERFECTIVE. The perfective describes a situation as a complete whole ('*Al ate an apple*'); the imperfective covers sentences describing ongoing activity ('*Al was eating an apple*'). The neutral perfective describes an event that has ended, but not at its natural endpoint ('*Al ate an apple (but not all of it)*'). The need for this additional category is motivated by examples from Japanese and Hindi, where it is essential; there is also a claim[46] that this category is useful for giving the semantics of *for*-adverbials in English.

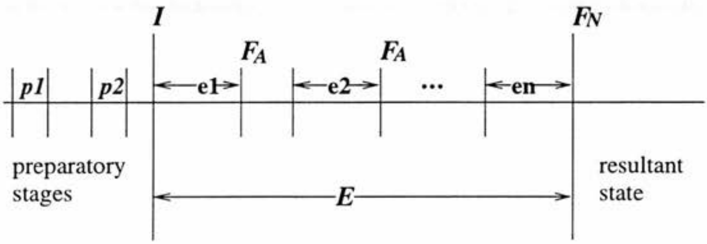


Figure 2.5: The event structure proposed by Singh (from [45, pg9]).

Singh[45] says that the tripartite event structure which is proposed by Moens & Steedman[28] has limitations. In particular, she argues for two additions: firstly that the preparatory process be distinguished from the preliminary phases of the event, and secondly that the preparatory and preliminary phases both be viewed as potentially containing sub-events. The motivation for these inclusions is the observation that there is a distinction between events that provide preconditions to a specific event and events that elaborate an event. An example is that of making a phone-call. Looking up the number is part of a preparatory process, while picking up the receiver, dialling and waiting for connection are part of the preliminary process of making the call. So, her proposal is for a more detailed structure, as shown in Figure 2.5.

Summary—situation type and aspect

It is clear (and has been noted by other authors such as How[18]) that there are two approaches to the issue of aspect. This has resulted in overloading of terminology, particularly for the term ASPECT. In this section I wish to clarify the terminology that will be used in the rest of this thesis, as well as stating which approach will be adopted.

I believe the distinction between situation type and aspect is a useful one. Seeing these issues as distinct allows us to separate those things that pertain to an event, and those things that pertain to how we talk about the event. This is in keeping with the overall stance taken in this thesis; thus the approach suggested by Song[49] and Singh & Singh[47] of aspect as speaker's viewpoint is the one that is adopted.

This approach was also adopted by How[18], whose work is relevant to this thesis as described in Section 2.2.1. Situation types are distinctions between kinds of eventualities, while aspect indicates which part of the event structure (Figure 2.5) is referred to by the language.

2.3 Language: temporal prepositions, adverbials and quantifiers

This section is concerned with research that has focused on the language that is used to describe eventualities of the type we are concerned with. It is divided into three parts, each of which deals with different language concerns. The first examines an approach that focuses on temporal prepositions, and attempts to determine what coverage of temporal functions are available through the prepositions. In the following part, two analyses of the semantics of temporal adverbials are discussed. Next, an approach that sees temporal modifiers as being part quantifiers is presented. This section is concluded with a comparison of the adverbial and quantifier approaches, which includes a note on an approach described earlier—the tense trees mechanism—and how it deals with temporal adverbials.

2.3.1 Prepositions

There has been various work done by Pratt & Brée[36, 37, 38] that focuses on prepositions as the mechanisms for delivering temporal information

In their earlier paper[36], their aim is to determine what coverage (of temporal notions) we obtain from the eighteen or so temporal prepositions they identify. This work is based on a simple notion of event and interval; the aim is to determine truth conditions for propositions (from tenseless sentences) relative to time intervals. A central premise is that temporal functions can be identified, corresponding to each temporal preposition, that perform the mapping to truth conditions. There are two general forms for such functions—existential and universal—and Pratt & Brée propose 5 groups of temporal functions that are expressed by different prepositions.

They use a Reichenbachian[7] notion of time, and see the time of utterance (TOU)

[7] Reichenbach[40] distinguished between three timepoints—the time of speech (ST), the time of

and time of reference (TOR) as significant. They then go about grouping temporal functions according to how they interact with a time interval I.

- Group-I are those temporal functions where one of the bounds on I is fixed at TOU or TOR; they are expressed using the prepositions '*since*', '*by*', '*until*', '*within*', '*in*' and '*for*'.
- Group-II functions are used to locate events, so that I is an interval straddling a point either relative or absolute to TOU or TOR; these use the propositions '*at*', '*in*', '*after*', '*before*' and '*ago*'.
- In Group-III are the temporal functions where one bound of I is either the start or end of the temporal universe of discourse; these correspond to the prepositions '*before*' and '*after*', and certain uses of '*until*', '*by*' and '*at*'.
- Group-IV contains two temporal functions that specify the endpoints of I, using '*between ... and*' and '*from ... until*'.
- Group-V contains the temporal functions where the entire reference interval, I, is given by a single argument, using '*through*', '*throughout*' or '*within*'.

For all of these groups, the temporal functions within them always belong to either of the two categories of existential or universal functions. Pratt & Brée[36] also include a group called Floating Temporal Functions, which do not belong to either category. That is, they are not used either to say that something is true over some subinterval J of a reference-interval I, or to say that something is true for all subintervals J of a reference-interval I. The floating functions are expressed using '*for*', '*within*', '*in*' and '*over*'. According to Pratt & Brée, they allow the ambiguous readings of the following examples.

(2.19) Angela was working on the letter for 1 hour.

(2.20) Charles solved the problem within 1 hour.

Although in this discussion Pratt & Brée note that in general, prepositions cannot be used together, and give the examples in (2.21) to demonstrate this, later work

the event (ET) and the reference time (RT).

of theirs[37, 38] does in fact deal with the issues involved in combining temporal adverbials, such as those in (2.22)

(2.21) a. Angela wasn't working on the letter until 10 o'clock until 10 o'clock.

b. Charles had sent me electronic mail within the last 5 minutes until 10 o'clock.

(2.22) a. In five minutes, David will have been working in his office for 6 hours.

b. Kwiksave will be open for 6 hours every Sunday until December.

In this later work, Pratt & Brée[37, 38] describe a 4-step process for translating English sentences into a temporal logic. The logic represents distinctions similar to those identified in their earlier paper[36]. The four steps are as follows:

1. Decompose the sentence into its underlying tenseless sentence and its temporal adverbials.
2. Map the components identified in 1. to data structures called operator triples.
3. Order the triples in some way, but with the triple generated by the tenseless sentence to be rightmost.
4. Fuse the list of triples to produce a formula.

So, each temporal adverbial, such as '*for 6 hours*' and '*every Sunday*' is translated to a triple; the form of the triple depends on the preposition or adverbial itself. Similarly, the tenseless sentence, such as '*Kwiksave open*' is translated to a triple, the form of which depends on the situation type of the activity. Fusing involves matching the rightmost component of a triple with the leftmost component of the triple to the right of it; distinct operators cannot be fused, identical ones can, and there is a 'wild-card' operator that fuses with anything, and resets its value to be that of the operator with which it has fused. This approach disallows combinations that do not make sense, like '*John drove to Aberdeen one day until May*'.

Pratt & Brée[38] also include a means of determining whether particular orderings of adverbials make sense, and distinguish between the sensible '*... every Monday* *for a year*' and the somewhat odd '*every Monday for a day*'; for this they use a semantic network of temporal predicates. The analysis they present covers a wide range of adverbial modifiers; however adverbials that are quantifiers ('*twice*', '*many times*') are not treated explicitly. It should be possible to incorporate these into their system of triples, but it is not clear exactly how this would best be done.

The work on prepositions by Pratt & Brée includes all of the prepositions that are applicable to extended activity structure; in particular '*until*' and '*for*' are prepositions that commonly express this. Pratt & Brée's work is thus very relevant to the work of this thesis. There are aspects of extended activity structure that I am interested in that do not come from prepositional phrases, such as quantified objects as the patient or recipient of an action. These and their interaction with those aspects that do come from prepositional phrases are discussed in depth through the thesis.

2.3.2 Verbal modifiers

Some research about natural language understanding has focused on semantic categories rather than on parts of speech in the manner described above in Section 2.3.1. In particular there has been research that attempts to account for the semantics of verbal modifiers, and research that attempts to account for the semantics of temporal adverbials. Here, I discuss three recent approaches to dealing with temporal adverbials.

Karlin's work on the semantics of verbal modifiers

Karlin's[23] discussion of the semantics of verbal modifiers provides some useful insights into ways to treat such constructs. Her analysis includes non-temporal modifiers, which are of less interest here, but deals primarily with temporal modifiers. She divides temporal modifiers into two sorts—those to do with duration and repetition, and those to do with speed. It is the modifiers of duration and repetition that are most relevant to this study, and Karlin's categorisation has informed the distinctions that are used here.

The analysis given by Karlin relies on a view of aspect like that of Moens & Steedman[28]. Based on this, for Karlin any expression belonging to the aspectual classes of points, culminations or culminated processes can have a number of repetitions associated with it. Processes, because they do not have an endpoint, cannot. Repetitions can be specified explicitly by a cardinal count adverbial or a frequency adverbial; they can also be expressed indirectly as a result of the patient being plural, having multiple parts or being a mass term. Durations, which can occur with a process or a culminated process, can be expressed in a number of ways: explicitly, by gradable terms, by a state change. In this category, Karlin also includes expression of duration that is co-extensive with the duration of another action, and gives the example

(2.23) ...fry the millet, stirring, for 5 minutes ...

Although the basic distinction between duration-type and repetition-type adverbials is useful, and Karlin presents a number of example sentences that exhibit interesting and pertinent behaviour, there are parts of the analysis that are flawed.

First, I believe that the claim that durations can occur with a culminated process is questionable.

All the examples that Karlin gives are of processes that occur with duration adverbials, and then become culminated processes; I believe that a culminated process, when occurring with a duration adverbial is actually behaving as if it is being coerced into a point, as Moens & Steedman would suggest. It seems that Karlin does not distinguish clearly enough between the event without modifier and the event with modifier. The sentences in (2.24) are all examples that Karlin considers to be in the class specified by duration.

(2.24) a. Stir for 1 minute.

b. Blend very briefly.

c. Saute over high heat until moisture is evaporated.

All of these describe processes ('*stir, blend, saute over high heat*') that become culminated processes because of the combination with a duration adverbial; none of them—without their verbal modifiers—describe culminated processes. In fact there is no evidence at all that culminated processes can be combined with duration modifiers. We do not say '*Blend very briefly for ten minutes*'; if we say '*Fry for 5* *minutes or until it is golden brown*' we are not modifying a culminated process—we are modifying the process '*fry*'.

Also discussing expressing duration, Karlin presents the example repeated in 2.25 and claims that it describes a culminated process.

(2.25) Chop the onion.

This claim is based on the notion that we know how much chopping to do through lexical knowledge about the state of an object that has been chopped; that we would do as much chopping as is required to get the onion into a state of being 'chopped'. However, I would say that there is all manner of world knowledge influencing our decision about the onion's 'choppedness'; it is also not obvious that '*chop*' and other verbs do in fact always carry the implication of a state of 'choppedness' (and therefore imply a culminated process).

Let us look at what happens if we combine a duration modifier with Karlin's notion of such a culminated process. (2.26) is an example of such a sentence.

(2.26) Chop the onions for a few minutes.

If we look closely at this, we find that it is not the case that a culminated process ≪*the chopping of the onions until they are in a state of being chopped*≫ is to be performed for a few minutes. Rather, what is happening is that a process—≪*the chopping of the onions*≫—is to occur; its endpoint is provided by the duration modifier. There is no notion whatsoever of 'choppedness' playing a role here.

Moens & Steedman do allude, as Karlin does, to the notion that some verbs may contain implied information about the endpoint of an action it describes, thereby informing the aspectual category that an event in which it participates is likely to take. However, unlike Karlin, they are very clear to say that this is not hard-and-fast, and many other factors such as context and perspective may play a part. I suggest that it is essential to incorporate this view; we can then rephrase Karlin's claim about (2.25). It is possible that this instruction means that an implied amount of chopping is to be performed; it is equally possible that the instruction means that a process of chopping is to be performed, for some unspecified duration.

In summary, Karlin's work provides useful insight into identifying the different types of activity that can be described using verbal modifiers. However, there are two problems with the approach she has developed. First, although she has adopted the Moens & Steedman model of aspectual classification, she does not always distinguish in her analysis between modified and unmodified eventualities. This results in generalisations that I have demonstrated are not appropriate, and in fact fly in the face of their proposals. Secondly, some of the analysis relies on a very particular view that certain verbs have inherent aspectual classifications; both Moens & Steedman[28] and Jackendoff[22] are less rigid about this.

Hitzeman's work on the syntax-semantic interface

The work presented by Hitzeman[16] focuses on temporal adverbials such as '*for several days*' and other temporal adverbials involving prepositions such as '*in, until, before, after*'.

In particular, Hitzeman attempts to account for the following:

1. The ambiguity in (2.27a), and the loss of the 'duration' reading in (2.27b).

   (2.27) a. They will build the bridge in ten weeks.

   b. In ten weeks they will build the bridge.

2. The interaction of multiple adverbials, and why (2.28a) is acceptable while (2.28b) is not

   (2.28) a. Rossini wrote operas in 3 weeks for 20 years.

   b. #Rossini wrote operas for 20 years in three weeks.

3. The different interactions that '*for*'-phrases have with different accomplishments; why (for her) (2.29c) is odd, but the others are not, and the different meanings of the '*for*'-phrase in (2.29a) and (2.29b).

   (2.29) a. John left the room for an hour.

   b. The student took the exam for an hour.

   c. ?They will build the bridge for ten weeks.

As a solution to the first problem, Hitzeman proposes that it is not ambiguity in the temporal adverbial that leads to ambiguity in the sentence. Rather, it is the case that in the syntax-semantics interface, when the two different parts of the semantic representation (for the sentence and for its adverbial) are combined, the adverbial can be attached at sentential or at VP-level. She argues that if attached at sentential level, the adverbial gets specific reading (as in occurring at a specific time, that of (2.27b)), while if attached at VP level, the reading is non-specific ('durative'). Because of the syntactic structure of (2.27b), only sentential level attachment is possible.

To solve the second problem, Hitzeman appeals to the notion of a generic operator. This is essentially the idea that a generic operator takes a sentence as the domain of its mapping function, and there are scope interactions with adverbials that influence the acceptability of sentences. The claim is then that the generic operator in (2.28), introduced by the presence of '*Rossini wrote operas*', is part of the reason for the difficulty arising.

For the third problem, Hitzeman proposes a recharacterization of the aspectual classes. First she develops a new set of classes that is based solely on the interaction of eventualities with temporal adverbials; then she combines these with the classic (Vendlerian) categorizations.

We return to these problems and their solutions in the latter part of Section 2.3.4 after looking at an analysis that treats measure adverbials as part quantifiers.

2.3.3 Temporal modifiers as quantifiers

A common approach to analysing temporal adverbials has been as event predicates, much in the way that other adverbials are viewed. However, some authors have taken the view that temporal measure adverbials have properties that set them apart from adverbial event predicates, and this makes them closer to being part quantifiers. In this section, I present Moltmann's[29] argument for this. It is important to note that Moltmann does not include all temporal adverbials, only measure adverbials—so '*yesterday*' and '*at noon*' are not part of the discussion. This makes sense—after all, quantification is connected with measure rather than with location. So, her analysis is distinct from those that try to account for temporal adverbials more generally [36, 16]. This distinction—between measure adverbials and other kinds of adverbials—is in my view an extremely important one.

Moltmann argues for the view that measure adverbials in general are part quantifiers. This view begins with Dowty's[14] claim—that temporal measure adverbials (such as '*for two hours; until noon*') are part quantifiers ranging over the parts of some measuring entity; Moltmann extends the claim to other measure adverbials ('*worldwide; throughout the country*') and attempts to strengthen the argument initiated by Dowty.

Measure adverbials have properties that set them apart from adverbial event predicates; these are used to support her thesis. She introduces the notion of an homogeneity requirement, that [29, pg631]

> "... the event predicate modified by a measure adverbial be homogeneous, and is satisfied, for instance, by an atelic intransitive verb ... or by a verb phrase containing bare plural or mass NPs (in certain argument positions)."

So, this explains the acceptability of the sentences in (2.30) but not those in (2.31)

(2.30) a. John slept for an hour.
b. John ate rice for an hour.
c. Throughout the garden John found flowers.

(2.31) a. #John crossed the line for an hour.
b. #John ate a bowl of rice for an hour.
c. #Throughout the garden John found a bunch of flowers.

Moltmann then shows that measure adverbials allow for some exceptions to the homogeneity requirement:

1. They are always allowed if they modify a negated clause (2.32a)

2. They allow for apparently nonhomogeneous event predicates such as vague quantifiers like '*few, often, seldom*' (2.32b)

3. They allow for nonhomogeneous event predicates if certain elements receive an interpretation dependent on the measure adverbial, such as indexicals—'*same*' in (2.32c)

(2.32) a. For an hour John did not cross the line.

b. Throughout his life John ate few bowls of rice.

c. Mary played the same minuet for several hours.

Because of the above, measure adverbials, for Moltmann, have the status of quantifiers; this is confirmed by the phenomena of scope interactions of measure adverbials with each other and with other quantifiers. Examples she gives are in (2.33) and (2.35); in (2.33b), there is no acceptable interpretation because of the syntax of the quantifiers and their scoping interactions.

(2.33) a. John listened to Mozart all the time for ten weeks.

b. #All the time John listened to Mozart for ten weeks.

I believe the syntax issue in (2.33) to which Moltmann alludes, can be seen if we translate (2.33a) and (2.33b) into logical forms (2.34a) and (2.34b) respectively, where $t \subseteq p$ represents $\ll$*t is a subinterval of p*$\gg$. It is clear in (2.34b) that there is a syntactic scope problem—in the logical form—with p.

(2.34) a. $\text{ten}(p, \text{week}(p), \text{all}(t, t \subseteq p, \text{listen}(john, mozart)))$

b. $\text{all}(t, t \subseteq p, \text{ten}(p, \text{week}(p), \text{listen}(john, mozart)))$

So, treating measure adverbials as quantifiers then means that they are subject to the same treatment as quantifiers in terms of how they interact with other measure adverbials and quantifiers. To further show the need for this, Moltmann demonstrates the shortcoming of treating such adverbials as event predicates, using the sentences in (2.35) as examples. In an event predicate treatment, where '*for several years*' is treated as predicating the complaining event, both of these sentences would have a semantic representation as in (2.36). That is, only the meaning of (2.35b) would be accounted for. However, treating '*for several years*' as a part quantifier allows both meanings to be represented, as in (2.37).

(2.35) a. For several years a lot of students complained about the requirements.

b. A lot of students complained about the requirements for several years.

(2.36) $\exists t.\exists x.\exists e$ (several-years(t) $\wedge$ lot-of-students(x) $\wedge$ complain-about($e, x, [requirements]$) $\wedge at(e,t)$)

(2.37) a. several(t, year(t),$\exists x.\exists e$(lot-of-students(x) $\wedge$ complain-about($e, x, [requirements]$) $\wedge at(e,t)$))

b. $\exists x.\exists e$(lot-of-students(x)$\wedge$ several(t,year(t), complain-about($e, x, [requirements]$) $\wedge at(e,t)$))

I have given only enough detail of Moltmann's proposal[8] to demonstrate its basic concepts and to show that it is a useful approach to dealing with the temporal adverbials that are examined in this thesis. In particular, it also connects with the object-eventuality analogy that forms the philosophical basis of the thesis; the quantifier approach is thus the one that I adopt.

2.3.4 Comparing approaches

In the previous two sections we have seen Moltmann's[29] quantifier approach and Hitzeman's[16] syntax-semantics mapping approach for dealing with temporal adverbials. We also, very briefly saw how Hwang & Schubert's[21] tense trees mechanism treats temporal adverbials. In this section, I demonstrate how Moltmann's approach is the more appropriate one for dealing with measure adverbials.

[8] I note that Moltmann[29] provides a strong motivation for the quantifier approach, but does not define a formal mechanism for it.

The limitations of the tense-trees approach

Hwang & Schubert's approach to dealing with tense and temporal ordering has been mentioned in Section 2.2.1. In this section, I focus on their approach to dealing with temporal adverbials.

A two-stage process is central to their approach. They say [21, pg 250]

> "... indexical LF is obtained ... first, by scoping ambiguously scoped quantifiers, logical connectives, and tense operators, and then by applying deindexing rules ..."

So, the example in (2.38) shows a sentence, its immediate indexical LF and its deindexed LF respectively. `adv-e` is the immediate form translation of adverbials of temporal location and durative adverbials; `adv-f` represents cardinal and frequency adverbials.

(2.38) a. Mary visited Paris *three times in two months.*

b. (past ((adv-e (in-span-of (K ((num 2) (plur month)))))
((adv-f ((num 3) (plur episode))) [Mary visit Paris])))

c. ($\exists e2$: [$e2$ before $u2$]
[[[$e2$ in-span-of (K ((num 2) (plur month)))] $\wedge$
[$e2$ ((num 3) (plur episode)) $\wedge$
(mult [Mary visit Paris])] ** $e2$])

The only scoping that is done is for quantifiers and tense—there is no mention of any other kind of scoping—and this is done before the indexical LF (which is 2.38b in the example) is obtained. Hwang & Schubert paraphrase the semantics of (2.38c) as "some time before the utterance event, there was a 2 month-long (multi-component) episode that consists of three episodes of type 'Mary visit Paris'."

This approach fixes the two modifiers '*three times*' and '*in two months*' during the syntactic analysis, and does not allow a quantifier treatment of them. That is, there is no way of representing the situation where ≪*Mary visited Paris in a two-month time period, and did this three times*≫, other than via a sentence with a different syntactic structure. Hwang & Schubert's approach does incorporate a treatment of conventional nominal quantifiers, as for example in '*Mary received an award for three years*'. They give (2.39) as the immediate indexical LF of this sentence.

(2.39) (past ((adv-e (lasts-for (K ((num 3) (plur years)))))
[Mary (*iter* $\lambda x(\exists y$: [y award] [x receive y])]))

It is not clear that in their approach it would be possible to distinguish between the situation where ≪*there is a particular award, and Mary received that particular award for three years*≫, and the equally valid reading ≪*for three years, Mary received something that is conceived of as an award*≫. This relates to the problem identified by Moltmann[29] regarding the interaction of quantified nominals and measure adverbials.

In addition, it is not clear that their approach will allow the interaction of multiple temporal (measure) adverbials in as complete a manner as is needed. They give the following example:

(2.40) a. John took medicine *every four hours for ten days.*

b. (past ((adv-e (lasts-for (K ((num 10) (plur day)))))
((adv-f λs [[s ((attr periodic) (plur episode))] ∧
[(period-of s) = (K ((num 4) (plur hour)))]])
[John take (K medicine)])))

c. ($\exists e4$: [$e4$ before $u4$]
[[[$e4$ lasts-for (K ((num 10) (plur day)))] ∧
[$e4$ ((attr periodic) (plur episode))] ∧
[(period-of $e4$) = (K((num 4) (plur hour)))] ∧
(mult [John take (K medicine)])] ** $e4$])

However, for the apparently similar example in (2.41a), the again similar indexical logical form of (2.41b) is produced—as they state, this is produced directly as a byproduct of parsing.

Deindexing rules "work their way inward" [21, pg 250], and so, we end up with the deindexed form of (2.41c). This actually makes little sense, as it is saying that the overall event, $e5$ has a duration of ten minutes, and is composed of four-hourly episodes of John taking medicine.

(2.41) a. John took medicine *every four hours for ten minutes.*

b. (past ((adv-e (lasts-for (K ((num 10) (plur minute)))))
((adv-f λs [[s ((attr periodic) (plur episode))] $\wedge$
[(period-of s) = (K ((num 4) (plur hour)))]])
[John take (K medicine)])))

c. ($\exists e5$: [$e5$ before $u5$]
[[[$e5$ lasts-for (K ((num 10) (plur minute)))] $\wedge$
[$e4$ ((attr periodic) (plur episode))] $\wedge$
[(period-of $e4$) = (K((num 4) (plur hour)))] $\wedge$
(mult [John take (K medicine)])] ** $e5$])

I note that the approach taken by Pratt & Brée[37] is not subject to the problem demonstrated above in (2.41).

Treating *in*-adverbials as quantifiers

Distinguishing between adverbials that are about measure and those that are not (and are, say, about location) is important. This is particularly true at the semantic level, and when we incorporate the concept of quantification. In the case of simple quantification over objects, this distinction is easy to accept. There is a clear difference between things like ≪*many people*≫—denoting a quantity—and ≪*people at the desk*≫—denoting a point in space. There is also a need to distinguish between phrases, with respect to eventualities, that are about quantity and those that are not.

Adverbial phrases involving the preposition '*in*', such as '*in ten weeks*' have two senses in which they can be used—as a measure adverbial and as an adverbial denoting temporal location. Let us return to Hitzeman's examples, repeated here:

(2.27) a. They will build the bridge in ten weeks.

b. In ten weeks they will build the bridge.

Hitzeman[16] claims that in (2.27a) we have the ambiguous reading[9], while in (2.27b) one reading is lost. I disagree with the claim that one reading is lost; it is certainly a dialectic use that I am happy with and would rather say we have a preferred reading. However, this is not crucially important. Instead, let us look more closely at the ambiguity issue, and focus on (2.27a).

If we take the view that measure adverbials are part quantifiers, we also need to accept that we have to distinguish semantically between the uses of *in*-adverbials as I describe above—as measure adverbials and as adverbials of location. I believe that it is the dual use of the adverbial that accounts for the ambiguity. In one case, it is being used as a measure adverbial, describing a period of ten weeks; in the other case it functions as a temporal location adverbial (much in the way '*tomorrow*' would function). The fact that both uses are delivered using an adverbial modifier is a phenomenon that is not unusual with prepositions—prepositions are well accepted as being ambiguous, and participating in ambiguous constructs. I would argue that recognising this ambiguity, **in conjunction with** a quantifier treatment of measure adverbials, is essential to developing a computational treatment of such adverbials. So, with Moltmann's analysis, in the first case it is a part quantifier, and in the second case it is not.

Hitzeman also presents the following examples:

(2.42) a. Rosa rode a horse in Ben's picture.

b. Rosa found a scratch in Ben's picture.

Here, '*in*' is functioning as a spatial adverbial; however again it is functioning as a measure adverbial in (2.42a) and as an adverbial of location in (2.42b). To be clearer, ≪*within, throughout—in all of—Ben's picture*≫ Rosa rode a horse; ≪*at a particular place in Ben's picture*≫ Rosa found a scratch. Moltmann's analysis extends to quantification over space, as well as time, and hence this ambiguity is accounted for in the same way.

---

[9] Recall that Hitzeman argues that the ambiguous reading does *not* come from ambiguity in the temporal adverbial, but rather from the way the adverbial combines with the rest of the sentence.

The interaction of multiple adverbials

I now demonstrate that the quantifier approach is also appropriate for dealing with multiple adverbials, and provides a solution that does not rely on the generic operator argument provided by Hitzeman[16].

The primary example given by Hitzeman is (2.28), reproduced here.

(2.28) a. Rossini wrote operas in 3 weeks for 20 years.

b. #Rossini wrote operas for 20 years in three weeks.

Moltmann also demonstrates a scope-interaction problem with the sentences in (2.33); this and the corresponding logical forms (2.34) are reproduced here.

(2.33) a. John listened to Mozart all the time for 10 weeks.

b. #John listened to Mozart for 10 weeks all the time.

(2.34) a. $\text{ten}(p, \text{week}(p), \text{all}(t, t \subseteq p, \text{listen}(john, mozart)))$

b. $^{**}\text{all}(t, t \subseteq p, \text{ten}(p, \text{week}(p), \text{listen}(john, mozart)))$

In the above examples, both of the modifiers are measure adverbials—'*for 10 weeks*' and '*all the time*' both describe quantities of time.

These examples are similar; Moltmann's solution uses the idea that it is scope interaction of the quantifiers causing the (b) examples to be problematic, and this solution also applies to the Rossini examples.

The other examples Hitzeman gives to support her argument for it being a generic operator that causes the difficulty are less convincing. Although her primary motivation for this is claimed to be an analysis of the interaction of multiple adverbials, she gives no other examples of multiple adverbials; all the examples given are of single adverbials. She claims that '*For an hour Mary swims*' has no acceptable interpretation; I disagree that this is the case. In addition, I claim that there are other examples where a generic interacts with multiple adverbials that do not cause problems; (2.43) and (2.44) are examples. In (2.43), although ≪*the neighbours going to Florida*≫ is a generic, we have two adverbials—one of which is a measure adverbial and the other which is not—that can interact with it.

(2.43) a. In March, for a week, the neighbours go to Florida.

b. For a week, in March, the neighbours go to Florida.

(2.44) a. The neighbours holiday in Florida in March every year.

b. The neighbours holiday in Florida every year in March.

2.4 Instructions, instruction execution and the role of context

In this section, work that relates particularly to instruction execution and visualisation is presented. This comes in three parts—first, a short section mentions various research that has focused on the analysis of natural language instructions; next is a section that discusses the role of context in understanding instructions; the last section discusses visualisation as a way of demonstrating language understanding.

2.4.1 Analysing instructions

Various authors, including Chapman[8] and Webber & Di-Eugenio[55] have commented that instructions are an under-researched area of natural language processing. Further, they are an extremely important area of natural language processing for a number of practical reasons:

- There are a variety of computational applications which make use of instructions, to varying degrees. Examples of such systems include query systems, robot systems[24, 11, 10], game-playing systems[8], systems for production.
- Machine Translation systems will sometimes need to translate language that is instructional.
- Certain syntactic and semantic constructs are unique to instructions; identifying them and dealing with them is important for the above two reasons.
- There are also phenomena that are generalisable to analysing other kinds of language; instructions provide a constrained domain within which to investigate these phenomena after which theory that has been developed can be

extended. This has been the basis of the work by Dale[12], Di-Eugenio[13] and How[18] among others.

Dale[12] comments in his thesis, which is about anaphora in the domain of cooking recipes—a special case of instructions—that the domain of cooking recipes is an interesting one for various reasons:

- It is well-defined, yet exhibits some complex phenomena.
- It is a useful domain for testing out ideas in planning.
- Issues of ontology that are fairly sophisticated can be investigated.

All of the above-mentioned researchers have dealt with instructional texts, without having anything too theoretically profound to say about instructions as such. Only Huntley[19] presents a philosophical perspective of the semantics of imperatives; much of his discussion is around whether imperative sentences should or should not be assigned truth-values. For this thesis, one important outcome of Huntley's discussion is the claim that context is crucial to the understanding of instructions.

2.4.2 Incorporating context information

Huntley[19] has noted the need for a theory of the role of context in interpreting imperatives. While it is also true that for non-instructional discourse, context can be relevant, the role played by context in interpreting instructions is very particular.

For this thesis, the most relevant work in computational linguistics that involves context is that of How[18]; this is summarised here. For How, context influences temporal analysis in a number of ways. Most simply, much of language cannot be interpreted without looking at previously mentioned situations; this is generally true of discourse. Tense can relate a situation that is being described to the time of speech, but also to a previously mentioned situation, as in (2.45).

(2.45) a. John went to the hospital.

b. He had twisted his ankle on a patch of ice.

This role of context is relatively well accepted. How notes two further roles that he says not been discussed in previous work. The first is the role of context in the interaction between resolving referring expressions and performing temporal analysis; though it is well established that interpreting referring expressions depends on context, the contribution this can make to temporal analysis is original to How. The second is the role of contextual information that is outwith the discourse; How calls this kind of extra-linguistic information the ENVIRONMENT surrounding a discourse.

The example that How gives to illustrate the role of the environment is to consider the different outcomes of executing the instructions of (2.46) with an environment that contains two cooks and two knives, and an environment that contains only one cook and one knife.

(2.46) a. Chop a carrot.
b. Chop a potato.

Though How does not say this explicitly—his discussion of execution context is with respect to activity more generally, rather than instructions in particular—I also take the view that the role of context and environment is particularly relevant to instructions and to instruction execution. Crangle & Suppes[10, 11] have noted the importance of context in interpreting commands; their work has focused on the role it has to play in determining the meanings of English words. In particular, they take the view that it is only in the context of use, and not before, that some interpretations of words can be fixed. This view is one that I adopt, and my use of it is discussed in later chapters of this thesis.

2.4.3 Visualisation

A crucial aspect of the research of this thesis is the requirement that the results of the understanding process are demonstrated in some way. The means that has been chosen is visualisation; this is clearly in line with the overall orientation of the thesis, which is to exploit the object-event analogies that have been noted in the literature. Put very simply, visualisation of eventualities provides objects—pictorial representations—corresponding to the event entities that are described by language.

Currently, visualisation is an active area of research interest, with many new proposals and systems appearing. However, it is a field that is still young and as such there is not a well-accepted body of wisdom on which to base decisions. I note also that there is a related research area—animation—which has some issues in common with that of visualisation; in fact some researchers take the view that visualisation is a special case of animation. However, issues of animation are not discussed in this thesis.

There are a few recent systems that have focused on visualising natural language meanings—in particular Pineda et al's GRAFLOG[33] and Mel'chuk et al's RITA[27] both focus on depicting spatial relationships between objects. Novak & Bulko's BEATRIX[30] is a system that can understand a combination of text and diagrams, where the diagrams help to disambiguate the text. The focus here is again on spatial relationships. Ludlow[25] presents a more comprehensive approach which deals with visualising temporal expressions as well as spatial expressions. All of these approaches make use of the idea of an intermediate representation. This was essential in RITA and GRAFLOG as they both provided two-way translation—that is, depiction of text as well as text generation from a picture. Although Ludlow is not concerned with generating text from pictures, he also uses an intermediate representation. Following Ludlow, though in some cases for different reasons, I have also chosen to use an intermediate representation between the language and the visualisation.

Ludlow's system for pictorial representation of text

The system described in Ludlow's thesis[25] is one which deals generally with the problem of representing text using pictures. His work includes some attention to depicting temporal expressions; this is the main area of relevance to this thesis.

A central tenet of the work is that it uses an intermediate representation—logical forms of the sort produced by the Core Language Engine (CLE[3]). Ludlow's work focuses on the production of pictorial representations of intermediate logical forms that have been produced by the CLE; his overall system design includes the component that translates natural language into logical forms, but he is explicit that this is simply a utilisation of the existing CLE processes.

He does however justify the use of an intermediate representation—

- it allows a modular design to be used
- it allows existing text processing research to be used
- it allows the picture generation process to use any text processing system that produces the same kind of representation

While I agree with this justification, I would go further and say that there are additional reasons for favouring an intermediate representation approach—

- it facilitates the production of picture-to-text as well as text-to-picture systems (RITA[27] and GRAFLOG[33] use this).
- it allows flexibility in the actual visualisation that is produced; for example it would be possible to choose whether to produce an animation from the intermediate representation or a graphical representation.

Ludlow describes the system he has developed, which produces pictures from logical forms representing language fragments. The bulk of his exposition is in three sections: general expressions, spatial expressions and temporal expressions. First I discuss his concept of pictorial representation. I follow this by brief comments about how he deals with general expressions as it orients his entire approach, and then discuss in some detail his approach to dealing with temporal expressions.

Ludlow defines various terms [25, pg29], of which the following are relevant to the work of my thesis:

images (also visual images) are representations of actual scenes; an abstraction of what one sees.

charts are a subset of visual images that display data; included are maps, diagrams and graphs.

visual scenes are visual images of some event or state.

icons are pictorial representations of objects.

A *pictorial representation window*, which is what his system aims to produce, is the display of a set of visual objects, and the relationships between these objects. The visual objects may be icons, or they may be other pictorial representation windows. Sometimes, icons may be introduced to represent relationships; sometimes relationships are represented by the relative positioning of visual object icons.

In general, expressions are depicted directly from their subject-verb-object (SVO) structure. So, an expression like (2.47) is depicted by three icons—visual object icons for Ian and the woman, and a relationship icon for seeing. These are displayed in a left-to-right order, reflecting the SVO structure.

(2.47) Ian saw the woman.

There are a number of issues with which Ludlow is concerned that are not of relevance to the work of this thesis. These are things like detail of icons, picture composition, representing spatial expressions, representing negation, etc, which will not be discussed here. We turn instead to representing temporal information.

For both tense and temporal expressions, Ludlow makes use of charts, which perform as adjuncts to visual scenes. For tense, a simple graph of a single axis marked '*Past*', '*Now*' and '*Future*' is used to indicate the temporal location of the depiction. For the example in (2.47), the fact that the expression is in the past tense is indicated by a pointer to the '*Past*' part of the graph; the graph is then placed below a depiction of ≪*Ian see the woman*≫.

The approach Ludlow takes to dealing with temporal expressions is similar in that he also makes use of a graph as adjunct to the visual scene. He identifies five primitives for temporal expressions—begin, end, after, before and duration. Each primitive is in the form of a function, and time relations relate the functions. A time relation is of the form

time-of-event(Begin,End,After,Before,Duration)

where

- Begin is the exact time of the beginning of the event
- End is the exact time of the completion of the event

- After
- Before
- Duration

The time relations that Ludlow presents are based on Allen's interval algebra[2]; they cover all possible permutations of values for the primitive functions. Each time relation is depicted by a time graph; Table 2.5 is a reproduction of Ludlow's pictorial representations of the time relations. The relations (Begin and After) and (End and before) are not allowed because they are contradictory. I note also that although Ludlow allows (After and Before and Duration)—as in '*for an hour between Monday and Friday*'—he does not include (Begin and End and Duration)—as in '*for an hour from 5 until 6*'. It is not clear why this relation has been omitted.

Ludlow distinguishes between *absolute time reference* and *relative time reference* by saying that the former is when there is an exact beginning or end to an event (2.48a), (2.48b) and the latter are those that indicate that the event occurred before or after some time without giving the exact time (2.49a), (2.49b).

(2.48) a. There is a meeting at three o'clock.

b. Suresh drove the car until six o'clock.

(2.49) a. There is a meeting before Saturday.

b. Claire drove the car after two o'clock.

It appears as if Ludlow is saying that those events that specify their Begin and/or End functions are absolute references, and those that specify Before and/or After are relative references. He then describes *non-specific time reference*, which can include absolute or relative references—(2.50a) and (2.50b) respectively are his examples—and reference the time of some event to another event.

(2.50) a. Sheila left on Ramajadeen.

b. Alison slapped him after his comment.

(2.51) a. Nelson will take office in three days time.

Begin and End	
Begin and Duration	
Duration and End	
Begin only	
End only	
After only	
Before only	
Before and After	
After and Duration	
Before and Duration	
Duration only	
After and Before and Duration	
Begin and Before	
After and End	
(Begin and After) **(End and Before)**	*Not allowed*

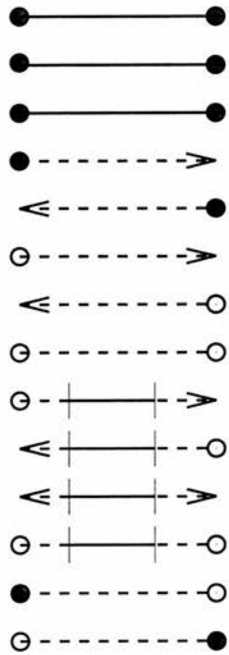


Table 2.5: Ludlow's time relations (from [25, pg136]).

b. Winnie is due to arrive now.

It is not clear how helpful these distinctions are. The examples in (2.51) would, according to Ludlow, be non-specific, absolute time reference; yet intuitively the first seems relative and the second seems absolute. Ludlow uses his distinction between absolute and relative reference to decide between using an open or closed circle in his depictions. I believe Ludlow's distinctions can be useful, but require a firmer notion of what is absolute and what is relative. For example, he is distinguishing between times that are referred to exactly ('*from two o'clock*'), and time that are referred to imprecisely ('*after two o'clock*'). I claim that this distinction is one that is different from the absolute-relative one, yet on Ludlow's scheme they are the same.

Ludlow chooses to represent temporal information using time lines; this is an approach that I have adopted. Sometimes, Ludlow also represents tense information, using an additional time line; this aspect of temporal information provides an anchor. However it is not necessary in representing instructions as I am doing since I do not incorporate tense into my analysis.

2.5 Summary

In this chapter, research work that is relevant to the various threads of this thesis has been presented. This fell into four general sections. First, work that pertains to the philosophical framework was discussed. In particular, we looked at different ways of viewing eventualities in the world, and different ways of categorising eventualities. Next, temporal issues that are relevant were presented. These were approaches to analysing temporal ordering of eventualities and the treatment of aspect. Then issues relevant to language were discussed; in particular, analysis of language that is used to convey information about the repeated or continuous execution of eventualities. Finally, research that concerns the broader context of these issues was presented—research about instruction understanding, about instruction execution and about visualisation of eventualities.

This then sets the scene for presenting the research that has contributed to this thesis, which draws on, consolidates and extends work from the areas presented in this chapter.

The main ideas that are adopted and taken forward from existing research can be summarised as follows:

- Mayo's[26] object-eventuality analogy will be used, with the exception of the notion that time should be seen as non-directional.
- Zemach's[57] view of the ontologies of entities in the world forms the basis of the philosophical stance that I will take in analysing eventualities. As well as seeing objects as being CONTINUANTS IN TIME and eventualities as being CONTINUANTS IN SPACE, Zemach's notion of masses being PURE CONTINUANTS is important. This notion applies to masses that are conventionally seen as objects (such as water) as well as to masses of activity (such as sleeping).
- The analysis of eventuality structure that is presented in the next chapter calls on notions of both MEREOLOGY and TOPOLOGY as advocated by Pianesi & Varzi[32] and Jackendoff[22] among others. I also use Jackendoff's view that these are eventuality analogues of plural and mass, although I argue that Jackendoff's assertion that repetition is like plural requires a broader interpretation than the one he gives.
- A comparison of eventuality hierarchies was presented—in particular, the ones of Moens & Steedman[28], Bach[4] and Vendler[52]; in principle the distinctions that they make will be adopted, although some clarification of terminology was required, in particular with regard to the overloading of terms like 'process', 'event' and ACTIVITY.
- How's[18] approach to temporal analysis was described; this work can be seen as the springboard for the computational approach that is described in later chapters of this thesis. How's concern is with temporal ordering between distinct eventualities, while mine is with internal eventuality structure; however I take the same view of eventualities and aspect that he does, and I see the work of this thesis as filling in some parts of eventuality analysis that How did not examine completely. Another important connection with How's work is the adoption of the notion that execution context plays a role in the understanding process—a view also proposed by Crangle & Suppes[10, 11].

- I have also described some work on language that is sometimes about eventuality structure. Karlin's[23] work in particular is about the kinds of language examples that I have used, and provides a good starting point, although I have noted some weaknesses in her analysis. Hitzeman's[16] work is also about temporal modifiers, many of which are of relevance to the work here. Again, some of the analysis she presents has difficulties. Most of the difficulties with Hitzeman's approach can be dealt with by adopting the view taken by Moltmann[29], which is that we should treat temporal—and spatial—measure adverbials as part quantifiers. I find this approach to be very useful in determining activity structure, and adopt it fully in the work that follows.

- I have also discussed the work of Hwang and Schubert[21, 20, 43], and that of Pratt & Brée[36], which also deal with temporal modifiers. These works are not beset with the problems of Hitzeman or of Karlin, and the analysis they present is useful. However, neither approach deals with the issue of the interaction of temporal modifiers and quantified objects in any depth, and so again Moltmann's approach can provide a useful extension to such approaches.

- Finally, I have described some work about visualising natural language about eventualities, in particular that of Ludlow[25], and have indicated that the approach of visualising temporal information using timelines is appropriate for the work here. I also note that this is in line with my use of the object-eventuality complementarity thesis.

Chapter 3

Conceptual Framework

The work presented in this thesis is concerned with two things—

- activity execution, and in particular continuous or multiple instances of activity in the course of such execution
- language that is used to describe such continuous or multiple activity structure.

This chapter examines some of the conceptual issues that pertain to continuous and multiple activity. We look at activities, at continuous and multiple activities, and we look at where the signals that some activity is to be continuously executed, or that there are to be multiple instances of an activity, come from. In particular, we look at the linguistic signals, and because of their intimate connection with activity, instructions have been chosen as the particular kind of language of interest. Issues to do specifically with language are discussed in detail in the next chapter.

So, this chapter is primarily concerned with conceptual issues, and we examine the notion of activity structure, and in particular that of

- continuously executed activity and multiple instances of activity
- the relation between instructions and activity, and how activity structure is expressed
- instruction execution, and how this affects how activity structure is viewed.

These conceptual issues are tightly inter-related, and each affects the others. The order in which they are discussed has been chosen to maximise clarity, but it must be noted that these concepts are not hierarchical.

The next chapter is about the actual language we use to describe activity structure; in it we will look at various linguistic constructs that signal repeated or continuous activity.

The organisation of this chapter is as follows:

- The chapter begins with a discussion in Section 3.1 about activity—which encompasses the notion that activity is made up of sub-activity—and how instructions and activity interact.
- Then, in Section 3.2, one of the central issues with which the thesis deals is presented: activity structure of the kind in which we are interested is examined. This is essentially a discussion of activity that is structured due to it being made of multiple or repeated instances of some sub-activity; or it being made of continuous execution of some sub-activity.
- Section 3.3 is a discussion of activity occurring in a context, and an investigation of how context might influence the perceived activity structure; this is in particular regard to instruction execution.
- A brief summary is presented as conclusion.

3.1 Activity and activities

This chapter begins with a discussion of activity and activities, and where they are situated in the world of objects and eventualities.

First we clarify what we see as activity entities, for the purposes of this thesis. Then we look at the connections between activities and instructions, which are essentially a particular mechanism for describing activity. This section ends with a discussion of viewing activity as composed of sub-activity—individuating activity—which then allows us to proceed with the next section which concerns activity structure.

There are two different uses of the word ACTIVITY:

- it can refer to the substance of some eventuality, such as '*Jo has been involved in some very dubious activity of late*'.
- it can refer to an eventuality entity, such as in '*The activity that Jo was busy with was very tiring*'

The distinction can also be seen as whether we talk about 'activity' or about 'an activity'. It relates to Zemach's[57] view that there is a distinction between, on the one hand

> pure continuants—water, activity, the Moonlight Sonata,

and on the other hand

> continuants in time—a glass of water or the River Thames, which are both particular instances of something that is water—or
> continuants in space—some particular instance of activity, like one rendition of the Moonlight Sonata.

The entity of most interest here is the continuant in space—this is an activity entity; importantly, it is bounded in time, which pure continuants are not. I shall use the words AN ACTIVITY or ACTIVITIES or ACTIVITY ENTITIES when referring to actual eventuality entities—continuants in space; ACTIVITY describes the substance of an eventuality—a pure continuant. The overall endeavour is to identify the structure of particular continuants in space.

3.1.1 What makes an activity?

In Chapter 2 we were introduced to the notion of eventualities and the idea that they cover the space of things that 'happen', or are 'done', or 'occur'; as such they are distinct from objects which 'exist'. Various views on eventuality classifications were presented. Specifically, we saw Bach's taxonomy of eventualities (Figure 2.1) which is a hierarchy[1] that, ultimately, distinguishes between six kinds of eventuality.

[1] Bach's taxonomy is actually a bifurcating hierarchy; however it is only the final categories that are of interest here, and so details of the hierarchical structure are not included.

The first distinction Bach makes is between states and non-states. States may be `dynamic` (sit, stand) or `static` (love, resemble); non-states may be `processes` (walk), `protracted eventualities` (walk to Boston), `happenings` (notice, flash once) and `culminations` (die, reach the top). So, non-state eventualities consist of activity, while states do not contain activity. The distinction that is being made is about whether something 'happens' or whether something 'is in a state of being'. (3.1a) describes something that happens, and (3.1b) describes something that is in a state of being.

(3.1) a. Vusi is walking to work.

b. Albertina owns a car.

For this thesis, we are interested in eventualities that are in the non-state part of Bach's hierarchy. In particular, we are interested in things that are not processes; processes describe things from the ontology of pure continuants—activity, rather than activities. Protracted events, happenings and culminations are all activities, and it is with these kinds of things that we are concerned.

Instructions are connected to activities in an important way. In (3.2a), what is actually being described is an activity, albeit one which is a passive one. Though the distinction between activities which are passive or not is important, I emphasise that a passive activity is an *activity* rather than a state, and I include it in the domain of entities with which I am concerned. It is possible of course to describe the situation using different language, and indeed convey the concept of it being a state, as is the case in (3.2b). However, in terms of instructions, it is not possible to instruct a state, other than to instruct the achievement of that state, which is then an activity. This would be the case in (3.2c).

(3.2) a. Leave it overnight to chill.

b. It is being left overnight to chill.

c. Albertina, own a car!

Activity and time

An important premise that much of the work of this thesis assumes is that all eventualities have a temporal component to them, whether this is explicit or not.

That is, an eventuality takes *time* to happen, or it occurs for a length of time. This time may be instantaneous, but nevertheless it is not zero or non-existent.

It is of course possible to view the world in a way where an instantaneous activity takes what seems to be no time. For example, in (3.3) the time between Jo not being at the top of the mountain, and then Jo reaching the top, is instantaneous. One moment she is not quite there; the next moment she is. However, at a different—closer, in this case—level of granularity, the time taken for this to happen is longer than the previous view of instantaneous.

(3.3) When Jo reached the top of the mountain, she felt exhilarated.

Now, one might argue that even at that closer granularity, there is again a point where this instantaneous change occurs; that no matter how close one gets, this point exists. This argument is the same as the one about whether an infinitely small point has any size, or whether it eventually has a size of zero. (Again, the object-eventuality analogy proves useful.) I do not believe that resolving this issue is within the remit of this thesis. I take the view that there is always a closer level of granularity, which allows us to say that every eventuality takes a (perhaps infinitesimally small) non-zero time to occur.

I believe that it is these particular infinitesimally short eventualities that come into the category that Moens & Steedman[28] and Bach call 'culminations'; taking the view that I propose, which incorporates some notion of granularity playing an important role, allows us to account for the fact that—in agreement with Moens & Steedman—on different views of an eventuality, we may classify the eventuality types differently.

3.1.2 Activity and instructions

It is important to consider the relationship between activity and instructions. Until now, we have not been very explicit about this. In Chapter 1, the notion of seeing an instruction as a recipe—or template—for an activity was introduced. Activity is something that exists in the world, and an activity occurs at a particular time and place. An instruction is an unlocated description of an activity, where the potential protagonist of the activity is the reader or agent understanding the instruction.

We have also said that instructions need to be understood in a context; this notion is further addressed in Section 3.3. How some unlocated description of activity translates into some actual activity in the world is certainly influenced by the context in which that particular instance of activity occurs. We leave this for later discussion, however, and discuss the more general issues of instruction execution being related to activity.

It is the case that one instruction can be executed—translated into an activity entity—many times; there is therefore not a simple one-to-one relationship here. It is also the case that an instruction, without it ever being executed, contains no activity at all. However, what is of great importance to this thesis is that there *is* a strong connection between instructions and activity. Instructions describe, in a fairly stereotyped way, potential activities. They do not describe objects, or states. They only describe activities. While it is true that there are some very contrived forms of instruction that can be said to describe states or objects, and instructions contain references to states or objects, they do not actually describe objects or states. It is not possible to ≪*instruct an object*≫, and ≪*instructing a state*≫ really involves instructing the *attainment* of that state, which is in fact an activity. The following examples illustrate this.

