SemNet: the knowledge representation of lolita

Baring-Gould, Sengan

SemNet: The Knowledge Representation of LOLITA

APPENDICES

(Volume II/II)

Sengan Baring-Gould

Ph.D. Thesis

University Of Durham

Department of Computer Science

2000

Contents

Appendix A

Assumptions

Knowledge, Understanding, and Ontology, three words whose meaning seems clear. But beyond the obvious, lies a bedrock whose deeper understanding dissolves many potential mistakes.

LOLITA's purpose is to process the knowledge texts express in various ways: summarisation, translation, template analysis, determining contradictions, query answering, searching for topics, planning, and so on. In order to perform such a wide spectrum of tasks, it must be assumed that all these tasks involve shared processes. Indeed, if the tasks were completely independent, a small set of independent solutions would be more appropriate. This assumption is based on the observation that all the tasks discussed are knowledge processing tasks. In particular, they are tasks involving mainly the processing of knowledge extracted from Natural Language texts. Thus a two-part model emerges where on the one hand processing of Natural Language text results in some form of knowledge, and on the other hand, this knowledge is processed to fulfil the tasks required of the system. This leads to three important questions:

1. What is the knowledge discussed?

2. What is knowledge processing?

3. What is meant by "extraction of knowledge from a text"?

A.1 Reasoning, and hence the symbolic paradigm

The different forms of knowledge processing can be thought of as different forms of reasoning. For instance, summarisation involves deducing certain facts from a mass of data, by prioritising the importance of each. This issue is examined in greater depth, and then means by which reasoning can be achieved are discussed.

A.1.1 Why is reasoning important?

The whole perspective of the LOLITA project is to a great extent determined by the requirement that the LOLITA tool should be able to reason with the knowledge provided by the texts it processes. Why require the ability to reason? To an extent the question is answered by considering the ultimate fulfilment of A.I.: the reproduction of all successful human behaviour. However there is a more pragmatic reason for this requirement. The assumption is made in this thesis that the features required of a knowledge base for reasoning will be useful, if not fundamental, for other types of processing.

It is worth discussing the reasons for such an assumption. Many of these will appear clearer when the notions of interpreting a text are discussed later in this appendix. At this stage, it is simply worth mentioning examples where the requirement provides benefits.

Valid reasoning is the process of making explicit knowledge already implicit in the knowledge base. In this sense it does not add any information. Non-valid methods of reasoning make assumptions, and thus add a degree of uncertainty to their conclusions. It is obviously possible to deduce a very precise fact from imprecise data, but with a very low degree of certainty. Since highly uncertain knowledge is pragmatically undesirable as it is of limited use, it is preferable for reasoning to deliver reasonably certain conclusions. This in turn results in a reduction of their precision. Thus each reasoning step can be seen as deteriorating the precision of the resulting knowledge. This means that for useful reasoning to be done, the initial information should be well defined and precise. Natural language tends however to be ambiguous. For instance the word "star" may either mean an astronomical object, a celebrity or a particular type of participation in a play, film, etc. In order for reasoning to be successful, the notion referred to must be determined. However this also proves useful for translation. Indeed each of these possibilities corresponds to a different word or expression in French (une étoile, une star, être la vedette) or German (eine Sterne, ein Star, die Hauptrolle spielen)[1].

Consider another type of tasks to be addressed: information extraction tasks such as contents scanning, or summarisation. Such processing requires the source information to be classified into a set of themes. A knowledge base designed to allow reasoning at various levels of detail helps this process: if information is organised in the knowledge base by theme, it can more easily be reasoned about at the subject level. Thus these benefits motivated by reasoning, also spill over to other tasks.

The need for reasoning is however best illustrated by its need in Natural Language processing itself. In the following examples, reasoning is needed to determine what the situation described is: *"John died. Bill pushed him"*. To determine that Bill pushed John before John died, causal reasoning is necessary. Similarly, quantification resolution such as *"John visited every house on a street"*, *"John visited every house on a square"*, and *"John visited every patient in a private room"*[2] can be argued to require reasoning. Similarly, in the single sentence *"John talked while he ate"* the only referent for *"he"* is John. The situation changes when the sentence *"John was not hungry, so Peter started eating by himself"* precedes it: now the most likely referent for *"he"* is Peter. Determining this requires a minimal degree of understanding, i.e. reasoning.

A.1.2 Means of reasoning

Having established that reasoning is desirable, the question of how reasoning can be achieved remains. Various alternatives exist: symbolic reasoning and probabilistic methods such as fuzzy logic, neural nets or Markov models. The choice for LOLITA

[1]For further details see [Morgan et al. 94].

[2]Examples taken from [Alshawi 92]

is however clear: symbolic reasoning.

A.1.2.1 The symbolic paradigm

Reasoning and the language in which knowledge is expressed are strongly linked. Most knowledge is expressed in natural language. This relies on human understanding and presents many problems. Natural language is rarely unambiguous, that is to say that there is often more than one meaning associated with each word. This makes keeping track of precisely which meaning was meant difficult in complicated arguments. Moreover assumptions can easily slip into an argument undetected.

Philosophers realized that these problems were due to the fact the whole reasoning process was being conducted at the conceptual level, and was thus to a large extent implicit. In search of even greater clarity and rigour with which their arguments could be established, they decided to bring the reasoning to the syntactic level where it could be made explicit. This explicit method of reasoning is based on the use of symbols to represent concepts, and statements written as strings of these symbols. Proofs consist of explicit sequences of reasoning steps, taken from a limited set of legal steps. Thus a clear path is established from the premises to the result. Hence, no assumptions are unintentionally introduced into the proof.

Because this explicit expression of reasoning is independent of the agent, it is of particular interest to the implementation of reasoning in LOLITA.

A.1.2.2 Successes of the symbolic paradigm: Science and Mathematics

Science distinguishes itself from other forms of knowledge by its methodology and evaluation criteria. Thus scientific models may be expressed in natural language as is the case for Biology. However, symbolic techniques are used when a model can be expressed precisely in order to gain in clarity and rigour. For instance in Chemistry reaction formula show what the reactants and the products of a given reaction are. These can be combined using the reactance of the various reactions and information giving the experimental conditions to obtain an idea of the products. The science which is the most expressed in a symbolic language is physics. It is expressed in mathematical language, which makes it completely agent-independent. Whatever the interpretation given to the Schrödinger equation in Quantum mechanics, over which a considerable debate still rages, its ability to make accurate predictions within the range of its application remains undisputed. Its success is obvious, as many of this century's major technical innovations such as micro-electronics, telecommunications, and the computer, are direct applications of it. This success associated with symbolic reasoning, in particular mathematics, warrants investigation as a possible basis for the reasoning to be implemented as part of the LOLITA tool.

For the purposes of this thesis, the discussion of mathematics will obviously be vastly simplified. However, certain aspects of it are extremely relevant. At its most surface level, mathematics can be considered as the manipulation of statements through substitution: expressions are replaced by others that are believed equivalent. Through this process, complex statements can be built from a small set of initial statements – the axioms. Inversely, complex statements can be decomposed into the axioms to prove whether they are based on the axioms, or not. Since the axioms are believed to be true, statements based on the axioms are believed to be true. Similarly those statements that contradict the axioms are considered false. This viewpoint is similar to that of the strict formalist in that no additional agent dependent faculties such as intuition are assumed. [3]

The problem of intuition usually arises when it comes to elaborate a proof: for every step in the proof, there are many possible substitutions to choose from. Most of these do not lead to the desired solution. However, this is rarely a problem for standard applications of mathematics, such as physics, where standard reasoning techniques usually lead directly to the answer. It is more a problem for

[3]This position has been dealt a blow by Gödel's theorem which showed that some statements may seem intuitively true within the framework of the axioms, yet not be provably either true or false. However Gödel's theorem is more of mathematical interest than of scientific since it does not appear in physics, or in any other science. Similarly such a special case is unlikely to emerge in a NLP system. Even if it were to, it would simply maintain its status as a proposition the system cannot prove or disprove.

the formulation of new problems, and new proofs, and thus more of concern to mathematicians. Thus, even limited to standard reasoning techniques, which can be automated, mathematical language can provide great power of precision and rigour with a wide variety of reasoning methods.

Mathematical language can therefore be seen for our purposes as statements manipulated by substitution rules. The statements themselves are composed of a string of symbols. Substitution can be thought of as pattern matching against strings of symbols and substitution therein. Thus mathematical reasoning can be thought of as symbolic reasoning within a substitution paradigm. However this view is slightly restricted: it might seem that all we are dealing with is various jumbled permutations of symbols. Although the reasoning process may function by substitution, in order for the process to be useful, various symbols must be given a meaning. Some symbols such as operators are given a meaning within the set of axioms and determine which substitution patterns are believed. Others such as variables must be assigned a meaning, stating what they are representing. A third category, such as numbers, must be partially assigned a meaning, such as the measurement unit they are expressed in. So a particular symbol, or class of symbols, is bound to a particular concept: the symbol is said to represent the concept. This process occurs outside of the substitution framework, or at the meta-level. Thus the mathematics and the substitution paradigm is free of the particulars of any concepts. It is an abstraction where relations between concepts such as their mutual behaviour or dependency can be specified. However if the substitution paradigm is useful for reasoning, it is only useful because of the conclusions it obtains within the conceptual framework considered. Hence the link between the symbols and the concepts they represent is essential. This assignment of concepts to symbols, and meaning to statements including them is called interpretation. It is essential yet lies outside of the symbolic reasoning paradigm.

A.1.2.3 Lack of equally successful alternatives

Reasoning symbolically thus allows conjectures to be proved formally. The development of mathematics over more than 2500 years provides a vast resource of varied explicit reasoning methods, and standard techniques. Although recent work has investigated other non-symbolic forms of reasoning, such as fuzzy logic or neural nets, none of these have attained the degree of sophistication that symbolic methods have. Indeed, the unique success of symbolic reasoning has been demonstrated by the importance and unique role of the scientific paradigm this century. No other method of reasoning has attained such heights. Thus, although other forms of reasoning are interesting possibilities in themselves, symbolic methods have the decisive advantage.

A.2 Symbolic, and other forms of Knowledge

Attempts at defining knowledge often reduce to tautologies. Rather than defining knowledge itself, emphasis will be laid on desired behaviour. By considering the effect on behaviour that is expected of knowledge, the nature of the type of knowledge considered is clarified. It will be helpful to consider widely accepted forms of knowledge and the behaviour which is expected after acquiring such knowledge. Scientific knowledge is of particular interest since the behaviour in which it results is expected to be identical whatever the agent which acquires it. That is to say, for instance, everyone solving a physics problem should obtain the same result, if they have all been trained in physics. In this sense, scientific knowledge can be taken as fully defined and independent from any knowledge that might vary from agent to agent. Thus by considering scientific knowledge, no particular agent need be considered either in the description of the knowledge, or in the resulting behaviour.

A.2.1 Scientific Knowledge

Scientific knowledge consists of a set of models which are intended to reflect the behaviour of the world. These models postulate a set of primitive entities and relations which hold between them. The purpose of the knowledge thus set forth is to allow events in the world to be predicted. Thus the usefulness of the model depends on its ability to predict outcomes correctly. Hence, within the limits established for the model, the relations should allow the model to reflect any sequence of behaviour observed in the world. Similarly the model should allow no sequence of behaviour which does not occur in the world. If these conditions are observed, the relations are said to "hold in the world". Thus if the model holds, it specifies sequences of behaviour possible in the world.