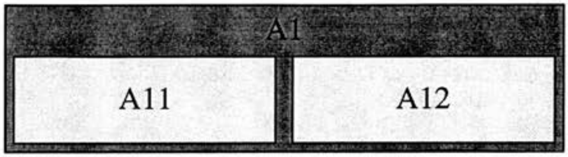
(3.4) a. Albertina, make sure you own a car.

b. Make sure you stir the soup.

In (3.4a), the actual thing being instructed is the ≪*make sure*≫; ≪*Albertina owning a car*≫ is a state. In (3.4b), '*you stir the soup*' is a description of an activity—that of someone stirring the soup—while the entire instruction is once again that of ≪*making sure*≫ some activity is executed.

Instructions as templates for activity composition

In the above section the suggestion that instructions are recipes for activity was introduced. That is, they are statements about the activities that an instruction-executing agent must perform in order to bring about required or desired changes in the state of the world.



A11 : <<pick up the pan >>

A12 : <<place the pan on the stove>>

A1 : <<pick up the pan and place it on the stove>>

Figure 3.1: The activity structure suggested by '*Pick up the pan and place it on the stove.*'

I now introduce the notion that along with instructions comes an implied way in which to view the structure or composition of the activities they instruct. There is thus an intimate connection between these two things, and the way in which the instruction is phrased will affect the way in which the activity it is instructing is 'viewed' as being composed. It is possible to conceptualise a complex activity without some linguistic description; however, the language plays the role of selecting some of the conceptualisations as more appropriate. Describing an activity by the instruction in (3.5) selects for an interpretation of there being two sub-activities—≪*picking up the pan*≫ and ≪*placing the pan on the stove*≫—making up an activity—≪*picking up the pan and placing it on the stove*≫.

(3.5) Pick up the pan and place it on the stove.

We can see this as the formation of a template, like that of Figure 3.1, which shows the activity composition suggested by the instruction. I do not claim that there actually *are* exactly these two, and only these two sub-activities; rather the language suggests that we *view* the situation as consisting of two sub-activities. Using language in a particular way allows us to communicate particular views; exploring the way this is done for particular activity structure is a central theme of this thesis.

3.1.3 Individuating activity

Now that we have established that it is activities that are the kinds of eventuality on which we wish to focus, and we have looked at how instructions and activities are related, we take a closer look at how activity entities are composed.

Crucial to this study is the approach that an activity is always itself made of activity entities—calling these sub-activities is a useful thing to do—and it is possible to *view* the activity from a number of different levels. This is a position analogous to the one of seeing objects as themselves consisting of objects; it is only the fact that we view an object from a particular perspective that makes us able to distinguish it from other objects and allows us to delimit its boundaries. For example, a dining suite may be seen as a dining suite, or as a table and six chairs, or as wood and nails and glue, or as carbon and water molecules. However it is seen though, it is still the same object. It is less usual, though certainly possible, to see the dining suite object as two chairs and some carbon molecules.

One important point to come from this discussion is that the dining table object can be seen as being composed of wood and glue and nail objects: an object is itself composed of things that can themselves be conceptualised as objects.

At this point I wish to make a distinction between GRANULARITY—which concerns the 'level' at which we conceptualise—and BOUNDING or GROUPING—which is to do with how we group things; how we draw the boundaries. This distinction is an important one, and is missed by many authors.

I illustrate this by returning to the example of the dining suite. For granularity, a physicist may be interested in the molecules, a carpenter the nails and wood, a furniture dealer the chairs and table, and so on. We may also think of this in terms of the magnification of a photograph we may take of the object; at a very high magnification, the details of the grain of the wood may be visible, while at much lower magnification it may only be possible to distinguish between one larger object and six smaller, similar objects.

Now, for bounding, the focus is instead on what sub-parts of the whole object entity we see as being closer to or further from other sub-parts of the object entity. This would for example concern whether we think of the 6 chairs as being close to each other—conceptually, or because they look visually similar, not only because they may be physically positioned close to each other—and as far from the table; whether we see the six seats as each being close to the six chair-backs in one sense, or we see the chairs as being close to each other in another sense, and quite far from the table object[2].

We now apply the same analysis to an activity—the example I use is that of dancing. Performing a dance can be seen as executing a series of dance steps, or as doing a stream of body movements. It can also be seen as the response to a series of nerve impulses. Performing a series of 32 pirouettes can be seen as a composite activity that consists of 32 sub-activities, each of which is the activity of performing one pirouette. The same issues regarding granularity and bounding are relevant here; essentially this is the issue of activity structure which is what this thesis is concerned with.

A central thread of this thesis is that language contributes very significantly to particular groupings being favoured or highlighted. Particular sentences, constructs, or even words, will suggest more strongly than others that groupings are possible or even intended. Language is not the only mechanism for this; we also look at the role of context in suggesting structure. However, the role that language plays, and its interaction with other mechanisms, is what the work of this thesis is about. The rest of this chapter is concerned with the issues regarding activity structure, while Chapter 4 examines the role of language in conveying information about such structure.

3.2 Activity structure

Using the notion that any activity is itself made of activity entities, we now look at particular kinds of activity groupings that can be seen. Of specific interest are activity entities that can be seen as consisting of multiple instances of some sub-activity, or as consisting of the continuous execution of some CORE activity. The term EXTENDED will be used to refer to any activity that is structured in this way;

[2] It is of course true that only at certain granularities can particular bounding concepts apply. If we are only distinguishing between seven objects—perhaps because we are so far away that we can only see that there are seven indistinguishable objects, or perhaps because we are looking at a very close granularity, say at the wood and glue, and are not looking at furniture shapes—the concept of grouping six of them because of some common property does not even enter the discussion.

a CORE activity then refers to a sub-activity of an extended activity.

In relation to this, we explore the notion of seeing this kind of structured activity as the eventuality analogue of plural and mass objects; this requires that we distinguish between the content—'what'—and extent—'how much'—of the activity structure.

In the next chapter we will look at language that describes such activity structures; in this section though, we are concerned with delimiting the kind of activity structure that is of interest.

3.2.1 An intuitive look at extended activities

Although this section is primarily concerned with activities, rather than with language that is about activities, the simplest way to introduce the issues of interest is through example sentences. The discussion that follows is situated in the domain of instructions[3], and in particular we look at those sentences that are instructing that some activity is to be performed either

more than once or

continuously over some time.

At this stage, the 'more than once' or 'continuously' is in the loosest sense of the word, and we attempt to identify cases where there is even a hint or suggestion of a repeated or continuous activity. We will later try to extract what it is in an instruction that gives the idea that there is to be repetition of this activity. This section examines various examples in an informal fashion.

We begin with instances where it is obvious that repetition is intended.

(3.6) a. Repeat the folding process three times.

b. Repeat the folding process on each of the sheets of pastry.

c. Break small pieces off the dough.

In (3.6a) it is clear that there is some basic activity—≪*the folding process*≫—that is to be performed a discrete number of times—≪*three*≫. A similar sense of

[3] Though the discussion is centered around instructions, it is more generally applicable to other kinds of sentences, including descriptions, and hence examples that are not instructions will occasionally be presented.

repetition is present in (3.6b) and (3.6c) as well. There is obviously something that we can see as repetition. At this stage it is not clear exactly what that something is; nor is it clear exactly what makes us believe there is repetition.

> There are two orthogonal things to consider: what is being repeated—the 'something', or the content—and how much of this repetition—the 'amount', or the extent—there is. These notions are elaborated in Section 3.2.4.

There are also cases where it is less clear that there is repetition required, yet we still have a sense of an activity being continuously executed. An example of this would be (3.7a), which is presented in contrast to (3.7b).

(3.7) a. Beat the egg until it is stiff.

b. Beat the egg 5 times.

In (3.7a), perhaps we are required to perform as many rotations of the beater—≪*beats*≫—as are required to achieve the required end state; perhaps we are required to constantly do something that is ≪*beating*≫ until the state is achieved. In (3.7b), however, it is easier to accept that there is a basic activity which is repeated five times. There are activities, such as in (3.7a), which can be either one continuous basic activity or a compound activity that is made only of a series of the same basic activity—perhaps the beating action in (3.7b).

> There are two basic ways in which there can be an *amount* of activity—continuous and discrete. This distinction is further elaborated in Section 3.2.3.

Instructions involving these continuous and discrete repeated activities are sometimes ambiguous. This is illustrated in

(3.8) Ring the bell for 5 minutes.

Here it is possible to read the sentence as requiring that the bell be rung by pressing it once and holding the button down for 5 minutes, or by performing a series of ringings over a 5 minute period. The instruction on its own is ambiguous; the context may clarify what activity is intended. Issues relating to context are discussed in Section 3.3.

As was discussed in Chapter 2, various authors including Zemach[57], Mayo[26] and Jackendoff[22], have proposed that there are analogies to be drawn between repeated and continuously executed activity, and plural count and mass nouns. I take the view that such analogies are relevant in the domain of instruction execution, and substantiate this in later sections.

We end our intuitive look at repeated activity with a more complex example.

(3.9) Slice the loaf of bread.

For this discussion, we assume that this means we want the entire loaf of bread to be cut into slices[4]. Realising this activity may consist of repeatedly executing an activity—that of cutting one slice of bread. However, each slicing action could itself consist of a repeated application of the action of moving a knife in a zig-zag fashion. On an intuitive level, it seems reasonable to consider that the first (cutting one slice) is an activity, that gets repeatedly executed over the whole loaf of bread. The second (the zig-zag movement) seems less like something that we obviously see as a part of the activity, and as repetition. We will examine reasons for this in later sections.

What becomes clear from the above is that a sentence in the abstract may be interpreted in a number of ways. It is also clear that whichever interpretation is chosen, the entire sentence describes some activity that may be made up of sub-activities, and these in turn may be made up of sub-activities. The extent to which we explicitly consider each activity's sub-activities is not fixed, and may depend upon on what we are focusing.

3.2.2 Some ontological clarifications

The purpose of this section is to clarify the use of the term SIMILAR, and the terms DISTINCT, TEMPORALLY DISTINCT and SPATIALLY DISTINCT as used in this thesis

[4] Of course, the interpretation that we want one slice cut is equally viable. In addition, if we have a bread slicer, the entire loaf may be cut in one action.

in discussing the structure of extended activities.

Activities that are SIMILAR

A notion of SIMILARITY is important in the work of this thesis. In general, when we think of plural objects, we consider them to be like each other in some sense; when we think of a mass object, each part of it is like the other parts in some sense. The work of this thesis is concerned with the eventuality analogues of these, and we are therefore concerned with eventualities that are like each other—similar to each other—or consist of some contiguous like activity.

We do not have a rigorous definition of SIMILAR. It is sufficient to say that the notion of similar used in this thesis is that it is influenced by the language, much in the way it may be with objects. If we talk about '*several chairs*', we expect them to be similar in the sense that they are all chairs, although the actual chairs may be vastly different from each other. In the same way, we may talk about '*visiting a friend three times*', where the essential activity in each of these three activities is similar—that of ≪*visiting a friend*≫. However, the details of the activity may vary in ways such as who the friend is, when and where the visit occurs, etc. Thus, we consider eventualities to be similar if the language points to this, by the use of plurals, adverbial phrases, quantifiers, etc.

Activities that are DISTINCT

By DISTINCT, I mean that it is possible to distinguish, in time or in space, between any two activity instances. That is, we can find some place—in time, or in space—within the spatio-temporal bounds of the extended activity, where the activity does not exist. As an example, in the extended activity described by '*Baste the roast three times*', there is a place (in time, and within the time that the entire three-basting activity occurs) between any two of the basting instances, where there is not a basting instance taking place. In the extended activity described by '*A bell rang in every room*', there is—in the scenario where there is a different bell for each room—a place (in space, and within the space in which the overall activity occurs) between any two of the ringing instances.

The notion of TEMPORALLY DISTINCT is related to this—two entities will be temporally distinct if they occur at different times. A similar notion of SPATIALLY DISTINCT exists. There are of course the issues regarding whether 'occurring at different times' allows overlap or not, and so on; fully resolving such questions is outwith the scope of this thesis. For purposes of the work here, because we are not crucially interested in the actual temporal size of activity entities, I say that two similar activity entities are temporally distinct if no part of their execution time is the same.

3.2.3 Plural and mass activities

Section 3.2.1 provided an intuitive introduction to the entities with which we are concerned; in this section we look more rigorously at these. In Chapter 2, work that has involved the analogies between objects and eventualities was discussed at length. In addition, Jackendoff's[22] view that repeated and continuous eventualities are the analogies of plural and mass objects, was examined.

In the intuitive discussion above, it was noted that there are two basic kinds of extended activity entities—those which are CONTINUOUS and those which are DISCRETE. Continuous activities are like those in (3.10a), where some activity is performed, continuously, for a length of time. Discretely repeated activities are like those in (3.10b), where some activity occurs more than once, and these multiple occurrences are temporally distinct. The concept of temporal distinctness may depend on the granularity with which the activity is being viewed.

(3.10) a. Chris slept for eight hours.

b. Chris visited Jo three times.

c. A bell rang in every room of the hotel.

For Jackendoff, these two kinds of activity are analogous to mass and plural objects respectively. I note however, that for this analogy to be useful, discretely repeated activity must be seen as a subclass of multiply occurring activity. This is illustrated using (3.10c)—it is clear that there are multiple instances of ≪*a bell ringing*≫, yet these activities not necessarily temporally distinct. They could all occur at the same time, and therefore the term 'repeated' is not appropriate to describe their collective structure.

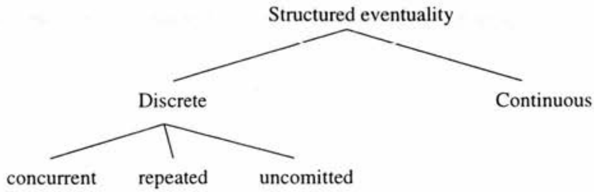


Figure 3.2: Minimal distinctions in extended eventualities.

To further clarify, in (3.10a), we have a 'mass' activity, and in (3.10b) and (3.10c), we have 'plural' activities. We describe 'plural' activities as being MULTIPLE INSTANCES of some core activity. Only (3.10b) is actually a repeated activity; in (3.10c) the activity of a bell ringing in one room of the hotel is not necessarily repeated in our conventional understanding of repetition. Thus we distinguish between multiple instances of activity that are made up of REPEATED sub-activities and multiple instances of activity that contain CONCURRENT sub-activities. Further, in (3.10c), it is not explicit that the multiply occurring activities are either concurrent or repeated; therefore we also need to be able to indicate when we don't actually know—from the language, for example—what kind of multiple activity structure we have, other than that is consists of discrete instances of some activity entity. This kind of plural eventuality has a multiple instance structure we will call UNCOMMITTED. The distinctions we have can now be summed up as shown in Figure 3.2.

With reference to the time-space analogy, what is of concern here is how we 'ground' these eventualities in time, in much the same way as we 'ground' objects in space. We are interested in temporal distinctness, or absence of distinctness, in the overall eventuality structure. The example of activity occurring at the same time but in different places is an interesting one; other than Zemach's philosophical approach there has been no attempt to encompass this into an overall theory of plural and mass in a computational framework. Allen's[2] work on temporal ordering of two events, and other related work that has followed from this, has not been concerned with multiple instances of events (though it is of course true that his approach is valid if the events under consideration are the same as each other); nor has it been particularly interested in the analogies that can be made with object positioning in space. The object analogy would involve conceiving of plural instances of an object, that occur in the same physical place; this is possible for example if we allow the time of each instance to differ—exactly analogous to allowing the space of each concurrent ringing event to differ. Imagine that we enter a room once each day, for a week. Each time we walk into the room, there is a table in exactly the same place in that room; we have seven different instances of a table, in the same physical location but at different temporal locations.

Again, the consistency of the object-eventuality analogy is evident when we consider whether or not these events (all of the bell ringings) and these objects (all of the tables) are distinct. It is possible in each case that they are the same ringing event, or the same table object. It is not of concern here how we decide whether such entities are the same or not: was the table moved out and a new one moved to replace it, or is it the same table? Is there one bell that is loud enough to be heard in all the rooms, or are there many bells that are set off by the same mechanism?

To conclude this section, I highlight two things:

- we do need to allow for multiple concurrent instances of activities;
- we can extend the object-eventuality analogy to include the notion that multiple instances of objects that are physically in the same place are analogous to multiple concurrent instances of activity.

In later chapters, we will see that there are linguistic devices that allow some of the distinctions to be made explicit.

3.2.4 Content and extent of structured activity

In Section 3.2.1, it was noted that there are two orthogonal concepts when considering complex activity structure, particularly when the structure is that of continuous or multiple execution of events. These are the CONTENT—what is being done—and the EXTENT—how much of the doing there is. We are more concerned with extent of activity; however we also do need some knowledge of activity content when determining extent, and distinguishing between these two components is important.

For example, in (3.11a), the content is the ≪*knocking on the door*≫; the extent is the fact that it happens ≪*three times*≫. In (3.11b), the content is the ≪*baking of the loaf*≫ while the extent is ≪*an hour*≫.

(3.11) a. Jo knocked on the door three times.

b. Bake the loaf for an hour.

In Section 3.2.3 we distinguished between two kinds of activity—continuous and discrete—that form the basis of structured activity of the kind in which we are interested. For this discussion here, we make use of the concept of viewing discrete activity as elements of a set; this allows some of the distinctions within discrete activity types to be made. Mass activity occurs in a contiguous fashion.

Returning to the examples in (3.11) then, (3.11a) describes repeated instances of an activity whose content is described by ≪*knocking on the door*≫. Deciding that the entire sentence describes a collection of discrete activity instances comes from considering both the core activity and the way the overall structured activity is being described. Similarly, (3.11b) describes continued, contiguous execution of an activity whose content is described by ≪*bake the loaf*≫. Again, deciding that the entire sentence describes in this case a mass activity that goes on for ≪*an hour*≫ is influenced by both the core activity and the way the entire activity is described. Very crudely, this involves the verb—'*knocking*', '*baking*'—and the modifier—'*three times*', '*for an hour*'.

Representing content and extent

For a structured activity that is discrete, we have the concept of it consisting of a set of elements. Describing a set requires two components: a structure and a substance. For (3.11a), we need to be able to say that it is a set; then we need to be able to say what kind of set it is, and what the elements of the set are. It is a set of three repeated elements; each element is one instance of ≪*Jo knocking on the door*≫.

For activity that is continuous, the whole activity forms the element. Describing a mass requires a description of its basic substance and a description of its extent. (3.11b) is a mass of an hour's worth of ≪*baking the loaf*≫.

The point I wish to stress here is that two interconnected aspects of extended activity need to be represented. Though they are interconnected, analysing a sentence in order to determine what kind of activity we have to represent requires both aspects to be considered:

- the underlying core activity, and
- the structure of the extended activity of which it forms a part.

3.2.5 Happenings and extended activities

In previous sections it has been made clear that I take the view that any activity is made of sub-activities, and that this thesis is concerned with structured activity of a particular kind, and with the ways language and other mechanisms are used to express this. We have, thus far, been mostly concerned with structure and the notion that language plays a role in describing this; only in Chapter 4 will we actually look at details of these linguistic mechanisms.

However it is appropriate at this point to discuss the coming together of language and activity structure in a general way; this is the focus of this section. At the crux of this endeavour is the wish to identify an activity structure signalled by an instruction. So, in the sentence

(3.12) Baste the roast three times.

we want to establish that there is the core activity of ≪*basting*≫, and that it happens ≪*three times*≫. The entire sentence describes an activity where this core event occurs three times. It is thus helpful to say that we can actually distinguish between two levels of event here—the basic element of the core activity (≪*basting the joint*≫) and the whole extended activity (≪*basting the joint three times*≫).

In the discussion in Section 3.1.3, it became clear that there is no single way in which an activity is composed. That is, it is not the case that the activity described by (3.12) is **necessarily** composed of three similar basting activities. It could equally be composed of two basting activities, and another activity that consists of a series of motions that involve spooning liquid over the roast. Seeing the activity as consisting of three like basting activities is just one *view* of the activity composition. However it is a reasonable composition, and is especially motivated by the language. The language refers to three eventualities and to basting the roast.

For the purposes of this thesis, no further analysis is done than that which is suggested by the language. This gives us two things:

- We limit the (infinite) number of composition permutations to the ones that come from the language. This means that we exclude, for example, the composition for (3.12) that consists of two basting activities and a collection of spooning motions. It is the case that from the language itself we may get a number of possible compositions—it is one of the tasks of this thesis to identify and isolate all of these.
- We have a way of stopping the levels of composition descending to that of nerve impulses. That is, we do **not** include the view that we have three lots of collections of spooning motions, because this is not suggested in the language. We do have '*baste*' and we do have '*three times*'.

Distinctions and terminology

At this juncture we may well ask what the point of identifying any of these is—after all identifying three bastings seems a very simple and obvious thing to do when presented with the sentence in (3.12). As suggested earlier however, this is a simple example. We have more complex activity compositions and we have more complex language examples, such as those in (3.13).

(3.13) a. Baste the roast four times every hour.

b. Cook the roast for two hours or until it is tender.

c. Peel each of the potatoes and place alongside the roast.

For describing activity structure of the sort that is of concern for the work here, I distinguish between two levels of activity. These distinctions are in the realm of how the activity is conceptualised, or described, rather than to do with actual differences in activity types[5]. The distinction that is needed is between activities that do not contain sub-activities, and activities that do.

Happenings: The term HAPPENING is used to describe an activity that does not, from the particular granularity at which we are viewing it, nor from the language that is used to describe it, have any sub-activity. Given that we wish to resolve the composition of an eventuality, described by a language fragment, into those components suggested in the language, it is the basic—discrete or continuous—eventuality we are trying to identify.

In (3.12) the happening is ≪*the roast being basted*≫; in (3.13a) it is also ≪*the roast being basted*≫.

Extended activity: I use the term EXTENDED ACTIVITY to describe an activity which consists of a composition of happenings. Note that in many examples, the components of an activity will themselves be activities, as discussed in Section 3.1.3. The need for distinguishing the separate term HAPPENING is to be able to note when a basic element has been used or encountered. It may be useful to think of happenings as the leaves of a tree and extended activities as sub-trees

(3.12) describes an extended activity, whose sub-activity is performed ≪*three times*≫; the extended activity is ≪*basting the roast three times*≫. In (3.13a), we have an extended activity of ≪*basting the roast four times*≫, which is in fact a sub-activity of the extended activity of ≪*repeatedly doing these four bastings every hour*≫.

As is often done in related literature, I use the term EVENTUALITY as a general term to refer to anything that is a state of being or doing. Thus, happenings and extended activities are themselves eventualities—the term eventuality will be used when I do not wish specifically to imply an extended activity or a happening.

So, a happening is a core activity eventuality that has no sub-activity that we wish to distinguish, and an extended activity is some composition of (one or more) happenings. As has been stated before, an extended activity may be composed of

[5] Distinctions between activity types—such as whether they consist of continuous or discretely repeated sub-activities—formed the basis of Section 3.2.3.

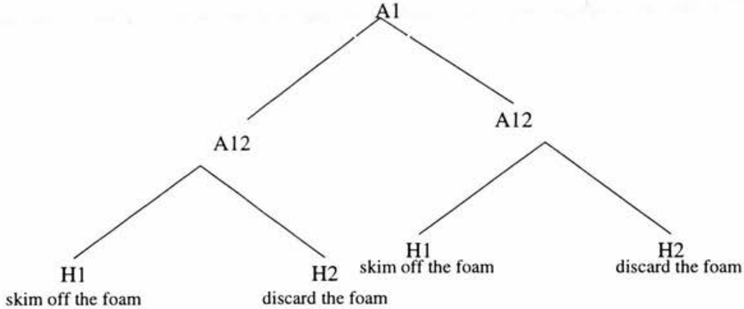


Figure 3.3: Activity composition of '*Skim off the foam and discard it twice*'.

extended activities itself. The extended activity described by '*Skim off the foam and discard it*' is composed of two happenings—one of ≪*skimming off the foam*≫ and one of ≪*discarding the foam*≫. The extended activity '*Skim off the foam and discard it twice*'[6] is composed of two extended activities—each one an extended activity of ≪*skimming off the foam and discarding it*≫. This extended activity and happening structure is depicted in Figure 3.3.

3.2.6 Different kinds of extended activity

We have seen two principal kinds of activity structuring—sets of discrete activity, and masses of continuous activity. We've also seen that it is possible to have an extended activity that itself contains extended activity entities.

For example, two or more happenings may occur concurrently in time, composing an extended activity. The extended activity described by '*halve the gooseberries*' is composed of a number of happenings—as many as there are gooseberries—each happening being the ≪*halving of one gooseberry*≫[7].

Two or more extended activities may also occur concurrently, as for example in the extended activity described by '*simmer the soup for ten minutes, stirring*

[6] This sentence is syntactically ambiguous; the example I am using is for the analysis where '*twice*' modifies the entire sentence—'*Skim off the foam and discard it*'—rather than only the clause—'*discard it*'.

[7] Thanks to Ian Sanders for pointing out to me that this could also mean ≪*divide the entire amount of gooseberries into two portions*≫.

occasionally' where there are the extended activities of ≪*simmering the soup for ten minutes*≫ and of ≪*stirring the soup occasionally*≫.

In Chapter 4, language that is used to describe extended activities is analysed.

3.3 A framework for understanding instructions

In this section, the role that context[8] plays in activity execution is investigated. The idea that execution context has a role to play was introduced in Chapter 2; in particular the work of How[18] and of Crangle & Suppes[10] was presented in support of this.

I also claim there is an even stronger role that context plays when the activity is connected to instruction execution. Basically, an instruction is an activity without a context—as soon as the instruction is to be executed, a context comes into play which will often affect the details of the activity that results from this instruction execution.

3.3.1 The role of context

When we try to examine the process of understanding instructions, it becomes apparent that it is not sufficient to consider just the instruction itself. World knowledge and execution context have different input to the understanding process.

World knowledge

It has long been accepted that world knowledge is essential if we want to cross the bridge between having a semantic representation of some entity and the entity itself. So world knowledge is what enables us to know, for example, that once the scenario described in (3.14a) has happened, the apple has been consumed, and it is no longer available; after (3.14b) the dog is in a different physical place but the road is unchanged.

(3.14) a. Laurie ate the apple.

[8] I have mentioned before that my main use of the word 'context' is with respect to the context of execution of an activity.

b. The dog crossed the road.

The need for world knowledge is equally clear when we try to understand an instruction. When we want to determine the actions required to achieve the execution of (3.15), it is important to know what it means ≪*to beat something*≫ and what it means ≪*to be eggs*≫.

(3.15) Beat the eggs.

To get a meaning for (3.15), it seems sufficient to say that when we connect ≪*beat*≫ and ≪*the eggs*≫ with entities in the world, we can say that (3.15) means we do ≪*beat*≫ to ≪*the eggs*≫. To successfully execute the instruction, we find the entity 'the eggs' and we perform the action 'beat' on the entity. It seems as if we are simply extending the process of understanding each part of the sentence to grouping the understandings into an activity. World knowledge tells us that beating requires an instrument, and so we use a beating instrument. At the end of executing the instruction, we have a world state where the eggs are beaten.

There are also other factors that will influence the process of understanding the instruction. I argue that the execution context also gives input to this, and although context is related to world knowledge, it is not the same thing.

Execution context

As argued in previous sections (Section 3.1.2), an instruction is considered to be an unlocated description of an activity. An instruction may be executed any number of times, and each execution can be seen as one instantiation of the activity described by the instructions. Each of these instantiations may occur in an entirely different set of circumstances. Context pertains to the particular set of circumstances under which the instruction is being executed.

An example of what may be part of the context is the items that are available for executing the instruction. If we have available a fork, performing the action ≪*beat*≫ means rapidly moving the fork in the eggs. If we have available an electric beater, ≪*beat*≫ means turning the beater switch to the 'on' position, with the blades in a bowl containing the eggs. At the end of the execution of the instruction, in either case we have a world state where the eggs are beaten. However, we have achieved the state via different actions, and this difference may be important if we wish to apply any of the understanding in a practical way. It may, for example, be useful to know that once an agent has switched on a beater, the agent is actually available for other activity. How[18] has used these notions of context in a system for temporal scheduling.

3.3.2 Execution of instructions

Executing an instruction involves the interaction of various components. This includes the instruction itself, the capabilities of the agent that is to execute the instruction, the resources available to the agent and the mechanisms for interpreting instructions. Figure 3.4 is a general overall schematic diagram of a model of instruction execution.

What is proposed here is a framework; in Section 7.1 we will look at how the functionality of this framework can lead to a system of interacting modules for instruction understanding. In this section, we look at the components involved in a general system of instruction understanding.

Linguistic entity

First, we have the linguistic entity, which is the instruction itself. This is text which refers implicitly or explicitly to a set of resources, and is understandable in a context, by an agent. What an instruction describes is a desired next state of the world—constraints on the manner in which this next state is achieved, or even the details of the state, may be influenced by other things in addition to the linguistic entity. Examples of linguistic entities follow:

(3.16)
- a. Make a hollandaise sauce.
- b. Put a spoonful of jam into each tart.
- c. Go to the post-office.
- d. Drive to the post office.
- e. Bake two 8″ cakes for 30 minutes.

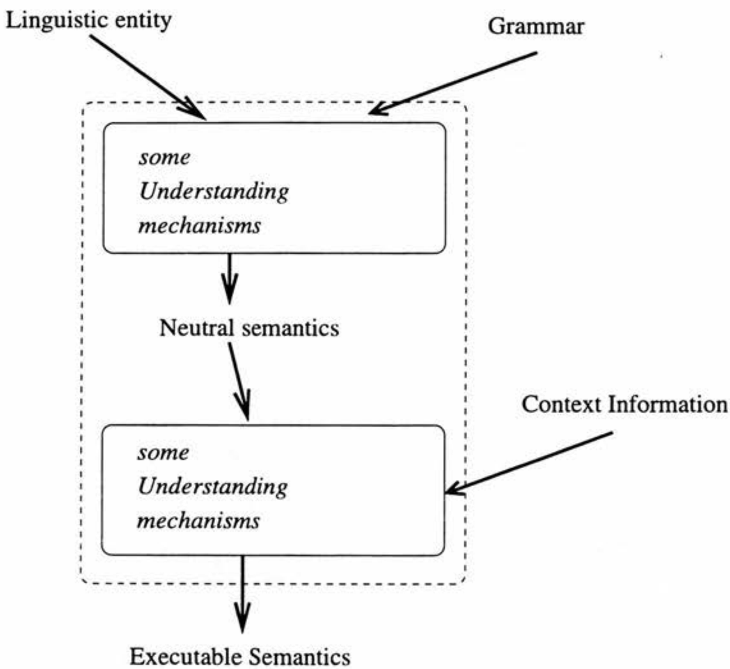


Figure 3.4: Overall system of instruction execution.

Each of these is nothing more than a sentence, which can be understood in any number of ways by any number of agents. They each contain potential for action, and the context in which they are executed (this refers to the agent doing the execution as well as the objects available for use) can affect what actions actually occur. But, in themselves they are nothing more than inanimate text.

Context information

In Section 3.3.1, a case for including context information in a framework for understanding instructions was presented. This information may be about the agent that will ultimately carry out the task, or it may be about the resources available to the agent. We view the agent as part of the context—it is indeed a resource available for the execution of the instruction. A robot with four arms will be able to execute instructions in a way different to one with one arm, and so the understanding of an instruction may be different for these two robots. Actually making use of the context information requires mechanisms; these are seen as being part of the understanding mechanisms.

The linguistic entities described above in (3.16) are instructions for the execution of certain activities—for the enabling of eventualities. These activities must be executed by an agent, and in a context or domain. The domain thus contains resources, which are objects on which the activities are to be performed, tools with which the activities can be performed and agents that are capable of performing the activities.

We can imagine resources contained in contexts as follows, which will be used in illustrative examples in later sections:

Context C: {6 tarts, a robot, a bucket of jam, a teaspoon, a soup ladle}

Context D: {a person, a bicycle, a motor-car}

Context E: {6 tarts, a robot, a bucket of jam, a 6-pronged teaspoon, a conventional single teaspoon, a soup ladle}

Context F: {6 tarts, a robot, a bucket of jam, a 2-pronged teaspoon}

Context G: {a person, one 8-inch cake tin, a bowl of batter}

Context H: {a person, two 8-inch cake tins, a bowl of batter}

Understanding mechanisms

These are the mechanisms that are available for the understanding of the linguistic entities. That is, they are the means by which a semantic representation of the instruction can be obtained.

They must therefore take account of the agent that will do the execution, and they must also take account of the context in which the execution will take place. Of course, they must be with respect to the linguistic item—I simply emphasise this point in order to present a comprehensive view of the framework.

It is possible to perform some syntactic and semantic analysis on the linguistic item, that is independent of context, from this producing what I will term a NEUTRAL SEMANTICS, which is a contextless representation of the meaning of the instruction.

Then, taking context information into account, an EXECUTABLE SEMANTICS will represent the meaning of the instruction with respect to the context in which it is executed.

The understanding mechanisms may be distributed through the understanding system; it is not the case that they need form a single entity. What is being discussed here is functionality rather than structure. In Section 7.1 we will look at how the framework proposed may be structured.

Neutral (contextless) semantics

The neutral semantics is the semantic information that can be obtained from the linguistic item alone. It is the result of syntactic analysis, combined with semantic analysis that is concerned with the meaning of the language.

There may be more than one neutral semantics for any linguistic item—either if the item is syntactically ambiguous, or if it is semantically ambiguous with respect to the language. For linguistic item (3.16e), for example, there are two possible meanings available:

- that for a time period of 30 minutes, two cakes are to have this thing called

≪*being baked*≫ done to them

- that for each of two cakes, they are to have this thing called ≪*being baked for 30 minutes*≫ done to them.

Executable (contextualised) semantics

Given a neutral semantics and a resource domain, or context, we can think of applying the context to the neutral semantics. The result of this is then some activity to be performed, which can be represented by say an animation, or by some other semantic representation. For linguistic item (3.16b) and **Context C**, this would most naturally mean the robot performing the action of putting a spoonful of jam into a tart, using the teaspoon and the jam, sequentially on each of the 6 tarts.

Instruction interpretation thus results in some representation of those actions that achieve the desired next state of the world, using the resources available in the domain. Again, it may be the case that there will be more than one such executable semantics, for the same context, if there is more than one way of achieving the result required by the instruction.

Of course, for different contexts, the likelihood of different executable semantics is high.

Mechanisms for utilising the information about the context and mechanisms for using the neutral semantics to produce an executable semantics are part of the understanding machinery. So, the instruction interpretation mechanisms are distributed through different parts of the understanding system, and interact with each other.

Some examples

In Section 3.2.5, distinctions were introduced to allow us to talk about activities in different ways, depending on how they interact to form structured activity. The terminology introduced in that section will be used here in presenting some examples of how different semantic representations of activity structures can be produced from linguistic items and contexts.

The relevant example fragments and contexts are repeated here for ease of explication.

(3.16b) Put a spoonful of jam into each tart.

(**Context C**) {6 tarts, a robot, a bucket of jam, a teaspoon, a soup ladle}

(**Context E**) {6 tarts, a robot, a bucket of jam,
a 6-pronged teaspoon, a conventional single teaspoon, a soup ladle}

(**Context F**) {6 tarts, a robot, a bucket of jam, a 2-pronged teaspoon}

Taking linguistic item (3.16b), a neutral semantics—incorporating world knowledge but not yet including context information—would represent the fact that some number of tarts each need to have a spoonful of jam put into them; that is, there should be a number of ≪*jam-putting happenings*≫—as many as there are tarts—at the end of which the state of the world is one where the tarts each have another spoon of jam inside them.

Now if we apply **Context C** to this neutral semantics, the activity structure we end up with is a set of six *repeated* ≪*jam-putting happenings*≫.

If we apply instead **Context F**, this might result in an activity structure that consists of three repeated activities, where each of these sub-activities is itself structured. The structure of each sub-activity is that it consists of two *concurrent* instances of ≪*jam-putting happenings*≫.

For **Context E**, there is a choice of tools for executing activity that requires teaspoons. Therefore, there will be more than one activity structure that is possible. One activity structure would be the same as that produced from **Context C**; another could consist of six concurrent happenings.

Of course, for all of these examples, non-standard use of the tools could result in completely different activity structure. An agent that decided to put in half-teaspoons of jam, or one that used only one spoon of the six-pronged teaspoon, would demonstrate different activity. However, we first of all assume that the linguistic item guides activity structure—if half-teaspoons are not referred to, then it is unlikely that half-teaspoons of jam will be placed into tarts. Secondly, it is the case that anyway, the activity structure template is similar for each possibility; it would just be the details of the number of half-teaspoon happenings that would change.

3.4 Summary

This chapter is the first one of the thesis that presents analysis. In it, we have examined ideas relating to activity—what structured activity entities are, and how instructions are a particular linguistic mechanism for describing activity entities. We then went on to look at extended activity structure and how this can be conceived. In particular, we looked at the eventuality analogues of plurals and masses. Finally, a discussion on how context can influence activity structure was presented, together with an overall scheme of how this interacts with instruction execution.

The following points are of importance:

- The entities of concern—activities—are from the ontology of continuants in space; these are always bounded in time. So, if we think of '*Jo slept*' in terms of this ontology, we assume an ending time exists. This is similar to the notion that when we think of '*Jo drank water*', although the water has not been explicitly delimited, we do not imagine that Jo drank all of everything that is water.

- We are interested in the structure of activities—we distinguish between activities that have no further evident sub-activities, which we call HAPPENINGS, and those that do, which are then EXTENDED ACTIVITIES.

- We distinguish between the CONTENT and EXTENT of extended activities.

- Extended activities encompass activities that are made of CONTINUOUS activity entities and activities that are made of collections of similar DISCRETE activities, as well as combinations of these.

In the next chapter, we look at the way language is used to talk about extended activities.

Chapter 4

Syntactic Analysis

How do we talk about activity of the sort described in Chapter 3? That is the purpose of this chapter—to discuss language that is used to express EXTENDED activity structure. What is ultimately of interest is the meaning that is being communicated, and we look at linguistic devices that achieve communication of the concepts that were introduced in Chapter 3.

Though the discussion is principally about the mechanisms for communicating that activity is occurring repeatedly or continuously, the context of much of the discussion is that of instruction. The links between instructions and activity as conceptual entities were noted in Section 3.1.2, and many of the examples that are used will be of instructional texts.

In this chapter, I use grammar rules when discussing some of the linguistic mechanisms. The convention I use is as follows:

1. Rules in this chapter are presented using the format
   Category ⟶Constituent1 Constituent2 ... ConstituentN
   That is, a rule consists of a left-hand side (LHS) of a single category, and a right-hand side (RHS) of one or more constituents that form that category. The constituents on the RHS are listed with spaces separating them.

   This typeface (sans serif) is used for syntactic rules which illustrate my syntactic analysis; in later chapters a different typeface (typewriter) is used for those grammar rules which come from the actual implementation. I note that all the rules presented here are realised in the implementation.

2. The category and each constituent is represented simply by its name, as a word or words without spaces—as example, the following represent two categories or constituents
   NounPhrase Sentence

I only present rules for analysing instructions; in view of this, when some of the demonstration examples are declarative sentences, the syntax trees show only the analysis of verb phrases, without subject.

The structure of this chapter is as follows:

- The next section, Section 4.1 looks at language that is special to instructions, and at language that is about time and temporal things. This is with reference to talking about extended activity.
- We then go on to look at the language used to describe continuous or repeated activity.
  1. First, in Section 4.2 we look at the underlying mechanisms for expressing extended activity in general.
  2. Then we look in turn at how protracted[1] activity is expressed—in Section 4.3—and how multiple instances of activity are expressed—Section 4.4
  3. Next, in Section 4.5 we examine how modifiers are combined to describe activity that is structured in a complex way.
  4. Then in Section 4.6, combining verb phrases—each of which describes its own structured activity—is discussed.
- Finally, in Section 4.7 some issues in discourse are mentioned and in Section 4.8 a summary of the main issues to be taken forward from this chapter is presented.

Thus the overall intention of this chapter is twofold—to discuss the various linguistic devices that are available for talking about structured activity, and to produce a

[1] The term PROTRACTED is often used in the same way as the term CONTINUOUS when applied to activity; it refers to activity that occurs contiguously for what is seen as a 'long' period of time.

list of issues that must be dealt with in the formulation of a model of understanding such language.

4.1 The language of instructions and the language of time

In this section two things are discussed—

- language constructs that are found predominantly in instructional texts, and
- language that is used to talk about time

—with respect to how they interact with describing extended activity.

4.1.1 Instructions

The research described in this thesis, and the computational system—described in Chapter 7—that was built to demonstrate some aspects of the research, have been developed using cooking recipes as the domain from which examples have been drawn. The reasons for this choice have been clarified in Chapters 1 and 3—they are primarily that there is a link between instructions and activity, and that instructions provide a rich but constrained language domain within which to investigate the issues of interest here.

Compared with prose, or with general descriptions, instructions display some unusual use of language. Like prose, recipes—a domain where instructions are prevalent—are written rather than spoken; therefore we might expect the language employed in them to be more precise than that of say dialogue or conversation. In view of this expected formality, it is interesting to see what constructs are commonly used.

Instructions usually omit the subject; that is they are imperatives. It is assumed that the subject is 'the reader'. Sometimes, instructional texts do include a subject, as in (4.1a).

(4.1) a. You place the cover in the centre of the box.

b. Place the cover in the centre of the box.

However, for this thesis, we focus on instructions that are in the form of imperatives, as in (4.1b). We take the view that instructions consist of a *verb phrase* without a subject, and we assume a syntax

Sentence ⟶Instruction
Instruction ⟶VerbPhrase(Imperative)

Tense in instructions

Instructions are intended as recipes or directions for activity. An important characteristic of instructions is that they refer to activity that will happen at some time in the future—possibly the immediate future, but nevertheless, at a time not yet reached—with regard to the time of the instruction being formulated or expressed. The activity happens at an execution time, which at the time of the instruction being expressed, can be unknown.

In general, instructions are expressed without tense. There may be some tense information, as for example in 4.2.

(4.2) After you have whipped the cream, put it aside to rest.

Although the phrase '*after you have whipped the cream*' does refer to the past tense, the main clause, '*put it aside to rest*' is untensed. That is, the primary instructional part—the imperative—is untensed. The instruction is to ≪*put the cream aside*≫. The first part, '*after you have whipped the cream*', is a temporal indicator. That is, it is telling the agent that will eventually execute the instruction something about when to execute it; just as '*to rest*' tells the agent something about why this is being done. So, it is not so much part of the instruction itself as a temporal modifier. In general, therefore, the interaction of tense will not be considered in the analysis I put forward.

Elliptical sentences

Almost all instructions have some form of ellipsis in them. A particular form of ellipsis is the omission of *articles*, which is extremely common in recipes. *Noun* *phrases* are often elided as well. Consider the discourse in 4.3:

(4.3) '*Mix eggs and tomato juice well and season. Place the bread triangles in the tomato and egg mix and leave to soak. When the bread has absorbed the liquid, fry the triangles on both sides in a non-stick frying pan to set the egg and to crisp. Pile the triangles on a plate and garnish with the parsley.*'[9, pg 89]

We can identify numerous places where constituents have been omitted, as in 4.4; however the ellipsis does not diminish our understanding of the recipe and nor does the language—in the context of it being a recipe—seem odd.

(4.4) Mix <*art*> eggs and tomato juice well and season <*np*>. Place the bread triangles in the tomato and egg mix and leave <*np*> to soak. When the bread has absorbed the liquid, fry the triangles on both sides in a non-stick frying pan to set the egg and to crisp <*np*> . Pile the triangles on a plate and garnish <*np*> with the parsley.

It is rare that verbs are omitted. This is an important, and probably obvious, point. Instructions are primarily about activity; verbs express activity. An instruction without a verb tells us nothing about the activity that is to be performed; omitting a noun or an article still allows us to construct some image of what the instruction is about, with some details missing. Without the verb, even the basic structure of the activity is missing. So, for the rest of this thesis, it is assumed that an instruction contains at least a verb, and that no verbs are elided.

4.1.2 Describing time

How activity is structured with respect to time is of central importance in the work of this thesis. In view of this, it is necessary to discuss briefly some of the language that people use to describe time.

There are two broad categories of how we talk about time: we talk about points in time, as in (4.5a), and we talk about amounts or quantities of time, as in (4.5b). Some time points have a size associated with them—for example, ≪*tomorrow*≫

indicates a point on the time axis, but it also is something that has a size of a number of hours; ≪*6 o'clock*≫ is a 'sizeless' point on the time axis.

(4.5) a. They went there at 5 o'clock.

b. They went there for 5 hours.

Characterising time

Activity occurs in time. One accepted way of understanding time requires a premise that there is one absolute time axis; it moves in one direction and in one dimension. In order to characterise activity in time, we need to know two things: the 'size' of that activity with respect to time, and where the activity is located with respect to time.

A fully characterised time entity consists of three things—a start time, an end time and a magnitude in time. From any two of these, we can obtain the third. If we have just one, we cannot without assumptions know the others. For the work presented in this thesis, because we focus on how much of an activity there is, the primary concern is to know the magnitude in time; we can get this either by knowing the magnitude or by knowing both the start time and the end time.

Talking about the time an activity occupies

We can see from examples that when people talk about activity, they give various degrees of detail about its occurrence in time. They may give no detail at all, as in (4.6a), or they may characterise it a significant amount, as in (4.6b). When detail is omitted, sometimes assumptions are made and sometimes vagueness is accepted.

(4.6) a. John slept.

b. On 25 October 1995, John slept for six hours from 11am.

In examining continuous activity in instructions, the primary concern here is with the amount of time such activity occupies. Of lesser concern, though still relevant, is locating the activity in time—after all, instructions are templates for activity and it is the execution of the instruction that locates it in time. It is certainly the case that an instruction may specify a time location as an inherent part of it, as in (4.7). When this is not specified, as for example it usually isn't in recipes, we often assume that the location is 'now'—that is, whenever the instructions are being executed and the current instruction is the next one to be executed.

(4.7) At 8 o'clock today, switch off the lights.

Of course, context—both linguistic context and execution context—can also give information about temporal location. In Section 4.4, where we discuss descriptions of repeated activity, we will look at language that *does* inform the time location of the activity.

4.2 Using language to express extended activity

Repeated or continuously executed activity can be indicated, in part at least, by language. Although, as mentioned in Section 3.3.2, the execution context may play some part in our interpreting an activity as being extended, we are for now interested in how language is used to indicate this. There are various ways in which we express that some activity is to be continuously executed. It may be done via the language about the activity itself, or it may be from the language about how the activity in which the action is embedded is to be achieved. In the next sections of this chapter we will look in turn at particular language constructs that express particular kinds of extended activity structures. In this section, however, we look at how extended activity structure is expressed in general.

There are two principal places where extended activity structure is indicated—

- in the verb itself
- in the verb complements and verb phrase modifiers

Syntactically, we have grammar rules of the form
VerbPhrase ⟶Verb Complement1 . . . ComplementN
VerbPhrase ⟶VerbPhrase Modifier1 . . . ModifierM
Extended activity structure can be expressed via the Verb, or via the complements or modifiers contributing to the VerbPhrase.

In addition, the interaction of these two may also cause some indication of extended activity. These claims are not new; they have been made by a variety of authors with a variety of views as to how the interaction occurs ([22, 50, 28, 23]), as discussed in Sections 2.1.2 and 2.1.3. This section is presented to clarify the particular approach taken in this thesis.

4.2.1 Different verb kinds

The above claims rely on the view that there are some activities that will inherently be seen as being MASS-LIKE—that is, that some verbs carry with them some sense of protracted, or continuous, execution—and some activities are COUNT-LIKE—with some sense of repeated execution[2]. It can be argued, for each verb, that there is a stronger or weaker sense of this—however, the basic premise is that such a range of distinctions does exist. For example, '*sleep*' could be seen as a verb that is inherently mass-like, while '*yawn*' is a verb that is count-like; '*leave*' has no suggestion of either continuous or repeated execution. This distinction can perhaps more easily be seen if the verbs are used in a sentence, as in 4.8.

(4.8) a. Thandi slept for half an hour.

b. Thandi yawned for half an hour.

c. Thandi left for half an hour.

In (4.8a), it is more likely that we will imagine a continuous, uninterrupted episode of sleeping, akin to a mass object, while in (4.8b) we are more likely to imagine repeated yawns, analogous to plural objects. This then indicates that '*yawn*' on its own is count-like, while '*sleep*' on its own is mass-like, with the '*for half an hour*' serving to delimit the magnitude of the extent. Again returning to the analogy with objects, this is like delimiting the extent of a mass object as in '*a bucket of water*'. In (4.8c), we imagine that the eventuality of ≪*Thandi leaving*≫ occurred once, and we do not imagine that it took half an hour to happen, but rather that she remained in the state of ≪*having left*≫ for this time; the ≪*leaving*≫ we will

[2] This is analogous to the view that some nouns describe objects that are inherently plural or mass—'*rice*' and '*water*' are examples of plural and mass objects respectively, while '*chair*' does not have any inherent suggestion of either.

expect to have taken a far shorter period of time. That is, it is a different use of '*for half an hour*'[3] to the use in (4.8a) and (4.8b).

Categorising verbs into one of 'mass-like', or 'count-like', or whatever, is not an entirely straightforward activity; it is certainly the case that there are verbs that have been viewed as one way by some authors and another by others—as demonstrated in Section 2.1.3. In addition different people have different intuitions about some verbs. Moens & Steedman[28], Jackendoff[22] and others note that their categorisations apply to *uses* of verbs, rather than raw lexical entities. However, I believe that the mechanism I have described above is a useful one for distinguishing in principle between verb kinds.

It is possible to imagine a scenario where (4.8b) does refer to one half-hour long yawn, or where (4.8a) describes repeated episodes of sleeping, all taking place within a half hour. However, it is useful and not unreasonable to claim that there will usually be a *preferred* view, and that some information about inherent 'massness' is available from the verb itself. This then means that such information can be part of the lexicon; what must also be part of any use of the lexicon is some information about what of this lexical information can be overridden, in what circumstances it can be overridden, and in what manner it can be overridden. Details of this—for the language of concern and the computational system developed to illustrate these ideas—are described in Chapters 5, 6 and 7.

4.2.2 Language outwith the verb

A second source of indication about extended activity is the language around the verb itself. Here we find language that explicitly indicates that some activity is repeated, or is continuously executed over some protracted period of time, as well as language where such indication is less explicit. (4.9a) is an example of language that does not explicitly describe a repetition structure, while (4.9b) is a clear example of explicit repetition of activity.

(4.9) a. Peel the carrots.

b. Stir the soup three times.

[3] In Section 4.3.2, these different uses of '*for*' are discussed further.

In (4.9a), it is possible to see the instruction as describing an activity of ≪*peeling a carrot*≫[4], which is repeated as many times as there are carrots. However, this repetitive structure is not essential; it is also possible to imagine, for example, that there is some device that could peel all the carrots simultaneously—that is, the execution context could override the suggestion of repetition. It is also the case that world knowledge could distill, or deny, a repetition structure, as exemplified in Section 4.4.5.

In (4.9b) however, it is explicit that there are three episodes of ≪*stirring the soup*≫. That is, in this sentence, the language itself insists that the activity is repeated—nothing can override this. So, I distinguish between extended activity that is INSISTED on by the language, and extended activity that is SUGGESTED by the language.

Outwith the verb there is a range of language mechanisms that are used to describe activity structure. I list those that are relevant to the work of this thesis, together with the sections in which they are discussed.

- Adverbial phrases—Sections 4.4.2, 4.4.3, 4.4.4, 4.5
- Prepositional phrases (used adverbially)—Sections 4.3.2, 4.3.3 and 4.5
- Plural and quantified objects—Sections 4.4.4, 4.4.5 and 4.5.1
- Sentence complements—Sections 4.3.3 and 4.5
- Combining sentences with infinitives—Section 4.6

4.2.3 Summary

In summary, this section has disussed the notion that there are two primary places where language may describe extended execution of activity—within and outwith the verb itself. In addition there is some information that comes from the interaction between these two; this is discussed in more detail in later sections of this chapter as well as in the next chapter. The language outwith the verb may itself indicate this extended activity INSISTENTLY via explicit constructs—such as

[4] Here, I am not discussing whether the activity of ≪*peeling a carrot*≫ is itself extended.

adverbials—or SUGGESTIVELY or implicitly—for example through the use of plural objects in the verb phrase.

In the two sections that follow, we will examine language—other than the verb—that describes the two kinds of extended activity with which this thesis is concerned: continuous activity and multiple instances of activity; the eventuality analogues of mass and plural objects respectively. Then we will go on to examine ways of talking about more complex activity structures.

4.3 The language of continuous activity

In this section, we look at language that describes activity that is continuously executed, for some protracted period of time. In doing this, we shall focus on the 'extra-verbal' language. However, when appropriate, some note will be made of the interaction between such language and the verb itself; I also note that as examples, I will usually choose verbs that, very loosely, contain suggestions of mass-like activity of the sort discussed in Section 4.2.1.

In general, then, we look at verbal modifiers that express that some activity is to be performed for some period of time. The constructs we focus on are those that use '*for*' and '*until*', since they are the ones most commonly found in recipes[5]. Where appropriate, however, we generalise to other prepositions or complementisers.

As has been noted in previous chapters, there are strong parallels that can be drawn between how we view objects and how we view activities. Here, we make use of the notion that a piece of continuous activity is in many ways analogous to a mass object. We have some mass of activity and we want to be able to express how much of the activity there is, in a similar way to having, say, some water and wanting to express how much water we have. This is illustrated by the sentences in 4.10:

(4.10) a. John slept.

[5] Pratt & Brée[36] identify eighteen temporal prepositions—'*until*', '*since*', '*by*', '*during*', '*over*', '*for*', '*in*', '*within*', '*at*', '*on*', '*from*', '*between*', '*through*', '*throughout*', '*upto*', '*before*', '*after*' and '*ago*'. Some of these ('*at*', '*in*', for example) do not express extended execution at all, while others ('*throughout*', '*upto*', etc) function in a similar way to '*until*' or '*for*'. I note that prepositions like '*between*' have not been accounted for in this thesis.

b. John slept for eight hours.

c. John slept until he was rested.

Although the analogy between objects and activities is a useful one, there are some places where it is less direct. Relevant to the current discussion is that when we talk about 'how much' of a mass object we have, there are a number of ways to measure it. We can measure its weight or its physical dimensions, or its extent in space. The two are directly related to each other, via the context and physical properties of the object itself, but we talk about both equally. For activity, we may measure how much in terms of time[6]; it is this that I refer to as EXTENT of activity, a notion introduced in Section 3.2.4. We may talk about this time explicitly, as in (4.10b), or implicitly, as in (4.10c). In (4.10c) there is a state of the world, that in a sense we are 'waiting for'; it occurs at a particular, maybe vaguely specified, time.

4.3.1 Specifying the duration of activity

We now focus on the language devices that we use to express the magnitude in time of a mass of activity, where that activity occurs as a connected entity in time. That is, we are looking at language describing the temporal extent of a protracted activity that is not necessarily temporally located.

There are two principal ways of doing this—explicitly, by stating how much time that activity occupies, and implicitly, by expressing a state of the world or an event that delimits an end point of the activity. We may be given information about the temporal location of the activity, particularly when the mechanism delivering the extent is an implicit one. However, this aspect will not be discussed until Section 4.4.

Prepositional phrases involving '*for*', together with some time duration, are one of the principal mechanisms for explicitly delimiting continuous activity, and in Section 4.3.2 these are examined in some detail. '*Until*' can be used—with explicit reference to time—for this purpose as well, although it is also often used together

[6] I note that it is possible to see the 'how-muchness' of an activity as pertaining to how much physical space it takes for the activity to be executed. However, this aspect of measure will not be explored here at all.

with a description of a state. These uses are discussed in Section 4.3.3. There are other prepositions and complementisers that are used for expressing protracted activity, and these will be mentioned in relation to the discussion of these two principal devices.

4.3.2 Using '*for*' to specify duration

English dictionaries describe upwards of 19 different uses of '*for*' in forming prepositional phrases. Those that pertain to quantity, taken from Collins, are listed below.

1. used after words such as 'time', 'space', 'money' or 'energy' when indicating how much there is, as in '*there is enough time for you to ...* ' or '*I have enough money for ...*' or '*he didn't have the concentration required for completing the crossword* '.
2. to indicate that something lasts for a particular time, as in '*I have known you for a long time*' or '*he stayed for three days*'
3. to indicate how far something stretches, as in '*we drove on for a few miles*'.
4. to indicate how often something happens, in particular use, as in '*for the first time ...* ' or '*for the last time ...*'

This use of '*for*' is often seen as forming modifiers of MEASURE, or MEASURE ADVERBIALS[29]—what they describe is 'how much' there is, in time or in space, of the thing they are modifying[7]. In this discussion we focus on those whose quantity is one of time, which are referred to as TEMPORAL MEASURE ADVERBIALS. Because of the analogies that can be drawn between objects in space and events in time, there will be similar stories to tell about distance and other resources, but these are not within the scope of this thesis.

[7] This is in contrast, for example, to the sentence '*They went south for their holiday*', which describes a reason rather than an amount.

Different senses of '*for*'

There are a number of different senses in which '*for*' can be used as a preposition in a temporal measure modifier. This interacts with the nature of the action itself—≪*travelling to London*≫ in (4.11b)—that the prepositional phrase is modifying.

(4.11) a. He went to London for ten days.

b. He travelled to London for ten days.

c. He stayed in London for ten days.

In (4.11a) it is the being in London—that is, the time *after* the ≪*going*≫ activity—to which ≪*for ten days*≫ is being applied[8]. In (4.11b), it is the process of going to London—that is, the time *of* the event—that ≪*for ten days*≫ is performing adjunct to. The sense of (4.11c) is similar to that of (4.11a)—that is, that ≪*he went to London and stayed there for ten days*≫.

In (4.11b), it is the ≪*travelling*≫ that occupies ten days, and in (4.11c), it is the ≪*staying*≫ that occupies ten days. However, it is not the ≪*'wenting'*≫ in (4.11a) that occupies ten days, but rather the state that occurred immediately after the ≪*'wenting'*≫ that exists for ten days.

Although (4.11a) has a semantic sense that is very similar to that of (4.11c), it is not within the scope of this thesis to identify this similarity. Furthermore, my interest is in how much of an activity we have, rather than the length of time for which any state exists[9]. So, for the purposes of my overall focus, I omit sentences like (4.11a) which are semantically more akin to describing states.

Sleeping for days

Using the view of repeated actions and their relation to mass and count concepts, we look at the following use of '*for*':

(4.12) He slept for five days.

[8] Moens & Steedman[28] have noted this use of '*for*' which is often ignored by other authors.

[9] I note though that there are connections between these two aspects of eventualities.

According to the analysis presented in Section 2.1.2, this sentence describes a mass of sleeping, that occurred for a duration of five days. Jackendoff[22] would claim that it is the main activity, ≪*sleeping*≫, that gives the massness.

However, the description in (4.13) appears to be only slightly different:

(4.13) He slept once a day for five days.

Here, it seems as if the sleeping activity is no longer a mass, but instead a discrete action, that has a start and end, but is repeated on a daily basis for five days. We now have a plural activity, of the count type[10].

I argue that the '*for*' tells us that the activity is an extended one—that is, it may be protracted, or may contain multiple instances of some activity—but it is the details of the action itself that tell us what kind of extended activity it is, namely mass or count. This further supports the view described in Section 4.2.1 that we need to examine the activity or *use* of the verb, rather than just the verb itself. In general, prepositional phrases involving '*for*' describe protracted, mass-like events; this is the default, but it can be overridden in the way suggested above. Also of relevance is the use of multiple modifiers—in this case '*once a day*' and '*for five days*'—and the ways in which they can be combined; this is discussed further in Section 4.5.1.

4.3.3 Using '*until*' to express duration

'*Until*' can function syntactically as either a *complementiser*[11] or as a *preposition*; because of this I present a brief discussion of the different syntactic ways it is used and the implications of this dual functionality.

We have the following contrasting definitions

- From the Oxford dictionary:

[10] The analysis that Moens & Steedman's[28] would give is that a process—≪*sleep*≫—has been coerced into a point—≪*sleep once a day*≫—and then becomes an iterated process—≪*for five days*≫.

[11] The complementiser analysis is found in Burton-Roberts[6, pp178–180] as well as in Radford[39, pp136].

"Until: prep & conj = TILL Used esp. when beginning a sentence and in formal style, e.g. '*until you told me I had no idea; he resided there until his decease*' [orig. northern ME '*untill*' f. ON '*und*' as far as + TILL"

- From the Chambers dictionary:

  "Until: conj. 1. up to (a time) that '*he laughed until he cried*'. 2. (used with a negative) before (a time or event) '*until you change, you can't go out*'.—prep 3. in or throughout the period before '*he waited until six*'. 4. (used with a negative) earlier than; before; he won't come until tomorrow"

So, '*until*' is used with regard to events, states, and also in a directly temporal way.

Until as a complementiser

We begin by looking at the use of '*until*' as a complementiser. The analysis that is adopted in this thesis, taken from Radford[39], says that *subordinating conjunctions*—'*after*', '*before*', '*until*', '*except*', etc.[12]—play complementising roles. So, in a sentence such as (4.14), '*until it is thick*' is functioning as an adverbial because '*until*' is a subordinating conjunction. The syntactic analysis, however, is that '*until it is thick*' is a SentenceBAR, with '*until*' being a Conjunction and '*it is thick*' being a Sentence.

The appropriate grammar rules would be
Instruction ⟶VerbPhrase
VerbPhrase ⟶VerbPhrase SentenceBAR
SentenceBAR ⟶Conjunction Sentence
Sentence ⟶NounPhrase VerbPhrase

(4.14) Stir the soup until it is thick.

Until as a preposition

Some conjunctions—in particular '*until*', '*after*', '*before*' and '*since*'—are also prepositions. So sentences like those in (4.15) contain clauses that are syntactically

[12] Some of these words are also prepositions; the prepositional analysis is presented in the next section.

of the form [Preposition NounPhrase].

The following grammar rules apply to instructions that use '*until*' as a preposition:

Instruction ⟶VerbPhrase

VerbPhrase ⟶VerbPhrase PrepositionalPhrase

PrepositionalPhrase ⟶Preposition NounPhrase

(4.15) a. Go there until the party.

b. Keep it open until noon.

c. Sunil decided to wait until dinner.

We can see all of these as meaning ≪*until the time of* ...≫. But it is interesting to note that this temporal connection is more direct when '*until*' is used as a preposition than when it is used as a conjunction. In prepositional use, the noun phrase must refer to an event. When used as a complement the Sentence may refer to an event or a state, as in (4.16) and (4.14) respectively.

(4.16) He waited until Pip came home.

(4.14) Stir the soup until it is thick.

Ellipsis and the use of '*until*'

We end this section with discussion of the interaction between elliptical sentences, and '*until*' used as a preposition and as a complementiser. The purpose of this discussion is twofold:

- to clarify that at times it may seem as though '*until*' is being used as a preposition when closer examination reveals that it is in fact performing syntactically as a complementiser;
- to motivate the view that when '*until*' is used as a complementiser, it most usually describes a state, and an elliptical reading may be needed.

The following examples are used:

(4.17) Wait until midnight.

(4.18) Heat the syrup until thick.

(4.19) Wait until dark.

(4.20) Heat the syrup until dark.

In (4.17), it is clear that '*midnight*' is a noun phrase, and '*until*' is functioning as a preposition. There is no ellipsis here.

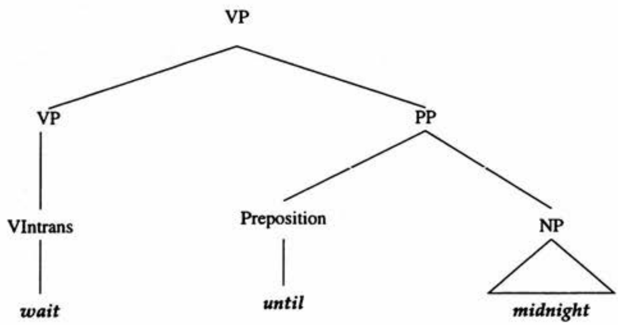
In (4.18), '*thick*' is an elliptical sentence that represents a state—that of ≪*the syrup being thick*≫. An appropriate gloss for this might be ≫*it is thick*≪, and we assume that '*it*' would refer to the syrup.

The phrase '*until dark*', shown in (4.19) and (4.20), has two senses. These can be glossed as ≪*until something has the property of being dark*≫ and ≪*until nightfall*≫. In (4.19) it is more likely that the sense ≪*until nightfall*≫ is being used. It is interesting that the complementiser use of '*until*' can achieve either sense, while the prepositional use only achieves the ≪*until nightfall*≫ reading.

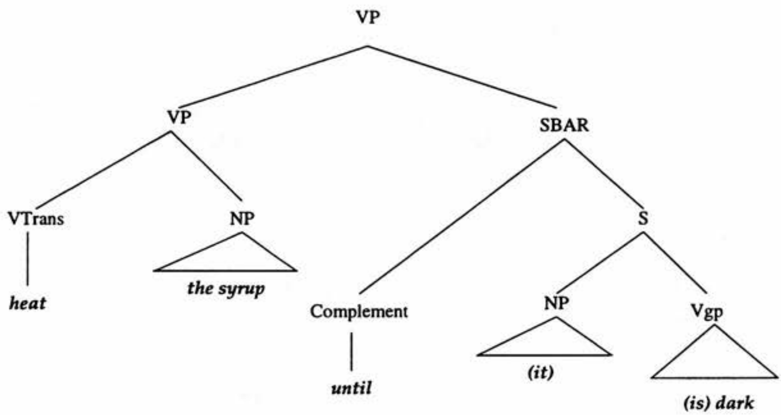
We can rephrase almost any of these '*until*'-phrases to be '*until it is*'. I note as an aside that when this elliptical reading is used, I make no claims about what the '*it*' refers to—it may be ≪*the situation*≫ or it may be '*the syrup*'. However, in order for (4.20) to have the sense where it refers to the property of the syrup being dark, it is *necessary* that this elliptical reading be used. In the non-elliptical reading, with '*until*' functioning as a preposition, we can only describe a particular time of day, which we flag as ≪*dark*≫, in the way we would do for '*dawn*' or '*midnight*'. Figure 4.1 shows the two syntactic structures, for '*until*' as a preposition and as a complementiser.

4.3.4 Other words that specify duration

There are other words, such as '*while*', '*during*', etc., that also indicate that an activity is to occur for an extended period of time. These words may be complementisers (such as '*while*') or they may be prepositions (such as '*during*'). For each, there are similar discussions to be had regarding their syntactic and semantic roles as we have seen about '*for*' and '*until*'. Such discussions, however, are not



"Until" as a preposition



"Until" as a complementiser

Figure 4.1: '*Until*' as a preposition and as a complementiser.

presented here because what is of interest is the general mechanism for describing activity duration; '*for*' and '*until*' provide a reasonable illustration of this.

4.4 The language of repeated or multiple instances of activity

As with continuous activity, repeated activity and multiple instances of activity are described or expressed in a variety of ways. The clearest way is through the use of explicit modifiers; such activity structure may also be implied using plural objects. When explicitly expressed, there is a range of adverbial modifiers available to us, and we will look at these in detail, particularly with respect to their role as quantifiers. We will also look at cardinality and frequency as mechanisms for expressing repeated activity.

Again, analogies between our view of objects in space and activities in time can prove useful. In this case, we use the notion that repeated or multiple instances of activity is the event analogue of plural count nouns. We have some activity, and we want to be able to express that there is more than one occurrence of the activity. That is, we are looking at mechanisms for expressing 'how many' similar activities we have, in the way we may be describing apples and wanting to refer to how many apples we have.

In all of the examples in (4.21), there is a primary activity, that of Lin visiting Debbie.

(4.21) a. Lin visited Debbie in Manchester.

b. Lin visited Debbie twice.

c. Lin visited Debbie every Monday for a year.

d. Lin visited Debbie in Manchester often.

In (4.21b) there are two instances of the activity, and in (4.21c) we are easily able to determine that there are roughly 52 such instances. In (4.21d), it is clear that there are multiple instances of Lin visiting Debbie, although we do not know how many there are. Very loosely, the analogies with say apples would be—for a, b, c and d respectively—one apple, two apples, 52 (evenly spaced) apples and many apples.

One apparent point of difference from objects that we have when we look at events in time is that it is possible for activity instances to occur at the same time. However, I note that it is possible for object instances to occur at the same place, if they are different objects at different times. Sometimes we are explicit that the activity instances must be simultaneous (4.22a), and sometimes we are explicit that they must be consecutive (4.22b). Also, we often know where in time as well as how many of them there are (4.22c).

(4.22) a. Everyone sat down at the same time.

b. Fold the pastry, roll it out and repeat twice more.

c. Shana saw a film once a month all of last year.

4.4.1 Repeated activity in time

Because we take the view that the domain in which activity exists is a temporal one, instances of an activity will have temporal properties. We have already seen—in Section 4.1.2—that activity has a temporal extent and a temporal location. If we see time as a directed one-dimensional axis, any single activity has a location and extent on that axis.

When we have more than one instance of an activity, as well as knowing the number of such instances, we can also know where each instance will occur on the time axis. We assume that the temporal extent of each activity instance is much the same as the others. Though this may not be the case exactly, from the point of view of us considering them to be similar we consider their extents to also be similar. This assumption is especially valid because the language groups the activities together as being similar. Analogously with objects—if we have a few apples, they are of roughly the same size as each other, though it is highly unlikely that any two will be exactly the same size as each other. We still happily consider them to be similar objects.

The remainder of this section is therefore concerned with the following: assuming we have multiple instances of an activity, how do we talk about how many such

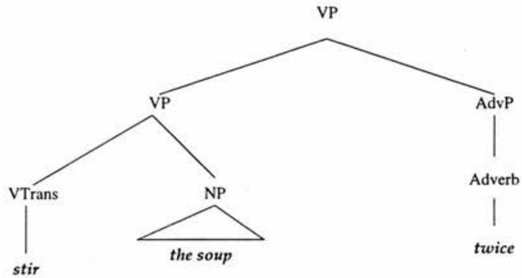


Figure 4.2: Syntactic analysis of '*Stir the soup twice.*'

instances there are, and how do we talk about them in relation to the time axis and in relation to each other?

4.4.2 Expressing multiple instances of activity

The most straightforward way to indicate repeated activity is by the use of adverbs or adverbial phrases, as in the following examples:

(4.23) a. Do this <u>two or three times</u>.

b. Roll into an oblong again and repeat <u>twice more</u>.

c. Stir the soup <u>twice</u>.

In each case, there is an explicit indication that the activity is to occur a number of times. We can think of this as the CARDINALITY of the activity's repetition. Sometimes this cardinality is expressed exactly ('*twice*', '*three times*'), and sometimes it is vague ('*a few times*'). A syntax tree for (4.23c) is shown in Figure 4.2; in general the grammar rule that is relevant is
VerbPhrase ⟶ VerbPhrase AdverbialPhrase.

Many of the points relevant to generalised quantifiers of objects are relevant here[5]. For example, what we mean by '*a few times*' depends on a variety of things including context, much as what we mean by '*a few*' when applied to objects. However, in all these cases, we are clearly expressing that there are multiple occurrences of object or activity, and there is vagueness with regard to the actual number of occurrences. When we say '*two or three*', there is far less vagueness; here we are given a range of cardinality—and the range *is* explicit—rather than an interpretable cardinality.

Linguistically then, we express explicitly repeated activity in the following ways:

- By expressing explicit cardinality, in the form ≫*some number*≪ '*times*'[13].
- By expressing a range of cardinality, such as '*two or three times*', '*more than ten times*', etc.
- By expressing an interpretable quantifier, such as '*a few times*', '*many times*', '*very few times*'.

In all of these, expression is of the form ≫*some quantifier*≪ '*times*'. There are exceptions to this, such as '*once more*', '*again*'. Semantically though, these are equivalent to things like ≪*one more time*≫.

4.4.3 Expressing frequency of repetition

In Section 4.4.1, we saw that when we have repeated activity instances, there are two aspects to this. First, as discussed in Section 4.4.2, there is the cardinality of repeated instances. The second aspect relates to the temporal location of the instances, in particular with respect to each other. This is most usually thought of as FREQUENCY, and it can be regular or not; regularity can also depend on the viewpoint taken.

In the examples that follow, it is clear that there are multiple instances of activity although this is never explicitly stated.

(4.24) a. Baste the roast every 15 minutes.

b. Stir the sauce often.

c. Turn the fritters occasionally.

d. Debbie visited Lin every month.

[13] '*Once*' and '*twice*' are special forms of this for cardinalities of one and two respectively.

With all of these examples, although we know something about the temporal 'distribution' of the activity instances, we do not know either how many there are or the time boundaries within which they occur. Some information can be inferred from the tense (in 4.24d) or from the fact that we have an instruction and therefore have a start boundary of at least $\ll now \gg$. However, these matters are not relevant to the discussion at hand.

Again, the appropriate syntax rule is
VerbPhrase ⟶VerbPhrase AdverbialPhrase

As with expressing cardinality, frequency can be expressed exactly or vaguely. '*Often*' and '*occasionally*' depend very much on 'what is normal'. For both of them, we also often assume some regularity though this is also not explicit.

4.4.4 '*Every*' and '*each*'

'*Every*' and '*each*' are important ways of expressing multiple instances of activity. They usually are used as quantifiers, but it is important to note that they can be applied to temporal concepts—as in (4.25a), or to objects—as in (4.25b).

(4.25) a. Check the temperature every hour.

b. Check each disk for scratches.

Either word may be used with either object or temporal concepts. It is not within the scope of this thesis to examine the subtle differences between the two words, and what may be inferred from choosing to use one rather than the other[14].

The Collins dictionary says that '*every*' and '*each*' are determiners; '*each*' is also an adverb. The use it gives in the temporal sense is the following:

> "each: used before a noun phrase to indicate the recurrent, intermittent or serial nature of a thing."

The Oxford dictionary says '*every*' is an adjective.

[14] Where such differences are obvious or important, they will be noted.

"1. each single ('*heard every word*'). 2 each at a specified interval in a series ('*comes every four days*'). 3. all possible"

'*each*' is also defined as an adjective, as well as a pronoun:

"every one of two or more persons or things, regarded separately"

Syntactically, '*every day*' is a *noun phrase*, in the same way that '*every man*' is. Semantically, both are performing as quantifiers; ≪*every day*≫ functions as an adverbial.

An approach to dealing with this data is to think of these kinds of phrases as TEMPORAL MEASURE NOUN PHRASES, in much the same way as temporal measure adverbials are described in Section 4.3.2. That is, they are *noun phrases* that describe some quantity, or measure, of time. Then with the rules

Modifier ⟶AdverbialPhrase | PrepositionalPhrase |
NounPhrase[+time, +measure][15]
VerbPhrase ⟶VerbPhrase Modifier

we are able to use this kind of modifier with VERB PHRASES appropriately. This approach also deals with phrases like '*three times*', which again can be seen to be *noun phrases* performing an adverbial role.

4.4.5 Plural objects

Activity, particularly in recipes, often involves some object that is the recipient—or patient—of the activity. In (4.26), ≪*the onion*≫ is the recipient of the action ≪*peel*≫. It is often the case that the recipient is a plural count object. When this is so, it sometimes suggests that the entire activity can be seen as a repeated application of the same action, for each of the count objects.

(4.26) Peel the onion.

(4.27) a. Peel each of the onions.

[15] This indicates that the noun phrase has the properties time and measure, indicating a temporal measure noun phrase.

b. Peel each onion.

c. Peel the onions.

In (4.27a), we can see the activity as consisting of a number of similar sub-activities, each one of which is like that described in (4.26). In this case, it is the determiner, '*each*', with the plural '*onions*', that suggests plural or repeated activity[16]. We get a similar sense from (4.27b)[17] and (4.27c); in (4.27c) the distinction of there being a similar activity for each onion is not as strong, but nevertheless it exists.

With regard to the role of plural objects in suggesting multiple instances of activity, I note the following:

- It is not only from the patient of an activity—when it is plural—that such suggestion may come. This is demonstrated in (4.28):

  (4.28) Every person peeled an onion.

- The meaning of the verb must be in some way 'distributive', with respect to its object, for the suggestion to be appropriate. That is, the following must hold

  Activity(Objects 1—N) <==>

  Activity(Object1), Activity(Object2) ... Activity(ObjectN)

  (4.29) is an example where this may not be the case—the result of applying ≪*weigh*≫ to ≪*each apple*≫ is not necessarily the same as the result of applying it to ≪*the apples*≫[18].

  (4.29) Weigh the apples.

  I make the assumption that all the verbs I deal with have this distributive property, though I acknowledge that there are verbs and uses of verbs where this is not in fact the case, and a more sophisticated analysis would be appropriate.

16 This is in contrast with something like '*Visit Mary each week*', where the '*each*' is being applied temporally rather than to a count object.

17 In this sentence, although syntactically we have '*onion*', semantically it is the case that there are ≪*onions*≫.

18 In fact, (4.29) is ambiguous in meaning ≪*do this thing called 'weigh' to each of the apples*≫ and ≪*for this thing that is 'the apples', weigh it*≫.

Linguistic plural—semantic mass

It is sometimes the case that we have plural objects that are not seen as count objects. So, while in (4.30a) we can imagine this as consisting of a number of instances of ≪*scrubbing of a single potato*≫, for (4.30b), it is hard to imagine us viewing this as an activity that consists of sub-activities, each being the ≪*boiling of a single lentil*≫.

(4.30) a. Scrub the potatoes.

b. Boil the lentils.

It is important to note that for the examples in (4.30a), we are *not* saying anything about whether the potato scrubbing activities occur consecutively or simultaneously, or in any other particular temporal fashion. We are saying nothing about their temporal relation to each other at all. It is conceivable that there exists a device that does potato scrubbing, and it can scrub a number of potatoes at a time. In (4.30b), we also say nothing about when one lentil gets boiled with respect to any other lentil.

Why then do we get a sense of plural individual activities when considering scrubbing potatoes or peeling onions, but one activity when boiling lentils? Karlin[23] claims that distinguishing between the objects as mass or count is not the best indicator; we need to know about the physical attributes of the objects, such as size and consistency, which we get from world knowledge. While I agree that simply distinguishing between objects as mass or count is in itself not the best approach, Karlin's approach also has flaws. In particular, she simply moves the focus from some semantic view of the objects (their ≪*countness*≫) to another semantic view of the objects (their ≪*physical attributes*≫). A broader approach, which takes into account things other than just the objects or patients of the activity, provides more insight, and so the view I present is that a combination of three things gives us the distinction:

1. the verb itself,
2. whether we consider the object that the activity is to be performed on a plural or mass object, and

3. the execution context.

Each of these factors have different relative significance, and importantly, the distinction is not a hard-and-fast one.

From a general cooking perspective, we see lentils as a mass, and potatoes as plural count objects; however, from the language we do not know this. It is world knowledge, and execution context that tells us that boiling is something that can happen to many objects simultaneously, while peeling usually cannot. However, there exist pots that are too small to boil more than one onion at a time, there exist pots in which we can boil ten onions simultaneously, and there exist machines that are able to peel a number of potatoes at once.

The approach adopted in this thesis is to consider lentils, potatoes, onions, etc. as plural count objects at the linguistic level. So, we would initially expect to boil the lentils one at a time, as we expect to peel the onions one at a time. Context and world knowledge may tell us that the lentils can be boiled at the same time; however, the ≪*boiling of one lentil*≫—whether or not it occurs at the same time as many other boilings of single lentils—is considered to be an atomic activity that is a sub-activity, or single instance, of the extended activity of ≪*boiling the lentils*≫.

Activity involving plurals of multiple objects

We also have activity that comes from ditransitive verbs; in this case there may be two places where plural objects exist, such as in (4.31). This situation is somewhat similar to that exemplified in (4.28)

(4.31) a. Take two balls from every container.

b. Put three currants into each of the five tarts.

So, in (4.31b), there are potentially fifteen sub-activities—each of which is a ≪*putting of one currant into one of the tarts*≫. There are other sub-activity combinations that are possible—five sub-activities of ≪*putting three currants into one of the tarts*≫, for example.

If we imagine that we have a robot that is only able to perform the activity of putting one currant into one tart, then achieving the execution of (4.31b) will require those fifteen sub-activities. A robot that can put three currants at a time will need to execute five sub-activities. So, for the activity indicated by this particular verb '*put*', the nature of the *both* objects that it is being applied to, as well as the context in which it finally is executed, will affect the overall activity structure.

No matter what activity structure is chosen, however, it is still the case that from the language, we identify the very basic activity of ≪*putting one currant into one tart*≫—this may then be overridden by world knowledge or context.

4.5 Multiple modifiers

The examples we have seen so far in this chapter have usually been quite simple—both in the linguistic sense and in terms of the activity they are instructing. It is however often the case that more complex language is used, and more complex activity structure is described. Here we examine what happens when more than one modifier is used in describing extended activity.

There are two ways in which two modifiers may be combined with an activity. The following two sentences have apparently similar syntactic structure, yet the modifiers in each are being combined in different ways.

(4.32) a. Lee played the piano every day until the holiday ended (and she had to return home).

b. Lee played the piano for five hours until he was exhausted (and he had to stop).

1. It is possible to combine a modifier with the activity, producing a new activity, which is then combined with the second modifier, producing a nesting effect on the activity structure. In (4.32a), there is an activity of ≪*Lee playing the piano*≫, which happened ≪*every day*≫; this activity of ≪*Lee playing the piano every day*≫ continued for a time until ≪*the holiday ended*≫. The activity structure is illustrated in Figure 4.3.

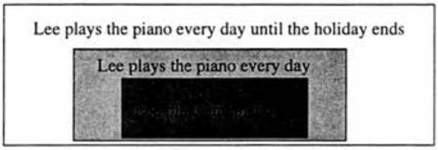


Figure 4.3: The activity structure of '*Lee played the piano every day until the holiday ended*'.

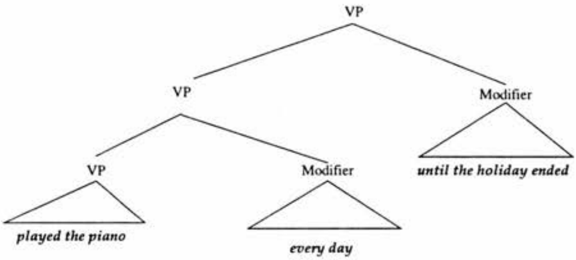


Figure 4.4: Syntactic structure of '*Lee played the piano every day until the holiday ended*'.

The appropriate syntax rule is

VerbPhrase ⟶VerbPhrase Modifier

which is applied for as many modifiers as there are in the sentence. The syntactic structure of (4.32a) is shown in Figure 4.4. Combining modifiers in this way is discussed further in Section 4.5.1

2. There is a second way of combining modifiers, which does not result in a nesting effect, as in (4.32b). In this example, we see ≪*for five hours*≫ and ≪*becoming exhausted*≫ as *both* describing the same activity structure, but in different ways. Both describe a continuous period of Lee playing the piano ≪*for some time*≫; in fact they both describe the same time period. This activity structure is shown in Figure 4.5[19].

   This essentially involves combining two modifiers with each other to produce a complex modifier, which then modifies an activity. Here, the appropriate syntax rules are

[19] It is possible to see (4.32b) as being semantically equivalent to the sentence '*Lee played the piano for five hours* or *until he was exhausted and he had to stop*'.

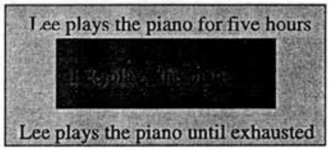


Figure 4.5: The activity structure of '*Lee played the piano for five hours until he was exhausted*'.

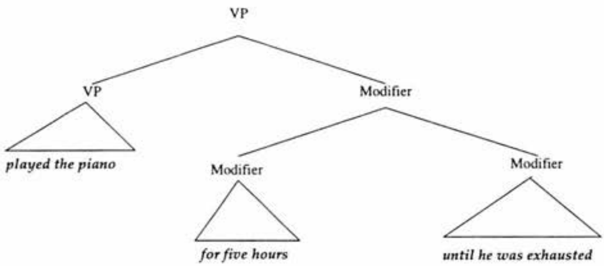


Figure 4.6: Syntactic structure of '*Lee played the piano for five hours until he was exhausted*'.

> Modifier ⟶Modifier Modifier
>
> VerbPhrase ⟶VerbPhrase Modifier
>
> Figure 4.6 is the syntax tree for (4.32b).
>
> An instance of this, which is the use of '*or*' in combining modifiers, is discussed further in Section 4.5.2.

It is usually the case that when multiple modifiers are present, the approach described in 1. above—of combining the modifier with the verb phrase, rather than combining modifiers with each other—is intended, unless there is punctuation or the use of conjunctions to indicate otherwise. I do note that, as in (4.32b), this is not always the case, but such instances are rare and so the approach I take in my implementation is to assume that the former is intended when there is no conjunction.

4.5.1 Combining modifiers with activity—nesting modifiers

When multiple modifiers are combined with activity, as in (4.33), there is a nesting effect on the activity. Here, there is a basic activity ≪*sleeping*≫. This happens ≪*for five hours*≫, producing a new activity—which happens to be extended—≪*[sleeping] for five hours*≫. This in turn happens ≪*every day*≫, producing a further new extended activity ≪*[[sleeping] for five hours] every day*≫. Finally, this is modified to produce the overall new activity ≪*[[[sleeping] for five hours] every day] for a month*≫. We have simply applied each modifier to the activity, as it has been encountered in the sentence.

(4.33) He slept for five hours every day for a month.

It is not always the case, however, that the syntactic order of modifiers gives us the semantic order of their application.

In both the examples of (4.34), there is a basic activity, ≪*sleeping*≫, that is then modified by two modifiers. In (4.34a) the activity is modified by ≪*once a day*≫ and ≪*for five days*≫. In (4.34b), the second modifier is ≪*for five minutes*≫.

(4.34) a. He slept once a day for five days.

b. He slept once a day for five minutes.

The sentences appear to be very similar—they have a similar syntactic analysis, shown in Figure 4.7—and it would seem that both are saying

(4.35) He slept once a day ≫*for some time period*≪.

Thus, it appears that the initial basic activity of ≪*sleeping*≫ is first modified by '*once a day*' to give a new activity of ≪*sleeping once a day*≫. It would seem that it is only our knowledge of how days are divided into minutes and an ability to compare sizes of time periods that allows us to distinguish between the different semantics of these sentences.

If we examine these sentences further, however, it becomes clear that the gloss in (4.35) is not well justified. In particular, if we examine the syntax and look for relationships between constituents, (4.34b) can become (4.36a) or (4.36b).

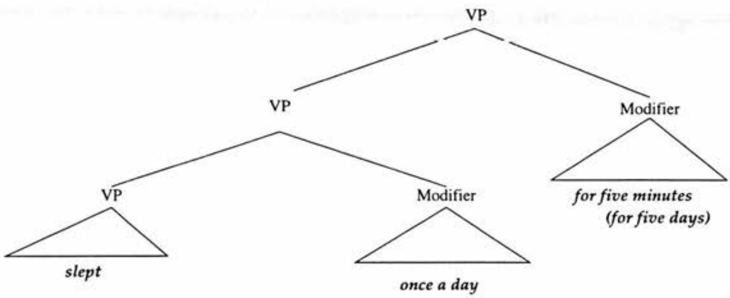


Figure 4.7: Syntactic analysis of '*He slept once a day ≪for some time period≫*.'

(4.36) a. He slept for five minutes once a day.

b. Once a day, he slept for five minutes.

On the other hand, (4.34a) can only become (4.37a); (4.37b) is semantically odd, or syntactically awkward.

(4.37) a. For five days, he slept once a day.

b. #He slept for five days once a day.

It is now clear that the new activities—at the second level of nesting, which is once the first modifier has been applied to the activity—should be, for (4.34a) and (4.34b) respectively:

(4.38) [He slept once a day]—and this extended activity occurs for some time period.

(4.39) [He slept for five minutes]—and this extended activity is subject to some frequency modifier.

The order of modifier application is thus not a simple matter of left-to-right; although this can be done syntactically, which would result in the basic form shown in (4.35), semantic processing rules this out in some cases. It is our knowledge of what time intervals can contain other time intervals that makes (4.37b) seem odd, rather than any properties of the syntactic constituents and where they are in relation to the verb.

I argue that this is related to the problem of quantifier scoping, when the modifiers are TEMPORAL MEASURE ones, and can in fact be dealt with by treating all temporal measure adverbials as part quantifiers, as suggested by Dowty[14] and then Moltmann [29]. Measure adverbials are adverbials that describe a measure, such as '*for half an hour*' or '*throughout the world*'[20]. According to Moltmann, adverbials like '*for half an hour*' are performing the same semantic function as adverbials like '*three times*'. A simple gloss is that both of these describe 'how much' of some activity there is; they therefore are performing some quantification function.

It has been long accepted that adverbials like '*three times*' are event quantifiers; this is straightforward and obvious. These are then quantifiers over discrete, distinct activity entities, much in the way quantifiers over objects are quantifiers over count objects, such as '*three people*'. Adverbials like '*for half an hour*' or '*until noon*' are quantifiers over—possibly—continuous activities; this is similar to quantification over mass objects, such as '*some water*' or '*a litre of milk*'. So, adverbials perform quantification over both repeated and continuously executed activity—they are applicable to both protracted activity and to multiple instances of activity.

Treating temporal modifiers as quantifiers allows some semantically odd readings—for example in (4.37b)—to be ruled out, as well as allowing readings that would not otherwise be found to be included.

The approach I take is as follows:

- First, we distinguish between temporal measure modifiers such as '*for ten minutes*', '*three times*', '*until noon*' and those that are either non-temporal such as '*on the table*' or non-measure such as '*tomorrow*'.
- Then, we treat those modifiers that are temporal measure in the same way as we do other quantifiers.
- When we combine temporal measure modifiers, if the modifiers are time intervals that are related to each other, we allow only those scopings that do

[20] I am only interested in temporal measure adverbials; however for completeness—and because of the philosophical stance I have taken of exploiting the object-eventuality analogy—I note that a similar story can be told about spatial measure adverbials.

not contravene the well-formedness of the semantic representations[21].

I clarify this using the sentences in (4.40) which are syntactically acceptable, yet (4.40d) and (4.40e) are semantically odd.

(4.40) a. Jo slept every afternoon for a week.

b. Every afternoon for a week, Jo slept.

c. For a week, Jo slept every afternoon.

d. #Jo slept for a week every afternoon.

e. #Every afternoon, Jo slept for a week.

The acceptability or oddness is due to the fact that ≪*every afternoon*≫ is a subinterval of ≪*for a week*≫. Any use where the subinterval takes wide scope over the interval that contains it is semantically odd; a logical form representing this will be syntactically ill-formed, as in (4.41b) where the variable w is not bound in $a \subseteq w$, which represents (4.40d) and (4.40e). (4.41a) represents (4.40a), (4.40b) and (4.40c), and is well-formed.

(4.41) a. one(w, week(w), all(a, afternoon(a), $a \subseteq w$, sleep(jo)))

b. **all(a, afternoon(a), $a \subseteq w$, one(w, week(w), sleep(jo)))

In the examples in (4.42), there is no subinterval relationship and none of the sentences are semantically odd.

(4.42) a. Jo slept twice every afternoon.

b. Every afternoon, Jo slept twice.

c. Twice, Jo slept every afternoon.

d. Jo slept every afternoon twice.

There are two temporal measure adverbials—'*twice*' and '*every afternoon*'—and because of the scope interactions there are two possible readings, glossed in (4.43).

[21] I note that this is in some ways analogous to the approach of Pratt & Brée[37, 38] who have also identified this issue and use an interval-based temporal logic to restrict quantification over sub-intervals.

(4.43) a. *≪On two occasions, Jo slept every afternoon.≫*

b. *≪Every afternoon, there were two occasions when Jo slept.≫*

Punctuation causes one or the other reading to be favoured, as do particular syntactic constructs. However, (4.42c) without punctuation allows both readings, and the quantifier approach allows this to be identified.

Combining modifiers and quantified objects

For completeness, a brief description of the scope interactions between modifiers and quantified objects is presented. There are many examples of sentences, such as (4.44), where an activity involving a quantified object is modified by a temporal measure modifier.

(4.44) Mary visited all of her friends for half an hour.

This sentence has the two readings, glossed in (4.45); these readings are easily obtainable by scoping the modifier and object with respect to each other and to the basic activity of ≪*Mary visiting one friend*≫.

(4.45) a. *≪In a period of half an hour, Mary visited all of her friends.≫*

b. *≪For all of the people that are Mary's friends, she visited each for half an hour.≫*

4.5.2 Combining modifiers with each other—the use of '*or*' in recipes

The conjunction '*or*' sometimes has a use—mostly found in recipes and other instructions—that is unusual; this use is examined here.

According to the Oxford dictionary, '*or*' is a

(4.46) "conjunction—introducing the second of two alternatives; introducing all but the first or only the last of any number of alternatives; introducing a synonym or explanation of a previous word; introducing a significant

afterthought ...
with '*rather*'—introducing a rephrasing or qualification of a preceding statement."

Syntactic analysis

A common syntactic analysis of words like '*and*' and '*or*' is that the phrases or clauses they conjoin are of the same syntactic category. So, in a sentence like (4.47a), '*the carrots*' and '*the pumpkin*' are being conjoined, and they are both noun phrases. This seems to be a useful analysis.

(4.47) a. Bake the carrots and the pumpkin.

b. Use a pound of butter or margarine.

Semantically, ≪*the carrots*≫ and ≪*the pumpkin*≫ are similar things, and doing ≪*bake*≫ to either of them is a similar thing, and the instruction is saying we are to do it to both. A similar analysis applies to (4.47b), except that we are to do ≪*use*≫ to a pound of either ≪*butter*≫ or of ≪*margarine*≫. It is also reasonable to say that this is logically disjunction—even though '*or*' is, linguistically, a conjunction, the semantics of this example point to disjunction.

However, there are linguistic constructs found in recipes that do not fit in with this analysis, such as those in (4.48).

(4.48) a. Steam for two minutes or until the mussels open.

b. Cook for about 5 minutes or until soft and transparent.

A syntactic analysis of (4.48a) would give the syntax tree of Figure 4.8. This tree uses the following rules:

VerbPhrase ⟶VerbPhrase PrepositionalPhrase
PrepositionalPhrase ⟶PrepositionalPhrase Conjunction SentenceBAR
SentenceBAR ⟶Complement Sentence

The second rule is unusual, in that the general form of rules involving conjunctions is X ⟶X Conjunction X; this rule, however, to allow for the unusual combination of clauses about the '*or*', is of the form X ⟶X Conjunction Y.

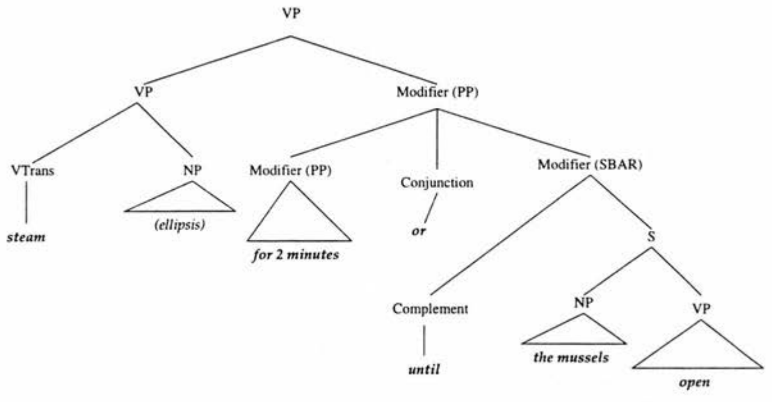


Figure 4.8: Syntactic analysis of '*Steam for two minutes or until the mussels open.*'

A better approach would be to say that PrepositionalPhrases and SentenceBars are Modifiers, and replace the rule involving the Conjunction with

Modifier ⟶Modifier conjunction Modifier

This can also be extended to considering AdverbialPhrases to be Modifiers, requiring the following:

Modifier ⟶AdverbialPhrase | PrepositionalPhrase | SentenceBAR

This approach then also allows sentences such as (4.49) to be treated in the same way.

(4.49) Fold the dough in half twice or until it begins to break.

Semantic issues

As well as this syntactically unusual form described above, there is also an associated unusual use of '*or*' from a semantic perspective. Although the usual view of the linguistic conjunction '*or*' is that its semantic role is logical disjunction, I argue that in sentences such as those in (4.48), the '*or*' is playing the role of conjunction. The scenario I propose is that we have disjunction for stopping—we stop if either the duration or the state change is true—but conjunction for success—stop and a normal or successful outcome is only true if both the clauses are true. This view is argued more fully in Rock[41], and I present only the summary of the argument here. I note that this semantic argument also applies when the '*or*' is omitted, and a comma is used instead. Both the sentences in (4.50) have a similar semantic form.

(4.50) a. Bake for 45 minutes or until the rabbit is tender[15, pg118].

b. Bake for about 2 hours, until the rabbit and lentils are tender[15, pg119].

In sentences that involve '*or*' in this way, I claim that we have an instruction that can be seen as describing one of these two scenarios:

- Do some action until an expected state change occurs. This should take the duration specified.
- Do some action for a specified duration. If the expected state change does not occur during this time, then it is likely that something has gone wrong.

In general, the combining of modifiers with each other first, rather than as a nesting of modifiers, happens through the use of conjunctions or punctuation. If neither punctuation nor conjunctions are present, as in (4.52a)[22], a nesting reading, such as described in Section 4.5.1, is preferred. When a conjunction is there, as in (4.52b), or punctuation is present, as in (4.52c), a combination of modifiers is preferred, as in (4.51b).

(4.52) a. Roast the vegetables until they are tender for about an hour.

b. Roast the vegetables for about an hour or until they are tender.

c. Roast the vegetables until they are tender—for about an hour.

[22] This sentence, without punctuation, in fact has three possible syntactic structures—

(4.51) a. [Roast the vegetables [[until they are tender] [for about an hour]]]

b. [[Roast the vegetables until they are tender] for about an hour]

c. [Roast the vegetables until [they are tender for about an hour]]

4.6 Combining verb phrases

We often wish to instruct that two different activities are to occur at the same time. This is especially true of cooking, but it also happens in many other areas.

In all the examples of (4.53), there are clearly two activities—cooking and stirring—that are to be executed simultaneously. Syntactically, this is expressed via two verb phrases—'*cook the sauce for five minutes*' and '*stirring continuously*'. It is the form of phrases like the latter, which I call gerundive verb phrases (GVP's), that are of interest here, and how they combine with modifiers, and with other verb phrases.

(4.53) a. Cook the sauce for five minutes, stirring continuously.

b. Cook the sauce, stirring continuously, for five minutes.

c. Cook the sauce, stirring continuously for five minutes.

d. Stirring the sauce continuously, cook it for five minutes.

In both (4.53a) and (4.53b), which are syntactically equivalent—there is simply a changed order of one of the core phrases—the two activities are ≪*cooking the sauce for five minutes*≫ and ≪*stirring the sauce continuously*≫. The relevant grammar rules are

GerundiveVerbPhrase ⟶GerundiveVerbPhrase Modifier[23]

VerbPhrase ⟶VerbPhrase GerundiveVerbPhrase

VerbPhrase ⟶VerbPhrase Modifier

Figure 4.9 and Figure 4.10 show syntax trees for (4.53a) and (4.53b) respectively. It is accepted that the stirring activity occurs for the ≪*five minutes*≫ of the cooking activity.

In (4.53c), the absence of the second comma results in syntactic ambiguity.[24] One possible syntactic form is the same as that of (4.53b), which is shown in Figure 4.10. Another syntactic form, where the entire phrase '*stirring continuously for five minutes*' modifies '*cook the sauce*', is shown in Figure 4.11.

[23] Recall that PrepositionalPhrases and AdverbialPhrases are modifiers.

[24] This sentence, with no punctuation at all, has a third syntactic structure, where '*continuously*' modifies the phrase '*cook the sauce stirring*'. This possibility is examined in Section 9.2.2.

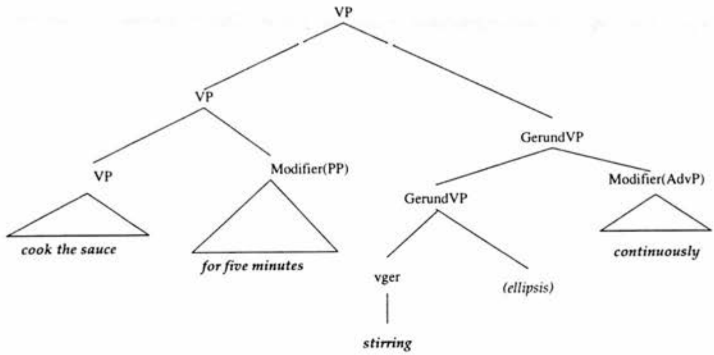


Figure 4.9: Syntactic analysis of '*Cook the sauce for five minutes, stirring continuously.*'

A similar argument applies to (4.53d), where the instruction could mean ≪*while stirring the sauce continuously—for some unspecified length of time—cook it for five minutes*≫. All that is certain is that there is some concurrent execution of the two activities.

When syntactic entities are combined in this way, punctuation plays an important role, as is clear from the semantic differences between (4.53b) and (4.53c). Discussion of punctuation in general is outwith the scope of the thesis; however I do note that it plays a role, particularly in demarcating phrases and modifiers, and this is incorporated in the computational system described in later chapters.

In the sentences in (4.53), there is a primary activity—in this case ≪*cook*≫—that is expressed by the verb in imperative form. There is a secondary activity—≪*stir*≫—that is expressed by the verb in gerundive form. The actual ordering of the phrases does not change what is viewed as primary or secondary; however in (4.3d) there is a possible sense that the ≪*stirring*≫ is 'less obviously secondary'.

4.7 Discourse and anaphoric reference to events

We may sometimes need to express an activity structure that is spread through some discourse. For example, we may wish to refer to an activity already described, such as in (4.54).