The type of event which can be predicted by the model will determine its usefulness. Indeed, a model specifying that something will happen is not very useful. Thus the precision of the model's prediction is relevant. However, the amount of information the model requires to make a prediction is also important: if more information is required than is usually available, the model will be of little use. Conversely, if little information is required in order to make a precise prediction, the model is very valuable. Taken to the extreme, a model requiring no information and predicting without failure the winning numbers of the national lottery would be of tremendous value! The value of a model therefore depends on the gain in knowledge it provides. It is a trade-off between the amount of information put in, and the amount obtained.

The usefulness of the model also increases with the range of phenomena of which it successfully captures the behaviour: the more a model can be used, the more useful it is, since effort is required to master it. Similarly, the usefulness of the model increases with its simplicity, since the effort involved in mastering it and its associated paradigm decreases. This stems from the fact that models and information in general only exist because agents created them to use them. Thus their value to their creators must be taken into account. For the type of agents considered here, remembering a fact incurs a cost. Thus remembering a large collection of facts is more expensive than remembering a smaller such collection. Hence if a set of facts is smaller than another and allows as much or more information to be derived, then it may be the more cost effective. Determining this smaller set of facts and then the process of deriving information from this set both also involve a computational cost [4]. Thus it is the total combination of these costs which will determine the cheapest form in which the model can be used and remembered.

The observations just made lead to some interesting consequences. The first is that the number of entities considered should be minimised, or as Ockham said, do not multiply entities without necessity: postulating the existence of an entity is a fact, and involves a cost. Only those mental entities, or concepts which can sustain the cost of existence are sustainable. These include such abstract concepts as gravity or the curvature of space.[5] Another consequence is that a model may include tautologies: facts which could be derived from the rest of the model, but which are used so often that it costs more to derive them than to remember them. Examples include the Compton equation that is an application of the special theory of relativity, or remembering one's multiplication tables when it is possible to calculate them by addition.

From the discussion, it may seem natural to assume that a general model of everything based on a *minimal* set of primitives should be the basis of the agent's knowledge. However no such general theory of everything exists as yet, omitting for this discussion the question of whether such a theory is possible. Scientific knowledge therefore consists of a set of models. The question arises under what conditions all these models can be combined into a generalised model covering the whole range of phenomena that all the models cover separately. For any model to be scientific, i.e. for it to predict anything which will hold, it must be self-consistent: any prediction based on any part of it should be the same as any other based on any other part of it, assuming an identical situation is considered in all cases. This is also true of general models built from many other models. It is an

[4]It should be noted that here the meaning of computation refers to the general notion of reasoning, reckoning

[5]It should of course be noted cost expenditure depends on one's purpose. The scientific purpose is prediction. Other purposes can include the creation of a secure feeling about life and death.

obvious necessity for theories covering the same range of phenomena, and which are submitted to the necessity of holding in the world. Newton's theory, Quantum mechanics and Relativity conform to this rule, by converging in each other's limits.

Mutual self-consistency between models incorporated within a general model is encouraged further, when one considers Ockham's razor applied to the whole general model. General models which postulate fewer additional concepts may be advantaged, since they will have a lower memory cost. Thus general models composed of models sharing concepts may be preferred. This sharing of concepts improves the integration of smaller models into the general one. Hence, an entity E which may be considered primitive in one model ($\mathcal{A}$), may be derived from another ($\mathcal{B}$). In the general model ($\mathcal{G}$), E could then be considered non primitive. However, this depends on the way it is to be used: when the general model is to be used to solve a problem easiest solved by model $\mathcal{A}$, then it may well be worth using E as a primitive. This is similar to the issue of maintaining tautologies, such as Bayes theorem, in the model if they reduce the computational cost of deriving desired information. The issue here is one of granularity: Instead of considering any particular level of the model as more "primitive" than any other, the question is how to use the system of relations provided by the model to derive with the least effort, desired information. Although science itself does not provide the answer here, this is an implicit part of a scientist's training: how best to apply the model.

There is a final aspect that has not been covered. For a model to be used to make a prediction, the details of the initial situation, or initial conditions, must be known. These facts are expressed in a form where they can be used by the model. Often they are chosen, rather than others, because they are usable by the model. Thus the concepts of the model are used in order to establish and express facts. Hence from a system of thought, a complete model of the world and means of expressing facts about it emerges.

A.2.2 Scientific knowledge: Too limitative? Alternatives?

The notion of knowledge indicated by science is one where prediction is the key feature. However not all forms of knowledge need stress this aspect. Indeed, a large number of phenomena are not predictable. Art falls in this category, where knowledge allows one to appreciate a work, but not to predict its form. The question therefore arises of whether the view of knowledge given by science is not too restricted.

Again, the question is partially solved by recalling the purpose of the LOLITA project. The problem is to produce a tool capable of identifying information and reasoning about it. For instance, template analysis, query answering and topic searching can all be roughly approximated to identifying information. Determining contradictions, summarisation and planning all involve some form of reasoning. Such reasoning often involves predicting outcomes. Hence at least some of LOLITA's knowledge must allow prediction.

Consider again the phenomena that do not allow predictions. For instance, practical skills such as knowing how to ride a horse are gained by practical experience. They are not something one can learn from another person by only following instructions. Instead direct experience, perception from one's senses is involved. Similarly, artistic knowledge is not readily expressible in human language. Where words are used, it is often to hint at shared direct experiences. This is not the type of knowledge that is readily used for the tasks LOLITA is to be assigned. Templates, queries and topic searching all involve factual knowledge. This is the type of information expressed by scientific knowledge.

Although science has been very successful at allowing predictions to be made, other forms of knowledge can be used in this way: History is an important example. It can be seen as an enormous case-study of human behaviour. But it is the agent-independence of science that distinguishes it from other forms of knowledge: the predictions each agent obtains will be the same. History, on the other hand, relies on the historian's common sense. Hence different historians can see the same facts in a different light. Since LOLITA requires an explicit expression of much of the knowledge of this assumed common sense, history-like knowledge is not a good basis from which to build a knowledge base. LOLITA's knowledge base cannot assume any already existing knowledge, but must be self-sufficient.

Hence, science distinguishes itself as an agent independent form of knowledge, which is self-sufficient. It provides in its model of the world, a set of facts and a means of prediction. These features are those required of the knowledge of a fact processing tool such as LOLITA. Hence, it is attractive as a model on which to base such a system. However, this knowledge should also be amenable to an explicitly defined form of reasoning, if it is to be implemented in a tool.

A.2.3 Limitations of the Scientific model of Knowledge

A.2.3.1 Belief

The discussion of knowledge has been limited to scientific models, which are considered true in the sense that they hold in the world. This assumption may not necessarily be fruitful for the knowledge bases of agents. It implies that all the knowledge in the knowledge base is correct. Hence no knowledge may be added to the knowledge base if it contradicts the knowledge already present, so as to preserve the consistency of the general model. This is very inflexible, and leads to practical difficulties if the agent is to perform tasks such as translation for texts with which he does not necessarily agree. Consider an example. The agent is told *"Jane lives on the moon"*. He does not agree with this statement, so the interpretation phase which is responsible for maintaining the consistency of the knowledge base fails, and the agent fails to produce any translation.

Since one of the purposes of the LOLITA tool is to provide translations, such a scenario is tantamount to failure. The only other possibility within the paradigm that all knowledge in the knowledge base is equally true, is to accept all statements as being true. This violates the notion of consistency, and allows both a statement and its opposite to be true simultaneously. This will lead to inconsistent behaviour, for instance of a planning component, where contradictory courses of action are suggested. Again, such behaviour is unacceptable. The problem lies with the paradigm: not all knowledge in the knowledge base need be true.

The problem is therefore of allowing statements to be represented in the knowledge base without assuming them to be true. The question of whether to accept a statement as true, as expressing something holding in the world, is the question of belief. If the agent does not consider the statement to hold, there is no reason for him to act upon it. However he is still able to access the statement within his knowledge base, and to process it in some way such as translation. If on the other hand, he believes it, he is both able to translate it and finds it useful for planing. This resolves the problem, but raises some interesting consequences.

The first consequence is that the degree to which the agent believes in each statement must be added to the knowledge base. This in turn means that the knowledge base is not some abstract knowledge available to all, such as science, absolute in a sense, but is the belief of the agent himself: it is what the agent believes the world to be like. Since a degree of belief must be assigned to each statement, one might wonder what degree should be given to statements which are "known". For instance, one might say that the agent "knows" that apples taste sweet. Semantically the verb "to know" is a little tricky. For instance, one does not say one knows that God exists, rather that one believes in Him. In this sense, "to know" is an issue of direct experience. Similarly, one could not say that a person without any sense of smell knows that a rose smells sweet, only that he believes so. However, one can say that one knows that Bill Clinton is the president of the United States, without having actually experienced it. The first meaning of the verb could not be applied to LOLITA's knowledge, since she lacks any external senses. However, the second implying common knowledge, facts for which there is agreement of the majority of people in one's peer group and to which one subscribes, could be applied to LOLITA. This is however rather complicated to deal with. Thus, for the sake of simplicity, LOLITA will treat knowing something as assigning it the highest degree of belief.

The second consequence lies in statements the agent does not believe. There is an infinite number of potential statements that the agent need not believe. The question therefore is which statements the agent does not believe, yet provide some information worth the cost of remembering them. The first such type of information is suggested by the example of translation: information stated by another agent. Although one may not agree with the statement itself, one can believe that it has been stated. Similarly, one may believe that the agent believes something, without believing it oneself.[6] Another example is if one believes something would cause something else to happen, without for that matter believing that the first thing actually has happened. For instance, one may believe that if global warming were to occur, London would be flooded; yet one may also not believe that global warming has occurred so far. All these cases share a common feature: they depend on something believed. In this way they still constitute part of the agent's knowledge: they correspond to something the agent believes holds in the world. Without this dependency, they would not constitute part of the agent's model of the world, and hence would not be part of his knowledge. Such vacuous statements would provide no information, yet exact a cost to be remembered.[7] Thus they should be purged from the knowledge base.

A.2.3.2 Inconsistencies

So far, within the framework of the knowledge base similar to scientific models, it has been stated that inconsistencies should not occur. However, exhaustive testing for inconsistencies during the interpretation phase would incur an exorbitant computational cost: every new statement would need to be tested not only against all the statements in the knowledge base, but also against all their consequences. This is clearly infeasible for any large scale knowledge base. The question therefore arises of the extent to which inconsistent information may be allowed in the knowledge base. Again, it is a matter of cost benefit analysis. As discussed in A.2 *(p. A-7)*,

[6]The means of deriving the other's belief is irrelevant, it could be because the other told one, or a deduction from his behaviour. The point is that this information must be representable in the knowledge base.

[7]Indeed, they also increase the search space considered when reasoning, hence also increase the cost of reasoning.

every inconsistency leads to a possibly incorrect prediction: the full knowledge base implying both a conclusion and its negation, but the reasoning process considering perhaps only one of these conclusions. Thus the question becomes dependent on the extent the inconsistency can affect the predictions important to the agent, and which will determine its success at its tasks. For important tasks, a higher cost will be payed in order to avoid inconsistencies, whereas virtually no checks will be performed for the unimportant ones. Another question is raised: whether the agent has any important tasks, and if so, what they are. This question will be considered later.