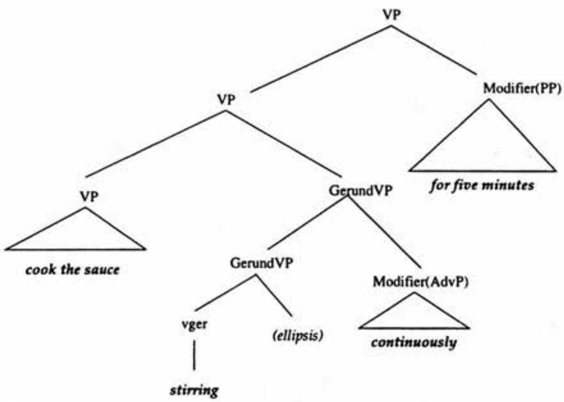


Figure 4.10: Syntactic analysis of '*Cook the sauce, stirring continuously, for five minutes.*'

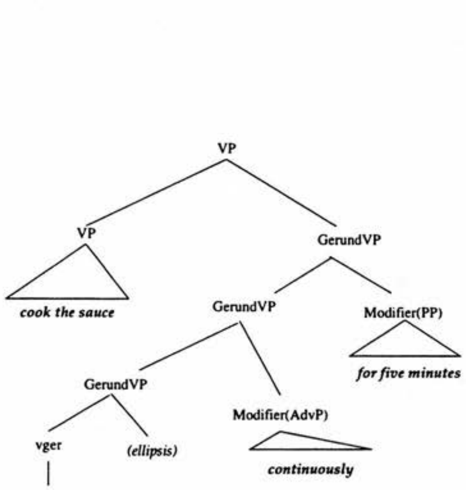


Figure 4.11: Syntactic analysis of '*Cook the sauce, stirring continuously for five minutes.*'

(4.54) Melt a pound of butter until it is foaming. Skim off the foam and discard it. Heat the butter and skim again. Do this twice more.

This discourse contains anaphoric reference to objects—the first use of '*it*', for example, is a reference to ≪*the butter*≫. However, there is also the possibility of anaphoric reference to activity, as in '*do this twice more*'. Resolving the (activity) referent of '*this*' is a similar problem to resolving the (object) referent of '*it*'.

A complete treatment of anaphoric reference to events is outwith the scope of this thesis, though I do note that this is an obvious topic for further research.

4.8 Summary

This chapter has been concerned with identifying linguistic devices that are used to express extended activity—devices that express protracted activity and then devices that express multiple instances of activity. These devices are found across a range of phenomena, from lexical information connected with the verb, to the use of modifiers, to plural objects. We also examined the way these devices interact, to express extended activity that is complex.

In the next chapters, the analysis of the conceptual phenomena—which was presented in Chapter 3—is combined with the analysis of the linguistic phenomena—which has been presented in this chapter—to provide an approach to understanding language that is about extended activity.

The following issues will be dealt with:

1. Identification of basic, unstructured activity, described by a verb phrase, and the kind of basic activity—discrete or protracted—that is, insofar as it is suggested by the language.
   '*Stir the soup*'
   '*Put the saucepan on the table*'

2. Identification of simple structured activity that comes from plural or quantified objects in the verb phrase.
   '*Put a spoonful of jam into each tart*'

3. Identification of simple structured activity that comes from modifiers of the verb phrase—
   (a) through the use of adverbials
   '*Do this two or three times*'
   '*Stir the soup every half hour*'
   (b) through the use of prepositional phrases and sentence complements
   '*Stir the soup for five minutes*'
   '*Beat the mixture until all the sugar has dissolved*'
4. Identifying complex structured activity that comes from combining modifiers
   (a) with each other
   '*[Steam the mussels [for two minutes or until they open]]*'
   (b) with modified verb phrases
   '*[[Stir the soup often] for half an hour]*'
   (c) with verb phrases containing plural or quantified objects
   '*[[Stir some soup] every half hour]*'

   including the scope interactions that arise in (b) and (c).
5. Identifying complex structured activity that comes from combining verb phrases
   '*Simmer the soup, stirring occasionally, for fifteen minutes*'
   '*Roast the meat for about an hour, basting twice*'
6. Identifying instances where verb category suggestions are overridden by modifier
   '*Stir the soup twice.*'
   '*Stir the soup for ten minutes.*'
7. The structured activity found in simple discourse
   '*Melt a pound of butter until it is foaming. Skim off the foam and discard it. Heat the butter and skim again. Do this twice more.*'

All temporal measure modifiers are treated as quantifiers, with standard scoping treatment applied in their interaction with each other and with quantified objects.

For each sentence, if there are multiple syntactic readings, all of these will be identified. In addition, for each syntactic reading, when there are scope interactions due to the presence of more than one temporal measure modifier or quantified object, all the relevant semantic readings will be identified.

Chapter 5

Representing the semantics of extended activities

In the previous two chapters we looked at the conceptual issues involved in extended activity and how extended activities are expressed in language.

We now look at how the semantics of sentences about such activities may be represented. In the next chapters we will look at mechanisms for obtaining such representations, and mechanisms for demonstrating these semantics visually. In this chapter, though, we look in detail at the representations that are needed to represent the semantics of the activity structure that is found in language that is about extended eventualities.

The structure of this chapter is as follows:

- We begin with a general discussion of issues pertaining to representation, in which I introduce the use of feature structures. I also clarify some issues regarding the representation of temporal information.
- Next, I present a description of how happenings are represented.
- The next section deals with the way in which extended eventuality structure is represented. This includes sets, masses, and more complex structures.
- Section 5.4 discusses the issue of quantification. All temporal measure adverbials are represented as quantifiers, in the same way as some adverbs ('*twice*') and all nominal quantifiers are. This section describes how these various quantifiers are represented and how they interact.

- The final section, 5.5, presents a BNF that is a summary of the representation language, and describes all valid feature structures.

It is important to clarify that the representations described in Section 5.3 are for scoped representations; the grammar in Chapter 6 produces feature structures that are unscoped, and the scoping algorithm in Chapter 7 takes these and produces scoped feature structures. In Section 5.4, unscoped and corresponding scoped feature structures are discussed, and the BNF in Section 5.4 covers valid feature structures for both of these.

5.1 Some representational basics

The principal concern here is the representation of eventuality structure and substructure. The fundamental entities in the semantics are therefore activities, and we are interested in representing relationships between such activity entities.

For example, given the sentence in (5.1),

(5.1) Baste the roast three times.

we can describe it informally as the following:

> "We have a complex activity that itself consists of three simple activities or happenings, executed consecutively. Each of these happenings consists of the action of basting the roast."

Thus we want to be able to represent two things:

- that there is a basic happening entity, say called e_2, representing a $\ll$*basting of the roast*$\gg$, and
- that the overall activity entity, say called e_1, consists of three such e_2 entities, which are similar to each other.

For the basic happening entity, we need to be able to represent what kind of happening it is. So, we want to be able to identify and distinguish between different kinds of happening. This is described in Section 5.2, which is about representing happenings.

For the overall activity entity, we need to be able to represent the relationships between its constituent happenings or activities. For this, we must distinguish between different kinds of relationships between activities. Relationships between activity entities that are structured are called STRUCTURINGS; in Section 5.3 we discuss structurings and their representation.

Feature structures will be used to demonstrate activity representations. They are a common notation for unification-based formalisms; such a formalism has been used for the semantic analysis described in this and later chapters. The underlying approach is to build up semantic representations by performing unification on structured collections of FEATURE-VALUE PAIRS, which are significant features and the values they take in representing meaning. The feature-value pair [**category=verb**], for example, may represent a syntactic feature [**category=**], which has the value [=**verb**]. See Shieber[44] for an exposition of unification-based approaches to syntactic and semantic analysis in general.

In the example of sentence (5.1) we have a happening that, as will be shown in Section 5.2, is [**type=discrete**]. Three of these form the structured activity, described by (5.1), that, as we will see in Section 5.3, is [**structure=ordered-set**]. A simple feature structure for (5.1) is shown in (5.2).

$$(5.2)\quad \begin{bmatrix} \text{structure} = \text{ordered-set} \\ \text{index} = e_1 \\ \text{extent} = \begin{bmatrix} \text{cardinality} = 3 \end{bmatrix} \\ \text{content} = \begin{bmatrix} \text{structure} = \text{happening} \\ \text{index} = e_2 \\ \text{type} = \text{discrete} \\ \text{substance} = \begin{bmatrix} \text{action} = \text{baste} \\ \text{patient} = \text{roast} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

5.1.1 Entities in the representation

The representation must contain two types of things—HAPPENINGS, which are the basic kinds of activity entities, and STRUCTURINGS, which are ways of combining activities, or relationships between activities. Because a happening is itself an activity, mechanisms for combining activities will of necessity apply to happenings.

Happening entities Using the object-eventuality analogy presented by Jackendoff and others, we have at the basis of our ontology for examining repeated and continuous activity two kinds of entity—a DISCRETE happening, analogous to a count object, and a CONTINUOUS happening, analogous to a mass object. These are the two basic elements that will be available to our characterisation. The activity described by '*pick up the pen*' is discrete; the activity described by '*sleep*' is continuous[1].

Structuring entities Using the continuous and discrete activity elements, and the concept of activity composing to form more complex activity, we also need to be able to characterise and thereby distinguish between the different ways in which complex activity can be made up of repeated discrete or continuous activity. So, the second part of the representation involves a characterisation of different kinds of activity structure. In Section 5.3 I present different ways in which activity can be structured, that cover the kinds of groupings that contribute to extended activity.

5.1.2 Temporal issues

We return briefly to an earlier topic, which is that of activity occurring in time. When we look for patterns in activity substructure, we find there are different ways in which we can distinguish between temporal patterns.

As was discussed in Chapter 2, there has been work centered on characterising the

[1] We will see that the information regarding this comes from the lexicon; it is also the case that this information may be overridden by further processing, for example in the case of '... *sleep twice a day for a week*', where the '*sleep*' refers to discrete occurrences of ≪*sleeping*≫.

various temporal relations that may exist between two events[2]. Our needs however are for fewer distinctions. We are interested in activities that are similar to each other, because our concern is with multiple instances of the same activity[3]. Given two similar activities, say A and B, it is not important here for example whether A occurs before or after B, because we do not distinguish between the content of A and B. If we have the complex activity described by (5.3), we may describe this as (5.4)[4]. I note however that the notation used in (5.4) is not the representation that is used in this work; it was chosen only to illustrate the point of this example.

(5.3) Stir the soup twice.

(5.4) ordered-set $(e_1) \wedge$ stirring $(e_2) \wedge$ stirring $(e_3) \wedge$ consists (e_1, e_2, e_3)[5]

I have used simple logical notation here in order to be able to distinguish between two similar sub-activities; as is demonstrated, such capability is not required of the feature structure representation that is used in the model presented. It is not important whether e_2 occurs before e_3 or after[6]. This is because from our perspective, e_2 and e_3 are indistinguishable; they are similar ≪*stirring*≫ sub-activities. We are only concerned that they occur—because of the adverbial *twice*—at temporally distinct times with respect to each other.

Between two similar instances of activity, then, the following temporal aspects are of concern:

- First, when we have multiple instances of similar discrete activity, these can be evenly spaced temporally, or not. That is, their temporal distance

[2] For example, Allen[2] presents 7 primary relations and, for some, their inverses; How[18] discusses 8 temporal relations.

[3] A note on language usage and terminology—multiple instances of the SAME activity result in multiple SIMILAR activity entities.

[4] The use of (e_2) and (e_3) to represent the two sub-events is problematic, as has been discussed in Section 2.2.1; for this work it is more appropriate to use only one event—say (e_2)—and indicate that multiple instances of this exist. Using distinct names for the two sub-events in this example allows us to refer to them separately; this is not something we will wish to do in general when talking about similar sub-activity entities.

[5] In this example—just for clarity—I omit predicates like *patient* that might be associated with the stirring events; however in general these are important.

[6] We do note however that these relations may be of interest in other types of temporal analysis, such as that of temporal ordering done by How[18].

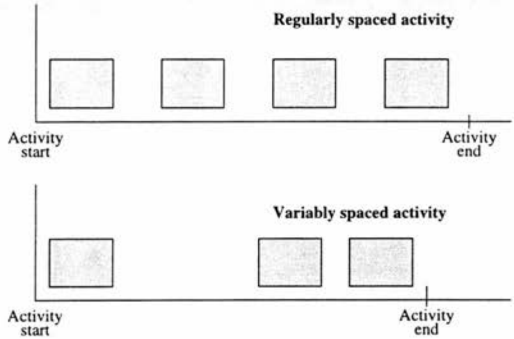


Figure 5.1: Diagrammatic representation of temporal spacing of sub-activity.

from each other may be constant, or it may be variable. These distinctions are illustrated in Figure 5.1, and are represented by a feature called [**temporal-spacing=**], whose values may be [=**regular**] or [=**unfixed**]. For regularly spaced instances, there may also be a measure of the actual distance.

- There is another temporal distinction that is important. It is possible for multiple instances of activity to occur simultaneously, to overlap each other or to be completely distinct from each other.

  The distinctions that are important are whether subactivities occur simultaneously, or consecutively, or whether their temporal relation in these terms is not important. These distinctions are illustrated in Figure 5.2; we will see in Section 5.3 that they correspond to the different kinds of set structure that have been identified as [=**simultaneous-set**], [=**ordered-set**] and [=**ordinary-set**]. The example of (5.3) may be represented by the feature structure in (5.5).

$$
(5.5)\qquad \left[\begin{array}{ll}
\textsf{structure} = \textsf{ordered-set} & \\
\textsf{index} = e_1 & \\
\textsf{extent} = & \left[\begin{array}{l} \textsf{temporal-spacing} = \textsf{unfixed} \\ \textsf{cardinality} = \textsf{two} \end{array}\right] \\
\textsf{content} = & \left[\begin{array}{ll} \textsf{structure} = \textsf{happening} & \\ \textsf{index} = e_2 & \\ \textsf{type} = \textsf{discrete} & \\ \textsf{substance} = & \left[\begin{array}{l} \textsf{action} = \textsf{stir} \\ \textsf{patient} = \textsf{soup} \end{array}\right] \end{array}\right]
\end{array}\right]
$$

It is sometimes the case that the language indicates whether the temporal distance or temporal relation between activities is significant. The approach that is taken in this thesis is the following:

- If the language indicates that it is important, then it is. That is, the language may explicitly, or insistently[7], describe a temporal structure.
- If the language does not indicate an actual temporal distance or temporal relation, then the only way to determine it is from context or from world knowledge, and so we simply leave it empty, or unspecified.

I clarify this with some examples. In (5.6a), there is a regular spacing of basting happenings, while in (5.6b) it is not made explicit whether or not the bastings are to occur at regular intervals. However, it *is* explicit that the bastings are to occur consecutively, while in (5.6c) the bakings are to occur simultaneously. In (5.6d), we do not know at all how each onion chopping relates temporally to any other onion chopping.

(5.6) a. Baste the roast every ten minutes.

b. Baste the roast three times using a spoon.

c. Bake both cakes together in a hot oven for 30 minutes.

[7] See Section 4.2.2 for clarification of the use of the word INSISTENT.

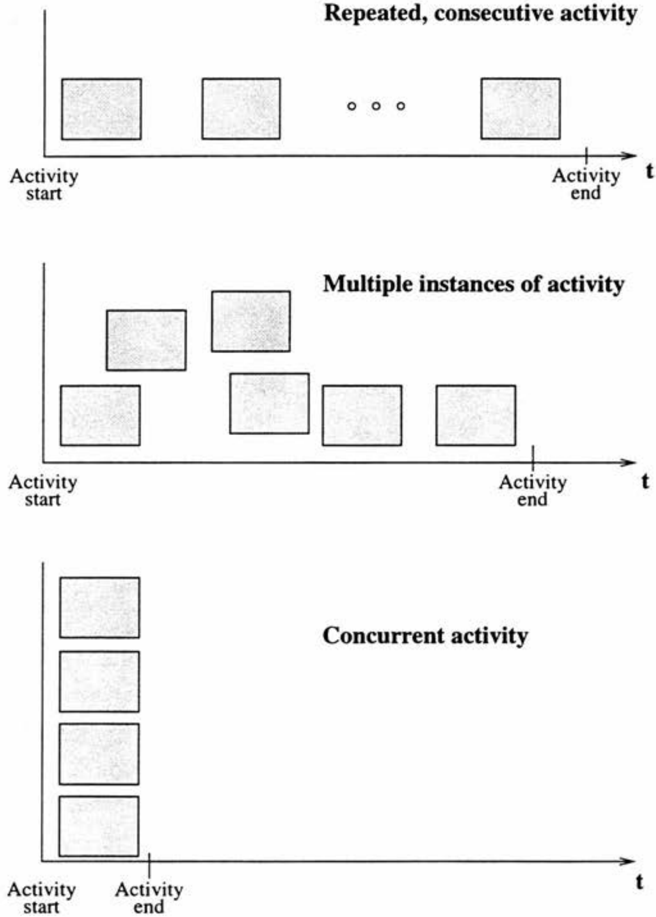


Figure 5.2: Diagrammatic representation of temporal relations between activities.

d. Chop six onions.

These distinctions are accounted for in our characterisation of activity structure by the use of features. In the representations and implementation that are described in the rest of this thesis, temporal relations are coded via structurings; temporal distance has not been incorporated into the computational implementation[8], but the appropriate places for its coding are indicated.

5.2 Representing happenings

From the previous chapter, in particular Section 4.2.1, it is clear that some information about the kind of activity that a verb contributes to, comes from the verb itself. In particular, we have distinguished, in principle, between extended activity that is 'masslike' and extended activity that is 'countlike'. We now make these distinctions more concrete, and examine the effects they produce.

5.2.1 Verb types

We have noted that there are certain verbs that, when the action they are instructing is realised, result in extended activity, and that this is a function of the action that the verb denotes. This is true of verbs like '*sleep*'. Other verbs, such as '*notice*', do not bring with them an association of extended activity. Finally, there are verbs like '*tap*' and '*flash*' which sometimes are viewed as representing a single instance (such as ≪*one tap*≫) and sometimes as representing an iteration of activity (such as ≪*tapping*≫). To capture these distinctions, I categorise verbs into one of three classes, using the feature [`verbclass=`]—

[`=masslike`] —these are those verbs whose underlying activity is one that we think of as continuous; this is the eventuality analogue of mass, and we imagine activity that, when we consider only the verb itself, is not delimited. Examples include '*sleep*', '*bake*', '*simmer*'.

[8] This simplification is justified by the fact that this work is not about the distinctions between things like the meanings of '*often*', '*occasionally*', '*every week*', etc., but it is rather about the kinds of extended activity structures that there are.

[=`simple`] —these are verbs whose underlying activity we conceive of as discrete; this is the eventuality analogue of count objects, and covers activity that is bounded. We imagine a completed, telic, event when we consider the verb. Examples include '*put*', '*shut*'.

[=`iterable`] —this class of verbs is akin to objects like '*rice*', which may sometimes be seen as mass and sometimes be seen as count, depending on context. It is possible for the activity they produce to be an 'iterated' execution of the basic action[9]. '*Beat*', '*peel*', '*stir*' and '*tap*' are examples.

The classes [=`masslike`] and [=`simple`] are analogous to mass and count nouns respectively, and these are the two primary categories. The class [=`iterable`] we identify as pertaining to verbs that can potentially 'go either way'. This covers group of verbs that, depending on their context of use—both linguistic and execution—may describe activity that could be seen as mass-like or activity that could be seen as count-like.

I note that while it would be very convenient to be able to categorise each verb into one of three types as above, this turns out to be a difficult exercise. Many authors have commented on such enterprise—when discussing aspectual class, Moens & Steedman [28] have been very clear in saying that it is not verbs that are being categorised but meanings of sentences. However, they also say that only certain verbs will be part of a sentence that has a particular aspectual category. According to Moens & Steedman, "... aspectual profiles are properties of sentences *used in a context*: sense-meanings of sentences or verbs in isolation are usually compatible with several ... profiles"[28, pg 17]. However, in order to propose any pragmatic approach, "... we have included ... examples of verbs which *typically* yield propositions of the relevant types, and we shall assume that such verbs ... are lexically specified as bearing that type"[ibid]. Taking this approach, but with the massness and countness as the property being classified for, and including a means of overriding classification in certain circumstances, we get a pragmatic approach which allows us to make progress in the understanding endeavour.

[9] I note that the certainty with which I claim that say '*peel*' is intrinsically iterable is somewhat fragile. That is to say: that the activities arising from some verbs are inherently iterable is certain, but that '*peel*' is one such example is open to discussion.

I believe that the above approach provides a mechanism for coding different views or perspectives. It is up to the particular 'understander' to decide which verbs are masslike, which are countlike, and which are iterable—using information about how the language allows them to be used, and also about the actual world and execution context in which that understander is to operate. The lexicon that, say, I choose as representing the meanings of verbs reflects my particular understanding of those verbs. The overall approach however, is one that allows for these different *kinds* of verbs.

Representing verbs

Representation of verbs involves representing their syntactic features—[`verbcat`=] for example distinguishes between intransitive, transitive and ditransitive verbs; [`category`=] refers to the surface syntactic category—as well as their semantic categories. The feature [`subst`=] refers to the actual substance of the activity that the verb represents; it is akin to its 'meaning'. The feature [`verbclass`=] encodes the class of verb with respect to its massness or countness, as described in the previous section.

The following show representations for the lexical items '*stir*', '*sleep*' and '*put*' respectively. The feature [`dict`=] indicates a dictionary entry; its value is the encoding of the corresponding lexical item.

(5.7) a.

$$\begin{bmatrix} \text{dict} = \begin{bmatrix} \text{category} = \text{verb} \\ \text{subst} = \text{stir} \\ \text{verbclass} = \text{iterable} \\ \text{verbcat} = \text{transitive} \end{bmatrix} \end{bmatrix}$$

b.

$$\begin{bmatrix} \text{dict} = \begin{bmatrix} \text{category} = \text{verb} \\ \text{subst} = \text{sleep} \\ \text{verbclass} = \text{masslike} \\ \text{verbcat} = \text{intransitive} \end{bmatrix} \end{bmatrix}$$

$$\text{c.}\quad \left[\text{dict} = \begin{bmatrix}\text{category} = \text{verb}\\ \text{subst} = \text{put}\\ \text{verbclass} = \text{simple}\\ \text{verbcat} = \text{ditransitive}\end{bmatrix}\right]$$

5.2.2 Happening types

Using the analogy with plural and mass objects, we have two happening types. When a verb is combined with a patient or patients (if appropriate), we have a happening that may be masslike—which we call CONTINUOUS and categorise as [type=continuous]—or may be countlike—which we call DISCRETE and categorise as [type=discrete]. It is also possible that the type of a happening will be underspecified[10] coming from a verb that is [=iterable], and will retain this value.

To illustrate with examples, the happening described by (5.8a) is [=continuous], because it comes from a verb that is [verbclass=masslike]. Similarly, the happening described by (5.8b) is [=discrete], as its verb is [verbclass=simple]. In contrast, the happening type in (5.8c) cannot be fully determined; its underlying verb has [verbclass=iterable], and so it may require more contextual information to determine which sense of ≪*stirring*≫—continuous or discrete—is being used. Feature structures for the examples in (5.8) are shown in (5.9). Each feature structure has [structure=happening], and an associated unique [index=]. The feature [type=] encodes the happening type, which may be [=discrete], [=continuous], or [=iterable]. The feature [=substance] holds a feature structure that represents the verb itself and any complements that the verb might take.

(5.8) a. Bake the cake.

b. Put the cake in the oven.

c. Stir the batter.

[10] This happens when further information such as the combining with modifiers is still required.

(5.9)

a.

$$\begin{bmatrix} \textsf{structure} = \textsf{happening} \\ \textsf{index} = e_1 \\ \textsf{type} = \textsf{continuous} \\ \textsf{substance} = \begin{bmatrix} \textsf{action} = \textsf{bake} \\ \textsf{patient} = \textsf{cake} \end{bmatrix} \end{bmatrix}$$

b.

$$\begin{bmatrix} \textsf{structure} = \textsf{happening} \\ \textsf{index} = e_2 \\ \textsf{type} = \textsf{discrete} \\ \textsf{substance} = \begin{bmatrix} \textsf{action} = \textsf{put} \\ \textsf{patient} = \textsf{cake} \\ \textsf{recipient} = \textsf{oven} \end{bmatrix} \end{bmatrix}$$

c.

$$\begin{bmatrix} \textsf{structure} = \textsf{happening} \\ \textsf{index} = e_3 \\ \textsf{type} = \textsf{iterable} \\ \textsf{substance} = \begin{bmatrix} \textsf{action} = \textsf{stir} \\ \textsf{patient} = \textsf{batter} \end{bmatrix} \end{bmatrix}$$

In Section 6.2 we will see the grammar rules for obtaining semantic representations for happenings, which act on the [`verbclass=`] feature.

5.3 Representing structurings

Using the basic elements of discrete and continuous happenings described in Section 5.2, and the notion of activity being compositional, we now look at ways in which activities may be composed.

We identify two basic kinds of composition—sets and masses[11]. There are, broadly, two ways in which chunks of similar activity can be grouped together.

- First, we can have multiple instances of some activity, where each instance

[11] This is closely analogous to the notion in objects of count plurals and masses.

of the activity is similar to the others. The temporal relations between these multiple instances may take a variety of forms, but each instance is distinct[12].

- Secondly, we may have a mass of similar activity, where any part of that activity is much the same as any other part of that activity and the activity is continuous—or 'contiguous in time' in Zemach's[57] terminology. Sleeping is an example of such activity; we do not have places that are not ≪*sleeping*≫ within a sleeping activity, without thinking that we have different sleeping activities.

We consider multiple similar happenings to be a SET (of discrete activities) and we consider a mass of similar happening to be a MASS (of continuous activity).

It is helpful to bring in the analogy that iterative and continuous actions are the eventuality analogue of count and mass nouns. With objects, it is both the object itself, and the context in which it exists, that informs whether it is a count or mass object. For example, water will always be a mass object, while an apple is a count object[13]. Further, though water is a mass object, a glass of water becomes a count object, but the water in the glass is still a mass object.

A similar analysis can be applied to activity. Sleeping is always a continuous action that is extended in time. Sleeping for half an hour is the eventuality analogue of a glass of water—it describes a discrete eventuality, but the activity within it is still a continuous eventuality. But in tapping, each tap (that makes up a tapping activity that is extended in time) has its own start, duration (albeit very short) and end, and the tapping is thus iterated. Tapping for five minutes is analogous to an ounce of lentils.

With objects like lentils, it is possible to see them as discrete items that form a mass. We can talk about '*five lentils*'—these are count objects—or we can talk about '*an ounce of lentils*'—this is a mass object. We apply the same approach to eventualities. Therefore, we take the view that '*Stir the soup until it is boiling*' describes a delimited mass eventuality; '*Stir the soup a few times*' describes an eventuality that is an ordered set of a few ≪*stir the soup (once)*≫ happenings.

Usually, mass eventualities are delimited in sentences that contain duration modi-

[12] Though not necessarily temporally distinct, as was clarified in Section 3.2.2.

[13] It is however true that there are situations where apple can be a mass noun.

fiers, such as '*for 10 mins*', '*until it boils*', or time complements, such as '*overnight*', '*a while*'. The underlying happening does not take place a definite number of times, but rather over a period of time or until some condition becomes true.

In addition to sets and masses, there are ways of combining non-similar activities which are relevant to the discussion here; we refer to this as COMPLEX activity. Further, structured activity may consist of combinations of activity that is itself structured as a set or mass, or as complex activity. That is, the underlying constituents of sets or masses may be sets, masses or other structured activity.

We now look in detail at sets, masses and combined activity composition.

5.3.1 Sets

An extended activity that is a set is a bagful of sub-activities or happenings that are similar to each other. In general, the elements of sets will be discrete activities, although there are some exceptions to this which will be described later. We also need to distinguish between sets where the temporal relation between elements is important and those where it is not. The distinctions introduced in Section 5.1.2 are important here—whether elements of the set follow each other temporally (are consecutive), whether elements of the set are simultaneous, or whether the temporal relation is not fixed to be either of these by the language.

The feature structures that represent sets have four features:

`[structure=]` encodes the kind of set structuring, and can be `[=ordered-set]`, `[=simultaneous-set]` or `[=ordinary-set]`.

`[index=]` has the unique index label that identifies this activity.

`[extent=]` represents the information about the extent of the set. Information about number of instances (for multiple instances), frequency, temporal spacing, etc are part of this.

`[content=]` contains a feature structure representing the sub-activity of which the set is multiple instances. This may be a happening, or it may itself be a structured entity.

Ordered sets

These consist of activities that occur consecutively—the elements in an ordered set cannot occur simultaneously. Examples come from sentences that express exactly how many times an action is to be done, or how frequently. They may have a frequency—which answers the question "how often is the activity to be performed?"—or a cardinality—which answers the question "how many times is the activity to be performed?".

Meaning: A collection of like sub-actions, that is to be executed sequentially. The only thing that differs between actions is the time when they are done. Any two elements of an ordered set are temporally distinct, and have a non-zero temporal distance between them.

The sentence '*Baste twice using a spoon*' describes an extended activity that is an ordered set, containing two activities. Each of these two activities is a happening that is discrete. A semantic representation of this sentence is the following[14]:

$$\left[\begin{array}{l}
\mathsf{structure} = \mathsf{ordered\text{-}set} \\
\mathsf{index} = e_1 \\
\mathsf{extent} = \left[\begin{array}{l} \mathsf{temporal\text{-}spacing} = \mathsf{unfixed} \\ \mathsf{cardinality} = 2 \end{array}\right] \\
\mathsf{content} = \left[\begin{array}{l}
\mathsf{structure} = \mathsf{happening} \\
\mathsf{index} = e_2 \\
\mathsf{type} = \mathsf{discrete} \\
\mathsf{substance} = \left[\begin{array}{l} \mathsf{action} = \mathsf{baste} \\ \mathsf{how} = \gg \mathit{using\ a\ spoon} \ll \end{array}\right]
\end{array}\right]
\end{array}\right]$$

Simultaneous sets

These consist of activities that occur simultaneously—the elements in a simultaneous set all occur at the same time. Examples come from sentences that express

[14] The details of non temporal, non measure modifiers like '*using a spoon*' are not analysed in any depth; this example simply indicates where the results of such analysis would fit.

quantification over spatial location, such as '*(At 8pm) a bell rang in every room*'.

Meaning: A collection of similar sub-actions, that is to be executed simultaneously. The temporal distance between any two elements of a simultaneous set is zero and they are spatially distinct.

The sentence '*Bake both of the cakes together for 30 minutes*' describes an extended activity that is a simultaneous set, containing two activities. Each of these two activities is a happening that is a delimited mass[15] of baking. A semantic representation of this sentence is the following:

$$
\left[\begin{array}{ll}
\text{structure} = \text{simultaneous-set} & \\
\text{index} = e_1 & \\
\text{extent} = & \left[\begin{array}{l} \text{cardinality} = 2 \\ \text{var} = i \\ \text{restr} = \text{cake} \end{array}\right] \\
\text{content} = & \left[\begin{array}{ll}
\text{structure} = \text{delimited-mass} & \\
\text{index} = e_2 & \\
\text{extent} = & \left[\begin{array}{l} \text{extent-type} = \text{time} \\ \text{start} = \text{now} \\ \text{end} = 30\ \text{minutes} \end{array}\right] \\
\text{content} = & \left[\begin{array}{ll}
\text{structure} = \text{happening} & \\
\text{index} = e_3 & \\
\text{type} = \text{continuous} & \\
\text{substance} = & \left[\begin{array}{l} \text{action} = \text{bake} \\ \text{patient} = i \end{array}\right]
\end{array}\right]
\end{array}\right]
\end{array}\right]
$$

The notation presented in the feature [`extent=`] using [`var=`] and [`restr=`] corresponds to the notion that some activity structure comes from the presence of quantified objects in the verb phrase. The details of these, and how they relate to the overall representations, is presented in Section 5.4.

[15] In Section 5.3.2, delimited masses are discussed in detail.

Ordinary sets

Here there are no conditions attached to the temporal ordering of the constituent elements of the set.

Actions are distinct, but may—at execution time—take place simultaneously, consecutively, in some random (undefined) ordering, or a mix of all of these. Extended activity of this sort usually pertains to activity involving plural objects. The structure of the semantic representation is similar to that of an ordered set. There may be a cardinality, and a related object. Here, the question is "how many times" or "on how many objects" is the action to be performed?

Meaning: A collection of similar actions, to be executed. The order or groupings in which the actions get executed is not given, and it will thus be the case that more than one correct execution sequence will be possible.

The sentence '*Top and tail the gooseberries*' describes an extended activity that is an ordinary set, containing as many activities as there are gooseberries. Each of these activities, the ≪*topping and tailing of one gooseberry*≫, is a happening that is discrete. Because we don't know how many gooseberries there are, we cannot determine the actual cardinality of the set, but we do know that it is related to the number of gooseberries that will be there when the instruction is actually being executed. A semantic representation of the sentence is the following:

$$\begin{bmatrix} \textsf{structure} = \textsf{ordinary-set} \\ \textsf{index} = e_1 \\ \textsf{extent} = \begin{bmatrix} \textsf{temporal-spacing} = \_ \\ \textsf{cardinality} = \textit{number-of-gooseberries} \\ \textsf{var} = i \\ \textsf{restr} = \textsf{gooseberry} \end{bmatrix} \\ \textsf{content} = \begin{bmatrix} \textsf{structure} = \textsf{happening} \\ \textsf{index} = e_2 \\ \textsf{type} = \textsf{discrete} \\ \textsf{substance} = \begin{bmatrix} \textsf{action} = \textsf{top\_and\_tail} \\ \textsf{patient} = i \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

I note that the symbol [=_] represents an empty feature value—that is the absence of any value, as opposed to a null value of some kind. In an ordinary set, the temporal spacing between elements may only become known once the execution context is available.

5.3.2 Masses

An extended activity that is a mass is protracted contiguous execution of similar activity. In general, a mass will be delimited; usually the delimitation will be explicit although it is possible for the delimitation to be implied[16].

The feature structures representing masses have four features:

[`structure=`] encodes the kind of set structuring, and is [`=delimited-mass`].

[`index=`] has the unique index label that identifies this activity.

[`extent=`] represents the information about the extent, or delimitation of the mass. Information about duration in time, or stopping conditions and states,

[16] As discussed in Section 3.1, a bare mass term, like a bare plural, refers to something that is a pure continuant, and is not within the scope of this discussion. If we say '*Jo slept*' or '*Bake the cake*', we really mean that ≪*Jo slept for some amount of time*≫ and that ≪*the cake is to be baked for some amount of time*≫. This is just the same as saying '*Drink water*' and meaning ≪*Drink some water*≫ rather than ≪*Drink all of water*≫.

etc. are part of this.

[`content=`] contains a feature structure representing the sub-activity of which the mass is a protracted execution. This may be a happening, or it may itself be a structured entity.

Delimited masses

This pertains to extended activity that is protracted execution of similar activity. There is a primary action, which either gets extrapolated to occur continuously or repeatedly over a time duration, or an action that has a start time and an end time. At any time during the activity's execution, the content or substance of the activity is the same as at any other time during that same execution.

Meaning: Continued execution of an activity; the overall perspective is one where all sub-parts of the extended activity are similar to all other sub-parts of the activity in different temporal, or spatial, locations. The underlying activity may have come from an iterable verb[17]; more usually it will come from a mass verb. The activity type of the happening is continuous.

The sentence '*Bake the cake for 30 minutes*' describes an extended activity that is a delimited mass, containing a mass activity of baking a particular cake, denoted by ≪*the cake*≫. A semantic representation of this sentence is given in (5.10a). In contrast, '*Bake the cake*' describes a mass that is not delimited; its representation is given in (5.10b).

[17] In this case the timing between sub-activity is assumed to be regular, or of no importance, unless there is a qualifier indicating otherwise.

(5.10) a.

$$\begin{bmatrix} \text{structure} = \text{delimited-mass} & \\ \text{index} = e_1 & \\ \text{extent} = & \begin{bmatrix} \text{extent-type} = \text{time} \\ \text{start} = \text{now} \\ \text{end} = \text{30 minutes} \end{bmatrix} \\ \text{content} = & \begin{bmatrix} \text{structure} = \text{happening} & \\ \text{index} = e_2 & \\ \text{type} = \text{continuous} & \\ \text{substance} = & \begin{bmatrix} \text{action} = \text{bake} \\ \text{patient} = \text{cake} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

b.

$$\begin{bmatrix} \text{structure} = \text{happening} & \\ \text{index} = e_2 & \\ \text{type} = \text{continuous} & \\ \text{substance} = & \begin{bmatrix} \text{action} = \text{bake} \\ \text{patient} = \text{cake} \end{bmatrix} \end{bmatrix}$$

The sentence '*Stir the syrup until all the sugar has dissolved*' describes an extended activity that is a delimited mass, containing an iteration of stirring activity. A semantic representation of this sentence is the following[18]:

[18] The details of conditions like '*all the sugar has dissolved*' are not analysed in any depth; all that is identified is that there is a condition dependent on some state.

$$\begin{bmatrix} \text{structure} = \text{delimited-mass} & \\ \text{index} = e_1 & \\ \text{extent} = & \begin{bmatrix} \text{extent-type} = \text{condition} \\ \text{start} = \text{now} \\ \text{end} = \gg \textit{all the sugar has dissolved} \ll \end{bmatrix} \\ \text{content} = & \begin{bmatrix} \text{structure} = \text{happening} & \\ \text{index} = e_2 & \\ \text{type} = \text{continuous} & \\ \text{substance} = & \begin{bmatrix} \text{action} = \text{stir} \\ \text{patient} = \text{syrup} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

The verb '*stir*' is an [=`iterable`] one; because it is being used as the basis of a delimited mass, it takes on the continuous reading of stirring. Contrast this with the feature structure in (5.5), where '*stir*' is used as the basis of a set, and it takes on the discrete reading.

5.3.3 Conjunctive structurings

These are extended activities, usually delimited mass, where the extent of the activity is expressed in two ways. Again, the feature structure will have four features:

[`structure=`] is [=`conjunctive`].

[`index=`] has the unique index label that identifies this activity.

[`extent=`] represents the information about the extent of the conjunctive activity. This is a feature structure that itself has two features—[`extent1=`] and [`extent2=`]—each of which represents one of the descriptions of activity extension. Although these features can themselves contain structure they do not represent sub-activity. Usually, the first will represent a time period, and the second will represent a state or condition that delimits the extent of execution.

[`content=`] contains a feature structure representing the sub-activity that constitutes the delimited mass, or set. This is usually a continuous happening, although it may be a structured entity or a discrete happening.

Meaning: There is only one action here, but two different expressions of its extended execution. The extent of execution is determined from either expression[19], and if both expressions can be satisfied[20] when the execution is terminated, then the execution is deemed to be successful.

The sentence '*Bake the cake for one hour or until it is firm*' describes an activity that is a delimited mass of baking; the delimitation is described in two ways. So, the baking takes place ≪*for an hour, or until it is firm, whichever happens first*≫. The meaning of these conjunctions was discussed in Section 4.5.2. It is likely that '*it*' refers to the cake, and it is once the cake has become firm that the baking ends. However, such anaphora resolution is not part of the analysis presented here. In general, these extended activities will be delimited masses; however it is possible for ordered sets to be expressed in this way.

The following feature structure represents '*Bake the cake for one hour or until it is firm*'.

---

[19] The shorter extent is usually used; however it is important to note that this can often only be decided at execution time.

[20] The word satisfied is used loosely, to mean the condition it expresses can roughly be said to be fulfilled.

$$
\left[\begin{array}{l}
\text{structure} = \text{conjunctive} \\
\text{index} = e_2 \\
\text{extent} = \left[\begin{array}{l}
\text{extent1} = \left[\begin{array}{l}
\text{structure} = \text{delimited-mass} \\
\text{extent} = \left[\begin{array}{l}
\text{extent-type} = \text{time} \\
\text{start} = \text{now} \\
\text{end} = \text{one hour}
\end{array}\right]
\end{array}\right] \\
\text{extent2} = \left[\begin{array}{l}
\text{structure} = \text{delimited-mass} \\
\text{extent} = \left[\begin{array}{l}
\text{extent-type} = \text{condition} \\
\text{start} = \text{now} \\
\text{end} = \gg \textit{it is firm} \ll
\end{array}\right]
\end{array}\right]
\end{array}\right] \\
\text{content} = \left[\begin{array}{l}
\text{structure} = \text{happening} \\
\text{index} = e_4 \\
\text{type} = \text{continuous} \\
\text{substance} = \left[\begin{array}{l}
\text{action} = \text{bake} \\
\text{patient} = \text{cake}
\end{array}\right]
\end{array}\right]
\end{array}\right]
$$

5.3.4 More complex compositions

There are various ways in which activities themselves may be combined to produce new activities.

In all of the representations of extended activity described here, there is a feature [`content=`]. The value of this feature may at any point be a structured activity. So, the [`content=`] of a feature that has [`structure=ordered-set`] may for example itself be [`structure=delimited-mass`], representing an ordered set of delimited masses, such as described by '*Visit your friend for half an hour every week*'.

In addition to sets and masses whose underlying activities are themselves structured, rather than being simple happenings, there are combinations of different activity sorts.

Compound

This is not extended activity of the kind this thesis focuses on, but is included to allow for general composition of activity. I use the term COMPOUND to refer to activity that consists of two (or more) unsimilar sub-activities or happenings. This is useful in some of the other complex compositions.

A feature structure representing a compound activity will have

`[structure=]` is `[=compound]`.

`[index=]` has the unique index label that identifies this activity.

`[participants=]` which is a collection of the sub-activities of the compound activity. These will themselves each be a feature structure representing an extended activity or a happening.

Meaning: There are two or more different activities that are linked in some way, perhaps through the use of '*and*', or through a discourse. The execution of the two activities follows each other. We assume for now that the order is the same as the order the activities are described in the language.

So, in an instruction like '*Skim off the foam and discard it*', we can identify a composition of two happenings—≪*skimming off the foam*≫ and ≪*discarding the foam*≫. The following feature structure represents this.

$$
\left[\begin{array}{ll}
\text{structure} = \text{compound} & \\
\text{index} = e_1 & \\
\text{participants} = & \left[\begin{array}{ll}
\text{participant1} = & \left[\begin{array}{ll}
\text{structure} = \text{happening} & \\
\text{index} = e_2 & \\
\text{type} = \text{discrete} & \\
\text{substance} = & \left[\begin{array}{l} \text{action} = \text{skim-off} \\ \text{patient} = \text{foam} \end{array}\right]
\end{array}\right] \\
\text{participant2} = & \left[\begin{array}{ll}
\text{structure} = \text{happening} & \\
\text{index} = e_3 & \\
\text{type} = \text{discrete} & \\
\text{substance} = & \left[\begin{array}{l} \text{action} = \text{discard} \\ \text{patient} = \mathit{referent} \end{array}\right]
\end{array}\right]
\end{array}\right]
\end{array}\right]
$$

It is assumed that that the participants in a compound activity occur consecutively, with regard to time.

Concurrent

This is a complex composition that consists of two extended activities—that is, of two eventualities that themselves are not basic happenings: a major one (usually mass type) and a minor one (usually an ordered set type).

The feature structure for concurrent activities is similar to the one for compound activities, except that the [`participants=`] feature contains only two features—one for the major activity and one for the minor.

The feature structure for a concurrent activity will have the following features:

[`structure=`] is [`=concurrent`].

[`index=`] has the unique index label that identifies this activity.

[`participants=`] has two features—[`major=`] and [`minor=`] which contain feature structures representing the corresponding extended activities.

Meaning: There are two different activities (say stirring and basting), each of which has its own extended framework, of a set, or mass, as described above. The execution of the two activities occurs at the same time; the start and end times of the concurrent activity—that is, of both the major and minor actions—are the start and end times of the major action.

The sentence '*Simmer the soup for 15 minutes, checking it every five minutes*' describes an extended activity that contains two structured activities—the major one of ≪*simmering*≫ and the minor one of ≪*checking*≫. A semantic representation of this sentence is the following:

$$
\left[\begin{array}{l}
\textsf{structure} = \textsf{concurrent} \\
\textsf{index} = e_1 \\
\textsf{participants} = \left[\begin{array}{l}
\textsf{major} = \left[\begin{array}{l}
\textsf{structure} = \textsf{delimited-mass} \\
\textsf{index} = e_2 \\
\textsf{extent} = \left[\begin{array}{l}
\textsf{extent-type} = \textsf{time} \\
\textsf{start} = \textsf{now} \\
\textsf{end} = \textsf{15 minutes}
\end{array}\right] \\
\textsf{content} = \left[\begin{array}{l}
\textsf{structure} = \textsf{happening} \\
\textsf{index} = e_4 \\
\textsf{type} = \textsf{continuous} \\
\textsf{substance} = \left[\begin{array}{l}
\textsf{action} = \textsf{simmer} \\
\textsf{patient} = \textsf{soup}
\end{array}\right]
\end{array}\right]
\end{array}\right] \\
\textsf{minor} = \left[\begin{array}{l}
\textsf{structure} = \textsf{ordered-set} \\
\textsf{index} = e_3 \\
\textsf{extent} = \left[\begin{array}{l}
\textsf{temporal-spacing} = \textsf{regular} \\
\textsf{frequency} = \gg \textit{five-minutes} \ll
\end{array}\right] \\
\textsf{content} = \left[\begin{array}{l}
\textsf{structure} = \textsf{happening} \\
\textsf{index} = e_5 \\
\textsf{type} = \textsf{discrete} \\
\textsf{substance} = \left[\begin{array}{l}
\textsf{action} = \textsf{check} \\
\textsf{patient} = \textit{referent}
\end{array}\right]
\end{array}\right]
\end{array}\right]
\end{array}\right]
\end{array}\right]
$$

5.3.5 Representing extent

Almost all of the extended activities (in particular, those whose structure is a set or delimited mass) have a feature representing the extent of the activity. When the extended activity is a set, the extent may represent

- the [`temporal-spacing=`] between elements

- the [`cardinality=`]
- the [`frequency=`].

For masses,

- [`extent-type=`] may be [`=time`], indicating that an actual time period was specified, or [`=condition`], to represent that the extent is related to some condition holding
- there may be a [`start=`] value, and the [`end=`] will be either the time specified or the condition.

5.4 Quantification

As motivated in Section 4.5.1, measure adverbials—such as '*for half an hour*'—are treated as event quantifiers, in much the same way that phrases like '*every week*' and '*three times*' are. In addition, the usual approach to treating nominal quantifiers applies. I therefore now present a representation that covers quantification for all of these.

I note that what is demonstrated here using feature structures is just a straightforward translation of the notion that quantification is ultimately encoded using first-order logic. That is, I have taken quantification—expressed most appropriately in first-order logic—and simply slotted it into a feature structure representation that is in line with the representation used in the discussion that has preceded this. This approach is presented briefly by Pollard & Sag[34, 35] for object quantification; here I extend it and also apply it to event quantification.

5.4.1 Nominal quantification

I note that the explanation in this section is for the purpose of demonstrating the manner in which ordinary nominal quantification can be represented using feature structures, and only in the next sections will I address the issue of event quantification.

I first demonstrate a feature structure representation of a simple quantification example—that in (5.11). This sentence is shown in unscoped form[21] in (5.12); in its two scoped forms, it can be represented in first-order logic as (5.13a) and (5.13b).

(5.11) Every chef bakes a cake.

(5.12) $bake(qterm(\forall, y, chef(y)), qterm(\exists, x, cake(x)))$

(5.13) a. $\exists x(cake(x), \forall y(chef(y) \Rightarrow bakes(y, x)))$

b. $\forall y(chef(y), \exists x(cake(x) \Rightarrow bakes(y, x)))$

Taking the approach advocated by Pollard & Sag, feature structure representations for (5.13) are presented in (5.14)[22]; these are scoped representations. An example of the feature structure for an unscoped representation of an event is shown in Section 5.4.4.

---

[21] In this, an unscoped, 'in-place' quantified term is represented using notation (**qterm**) that indicates that it is a single expression containing the quantifier and its restriction, connected by a variable. So, '*many chefs*' would be represented by the quantified term `qterm(many, X, chef(X))`.

[22] Pollard& Sag would represent the feature structure for a quantifier—of '*a cake*', for example—as

$$\begin{bmatrix} \text{determiner} = \text{exists} \\ \text{restind} = \begin{bmatrix} \text{var} = x \\ \text{restriction} = \begin{bmatrix} \text{reln} = \text{cake} \\ \text{inst} = x \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

(In that this notation distinguished between the restriction and scope of a quantified expression, it is effectively a generalised quantifier notation, like that proposed by Barwise & Cooper[5]. Since I wish to cite examples involving determiners such as '*most*' and '*fifteen*', I require such a notation.)

In order to aid clarity and simplicity, I represent the restriction using the [**restriction=**] feature, without the [**reln=**] and [**inst=**] features, because in the work I present the value for [**inst=**] is always the same as that for [**var=**]. So, the form of quantifier feature structures is

$$\begin{bmatrix} \text{determiner} = \text{exists} \\ \text{var} = x \\ \text{restriction} = \text{cake} \end{bmatrix}$$

(5.14) a.

$$\begin{bmatrix} \text{quantifier} = \begin{bmatrix} \text{determiner} = \text{exists} \\ \text{var} = x \\ \text{restriction} = \text{cake} \end{bmatrix} \\ \text{scope} = \begin{bmatrix} \text{quantifier} = \begin{bmatrix} \text{determiner} = \text{for-all} \\ \text{var} = y \\ \text{restriction} = \text{chef} \end{bmatrix} \\ \text{scope} = \begin{bmatrix} \text{substance} = \text{bake} \\ \text{actor} = x \\ \text{patient} = y \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

b.

$$\begin{bmatrix} \text{quantifier} = \begin{bmatrix} \text{determiner} = \text{for-all} \\ \text{var} = y \\ \text{restriction} = \text{chef} \end{bmatrix} \\ \text{scope} = \begin{bmatrix} \text{quantifier} = \begin{bmatrix} \text{determiner} = \text{exists} \\ \text{var} = x \\ \text{restriction} = \text{cake} \end{bmatrix} \\ \text{scope} = \begin{bmatrix} \text{substance} = \text{bake} \\ \text{actor} = y \\ \text{patient} = x \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

5.4.2 Simple event quantifiers

A primary difference in representing quantification of events—and indeed simply the representing of events in general—is the need for the introduction of an event variable, which is then used for predication.

When we represent something like '*a chef bakes a cake*' from an object perspective, a representation such as those shown in (5.13) is sufficient. However, when we are trying to represent this from an event perspective, we find that we need to introduce something to represent the ≪*baking event*≫ in order to refer to it. So, (5.15) is an example of such representation.

(5.15) $\exists e, x, y(event(e) \wedge cake(x) \wedge chef(y) \wedge actor(e, y) \wedge patient(e, x) \wedge bake(e))$

This then carries through to the feature structure representation, so that structures of the form of (5.16) are appropriate. The innermost feature structure, representing '$actor(e, y) \wedge patient(e, x) \wedge bake(e)$', utilises the notion that it is structurally part of the feature structure for event e.

(5.16)
$$\begin{bmatrix} \text{quantifier} = & \begin{bmatrix} \text{determiner} = \text{exists} \\ \text{var} = e \\ \text{restriction} = \text{event} \end{bmatrix} \\ \text{scope} = & \begin{bmatrix} \text{quantifier} = & \begin{bmatrix} \text{determiner} = \text{exists} \\ \text{var} = x \\ \text{restriction} = \text{cake} \end{bmatrix} \\ \text{scope} = & \begin{bmatrix} \text{quantifier} = & \begin{bmatrix} \text{determiner} = \text{exists} \\ \text{var} = y \\ \text{restriction} = \text{chef} \end{bmatrix} \\ \text{scope} = & \begin{bmatrix} \text{substance} = \text{bake} \\ \text{actor} = x \\ \text{patient} = y \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Anything that is a basic event—a happening—is assumed to be quantified by a basic event quantifier that always takes narrow scope over the description of the happening.

5.4.3 Measure adverbials as quantifiers

We now look at an example of the use of feature structures to represent the semantics of measure adverbials as event quantifiers. At this point it is appropriate to note that the feature structures already demonstrated in Section 5.3 are in a form that is equivalent to the quantified forms described in this section. This equivalence is demonstrated using the sentence of (5.3), repeated here:

(5.3) Stir the soup twice.

In this example, there is only one scoping possible, because there is only one measure adverbial, or quantifier. The feature structure (5.17) shows a direct representation of (5.3), using the usual quantification terminology.

(5.17)

$$\begin{bmatrix} \text{quantifier} = \begin{bmatrix} \text{determiner} = \text{exists} \\ \text{var} = e_1 \\ \text{restriction} = \text{event} \end{bmatrix} \\ \text{structure} = \text{ordered-set} \\ \text{scope} = \begin{bmatrix} \text{quantifier} = \begin{bmatrix} \text{determiner} = 2 \\ \text{var} = e_2 \\ \text{restriction} = \text{event} \end{bmatrix} \\ \text{structure} = \text{happening} \\ \text{scope} = \begin{bmatrix} \text{substance} = \begin{bmatrix} \text{action} = \text{stir} \\ \text{patient} = \text{soup} \end{bmatrix} \\ \text{type} = \text{discrete} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

However, this structured event is of the same form as that shown for (5.3), whose feature structure is shown in (5.5), repeated here. So, we already have available the machinery for representing it as quantification. The feature [**extent=**] performs the quantification role, and the feature [content=] is analogous to the scope. The [index=] feature can be obtained from the event variable, and allows reference to sub-events when appropriate.

$$
\left[\begin{array}{l}
\textsf{structure} = \textsf{ordered-set}\\
\textsf{index} = e_1\\
\textsf{extent} = \left[\begin{array}{l}\textsf{temporal-spacing} = \textsf{unfixed}\\ \textsf{cardinality} = \textsf{two}\end{array}\right]\\
\textsf{content} = \left[\begin{array}{l}\textsf{structure} = \textsf{happening}\\ \textsf{index} = e_2\\ \textsf{type} = \textsf{discrete}\\ \textsf{substance} = \left[\begin{array}{l}\textsf{action} = \textsf{stir}\\ \textsf{patient} = \textsf{soup}\end{array}\right]\end{array}\right]
\end{array}\right] \tag{5.5}
$$

5.4.4 Representation prior to scoping

Until now, all the representations that have been shown are of feature structures that are already in scoped form. However, these representations are obtained by applying a scoping algorithm to an unscoped representation. An unscoped representation has 'in-place' quantified terms that are marked as such by the use of special features.

Event quantifiers in an unscoped form are collected into a list of [`modifiers`=], at the end of the basic event that they are modifying. Each modifier is simply represented as such; its quantification role comes from the fact that it is placed into the list of modifiers.

Object quantifiers in an unscoped form are those patients and recipients of the action of the basic event, that have a feature [`term`=][23], which has a [`det`=], a [`var`=] and a [`restr`=], corresponding to the determiner, variable and restriction of the generalised quantifier notation.

The sentence in (5.18) has two temporal measure modifiers and one object quantifier; the sentence in unscoped form would be represented as (5.19), where the list of modifiers represents the in-place event quantifiers.

(5.18) Stir some soup occasionally for half an hour.

[23] The feature is called term because it represents a quantified term.

(5.19)

$$
\begin{bmatrix}
\text{structure} = \text{happening} \\
\text{index} = \text{e2} \\
\text{type} = \text{iterable} \\
\text{substance} = \begin{bmatrix}
\text{action} = \text{stir} \\
\text{patient} = \begin{bmatrix}
\text{term} = \begin{bmatrix}
\text{det} = \text{some} \\
\text{var} = v \\
\text{restr} = \text{soup}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix} \\
\text{modifiers} = \begin{bmatrix}
\text{first} = \begin{bmatrix}
\text{structure} = \text{delimitedmass} \\
\text{index} = \text{e5} \\
\text{extent} = \begin{bmatrix}
\text{extent-type} = \text{time} \\
\text{term} = \begin{bmatrix}
\text{det} = \text{exists} \\
\text{var} = w \\
\text{restr} = \text{hours}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix} \\
\text{rest} = \begin{bmatrix}
\text{first} = \begin{bmatrix}
\text{structure} = \text{ordered-set} \\
\text{index} = \text{e12} \\
\text{extent} = \begin{bmatrix}
\text{det} = \text{occasional} \\
\text{var} = x \\
\text{restr} = \text{event}
\end{bmatrix}
\end{bmatrix} \\
\text{rest} = \text{end}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}
$$

5.4.5 Scoped representations

In the example of (5.20b), which is the representation of the sentence in (5.20a), there are two interacting event quantifiers, resulting in two possible scopings. The more natural one is that of ≪*stirring occasionally*≫ happening through a period of ≪*half an hour*≫. This is demonstrated in (5.21). However, another possible reading is that we have ≪*occasionally*≫ the ≪*stirring of soup for half an hour*≫, as shown in (5.22).

(5.20) a. Stir the soup occasionally for half an hour.

b.

$$
\left[\begin{array}{ll}
\mathsf{structure} = \mathsf{happening} & \\
\mathsf{index} = e_1 & \\
\mathsf{substance} = & \left[\begin{array}{l} \mathsf{action} = \mathsf{stir} \\ \mathsf{patient} = \mathsf{soup} \end{array}\right] \\
\mathsf{type} = \mathsf{iterable} & \\
\mathsf{modifiers} = & \left[\begin{array}{ll}
\mathsf{first} = & \left[\begin{array}{ll}
\mathsf{structure} = \mathsf{delimitedmass} & \\
\mathsf{index} = e_2 & \\
\mathsf{extent} = & \left[\begin{array}{ll}
\mathsf{term} = & \left[\begin{array}{l} \mathsf{det} = \mathsf{exists} \\ \mathsf{var} = w \\ \mathsf{restr} = \mathsf{hours} \end{array}\right] \\
\mathsf{extent\text{-}type} = \mathsf{time} &
\end{array}\right]
\end{array}\right] \\
\mathsf{rest} = & \left[\begin{array}{ll}
\mathsf{first} = & \left[\begin{array}{ll}
\mathsf{structure} = \mathsf{ordered\text{-}set} & \\
\mathsf{index} = e_3 & \\
\mathsf{extent} = & \left[\begin{array}{l} \mathsf{det} = \mathsf{occasional} \\ \mathsf{var} = x \\ \mathsf{restr} = \mathsf{event} \end{array}\right]
\end{array}\right] \\
\mathsf{rest} = \mathsf{end} &
\end{array}\right]
\end{array}\right]
\end{array}\right]
$$

$$\left[\begin{array}{ll}
\textsf{structure} = \textsf{delimited-mass} & \\
\textsf{index} = e_2 & \\
\textsf{extent} = & \left[\begin{array}{l}\textsf{end} = \textsf{for-half-hour}\end{array}\right] \\
\textsf{content} = & \left[\begin{array}{ll}
\textsf{structure} = \textsf{ordered-set} & \\
\textsf{index} = e_3 & \\
\textsf{extent} = & \left[\begin{array}{l}\textsf{temporal-spacing} = \textsf{unfixed} \\ \textsf{frequency} = \textsf{occasionally}\end{array}\right] \\
\textsf{content} = & \left[\begin{array}{ll}
\textsf{structure} = \textsf{happening} & \\
\textsf{index} = e_1 & \\
\textsf{type} = \textsf{discrete} & \\
\textsf{substance} = & \left[\begin{array}{l}\textsf{action} = \textsf{stir} \\ \textsf{patient} = \textsf{soup}\end{array}\right]
\end{array}\right]
\end{array}\right]
\end{array}\right] \tag{5.21}$$

$$\left[\begin{array}{ll}
\textsf{structure} = \textsf{ordered-set} & \\
\textsf{index} = e_3 & \\
\textsf{extent} = & \left[\begin{array}{l}\textsf{temporal-spacing} = \textsf{unfixed} \\ \textsf{frequency} = \textsf{occasionally}\end{array}\right] \\
\textsf{content} = & \left[\begin{array}{ll}
\textsf{structure} = \textsf{delimited-mass} & \\
\textsf{index} = e_2 & \\
\textsf{extent} = & \left[\begin{array}{l}\textsf{end} = \textsf{for-half-hour}\end{array}\right] \\
\textsf{content} = & \left[\begin{array}{ll}
\textsf{structure} = \textsf{happening} & \\
\textsf{index} = e_1 & \\
\textsf{type} = \textsf{discrete} & \\
\textsf{substance} = & \left[\begin{array}{l}\textsf{action} = \textsf{stir} \\ \textsf{patient} = \textsf{soup}\end{array}\right]
\end{array}\right]
\end{array}\right]
\end{array}\right] \tag{5.22}$$

I note that the feature structure representation brings along its own inherent structuring that contributes to the semantics of what it is representing. That is, I am using a structured representation to represent structured entities—activity events; the nesting that is provided in feature structures is used to show activity nesting. So, any activity that is nested in the feature structure representation of another activity, is actually a sub-activity of that other activity. In the example in (5.22), the activity indexed by e_1 is a sub-activity of e_2; similarly e_2 is a sub-activity of e_3. However, this information comes from the structure of the representation, and does not have to be explicitly represented.

5.4.6 Combining object and event quantification

The final example is one where a mix of object and event quantifiers is demonstrated; again the scope interactions allow for more than one reading. The example used is (5.23), which has two readings as glossed in (5.24a) and (5.24b). Feature structures for these readings are in (5.25a) and (5.25b)[24] respectively.

(5.23) Bake some cakes every week.

(5.24) a. ≪*Every week, you are to bake some cakes.*≫

b. ≪*There are some cakes that are to be baked every week.*≫

[24] The second reading is unusual, because we know that baking the same cake object week after week is an unlikely thing to do. However, it is a valid reading—especially if it is understood to mean ≪*bake a cake of a particular sort*≫— and so we must allow for it.

(5.25) a.

$$
\left[\begin{array}{ll}
\mathsf{structure} = \mathsf{ordered\text{-}set} & \\
\mathsf{index} = e_3 & \\
\mathsf{extent} = & \left[\mathsf{frequency} = \mathsf{every\text{-}week}\right] \\
\mathsf{content} = & \left[\begin{array}{ll}
\mathsf{structure} = \mathsf{ordinary\text{-}set} & \\
\mathsf{index} = e_2 & \\
\mathsf{extent} = & \left[\begin{array}{l} \mathsf{cardinality} = \mathsf{some} \\ \mathsf{var} = i \\ \mathsf{restriction} = \mathsf{cake} \end{array}\right] \\
\mathsf{content} = & \left[\begin{array}{ll}
\mathsf{structure} = \mathsf{happening} & \\
\mathsf{index} = e_1 & \\
\mathsf{substance} = & \left[\begin{array}{l} \mathsf{action} = \mathsf{bake} \\ \mathsf{patient} = i \end{array}\right]
\end{array}\right]
\end{array}\right]
\end{array}\right]
$$

b.

$$
\left[\begin{array}{ll}
\mathsf{structure} = \mathsf{ordinary\text{-}set} & \\
\mathsf{index} = e_2 & \\
\mathsf{extent} = & \left[\begin{array}{l} \mathsf{cardinality} = \mathsf{some} \\ \mathsf{var} = i \\ \mathsf{restriction} = \mathsf{cake} \end{array}\right] \\
\mathsf{content} = & \left[\begin{array}{ll}
\mathsf{structure} = \mathsf{ordered\text{-}set} & \\
\mathsf{index} = e_3 & \\
\mathsf{extent} = & \left[\mathsf{frequency} = \mathsf{every\text{-}week}\right] \\
\mathsf{content} = & \left[\begin{array}{ll}
\mathsf{structure} = \mathsf{happening} & \\
\mathsf{index} = e_1 & \\
\mathsf{substance} = & \left[\begin{array}{l} \mathsf{action} = \mathsf{bake} \\ \mathsf{patient} = i \end{array}\right]
\end{array}\right]
\end{array}\right]
\end{array}\right]
$$

5.5 The representation language summarised

The representation can be summarised by the grammar that follows. It uses a BNF notation to describe the space of valid feature structures. This covers unscoped and scoped, but not partially-scoped forms. Some additional functionality has been incorporated as follows:

- Both feature names and feature values are terminal symbols, but they are presented in different typefaces in order to capture their placeholding information:
  Feature names use the typeface **feature-name**.
  Anything that is an atomic feature value is presented using the typeface *feature-value*.

- Rules of the form **feature-name** ::– *feature-value* mean that **feature-name** can take *feature-value* as a possible value;
  rules of the form **feature-name** ::– **feature-name1** ... **feature-nameN** mean that **feature-name** can take as a value a feature structure with top level features **feature-name1** ... **feature-nameN**.

- When feature values can take on an infinite number of values, they are presented as enumerated alternates, such as in *1* | *2* | *3* ....

  Feature values may be expressed using the notation for semantic glosses, ≪*as this example demonstrates*≫.

- The start symbol is the extended event, abbreviated **EE**.

- The usual BNF notation applies for

  optional elements: **[ ]**

  alternates: **|**

  repeated elements: *

EE ::– structure index [extent] content |
structure index type substance [modifiers]

index ::– e_1 | e_2 | e_3 ...

structure ::– *happening* | *delimited-mass* |
ordered-set | *ordinary-set* | *simultaneous-set* |
conjunctive | *concurrent* | *compound*

content ::– structure index [extent] content |
structure index participants |
structure index type substance [modifiers]

participants ::– participant1 participant2 participant3 ... | major minor

participantN ::– structure index [extent] content

major ::– structure index [extent] content

minor ::– structure index [extent] content

type ::– *discrete* | *continuous* | *iterable*

substance ::– action [patient] [recipient] [how]

action ::– ≪*the meaning of some verb*≫

patient ::– term [cardinality] | ≪*a particular object*≫

recipient ::– term [cardinality] | ≪*a particular object*≫

how ::– ≪*some non-temporal modifier*≫

extent ::– [temporal-spacing] [cardinality] [frequency] [var restr] |
extent-type start end | extent1 extent2 |
term [extent-type] | det var restr |
structure index extent-type substance [modifiers]

extent-type ::– *time* | *condition*

extent1 ::– structure [index] extent

extent2 ::– structure [index] extent

temporal-spacing ::– *regular* | *unfixed* | —

cardinality ::– ≪*some cardinality description*≫|
1 | *2* | *3* ...

frequency ::– ≪*some frequency description*≫

start ::– ≪*a time*≫

end ::– ≪*a time*≫| ≪*a length of time*≫| ≪*some condition*≫

modifiers ::– first rest | *end*

rest ::– first rest | *end*

first ::– structure index [extent] content

term ::– det var restr

det ::– ≪*the meaning of some quantifier or determiner*≫

var ::– $i\,|\,j\,|\,k \ldots$

restr ::– ≪*the meaning of some noun*≫

This BNF covers both unscoped and scoped feature structures.

Chapter 6

From language to representations—the grammar

This chapter is concerned with describing the way in which sentences about activity structure are connected to semantic representations of that structure. That is, given language (described in Chapter 4) about structured activities (described in Chapter 3), we now examine how we can obtain the semantic representations (described in Chapter 5) of such structured activities.

Primarily, then, this chapter discusses a grammar that allows the obtaining of semantic representations. The chapter is structured as follows:

- We begin with a general description of the unification approach that the grammar is based on. This includes a description of notation used, as well as descriptions of the form of input sentences and the lexicon.
- Then I describe those parts of the grammar that are relevant to building feature structures to represent happenings. These primarily involve verb phrases.
- In the next section, the grammar rules for building structurings are described.
- Finally, the rest of the grammar—not involved directly in building happenings or structurings, but still pertinent to identifying activity structure—is described.

6.1 Analysing sentences about activity structure

To clarify the role that the grammar described in this chapter plays in semantic analysis, I describe how a representation of its meaning is obtained for an instruction. The main focus is determining the semantic aspects of the instruction, using syntactic and semantic information available from the language itself.

This analysis takes place in two stages :

- First a representation of the sentence with all quantifiers and temporal measure adverbials in their unscoped forms is obtained. These are feature structures with 'in place' quantifiers and measure adverbials. There is potentially more than one representation for any input sentence, if it is syntactically ambiguous.
- The second stage involves applying a scoping algorithm to the feature structures, to produce scoped forms—also feature structures, but with quantifiers and measure adverbials in scoped form in their scoping positions. Of course, there will potentially be more than one scoping for any sentence with more than one quantifier or measure adverbial.

The primary role of the grammar is in the building of feature structures, using information from the syntactic and semantic realms. In Chapter 4 some of the syntactic structures were discussed, while Chapter 5 presented the semantic representations that correspond to these. Here, we focus on how the semantic information gets incorporated into the representation—this comes from both semantic and syntactic information—using unification in a phrase structure approach. In the rest of this chapter, we look at the grammar that describes the unscoped representations of instructions; in Chapter 7 I describe the mechanisms for obtaining scoped forms, as well as discussing other algorithms that are involved in semantic processing.

6.1.1 Unification and path equations

I have used a PATR-II formalism[1] for encoding the semantic analysis; this is a simple unification-based approach and so I shall describe the processing in these

[1] Again, Shieber[44] is a useful introduction to the PATR-II formalism, and unification.

terms.

Grammar rules These are expressed as PHRASE STRUCTURE RULES which reflect the surface syntactic structure. As an example, say we wish to analyse the language fragment '*John runs*'. This can be done using the rule in 6.1, which says very crudely ≪*if we have a Noun Phrase and a Verb Phrase, we can put them together to make a Sentence*≫. There will be other rules that construct NPs and VPs—perhaps utilising intermediate forms and other phrases—ultimately out of entities in the surface form, which are words. The most appropriate way in which to construct grammar rules is to see

(6.1) S ⟶NP VP

Path equations To describe more than simple syntactic structure, grammar rules are annotated with PATH EQUATIONS. A PATH in a feature structure is simply a sequence of features that leads to a particular place in the feature structure. The notation used is to specify in angle brackets the feature structure followed by the feature names in the path. For example, ⟨NS/sem/action⟩ picks out the value [=run] in the feature structure NS of (6.2); ⟨NS/sem⟩ picks out the entire feature structure belonging to [sem=].

$$(6.2) \qquad \mathrm{NS} = \begin{bmatrix} \textsf{index} = e \\ \textsf{syn} = \begin{bmatrix} \textsf{cat} = \textsf{sentence} \end{bmatrix} \\ \textsf{sem} = \begin{bmatrix} \textsf{agent} = \textsf{John} \\ \textsf{action} = \textsf{run} \end{bmatrix} \end{bmatrix}$$

Path equations describe the relationship between feature structures, and as they get built through unification, semantic and syntactic information pass between feature structures through use of these. So, semantic and other information—such as tense and agreement—is passed between feature structures via path equations in the rule. For the rule in (6.1) reasonable path equations might be those in (6.3).

(6.3) $\text{S} \longrightarrow \text{NP} \ \ \text{VP}$

$$\langle \text{S/sem/agent} \rangle = \langle \text{NP/sem} \rangle$$
$$\langle \text{S/sem/action} \rangle = \langle \text{VP/sem} \rangle$$

Unification using path equations The basic tenet of unification is that feature structures with compatible path equations can unify. A semantic representation of a sentence is built through repeated unification using the appropriate grammar rules, and the restrictions and semantic information associated with those grammar rules in the path equations.

6.1.2 The start rule

We are primarily concerned with understanding imperative sentences that are instructions. The rule[2] that governs the overall understanding is thus the following:

R1:
```
INSTR ---> [VP] @ [
    INSTR/cat <=> instr,
    VP/cat <=> vp,
    VP/level <=> phrasal,
    INSTR/level <=> instructional,
    INSTR/sem <=> VP/sem].
```

In future rules, when the names of the constituents reflect their syntactic category, as is the case with INSTR and VP in rule **R1**, I omit the path equations that specify this. These path equations will however be included when the category value differs from the category name, as is the case in rules **R5**, **R6** and **R7**. I also omit, in future rules, path equations that are concerned with level; these simply constrain the generation of feature structures on a syntactic basis. All of these path equations are, however, shown in the rules presented in Appendix A.1.

For the grammar described here, the start symbol is INSTR.

[2] From here onwards, rules are presented in the form that they take in the implementation.

6.1.3 The input instructions and the lexicon

Chapter 8 presents worked examples for a representative range of input instructions. However, it is appropriate here to give a general description of the type of sentences with which the system is able to deal. The PATR-II engine implementation that I have used as a basis for implementing the grammar was developed by Robert Dale in Prolog. I have extended the PATR-II implementation in various ways, described in Section 7.1.1; however the form of the lexicon is as implemented by Dale.

Input sentences The general form of input sentences is that they will be instructions. That is, they will be of the form of imperatives. They are simply lists of words, separated by commas and enclosed in square brackets. Ellipsis is coded using the symbol `ellipsis`; this is discussed further in Section 6.1.3. An example input[3] sentence is
`[stir,the,soup,for,fifteen,minutes]`.

Discourse To express a discourse, I use a list of sentences, also enclosed in square brackets, such as `[[melt,the,butter,until,it,is,foaming],`

```
[skim,the,foam,and,discard,it],
[heat,the,butter,and,skim,ellipsis,again],
[do,this,twice,more]]
```

A discourse is processed one sentence at a time, going through all sentences in the list, to create a feature structure that is [`structure=compound`], and has its own index. Each sentence is processed by the PATR-II engine with respect to the grammar, and its representation added to the list of sentence representations for this complex structure. Then each sentence in the discourse is scoped; the mechanism for scoping extraction is described in Section 7.2. No reference resolution or other intra-discourse processing is performed; however the architecture would allow for such processing to be incorporated.

[3] The form of input sentences is also as implemented by Dale—a Prolog list.

The lexicon is encoded as feature structures that are dictionary items, signified by the predicate `dict`. Each item has a [`cat`=] feature, which is its syntactic category. Other features are also included as appropriate; for example items whose syntactic category is [=`verb`] will also have a feature [`verbclass`=]. (6.4) shows some example lexical entries. Appendix A.2 shows more detail of the lexicon that has been used in the implementation.

(6.4)
```
dict(stir, X@[X/cat  <=> verb, X/verbcat <=> transitive,
           X/level <=> lexical,X/verbclass <=> iterable,
           X/subst <=> stir]).
dict(until, X@[X/cat <=> prep, X/sem <=> condition,
            X/type <=> measure, X/level <=> lexical]).
dict(some, X@[ X/cat <=> det, X/class <=> quant,
           X/type <=> measure, X/level <=> lexical,
           X/agr <=> pl, X/sem <=> some]).
```

All lexical items have a feature [`level`=] whose value is [=`lexical`]; this feature is used through the grammar to control parsing, and its values may be [=`phrasal`], [=`intermediate`] and [=`instructional`], depending on the syntactic level to which the particular rule is appropriate. As has been noted, the path equations pertaining to this feature have been omitted in this chapter, though they are included in the listing in Appendix A.1

Ellipsis

As has been discussed in Section 4.1.1, instructions have a lot of elided items. The approach described here does not deal with ellipsis in any way other than at the level of input sentences. The limited way in which ellipsis is allowed is through the use of the lexical item `ellipsis`, whose lexical entry is as follows:

```
dict(ellipsis, X@[X/cat <=> np, X/level <=> phrasal,
                X/sem <=> elided]).
```

Thus, the sentence in (6.5a) would need to be input as (6.5b), because '*garnish*' is a transitive verb and requires a subject.

(6.5) a. Garnish with the parsley.

b. `[garnish, ellipsis, with, the, parsley]`

6.1.4 Terms and modifier lists

Before describing in detail the grammar rules that build happenings and structurings, I explain some concepts that have been used in constructing the grammar.

Quantified noun terms representing plural objects are important in this approach, and so are temporal measure modifiers. Put simply, quantified noun terms can give rise to ordinary set structure; temporal measure modifiers may indicate ordered sets, simultaneous sets[4] or delimited masses.

I distinguish between quantified terms representing objects that are significant to structure and those that are not. In the sentence

(6.6) Bake two cakes for ten minutes.

there are two apparently similar noun phrases—'*two cakes*' and '*ten minutes*'. However, of these two, it is only '*two cakes*' that indicates that this event may be structured; '*ten minutes*'—even though it has what would be seen as a quantifier—has no input into the structuring. It is the case that '*for ten minutes*' indicates a structuring, but this comes from the preposition, and not the fact that the time is quantified. In practical terms, it is the position of the term in the feature structure—if it is the subject or object of an action, for example—that indicates whether it contributes to activity structure in the way described above.

6.2 Building happenings

Happenings come from simple verb phrases; they are built from rules for analysing verbs.

The substance of the activity comes from analysing the verb phrase in the usual way, and identifying the patient and recipient—from noun phrases and prepositional phrases—if appropriate.

[4] There will be little discussion of simultaneous sets per se.

The following rules build happenings for intransitive, transitive and ditransitive verbs respectively. They indicate that the verb phrase analysis does not get fully resolved (using the category `vpunresolved`. This is because the happening type has not yet been instantiated; this is done using the [`verbclass=`] feature in rules **R5**,**R6** and **R7**, described later in this section.

R2:
```
VP ---> [VERB] @ [
    VP/cat <=> vpunresolved,
    VERB/verbcat <=> intransitive,
    VP/verbclass <=>  VERB/verbclass,
    VP/sem/structure <=> happening,
    VP/sem/index <=> nextevent[5],
    VP/sem/substance/action <=> VERB/subst].
```

R3:
```
VP ---> [VERB,NP] @ [
    VP/cat <=> vpunresolved,
    VERB/verbcat <=> transitive,
    VP/verbclass <=>  VERB/verbclass,
    VP/sem/structure <=> happening,
    VP/sem/index <=> nextevent,
    VP/sem/substance/action  <=> VERB/subst,
    VP/sem/substance/patient <=> NP/sem].
```

R4:
```
VP ---> [VERB,NP,PP] @ [
    VP/cat <=> vpunresolved,
    VERB/verbcat <=> ditransitive,
    PP/type <=> nonmeasure,
    VP/verbclass <=> VERB/verbclass,
    VP/sem/structure <=> happening,
    VP/sem/index <=> nextevent,
    VP/sem/substance/recipient <=> PP/sem,
    VP/sem/substance/patient <=> NP/sem,
    VP/sem/substance/action  <=> VERB/subst].
```

[5] This assumes the existance of a generator that will generate the next event index; such a generator has not been implemented, but its implementation is trivial

For the sentence

(6.7) Put a cake on the table.

the feature structure in (6.8d) would be built using **R4**, with the lexical entry for '*put*' as in (6.8a), and assuming feature structures for the NP '*a cake*' and the PP '*on the table*' as in (6.8b) and (6.8c) respectively. Note that '*the table*' is not represented as a quantified term, while '*a cake*' is; the rules for building representations of objects are described in Section 6.4.1.

(6.8) a. `item = put`
cat = verb, verbcat = ditransitive, level = lexical, verbclass = simple, subst = put, sem = put

b.

$$\begin{bmatrix} \text{cat} = \text{np} \\ \text{sem} = \begin{bmatrix} \text{det} = \text{exists} \\ \text{var} = \text{y} \\ \text{restr} = \text{cake} \end{bmatrix} \\ \text{agr} = \text{sing} \\ \text{type} = \text{nonmeasure} \end{bmatrix}$$

c.

$$\begin{bmatrix} \text{cat} = \text{pp} \\ \text{type} = \text{nonmeasure} \\ \text{sem} = \text{table} \end{bmatrix}$$

d.

$$\begin{bmatrix} \text{cat} = \text{vpunresolved} \\ \text{verbclass} = \text{simple} \\ \text{sem} = \begin{bmatrix} \text{structure} = \text{happening} \\ \text{index} = \text{e1} \\ \text{substance} = \begin{bmatrix} \text{action} = \text{put} \\ \text{patient} = \begin{bmatrix} \text{det} = \text{exists} \\ \text{var} = \text{y} \\ \text{restr} = \text{cake} \end{bmatrix} \\ \text{recipient} = \text{table} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

The verb class information then contributes to the semantics of the happening as follows: If the verb is a [=simple] one, the [type=] of happening it produces will be [=discrete]. If the verb is [=masslike], it will produce a [=continuous] happening. This is achieved using **R5** and **R6** respectively.

For [=iterable] verbs, the happening type is not set until more information is available; this information will be obtained once the scoping is complete. The value [=iterable] is passed up to the verb phrase, using rule **R7**.

The feature value [=unresolved] for the category of VP2 is used to ensure that for future productions, only verb phrases that have had their verb resolved are used. Thus, rules **R2** to **R7** together analyse basic verb phrase and establish what kind of happening is being produced by the verb phrase. For any basic verb phrase, one of rules **R2**, **R3** and **R4** provides an intermediate step, which performs only syntactic analysis. Then, one of **R5**, **R6** and **R7** essentially performs semantic processing. This 2-part approach was chosen to avoid carrying around an additional feature to indicate the resolved status—this would have needed to be carried through all verb phrases. I note however, that the 1-part approach, with additional features, would have been syntactically more usual.

R5:
```
VP1 ---> [VP2] @ [
    VP1/cat <=> vp,
    VP2/cat <=> vpunresolved,
    VP2/verbclass <=> simple,
    VP1/sem <=> VP2/sem,
    VP1/sem/type <=> discrete,
    VP1/sem/modifiers <=> end].
```

R6:
```
VP1 ---> [VP2] @ [
    VP1/cat <=> vp,
    VP2/cat <=> vpunresolved,
    VP2/verbclass <=> masslike,
    VP1/sem <=> VP2/sem,
    VP1/sem/type <=> continuous,
    VP1/sem/modifiers <=> end].
```

R7:
```
VP1 ---> [VP2] @ [
```

```
VP1/cat <=> vp,
VP2/cat <=> vpunresolved,
VP2/verbclass <=> iterable,
VP1/sem <=> VP2/sem,
VP1/sem/type <=> VP2/verbclass,
VP1/sem/modifiers <=> end].
```

Through **R5**, the following feature structure would result from the one in (6.8d):

(6.9)
$$\begin{bmatrix} \text{cat} = \text{vp} \\ \text{sem} = \begin{bmatrix} \text{structure} = \text{happening} \\ \text{index} = \text{e1} \\ \text{substance} = \begin{bmatrix} \text{action} = \text{put} \\ \text{patient} = \begin{bmatrix} \text{term} = \begin{bmatrix} \text{det} = \text{exists} \\ \text{var} = \text{x} \\ \text{restr} = \text{cake} \end{bmatrix} \end{bmatrix} \\ \text{recipient} = \text{table} \end{bmatrix} \\ \text{type} = \text{discrete} \\ \text{modifiers} = \text{end} \end{bmatrix} \end{bmatrix}$$

6.3 Building structurings

There are a number of rules that take verb phrases and other constituents and build representations of structured events. The primary mechanism for this is the building of a list of modifiers; each modifier that is a *temporal measure* one—whatever its syntactic origin is—is added to the start of an initially empty list of modifiers associated with the happening represented by the basic verb phrase.

The rule governing all of this is **R8**:

R8:
```
VP1 ---> [VP2,MODIFIER] @ [
    MODIFIER/type <=> tempmeasure,
```

```
VP1/sem/structure <=> VP2/sem/structure,
VP1/sem/index <=> VP2/sem/index,
VP1/sem/substance <=> VP2/sem/substance,
VP1/sem/type <=> VP2/sem/type,
VP1/sem/modifiers/first <=> MODIFIER/sem,
VP1/sem/modifiers/rest <=> VP2/sem/modifiers].
```

Modifiers may come from prepositional phrases, adverbial phrases and sentence complements. The information in these modifiers includes that describing the actual structure of the extended events to which they are contributing. **R9** demonstrates a rule that identifies a temporal measure prepositional phrase as a modifier; similar rules exist for adverbial phrases and sentence complements.

R9:

```
MODIFIER ---> [PP] @ [
    PP/type <=> tempmeasure,
    MODIFIER/type <=> tempmeasure,
    MODIFIER/sem <=> PP/sem].
```

All temporal measure phrases, which subsequently become modifiers via rules like **R9**, will contain information about structure and extent. The feature [`sem`=] will contain this information; an example is shown in the feature structure of (6.10). The scoping algorithm described in Section 7.2 finds these modifiers and [`term`=] features that are patients or recipients of verb phrases, and pulls them to appropriate positions in the feature structure.

6.3.1 Delimited masses

Delimited masses are usually described by prepositional phrases that are temporal measure in nature, as in '*for fifteen minutes*'. **R10** is the rule for identifying prepositional phrases of this sort.

R10:

```
PP ---> [PREP,NP] @ [
    PREP/type <=> measure,
    NP/type <=> time,
    PP/type <=> tempmeasure,
```

```
PP/sem/structure <=> delimitedmass,
PP/sem/index <=> nextevent,
PP/sem/extent <=> NP/sem,
PP/sem/extent/extent_type <=> PREP/sem].
```

This rule would produce the feature structure in (6.10) for the phrase '*for fifteen minutes*'; **R9** would then identify it as a modifier, producing the feature structure in (6.10b).

(6.10) a.

$$\left[\begin{array}{l} \text{cat} = \text{pp} \\ \text{type} = \text{tempmeasure} \\ \text{sem} = \left[\begin{array}{l} \text{structure} = \text{delimited-mass} \\ \text{index} = \text{e5} \\ \text{extent} = \left[\begin{array}{l} \text{term} = \left[\begin{array}{l} \text{det} = 15 \\ \text{var} = \text{z} \\ \text{restr} = \text{minutes} \end{array}\right] \\ \text{extent-type} = \text{time} \end{array}\right] \end{array}\right] \end{array}\right]$$

b.

$$\left[\begin{array}{l} \text{cat} = \text{modifier} \\ \text{type} = \text{tempmeasure} \\ \text{sem} = \left[\begin{array}{l} \text{structure} = \text{delimited-mass} \\ \text{index} = \text{e5} \\ \text{extent} = \left[\begin{array}{l} \text{term} = \left[\begin{array}{l} \text{det} = 15 \\ \text{var} = \text{z} \\ \text{restr} = \text{minutes} \end{array}\right] \\ \text{extent-type} = \text{time} \end{array}\right] \end{array}\right] \end{array}\right]$$

Using rule **R8**, the feature structures in (6.11)—representing '*bake the cake*'—and (6.10b)—representing '*for fifteen minutes*'—unify to produce the feature structure in (6.12).

$$
\text{(6.11)} \quad \left[\begin{array}{l}
\text{cat} = \text{vp} \\
\text{sem} = \left[\begin{array}{l}
\text{content} = \left[\begin{array}{l}
\text{structure} = \text{happening} \\
\text{index} = \text{e2} \\
\text{substance} = \left[\begin{array}{l} \text{action} = \text{bake} \\ \text{patient} = \text{cake} \end{array}\right] \\
\text{type} = \text{continuous}
\end{array}\right] \\
\text{modifiers} = \text{end}
\end{array}\right]
\end{array}\right]
$$

$$
\text{(6.12)} \quad \left[\begin{array}{l}
\text{cat} = \text{vp} \\
\text{sem} = \left[\begin{array}{l}
\text{structure} = \text{happening} \\
\text{substance} = \left[\begin{array}{l} \text{action} = \text{bake} \\ \text{patient} = \text{cake} \end{array}\right] \\
\text{type} = \text{continuous} \\
\text{modifiers} = \left[\begin{array}{l}
\text{first} = \left[\begin{array}{l}
\text{structure} = \text{delimited-mass} \\
\text{index} = \text{e5} \\
\text{extent} = \left[\begin{array}{l}
\text{term} = \left[\begin{array}{l} \text{det} = 15 \\ \text{var} = \text{z} \\ \text{restr} = \text{minutes} \end{array}\right] \\
\text{extent-type} = \text{time}
\end{array}\right]
\end{array}\right] \\
\text{rest} = \text{end}
\end{array}\right]
\end{array}\right]
\end{array}\right]
$$

Delimited masses may also be described by complements, together with sentences describing states, such as in '*until the sauce is thick*', whose syntactic category is SBAR following Burton-Roberts[6]. The appropriate rules are **R11** and **R12**.

R11 builds the SBAR structure from the complement and the state-describing sentence; it encodes the information that the COMPLEMENT and SENTENCE together describe a delimited mass. The SENTENCE describes the state, as in '*the sauce is thick*'[6]. The semantics of the SENTENCE are obtained through a simple NP VP rule,

[6] '*The sauce becomes thick*', which could arguably be seen as an event, is also an acceptable sentence.

detailed in Appendix A.1; the machinery that I have provided for such analysis, while itself not being very comprehensive, allows easy extension to a more detailed coverage. '*Until*' is the COMPLEMENT in the SBAR structure.

R12 identifies the SBAR as a modifier in the same way as was done in **R9**.

R11:
```
SBAR ---> [COMPLEMENT,SENTENCE] @ [
  COMPLEMENT/type <=> measure,
  SENTENCE/sem/vptype <=> state,
  SBAR/sem/structure <=> delimitedmass,
  SBAR/sem/index <=> nextevent,
  SBAR/sem/extent/structure <=> SENTENCE/sem/structure,
  SBAR/sem/extent/index <=> SENTENCE/sem/index,
  SBAR/sem/extent/substance <=> SENTENCE/sem/substance,
  SBAR/sem/extent/modifiers <=> SENTENCE/sem/modifiers,
  SBAR/sem/extent/extent_type <=> COMPLEMENT/sem,
  SBAR/type <=> tempmeasure].
```

R12:
```
MODIFIER ---> [SBAR] @ [
  SBAR/type <=> tempmeasure,
  MODIFIER/type <=> tempmeasure,
  MODIFIER/sem <=> SBAR/sem].
```

6.3.2 Sets

Ordered sets

Ordered sets are described using adverbial phrases such as '*five times*' and '*every hour*'. Similarly to **R12**, adverbial phrases that are temporal measure will be identified as modifiers, and then added to the front of the list of temporal measure adverbials by **R8**; building the representations for adverbial phrases that are not temporal measure ones is described in Section 6.4.3.

R13 allows simple adverbials, such as '*often*', to become adverbial phrases. The lexical entry for '*often*' will have ADV as its syntactic category. **R14** applies to phrases like '*every week*'—composed of a quantifier and a noun phrase. **R15** allows analysis of '*four times*' as a quantified term that is an adverbial phrase rather than a noun phrase.

All three rules construct representations that have a [structure=] that is an [=ordered-set]

R13:
```
ADVP ---> [ADV] @ [
  ADVP/sem/structure <=> ordered_set,
  ADVP/sem/index <=> nextevent,
  ADVP/sem/extent/det <=> ADV/sem,
  ADVP/sem/extent/var <=> nextvar[7],
  ADVP/sem/extent/restr <=> ADV/advsort].
```

R14:
```
ADVP ---> [DET,NP] @ [
  DET/type <=> measure,
  NP/type <=> time,
  ADVP/type <=> tempmeasure,
  DET/class <=> quant,
  ADVP/sem/structure <=> ordered_set,
  ADVP/sem/index <=> nextevent,
  ADVP/sem/extent/det <=> DET/sem,
  ADVP/sem/extent/var <=> nextvar,
  ADVP/sem/extent/restr <=> NP/sem].
```

R15:
```
ADVP ---> [DET,NOUN] @ [
  DET/class <=> quant,
  NOUN/sem <=> instances,
  ADVP/type <=> NOUN/type,
  ADVP/sem/structure <=> ordered_set,
  ADVP/sem/index <=> nextevent,
  ADVP/sem/extent/det <=> DET/sem,
  ADVP/sem/extent/var <=> nextvar,
  ADVP/sem/extent/restr <=> NOUN/sem].
```

As with PP and SBAR constructs, a rule similar to **R9** will identify these kind

[7] This assumes the existance of a generator that will generate the next variable.

of adverbial phrases as modifiers, which will then be added to the front of the modifiers list.

Ordinary sets

Ordinary sets are usually the result of plural or quantified objects as patients or recipients of an action, as in '*Put a cherry on top of each cupcake*'. These are made during the scoping procedure—described in Section 7.2—if the patient or recipient of a happening is represented as an object that is a quantified term. The incorporation of object representations into happenings occurs during the processing of **R3** or **R4**, which apply to transitive and ditransitive verbs. Building representations of objects—both for quantified terms and for unquantified objects—is described in the rules in Section 6.4.1, which are about noun phrases.

Simultaneous sets

Adverbial phrases such as '*at the same time*', together with plural or quantified objects as patients or recipients, contribute to the semantics of simultaneous sets. The building of unscoped representations of simultaneous sets occurs in the same way as that of ordered sets, where the appropriate modifier is added to the list of modifiers. Resolving the connection between the modifier and quantified objects occurs, as with ordinary sets, during scoping. I note that actual treatment of simultaneous sets has not been included in the implementation.

6.3.3 Conjunctions

The extended event that is viewed as a conjunction comes from sentences like '*Cook the sauce for two minutes or until it thickens*'. These sentences are syntactically unusual in that the phrases on either side of the adjunct—'*or*' in this case—are not of the same form.

This is dealt with through the approach of treating both of these phrases ultimately as modifiers, using **R10** and **R8** to identify '*for two minutes*' as a modifier, and **R11** and **R12** to identify '*until it thickens*' as a modifier. Then, **R16** conjoins the two modifiers in the appropriate way.

R16:
```
MODIFIER1 ---> [MODIFIER2,ADJUNCT,MODIFIER3] @ [
    ADJUNCT/sem <=> conjunct,
    MODIFIER1/sem/structure <=> conjunctive,
    MODIFIER1/sem/index <=> nextevent,
    MODIFIER1/sem/extent/extent1 <=> MODIFIER2/sem,
    MODIFIER1/sem/extent/extent2 <=> MODIFIER3/sem,
    MODIFIER1/type <=> tempmeasure].
```

The feature structures in (6.13a) and (6.13b) represent '*for two minutes*' and '*until it thickens*' respectively.

(6.13) a.

$$\begin{bmatrix} \text{cat} = \text{modifier} \\ \text{sem} = \begin{bmatrix} \text{structure} = \text{delimited-mass} \\ \text{index} = \text{e7} \\ \text{extent} = \begin{bmatrix} \text{term} = \begin{bmatrix} \text{det} = 2 \\ \text{var} = \text{x} \\ \text{restr} = \text{minutes} \end{bmatrix} \\ \text{extent-type} = \text{time} \end{bmatrix} \end{bmatrix} \\ \text{type} = \text{tempmeasure} \end{bmatrix}$$

b.

$$\begin{bmatrix} \text{cat} = \text{modifier} \\ \text{sem} = \begin{bmatrix} \text{structure} = \text{delimited-mass} \\ \text{index} = \text{e6} \\ \text{extent} = \begin{bmatrix} \text{structure} = \text{state} \\ \text{index} = \text{e9} \\ \text{substance} = \begin{bmatrix} \text{condition} = \text{become-thick} \\ \text{patient} = \text{referent} \end{bmatrix} \\ \text{extent-type} = \text{condition} \end{bmatrix} \end{bmatrix} \\ \text{type} = \text{tempmeasure} \end{bmatrix}$$

R16 would allow the unification of these feature structures to produce the feature structure whose semantic value is as follows[8]:

$$\left[\begin{array}{l}
\text{structure} = \text{conjunctive} \\
\text{index} = \text{e10} \\
\text{extent} = \left[\begin{array}{l}
\text{extent1} = \left[\begin{array}{l}
\text{structure} = \text{delimited-mass} \\
\text{index} = \text{e7} \\
\text{extent} = \left[\begin{array}{l}
\text{term} = \left[\begin{array}{l}
\text{det} = 2 \\
\text{var} = \text{x} \\
\text{restr} = \text{minutes}
\end{array}\right] \\
\text{extent-type} = \text{time}
\end{array}\right]
\end{array}\right] \\
\text{extent2} = \left[\begin{array}{l}
\text{structure} = \text{delimited-mass} \\
\text{index} = \text{e6} \\
\text{extent} = \left[\begin{array}{l}
\text{structure} = \text{state} \\
\text{index} = \text{e9} \\
\text{substance} = \left[\begin{array}{l}
\text{condition} = \text{become-thick} \\
\text{patient} = \text{referent}
\end{array}\right] \\
\text{extent-type} = \text{condition}
\end{array}\right]
\end{array}\right]
\end{array}\right]
\end{array}\right]$$

6.3.4 Concurrent events

Concurrent events of the sort described in Section 5.3.4 are expressed using what I have termed gerundive verb phrases[9] (GVP's) at appropriate places in the verb phrase. Sentences like those in 6.14 describe concurrent events; the gerundive verb phrases are italicised.

(6.14) a. *Having been invited to the castle by the count*, we accepted.

b. *Having furnished ourselves with garlic*, we set off.

[8] I only show the feature structure for the semantics because of space constraints.

[9] I note that Burton-Roberts'[6, pg223] analysis of non-finite clauses provides a more general exposition of such verb phrases; however Burton-Roberts' analysis requires the incorporation of empty constituents in order to view these as clauses. For my purposes, it is clearer to instead take the view that these clauses are always in the form of verb phrases that contain verbs in gerundive form.

c. We hung around for several hours *waiting for the count to appear.*

In many of the sentences, moving the position of the adverbial, from start to end or vice versa, is quite acceptable. Although (6.14a) sounds odd when the GVP is moved, it is still marginally acceptable.

Syntactically, the comma is needed when the adverbial is in the form of a gerundive verb phrase. In recipes, the GVP is often marked by commas and I argue that without the commas the instruction becomes ambiguous. The commas help us attach the adverbials to the right entity. This has been discussed in Section 4.6.

In my implementation, I do not account for punctuation in any way. Sentences are analysed without punctuation, and with ambiguous syntactic structures all available readings are found. The syntax diagrams that were shown in Figure 4.10 and Figure 4.11 demonstrate this ambiguity. See Section 8.8 for an example of processing a sentence that has multiple readings.

GVP's describing extended activity

In examples of (6.14a) and (6.14b), the gerundive verb phrases do not describe extended activity. However, there are many cooking examples that use GVP's that do. '*Cook the sauce for five minutes, stirring it continuously*' contains the GVP '*stirring it continuously*'[10]. **R17** forms the representation of the activity described by, say, '*stirring it*', while **R18**[11] forms the representation of the extended activity described by '*stirring it occasionally*'. **R19** combines the two verb phrases—the first a conventional verb phrase, and the second a gerundive verb phrase (possibly modified)—from, say, '*cook the sauce for ten minutes*' and '*stirring it continuously*'.

R17:
```
GERUNDVP ---> [VGER,NP] @ [
   VGER/verbcat <=> transitive,
   GERUNDVP/sem/structure <=> happening,
   GERUNDVP/sem/index <=> nextevent,
   GERUNDVP/sem/substance/action  <=> VGER/sem,
   GERUNDVP/sem/substance/patient <=> NP/sem,
```

[10] Note that this GVP is in fact the GVP '*stirring it*' that has been modified by '*continuously*'.

[11] This rule is similar to rule **R8**, for the modifying of an ordinary verb phrase.

```
           GERUNDVP/sem/type <=> VGER/verbclass].
```

R18:
```
        GERUNDVP1 ---> [GERUNDVP2,MODIFIER] @ [
           MODIFIER/type <=> tempmeasure,
           GERUNDVP1/sem/structure <=> GERUNDVP2/sem/structure,
           GERUNDVP1/sem/index <=> GERUNDVP2/sem/index,
           GERUNDVP1/sem/substance <=> GERUNDVP2/sem/substance,
           GERUNDVP1/sem/type <=> GERUNDVP2/sem/type,
           GERUNDVP1/sem/modifiers/first <=> MODIFIER/sem,
           GERUNDVP1/sem/modifiers/rest <=>
                                     GERUNDVP2/sem/modifiers].
```

R19:
```
         VP1 ---> [VP2,GERUNDVP] @ [
           VP1/sem/structure <=> concurrent,
           VP1/sem/index <=> nextevent,
           VP1/sem/substance/major <=> VP2/sem,
           VP1/sem/substance/major/modifiers <=>
                                       VP2/sem/modifiers,
           VP1/sem/substance/minor <=> GERUNDVP/sem,
           VP1/sem/substance/minor/modifiers <=>
                                       GERUNDVP/sem/modifiers,
           VP1/sem/modifiers <=> end].
```

6.4 The rest of the grammar

In addition to the rules that produce happenings and structurings, as described in Sections 6.1 to 6.3, there are suites of rules for obtaining noun phrases, verb phrases, adverbial phrases, etc., that provide essential syntactic information. There has also been a need to include additional semantic information—although it is not directly expressive of structuring, it contributes to the building of structuring information. We now look at these rules, which I present organised according to their syntactic categories for ease of exposition.

6.4.1 Noun Phrases

Noun phrases are important to identifying structure in various ways, including the following:

- plural objects as patients or recipients of action contribute to the establishment of an ordinary set
- adverbial phrases such as '*five times*' and '*every week*'—which establish ordered sets—contain noun phrases
- Prepositional phrases such as '*for ten minutes*'—which establish the extent of delimited masses—contain noun phrases.
- Other syntactic constituents, such as sentence complements like '*until the water boils*', may contain noun phrases.

There is a suite of productions for dealing with noun phrases—I have included only as much as is required for coverage of the kinds of instructions I deal with. Clearly, it would be possible to provide a far more comprehensive coverage of noun phrases and related syntactic constituents. In the realm of cooking recipes, for example, there are some relatively complex noun phrases, such as '*a spoonful of jam*'. It is not within the scope of the work presented here to provide a complete analysis of such expressions. Nor are we concerned with the role of adjectives. All that has been implemented is the ability to deal with noun phrases that are important to our analysis, and what I present here is sufficient to provide the functionality required for identifying eventuality structure, as well as allowing for a range of example sentences to be analysed.

Determiners All indefinite determiners are treated as quantifiers. Analysing '*a cake*' results ultimately in the same structuring information as '*some cakes*'[12]. Definite determiners[13] receive a conventional determiner treatment—one where it is assumed that the determiner takes narrow scope over its associated object.

[12] In fact, both of these would potentially indicate a set structure, implying possible iteration; this is appropriate since we wish the sentence '*Mary bakes a cake every week*' to result in two readings.

[13] The only definite determiner we use is in fact '*the*'.

However, a definite determiner with a plural object is annotated as being this, as this information is used to establish ordinary sets. So, all of '*a cake*', '*some cakes*', '*many cakes*', '*five cakes*' and '*the cakes*' will produce a quantified term that indicates a potential set structure[14] through scoping. Only '*the cake*' does not result in a quantified term.