It should be appreciated that even information already within the knowledge base may be inconsistent. Indeed, in any real situation, the range of application of some of the partial models may intersect, yet provide contradictory conclusions. For instance, one may both believe in the weather forecasts and in traditional rythmes such as *"Sky at night, shepherd's delight"*. Yet in certain circumstances, they may contradict each other, so one must choose which to believe. In this case, one may choose the latter, if one finds it makes correct predictions most of the time. This is a matter of belief: one believes more in one model than in another. Thus belief may provide a means of quick resolution of inconsistencies.

A.2.3.3 Certainty

Both the notions of scientific truth and consistency of the knowledge base have been shown naive or insufficient within the context of modelling the knowledge of an agent. Instead, the notions of belief and consistency-checking incurring a cost have been introduced. This leads to the idea of two measures of belief: belief and certainty. Belief itself states the amount the agent believes a fact, whereas certainty expresses the amount of effort spent on checking it. This is best illustrated by an example: One may believe that one has locked the front door, yet not be willing to bet one's life on it. Certainty is used in the process of recovering from an inconsistency when contradictory conclusions hold similar degrees of belief: it indicates for which further checks may be required.

A.2.4 Common Sense Symbolic Models

Despite the advantages of symbolic methods, a certain amount of cynicism pervades the N.L.P. field about the application of symbolic methods to common sense reasoning: *"We've tried, and failed. Even such a large project as CYC tried, and failed"*[8] seemed to be a wide-spread attitude at the recent COLING 96 conference. This section investigates the difficulties involved in the building of a Common Sense Knowledge Base (C.S.K.B.), and the requirements it must satisfy. A case study of CYC, the best known attempt at building such a C.S.K.B, follows.

A.2.4.1 Common Sense Knowledge Base: Difficulties and Requirements

Rather than being just a "data warehouse", a C.S.K.B. will be assumed to be associated with advanced reasoning capabilities, to allow it to solve previously unencountered problems.

A.2.4.2 Mutual Dependencies

Some of the difficulties facing C.S.K.B.s are described in 2.2 *(p. 11)*: A C.S.K.B. is more than a collection of expert systems. Just as different scientific models must be consistent and not contradict each other to form a general model, a C.S.K.B.'s knowledge about various fields must not produce contradictions. This would be a simple matter if they were completely independent, but they are not. Indeed, when relevant, this mutual dependency between fields is the source of the depth an agglomeration of expert systems lacks. For instance, the meaning of death within the medical field is quite different to that within a military, judicial or emotional field. For a C.S.K.B. to understand why John attacked the drunk doctor who told him his mother had died, it must incorporate knowledge from more than just the medical field. It is the ability to reason at many levels of granularity, and in many different models that gives a deep understanding to a system, in the sense discussed in A.3.1 *(p. A-28)*.

[8]Personal Communication, Pierre Isabelle (a developer of METEO) in 1996

Although mutual dependency between knowledge of different fields is important for deep understanding to be displayed, it is also the cause of many problems to developers, if ignored. It means that adding new knowledge has a non-local effect to the knowledge base. Thus, reasoning which previously worked, may now fail. To appreciate how important this is, consider the simplest model of dependency, where a rule depending on another has a detrimental effect on reasoning. If the K.B. contains n assertions, there are $\frac{n*(n+1)}{2}$ such possible dependencies[9]. Some of these will be wanted, but most will not. Thus, as the size n of the knowledge base increases, the scope for error increases as n^2. Furthermore, most dependencies will not be so obvious, but will be made through many statements. Each dependency taken alone may seem reasonable, but overall they lead to fallacious reasoning. Means of keeping these dependencies explicit and under control are therefore essential.

A consequence of mutual dependency is that the quality of the models input into the K.B. is of greater importance than the quantity. Adding kludges to make things work will only achieve fleeting success and long-term failure. Each kludge adds to the size of the knowledge base, increasing the risk of errors in the rest of the knowledge base.

A.2.4.3 New common sense models

For common sense reasoning to be possible, many models of common-sense happenings must be built. These models capture the relations between different concepts, which are used in reasoning. Unlike a collection of simple facts, they provide a generic framework for all reasoning in a particular field. Indeed, they can be seen to define the concepts in terms of which facts will be expressed (see A.2 *(p. A-7)*). Thus it is the organisation of knowledge which they provide, which enables the full set of implications of any fact to be grasped.

Most of these models must be created, since they are not available in books, or the other usual sources of knowledge. They will determine the concepts the C.S.K.B. can represent and reason about. Thus if a model is incomplete, whole classes of

[9]Each new rule can interact with the n previously existing rules.

concepts may not be expressible to the C.S.K.B. in a manner that it can reason about them. Creating these models, just as creating scientific models, requires much experimentation to determine what is inferred wrongly or approximately, and what still needs to be covered. This might seem easy: most people have common sense knowledge. However first impressions are deceptive: Making implicit knowledge explicit is very difficult, as the example of linguists demonstrates: they too, have the task of making explicit the implicit knowledge of the rules people use in language.

A.2.4.4 Implicit and Explicit Models: Important differences

In theory an implicit model is equivalent to determining many of the consequences of an explicit model. If all of these consequences are present, the assertions stated by either model are equivalent. However there are important differences:

An explicit model is clear, maintainable, relatively small, flexible and internally well structured. It is however difficult to build, as the general rules that form it can easily encompass too wide a scope, leading to incorrect predictions. However, the concepts it uses are clearly defined, so that the statements made are clear. The internal structure is a boon to reasoning algorithms which can use it to limit the search space they must consider when determining new facts. Furthermore, as explained in A.2 *(p. A-7)*, there is no efficiency penalty, since often used combinations of rules can be expressed explicitly, allowing reasoning at a lower granularity. Finally, general rules are more flexible since they allow new concepts to be added to the knowledge base, and will apply to them too.

An implicit model has little internal structure, is hard to maintain, and sprawls. As a result it may prove difficult to build as it does not provide clear definitions of concepts, lacking some model in which to describe them. The lack of clarity can result in terms being used for similar but different concepts resulting in lack of resolution, and thus reasoning power[10]. However, implicit models have the advantage of

[10]For instance, at one point CYC's K.B. contained in 2 million facts, but this decreased to 400,000 after work to generalise axioms. The extent of this decrease could be understood to reflect a naive bottom up approach to building knowledge, due to the implicit model.

reducing the chance of over-generalised claims by rules. They are also less difficult to build and require less qualified knowledge enterers. Efficiency-wise, common-sense knowledge tends to be more local to the concepts it describes, and the lack of deep models decreases the search space. But this is accompanied by a decrease in flexibility, in particular with respect to new information, since knowledge is not encompassed in "first-principle" models.

Thus a C.S.K.B. should use explicit models.

A.2.4.5 Example

An example gives a flavour of the type of C.S.K.B. satisfying the above requirements, and of the difficulties in writing it.

Suppose that one wished the C.S.K.B. to know that hand-gliding is a dangerous activity. One could write this explicitly in the C.S.K.B.: hand-gliding is a specific activity, so there is little chance that making this statement could lead to incorrect reasoning. On the other hand, being a safe bet, it says very little. What about other similar activities, such as racing cars? Are they dangerous? One might deduce that a better rule would be "all sports are dangerous". However this rule would lead to the incorrect conclusions that chess and snooker are dangerous, as they are sometimes classified as sports. Thus neither rule is satisfactory: the first is over-specific, the second is over-general.

Instead, one could build an explicit model from which hand-gliding and touching toxic substances could be derived to be dangerous. One means of achieving this is to consider the human and in particular his body as a system of interdependent phenomena working within certain normal ranges. Examples of such phenomena are the heart rate, bodily strength, speed of reflexes, amount of time concentration can be maintained, amount of pressure that can be withstood, stress levels, minimum food consumption and so on. What characterises physical sports is that they involve using various of these phenomena outside their normal ranges and close to their limits. But these limits are precisely the point where the system collapses, breaks down. For the human body, such a breakdown is called death.

If danger is defined as increasing the potential of destruction, sport's usage of the phenomena constitutes a danger to the human body. This allows a much more general understanding of danger. Not only are physical activities involving such high tolerance levels dangerous such as car-racing, but also are excesses such eating too much (binging) or eating too little (starving), eating too much of a particular thing, taking too much of some drug (although this has not a minimal level)... It also provides clearer definitions of notions such as food: food is anything that feeds a human, i.e. increases his food level without incurring too much damage to his body. Thus the explicit model results in far more power.

Consider the systems model further: it accounts for many other phenomena. The fact that the parts of the system are interlinked allows some of them to adapt in response to the needs of others. Thus when making a physical effort, more oxygen is needed by the muscles. I.e., the heart-rate increases, and one pants. Similarly if one breathes at high altitude without being used to it, one pants until one's body has increased the number of red blood cells in one's blood. Not only human bodies are such systems. So are animals, and indeed many vehicles like cars or tools. From this one can deduce, horses can be damaged if ridden too hard, car engines can be burnt out, nuclear stations can suffer meltdown, and employees can suffer burn out after too much stress.

The systems model also provides a basis for interpretation of analogies and metaphors: It is because a car is a system, that one can say that it drinks, or or that it died. One can talk of torturing a machine, because torturing involves taking constituent parts of a system over their safe limits or damaging them resulting in "pain". Indeed, any arduous activity, like sports, can be called torture. Pain itself can be defined as a warning signal from a particular part of one's body that the safe limits have been transgressed, resulting in the understanding that painful things should be avoided.

Hopefully this example conveys the power that a good explicit model can express. However, building this model requires a deep understanding of how a computer reasons. Determining causal models for simple phenomena requires the ability to generalise facts in a coherent way, so that phenomena which have similar abstract behaviours are modelled as instantiations of the same abstract behaviour. Far from being a job for underpaid knowledge enterers, who simply flesh out the ontology A.I. researchers devised, this is research task in itself.

A.2.4.6 Requirements particular to LOLITA

For LOLITA to communicate with people, her concepts must attain a certain degree of commonality with theirs. Since she communicates via language, language will be the basis from which the set of her concepts will be determined. Most words used in natural language are associated with at least one concept. Those that are not perform some structural role building the sentence. For instance "the" which has no independent meaning. Thus any dictionary might be thought to provide a list of concepts to use. Similarly, more recent developments such as Wordnet [Miller 90], provide a set of concepts associated with each word. However, the problem of which concepts to choose is not simply a matter of adopting somebody's ontology. Some ontologies may distinguish finely between many nuances of meaning. These differences may require a lot of effort to disambiguate, yet return little benefit to later processing. Other ontologies may prove too coarse, leading to mistakes in later processing, for example translation. Therefore, the value of such an ontology will be measured purely in terms of how effectively it helps in processing and reasoning tasks, and in the general working of LOLITA[11].

Since LOLITA is to communicate with human agents in a successful manner, she must be able to draw inferences normally expected of a normal person: As discussed in A.3.2 *(p. A-32)*, communication involves predicting what to say for one's interlocutor to understand what is meant. This only can work if all people share some degree of commonality, not only in their concepts but also in their knowledge and ability to reason. If LOLITA is unable to make the inferences expected of her by her interlocutor, either he will have to adapt his expression to her or he

[11]Notice that as the disambiguation and reasoning abilities of LOLITA grow, so can her ontology. It is therefore useful to start with a smaller ontology than expected later

may find her useless for his purposes. Either way, her behaviour cannot be deemed successful. Determining the expected inferences is not easy.

A.2.5 CYC: a case study

CYC started in 1984 at MCC, the U.S.A.'s oldest think-tank [Stipp 95]. It was to be a 10-year 25-million-dollar project and was initially funded by the U.S. department of defence. Its brief: build a knowledge base and its associated reasoning tools which can capture the common sense knowledge people share. For instance it should know that *"people stay dead once they've died"*, and most other such well-known facts. This type of knowledge cannot be acquired automatically, as it is too obvious to be explicitly stated in books. The final goal was to produce tools that display "common sense", such as giving (credible) reasons for John's sadness on the death of his dog. Twelve years later, the project is still "in business", but is developed by a private venture called Cycorp.

Despite the vast amount of money put into it, totalling over 25-million dollars, the CYC project has not been a resounding success. This section discusses its current status, and examines the CYC methodology in light of the requirements for a C.S.K.B.

A.2.5.1 CYC's functionality

Little information is publicly available about CYC's current functionality. However the little available does not prove impressive. Vaughan Pratt of Stanford University attended a demonstration in 1994. He wrote and distributed a report of his visit, [Whitten 95], which gives the impression of a bugged implementation, and a very unstructured method of building CYC's knowledge. For example, CYC knew that hand-gliding and touching toxic substances had as consequences death, and it knew about food, but it did not know that people need food in order not to die.

Dr. V. Pratt was shown two demonstrations of commercial applications of CYC commissioned by corporate sponsors. The first demonstration involved CYC detecting errors in multiple relational databases for the U.S. Department of Defence. This was less a demonstration of CYC's abilities, than proof that CYC can be used for tasks of practical use. Although expert systems could perform this task, the demonstration indicates that CYC contains knowledge that no one might have thought relevant when building a dedicated expert system. The second demonstration involved finding pictures in an database of 20 pictures using queries in Natural Language. Each picture had been previously described by CycL assertions. This appeared to work for simple sentences, using common sense reasoning. For instance asking who would be at risk from skin cancer found pictures of 3 surfers and a girl on the beach. Although a commercial demonstration, CYC was unable to find a picture of a Christmas tree given the word "tree", apparently a bug. It was very clear from the descriptions of the demonstrations that they were not intended for an academic audience, but for a commercial audience interested in applications.

Dr Pratt was also able to test CYC's abilities outside the framework of any commercial demonstration. He reports that Guha and he found it difficult to find their way around half a million axioms, and even to find the area of where the information they were looking for might be found. For instance, CYC did not know the term earth, but the term PlanetEarth. This was compounded by the strange choices of CYC's information about concepts. For instance, CYC knew the current prices of cars, but did not know their number of wheels, or their maximum velocity. Choosing to inform CYC of car prices seems strange, since a car's price is volatile, but its functionality or its number of wheels rarely vary and are more relevant to common-sense reasoning.

Later, Ramanathan Guha, the co-leader of the CYC project until 1994, told [Stipp 95] that *"We were killing ourselves trying to create a pale shadow of what had been promised"*. He thinks that CYC may prove useful in commercial applications such as data-mining, but that *"the goal of creating a system that would exhibit real common sense failed."* This disillusionment might reflect real failures in the basic approach, or naivety in its implementation.

- **CYC's Projected size**

In [Guha et al. 90b], an order of a hundred million (10^8) assertions was claimed to be the amount needed to represent most common sense knowledge. In [Lenat et al. 90], the figure was slightly lower at ten million (10^7). The justification for this figure is Marvin Minsky's statement that a child of eight would have acquired this many facts if one assumed that he learnt a new fact every 10 seconds of his life. After ten years, CYC actually contains only four hundred thousand (4.10^5) facts, although the number initially increased to two million. However, after removing redundant facts, the number slumped to its present figure.

However, the figure of 10^8 needs further justification. Assuming a K.B. of 50000 concepts, and 100 facts per concept, the number of five million facts appears adequate. Although the figure of 100 facts per concept might seem small, it should be noticed that there are many words in English that are rarely used, and that 2000 basic highly ambiguous words are sufficient for basic language abilities. Assuming 10 concepts per word still undershoots the number of concepts assumed. Thus the approach of encoding "common sense" symbolically need not be assumed infeasible.

- **CYC's Current size**

CYC's current size of 4.10^5 could be naively interpreted, as stating that if CYC had on average 10 employees for 10 years, each employee added about 16 assertions a day, for 250 working days a year, which would make each statement worth 62.5 dollars a statement. This is quite cheap per possible dependency: $\frac{25*10^6}{(4*10^5)^2} = 1.5625*10^{-4}$ dollars. But if CYC reached its projected size of 10^8 facts, the price becomes one and a half thousand billion dollars $(1.5 * 10^{12})$[12].

A.2.5.2 CYC's knowledge

Paradoxically, although the purpose of the project was to enter common sense knowledge, it seems that most attention was payed to the size of the project,

[12]only one thousand times more that the 6.25 billion dollars required for 10^8 facts at 62.5 dollars each

efficient algorithms, or a general ontology.

- **Anecdotal evidence**

Anecdotal evidence of this view comes from the examples given in papers and from Dr Pratt's report. For instance, Dr Pratt reports that CYC knew hand-gliding and touching toxic substances could have as consequences death, and it knew about food, but it did not know that people need food in order not to die. It knew the current prices of cars, but did not know the number of their wheels, or their maximum velocity. [Guha et al. 90b] provides similar examples, where CYC knows that seeing a movie at a cinema costs approximately five dollars in 1990. Although this information is common-sense, its choice reveals an apparent lack of understanding of the way the reasoning mechanisms of the K.B. can exploit the data.

Some examples are over-specialised: "hand-gliding causes death". Some are of little relevance: the current price of cars varies, but the number and function of their wheels or a car's average speed is critical[13] Similarly, knowing that seeing a film at a cinema costs five dollars is of less use than knowing that many people go to entertainment-centres[14] explains why a fire at these venues is very important to the fire-service, or why it is easy to lose someone there. To reiterate, the problem is that each of these pieces of data entered into the K.B. allows little further reasoning.

Although the dangers of over-general or over-specific facts were referred to in [Lenat et al. 90], the examples given were again quite superficial: *"One should not say 'People are younger than their parents', but 'things are younger than the things that brought them into existence'"*. For simplicity we shall refer by progeny to the former, and by progenitors to the latter[15]. The problem is that this statement should be inferred from the meaning of "to bring into existence". If a progenitor

[13]The average velocity of cars can be used to estimate the average length of a journey for instance to determine the time someone left home, or to work out that driving to Vladivostok from Lisbon would take a few weeks. The fact a car has wheels, which work by friction, plays a role in determining that cars, as well as sledges, slip on ice or oil. Many of these features can be inherited from a general class of wheeled vehicles, which includes trains, lorries, cars and bicycles.

[14]such as cinemas, theatres, circuses, zoos, sports stadia, concert-halls, perhaps even museums

[15]The general statement is not strictly true, as it assumes that progenitors still exist. But this discussion puts this objection to one side.

did something to cause a progeny to exist, then what it did occurred before the progeny existed. Since the progenitor performed the event, it existed before the latter did. Age states how long something existed, so if the progenitors still exist, they are older than the progeny.

The information entered into the K.B. appears therefore not so much to be common sense models, but simply facts that happen to be true: it appears to be an implicit rather an explicit model.

- **CYC methodology**

CYC's methodology is presented in [Lenat et al. 86]. All quotations in this section are taken from it, unless otherwise stated. The project's ten year span is divided into two stages.

The first stage involves a small group of AI researchers who must *"carefully represent 400 articles (about 1000 paragraphs worth of material) from a one-volume desk encyclopedia. These are chosen to span the encyclopedia and be as mutually distinct types of articles as possible"*. The content of each article must not only be encoded, but so must the implicit knowledge it referred to: the common-sense knowledge. If required, the knowledge representation is expanded to *"handle new information to be represented"*. Similarly, if knowledge previously encoded at a specific level turns out to be true at a more general level it is moved appropriately: for example, much of the information originally stated at irrigating's level was also true for transporting, so it was moved to transporting and inherited to irrigating. An additional task is to formulate a set of questions the system should be able to answer given the information it has been given. This is used to test the reasoning algorithms. The three principal results of this first stage are a relatively stable representation and ontology able to represent most statements that will be required in the full K.B.; a wide range of examples (400) of articles on most types of topic in the encyclopedia; and approximately 50% of the common-sense knowledge encoded.

The second stage involves *"a large cadre of lightly trained knowledge enterers, working together on a common, consistent version of the system. Each enterer will take* *a article, locate the already-represented similar article(s), and perform a machine-assisted 'copy & edit' procedure to produce a machine-understandable version of the new article."* This will result in *"enforced semantics"*, so that different knowledge enterers mean the same thing when using some statement of the language by *"copy*(ing) *existing structures rather than try*(ing)*to come up with ways to organise things on their own"*.

- **Discussion**

All the evidence from papers, demonstrations, and the methodology presented in [Guha et al. 90b] differ quite radically from the requirements for C.S.K.B.s.

Instead of models to be used by reasoning the idea appears to be to enter many commonly known facts, and hope that the resulting knowledge is sufficient for reasoning algorithms to provide common sense behaviour: to build an implicit model.

Virtually no work appears to have been invested into the critical issue of mutual dependencies. Indeed, the quality of the common sense information seems belittled: *"a large cadre of lightly trained knowledge enterers"* who *"copy existing structures rather than try*(ing)*to come up with ways to organise things on their own"*. Indeed, that the knowledge enterers did not participate in the writing of CYC's publications not only indicates their low status, but also demonstrates that the common sense models are not considered a research worthy subject.

A.2.6 Conclusion

Symbolic knowledge, in particular as given by the example of physics provides an attractive model for the knowledge base of a tool such as LOLITA. The knowledge is self-sufficient and provides means for it to be used for reasoning tasks such as prediction, which are required for the tool. Moreover its development over many centuries provides a great wealth of reasoning methods, and solutions to problems that may arise during development of the system. Finally, the symbolic paradigm itself has been the subject of much analysis. Thus using this paradigm has the advantage that the reasoning methods implemented may themselves be subjected to mathematical analysis, allowing possible flaws to be detected. Thus, all the arguments militate for using a symbolic paradigm as the basis of LOLITA.

A.3 Information extraction?

A two-part model for LOLITA was envisaged in A *(p. A-1)* where on the one hand processing of Natural Language text results in some form of knowledge, and on the other hand, this knowledge is processed to fulfil the tasks required of the system. This lead to two questions, the first of which has been discussed above. The second, which was what is meant by "extraction of knowledge from a text", will be discussed in this section.

"Extraction of knowledge from a text" has a mechanistic overtone, as if knowledge can be extracted simply by applying some simple transformations to the text. But this displays a lack of understanding of the nature of knowledge and what texts actually express. This section studies this issue, first by enquiring what we mean by "understanding a text", and then by discussing the interpretative process involved in text understanding.

A.3.1 Understanding

Considering the range of behaviours expected of someone who understood a text sheds light on what understanding a text involves. This is best illustrated through a thought experiment: Imagine that one has just explained something to someone. How would one test whether he had understood it or not?