(6.15a) shows the lexical representation for '*some*', which is an indefinite determiner; (6.15b) shows the representation of '*the*', which is a definite determiner. Appendix A.2 contains the lexical representations of the kinds of determiners that the system allows. Numbers are represented as indefinite determiners

(6.15) a.
```
dict(some, X@[ X/cat <=> det, X/class <=> quant,
               X/type <=> measure, X/level <=> lexical,
               X/agr <=> pl, X/sem <=> some]).
```
b.
```
dict(the, X@[X/cat <=> det,X/level <=> lexical,
             X/class <=> definite, X/sem <=> all ]).
```

R20 builds a quantified term—in unscoped form—from an indefinite determiner and a nominal[15].

R21 and **R22** build a simple noun phrase from a definite determiner and a singular or mass nominal, while **R23** builds a noun phrase containing a quantified term, due to the plural nominal. The information regarding singular, mass and plural objects is passed through using the [agr=] feature.

R20:
```
NP ---> [DET,NOM] @ [
   DET/class <=> quant,
   NP/sem/term/det <=> DET/sem,
   NP/sem/term/var <=> nextvar,
   NP/sem/term/restr <=> NOM/sem,
```

[14] I note that analysing '*a*' in this way opens up the issue of whether '*a cake*' and '*one cake*' are qualitatively—and not just quantitatively—different from '*three cakes*'. And this leads to the issue of whether iterating from 1 to 1 is in fact iteration. And this leads to the issue of whether a set containing one element only is still a set. I do not claim to have resolved these issues; only that the approach I take is to see '*one cake*' as the same as '*a cake*', and that these differ only in quantity from '*three cakes*', but they are all different from '*the cake*'.

[15] A nominal is a noun or a pronoun, or noun that has been qualified by an adjectival phrase. Examples are '*cat*', '*it*' and '*big cat*' respectively.

```
          DET/agr <=> NOM/agr].
```

R21:
```
        NP ---> [DET,NOM] @ [
          DET/class <=> definite,
          NOM/agr <=> sing,
          DET/agr <=> NOM/agr,
          NP/agr <=> NOM/agr,
          NP/sem <=> NOM/sem].
```

R22:
```
        NP ---> [DET,NOM] @ [
          DET/class <=> definite,
          DET/agr <=> NOM/agr,
          NOM/agr <=> mass,
          NP/agr <=> NOM/agr,
          NP/sem <=> NOM/sem].
```

R23:
```
        NP ---> [DET,NOM] @ [
          DET/class <=> definite,
          NOM/agr <=> pl,
          DET/agr <=> NOM/agr,
          NP/agr <=> NOM/agr,
          NP/sem/term/det <=> DET/sem,
          NP/sem/term/var <=> nextvar,
          NP/sem/term/restr <=> NOM/sem,
          NP/sem/cardinality <=> Numberof(NOM/sem)].
```

The value [=`Numberof(NOM/sem)`] in rule **R23** indicates that the cardinality of a plural object should be connected to the number of that object that happen to be available when the activity takes place. It is feasible to leave this in the form of a function to be evaluated at execution time. For example, in the set produced from '*Peel the oranges*', we only know how many peeling activities there are when we know how many oranges there are.

There are other rules for building nominals and noun phrases, which are not especially relevant to structuring, except for the distinguishing of plural objects in the nominal and noun level. These rules appear in Appendix A.1.

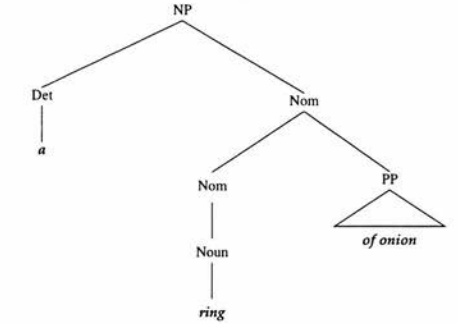


Figure 6.1: The syntactic structure of '*a ring of onion*'.

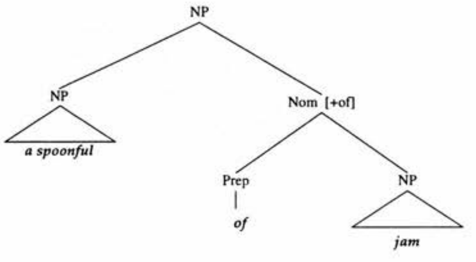


Figure 6.2: The syntactic structure of '*a spoonful of jam*'.

Analysing '*a spoonful of jam*' .

I end this discussion of noun phrase analysis with a brief examination of a more complex noun phrase, in order to illustrate the way in which the treatment I have presented can be extended to deal with complex objects.

Phrases like '*a spoonful of jam*' or '*a pound of butter*' need a different analysis to that of '*a student of physics*', given by Radford[39]. Dale's analysis[12] says that '*a ring of onion*' is like '*a student of physics*' in that we can say '*an onion ring*' and '*a physics student*'. The syntactic structure is demonstrated in Figure 6.1.

However, we cannot say '*a butter pound*', and so rules that can deal with the syntactic structure of Figure 6.2 are necessary.

So, for '*a ring of onion*', **R24**a is needed; and for '*a spoonful of jam*' we need

R24b and **R25**. To distinguish semantically between these two kinds of structure, more machinery, in the form of additional path equations, would be needed. I have chosen instead to implement only rules **R24**b and **R25**; this means that my grammar does analyse phrases like '*a ring of onion*' as though their structure was similar to phrases like '*a pound of butter*'. However, such analysis is not a central issue here; it is sufficient to know that there is some singular object that is referred to using a complex noun phrase.

R24: a.

```
NOM1 ---> [NOM2,PP] @ [
    NOM1/sem <=> NOM2/sem,
    NOM1/agr <=> NOM2/agr].
```

b.

```
NOM1 ---> [PREP,NOM2] @ [
    PREP/sem <=> of,
    NOM1/preptype <=> consists,
    NOM1/type <=> nonmeasure,
    NOM1/sem <=> NOM2/sem,
    NOM1/agr <=> NOM2/agr].
```

R25:

```
NP1 ---> [NP2,NOM] @ [
  NOM/preptype <=> consists,
  NP1/agr <=> NP2/agr,
  NP2/type <=> measure,
  NP1/sem/contains <=> NOM/sem,
  NP1/sem <=> NP2/sem,
  NP1/type <=> nonmeasure].
```

6.4.2 Adverbial Phrases

The most important role played by adverbial phrases in this context is as temporal modifiers to the verb phrase, such as '*twice, daily, every five minutes, often*'. There are of course other adverbial phrases, such as '*quickly*', which do not contribute to the extended activity structure. Such adverbials will be analysed to be of [`type=nonmeasure`].

Rules for temporal measure adverbial phrases were discussed in Section 6.3.2. In addition to those rules—**R13**, **R14** and **R15**—there are also rules to incorporate premodifiers ('*another three times*') and postmodifiers ('*three times more*') to the adverbial phrase. These are **R26** and **R27** respectively. I have not implemented any semantic information with these; they simply take the semantics of the unmodified adverbial phrase—again I note that a fuller treatment would distinguish between, for example, '*three times*' and '*another three times*'.

R26:
```
ADVP1 ---> [PREMODIFIER,ADVP2] @ [
  ADVP1/type <=> ADVP2/type,
  ADVP1/level <=> ADVP2/level,
  ADVP1/sem <=> ADVP2/sem].
```

R27:
```
ADVP1 ---> [ADVP2,POSTMODIFIER] @ [
  ADVP1/type <=> ADVP2/type,
  ADVP1/level <=> ADVP2/level,
  ADVP1/sem <=> ADVP2/sem].
```

6.4.3 Prepositional Phrases

I distinguish again between two kinds of prepositional phrase — those that are temporal measure ('*for ten minutes; until the party*') and those that are not ('*with a spoon; on the table*'). It is prepositional phrases of the temporal measure sort that contribute to extended activity structure; these were discussed in Section 6.3.1, and are covered by **R10**.

R28 is an example of how a more comprehensive treatment might deal with prepositional phrases that are not temporal-measure ; again, a full treatment has not been included here because such phrases do not contribute to the activity structure.

R28:
```
PP ---> [PREP,NP] @ [
  PREP/type <=> nonmeasure,
  PREP/sem <=> place,
  PP/type <=> nonmeasure,
  PP/sem/head <=> PREP/sem,
  PP/sem/rest <=> NP/sem].
```

6.5 Summary

In this chapter we have looked in some detail at a PATR-II grammar that would allow the formation of feature structures that represent the semantics of sentences that are about extended activity structure. These representations are in unscoped form; in the next chapter an overall system of processing is described, which includes a scoping algorithm that converts these unscoped representations to scoped representations as described in Chapter 5.

Chapter 7

Mechanisms for understanding instructions

This chapter is concerned with describing and discussing aspects of a computational realisation of the model that has been discussed in the previous chapters. That is, using the architecture proposed for a system of instruction understanding, together with a model of how the language about extended activity maps onto the activity itself, it presents a computational realisation of this mapping.

The chapter deals only with the processing involved in obtaining the neutral semantics[1] of a sentence, and producing a visual representation of this. The further processing required to obtain an executable semantics has not been implemented; however the modular architecture allows for this to be done.

The structure of this chapter is as follows:

- We begin by looking at the overall architecture that is used for the understanding of instructions. This involves seeing how the various components, described in earlier chapters, will fit together.

- Then I describe the scoping algorithm, that takes feature structures representing the activity structure in unscoped form and produces scoped representations.

- The processing that then takes a scoped feature structure and produces a

[1] This term was introduced in Chapter 3, and refers to the semantics of an instruction that can be obtained independent of execution context.

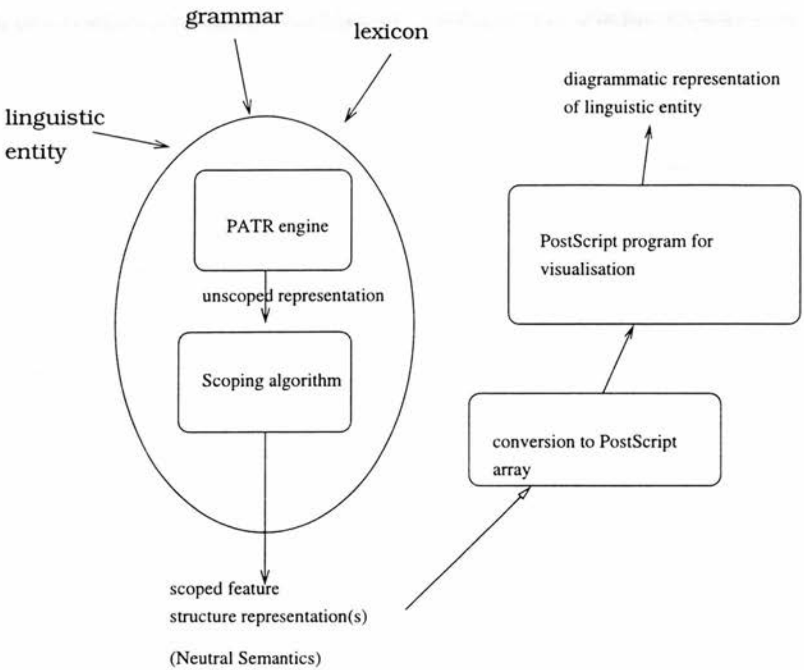


Figure 7.1: Overall system of instruction execution.

visualisation of the activity structure it represents is described in Section 7.3.

7.1 An understanding system

In Section 3.3, an overall framework for understanding instructions in an execution context was presented. In that section, the role of context was emphasised, and in Section 3.3.2, a general architecture was presented. In this section, that general architecture is used in the formation of a particular system that achieves understanding of instructions that involve extended activity.

This is a proposed architecture; a closer look at the overall system. The architecture here is similar to How's[18] model—in particular the separation of language and environment (context), and the cascaded processing is the same. I am only focusing on the processing until the neutral semantics is obtained, and then looking at how this neutral semantics can be input to a visualisation module that demonstrates the extended event structure pictorially. This architecture is shown in Figure 7.1.

7.1.1 Getting the neutral semantics

This is the first processing module, which itself comprises two stages. Its function is to extract as much information as possible from the language, to produce a semantics that does not yet take into account any of the context.

Briefly, the entities that serve as input to this processing are as follows:

- *Linguistic Entity:* The form of the linguistic entity has been described in Section 6.1.3. It is basically a sentence whose words are all separated by commas, and must include indications of where ellipsis or gaps from constituent movement occur.
- *Lexicon:* The lexicon encodes a lot of syntactic and semantic information. This approach also allows easy reconfiguration of some of the semantics; in particular it is easy to change the verb classifications—whether verbs imply discrete or continuous activity, for example—as discussed in Section 5.2. The form of the lexicon is described in 6.1.3, while Appendix A.2 contains a representative subset of the lexical entries that have been implemented.
- *Grammar:* The grammar has been described in detail in Chapter 6 and is listed in Appendix A.1; it is a phrase-structure grammar that uses path equations to associate representations with sentences. Any temporal measure adverbials and any quantified objects in these representations are marked as such, for use in performing scoping.

The representations that are produced are feature structures, of the following form:

- *Unscoped representations:* As stated above, after applying the grammar, unscoped representations of the input language are produced. Quantified terms for objects are represented in features called [`term=`], while quantified terms for temporal measure adverbials are represented by features describing the activity structure they would produce. An unscoped representation contains a representation of the basic happening, plus a list of modifiers from the temporal measure adverbials.

- *Neutral Semantics:* The neutral semantics is the meaning that can be extracted from the linguistic entity, that is neutral to the context. Using grammar rules, and then a scoping algorithm, we obtain a semantics for the linguistic item. This contains the information given by the instruction itself, but does not yet include information from the domain or execution context.

The two modules involved in processing are as follows:

The PATR-II engine

The PATR-II engine is a parser for a PATR-II grammar formalism. It can be seen as consisting of a parsing component (analogous to a simple context-free parser) and an implementation of graph unification expressed via path equations. the grammar that the engine uses is in the form of phrase structure rules that are annotated with path equations. As was stated in Section 6.1.3, the PATR-II implementation is one that was developed in Prolog by Robert Dale, and which I have extended as follows:

- I have included an ability to process productions with less or more than exactly two items on the right-hand-side.
- The engine is wrapped in Prolog code that processes a discourse in a simple fashion.

Scoping

The scoping algorithm, described in Section 7.2, takes an unscoped representation, and performs scoping on it to produce as many representations as are available subject to syntactic and semantic well-formedness of the feature structure.

The result of the scoping is the production of a neutral semantics.

7.1.2 Instruction interpretation

Any comprehensive understanding system needs to incorporate information from places other than the linguistic item itself. For instruction understanding, as well as world knowledge and information from the linguistic context, information about available resources is important. Thus, instruction interpretation involves applying the neutral semantics obtained from the linguistic item to an execution context, that contains resource information. I defer discussion of this to Chapter 9; I have not included any resource incorporation in the demonstration system that has been built, but the architecture for doing so—in particular, the separation of context-related processing from that which is only language dependent—exists.

7.1.3 Visualisation

In order to demonstrate that the neutral semantics is meaningful, I include an implementation of a simple visualisation. This is described in more detail in Section 7.3. The visualisation is able to demonstrate extended activity structure from a neutral semantics.

7.1.4 Multiple readings

It is important to be aware of the various stages at which ambiguity may occur. For each sentence that is processed, there may be be more than one possible semantic representation for it. This can occur at one of three places:

1. If the sentence is syntactically ambiguous, then the PATR-II engine will produce one representation for each syntactic structure.

2. For each syntactic structure, there may be be more than one semantic structure if there are quantified objects or modifiers. In such cases, the scoping algorithm may produce multiple semantic readings for any one syntactic structure.

3. Finally, if execution context is to be incorporated, there may for each neutral semantics be more than one possible executable semantics, for example if tools are available that allow more than one way of doing things.

In the processing system that has been implemented, all of these levels of ambiguity have been allowed for.

7.2 Scoping algorithm

The algorithm that extracts scopings has been based on the algorithm described by Hobbs & Shieber[17][2] for scoping unscoped logical forms. Their algorithm—summarised in Appendix B.1—is a simple one that examines a logical form for QUANTIFIED TERMS, and systematically 'pulls' these to outer levels of the logical form, ensuring syntactic well-formedness of the logical form.

The Hobbs & Shieber algorithm provides a very useful starting point; however there are three differences that are important in its use here:

1. I am concerned with feature structures, whereas their algorithm applied to logical forms.
2. I am concerned with events, and thus with temporal measure adverbials as well as event quantifiers and object quantifiers.
3. Feature structures already provide some 'structure' that unscoped logical forms do not.

The approach that is taken here is to use the notion of quantified terms, and look through the feature structure to find them. Once they are found, they are pulled into higher levels of the feature structure, in ways that are syntactically[3] appropriate; their position in the unscoped form is replaced by the variable of the three-part quantifier that has been 'pulled'. This is similar to the way the Hobbs & Shieber algorithm works.

[2] I have in fact used an implementation in Prolog of the Hobbs & Shieber algorithm that was developed by Robin Cooper, and modified by Lex Holt, that works on logical forms. I have taken this implementation, and adapted it to work on feature structures. So, while the original implementation works on first-order-like formulas, the implementation here works on Prolog representations of feature structures. I have also extended it it deal with event quantification. I have not included any treatment of opaque predicates or nested quantifier terms, as none of my examples include these.

[3] Here, syntactically refers to the syntax of the feature structure. I note that there are additional notions of syntax, particularly with regard to the relationship of sub-intervals, that should also affect well-formedness; this is discussed further in Section 9.4.2.

However, rather than just pulling all of the quantifiers to the outermost position in appropriate orderings, we also need to ensure that they are pulled to appropriate levels in the feature structure. The level incorporates information about scoping range (the wide-narrow distinction in conventional scoping notions). So, it is a mix of quantifier level within the feature structure, and nesting of entities within the feature structure, that provides the scoping information for both quantified objects, and events that have been quantified by conventional quantifiers or measure adverbials.

7.2.1 Quantified terms

There are two kinds of quantified terms—those pertaining to objects, and those pertaining to events. Both of these contribute to extended event structure, but in slightly different ways. Quantified terms from objects suggest ordinary set structure, as described in Section 5.3, while quantified terms from events—such as measure adverbials, and prepositional phrases—produce delimited masses and ordered sets.

Quantified terms corresponding to objects are represented by feature structure of the form in (7.1a), that are patients or recipients[4] of actions, while those coming from events are of the form shown in (7.1b), and are in a 'first-rest' list of modifiers. Those feature values beginning with uppercase letters indicate I am demonstrating the general form, rather than a specific example; they play the role of variables, and may represent actual values or feature structures.

(7.1) a. $$\left[\mathsf{patient} = \left[\mathsf{term} = \begin{bmatrix}\mathsf{det} = \mathsf{Det}\\ \mathsf{var} = \mathsf{Var}\\ \mathsf{restr} = \mathsf{Restr}\end{bmatrix}\right]\right]$$

[4] Everything that applies to a [**patient=**] is applicable to a [**recipient=**]; for the rest of this discussion I do not explicitly refer to recipients.

b.

$$\left[\begin{array}{ll} \mathsf{first} = & \left[\begin{array}{l}\mathsf{structure} = \mathsf{Structure}\\ \mathsf{index} = \mathsf{Index}\\ \mathsf{extent} = \mathsf{Extent}\end{array}\right] \\ \mathsf{rest} = \mathsf{Rest} & \end{array}\right]$$

A feature structure that is of either of the above forms will be identified as a term, and is a candidate for being pulled. This means it is appropriate to 'pull' that term during the scoping process.

Prolog code for identifying terms in feature structures is given in Appendix B.2.3.

7.2.2 Pulling terms

For any term that is identified, it is 'pulled' by substituting it with an appropriate feature structure, and moving the term, in an appropriate form, to the outside of the feature structure currently being scoped.

Quantified terms from objects Terms that are of the form in (7.1a) are replaced with the feature structure [`patient=Var`]. The feature structure is pulled to the outside and a new feature structure of the form shown in (7.2a) is created. The value of the [`content=`] feature is the old feature structure, with its term removed, and the [`=Rest`] feature or value left in its place.

Quantified terms from events Terms that are of the form in (7.1b) are replaced with the feature structure [`dummyfirst=dummyrest`][5]. The feature structure is pulled to the outside and a new feature structure of the form shown in (7.2b) is created. The value of the [`content=`] feature is the old feature structure (with its replaced term).

---

[5] I note that it might be preferable to replace these with empty feature structures; however this was an implementation restriction.

(7.2) a.

$$\begin{bmatrix} \textsf{structure} = \textsf{ordinary-set} \\ \textsf{index} = \textit{Nextevent} \\ \textsf{extent} = \begin{bmatrix} \textsf{Det} = \textsf{Var} \\ \textsf{restr} = \textsf{Restr} \end{bmatrix} \\ \textsf{content} = \_ \end{bmatrix}$$

b.

$$\begin{bmatrix} \textsf{structure} = \textsf{Structure} \\ \textsf{index} = \textsf{Index} \\ \textsf{extent} = \textsf{Extent} \\ \textsf{content} = \_ \end{bmatrix}$$

The Prolog implementation of this is via the predicate `apply`; it is included in Appendix B.2.2.

7.2.3 The overall scoping process

I begin with a clarification of the distinction between being scoped and being pulled:

- A feature structure is a candidate for being scoped if it contains at its topmost level, a feature of the form [`structure=X`]. This means that it is appropriate to look in that feature structure for quantified terms, and to pull any quantified terms found within it to a position that is immediately outside of the feature structure. In the feature structures in (7.3), which correspond to different syntactic structures for '*Simmer the soup stirring occasionally for five minutes*', there are in fact three feature structures that are candidates for being scoped. These are the ones beginning with [`structure=concurrent`], [`structure=happening`] and [`structure=happening`].

- A term (itself in the form of a feature structure) is a candidate for being pulled if it is an appropriate quantified term, as described in Section 7.2.1. In the feature structures in (7.3), there is only one of these—the feature [`first=Firstmod`]

The scoping of an entire feature structure is done by recursively scoping the feature structures within it. The quantified term to be pulled can only be pulled to a position that is immediately outside of the candidate for scoping in which it is found. Thus the scoping process is controlled. The unscoped feature structure in (7.3a), which is a simplified representation of '*Simmer the soup, stirring it, for five minutes*'[6], demonstrates why this is important. In this, it is the happening e9 that is being modified; when scoping is performed, it is important that this modifier ends up surrounding the happening, and not the entire concurrent event e6. In (7.3b) however, which represents an alternative syntactic structure of the same sentence[7], the modifier should be pulled to scope the entire activity of e6.

(7.3) a.

$$\left[\begin{array}{l} \mathsf{structure} = \mathsf{concurrent} \\ \mathsf{index} = \mathsf{e6} \\ \mathsf{substance} = \left[\begin{array}{ll} \mathsf{major} = & \left[\begin{array}{l} \mathsf{structure} = \mathsf{happening} \\ \mathsf{index} = \mathsf{e2} \\ \mathsf{substance} = \mathsf{Substance1} \\ \mathsf{modifiers} = \mathsf{end} \end{array}\right] \\ \mathsf{minor} = & \left[\begin{array}{l} \mathsf{structure} = \mathsf{happening} \\ \mathsf{index} = \mathsf{e9} \\ \mathsf{substance} = \mathsf{Substance2} \\ \mathsf{modifiers} = \left[\begin{array}{l} \mathsf{first} = \mathsf{Firstmod} \\ \mathsf{rest} = \mathsf{end} \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]$$

[6] This corresponds to the the syntactic analysis where '*for fifteen minutes*' is modifying only '*simmer the soup*'

[7] Here, '*for fifteen minutes*' modifies '*simmer the soup stirring occasionally*'

b.
$$\begin{bmatrix} \text{structure} = \text{concurrent} \\ \text{index} = \text{e6} \\ \text{substance} = \begin{bmatrix} \text{major} = \begin{bmatrix} \text{structure} = \text{happening} \\ \text{index} = \text{e2} \\ \text{substance} = \text{Substance1} \\ \text{modifiers} = \text{end} \end{bmatrix} \\ \text{minor} = \begin{bmatrix} \text{structure} = \text{happening} \\ \text{index} = \text{e9} \\ \text{substance} = \text{Substance2} \end{bmatrix} \end{bmatrix} \\ \text{modifiers} = \begin{bmatrix} \text{first} = \text{Firstmod} \\ \text{rest} = \text{end} \end{bmatrix} \end{bmatrix}$$

The algorithm that performs the overall scoping is thus a controlled version of the basic Hobbs & Shieber algorithm, which recursively traverses a feature structure, to find the innermost [`structure=`] feature, and then performs scoping—as it comes out of the recursion—to the level of the feature structure, which will be a [=`happening`], in which the term was found. The algorithm can be summarised as follows; the Prolog code that implements it appears in Appendix B.2.1.

```
To scope a feature structure FS --
  If FS directly includes a feature structure of the form
          'structure : FS2'
  then
     1. scope all the other parts of FS
     2. pull
          'structure : FS2'


To pull a feature-value pair  'structure : FS' --
  Non-deterministically select quantified terms within FS, and
    for each one, apply the quantified term to 'structure : FS'
    to construct a scoped feature structure
```

```
To apply a quantified term Q to a feature structure FS,
  and return NewFS --
    If Q is of the form
      'first : Fvalue'
    then
      1. substitute an empty feature structure for Q in FS
      2. make NewFS, which is of the form
              '[Fvalue,
                content : FS]'

    If Q is of the form
      'patient : [ term : [det : Det, var : Var, restr : Restr ]]'
    then
      1. substitute 'patient : Var' for Q in FS
      2. make NewFS, which is of the form
              '[structure : ordin-set,
                index     : Nextindex,
                extent : [Det : Var,
                          restr : Restr],
                content : FS ]
```

7.2.4 Resolving happening types

It is only once scoping has been done that the [type=] of a happening—if it comes from a verb that has [verbclass=] [=iterable]—can be resolved. This is decided by the structure of the extended event immediately enclosing the happening. If it is a [=delimited-mass], or a [=conjunctive], then the underlying happening becomes [=continuous]; otherwise it becomes [=discrete]. This resolution has to be delayed to after scoping, because it is possible to have a happening that in one scoping is enclosed in a delimited mass, and in another it is enclosed in a set; depending on the order of pulling. The example in Section 8.6 demonstrates this.

7.3 Graphical representations

Throughout this thesis, I have made use of the notion that representing the results of semantic processing visually is useful. In this section I describe the manner in which this notion has been implemented.

As has been indicated, this visual representation's main purpose is to show the results of the semantic analysis in a form other than feature structures or logical forms; to demonstrate that the analysis described in Chapters 5 and 6 results in something meaningful.

I also draw, again, on the analogy between objects and events. The visual representations that are produced can be seen as 'the structure of objects in space', that perform the role of demonstrating 'the structure of eventualities in time'. To this end, I use simple time graphs, where the time axis moves from left to right across the page, and use spatial composition (nesting) to demonstrate temporal composition (event sub-structure).

As is the case with the feature structure representations, there are two basic entity types that are used:

1. Shaded blocks, representing happenings. There are two kinds of shaded blocks: one for representing discrete happenings and one for representing mass happenings. This is elaborated further in Section 7.3.2.

2. The arrangement of shaded blocks spatially on the time graph, representing structurings. There are two basic kinds of arrangement—for mass structurings and for set structurings—and complex structurings are made through 'arranging the arranged sub-structures' in much the same way that the feature structures for complex events described in Section 5.3.4 were made by structuring structurings. Arranging components is elaborated further in Section 7.3.3

I note that Rock[42] describes much of the material that is presented in this section; in particular the practical and philosophical motivation for this implementation is discussed in more depth there, as is the overall functionality of the provision of time graphs. However, what I describe here is a more comprehensive approach that deals with a wider range of structurings in a more rigorous way. I note also that the terminology used in that paper is different; the terms PROCESS and HAPPENING used there, refer to what are called CONTINUOUS and DISCRETE happenings, respectively, here.

7.3.1 Visualising extended activity described in language

To briefly recap, I am only concerned with visualising that degree of structure that can be determined from the language. Contrast the following language fragments:

(7.4) a. Albertina bought a dining suite.

b. Albertina bought eight chairs.

(7.5) a. Do the pirouette sequence.

b. Do that pirouette 16 times.

In (7.4a) and (7.5a), there is no suggestion[8] that there is more than one object or eventuality, and each sentence could be represented by an unstructured object or eventuality entity. However, in (7.4b), we clearly have access to eight objects, and in (7.5b) there is an eventuality whose execution would involve the execution of 16 sub-eventualities.

Appealing to the object-eventuality analogy, activity structure may be represented in the form of object structure, where each activity is represented as an object and sub-activities are represented as sub-objects.

It is feasible to represent the eventuality structure of (7.5b) by drawing an unmarked time graph that contains 16 entities, as in Figure 7.2. That is, the structure of the entire eventuality is represented by the outer block; it contains—within it and arranged along the time-axis—16 smaller blocks. Each of these blocks represents one instance of ≪*do that pirouette*≫ and all of the blocks have boundaries to indicate that their extent is defined. So, using objects—the blocks—and arranging

[8] It is world knowledge, rather than the language itself, that enables us to know that these entities may consist of multiple objects or events.

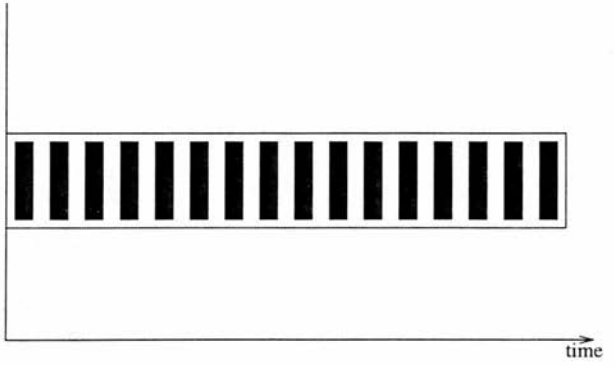


Figure 7.2: Visualising the eventuality structure of '*Do that pirouette 16 times*'.

them spatially with respect to each other, it is possible to represent events and their temporal relationships[9].

7.3.2 The components—shaded blocks

In earlier chapters, two basic components—analogous to count and mass objects—have been identified; the basic components are DISCRETE HAPPENINGS and CONTINUOUS HAPPENINGS[10].

Discrete happenings are drawn as shaded rectangular blocks, which are then arranged on a graph to indicate how they occur with regard to time. Different happenings have different shading; a set of discrete happenings, for example, has a number of blocks all with the same shading but arranged in a way on the time graph to show their contribution to the structure of the extended event. A different set of discrete happenings will have a different shading, though again, each element within that set will be shaded in the same way..

Continuous happenings are drawn as shading between horizontal lines. The temporal delimitation of these events is absent. However, once a delimiter is present—

[9] I note that because I am not interested in representing exact timing, but concepts that are relevant to temporal substructure, the graph is not exact spatially. Rather, it represents in spatial terms the extended event *structure*. In Section 7.3.5, some of the problems of visualisation are discussed.

[10] These terms were introduced in Chapter 5

Figure 7.3: Visualising basic happenings.

such as might come from '*for three hours*', turning the eventuality into a delimited mass—the process is represented as a block, similar to a happening. This is a reasonable approach; once '*water*' becomes '*a glass of water*', it has become a count object. Figure 7.3 shows visualisation of discrete and continuous happenings.

A key is also provided, that indicates which happenings are represented by the different blocks. The key consists of one block for each basic happening kind in the discourse, which is annotated with the text corresponding to the execution of that happening. Essentially, this is the information that is in the [`substance=`] feature of the semantic representation.

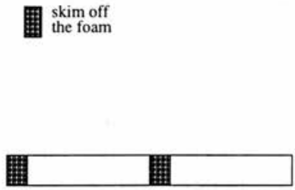
7.3.3 Arranging the components: time graphs

Ultimately, the pictorial representation requires arranging the basic components on a time line, to reflect the understanding of the repetition. Descriptions of an extended activity have information that contains more detail about the activity structure. For example, we may know that a discrete happening occurs ≪*every 10 minutes*≫, or that a continuous happening occurs ≪*for an hour*≫. The individual components and timelines are used. These are arranged in a particular way with annotation, to include the depiction of this additional information. The passage of time is depicted from left to right across the page.

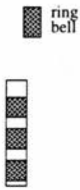
When visualising extended events, the following arrangements are used:

Sets always contain either discrete happenings or extended events as elements; they are never composed of continuous happenings unless the happening is part of a delimited event.

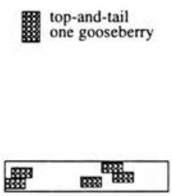
Ordered sets The elements of ordered sets are arranged horizontally along the time axis. There is space between elements—appropriate to the frequency and regularity of execution.



Simultaneous sets For a simultaneous set, the elements are arranged in the same vertical line



Ordinary sets To indicate that we know nothing of the temporal relation between set elements, these sets are drawn with their elements randomly placed. This allows for the possibility of the temporal relations between subevents being dependent on context[18].



Masses Only delimited masses are visualised; masses without delimitation will occur only in the key. The delimitation is annotated. An example of a delimited mass is shown in Figure 8.4.

Complex activity :

Concurrent The two participants in concurrent activity are arranged vertically with respect to each other—the major activity appears above the minor one. The arrangement within a participant is as it would be for activity of its structure. An example of concurrent activity is shown in Figure 8.6.

Compound Arrangement of activity that is made of consecutive execution of different activity is simply done horizontally and adjacently. The discourse example of Figure 8.7 shows this.

7.3.4 A computational system that visualises event structure

In order to show the feasibility of the approach described above, I have developed a program that takes semantic representations of language about extended events, as described in Chapter 5, to produce graphs that show the event structure visually.

The drawing is done by a PostScript[1] program—all that is input to the PostScript is a single data structure[11] for each discourse[12]. First, I describe the overall drawing process using an example. Then I discuss some issues pertaining to the implementation.

The overall drawing process

A semantic analysis of the discourse in (7.6a) determines that the activity consists of two concurrent activities—simmering the soup and stirring the soup. Further, stirring is a discrete happening; that it occurs frequently gives the information that it is part of an activity that is an ordered set. The simmering the soup activity is a delimited mass that has a duration of one hour. The semantic analysis produces the feature structure in (7.6b), which is in turn represented by the PostScript data structure in (7.6c).

(7.6) a. Simmer the soup for one hour, stirring frequently.

b.
```
structure : concurrent
index : e0
```

[11] This data structure is what in PostScript is an array, which is syntactically similar to a Prolog list. That is, it is a single data item, demarcated by angle brackets. It contains zero or more items, which may themselves be of any sort, and in fact will sometimes be (PostScript) arrays. One syntactic difference to a Prolog list is that in PostScript, elements in the array are separated by spaces rather than commas.

[12] A discourse is one or more sentences; each sentence is in the form of an instruction.

```
substance : major : structure : delimitedmass
                    index : e1
                    extent : term : det : 1
                                    var : w
                                    restr : hours
                             extent_type : time
                    content : structure : happening
                              index : e2
                              substance : action : simmer
                                          patient : soup
                              type : continuous
            minor : structure : ordered_set
                    index : e3
                    extent : det : often
                             var : x
                             restr : event
                    content : structure : happening
                              index : e4
                              substance : action : stir
                                          patient : ellided
                              type : discrete
```

c.

```
[[60  [concurrent [1 3]]]
 [60  [delimited-mass [2 (one hour) ]]
 [ 0  [happeningc [(simmer) (soup) ]]]
 [60  [ordered-set [12 4]]]
 [ 2  [happeningd [(stir) (ellided) ]]]  ]]
```

(7.6c) is a PostScript array, containing five sub-arrays, one for each activity. Thus, numbering the arrays 0–4, the activities each represents are:

0 ≪*simmering the soup for one hour, stirring frequently*≫

1 ≪*simmering the soup for one hour*≫

2 ≪*simmering the soup*≫

3 ≪*stirring the soup frequently*≫

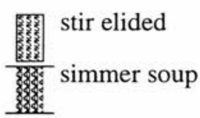
4 ≪*stirring the soup*≫

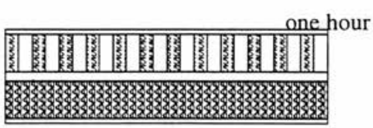
To enable the actual drawing of the blocks, each activity is given a size; conveniently this could be related to the time that such an activity might take. I note that in the current implementation, this size is entered manually; there is no facility for reasonably calculating appropriate block sizes. In the array then, the number at the left is the size of the activity; we could imagine it represents in minutes the time taken for the activity to occur.

Each happening is accompanied by an indication of whether it is discrete or continuous, as well as text describing the activity. This text is used in the creation of the key; elements 2 and 4 in the array of (7.6c) are examples.

For the activity that is structured, the places where the activity descriptions will be found (simply the element numbers) follow. Element 0 is an example—the concurrent elements of this activity are found in elements 1 and 3. For sets, the cardinality and the place where the description of the set elements will be found follows; thus in element 3, the ordered set is made of twelve instances of element 4.

This is input into the PostScript program, which produces this pictorial representation. Each activity type (such as **`happeningc, ordered-set`**) corresponds to a call to a PostScript routine. Thus, additional activity types can be added in a modular fashion, by adding new routines. More importantly, changes to the way an activity type is drawn can also be made in a modular fashion, if we wish for instance to depict a more comprehensive analysis of intra-event relations as mentioned in Section 3.2.4.





In Chapter 8, visualisation of other examples, including discourse of more than one sentence, is presented. Appendix C describes the implementation of the visualisation in more detail.

Converting a scoped feature structure to a PostScript array

PostScript arrays, appropriate for input to the visualisation program, are produced from scoped feature structures of the form presented in this and the previous chapters. The algorithm to do this involves recursively traversing the feature structure to find any feature that is a [`structure=`], and creating a new array element for it. Depending on the value of that structure, the array element is given appropriate values; for happenings this will simply be the annotation that describes the happening, while for structurings, further recursive traversal of the feature structure is required to determine the appropriate elements that should be part of that structure.

So, for sets, delimited masses, concurrent activities, and complex activities, a recursive traversal of that feature structure identifies its constituent activities, at the same time as creating the array element for each constituent activity. The array index of the created element is passed back and incorporated into the array element of the overall structured activity.

The actual values for iteration—for example, a numeric value corresponding to '*a few times*'—are implemented via a lookup table; the sizes of blocks however must be input manually. The algorithm has been implemented in Prolog, and is presented in Appendix C.3.

7.3.5 Discussion

As well as advantages, there are of course limitations to visualisation, and these come in two forms—those pertaining to visualisation itself and those pertaining to any computational implementation of visualisation.

Limitations to visualisation

The principal difficulty with visualisation as described above is the implicit information that comes along with any picture. It is unclear how best to exclude things from being represented by implication. This is true of things like implied groupings and associations, implied regularities, implied common properties, implied relative sizes, and more.

If an extended event is represented by evenly spaced boxes, there is an implication that the repetition occurs at equal time intervals; boxes of similar sizes will be assumed to represent events that occur for similar lengths of time. In both of these cases, it is possible to imagine examples where the evenness or the temporal similarity should not be implied. The solution of say spacing the boxes unevenly, may, rather than removing the implication, imply the opposite—that the events explicitly occur at random intervals.

A second problem is how to visualise concepts like '*a few times*' or '*a hundred times*'. For the first, representing three blocks may imply that '*three times*' is intended. For the second, do we draw one hundred blocks? How do we fit them all in? Or do we use dots to indicate some missing items?

Is a diagram a more specific version of a text description? Is it sometimes a less specific version? The former seems more plausible—a diagram is often a concrete realisation of some concept; an 'instantiation'. The diagram itself has a size; because of this it is sometimes difficult to represent size in the abstract, and it is difficult to exclude size and shape and other concepts from being represented by implication.

The value of visualisation in the context of natural language semantics

Although the visualisation has some limitations, it is extremely useful in providing a means of representing semantic information in a medium that is distinct from the conventional ways of doing this, such as logical forms. Moving to a different medium has advantages that include being able to represent information that could not otherwise be easily represented—such as the passage of time. It is also argued by Ludlow[25] that applying the results of semantic processing to another application can be a good test of the validity of semantic output.

7.4 Summary

In this chapter we have seen an architecture for a system for understanding instructions and visualising this understanding. We have also seen an actual implementation of the important parts of such a system. The computational system that has been described in this chapter, that utilises the grammar and representations described in the two previous chapters, was implemented in order to demonstrate the validity of the theory proposed in earlier chapters.

The system is able to process a range of instructions that contain language in them that is about extended activity, and produce semantic representations of them—first in the form of feature structures and then in the form of diagrams. The implementation includes a lexicon and grammar that caters for a range of verbs, temporal measure adverbials, noun phrases and other syntactic items that are needed to form coherent instructions. It is also able to process a series of instructions.

In particular, the following have been discussed in some detail:

- a grammar for the obtaining of unscoped forms of linguistic items
- an algorithm for taking such unscoped forms and producing the appropriate scoped forms.
- a visualisation system that takes these forms (once they have been converted to an appropriate input) and produces diagrammatic representations of the event structure they describe.

The coverage that has been provided deals with a range of syntactically different cooking instructions—some of these are listed in Appendix A.3. The PATR-II engine was already available and has had some minor additions made to cope with the range of rules. The process from input to the production of scoped forms, for all of the syntactic readings, is completely under program control. Taking these forms and converting them to an input form that is appropriate for the visualisation is also done under program control. However, the decision regarding the relative sizes of each visual component is done manually; this is because it is not possible to easily know or determine such things, and would require world knowledge, or some form of heuristics perhaps.

For clarity, I note that the automatic generation of unique index numbers (for each activity and sub-activities), and the automatic generation of unique variable names for quantified terms, has not been implemented. Such automation I believe to be a matter pertaining to implementation; a trivial matter and not of crucial importance to any of the theory and implementation presented here.

In the next chapter, the actual operation of this system is demonstrated via the presentation of worked examples that illustrate the computational treatment of the conceptual and linguistic issues identified in Chapters 3 and 4.

Chapter 8

Worked examples

This chapter is devoted to the presentation of some worked examples, that demonstrate the concepts and theory described in earlier chapters, as well as showing the feasibility of a computational implementation of the proposed model of understanding.

I begin with an example of a simple sentence, and show its analysis from start to finish in detail. Then, examples of more complex sentences are given, but with less detailed presentation. These illustrate the various topics presented in the thesis. In particular, examples illustrating each of the issues that were presented in the summary list in Section 4.8 are given. Finally, the processing of a discourse is shown.

8.1 An initial simple example—a prepositional phrase

When the sentence
`[simmer,the,soup,for,five,minutes]`
is given as input to the understanding system, the feature structure in (8.1) is constructed to represent the semantic information of interest:

(8.1)
```
structure : happening
index : e2
substance : action : simmer
            patient : soup
type : continuous
```

```
modifiers : first : structure : delimitedmass
                    index : e5
                    extent : term : det : 5
                                    var : w
                                    restr : minutes
                             extent_type : time
            rest : end
```

This feature structure is obtained through application of appropriate grammar rules described in Chapter 6; the tree that is spread over Figures 8.1, 8.2 and 8.3 shows this. Appendix A.1 also contains a list of all the grammar rules, for easy reference.

The processing of the initial sentence to obtain the unscoped feature structure goes through the following steps:

1. There are two phrases—a verb phrase '*simmer the soup*' and a prepositional phrase '*for five minutes*'.

2. Analysing `[simmer,the,soup]`, shown in Figure 8.1, results in the building of the [=`happening`] whose index is [=`e2`]. The [`patient`=] comes from a simple noun phrase analysis, that identifies `[the,soup]` as not being quantified. The feature structure corresponding to `[simmer,the,soup]` is

   ```
   cat : vp
   sem : structure : happening
         index : e2
         substance : action : simmer
                     patient : soup
         type : continuous
         modifiers : end
   ```

   The list of modifiers—which is the place-holder for temporal-measure modifiers that contribute to eventuality structuring—is empty to begin with, and has the value [=`end`].

3. Analysing `[for,five,minutes]`, shown in Figure 8.2, produces a prepositional phrase, as follows:

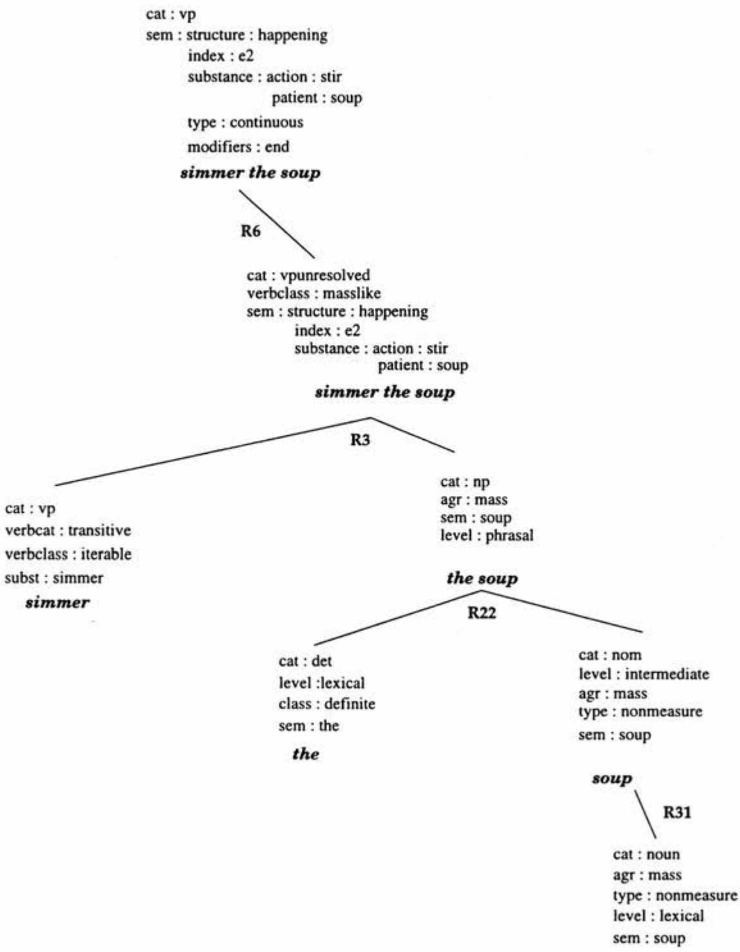


Figure 8.1: Building the feature structure for '*Simmer the soup*'.

```
cat : pp
sem : structure : delimitedmass
      index : e5
      extent : term : det : 5
                      var : w
                      restr : minutes
               extent_type : time
type : tempmeasure
```

Because the prepositional phrase is a temporal-measure one, it is also a modifier.

Combining the resulting verb phrase and prepositional phrase essentially involves adding the prepositional phrase onto the list of modifiers of the verb phrase. As demonstrated in Figure 8.3, this produces the unscoped form—whose semantic representation was shown in (8.1)—which is then input to the scoping algorithm.

After scoping is performed, and the [`first=`] feature is pulled to the outer level of structure from the list of modifiers, we are left with the new—scoped—feature structure shown in (8.2).

(8.2)
```
structure: delimitedmass
index: e5
extent: term: det: 5
              var: y
              restr: minutes
        extent_type: time
content: structure: happening
         index: e2
         substance: action: simmer
                    patient: soup
         type: continuous
```

At this point, visualisation can take place, by transforming the scoped feature structure into the PostScript array

```
[[300  [/delimitedmass [1 (5 minutes) ]]
 [ 0   [/happeningc [(simmer) (soup) ]]] ]]
```

This then serves as input to the drawing program to produce the visual representation shown in Figure 8.4. The size of the picture is set to 300 (taken simply

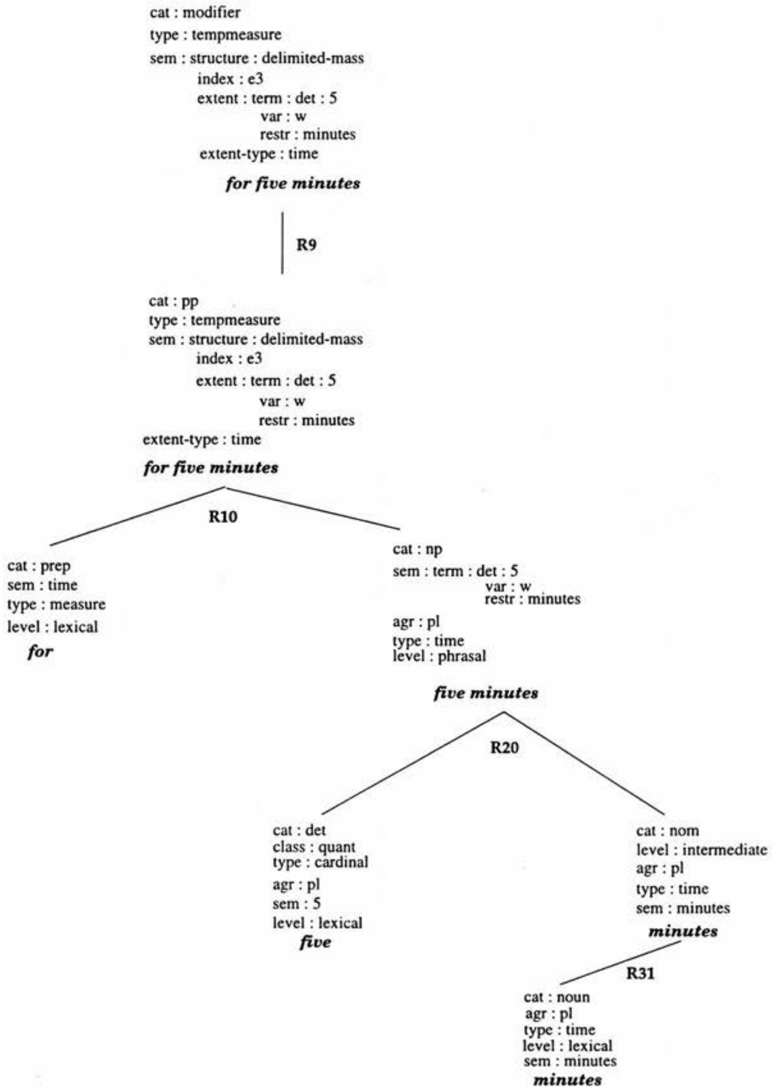


Figure 8.2: Building the feature structure for '*for five minutes*'.