The first step, would be to check if he recognised an instance of what was being explained. For instance, after having explained words expressing family relationships, such as mother, aunt, nephew, father-in-law, one could expect the person to recognise that his father-in-law and the father of his wife are the same person. However, in less trivial situations, this recognition is not enough to use the information usefully: one might recognise that ΠR^2 is a number multiplied by another number squared, or that it is the *"formula for the surface of a disk"*, without for that matter knowing how to use it.

A task similar to recognition is determining knowledge the subject knew previously which is similar to what has just been learnt. For instance one might show someone how something heavier than air such as paper can fly. Then one could ask him what it reminded him of, and expect as answer something like a seagull, or a plane. This appears to be one of the most fundamental aspects of understanding: if someone thinks a cat is more similar to a shopping mall than to a dog, one would not expect him to have understood what a cat is.

The next step of understanding would be to ask the person to reason with the newly acquired knowledge. His ability at this will depend on the application of various steps. Consider these in turn. The first is the ability to recognise the type of the problem being dealt with. This is indissociable from the ability to recognise which knowledge to apply to it. For instance, when confronted with an equation, he must detect its type (polynomial of what degree? differential equation? equation involving matrices ?), and then determine what method should be used to solve it. The second step is to be able to apply the knowledge, or to reason with it. In the case of the equations, this means being able to apply the method. To test this understanding one could give him an exercise to solve. At this simple stage, either he is able to solve it, or he is not.

Non trivial examples involve the judicious use of many different forms of reasoning, such as causal, temporal, or spatial reasoning. When this point is reached, the complexity of reasoning is such, that further knowledge is needed to determine the best reasoning path to choose. Such decisions demonstrate yet another level of understanding: an ability to link situations with the knowledge and reasoning method that should be used, at a much finer level of granularity. This kind of understanding can be tested for by more complicated exercises, which pay as much attention to how the problem was solved as to whether it was solved.

Finally, once an agent is capable of reasoning with the material, he can be asked why he reasons in the way he does: what he does, what he assumes. This final type of understanding considered here involves the ability to reason about new knowledge, and what assumptions it makes. This type of understanding helps determine whether a piece of knowledge can be used in some situation. For instance, if one has to decide whether to apply quantum, classical or relativistic mechanics to tiny fast objects, it is important to understand the underlying assumptions of each. Understanding these assumptions also helps determine which factors play a determining role in the solution, and which can be approximated or simplified, simplifying the problem to solve. For instance, understanding the nature of the Ptolemaic model of the solar system allowed Copernicus to replace it by the heliocentric model, greatly simplifying calculations. This type of understanding can also be tested for by setting exercises requiring approximations which must be justified.

These types of understanding have been placed in an order as understanding of each one usually requires a minimal understanding of the preceding one:

- what does the knowledge apply to, or recognition.
- how to use the knowledge requires recognition of the knowledge to use
- why the knowledge is so, is linked to understanding how it is used, and thus what it entails and does not entail.

However, this order is only useful at the earlier stages of understanding. Later the three types are more and more linked complementing each other. This second dimension of understanding is not linked to the type of knowledge, but to the degree of control it gives.[16] Thus understanding emerges as the ability to use the information, and to increase the degree of control one can exercise.

In order to see that the three types of knowledge complement each other, consider the following examples. The first type of knowledge, recognition of instances of the situation considered by the knowledge is linked to the second type, when dealing

[16]Note however that this is assuming the knowledge holds in the world. Otherwise, the degree of understanding depends on the extent to which correct predictions with respect to the knowledge can be made. What constitutes such a prediction is however outside the scope of this thesis.

with complex reasoning. For instance, recognising a particular configuration in a chess game evokes a lot of knowledge in a chess player, such as the possible and useful moves from this point. This type of understanding is tested exclusively in speed chess, but in games involving more time, it plays an important role in determining the strategy to be used. Similarly the third kind of knowledge can play an important role in determining the kinds of assumptions that can be made and which will simplify the problem, or why a particular type of reasoning need not be considered as a possibility to help in solving the problem. Again the first and third types of knowledge are closely linked since the first includes implicitly the assumptions underlying the model.

If this behaviour is considered more deeply, the degree to which the new knowledge is linked to the old emerges as a determining factor: recognition uses these links, since to be recognised an instantiation of some knowledge, the problem must be associated with the knowledge. Reasoning involves these links since knowledge associated with some problem is usually useful in solving it: If one knows that one has a nail in one's hand, and someone tells one that nails are sharp, one should be able to link the two facts in order to avoid puncturing oneself. Similarly, the quality of these links is important. In order to solve a problem it is preferable to use knowledge associated with the problem than any knowledge at all: the price of banana futures is irrelevant to working out how to catch a stray goat. Finally, the third type of knowledge involves an explicit discussion of these links.

Thus the links between the various parts of the agent's knowledge are a determining factor as to how the knowledge can be used for various tasks. These links are specifically tested for by the questions involving similarity, and the meta-level questions. Moreover, they are used for determining the path to follow in complex problems: the relevant knowledge. Finally they determine which forms of reasoning should be used. Thus they are fundamental to the processes tested by understanding, and should be considered in their own right.

To conclude, the important point is that the standard processes for testing understanding involve many different forms of knowledge processing. These different types of processing test in different ways how the new knowledge was related to the old. Moreover, they all participate in more complex tasks such as reasoning about a complex situation. Thus at the end of the day, the degree to which something has been understood depends on the power conferred to the agent on that thing. However, a deeper inquiry into the underlying processes involved show the importance of the linkage between the new and old knowledge. Thus, understanding can be tested in a weaker way by investigating these links.

A.3.2 Interpretation

Now that the nature of the agent's knowledge and understanding has been discussed, the notion of extracting information from a text will be reconsidered. As discussed previously, the agent's knowledge is assumed to be a symbolic model. Any new information will be added to this model, and should preserve its qualities of conciseness and consistency: if the model expands, it has a higher memory cost; if it becomes inconsistent, the predictions it will make cannot hold, i.e. cannot be useful. Hence, information obtained from a text will be expressed in terms of the agent's existing concepts, and not in terms of any universal concepts understood by all. Hence "extracting information from a text" hides a more active process, in which new concepts, probably different from those the text's author had in mind, are created[17]. Processing of text is strictly speaking an interpretative process.

Such a non-absolute viewpoint may seem shocking. Indeed, without the notion of absolute concepts shared by everybody, one could conclude that no one can understand anyone else. Daily experience shows that people, do in the whole, understand each other. So does the fact that humans evolved language despite the considerable expenditure of energy it requires to to learn and use: it must be worthwhile. However this does not mean that everybody must share a set of concepts they all understand in exactly the same way. Instead it implies that there is a sufficient degree of commonality between their concepts, and indeed their models of the world, for each to predict what the understanding of any statement

[17]This sheds further light on this thesis' treatment of hidden assumptions

they make will have on the other.

The lack of absolute concepts can be illustrated by an example. Consider for instance, the difference between someone who is an expert at hand-gliding and someone who isn't. The first will know much more about hand-gliders and so have a more precise idea of them than the second. This means that their concepts of hand-gliders differ. However, this does not stop them communicating. One can ask the other to carry the hand-glider.

This notion that expressing a statement also involves predicting what the behaviour of the other will be sheds an interesting light on the notion of understanding. Since prediction is to a varying degree an uncertain process, depending on what the agent knows, rather than on some absolute understanding, it may fail. In this case, the agent may mispredict the effect of his statement on the other. Hence, communication is often a two-way process where the hearer makes it clear to the speaker that his understanding of what the speaker said still makes sense – i.e. is consistent with his knowledge, and can be formulated with respect to his existing concepts.[18] The processes of testing the other's understanding is an extension of this process: it ensures that the other's understanding in terms of his concepts leads to the behaviour the speaker expects, in a given circumstance. This not only tests the hearer's formulation of the information in terms of his own concepts, but tests the whole of the hearer's knowledge, with respect to the circumstance. Hence, it is possible for agents to change the meaning of their concepts for them to accord with those of others. This allows a general convergence of concepts between various people, and in turn communication. Note however, that effort is required for this convergence. This means that convergence will only happen as long as it is useful. This includes successful communication, and successful practical application of the knowledge, such as correct predictions.

Just as it is unnecessary to assume absolute concepts, so is it unnecessary to assume an identical interpretative process among all agents. All that is required is that the desired behaviour is obtained. Again, the testing of the other's understanding

[18]For instance, the lack of such a response often gives people a disagreeable feeling of uncertainty... as when giving a lecture, or when talking to an answering machine

will play this role. If behaviour inconsistent with what has been said is detected, the statement may be reformulated: the hearer may not understand a particular formulation the way the speaker does. Reformulating the statement in another way exploits the possibility that the hearer may understand the statement in a different form.

Since neither absolute concepts nor identical interpretative processes are assumed, there are no constraints on the implementation of LOLITA. For instance, there is no need to understand the way in which people understand language before implementing the system. Thus there is no obligation to use psychology in the development of the tool. The only requirement is that the program's behaviour is consistent with what people would expect for a particular task.

Appendix B

Basic Representation: Detailed Issues

Sorts and quantification are argued to be intrinsic properties of the relations between concepts, rather than intrinsic properties of concepts themselves. This leads to them being represented on the arcs. A richness argument motivates the choice of multilevel quantification, and efficiency concerns determine how the representation should be implemented.

B.1 Quantification

The final quantification scheme presented in 5.3.2 *(p. 127)* is reached through a richness argument: Alternatives of increasing richness are proposed, each to be shown insufficient until the final quantification scheme is reached.

B.1.1 Attempt 1: Quantification on the node ?

Quantification on the node was discussed in the evaluation of LOLITA 92. The conclusions were that it led to weak cohesion (predicates on sets behave differently to predicates on sets' elements), and insufficient richness (problems with existentials (cannot express *"every mother has a child, and each of these children loves a toy"*) and quantification of events (cannot express *"every day I water an apple tree"*)).

Overall, quantification was shown to be intrinsic to relations and not to concepts.

B.1.2 Attempt 2: Simplified Quantification on the arc

The evaluation of LOLITA 92 uncovered various problems with expressing quantification on the nodes. The first was that quantification is a means of expressing how a relation applies to a concept. Thus the same concept may be involved in relations which apply to it differently by using different quantification schemes. The second point was that this fact not only applies to entities, but also to events for a representation that allows events to be the arguments of other events.

If a quantification is a means of expressing how a relation applies to a concept, it qualifies the relation more than the concept. This suggests the quantification should be placed on the arc rather than on the event. In this way the same concept may be involved in relations that apply to it differently because of the different quantification they carry.

Similarly, 4.9.6.1 *(p. 107)* showed that the choice of `object_` of an event need not only depend on the event's `subject_`, but can also depend on the choice of event instance. This can be fully accounted for if each arc is associated with a complete quantification scheme: a quantification states how the arc applies to its source and another states how it applies to its target. The arc therefore has "a quantification at each end". Quantificational dependency is restricted to the two quantifications of every arc.

B.1.2.1 Three basic quantifications

Three basic quantifications are assumed by this scheme: Individual, Universal and Existential quantification. These follow the definitions expressed when discussing quantification on the nodes, subject to the modifications below:

- **Individual quantification**

Individual quantification is used for events attached to instances of sets as before.

However it can also be used to attach events to sets. In particular this allows predicates on sets to be expressed in a manner which shows that they treat sets as constants. It avoids the additional rules required for quantification on the node, and does not break distributedness. However it also allows sets to participate as individuals in normal events, such as "to own". In this case, the set corresponds to the group of its elements, and it is the group as a whole which participates in the event rather than any particular element of it. For instance, a community of priests can be modelled as a set of priests. An event stating that *"the community owns the monastery"* would have a `subject_` arc to the set of priests with individual quantification:

E_0 : {I-`subject_`-I: [`community`]; I-`action_`-I: [`own`]; I-`object_`-I: [`monastery`]}

- **Existential quantification**

The main difference with existential quantification as done in LOLITA 92, is that existential quantification corresponds to unique existentials in $\mathcal{FOL}$. Thus an arc with quantification $\forall - \exists!$ states that every element l of the left node $\mathcal{L}$ is bound to one and only one corresponding element r of the right node $\mathcal{R}$ by the arc[1]. This means there is one r for each l, but there may be more than one l connected to each r.

E_0 : {$\forall$-`subject_`-I: [`John`]; $\forall$-`action_`-I: [`kick`]; $\forall$-`object_`-$\exists!$: [`ball`]}

Practically this allows statements such as *"John kicked every ball at least once"*. Each event only has one element of the set of `balls` as `object_`. This means that John kicked only kicked one ball at a time. There may however be different restrictions on the set of events, which distinguish two events with the same `subject_` and `object_`. Such a restriction could be a time event, allowing John to kick the same ball at different times.

However, the lack of an existential quantifier in the $\mathcal{FOL}$ sense, makes it impossible in the simplified scheme to represent statements such as *"Every farmer owns one or more donkeys"*. Here an $\mathcal{FOL}$ existential is used to refer to the donkeys each farmer owns. Unique existentials only allow *"every farmer owns a (one) donkey"*

[1]B.2 *(p. B-18)* provides $\exists!$'s definition, and an illustration of its usage in $\mathcal{FOL}$.

to be expressed.

Note that as with LOLITA 92, a relation referring to a node by an existential quantification refers to all the elements of the set represented by the node:

$$(\forall x \in \mathcal{A}\ \exists! y \in \mathcal{B}\ .\ r(x,y)) \wedge (\forall y \in \mathcal{B}\ \exists x \in \mathcal{A}\ .\ r(x,y))$$

B.1.2.2 Common element rule

Because quantification is expressed on the arc, a new question arises: if two events refer to elements of a node they qualify, then are the instantiations to which they refer related in any way? The common element rule (6.1.1 *(p. 156)*) states that the two events will always select the same instantiation.

B.1.2.3 Shorthand quantifications

Two other quantifications prove useful as shorthands.

- **Framed Universal**

Framed universal, F, is used to state that the relation expressed by an arc behaves like a bijective function. In other words for every element x of the arc's source set $\mathcal{S}$, there is one and only one element y of the arc's target set $\mathcal{T}$ bound to x by the arc, and for every element y of $\mathcal{T}$ there is one and only one element x of $\mathcal{S}$ such that x is bound to y by the arc. The arc corresponds to a one-to-one mapping. Since framed universal describes the behaviour of the whole arc, it makes no sense to have a framed universal at one end of an arc, and some other quantification at the other. Therefore the only variety of arcs involving framed universals have one on each end.

E_0: { F-`subject_`-F: [`Man`$_0$]; $\forall$-`action_`-I: [`love`]; $\forall$-`object_`-$\exists!$: [`Mother`$_1$] }

E_1: { $\forall$-`subject_`-$\exists!$: [`Mother`$_1$]; $\forall$-`action_`-I: [`mothers`]; F-`object_`-F: [`Man`$_0$] }

This example corresponds to *"every man loves his mother"*: There are as many `loving` (E_0) and `mothering` (E_1) events as there are men: each man is loved and cared for separately, but each mother may have more than one son: hence the (∀-`object_`-∃!: [`Mother`$_1$]) and (∀-**`subject_`**-∃!: [`Mother`$_1$]) arcs.

B.2 *(p. B-17)* shows that an arc of label l with framed universal quantification from s to d, corresponds to two arcs of label l from s to d, one with quantification $\forall - \exists!$ and the other with $\exists! - \forall$. Framed universals are therefore a shorthand for something that can be represented with the 3 basic quantifications. Using a shorthand proves very useful as framed universals are used very often, and testing for the existence of two arcs linking the same two nodes involves a search which albeit small, adds up if it must be done every time an arc of the network is traversed.

Other pairs of arcs involving the same nodes but with different quantifications exist, but they do not contribute together in restricting the relation between the nodes.

Consider the pair $\forall - I$ and $I - I$, where the lefthand node is the set of people in a company, the righthand node is an event, and the arcs are **`subject_`** arcs. For this example, it will be assumed that the set of people in a company is the company itself. The first arc states that each person participates as the subject of the event. The second arc states that the company also participates in the event. Such a situation could occur in a sentence such as *"The employees of the company, and the company itself would like to thank you for the long years of service you have given it."* This is an example where each arc contributes an additional subject, but does not combine with the others to restrict the relation. Such arcs are called partial arcs since they contribute part of the set of and event's subjects or objects.

Other pairs of arcs are contradictory, such as the pair $\forall - \forall$ and $\forall - \exists!$, and should never occur.

There are over 81 such possible pairs[2], so they will not all enumerated in this thesis.

- **Arbitrary quantification**

Arbitrary quantification, A, is used to refer to an instance of a set without building one explicitly. Unlike the other quantifications, for which all arcs refer to the same

[2]9 possible arcs involving single quantification: 3 possible quantifications at each end, and two ends = $3 * 3$

element of a set once it has been chosen, arcs involving an arbitrary quantification are independent from all others. This means that it is impossible to describe an instance of a set by two events using arbitrary quantification: an explicit instance node must be built for this purpose.

Arbitrary quantification proves useful as a compression device to avoid building a very large semantic network. Many events need to refer to an instance of a set, but this instance never becomes qualified by more than one event. This happens often in statements that use the values representation introduced in section 7.1 *(p. 247)*. But even sentences such as *"A dog howled"*, in the middle of a narrative can benefit from it. In this case the howling event's subject has arbitrary quantification with respect to its subject: the set of all dogs.

It might appear that arbitrary quantification would lead to an increase in the complexity of algorithms dealing with quantification. But it should be noticed that it can be hidden inside the abstract data type interface to the semantic net. To the rest of the system only explicit instances need exist, but to the semantic net access routines some of these are real in the sense they are explicitly written in the semantic net, and some are virtual in the sense that they are simulated and correspond to arbitrary quantification.

B.1.2.4 The use of quantification on the arcs

Extending the meaning of individual quantification and allowing different quantification for both the source and the target of the `subject_`, `object_`, and `action_` arcs allows many more forms of event to be expressed than allowed by the quantification on the node scheme. Indeed, quantification at both ends of the arcs changes the behaviour of the arcs with respect to the event they belong to: now they behave similarly to events describing some node. This section discusses this change.

- **Quantification at both ends**

Quantification on both ends of the arc results in a different view of `subject_`, `object_` and `action_` arcs. For instance,

$$E_0: \ \{\forall\text{-}\texttt{subject\_}\text{-}I\text{: } \texttt{Jane}; \forall\text{-}\texttt{action\_}\text{-}I\text{: } \texttt{buy}; F\text{-}\texttt{object\_}\text{-}F\text{: } \texttt{clothes}_1\}$$

states that E_0 is a set of events, each having as `subject_` Jane and as `action_` `buy`. Every element of $\texttt{clothes}_1$ is an `object_` of one such event. This means that Jane bought all the clothes of $\texttt{clothes}_1$ separately.

In $\mathcal{FOL}$, the event could be written:

$$\begin{aligned} &(\forall e \in E_0 \ \texttt{subject\_}(\texttt{Jane}, e) \wedge \texttt{action\_}(\texttt{buy}, e) \wedge \exists! c \in \texttt{clothes}_1 \ \texttt{object\_}(c, e)) \\ \wedge \ &(\forall c \in \texttt{clothes}_1 \ \exists! e \in E_0 \ \texttt{object\_}(c, e)) \end{aligned}$$

which is little different from the way an event behaves with respect to a node. For instance, if the "`object_`" arc were expressed as

$$E_1: \ \{F\text{-}\texttt{subject\_}\text{-}F\text{: } E_0 \ ; \forall\text{-}\texttt{action\_}\text{-}I\text{: } \texttt{obj}; F\text{-}\texttt{object\_}\text{-}F\text{: } \texttt{clothes}_1\}$$

the dependency between E_0 and $\texttt{clothes}_1$ would be written in $\mathcal{FOL}$ as:

$$\begin{aligned} &(\forall e \in E_0 \ \exists! c \in \texttt{clothes}_1 \ \texttt{obj}(c, e)) \\ \wedge \ &(\forall c \in \texttt{clothes}_1 \ \exists! e \in E_0 \ \texttt{obj}(c, e)) \end{aligned}$$

since the choices of c and e are mutually dependent in both cases, even if in the second case they are only dependent because they are mutually dependent on the choice of element of E_1.

- **Individual quantification**

Individual quantification on the arc introduced the notion of groups, where a set is referred to individually to refer to it as a group. There is a difference between all members of a group participating in an event, and the group itself participating. When the group participates, not all its members have to participate. This is illustrated by a football team being said to have won the UEFA cup. Not every member of the team need have directly participated in the game. For instance replacement players, the trainer and injured players were not on the football field during the various matches. Similarly, in a war one side wins, yet not all the people of that side need have fought.

To illustrate these subtle differences consider the three events:

- E_0 states that John squashed a group of ants. Not all the ants of the group need have been squashed and there is only one event, so this could be interpreted as accidental or as a half-hearted attempt.

  E_0: $\{I$-`subject_`-I: [`John`]; I-`action_`-I: [`squash`]; I-`object_`-F: [`ants`]$\}$

- E_1 states that John squashed every ant in the group of ants. None of the ants of the group survived it. Moreover there is one event per ant, so he squashed them each individually. This can only be interpreted as deliberate.

  E_1: $\{\forall$-`subject_`-I: [`John`]; $\forall$-`action_`-I: [`squash`]; F-`object_`-F: [`ants`]$\}$

- E_2 states that John squashed all the ants in one go. None escaped. There is only one event but it has all the ants as object. This could be either an unfortunate accident, or a deft deliberate action.

  E_2: $\{I$-`subject_`-I: [`John`]; I-`action_`-I: [`squash`]; I-`object_`-$\forall$: [`ants`]$\}$

  This event has the same meaning as an event which has as objects a set of partial arcs connected to each of the explicit instances of a set. For example if the group of ants had three members, Samantha, Sharon and Katherin, an event with `subject_` John `action_` squash, and three $I - I$ quantified partial `object_` arcs to Samantha, Sharon and Katherin would be equivalent to E_2. A similar example is *"The children buried all their toys in one go"*. Here all the toys are buried by the same event, for instance if one hole was dug, all the toys were placed in it, and then the hole was filled up.

Groups may have some of the properties of their members, but they need not have all of them, nor are they restricted to properties that their members can have individually. For instance, every person of a group may have a mother but the group itself cannot: mothering is a relation defined only for people. However people and groups of people may own an object, since owning is defined for both. Finally only groups may perform certain actions. For instance it is impossible to play a symphony solo, it must be done by a group of people, such as an orchestra.