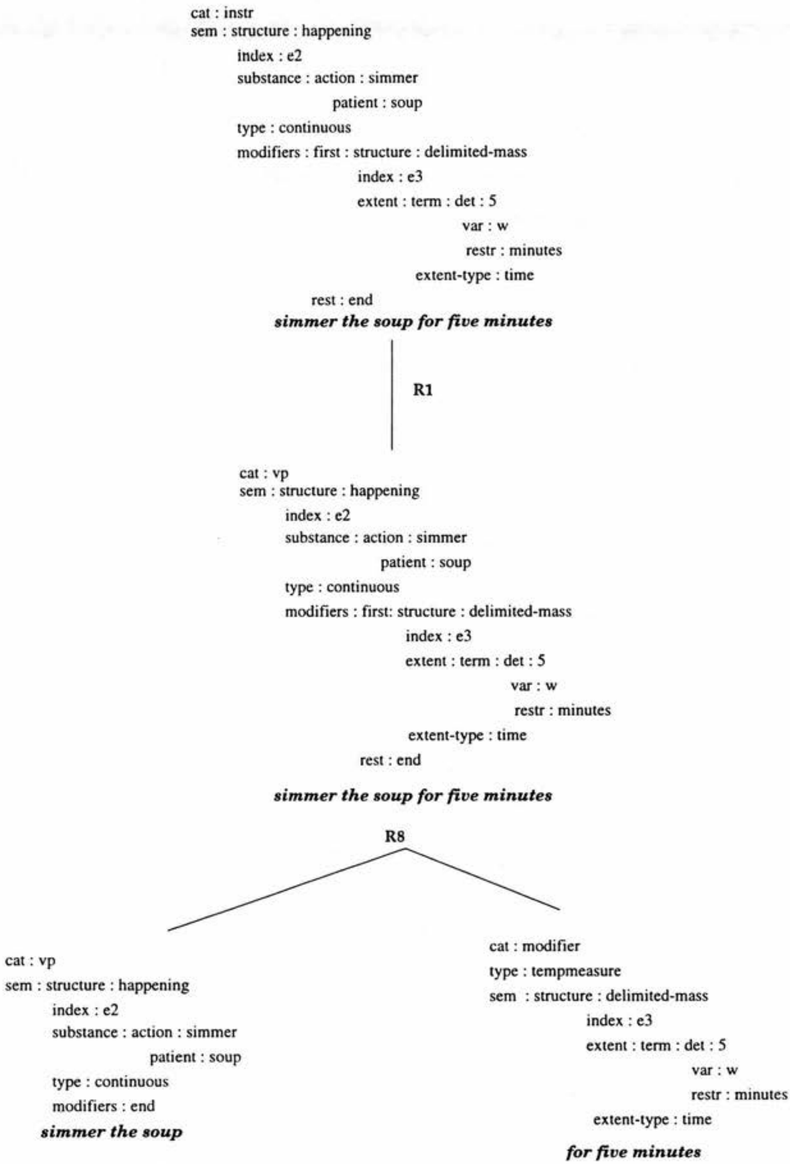


Figure 8.3: Building the feature structure for '*Simmer the soup for five minutes*'.

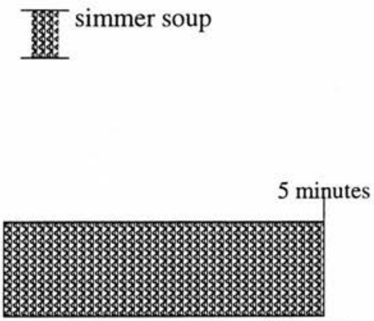


Figure 8.4: Visualising '*Simmer the soup for five minutes*'.

from the number of seconds in five minutes); this is done via a default rather than through any computational reasoning, as was described in Section 7.3.4.

8.2 Plural objects in the verb phrase

This example illustrates in particular the effect of plural or quantified objects in the verb phrase, as in
`[top, and, tail, the, gooseberries]`.

In this case, it is '*the gooseberries*' that will indicate that there is extended activity, and the analysis produces the feature structure shown in (8.3).

(8.3)
```
structure : happening
index : e2
substance : action : structure : compound
                     act1 : top
                     act2 : tail
            patient : term : det : all
                             var : y
                             restr : gooseberries
                      cardinality : numberof
type : discrete
modifiers : end
```

The action described by '*top and tail*' is not decomposed into consisting of two (different) subactivities; for the analysis presented here, the topping and tailing of one gooseberry is one activity. The list of modifiers is empty, and the happening is [=`discrete`] because of the verb classification of '*top*' and '*tail*'.

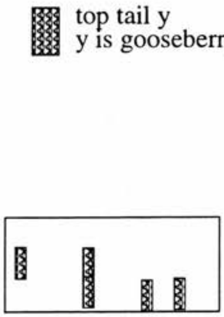


Figure 8.5: Visualising '*Top and tail the gooseberries*'.

The representation in (8.3) is the unscoped form; it is performing scoping, and identifying that the patient is a quantified term—because the [`patient=`] is a quantified term, containing features for [`det=`], [`var=`] and [`restr=`]—that determines that this describes an ordinary set. The cardinality of the set corresponds to the number of gooseberries, and the extent then of the activity is the number of $y's$, where each y has the restriction *gooseberry*.

(8.4)

```
structure : ordinary-set
index : e1
extent : all : y
         restr : gooseberries
content : structure : happening
          index : e2
          substance : action : structure : compound
                                  act1 : top
                                  act2 : tail
                      patient : y
          type : discrete
          modifiers : end
```

An example visualisation is shown in Figure 8.5. I note that the visualisation in this case uses the assumption that there are five gooseberries, representing '*the gooseberries*'.

8.3 Adverbial modifiers

This example illustrates the manner in which adverbials, such as '*every fifteen minutes*', are treated.

`[baste,the,joint,every,hour]`
produces the feature structure below:

```
structure : happening
index : e0
substance : action : baste
            patient : joint
type : discrete
modifiers : first : structure : ordered_set
                    index : e1
                    extent : det : every
                             var : x
                             restr : hours
            rest : end
```

In this case, it is the adverbial modifier '*every hour*', that is a modifier, that gives the extended activity structure. The quantifier '*every*' results in a quantified term. The entire [`first=`] feature from the list of modifiers is pulled during the scoping algorithm[1]. The patient here is not a quantified term, so it is also not pulled.

```
structure : ordered_set
index : e1
extent : term : det : every
                var : x
                restr : hours
content : structure : happening
          index : e0
          substance : action : baste
                      patient : joint
          type : discrete
          modifiers : end
```

[1] Recall that only [`first=`] features—which appear only in modifier lists—and quantified [`patient=`] and [`recipient=`] values—corresponding to event and object quantification respectively—are significant to the extended activity semantics.

8.4 Sentence complements

Sentence complements involving '*until*' are semantically similar to prepositional phrases. Here, the unscoped and corresponding scoped forms are shown for the example
[beat,the,mixture,until,all,the,sugar,has,dissolved].

```
structure : happening
index : e2
substance : action : beat
            patient : mixture
type : iterable
modifiers : first : structure : delimitedmass
                    index : e8
                    extent : structure : state
                             index : e9
                             substance : condition : dissolved
                                         vptype : state
                                         patient : sugar
                             modifiers : end
                             extent_type : condition
            rest : end
```

```
structure : delimitedmass
index : e8
extent : structure : state
         index : e9
         substance : condition : dissolved
                     vptype : state
                     patient : sugar
         modifiers : end
         extent_type : condition
content : structure : happening
          index : e2
          substance : action : beat
                      patient : mixture
          type : continuous
          modifiers : end
```

Note that the extent of the delimited mass is of [`extent-type=`] [=`condition`], in contrast to those of type [=`time`] that are associated with '*for*'-adverbials; additionally, it is of the form of a structure, indicating that the condition depends on some kind of eventuality.

8.5 Combining modifiers with each other

This example illustrates the processing of an instruction where the two modifiers, because of the presence of the conjunction '*or*', are combined with each other before together modifying the activity.

`[steam,the,mussels,for,two,minutes,or,until,they,open].`

The example is further complicated by the presence of a plural object (the quantified term that represents '*the mussels*') as the patient of the activity, which after scoping would result in an ordinary set structure. However, for this example, I focus only on the conjunctive expression of the activity structure.

The modifier (in this case, there is only one, and would be the value of the feature [`first=`]) in the unscoped form is the following:

```
structure : conjunctive
index : e13
extent : extent1 : structure : delimitedmass
                   index : e5
                   extent : term : det : 2
                                   var : w
                                   restr : minutes
                            extent_type : time
         extent2 : structure : delimitedmass
                   index : e8
                   extent : structure : happening
                            index : e1
                            substance : action : open
                                        patient : referent
                            modifiers : end
                            extent_type : condition
```

The features [**extent1=**] and [**extent2=**] are part of the same conjunctive expression of extent; thus only one extended activity structure is indicated[2].

[2] There is a second structuring, indicated by the plural patient in the [=**happening**]. So, after scoping, there would be two readings—one where ≪*each of the mussels*≫ experiences an extended activity of ≪*being steamed for 2 minutes or its opening*≫, and the second where there is an extended activity of ≪*steaming*≫ that happens to ≪*the mussels*≫. I only demonstrate the scoped feature structure for the latter.

Here, the conjunctive expression of the extents is combined before being applied to the activity, which in this case is ≪*the steaming of the mussels*≫. After scoping, the representation is the following:

```
structure : conjunctive
index : e13
extent : extent1 : structure : delimitedmass
                   index : e5
                   extent : term : det : 2
                                   var : w
                                   restr : minutes
                            extent_type : time
         extent2 : structure : delimitedmass
                   index : e8
                   extent : structure : happening
                            index : e1
                            substance : action : open
                                        patient : referent
                            modifiers : end
                            extent_type : condition
content : structure : happening
          index : e2
          substance : [...]
          type : continuous
          modifiers : end
```

I note that to shorten the exposition, I have replaced the feature structure representing '*steam the mussels*' with [...].

8.6 Nesting modifiers

This example deals with the nesting of temporal-measure modifiers, and shows that two scopings are available because there are two appropriate modifiers—an adverbial ('*often*') and a prepositional phrase ('*for half an hour*').

`[stir,the,soup,often,for,half,an,hour]`
corresponds to the unscoped representation that follows:

```
structure : happening
index : e2
substance : action : stir
            patient : soup
```

```
type : iterable
modifiers : first : structure : delimitedmass
                    index : e5
                    extent : term : det : exists
                                    var : w
                                    restr : hours
                             extent_type : time
            rest : first : structure : ordered_set
                           index : e12
                           extent : det : often
                                    var : x
                                    restr : event
                   rest : end
```

Note that '*for half an hour*' is represented simply as an 'hours' modifier; no detail about the fact that is half an hour is represented. Such detail of time is seen as appropriate for further work.

The two [`first=`] modifiers are pulled in scoping, with alternate wide and narrow scope, resulting in the following two representations. The first one represents ≪*the stirring of the soup for half an hour*≫ occurring ≪*often*≫.

```
structure : ordered_set
index : e12
extent : det : often
         var : x
         restr : event
content : structure : delimitedmass
          index : e5
          extent : term : det : exists
                          var : w
                          restr : hours
                   extent_type : time
          content : structure : happening
                    index : e2
                    substance : action : stir
                                patient : soup
                    type : continuous
                    modifiers : end
```

The following represents ≪*the stirring of the soup often*≫ occurring continuously ≪*for half an hour*≫.

```
structure : delimitedmass
```

```
index : e5
extent : term : det : exists
                var : w
                restr : hours
         extent_type : time
content : structure : ordered_set
          index : e12
          extent : det : often
                   var : x
                   restr : event
          content : structure : happening
                    index : e2
                    substance : action : stir
                                patient : soup
                    type : discrete
                    modifiers : end
```

Note that in the second representation, the basic ≪*stirring*≫ happening is a discrete one, because it is directly within a set; in the first reading it is continuous because it is directly within a delimited mass.

8.7 Combining modifiers with quantified or plural objects

Again, multiple[3] scopings are available, but in contrast with the example in Section 8.6, this is because there is a quantified object ('*some soup*') interacting with a temporal-measure modifier ('*every half hour*').

`[stir,some,soup,every,half,hour]`
has the following unscoped representation:

```
structure : happening
index : e2
substance : action : stir
            patient : term : det : some
                             var : w
                             restr : soup
type : iterable
modifiers : first : structure : ordered_set
```

[3] There are always potentially $n!$ scopings when there is a total of n modifiers and quantified objects.

```
                index : e12
                extent : det : every
                         var : y
                         restr : hours
        rest : end
```

Here, the [first=] feature and the [=term] of the [patient=] are both pulled, again to alternate wide and narrow scopes for the two representations.

The first representation shows ≪*some particular soup*≫ being ≪*stirred every half hour*≫.

```
structure : ordin_set
index : e99
extent : some : w
         restr : soup
content : structure : ordered_set
          index : e12
          extent : det : every
                   var : y
                   restr : hours
          content : structure : happening
                    index : e2
                    substance : action : stir
                                patient : w
                    type : discrete
                    modifiers : end
```

The following represents ≪*every half hour*≫, the ≪*stirring of some soup*≫ occurring.

```
structure : ordered_set
index : e12
extent : det : every
         var : y
         restr : hours
content : structure : ordin_set
          index : e99
          extent : some : w
                   restr : soup
          content : structure : happening
                    index : e2
                    substance : action : stir
                                patient : w
                    type : discrete
                    modifiers : end
```

In both cases, the [happening=] is immediately enclosed in a set, and so the happening [type=] is [=discrete].

8.8 Combining verb phrases

Here, I present the analysis of
`[simmer,the,soup,for,fifteen,minutes,`
`stirring,ellipsis,occasionally]`.

This sentence is syntactically ambiguous; it is possible for the adverbial '*occasionally*' to modify either only the gerundive verb phrase '*stirring*', or to modify the whole verb phrase '*simmer the soup for fifteen minutes, stirring*'. We get two feature structures after grammatical analysis. The first, corresponding to ≪*simmering for fifteen minutes while stirring*≫ happening ≪*occasionally*≫ has the following representation:

```
structure : concurrent
index : e6
substance : major : structure : happening
                    index : e2
                    substance : action : simmer
                                patient : soup
                    type : continuous
                    modifiers : first : structure : delimitedmass
                                        index : e5
                                        extent : term : det : 15
                                                        var : w
                                                        restr : minutes
                                                 extent_type : time
                                rest : end
            minor : structure : happening
                    index : e7
                    substance : action : stir
                                patient : elided
                    type : iterable
                    modifiers : end
modifiers : first : structure : ordered_set
                    index : e12
                    extent : det : occasional
                             var : x
                             restr : event
```

```
            rest : end
```

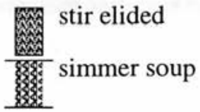
The second syntactic structure, corresponding to ≪*simmering*≫ while ≪*stirring occasionally*≫, gives rise to:

```
structure : concurrent
index : e6
substance : major : structure : happening
                    index : e2
                    substance : action : simmer
                                patient : soup
                    type : continuous
                    modifiers : first : structure : delimitedmass
                                        index : e5
                                        extent : term : det : 15
                                                        var : w
                                                        restr : minutes
                                                 extent_type : time
                                rest : end
            minor : structure : happening
                    index : e7
                    substance : action : stir
                                patient : elided
                    type : iterable
                    modifiers : first : structure : ordered_set
                                        index : e12
                                        extent : det : occasional
                                                 var : x
                                                 restr : event
                                rest : end
modifiers : end
```

It is clear from comparing these two representations that the feature [`first=`] representing '*occasionally*' modifies the whole verb phrase in the first example, and only the ≪*stirring*≫ in the second.

The above are unscoped forms; I show the scoped form corresponding to the second analysis:

```
structure : concurrent
index : e6
substance : major : structure : delimitedmass
                    index : e5
                    extent : term : det : 15
```



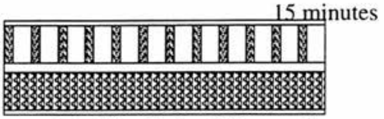


Figure 8.6: Visualising '*Simmer the soup for fifteen minutes stirring occasionally*'.

```
                              var : w
                              restr : minutes
                       extent_type : time
              content : structure : happening
                        index : e2
                        substance : action : simmer
                                    patient : soup
                        type : continuous
                        modifiers : end
      minor : structure : ordered_set
              index : e12
              extent : det : occasional
                       var : x
                       restr : event
              content : structure : happening
                        index : e7
                        substance : action : stir
                                    patient : elided
                        type : discrete
              modifiers : end
modifiers : end
```

A visualisation appropriate to this is shown in Figure 8.6. Again as before, ≪*occasionally*≫ has been given a numeric value in order for drawing to take place, and the respective dimensions of the different components have also been included.

8.9 Processing a discourse

To conclude this chapter, I present the computational processing of a simple discourse, that of (8.5).

(8.5)
```
[[melt,the,butter,until,it,is,foaming],
[skim,the,foam,and,discard,it],
[heat,the,butter,and,skim,it,again],
[do,this,twice,more]]
```

This produces two syntactic analyses due to the sentence '*Heat the butter and skim it again*'. The first sense is where the ≪*heating and skimming*≫ occurs ≪*again*≫; the second is where the ≪*heating*≫ occurs, followed by ≪*skimming again*≫.
The semantic representation of the overall discourse would be of the form:

```
structure : compound
index : e100
participants : [list of participants]
```

I show the list of participants in the complex event that represents the above discourse, for the second reading; each participant, at the top level, corresponds to one of the sentences.

```
/--
  structure : happening
  index : e0
  substance : action : melt
              patient : butter
  type : continuous
  modifiers : first : structure : delimitedmass
                      index : e1
                      extent : structure : state
                               index : e2
                               substance : condition : be_foaming
                                           vptype : state
                                           patient : referent
                               modifiers : end
                               extent_type : condition
              rest : end
|--
  structure : compound
```

```
  index : e3
  substance : participant1 : structure : happening
                             index : e4
                             substance : action : skim
                                         patient : foam
                             type : discrete
                             modifiers : end
              participant2 : structure : happening
                             index : e5
                             substance : action : discard
                                         patient : referent
                             type : discrete
                             modifiers : end
  type : nil
  modifiers : end
|--
  structure : compound
  index : e6
  substance : participant1 : structure : happening
                             index : e7
                             substance : action : heat
                                         patient : butter
                             type : continuous
                             modifiers : end
              participant2 : structure : happening
                             index : e8
                             substance : action : skim
                                         patient : referent
                             type : discrete
                             modifiers : first : structure : ordered_set
                                                 index : e9
                                                 extent : det : again
                                                          var : x
                                                          restr : event
                                         rest : end
  type : nil
  modifiers : end
|--
  structure : happening
  index : e10
  substance : action : act : do
                       sem : activity
              patient : referent
  type : discrete
  modifiers : first : structure : ordered_set
                      index : e11
                      extent : det : two
                               var : y
```

```
                                   restr : event
              rest : end
\--
```

Scoping produces only one scoped form for the discourse, as there is no interaction of modifiers[4] and there are no quantified objects.

```
structure : compound
index : e100
participants :

/--
  structure : delimitedmass
  index : e1
  extent : structure : state
           index : e2
           substance : condition : be_foaming
                       vptype : state
                       patient : referent
           modifiers : end
           extent_type : condition
  content : structure : happening
            index : e0
            substance : action : melt
                        patient : butter
            type : continuous
            modifiers : end
|--
  structure : compound
  index : e3
  substance : participant1 : structure : happening
                             index : e4
                             substance : action : skim
                                         patient : foam
                             type : discrete
                             modifiers : end
              participant2 : structure : happening
                             index : e5
                             substance : action : discard
                                         patient : referent
                             type : discrete
                             modifiers : end
  type : nil
  modifiers : end
```

[4] Recall that a quantified term is pulled to just outside of the level of the structure in which the term is found.

```
|--
  structure : compound
  index : e6
  substance : participant1 : structure : happening
                             index : e7
                             substance : action : heat
                                         patient : butter
                             type : continuous
                             modifiers : end
              participant2 : structure : ordered_set
                             index : e9
                             extent : det : again
                                      var : x
                                      restr : event
                             content : structure : happening
                                       index : e8
                                       substance : action : skim
                                                   patient : referent
                                       type : discrete
                                       modifiers : end
  type : nil
  modifiers : end
|--
  structure : ordered_set
              index : e11
              extent : det : two
                       var : y
                       restr : event
              content :  structure : happening
                         index : e10
                         substance : action : act : do
                                     sem : activity
                                     patient : referent
                         type : discrete
                         modifiers : end
\--
```

This then translates to the following PostScript array with accompanying visualisation in Figure 8.7.

```
[[750 [compound [1 3 6 10]]]                          0
 [ 60 [delimitedmass [2 [(referent) (foam)]]]]        1
 [  0 [happeningc [(melt) (butter) ]]]                2
 [ 60 [compound [4 5]]]                               3
 [ 30 [happeningd [(skim) (foam)]]]                   4
```

```
[ 30 [happeningd [(discard) (referent)]]]        5
[210 [compound [7 8]]]                           6
[ 90 [happeningd [(heat) (butter)]               7
[ 90 [ordered-set [1 9]]                         8
[ 90 [happeningd [(skim)]]]                      9
[420 [ordered-set [2 6]]]   ]                   10
```

The overall compound activity, in array element 0, consists of

1. the delimited mass in 1
2. the compound activity in 3
3. the compound activity in 6
4. the ordered set in 10

These subactivities themselves consist of sub-activities, as indicated in the array.

I note that there is no sophisticated processing that resolves '*this*'—represented by [=`referent`] as the [`patient=`] of the happening—in the last instruction to refer to the third sentence (the compound activity in array element 6 in fact)—rather than the whole preceding discourse, or just the ≪*skimming again*≫, for example. In this example, such a resolution has been done manually. Such processing has been identified as a valid topic for future research and is discussed in the final chapter.

It is also the case that language fragments like '*again*' and '*twice more*' are not analysed to any great depth, other than to identify that they contribute to extended activity. Again, such analysis would be useful.

The PostScript array, other than determining the relative physical sizes of the visual components, and other than resolving the anaphoric event reference described above, is produced automatically from the scoped feature structure.

This discourse example has been included in order to show the range of processing that has been included, and its limits, and to identify some of the issues in the inter-relation between the processing of single instructions. I believe it provides an appropriate indication for further work.

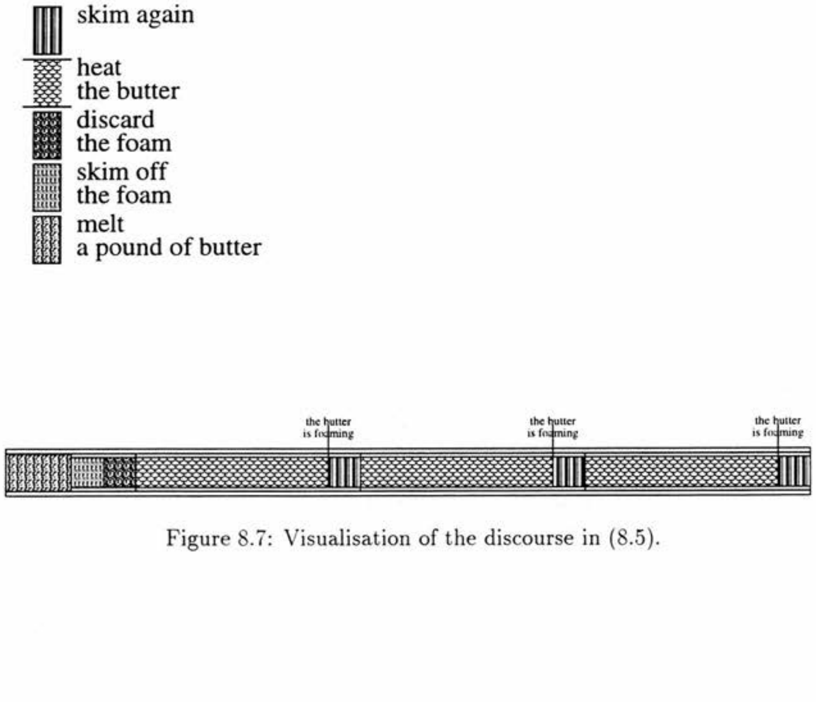


Figure 8.7: Visualisation of the discourse in (8.5).

Chapter 9

Discussion and Conclusions

9.1 Summary

The story that I have tried to tell in this thesis is as follows:

There are entities in the world that we conceive of as eventualities, which are things that happen or are states of being. This is in contrast to objects, which exist.

A particular subset of eventualities is activities, which are things that happen; things that occur. Sometimes activities may occur repeatedly, or there may be multiple instances of an activity, or an activity may occur continuously. I have noted that this view of activity is a way of conceptualising structure, and I have also noted that we use language to express such structure. I call activities that are structured this way, extended. It is these extended activities, and the language that we use to talk about them, on which the thesis has focussed.

The contrast of eventualities to objects is not incidental. Philosophical work on the object-eventuality analogy has been drawn on, and this has been concretised. In particular, activities of the type that are of interest—extended activities—are seen as the eventuality analogue of plural and mass objects.

I have presented an analysis of those parts of language that we use to express extended activity structure, and I have implemented a computational system that is able to perform such analysis on a restricted subset of English. The restriction I have chosen is instructions, which have a special relationship to activity; they are templates for activity.

The analysis focuses clearly on the information that can be obtained from the language itself. I have argued that context—execution context, as well as world knowledge—will always play a part in understanding of language. With this in mind, although the analysis does not perform context incorporation, it does clearly indicate the places where context might play a part.

I have also exploited the object-eventuality analogy in performing analysis. I have taken the approach that the language we use to express extended activity structure is analogous to object quantification. To demonstrate this, I have adapted a standard approach to object quantification to perform quantification over events. I have also examined this notion when the quantification of both objects and events interacts.

Finally, I have demonstrated that the computational analysis is reasonable and meaningful by providing a simple semi-automatic visualisation of the semantic representations that are obtained from the language. This again exploits the object-eventuality analogy by mapping activity structure to objects in the visualisation.

In particular, the following has been done:

1. A comprehensive analysis of how we can view those kinds of activities that are either multiple instances of some core activity, or continuous execution of some activity.

2. An examination of those language constructs that are used to express such activity, including—

    - Temporal adverbials like '*twice*', '*every day*'
    - Temporal modifiers in prepositional phrases, like '*for ten minutes*', '*until it is ready*'.
    - Plural objects as patients or recipients of actions.
    - Complex sentences containing two clauses, such as '*Simmer the soup, stirring often, for ten minutes*'.
    - Sentences containing complex modifiers such as '*for ten minutes every afternoon*' and '*for ten minutes or until it softens*'.

3. The implementation of a computational system that

- Performs syntactic and semantic analysis on linguistic items of the sort described above, and produces semantic representations of their activity structure with respect to that described in 1.
- Performs scoping on these representations and identifies cases where scope interactions could result in more than one reading
- Takes scoped representations and produces simple visual representations that demonstrate the activity structure (1.) described by the language (2.)

9.2 Discussion

9.2.1 The object-eventuality analogy

When I began this work, I intended to examine the extent to which the object-eventuality analogy was useful, and to identify those places where it broke down. I certainly expected to find some examples in both categories.

However, to the extent that I have pushed it, I have found that the analogy holds and is valid. I had thought that one example of a weakness was the idea that multiple instances of an activity, occurring at the same time but not necessarily in the same place, did not have an object analogue. If multiple instances of an object were to exist, they *could* not exist in the same place. However, further thought on the matter clarified that multiple instances can exist at the same place, if they exist at different times; the example of my desk, which tomorrow may be a different desk, but in the same place.

I am not claiming to have tested the analogy fully; I only wish to note that it has served me well and I believe it is worth further exploitation.

9.2.2 An interesting example

The following example highlights what has been achieved by the work presented in this thesis.

The sentence

(9.1) Simmer the soup stirring it occasionally for fifteen minutes

has three syntactic readings if we omit punctuation[1]. For each reading, the modifiers—the temporal measure adverbials that come from ≪*occasionally*≫ and ≪*for fifteen minutes*≫—attach to different parts of the syntactic structure.

They can be summarised as follows for the three structures

1. We have a core sentence,'*simmer the soup stirring it*'—which means that we have a concurrent activity that consists of some ≪*simmering of soup*≫ and some ≪*stirring it*≫—that is modified by both modifiers.

2. We have a concurrent activity, consisting of ≪*simmering the soup*≫ at the same time as ≪*stirring it occasionally*≫, and this concurrent activity is modified by the ≪*for fifteen minutes*≫ modifier.

3. We again have a concurrent activity, but in this case we have a ≪*simmering the soup*≫ at the same time as ≪*stirring it occasionally for fifteen minutes*≫. That is, both '*occasionally*' and '*for fifteen minutes*' modify only '*stirring it*'.

Now for the first and third of these syntactic structures, because in these both modifiers are at the same structural level (in the first case they modify the concurrent activity, while in the third they modify the ≪*stirring*≫ sub-activity), each has two potential scopings. The second syntactic structure only has one scoping; each modifier modifies one of the concurrent activities. Thus in total there are five possible readings for this sentence, once a complete semantic analysis has been done. Capturing all of these possibilities is possible due to the 'temporal measure adverbials as quantifiers' approach argued for earlier. In addition, the use of feature structures has provided a degree of structuring that has allowed modifiers to be attached at different levels, thereby highlighting the significance of the scoping. Although it is possible to code up such structure in logic, the feature structure representation makes it clearly accessible.

The feature structures for the five scoped versions are shown below; these have been edited to demonstrate only structure.

[1] I note that with punctuation, some of these readings could be excluded. However, this approach allows all possibilities, and punctuation can later be used to exclude where appropriate.

```
''Simmering the soup and stirring it, for fifteen minutes,
  and doing this occasionally''
structure : ordered_set
extent :  occasional
content : structure : delimitedmass
          extent : 15 minutes
          content : structure : concurrent
                    substance : major : structure : happening
                                        substance : action : simmer
                                                    patient : soup
                                        type : continuous
                                 minor : structure : happening
                                        substance : action : stir
                                                     patient : referent
                                        type : discrete


''Simmering the soup and stirring it, doing this occasionally,
  for fifteen minutes''
structure : delimitedmass
extent : 15 minutes
content : structure : ordered_set
          extent : occasional
          content: structure : concurrent
                    substance : major : structure : happening
                                        substance : action : simmer
                                                    patient : soup

                                        type : continuous
                                minor : structure : happening
                                        substance : action : stir
                                                    patient : referent
                                        type : discrete

''Simmering the soup and stirring it occasionally,
  and doing this for fifteen minutes''
structure : delimitedmass
extent : 15 minutes
content : structure : concurrent
          substance : major : structure : happening
                              substance : action : simmer
                                          patient : soup
                              type : continuous
                      minor : structure : ordered_set
                              extent : occasional
                              content : structure : happening
                                        substance : action : stir
```

```
                                                patient : referent

''Simmering the soup, at the same time as stirring it
  for fifteen minutes occasionally''
structure : concurrent
substance : major : structure : happening
                    substance : action : simmer
                                patient : soup
                    type : continuous
            minor : structure : ordered_set
                    extent : occasional
                    content : structure : delimitedmass
                              extent : 15 minutes
                              content : structure : happening
                                        substance : action : stir
                                                    patient : referent

''Simmering the soup, at the same time as stirring
  it occasionally for fifteen minutes''
structure : concurrent
substance : major : structure : happening
                    substance : action : simmer
                                patient : soup
                    type : continuous
            minor : structure : delimitedmass
                    extent : 15 minutes
                    content : structure : ordered_set
                              extent : occasional
                              content : structure : happening
                                        substance : action : stir
                                                    patient : referent
```

The importance of this example is in demonstrating that we are now able to successfully predict all the meaningful readings of a sentence; this includes valid readings that were not initially apparent, and certainly not obvious.

9.3 Relationships—an alternative perspective

As an aside I present a discussion of some of the contrasts that are utilised in this thesis.

The thesis is concerned with a variety of different kinds of relationships—not always direct—that exist between different kinds of entities. Each of these relationships is significant in its contribution to the overall aspect of the thesis. This section simply clarifies the gamut of relationships that bears on the content of the thesis, illuminating a different perspective on the work presented. Table 9.1 is a summary of the nature of the relationships concerned. All of these have been developed in more detail in appropriate parts of the thesis.

Objects and eventualities

It has been noted that we can divide the space of things that we talk about into objects and eventualities. It has also been noted that objects and eventualities are analogous, in many ways, with respect to space and time. This important relationship between these kinds of entities is a strong theme throughout this thesis.

Instructions and activities

There is an important relationship between instructions and activities. It is reasonable to say that an instruction is a recipe, or template, for activity. Activity is something that exists in the world, and occurs at a particular time and place. An instruction is an unlocated description of activity, where the protagonist of the activity is the reader or agent understanding the instruction. This relationship is a fairly direct one; that is, an instruction together with a current execution context forms an activity.

Activity and sub-activity

The primary focus of this thesis has been the provision of an account of how we talk about activity structure. It is the case that an activity is always made of sub-activities, just as matter is made of sub-matter. There is a whole range of ways in which sub-activities can be grouped together to form larger activity bodies. In (9.2)—which is the same example shown in Section 1.7—we see a description of an activity that is made of two sub-activities—≪*simmering the soup for twenty minutes*≫ and ≪*stirring the soup often*≫. For each of these two sub-activities, there is further sub-activity. We may, for example, see the first as containing a sub-activity ≪*simmering the soup*≫ which occurs for twenty minutes.

(9.2) Simmer the soup for 20 minutes, stirring often.

We could conceptualise the entire activity described by (9.2) as having the activity structure that was depicted in Figure 1.3, in Section 1.7. It is the various relationships between activity and sub-activity, and between sub-activities themselves, that are of concern in this thesis.

Activity content and activity structure

We can look at this as distinguishing between what the activity contains and how it is packaged. This relates to mereology and topolgy. Taking the simple example in (9.3), this describes an instruction that is telling us two things: what to do—≪*stir the soup*≫—and the composition or structure of the activity—≪*do this thing every ten minutes*≫.

(9.3) Stir the soup every 10 minutes

It is not always the case that content and structure are so neatly aligned with the language. Nor is it always the case that content and structure are readily distinguishable. However, they are conceptually orthogonal things, and we try to distinguish those aspects of language—about eventualities—that are about structure, from those that are about content.

9.4 Further work

Perhaps one of the most valuable things to result from this work is the knowledge of how much more work there is to do!

There are three broad categories that would benefit from further work:

1. Extending the detailed analysis and implementation.

Entities		Relationship
objects	eventualities	analogous
instructions	activities	direct
activity	sub-activity	hierarchical
activity content	activity structure	orthogonal

Table 9.1: Summary of the relationships that have been investigated in the thesis.

2. Extending the analytical framework.

3. Examining how the work presented here fits in with the broader enterprise of understanding language in general.

It is the second of these that contains the most worthwhile and clearly identifiable issues; however, all of these areas are worth mentioning.

9.4.1 Extending the low-level analysis

The analysis that has been presented and its computational implementation have of necessity covered only a very small subset of instructional language. There are other language constructs that are used to express extended activity. I have only included a very small lexicon, and limited analysis of constructs that do not directly express activity structure. I do not analyse language like '*every half hour*' to any degree greater than to identify that some frequency of occurrence is meant. These are obvious places for extension. It is also feasible to provide a more sophisticated visualisation.

I do believe though that the basic analysis that has been provided is sound, and such extension would be modular.

9.4.2 Extending and completing the analytical framework

The thesis has been a drawing together of a number of concepts, and as a result of the broadness of this enterprise, there are many remaining paths left to follow. Again, using the basic analysis and framework provided, there is scope for profitable work in various areas.

I believe that there are two principal areas in which further work could usefully be pursued:

- More exploitation of the object-eventuality analogy, in particular in the processing of discourse and resolving anaphoric reference to events. This thesis has demonstrated that the approaches taken to quantification of objects apply equally to events; interesting research would investigate whether approaches to resolving anaphoric reference to objects also apply to resolving event anaphora.

  As an example, in the discourse presented in Section 8.9, the last instruction is '*Do this twice more.*' This is an anaphoric reference to an event; resolving these kinds of references in this framework would be useful. I believe that the analysis I have presented already gives some machinery that could be used in an approach to deciding on possible candidates for anaphora resolution. It would be interesting to see, for example, how the approach presented by Webber[54] for pronoun resolution, would merge with the already available structure in finding and then assessing candidates for event anaphora resolution.

- Deeper analysis of the relations between the temporal-measure adverbials, and deeper analysis of their meanings. In particular, I believe the recent work of Pratt & Brée[37] could usefully be combined with the work presented in this thesis. My thesis does not deal with the actual distinctions between say '*for half an hour*' and '*for three weeks*', whereas their work does; in addition, they have looked at the fact that '*for three days every week*' makes sense, while '*for a week every three days*' does not. I believe that this interacts with Moltman's[29] concept of intervals, and is particularly relevant to temporal measure adverbials.

  The analysis I have provided gives a framework and identifies in general the kinds of extended activity that correspond to plural and mass, and in general the kind of language we use to talk about them. Distinguishing between the details of these language constructs in a more refined way is something that Pratt & Brée have made progress with, although their work has not explicitly exploited the analogues with objects; nor do they distinguish explicitly between temporal measure adverbials, and other temporal adverbials.

I believe combining their detailed analysis with the overall approach here would result in broad as well as deep understanding of such language. In particular, incorporating such analysis into the scoping process, and including the relationship between intervals and sub-intervals as part of the well-formedness of a scoped representation, would be useful.

In addition to these particular extensions, it would also be appropriate to investigate the incorporation of more of the theory regarding execution context into the model. The model that I have presented has been developed with this in mind, and all the processing that has been implemented has been independent of context information. In addition, I have indicated clearly those points where context incorporation would be appropriate. In particular, after the neutral semantics has been obtained, it is appropriate to incorporate context information.

9.4.3 The language understanding process in general

In addition to the above, which relate to further developing the theory that has been presented, it would also be valuable to investigate how this work, which covers a subset of language and allows us to talk only about a subset of our experience, would mesh with work that is about other parts of language. One suggestion along these lines would be to include incorporating tense information, and other temporal information—perhaps for example attempting to integrate what I have done here with How's[18] approach to temporal ordering which, as was indicated in Chapter 2, takes a similar stance on the role of context in understanding, but does not deal with extended events.

Another valid investigation would be to take this work to a broader language subset than that of instructions, and to test its worth in understanding descriptive—rather than prescriptive—language.

The above pertain to a much bigger—and more vaguely defined—endeavour; however I believe that in our attempt to develop theories and systems for understanding natural language, the two-pronged approach of deep investigation combined with integration is vital.

Appendix A

The PATR-II grammar

A.1 The grammar rules

This appendix contains all the grammar rules that have been discussed in the thesis, as well as additional ones that allow syntactic analysis as required. The rules are presented here in the order in which they were introduced in Chapter 6, retaining the same numbering. Associated rules that were not discussed in the chapter are presented together with discussed rules, but are numbered with alphabetic characters.

```
% All sentences are instructions.
% This is the basis of the entire analysis, it is assumed that all
% sentences to be processed will be of the form of being a verb phrase

% R1 : instr --> vp
% Example : stir the soup for five minutes
    INSTR ---> [VP] @ [
        INSTR/cat <=> instr,
        VP/cat <=> vp,
        VP/level <=> phrasal,
        INSTR/level <=> instructional,
        INSTR/sem <=> VP/sem].

% -------------------------------------------------------------------
% Verb phrases.
% -------------

% the following  productions deal with simple verb phrases for
% intransitive, transitive and ditransitive verbs respectively
% in all three, the semantics of the sentence is the semantics of the
% verb; the destination and patient, where applicable, are obtained
% from the associated noun phrase and prepositional phrase
```

```
% R2 : vp --> vintrans
% Example : laugh
    VP ---> [VINTRANS] @ [
        VP/cat <=> vpunresolved,
        VINTRANS/cat <=> verb,
        VINTRANS/verbcat <=> intransitive,
        VP/verbclass <=>  VINTRANS/verbclass,
        VP/sem/structure <=> happening,
        VP/sem/index <=> e1,
        VP/sem/substance/action <=> VINTRANS/subst].

% R3 :vp --> vtrans np
% Example : stir the soup
    VP ---> [VTRANS,NP] @ [
        VP/cat <=> vpunresolved,
        VTRANS/cat <=> verb,
        VTRANS/verbcat <=> transitive,
        NP/cat <=> np,
        VP/verbclass <=>  VTRANS/verbclass,
        VP/sem/structure <=> happening,
        VP/sem/index <=> e2,
        VP/sem/substance/action  <=> VTRANS/subst,
        VP/sem/substance/patient <=> NP/sem].

% R4:vp --> vditrans np pp
% Example : put the carrot into the pot
    VP ---> [VDITRANS,NP,PP] @ [
        VP/cat <=> vpunresolved,
        VDITRANS/cat <=> verb,
        VDITRANS/verbcat <=> ditransitive,
        NP/cat <=> np,
        PP/cat <=> pp,
        PP/type <=> nonmeasure,
        VP/verbclass <=> VDITRANS/verbclass,
        VP/sem/structure <=> happening,
        VP/sem/index <=> e3,
        VP/sem/substance/recipient <=> PP/sem,
        VP/sem/substance/patient <=> NP/sem,
        VP/sem/substance/action  <=> VDITRANS/subst].

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%getting a vp --- this involves setting the happening type of an
% unresolved vp, and is done for verbs whose class is simple, masslike
% and iterable

%R5
    VP1 ---> [VP2] @ [
```

```
       VP1/cat <=> vp,
       VP2/cat <=> vpunresolved,
       VP2/verbclass <=> simple,
       VP1/sem <=> VP2/sem,
       VP1/sem/type <=> discrete,
       VP1/sem/modifiers <=> end].

%R6
   VP1 ---> [VP2] @ [
       VP1/cat <=> vp,
       VP2/cat <=> vpunresolved,
       VP2/verbclass <=> masslike,
       VP1/sem <=> VP2/sem,
       VP1/sem/type <=> continuous,
       VP1/sem/modifiers <=> end].

%R7
    VP1 ---> [VP2] @ [
       VP1/cat <=> vp,
       VP2/cat <=> vpunresolved,
       VP2/verbclass <=> iterable,
       VP1/sem <=> VP2/sem,
       VP1/sem/type <=> VP2/verbclass,
       VP1/sem/modifiers <=> end].

%-------------------------------------------------------------------
% combining a Verb Phrase with a Modifier
% ---------------------------------------

% If the modifier is one that is a temporal measure one
% R8 vp1 --> vp2 modifier

    VP1 ---> [VP2,MODIFIER] @ [
       VP1/cat <=> vp,
       VP2/cat <=> vp,
       MODIFIER/cat <=> modifier,
       MODIFIER/type <=> tempmeasure,
       VP1/sem/structure <=> VP2/sem/structure,
       VP1/sem/index <=> VP2/sem/index,
       VP1/sem/substance <=> VP2/sem/substance,
       VP1/sem/type <=> VP2/sem/type,
       VP1/sem/modifiers/first <=> MODIFIER/sem,
       VP1/sem/modifiers/rest <=> VP2/sem/modifiers].


% If the modifier is a prepositional phrase that is nonmeasure
% R8b vp1 --> vp2 pp
% this deals with sentences that say how, where, etc the action
```

```
% is to be done; we simply pass the semantics of the vp up.
% Example : cut the carrot with a knife
% real example : (<stir the soup> <on the table>)
   VP1 ---> [VP2,PP] @ [
        VP1/cat <=> vp,
        VP2/cat <=> vp,
        PP/cat <=> pp,
        PP/type <=> nonmeasure,
        VP1/sem <=> VP2/sem].

% When the modifier is an adverbial phrase that is nonmeasure
% R8c vp1 --> vp2 advp
% this deals with sentences that say how the action is to be done.
% these adverbials are not temporal measure adverbials
% Example : cut the carrot quickly
    VP1 ---> [VP2,ADVP] @ [
        VP1/cat <=> vp,
        VP2/cat <=> vp,
        ADVP/cat <=> advp,
        ADVP/type <=> nonmeasure,
        VP1/sem <=> VP2/sem,
        VP1/sem/manner <=> ADVP/sem].




% Modifiers
% ---------
% A modifier is a prepositional phrase that is a temporal measure one,
% such as "for ten minutes"

% R9 modifier --> pp

    MODIFIER ---> [PP] @ [
        MODIFIER/cat <=> modifier,
        PP/cat <=> pp,
        PP/type <=> tempmeasure,
        MODIFIER/type <=> tempmeasure,
        MODIFIER/sem <=> PP/sem].


% It is also an adverbial phrase that is a temporal measure one,
% such as "three times"

% R9b modifier --> advp

    MODIFIER ---> [ADVP] @ [
        MODIFIER/cat <=> modifier,
```

```
        ADVP/cat <=> advp,
        ADVP/type <=> tempmeasure,
        MODIFIER/type <=> tempmeasure,
        MODIFIER/sem <=> ADVP/sem].


% ------------------------------------------------------------------
%Producing delimited masses:
% Prepositional phrases
% ---------------------
% R10 pp --> prep np; for temporal measure PPs these result in
% delimited masses

    PP ---> [PREP,NP] @ [
        PP/cat <=> pp,
        PREP/cat <=> prep,
        NP/cat <=> np,
        PREP/type <=> measure,
        NP/type <=> time,
        PP/sem/structure <=> delimitedmass,
        PP/sem/index <=> e5,
        PP/sem/extent <=> NP/sem,
        PP/type <=> tempmeasure,
        PP/sem/extent/extent_type <=> PREP/sem
         ].

% Complements
% -----------
%
% this is to allow complements like ... "until it is cooked";
% here "until" is functioning as a complement.

% R11 sbar --> complement sentence
    SBAR ---> [COMPLEMENT,SENTENCE] @ [
        SBAR/cat <=> sbar,
        COMPLEMENT/cat <=> complement,
        COMPLEMENT/type <=> measure,
        SENTENCE/cat <=> sentence,
        SENTENCE/sem/vptype <=> state,
        SBAR/sem/structure <=> delimitedmass,
        SBAR/sem/index <=> e8,
        SBAR/sem/extent/structure <=> SENTENCE/sem/structure,
        SBAR/sem/extent/index <=> SENTENCE/sem/index,
        SBAR/sem/extent/substance <=> SENTENCE/sem/substance,
        SBAR/sem/extent/modifiers <=> SENTENCE/sem/modifiers,
        SBAR/type <=> tempmeasure,
        SBAR/sem/extent/extent_type <=> COMPLEMENT/sem].
```

```
%=====================================================================
% Sbars as modifiers
% it is also an sbar that is temporal measure,
% such as "until it is thick"
% R12 modifier --> sbar

    MODIFIER ---> [SBAR] @ [
        MODIFIER/cat <=> modifier,
        SBAR/cat <=> sbar,
        SBAR/type <=> tempmeasure,
        MODIFIER/type <=> tempmeasure,
        MODIFIER/sem <=> SBAR/sem].

%=====================================================================
% Adverbial phrases
% -----------------
% Temporal measure adverbial phrases result in ordered sets

% An adverbial like "twice" is an adverbial phrase.
% R13 advp --> adv
 ADVP ---> [ADV] @ [
        ADVP/cat <=> advp,
        ADV/cat <=> adv,
        ADVP/type <=> ADV/type,
        ADVP/sem/structure <=> ordered_set,
        ADVP/sem/index <=> e12,
        ADVP/sem/extent/det <=> ADV/sem,
        ADVP/sem/extent/var <=> x,
        ADVP/sem/extent/restr <=> ADV/advsort,
        ADV/level <=> lexical,
        ADVP/level <=> phrasal].


% This caters for 'every ten minutes' being an adverbial phrase.
% R14 advp -->  det np
 ADVP ---> [DET,NP] @ [
        ADVP/cat <=> advp,
        DET/cat <=> det,
        NP/cat <=> np,
        DET/type <=> measure,
        NP/type <=> time,
        ADVP/type <=> tempmeasure,
        DET/class <=> quant,
        ADVP/sem/structure <=> ordered_set,
        ADVP/sem/index <=> e12,
        ADVP/sem/extent/det <=> DET/sem,
        ADVP/sem/extent/var <=> y,
        ADVP/sem/extent/restr <=> NP/sem,
```

```
       NP/level <=> phrasal,
       ADVP/level <=> phrasal].


% This rule identifies "five times" as an adverbial phrase
% R15 advp --> det noun
 ADVP ---> [DET,NOUN] @ [
       ADVP/cat <=> advp,
       DET/cat <=> det,
       DET/class <=> quant,
       NOUN/cat <=> noun,
       NOUN/sem <=> instances,
       ADVP/type <=> NOUN/type,
       ADVP/sem/structure <=> ordered_set,
       ADVP/sem/index <=> e12,
       ADVP/sem/extent/det <=> DET/sem,
       ADVP/sem/extent/var <=> z,
       ADVP/sem/extent/restr <=> NOUN/sem,
       ADVP/level <=> phrasal].

%---------------------------------------------------------------
% Adjuncts
% --------
% Dealing with adjuncts when modifiers
% A rule to deal with a special case adjunct where the two adjoining
% cat's are different ("for 5 minutes or until the mussels open")
% I could also have a rule for "until ... or for ...", but this
% seems unnecessary
% R16 modifier ---> modifier adjunct modifier
    MODIFIER1 ---> [MODIFIER2,ADJUNCT,MODIFIER3] @ [
        MODIFIER1/cat <=> modifier,
        MODIFIER3/cat <=> modifier,
        MODIFIER2/cat <=> modifier,
        ADJUNCT/cat <=> adjunct,
        ADJUNCT/sem <=> conjunct,
        MODIFIER1/sem/structure <=> conjunctive,
        MODIFIER1/sem/index <=> e13,
        MODIFIER1/sem/extent/extent1 <=> MODIFIER2/sem,
        MODIFIER1/sem/extent/extent2 <=> MODIFIER3/sem,
        MODIFIER1/type <=> tempmeasure].
% -----------------------------------------------------------
% This rule deals with constructs like "bread and butter";
% the conjuction of two lexical items from the same category
% R16b cat ---> cat adjunct cat
    CAT1 ---> [CAT2,ADJUNCT,CAT3] @ [
        CAT2/cat <=> CAT3/cat,
        CAT2/cat <=> det,
        CAT1/cat <=> CAT2/cat,
```

```
        CAT2/level <=> CAT3/level,
        CAT2/level <=> lexical,
        CAT1/level <=> phrasal,
        ADJUNCT/cat <=> adjunct,
        CAT1/class <=> CAT2/class,
        CAT1/sem/part1 <=> CAT2/sem,
        CAT1/sem/part2 <=> CAT3/sem
                         ].
% ...............
% Adjuncts of verbs and verb phrases
%
% R16c verb --> verb  adjunct verb
% (deals with 'and' & 'or', when Y and Z are verbs)
% Example: "top and tail the gooseberries"

    VERB1 ---> [VERB2,ADJUNCT,VERB3] @ [
        VERB2/cat <=> VERB3/cat,
        VERB1/cat <=> VERB2/cat,
        VERB2/cat <=> verb,
        ADJUNCT/cat <=> adjunct,
        VERB2/verbcat <=> VERB3/verbcat,
        VERB2/verbclass <=> VERB3/verbclass,
        VERB2/level <=> VERB3/level,
        VERB2/level <=> lexical,
        VERB1/level <=> VERB2/level,
        VERB1/verbcat <=> VERB2/verbcat,
        VERB1/verbclass <=> VERB2/verbclass,
        VERB1/subst/structure <=> compound,
        VERB1/subst/act1 <=> VERB2/subst,
        VERB1/subst/act2 <=> VERB3/subst].

% R16d vp --> vp adjunct vp (deals with 'and' & 'or', when Y and Z are
%     verb phrases)
%     Example: "peel the potatoes and stir the soup"

    VP1 ---> [VP2,ADJUNCT,VP3] @ [
        VP2/cat <=> VP3/cat,
        VP2/cat <=> vp,
        VP1/cat <=> VP2/cat,
        VP2/level <=> VP3/level,
        VP2/level <=> phrasal,
        VP1/level <=> phrasal,
        ADJUNCT/cat <=> adjunct,
        VP1/sem/structure <=> compound,
        VP1/sem/index <=> e3,
        VP1/sem/substance/participant1 <=> VP2/sem,
        VP1/sem/substance/participant2 <=> VP3/sem,
        VP1/sem/type <=> nil,
```

```
        VP1/sem/modifiers <=> end].
%-----------------------------------------------------------------
% Gerundive verb phrases
% -------------
% Forming verb phrases that might contribute to gerundive phrases,
% such as "stirring it" in "simmer the soup, stirring it, for an hour"
% gerundive vps (transitive)
% Ex ... stirring it
% R17 gerundiveVerbPhrase ---> gerundiveVerb NounPhrase
    GERUNDVP ---> [VGER,NP] @ [
        GERUNDVP/cat <=> gerundvp,
        VGER/cat <=> vger,
        NP/cat <=> np,
        VGER/verbcat <=> transitive,
        GERUNDVP/sem/structure <=> happening,
        GERUNDVP/sem/index <=> e7,
        GERUNDVP/sem/substance/action  <=> VGER/sem,
        GERUNDVP/sem/substance/patient <=> NP/sem,
        GERUNDVP/sem/type <=> VGER/verbclass].


% gerundive vps with intransitive verbs
%  Ex... opening
% R17b
    GERUNDVP ---> [VGER] @ [
        GERUNDVP/cat <=> gerundvp,
        VGER/cat <=> vger,
        VGER/verbcat <=> intransitive,
        GERUNDVP/sem/structure <=> happening,
        GERUNDVP/sem/index <=> e9,
        GERUNDVP/sem/substance/action  <=> VGER/sem,
        GERUNDVP/sem/type <=> VGER/verbclass].

%
% combining a gerundive Verb Phrase with a Modifier
% % If the modifier is one that is a temporal measure one
% R18 gerundvp --> gerundvp modifier

    GERUNDVP1 ---> [GERUNDVP2,MODIFIER] @ [
        GERUNDVP1/cat <=> gerundvp,
        GERUNDVP2/cat <=> gerundvp,
        MODIFIER/cat <=> modifier,
        MODIFIER/type <=> tempmeasure,
        GERUNDVP1/sem/structure <=> GERUNDVP2/sem/structure,
        GERUNDVP1/sem/index <=> GERUNDVP2/sem/index,
        GERUNDVP1/sem/substance <=> GERUNDVP2/sem/substance,
        GERUNDVP1/sem/type <=> GERUNDVP2/sem/type,
        GERUNDVP1/sem/modifiers/first <=> MODIFIER/sem,
```

```
        GERUNDVP1/sem/modifiers/rest <=> GERUNDVP2/sem/modifiers].

% Here we deal with how verb phrases  may be combined with gerundvp's
% R 19 vp --> vp gerundvp
% like <simmer the soup for fifteen minutes> <stirring occasionally>

    VP1 ---> [VP2,GERUNDVP] @ [
        VP1/cat <=> vp,
        VP2/cat <=> vp,
        GERUNDVP/cat <=> gerundvp,
        VP1/sem/structure <=> concurrent,
        VP1/sem/index <=> e6,
        VP1/sem/substance/major <=> VP2/sem,
        VP1/sem/substance/major/modifiers <=> VP2/sem/modifiers,
        VP1/sem/substance/minor <=> GERUNDVP/sem,
        VP1/sem/substance/minor/modifiers <=> GERUNDVP/sem/modifiers,
        VP1/sem/type <=> nil,
        VP1/sem/modifiers <=> end].

%..............................................................
% Noun phrases
%.................................................................
% the following rules pertain to the way a determiner and a nominal
% combine, to form a noun phrase. Depending on the kind of determiner
% (definite, indefinite) and the kind of nominal (plural, mass,
% singular). The resulting np may be quantified or not.
% In the implementation here, only "the" combined with a singular
% nominal results in a 'determined' np; all other combinations
% result in a 'quantified' np.

% R20 np --> det nom
% Allows for indefinite determiners;  all treated as quantifiers
% Example : a cake, some cake, some cakes, some soup
    NP ---> [DET,NOM] @ [
        NP/cat <=> np,
        DET/cat <=> det,
        DET/class <=> quant,
        DET/agr <=> NOM/agr,
        NP/sem/term/det <=> DET/sem,
        NP/sem/term/var <=> w,
        NP/sem/term/restr <=> NOM/sem,
        NP/agr <=> NOM/agr,
        NOM/cat <=> nom,
        NP/type <=> NOM/type,
        NP/level <=> phrasal].

% R21 np --> det nom
% Allows for definite determiners; only "the" singularobj is seen as
```

```
% definite
% Example : the cake
    NP ---> [DET,NOM] @ [
        NP/cat <=> np,
        DET/cat <=> det,
        DET/class <=> definite,
        DET/agr <=> NOM/agr,
        NOM/agr <=> sing,
        NOM/cat <=> nom,
        NP/agr <=> NOM/agr,
        NP/sem <=> NOM/sem,
        NP/level <=> phrasal].

% R22 np --> det nom
% Allows for definite determiners with a mass obj;
% Example : the soup
    NP ---> [DET,NOM] @ [
        NP/cat <=> np,
        DET/cat <=> det,
        DET/class <=> definite,
        DET/agr <=> NOM/agr,
        NOM/agr <=> mass,
        NOM/cat <=> nom,
        NP/agr <=> NOM/agr,
        NP/sem <=> NOM/sem,
        NP/level <=> phrasal].

% R23 np --> det nom
% Allows for definite determiners with a pl object -- this is a qterm
% Example : the gooseberries
    NP ---> [DET,NOM] @ [
        NP/cat <=> np,
        DET/cat <=> det,
        DET/class <=> definite,
        DET/agr <=> NOM/agr,
        NOM/agr <=> pl,
        NOM/cat <=> nom,
        NP/agr <=> NOM/agr,
        NP/sem/cardinality <=> numberof,
        NP/sem/term/det <=> DET/sem,
        NP/sem/term/var <=> v,
        NP/sem/term/restr <=> NOM/sem,
        NP/level <=> phrasal].
%--------------------------------------------------------------------
% R24(b) nom --> prep nom[+of]
% This does things like "of jam" or "of butter",
% and passes the semantics up
% Example : the soup
```

```
    NOM1 ---> [PREP,NOM2] @ [
        NOM1/cat <=> nom,
        PREP/cat <=> prep,
        NOM2/cat <=> nom,
        PREP/sem <=> of,
        NOM1/preptype <=> consists,
        NOM1/type <=> nonmeasure,
        NOM1/sem <=> NOM2/sem,
        NOM1/agr <=> NOM2/agr].

% R25 np --> np nom[+of]
% Allows for things like "a spoonful of jam"
% Example : the soup
    NP1 ---> [NP2,NOM] @ [
        NP1/cat <=> np,
        NP2/cat <=> np,
        NOM/cat <=> nom,
        NOM/preptype <=> consists,
        NP1/agr <=> NP2/agr,
        NP2/type <=> measure,
        NP1/sem/contains <=> NOM/sem,
        NP1/sem <=> NP2/sem,
        NP1/type <=> nonmeasure].
%==================================================================
% Pre- and post-modifiers of the adverbial phrase

% This is to allow constructs like "another five times",
% R26 advp -->  premodifier advp
 ADVP1 ---> [PREMODIFIER,ADVP2] @ [
        ADVP1/cat <=> advp,
        ADVP2/cat <=> advp,
        PREMODIFIER/cat <=> premod,
        ADVP1/type <=> ADVP2/type,
        ADVP1/level <=> ADVP2/level,
        ADVP1/sem <=> ADVP2/sem].

% This  allows things like "twice more", and "twice again"
% R27 advp -->  advp postmodifier
 ADVP1 ---> [ADVP2,POSTMODIFIER] @ [
        ADVP1/cat <=> advp,
        ADVP2/cat <=> advp,
        POSTMODIFIER/cat <=> postmod,
        ADVP1/type <=> ADVP2/type,
        ADVP1/level <=> ADVP2/level,
        ADVP1/sem <=> ADVP2/sem].
% =================================================================
% non-temporalmeasure modifiers
% R28 pp --> prep np; for non-measure PPs
```

```
    PP ---> [PREP,NP] @ [
        PP/cat <=> pp,
        PREP/cat <=> prep,
        NP/cat <=> np,
        PREP/type <=> nonmeasure,
        PREP/sem <=> place,
        PP/type <=> nonmeasure,
        PP/agr <=> NP/agr,
        PP/sem/head <=> PREP/sem,
        PP/sem/rest <=> NP/sem].



%--------------------------------------------------------------------
% R29 sentence --> np vp
% This is for the conventional sentence structure, that is found
% after a complement
 SENTENCE ---> [NP,VP] @ [
        SENTENCE/cat <=> sentence,
        NP/cat <=> np,
        VP/cat <=> vp,
        VP/sem/vptype <=> state,
        SENTENCE/sem <=> VP/sem,
        SENTENCE/sem/substance/patient <=> NP/sem].
%====================================================================
% Other noun-related rules
% A nominal on its own is an np
% R30 np --> nom

    NP ---> [NOM] @ [
        NP/cat <=> np,
        NOM/cat <=> nom,
        NP/level <=> phrasal,
        NOM/level <=> intermediate,
        NP/sem <=> NOM/sem,
        NP/agr <=> NOM/agr,
        NP/type <=> NOM/type].

% Nominals may be nouns or pronouns --
% R31 nom --> noun

 NOM ---> [NOUN] @ [
        NOM/cat <=> nom,
        NOM/level <=> intermediate,
        NOUN/level <=> lexical,
        NOUN/cat <=> noun,
        NOM/agr <=> NOUN/agr,
```

```
        NOM/type <=> NOUN/type,
        NOM/sem <=> NOUN/sem].

% R32 nom --> pronoun

 NOM ---> [PRONOUN] @ [
        NOM/cat <=> nom,
        PRONOUN/cat <=> pronoun,
        NOM/level <=> intermediate,
        PRONOUN/level <=> lexical,
        NOM/agr <=> PRONOUN/agr,
        NOM/sem <=> PRONOUN/sem].

% Other noun-related rules, to achieve the coverage required
% R33 nom --> ap noun
% allows for adjectival phrases
% Example : five minutes;  big spoon
 NOM ---> [AP,NOUN] @ [
        NOM/cat <=> nom,
        AP/cat <=> ap,
        NOUN/cat <=> noun,
        NOM/level <=> intermediate,
        NOM/sem/mod <=> AP/sem,
        NOM/sem/subst <=> NOUN/sem,
        NOM/agr <=> NOUN/agr].

% R34 np --> predet np
% Allows for predeterminers, such as both, half, all
% Example : all the time
    NP1 ---> [PREDET,NP2] @ [
        NP1/cat <=> np,
        PREDET/cat <=> predet,
        NP2/cat <=> np,
        NP1/sem <=> NP2/sem,
        NP1/level <=> NP2/level,
        NP1/agr <=> NP2/agr,
        NP1/mod <=> PREDET/sem

% Pre and post modifiers of the noun phrase...
% R35 np -->  np postmodifier
    NP1 ---> [NP2,POSTMODIFIER] @ [
        NP1/cat <=> np,
        NP2/cat <=> np,
        POSTMODIFIER/cat <=> postmod,
        NP1/type <=> NP2/type,
        NP1/level <=> NP2/level,
        NP1/sem <=> NP2/sem].
```

```
% R36 np -->  premodifier np
    NP1 ---> [PREMODIFIER,NP2] @ [
        NP1/cat <=> np,
        NP2/cat <=> np,
        PREMODIFIER/cat <=> premod,
        NP1/type <=> NP2/type,
        NP1/level <=> NP2/level,
        NP1/sem <=> NP2/sem].


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This is for those vgps that describe states
% using gerundive verbs (transitive)
% Ex ... "is helping" in "Jo is helping Lee"
% R37
    VP ---> [VPSTATE,NP] @ [
        VP/cat <=> vp,
        VPSTATE/cat <=> vpstate,
        NP/cat <=> np,
        VPSTATE/sem/vptype <=> state,
        VPSTATE/verbcat <=> transitive,
        VP/verbclass <=> VPSTATE/verbclass,
        VP/sem/structure <=> state,
        VP/sem/index <=> e9,
        VP/sem/substance  <=> VPSTATE/sem,
        VP/sem/substance/patient <=> NP/sem,
        VP/sem/modifiers <=> end].

% gerundive verb groups with intransitive verbs
%  Ex... has dissolved
% R38
    VP ---> [VPSTATE] @ [
        VP/cat <=> vp,
        VPSTATE/cat <=> vpstate,
        VPSTATE/verbcat <=> intransitive,
        VPSTATE/sem/vptype <=> state,
        VP/sem/structure <=> state,
        VP/sem/index <=> e9,
        VP/sem/substance  <=> VPSTATE/sem,
        VP/sem/modifiers <=> end].


%--------------------------------------------------------------------
% These productions also with verb phrases which have different forms
% or have adjuncts of some sort

% Example : is boiling, as an AUX
```

```
% R39 VP(state) ---> auxillary gerundiveVerb
    VPSTATE ---> [AUX,VGER] @ [
        VPSTATE/cat <=> vpstate,
        AUX/cat <=> aux,
        VGER/cat <=> vger,
        VPSTATE/verbclass <=> VGER/verbclass,
        VPSTATE/verbcat <=> VGER/verbcat,
        VPSTATE/subst <=> VGER/subst,
        VPSTATE/sem/condition <=> VGER/sem,
        VPSTATE/tense <=> AUX/tense,
        VPSTATE/sem/vptype <=> state].

% Example : has dissolved
% R40 VP(state) ---> auxillary infiniteVerb
    VPSTATE ---> [AUX,VGER] @ [
        VPSTATE/cat <=> vpstate,
        AUX/cat <=> aux,
        VGER/cat <=> vinf,
        VPSTATE/verbclass <=> VGER/verbclass,
        VPSTATE/verbcat <=> VGER/verbcat,
        VPSTATE/subst <=> VGER/subst,
        VPSTATE/sem/condition <=> VGER/sem,
        VPSTATE/tense <=> AUX/tense,
        VPSTATE/sem/vptype <=> state].
```