Groups occur often in natural language, as illustrated by words expressing groups of objects: school of fish, a herd of cattle, pack of wolves, crowd of people. However not all references to groups need be so explicit. For instance, the sentence *"The* *children washed the cars"* may not imply either that every child participated in the washing, nor that every car was washed.

- **Partial arcs**

Two ended arc quantification also plays a role for partial arcs. For instance, *"The shareholders own the company"*:

E_0: $\{I$-`subject_`-$\forall$: `shareholders`$_1$; I-`action_`-I: `own`; I-`object_`-I: `company`$_1\}$

Here the individual quantification for the event E_0 and the universal quantification for `shareholders`$_1$ imply that E_0 takes as partial `subject_s` all the elements of `shareholders`$_1$. This idea is also used in:

E_1: $\{I$-`subject_`-$\forall$: `people`$_1$; I-`action_`-I: `build`; I-`object_`-$\forall$: `thing`$_1\}$

which states that a group of people `people`$_1$ built `thing`$_1$.

Note that partial arcs are not restricted in that they must have different targets. For instance, the $\oplus$ operator[3] which states that the sum of its partial `subject_s` is its `object_`, may take three identical `subject_s`:

E_0: $\{I$-`subject_`-I: [1, 1, 1]; I-`action_`-I: $\oplus$; I-`object_`-I: 3$\}$

B.1.2.5 Problems

Despite being a substantial improvement, two problems remain. The first was the lack of an existential quantifier in the $\mathcal{FOL}$ sense: This makes it impossible in the simplified scheme to represent statements such as *"Every farmer owns one or more donkeys"*.

The second problem comes from the inability to quantify over groups themselves. Thus it is possible, for instance, to state that a group of people built some things:

E_1: $\{I$-`subject_`-$\forall$: `people`$_1$; I-`action_`-I: `build`; I-`object_`-$\forall$: `thing`$_1\}$

But it is not possible to state *"Groups of people build things"*. This statement says that there are many events, each of which involve a group of people building a set

[3]For further information on this, see D.1.1.2 *(p. D-5)*

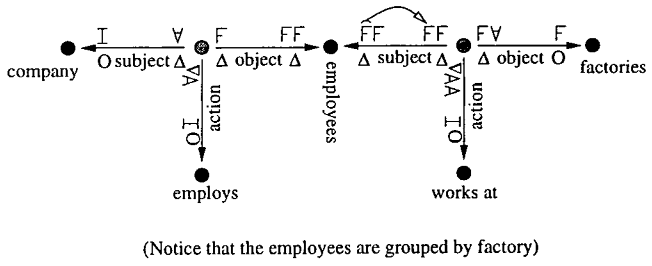


Figure B.1: A company employs employees at different factories

of things. Thus what is needed is some form of higher level quantification, where a concept can represent a set of sets of things: a set of groups of events. This is not possible within the representational framework defined so far.

B.1.3 Final Quantification Scheme

The full scheme extends on the idea of group introduced in the second attempt. In the second attempt it was possible to refer to a set by an individual quantification thus treating it as a group. However, there was no means of expressing statements about groups of groups. In other words the grouping was limited to depth 1 . In practice however it is useful to be able to make statements about groups of groups. For instance, one may wish to group a company's workforce into groups working at each of its factories. In order to do this, one needs to be able to refer to a group (the workforce) of groups (the employees of each factory). Furthermore, one needs to be able to refer to each group of employees and to the employees themselves in order to partition them in this way:[4] figure B.1 *(p. B-10)*.

Similarly, if one wishes to be able to express for a group of children, all the modalities that one was able to for John squashing ants, one needs to be able to use quantification to refer to each group of ants being squashed, and the ants of each such group: *"Every child squashed a group of ants"* (E_0), *"Every child squashed every one of his[5] ants individually"* (E_1) and *"Every child squashed all of his ants"* (E_2):

[4]since the representation of location has not been described, the example is simplified to use a works_at action. In the network this would however be expressed using the representation of location.

[5]his refers to the ants of the group he squashed

E_0: $\{F$-subject_-F: [child$_1$]; $\forall$-action_-I: [squash]; F-object_-F: [ant_group$_1$]$\}$

E_1: $\{F\forall$-subject_-F: [child$_1$]; $\forall\forall$-action_-I: [squash];
FF-object_-FF: [ant_group$_1$]$\}$

E_2: $\{F$-subject_-F: [child$_1$]; $\forall$-action_-I: [squash]; F-object_-$F\forall$: [ant_group$_1$]$\}$

In order to express these statements a many levelled quantification is introduced. This can not only refer to the elements of sets, but also the elements of sets of sets, the elements of sets of sets of sets, and beyond. This is achieved by extending the quantification to a string of quantification symbols $s_1 s_2 \cdots s_i$, where the jth symbol s_j refers to the jth level of elements of the set referred to. The first such level of a set $\mathcal{S}$ is its elements, the second is the elements of its elements and so on. Because the choice of each symbol is free between $\forall$, $\exists!$, I and the shorthands F and A, complex relations can be built. Each end of each arc is associated with the string of symbols expressing the way in which the arc relates to the node it is connected to at that end. In all other respects the scheme is the same as the second attempt.

Although in theory an infinite number of quantification levels can be built, in practice only three appear to be needed to express most statements in natural language, discounting artificially contrived examples. A maximal number of five appears necessary to express the most complex of template events.

B.1.3.1 Quantification Dependencies

A problem with the new scheme is quantificational dependencies between the $\forall$ and $\exists!$ quantifications. For instance, can the choice of an element at one level of a set $\mathcal{S}$ be determined by that of another at another level of $\mathcal{S}$? This situation corresponds to dependencies between quantifications of a same string. Such dependencies are of little use since they do not express dependency of elements on other sets. Thus quantification dependency is only allowed between symbols on either end of an arc.

Usually quantification dependencies occur at the same level: if an arc links two sets $\mathcal{L}$ and $\mathcal{R}$, dependencies will be between the xth level of set $\mathcal{L}$ and that of set $\mathcal{R}$ for any x. For instance, *"every child squashed every one of his group of ants individually"* can be expressed as the event E_0:

$$E_0\text{: } \{F\forall\text{-}\mathtt{subject\_}F\text{: } \mathtt{children}\text{; } \forall\forall\text{-}\mathtt{action\_}I\text{: } \mathtt{squash}\text{;}$$
$$FF\text{-}\mathtt{object\_}FF\text{: } \mathtt{ant\_group}\}$$

Here E_0 is a set of set of events. Each instance E'_0 of E_0 is a set of events which has as subject_ a particular child of the children set. Because of the $F - F$ quantification, there is one such set per child. Because the event end of the quantification is $F\forall$, each instance of E'_0 is connected to the subject_ arc, rather than the group of events E'_0 being connected itself. The quantification dependency is therefore limited to the 1st level of the sets E_0 and children. In $\mathcal{FOPL}$, this could be expressed as:

$$\forall e \in E_0 \ \exists! c \in \mathtt{children} \ \forall e' \in e \, . \, \mathtt{subject\_}(c, e')$$
$$\wedge \ \forall c \in \mathtt{children} \ \exists! e \in E_0 \ \forall e' \in e \, . \, \mathtt{subject\_}(c, e')$$

Similarly, the action_ is connected to every instance of E'_0 in light of the $\forall\forall$ quantification. Finally, the object_ arc states that there is a group of ants for every instance E'_0 of E_0. It also states that there is an ant for every event of every such set E'_0 which is its object. There are therefore two quantification dependencies between E_0 and ant_group, one at level 1 and the other at level 2. In $\mathcal{FOPL}$, this could be expressed as:

$$(\forall e \in E_0 \ \exists! a \in \mathtt{ant\_group}) \ (\forall e' \in e \ \exists! a' \in a) \, . \, \mathtt{object\_}(a', e')$$
$$\wedge \ (\forall a \in \mathtt{ant\_group} \ \exists! e \in E_0) \ (\forall a' \in a \ \exists! e' \in e) \, . \, \mathtt{object\_}(a', e')$$

where the e and a are quantificationally dependent, as are e' and a'. This situation where quantificational dependencies are restricted to the same levels is referred to by the expression "independent levels of quantification". It allows different levels of quantification to be treated independently and is of particular importance in the treatment of inheritance over inst_ events (see 6.4 *(p. 164)*)[6]

Sometimes however, it is useful to represent quantificational dependencies between levels. In many cases, such as $\forall F - F$ or $\forall - \forall\exists!$, the dependency is unambiguous and shall therefore not be expressed explicitly. In other cases however, such as $\forall\forall - \exists!$ it is unclear: does the $\exists!$ depend on the first, the second, or both $\forall$s? In such cases, the dependency is expressed explicitly as follows: $\boxed{\forall}\forall - \boxed{\exists!}$, $\forall\boxed{\forall} - \boxed{\exists!}$, and $\boxed{\forall\forall} - \boxed{\exists!}$, respectively. In figures, they are expressed by arrows.[7]

[6]The fact that $Is_1s_2 \cdots s_j \equiv s_1s_2 \cdots s_j$ is used extensively

[7]Of course, in the computer implementation, the dependency is always expressed explicitly.

B.1.3.2 Sets of sets of ...

With the full quantification scheme, sets of sets, sets of sets of sets, and so on have been introduced. These sets must be built by powerset relations from other sets consisting of single instantiations of concepts. Such a set would be the set of dogs, containing only dogs as instantiations, and no groupings of dogs. Section 5.4 *(p. 135)* discusses how such simple sets are built.

Sets of sets can be built using the `powerset_` or the `infinite_powerset` events[8]. The `powerset_` event states that its `object_` is a powerset of its `subject_`. This means that the `object_` set contains subsets of the `subject_` set. The `object_` of an `infinite_powerset` event includes not only subsets of its `subject_`, but also sets of sets of the elements of the `subject_`, sets of sets of sets of ..., and even sets including elements of the `subject_`and sets of elements of the `subject_`, and so on to any depth.

B.1.3.3 Quantification and Intensionality

The interaction of quantification and empty sets raises some questions about the nature of quantification. Indeed, if a set is empty, how can one make statements about its contents? The problems stem from the extensional viewpoint: that the set is equivalent to its elements. An alternative viewpoint is intensional: the set is equivalent to its definition: see 5.4 *(p. 135)*. Whether or not the set happens to have any elements, one can talk about the elements it does or would have. Thus equality in SemNet is intensional.

From the intensional viewpoint, quantification does not state that there are elements which are taken from the quantified-over set, but that if there were elements in that set, they would be chosen in the manner stated by the quantification. Thus, one can state *"Every flying pig owns a German"*:

[8]These are defined in D.6.3.4 *(p. D-81)*

E_0: { F-subject_-F: [flying_pig]; $\forall$-action_-I: [own]; F-object_-F: [German$_1$] }

E_1: { I-subject_-I: [flying_pig]; I-action_-I: [size_]; I-object_-I: [0] }

where German$_1$ is the set of Germans owned by flying pigs.