A.2 The Lexicon

This contains a representative sample of lexical items. Each item will have a constraint of the form `X/level <=> lexical`; these are omitted from all but the first item shown:

```
% intransitive verbs
dict(open, X@[X/cat  <=> verb, X/verbcat <=> intransitive,
               X/verbclass <=> simple, X/subst <=> open]).


% transitive verbs
dict(stir, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> iterable, X/subst <=> stir]).
dict(simmer, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> masslike, X/subst <=> simmer]).
dict(cook, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> masslike, X/subst <=> cook]).
dict(steam, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> masslike, X/subst <=> steam]).
```

```
dict(bake, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> masslike, X/subst <=> bake]).
dict(cut, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> simple, X/subst <=> cut]).
dict(put, X@[X/cat  <=> verb, X/verbcat <=> ditransitive,
             X/verbclass <=> simple,  X/subst <=> put]).
dict(melt, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> masslike, X/subst <=> melt]).
dict(heat, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> masslike, X/subst <=> heat]).
dict(skim, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> simple, X/subst <=> skim]).
dict(baste, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> simple, X/subst <=> baste]).
dict(roast, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> masslike, X/subst <=> roast]).
dict(roll, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> simple, X/subst <=> roll]).
dict(top_and_tail, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> simple, X/subst <=> top]).
dict(top, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> simple, X/subst <=> top]).
dict(tail, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> simple, X/subst <=> tail]).
dict(leave, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> masslike, X/subst <=> leave]).
dict(peel, X@[X/cat  <=> verb, X/verbcat <=> transitive,
               X/verbclass <=> iterable, X/subst <=> peel]).
dict(beat, X@[X/cat  <=> verb, X/verbcat <=> transitive,
               X/verbclass <=> iterable, X/subst <=> beat]).
dict(discard, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> simple,
               X/subst <=> discard]).
% 'open' can be transitive
dict(open, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> simple, X/subst <=> open,
              X/sem <=> open_something]).
% do and repeat are 'meta' verbs ...
dict(do, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> simple, X/subst/act <=> do,
              X/subst/sem <=> activity]).
dict(repeat, X@[X/cat  <=> verb, X/verbcat <=> transitive,
              X/verbclass <=> simple, X/subst/act <=> repeat,
              X/subst/sem <=> activity]).
% intransitve gerundive verbs
dict(foaming,  X@[X/cat <=> vger, X/verbcat <=> intransitive,
               X/verbclass <=> iterable, X/subst <=> foam,
               X/sem <=> be_foaming]).
```

```
dict(boiling,  X@[X/cat <=> vger, X/verbcat <=> intransitive,
               X/verbclass <=> masslike, X/subst <=> boil.
               X/sem <=> be_boiling]).
dict(tender,  X@[X/cat <=> vger, X/verbcat <=> intransitive,
               X/verbclass <=> iterable, X/subst <=> tender,
               X/sem <=> be_tender]).
% intransitive infinitival verbs
dict(dissolved,  X@[X/cat <=> vinf,  X/verbcat <=> intransitive,
               X/verbclass <=> iterable, X/subst <=> dissolve,
               X/tense <=> past, X/sem <=> dissolved]).
dict(thick,  X@[X/cat <=> vinf,  X/verbcat <=> intransitive,
               X/verbclass <=> masslike, X/subst <=> thicken,
               X/tense <=> present, X/sem <=> thick]).
% transitive gerundive verbs
dict(boiling,  X@[X/cat <=> vger, X/verbcat <=> transitive,
               X/verbclass <=> iterable, X/subst <=> boil,
               X/sem <=> boil]).
dict(basting,  X@[X/cat <=> vger,X/verbcat <=> transitive,
                  X/verbclass <=> iterable, X/subst <=> baste,
                  X/sem <=> baste]).
dict(stirring, X@[X/cat <=> vger, X/verbcat <=> transitive,
               X/verbclass <=> iterable, X/subst <=> stir,
               X/sem <=> stir]).

% adjuncts
dict(and, X@[X/cat <=> adjunct, X/sem <=> both]).
dict(or, X@[X/cat <=> adjunct, X/sem <=> conjunct]).

% prepositions
dict(for, X@[X/cat <=> prep, X/sem <=> time, X/type <=> measure,
            X/level <=> lexical]).
dict(on, X@[X/cat <=> prep, X/sem <=> place, X/type <=> nonmeasure,
            X/level <=> lexical]).
dict(in, X@[X/cat <=> prep, X/sem <=> place,  X/type <=> nonmeasure,
            X/level <=> lexical]).
dict(of, X@[X/cat <=> prep, X/sem <=> of, X/type <=> nonmeasure,
             X/level <=> lexical]).
dict(into, X@[X/cat <=> prep, X/sem <=> place,  X/type <=> nonmeasure,
            X/level <=> lexical]).
dict(until, X@[X/cat <=> prep, X/sem <=> condition,
               X/type <=> measure, X/level <=> lexical]).

% complements
dict(until, X@[X/cat <=> complement,X/sem <=> condition,
               X/type <=> measure]).

% pronouns
dict(it, X@[X/cat <=> pronoun, X/agr <=> sing, X/sem <=> referent]).
```

```
dict(this, X@[X/cat <=> pronoun, X/agr <=> sing, X/sem <=> referent]).
dict(they, X@[X/cat <=> pronoun, X/agr <=> pl, X/sem <=> referent]).

%nouns
dict(butter, X@[X/cat <=> noun, X/agr <=> mass, X/type <=> nonmeasure,
                X/sem <=> butter]).
dict(foam, X@[X/cat <=> noun, X/agr <=> mass, X/type <=> nonmeasure,
                X/sem <=> foam]).
dict(joint, X@[X/cat <=> noun, X/agr <=> sing,X/type <=> nonmeasure,
                X/sem <=> joint]).
dict(soup, X@[X/cat <=> noun,X/agr <=> mass, X/type <=> nonmeasure,
                X/sem <=> soup]).
dict(sauce, X@[X/cat <=> noun,X/agr <=> mass, X/type <=> nonmeasure,
                X/sem <=> sauce]).
dict(spoonful, X@[X/cat <=> noun, X/agr <=> sing,X/type <=> measure,
                X/sem <=> spoonful]).
dict(jam, X@[X/cat <=> noun,X/agr <=> mass,  X/type <=> nonmeasure,
                 X/sem <=> jam]).
dict(sugar, X@[X/cat <=> noun,X/agr <=> mass,  X/type <=> nonmeasure,
                 X/agr <=> mass, X/sem <=> sugar]).
dict(gooseberries, X@[X/cat <=> noun, X/type <=> nonmeasure,
                      X/agr <=> pl, X/sem <=> gooseberries]).
dict(mussels, X@[X/cat <=> noun, X/type <=> nonmeasure,
                 X/agr <=> pl, X/sem <=> mussels]).
dict(potato, X@[X/cat <=> noun,X/type <=> nonmeasure,
                   X/agr <=> sing, X/sem <=> potato]).
dict(potatoes, X@[X/cat <=> noun,X/type <=> nonmeasure,
                   X/agr <=> pl, X/sem <=> potatoes]).
dict(tart, X@[X/cat <=> noun, X/agr <=> sing,X/type <=> nonmeasure,
                  X/sem <=> tart]).
dict(cake, X@[X/cat <=> noun, X/type <=> nonmeasure,
              X/sem <=> cake, X/agr <=> sing]).
dict(cakes, X@[X/cat <=> noun,X/type <=> nonmeasure, X/sem <=> cake,
              X/agr <=> pl]).
dict(mixture, X@[X/cat <=> noun,X/agr <=> mass, X/type <=> nonmeasure,
                X/sem <=> mixture]).
dict(dough, X@[X/cat <=> noun, X/agr <=> mass, X/type <=> nonmeasure,
                  X/sem <=> dough]).
dict(oblong, X@[X/cat <=> noun, X/agr <=> sing, X/type <=> nonmeasure,
                        X/sem <=> oblong]).

% nouns with temporal properties
dict(minute, X@[X/cat <=> noun, X/agr <=> sing, X/type <=> time,
                  X/sem <=> minutes]).
dict(hour, X@[X/cat <=> noun, X/agr <=> sing ,X/type <=> time,
                  X/sem <=> hours]).
dict(minutes, X@[X/cat <=> noun, X/agr <=> pl, X/type <=> time,
                  X/sem <=> minutes]).
```

```
dict(hours, X@[X/cat <=> noun, X/agr <=> pl, X/type <=> time,
                X/sem <=> hours]).

% determiners
dict(the, X@[X/cat <=> det, X/class <=> definite, X/sem <=> all ]).
% quantifiers
dict(a, X@[X/cat <=> det, X/class <=> quant, X/type <=> measure,
                      X/agr <=> sing, X/sem <=> exists]).
dict(an, X@[X/cat <=> det, X/class <=> quant, X/type <=> measure,
                     X/agr <=> sing, X/sem <=> exists]).
dict(some, X@[X/cat <=> det, X/class <=> quant, X/type <=> measure,
                     X/sem <=> some]).
dict(a_few, X@[X/cat <=> det, X/class <=> quant, X/type <=> measure,
                      X/agr <=> pl, X/sem <=> three]).
dict(every, X@[X/cat <=> det, X/class <=> quant, X/type <=> measure,
                     X/agr <=> sing, X/sem <=> every]).
dict(many,X@[X/cat <=> det, X/class <=> quant, X/type <=> measure,
                     X/agr <=> pl, X/sem <=> many]).
dict(several,X@[X/cat <=> det, X/class <=> quant,X/type <=> measure,
                      X/agr <=> pl, X/sem <=> many]).
% dets and quantifiers
dict(one,X@[X/cat <=> det,  X/class <=> quant, X/type <=> cardinal,
                      X/agr <=> sing, X/sem <=> 1, X/level <=> lexical]).
dict(two,X@[X/cat <=> det,  X/class <=> quant,X/type <=> cardinal,
                       X/agr <=> pl, X/sem <=> 2, X/level <=> lexical]).
dict(three,X@[X/cat <=> det,  X/class <=> quant, X/type <=> cardinal,
                      X/agr <=> pl, X/sem <=> 3, X/level <=> lexical]).
dict(five,X@[X/cat <=> det,  X/class <=> quant,X/type <=> cardinal,
                       X/agr <=> pl, X/sem <=> 5, X/level <=> lexical]).
dict(ten,X@[X/cat <=> det,  X/class <=> quant, X/type <=> cardinal,
                      X/agr <=> pl, X/sem <=> 10, X/level <=> lexical]).
dict(fifteen,X@[X/cat <=> det,  X/class <=> quant,X/type <=> cardinal,
                       X/agr <=> pl, X/sem <=> 15, X/level <=> lexical]).

% Adverbials
dict(often, X@[X/cat <=> adv, X/advsort <=> event,
    X/type <=> tempmeasure, X/sem <=> often,  X/level <=> lexical]).
dict(frequently, X@[X/cat <=> adv, X/advsort <=> event,
    X/type <=> tempmeasure, X/sem <=> often, X/level <=> lexical]).
dict(occasionally, X@[X/cat <=> adv, X/advsort <=> event,
    X/type <=> tempmeasure, X/sem <=> occasional, X/level <=> lexical]).
dict(again, X@[X/cat <=> adv, X/type <=> tempmeasure, X/sem <=> again,
               X/advsort <=> event, X/level <=> lexical]).
dict(quickly, X@[X/cat <=> adv, X/advsort <=> event,
     X/type <=> nonmeasure, X/sem <=> speed, X/level <=> lexical]).
dict(continuously, X@[X/cat <=> adv, X/advsort <=> event,
    X/type <=> tempmeasure, X/sem <=> continuous, X/level <=> lexical]).
```

```
% These are special forms of one time and two times
dict(once, X@[X/cat <=> adv, X/advsort <=> event,
    X/type <=> tempmeasure, X/sem <=> one, X/level <=> lexical]).
dict(twice, X@[X/cat <=> adv, X/advsort <=> event,
    X/type <=> tempmeasure, X/sem <=> two,  X/level <=> lexical]).

% predeterminers
dict(both,X@[X/cat <=> predet, X/level <=> lexical]).
dict(all,X@[X/cat <=> predet, X/level <=> lexical]).
dict(half,X@[X/cat <=> predet, X/sem <=> half, X/level <=> lexical]).
dict(each,X@[X/cat <=> predet, X/level <=> lexical]).

% modifiers of the (temporal) noun phrase and advphrase
dict(almost,X@[X/cat <=> premod,X/sem <=> approx, X/level <=> lexical]).
dict(nearly,X@[X/cat <=> premod,X/sem <=> approx, X/level <=> lexical]).
dict(about,X@[X/cat <=> premod, X/sem <=> approx, X/level <=> lexical]).
dict(more, X@[X/cat <=> postmod, X/sem <=> more, X/level <=> lexical]).
dict(another, X@[X/cat <=> premod,X/sem <=> more, X/level <=> lexical]).

% auxilliaries --- an indication of where tense might begin to be
% incorporated
dict(is, X@[X/cat <=> aux, X/tense <=> present, X/sem <=> state]).
dict(was, X@[X/cat <=> aux, X/tense <=> past, X/sem <=> state]).
dict(has, X@[X/cat <=> aux, X/tense <=> past]).

% this is to deal with "all the time ...",  etc.
% dict(times, X@[X/cat <=> noun, X/agr <=> pl, X/sem <=> time,
                 X/level <=> lexical]).
% dict(time, X@[X/cat <=> noun, X/agr <=> sing, X/sem <=> time,
                 X/level <=> lexical]).

% making constructs like "three times" be adverbial phrases
dict(times, X@[X/cat <=> noun, X/type <=> tempmeasure, X/agr <=> pl,
               X/sem <=> instances, X/level <=> lexical]).

% coping with sentences that have ellipsis
% I allow an empty noun phrase, for things like
% "steam (...) for two minutes or until the mussels open"
% or "steam the mussels for two minutes or until (...)  open"
dict(ellipsis, X@[X/cat <=> np, X/level <=> phrasal,X/sem <=> elided]).
```

A.3 Example sentences

The following demonstrate the range of sentences that the system is able to process.

```
[do,this,two,or,three,times].
[top,and,tail,the,gooseberries].
[put,a,spoonful,of,jam,into,each,tart].
[stir,the,soup,for,five,minutes].
[leave,it,for,five,minutes].
[beat,the,mixture,until,all,the,sugar,has,dissolved].
[peel,potatoes,for,a_few,hours].
[stir,the,soup,often,for,half,an,hour].
[stir,some,soup,every,half,hour].
[peel,many,potatoes,for,several,hours].
[baste,the,joint,every,fifteen,minutes].
[steam,the,mussels,for,two,minutes,or,until,they,open].
[roll,it,into,an,oblong,again,and,repeat,this,twice,more].
[simmer,the,soup,stirring,ellipsis,occasionally,for,fifteen,minutes].
[roast,the,meat,for,about,an,hour,basting,it,twice].
[[melt,the,butter,until,it,is,foaming],
 [skim,the,foam,and,discard,it], [heat,the,butter,and,skim,it,again],
 [do,this,twice,more]].
endoffile.
```

Appendix B

Scoping algorithms

B.1 The Hobbs and Shieber scoping algorithm

The core algorithm that Hobbs and Shieber[17] present can be summarised below. This looks for quantified terms of the form (Quant, Var, Restr), and assumes that the input is well-formed, with respect to the definition of `wff`.

```
gen(Form, ScopedForm) :-
    apply-terms(Form, ScopedForm).

apply-terms(Form, Form) :-
    \+ term(Form, Term), !.
apply-terms(Form, ScopedForm) :-
    term(Form, Term),
    apply(Term, Form, AppliedForm),
    apply-terms(AppliedForm, ScopedForm).

apply(term(Quant,Var,Restrict), Body,
        wff(Quant,[Var,Restrict,OutBody])) :-
    subst(Var, term(Quant,Var,Restrict), Body, OutBody).
```

The predicates `term` and `subst` are described as follows:

- "term(Form, Term)" is true whenever "Term" is a quantifier term (i.e., an expression of the form "term(...)") occurring inside the formula "Form". For a given "Form", "term/2" will return all the quantifier terms inside "Form" in turn, as alternative solutions.
- "subst(New, Old, OldForm, NewForm)" is true when "NewForm" is the same as "OldForm" except that any occurrences of "Old" inside it have been replaced by "New": this is a straightforward substitution predicate.

B.2 The feature structure implementation of the scoping algorithm

Here, all the code that performs the scoping of feature structures, as described in Chapter 7, is presented. First, the controlled implementation of **gen**, the main scoping predicate, is given. Next, the feature structure implementation of **apply-terms** and its associated predicates (**term**, **apply** and **subst**) are presented.

Feature structures are coded in Prolog as lists of feature-value pairs. A feature-value pair **feature=value** is coded as **feature:value**, and the lists themselves include a variable tail.

B.2.1 The main controlling predicate

```
% gen(Form, ScopedForm)
% ======================
%
%       Form            ==> a FS with in-place complex terms
%       ScopedForm      <== a full scoping of Form

gen(V, V) :-
        var(V), !.

gen([structure:H|Rest], Scoped) :- !,
        gen(Rest,ScopedRest),
        pull([structure:H|ScopedRest], Scoped).

gen(F:V, F:SV) :- !,
        gen(V, SV).

gen([H|T], [SH|ST]) :- !,
        gen(H,SH),
        gen(T,ST).

gen(X, X).

% pull(Form, ScopedForm)
% ================================
%
%       Form            ==> a FS with in-place complex terms
%       ScopedForm      <== a full or partial scoping of Form
%
%       Applies terms in Form.
```

```
pull(Form, ScopedForm) :-
        apply_terms(Form, ScopedForm).
```

B.2.2 Predicates that are used in apply-terms

```
% apply(Term, Form, NewForm)
% ==========================
%
%       Term            ==> a complex term
%       Form            ==> the FS to apply Term to
%       NewForm         <== Form with the quantifier wrapped around it

/* finding an event quantifier */
apply(first:Evalue,
                Body,
                NewFS) :-
        subst(dummy:dummy, first:Evalue, Body, OutBody),
        append(Evalue, [content : OutBody], NewFS).

/* finding an object quantifier */
apply(patient:[term : [det:Det,var:Var,restr:Restr]],
                Body,
                NewFS) :-
      subst(patient:Var, patient:[term : [det:Det,var:Var,restr:Restr]],
            Body, OutBody),
      append([structure : ordin_set, index : e99,
              extent :[Det:Var, restr:Restr]] ,
                            [content : OutBody], NewFS).

% apply_terms(Form, ScopedForm)
% ========================================
%
%       Form            ==> a FS with in-place complex terms
%       ScopedForm      <== a full or partial scoping of Form
%
%       Applies one or more terms to the Form alone (not to any embedded
%       forms).

apply_terms(Form, Form) :-
        not(term(Form, Term)), !.

apply_terms(Form, ScopedForm) :-
        term(Form, Term),
        apply(Term, Form, AppliedForm),
        apply_terms(AppliedForm, ScopedForm).
```

```
% subst(New, Old, OldForm, NewForm)
% =================================
%
%       New             ==> A pattern to substitute for Old
%       Old             ==> A pattern to be replaced by New
%       OldForm         ==> a FS with in-place complex terms
%       NewForm         <== OldForm with each occurrence of
%                                Old replaced by New

subst(_, _, V, V) :-
        var(V), !.

subst(New, Old, Old, New) :- !.

subst(New, Old, :(Quant, ArgList), :(Quant, NewArgList)) :- !,
        subst(New, Old, ArgList, NewArgList).

subst(B, A, [A|T], [B|NT]) :- !, subst(B, A, T, NT).

subst(B, A, [H|T], [NH|NT]) :- !, subst(B, A, H, NH),
                                   subst(B, A, T, NT).

subst(B, A, Form, Form).
```

B.2.3 Identifying terms in feature structures

```
% term(Form, Term)
% ================
%       Form            ==> a Feature structure or complex term
%       Term            <== a complex term contained in Form
%
%       Extracts a term from Form.

term(V, _) :-
        var(V), !,
        fail.

% If Form is a term, then it is a term.
term(first:Evalue,first:Evalue).
term(patient : [term:[Quant, Var, Restrict]],
        patient : [term:[Quant, Var, Restrict]]).

% If Form is an argument list, a term is
%                       a term of its head or of its tail.
term([H|T], Term) :-
        term(H, Term);
```

```
        term(T, Term).

% If Form is a FS, a term of form is a term of its feature list.
term(:(Head, ArgList), Term) :-
        term(ArgList, Term).
```

Appendix C

Visualisation

This appendix presents additional information about the visualisation implementation.

C.1 Implementation notes

Each PostScript array represents a discourse— which is an activity—and each element in the array is itself an activity—a sub-event at some level of the overall array event. There are two compulsory items for each element:

- the 'size' of the activity (this can be viewed as the time it will take)
- the 'substance' of the activity (what it is, its content or constituents)

Overall size and scale For each visualisation, an overall size and scale are chosen; these are parameters to PostScript. For a very complicated visualisation, a bigger size is more appropriate. The entire visualisation is scaled with respect to the chosen size. This approach is reasonable—when doing a diagram by hand, one does pick an overall size that can encompass the total activity. However, as with all computational implementations—and with all visualisations—practical limits constrain visualisations that are extreme, at either end of the scale, in size.

Size of components The sizing of happenings and of sub-activity within structured activity is done using the size of the activity; for each happening or activity, this is encoded in the first element of the PostScript array representing that activity. So, for element 4 of the array in (7.6c), reproduced below, which represents ≪*stirring the soup*≫, we are saying that the size of each ≪*stir the soup*≫ happening is 2. The size of ≪*simmering the soup for one hour*≫—array element 1—is 60, as is the size of ≪*simmering the soup for one hour, stirring frequently*≫.

(7.6c)
```
[[60  [concurrent [1 3]]]
 [60  [delimitedmass [2 (one hour) ]]
 [ 0  [happeningc [(simmer) (soup) ]]]
 [60  [ordered-set [12 4]]]
 [ 2  [happeningd [(stir) (elided) ]]]  ]]
```

These sizes are not generated automatically. First, there is not enough information in the discourse to decide, for example, that one ≪*stirring the soup*≫ constitutes 1/30th of the total activity. Secondly, some amount of modelling would be required to work such things out, and this is outwith the scope of the thesis.

Essentially, the approach taken for each array element is to decide on a size for it, by taking into account the sizes of the activities that it contains or that it contributes to. As I have noted, an appropriate way seems to be to relate this to time—and to say that a ≪*stirring the soup*≫ activity takes about 2 minutes, while ≪*simmering the soup for one hour*≫ takes 60 minutes. It is essential that the sizes between sub-activities tally; again, this is not done automatically in this implementation.

I note that in view of all of this, the sizings of components are only accurate when their size in time is specified in the language.

Shading of happenings I use shading to distinguish between basic happenings. Given the discourse in (C.1), '*Stir the soup*' would be shaded using, say, Pattern1 and '*Discard the foam*' would use Pattern2. '*Discard again*' would have a different pattern (say Pattern3). My system is not able to decide that Pattern2 and Pattern3 are representing the same happening sort; this distinction would need to come from the semantic analysis and would require some sophisticated world modelling.

(C.1) Stir the soup until it is boiling. Discard the foam. Let it cool and then bring to the boil. Discard again.

Happening key There is a 'key' for the primitive happenings, containing pattern (which is related to event number) and annotation (which is the language describing the action, such as '*stir the soup*' or '*bake a cake*').

Activity structure The only events that are actually being drawn are discrete and continuous happenings. These are the primitives. Everything else—all information about extended eventualities—is there by virtue of grouping and boundary information. That is, it is there from the way the graph is constructed.

C.2 The PostScript Implementation

This section describes the algorithms that form the basis of the visualisation implementation.

The main algorithm takes each element of the array in turn, and depending on the value of its event kind (the second entry in the the element), calls the routine that corresponds to that name. So, for example, in (7.6c) that was reproduced above, five routines would be called: concurrent, delimitedmass, happeningc, ordered-set and happeningd.

There is thus one routine for each kind of happening, and one routine for each kind of structuring. At the start, the overall size is input to the routine. Each routine calculates the relative sizes of its own components (for example, the ordered-set routine determines what size each of its elements would need to be), and the new relative centres (this is particularly important in concurrent structurings, as there will now be two centres), once it has drawn its own element. This information is passed back, and is then input as the new dimensions for the next component's drawing.

So, the size of components gets progressively smaller; the actual dimensions depend on the contents of the array element.

The routines for drawing happenings (happeningc and happeningd) draw outlines and also perform shading. The rules for drawing structurings (compound, ordinset, orderedset, conjunctive, concurrent) draw outlines, and determine the new relative sizes of their internal components.

Finally, the key is drawn; this involves again looking through the array for happenings, and drawing a key item for each of these.

C.3 Converting feature structures to PostScript arrays

This section contains the code that is used to convert a scoped feature structure into a form appropriate for visualisation.

```
/* There is one rule for each kind of structure */

/* If it is a continuous happening, make an array element with
   its annotation */
dostruct([structure:happening|Rest], NumIn, NumOut, ArrayIn,
            [[size,'/happeningc',[A,P]]|ArrayIn]) :-
        member(type:continuous, Rest), !,
        member(substance:Subs, Rest),
```

```
        member(action:Action, Subs),
        bracket(Action, A),
        member(patient:Patient, Subs),
        bracket(Patient, P),
        NumOut is NumIn + 1.

/* If it is a discrete happening, make an array element with
   its annotation */
dostruct([structure:happening|Rest], NumIn, NumOut, ArrayIn,
           [[size,'/happeningd',[A,P]]|ArrayIn]) :-
        member(type:discrete, Rest), !,
        member(substance:Subs, Rest),
        member(action:Action, Subs),
        bracket(Action, A),
        member(patient:Patient, Subs),
        bracket(Patient, P),
        NumOut is NumIn + 1.

/* If it is a delimited mass, make an array element with the
   delimitation annotation, call dostruct to process what is
   contained inside this structure, and incorporate the array
   address of the content of the delimited mass */

dostruct([structure:delimitedmass|Rest], NumIn, NumOut, ArrayIn,
          ArrayOut) :- !,
        member(extent:Extent, Rest),
        member(term:Term, Extent),
        member(det:Det, Term),
        bracket(Det, D),
        member(restr:Restr, Term),
        bracket(Restr, R),
        member(content:Content, Rest),
        NumNext is NumIn + 1,
        dostruct(Content, NumNext, NumOut,
            [[size,'/delimitedmass',[NumNext,[D,R]]]|ArrayIn],
            ArrayOut).

/* If it is an ordered set, make an array element with the
   cardinality of the ordered set, call dostruct to process what
   is contained inside the set structure, and incorporate into
   this array element the array address of the content */
dostruct([structure:ordered_set|Rest], NumIn, NumOut,
          ArrayIn, ArrayOut) :- !,
        member(extent:Extent, Rest),
        member(det:Det, Extent),
        member(content:Content, Rest),
        NumNext is NumIn + 1,
        dostruct(Content, NumNext, NumOut,
```

```
            [[size,'/ordered-set',[Det,NumNext]]|ArrayIn], ArrayOut).

/* If it is an ordinary set, make an array element with the
   cardinality of the ordered set, call dostruct to process what
   is contained inside the set structure, and incorporate into
   this array element the array address of the content */
dostruct([structure:ordin_set|Rest], NumIn, NumOut,
          ArrayIn, ArrayOut) :- !,
        member(extent:Extent, Rest),
        member(DetName:_, Extent),
        det2int(DetName, Int),
        member(content:Content, Rest),
        NumNext is NumIn + 1,
        dostruct(Content, NumNext, NumOut,
            [[size,'/ordinset',[Int,NumNext]]|ArrayIn], ArrayOut).

/* If it is concurrent, make an array element , call dostruct to
   process the major substructure and put the appropriate array
   index into the first position, call dostruct to process the minor
   substructure and put the appropriate array index into the second
   position of the array element*/
dostruct([structure:concurrent|Rest], NumIn, NumOut,
          ArrayIn, ArrayOut) :- !,
        member(substance:Subs, Rest),
        member(major:Major, Subs),
        member(minor:Minor, Subs),
        NumMajor is NumIn + 1,
        dostruct(Major, NumMajor, NumMinor,
          [[size,'/concurrent',[NumMinor,NumMajor]]|ArrayIn], ArrayMid),
        dostruct(Minor, NumMinor, NumOut, ArrayMid, ArrayOut).

/* If it is compound,make an array element, and call doparts to process
   all the items in the list of participants */
dostruct([structure:compound|Rest], NumIn, NumOut, ArrayIn, ArrayOut) :-
        member(participants:Parts, Rest),
        NumNext is NumIn + 1,
        doparts(Parts, NumNext, NumOut,
            [[size,'/compound',PartList]|ArrayIn], ArrayOut),
        series(NumNext, NumOut, PartList).

doparts([], Num, Num, Array, Array).
doparts([Part|Rest], NumIn, NumOut, ArrayIn, ArrayOut) :-
        dostruct(Part, NumIn, NumMid, ArrayIn, ArrayMid),
        doparts(Rest, NumMid, NumOut, ArrayMid, ArrayOut).

/* Utilities */
bracket(A, B) :-
        name(A, Achars),
```

```
        Left is "(",
        append([Left|Achars], ")", Bchars),
        name(B, Bchars).

series(From, To, []) :-
        From >= To, !.
series(From, To, [From|Rest]) :-
        Next is From + 1,
        series(Next, To, Rest).

det2int(exists, 1).
det2int(some, 4).
det2int(three, 3).
det2int(every, 15).
det2int(many, 10).
det2int(much, 10).
```

Bibliography

[1] Adobe Systems Incorporated. *The PostScript Language Reference Manual.* Addison-Wesley, 1985.

[2] Allen, J. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 11(26):832–843, 1983.

[3] Alshawi, H., editor. *The Core Language Engine.* Cambridge,Mass.:MIT Press, 1992.

[4] Bach, E.A. The algebra of events. *Linguistics and Philosophy*, 9:5–16, 1986.

[5] Barwise, J. and Cooper, R. Generalised quantifiers and natural language. *Linguistics and Philosophy*, 4(2):159–219, 1981.

[6] Burton-Roberts, N. *Analysing Sentences.* London: Longman, 1988.

[7] Carlson, L. Aspect and quantification. *Syntax and Semantics*, 14, 1981.

[8] Chapman, D. *Vision, instruction and action.* MIT Press, 1991.

[9] Cousins, M. and Metcalfe, J. *The vegetarian on a diet.* Thorsons Publishers Limited, 1984.

[10] Crangle, C. and Suppes, P. Context-fixing semantics for instructable robots. *International Journal of Man-Machine Studies*, 27:371–400, 1987.

[11] Crangle, C. and Suppes, P. Context-fixing semantics for the language of action. In J. M. E. Dancy, J. Moravcsik and C. C. W. Taylor, editors, *Human agency: Language, Duty and Value*, pages 47–76, 288–290. Stanford University Press, 1988.

[12] Dale, R. *Generating Referring Expressions in a Domain of Objects and Processes.* PhD thesis, University of Edinburgh, 1989.

[13] Di Eugenio, B. Understanding natural language instructions : the case of purpose clauses. In *Proceedings of 30th annual meeting of the Association for Computational Linguistics*, pages 120–127, Delaware, June 1992.

[14] Dowty, D. *Word Meaning and Montague Grammar.* D.Reidel, Dordrecht, 1979.

[15] Floyd, K. *Floyd on Great Britain and Ireland.* BBC Books, 1988.

[16] Hitzeman, J. *Temporal Adverbials and the Syntax-Semantics Interface.* PhD thesis, University of Rochester, 1993.

[17] Hobbs, J.R. and Shieber, S.M. An algorithm for generating quantifier scopings. *Computational Linguistics*, 13:47–63, 1987.

[18] How, K. Y. *A Processing Framework for Temporal Analysis and its Application to Instructional Texts.* PhD thesis, University of Edinburgh, 1993.

[19] Huntley, M. The semantics of english imperatives. *Linguistics and Philosophy*, 7:103–133, 1984.

[20] Hwang, C. H. and Schubert, L. K. Tense trees as the fine structure of discourse. In *Fall Symposium on Discourse Structure in Natural Language Understanding and Generation Working Notes*, pages 33–38, 1991.

[21] Hwang, C. H. and Schubert, L. K. Interpreting tense, aspect and time adverbials: A compositional, unified approach. In *Proc. 1st Internat. Conf. on Temporal Logic*, Bonn, Germany, July 11-14, 1994.

[22] Jackendoff, R. Parts and boundaries. *Cognition*, 41:9–45, 1991.

[23] Karlin, R. Defining the semantics of verbal modifiers in the domain of cooking tasks. In *Proc. 26th Annual Meeting of the Association for Computational Linguistics*, pages 61–67, Buffalo NY, USA, June 1988.

[24] Levison, L. Action composition for the animation of natural language instructions. Technical Report MS-CIS-91-28, University of Pennsylvania, Computer Graphics Research Lab, 1991.

[25] Ludlow, N. D. *Pictorial Representation of Text: Converting Text to Pictures.* PhD thesis, University of Edinburgh, 1992.

[26] Mayo, B. Objects, events and complementarity. *Philosophical Review*, LXX:340–361, 1961.

[27] Mel'chuk, I. A., Ershov, A.P. and Nariniany, A.S. Rita-an experimental man-computer system on a natural language basis. In *IJCAI Conference Proceedings*, pages 387–390, Tbilisi, USSR, 1975.

[28] Moens, M. and Steedman, M. Temporal ontology and temporal reference. *Computational Linguistics*, 14(2):15–28, June 1988.

[29] Moltmann, F. Measure adverbials. *Linguistics and Philosophy*, 14:629–660, 1991.

[30] Novak, Gordon S. Jr and Bulko, William C. Understanding natural language with diagrams. In *AAAI 8th national conference on Artificial Intelligence*, pages 465–470, 1990.

[31] Passonneau, R. J. A computational model of the semantics of tense and aspect. *Computational Linguistics*, 14:44–60, 1988.

[32] Pianesi, F. and Varzi, A. C. The mereo-topology of event structures. In *Proceedings of the Ninth Amsterdam Colloquium*, 1994.

[33] Pineda, L. A., Klein, E. and Lee, J. Graflog: Understanding drawings through natural language. In *6th Annual EUROGRAPHICS Conference*, 1988.

[34] Pollard, C. and Sag, I.A. *Information-based syntax and semantics.* CSLI Press, 1987.

[35] Pollard, C. and Sag, I.A. *Head-Driven Phrase Structure Grammar.* University of Chicago Press, 1994.

[36] Pratt, I. & Brée, D. The expressive power of the english temporal prepositional system. Technical Report UMCS-93-1-7, University of Manchester, Department of Computer Science, 1993.

[37] Pratt, I. and Brée, D. An approach to the semantics of some english temporal constructs. In *Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society*, pages 118–123. Lawrence Earlbaum Associates, 1995.

[38] Pratt, I. and Brée, D. How to translate some english temporal constructions into temporal logic. In Mmsili, P., Borillo, M. and Vieu, L., editor, *Workshop Notes: Fifth International Workshop on Time, Space and Motion*, pages D28–D38, 1995.

[39] Radford, A. *Transformational Grammar-a first course.* Cambridge University Press, 1988.

[40] Reichenbach, H. *Elements of Symbolic Logic.* London: Macmillan, 1947.

[41] Rock, S.T. Understanding repetition in natural language instructions - the semantics of extent. In *Proc. 30th Annual Meeting of the Association for Computational Linguistics*, Delaware, USA, 1992.

[42] Rock, S.T. Visualising multiple or extended execution of eventualities. Technical report, University of Edinburgh, Department of Artificial Intelligence, 1995.

[43] Schubert, L. and Hwang, C. An episodic knowledge representation for narrative texts. Technical Report 345, University of Rochester, Department of Computer Science, 1990.

[44] Shieber, S. M. *An introduction to unification-based approaches to grammar.* CSLI, 1986.

[45] Singh, M. A cognitive analysis of event structure. Technical Report TM-91-07, DFKI, Postfach 2080, D-6750 Kaiserslautern,Germany, 1991.

[46] Singh, M. The perfective aspect : an algebraic analysis. *ACL SIGMOL*, 1991.

[47] Singh, M. and Singh, M. P. Computing the temporal structure of events in natural language. In *ECAI 92*, pages 528–532, 1992.

[48] Smith, C. *The Parameter of Aspect.* Reidel, 1992.

[49] Song, F. Temporal analysis and discourse processing. In *Fall Symposium on Discourse Structure in Natural Language Understanding and Generation Working Notes*, pages 106–111, 1991.

[50] Steedman, M. J. Verbs, time and modality. *Cognitive Science*, 1:216–234, 1977.

[51] Talmy, L. The relation of grammar to cognition - a synopsis. In D. Waltz, editor, *Theoretical Issues in Natural Language Processing 2.* Association for Computing Machinery, New York, 1978.

[52] Vendler, Z. *Linguistics in Philosophy.* Cornell University Press, Ithica, New York, 1967.

[53] Verkuyl, H. J. Aspectual classes and aspectual composition. *Linguistics and Philosophy*, 12:39–94, 1989.

[54] Webber, B. L. So what can we talk about now? In R. C. Berwick and M. Brady, editors, *Computational models of Discourse*, pages 331–371. Cambridge, Mass. : MIT Press, 1983.

[55] Webber, B. L. and Di Eugenio, B. Free adjuncts in natural language instructions. In *Proceedings of 14th International Conference of Computational Linguistics*, pages 395–400, 1990.

[56] Wilensky, R. The ontology and representation of situations. In Allen, J., Fikes, R. and Sanderwall, E., editor, *Proceedings of the second international conference on principles of knowledge representation and reasoning*, pages 558–569. Morgan Kaufmann, 1991.

[57] Zemach, E. Four ontologies. In F. J. Pelletier, editor, *Mass Terms : some philosophical problems*, pages 63–80. D.Reidel, Dordrecht, 1979.