One problem remains: What happens if one has a set of sets, where all the inner sets are empty? This might occur, for instance, if one defined the dispossessed as the set of people who own nothing:

E_0: { F-subject_-F: [Dispossessed]; $\forall$-action_-I: [own];
F-object_-$F\forall$: [Dispossessed_Possessions] }

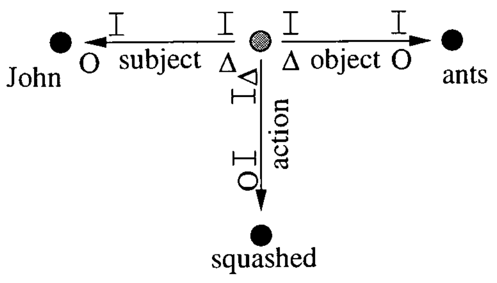
E_1: { F-subject_-F: [Dispossessed_Possessions]; $\forall$-action_-I: [size_];
$\forall$-object_-I: [0] }

In set theory, there is only one empty set, so all elements of Dispossessed_Possessions would be the same empty set. This implies that there is only one element of Dispossessed_Possessions, hence only one element in E_0, and hence only one Dispossessed. Furthermore, it assumes that a set can have $\emptyset$ as a element.

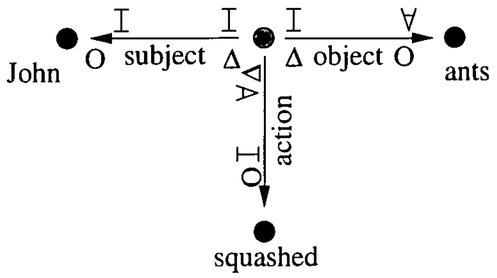
From the intensional viewpoint, all of John's possessions is a different concept to all of Jack's, even if both turn out to own nothing. Rather than being considered an intrinsic feature of sets, cardinality is is considered incidental: a property like any other. Because equality is intensional, two concepts are only equal in SemNet if they have the same definition. I.e. SemNet does not have the rule that any two empty sets are equivalent: a concept can be a set of many 'empty groups', where each such 'empty group' has a type given by its definition (in particular, the type of the arguments of the events defining it and their quantification). In practice this means that E_0 and E_1 do not imply that there is only one dispossessed.

B.1.3.4 Logical existential quantification

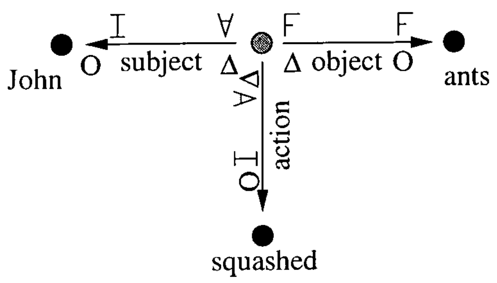
Previously, it was stated that existential quantification in the representation was equivalent to unique existential ($\exists!$) quantification in logic. Until now there was no way of expressing quantification corresponding to existential ($\exists$) quantification in logic: one or more. This can be achieved in the full quantification scheme by using $\exists!\forall$ quantification. Thus a $\forall - \exists!\forall$ arc states that for every instance l on its



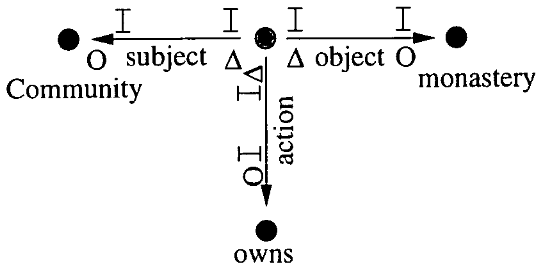
John squashed the (group of) ants



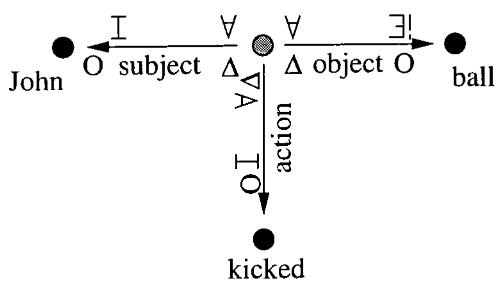
John squashed all the ants



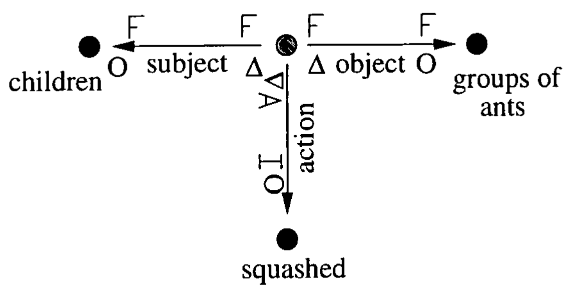
John squashed every ant (individually)



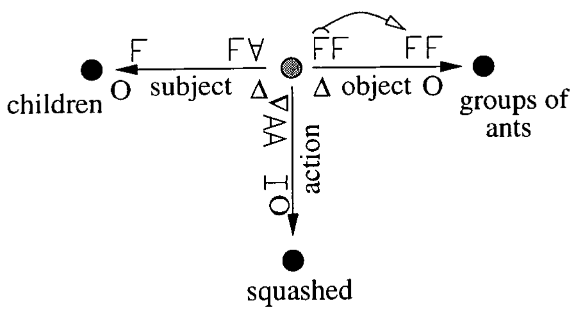
The community owns the monastery



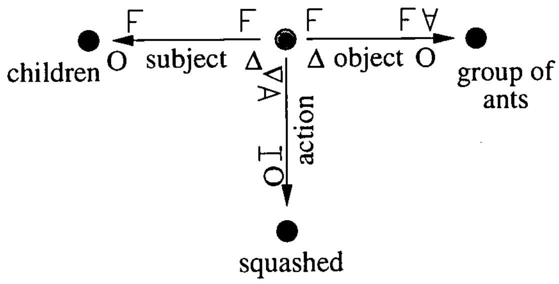
John kicked every ball (at least once)



Every child squashed a group of ants

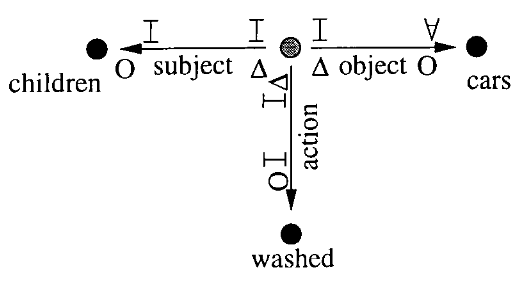


Every child squashed every one of his ants individually

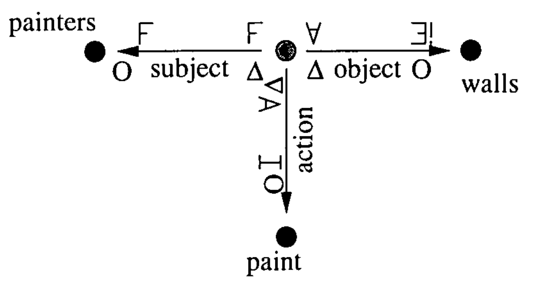


Every child squashed all of his ants

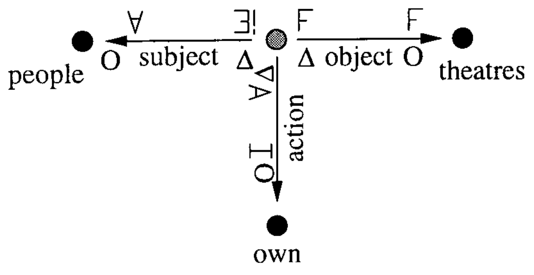
Figure B.2: Quantification Examples



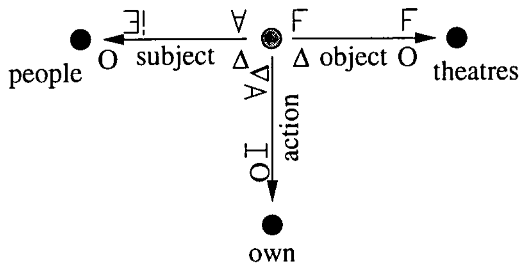
The (team of) children washed all the cars



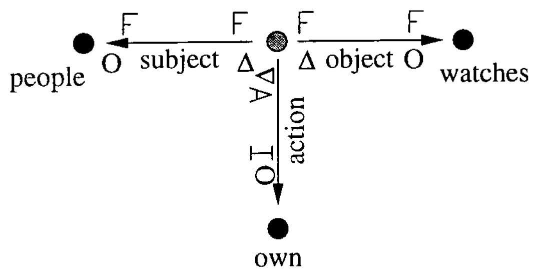
Every painter paints a wall, but the same wall can be painted by more than one of them.



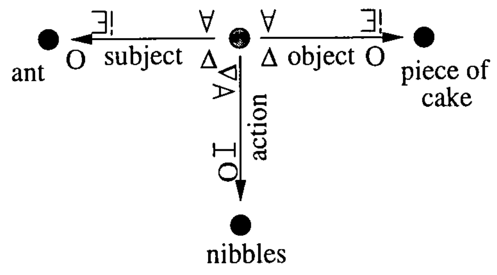
All the people who own, individually or as a group, one theatre.



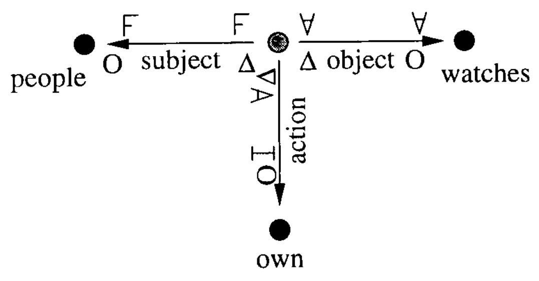
Every theatre is owned by a person, and each person can own more than one theatre.



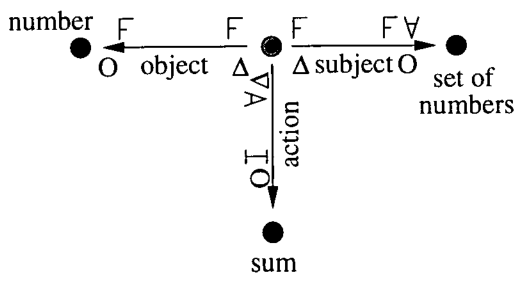
Every person owns a watch, and each watch is owned by a person.



Every ant nibbles one or more pieces of cake, and each piece of cake is nibbled by one or more ants.



Every person owns every watch, and each watch is owned by every person.



Every set of numbers has a unique sum.

Figure B.3: Quantification Examples

left, there is one set r on its right, such that it relates l to all the elements of r. In this manner, statements such as *"every farmer owns at least one donkey"* can be expressed:

E_0: $\{F\forall$-`subject_`-F: `farmers`; $\forall\forall$-`action_`-I: `own`;

FF-`object_`-FF: `donkeys`$\}$

E_1: $\{$ F-`subject_`-F: `donkeys`; $\forall$-`action_`-I: `size_`; $\forall$-`object_`-$\exists!$: `one_more'`$\}$[9]

This states that for every farmer there is a (non-empty) set of donkeys that he owns. The $\forall - \exists!$ is hidden in the `subject_`'s $F\forall - F$ quantification.

B.2 Proof that F-F events are bijective

First recall the full form of the F-F quantification for an event r, where the subject_ is the set $\mathcal{X}$ and the object_ is the set $\mathcal{Y}$.

$$[\,(\forall x\in\mathcal{X}\ \exists! y\in\mathcal{Y}\ :\ r(x,y)\,)\ \wedge\ (\forall y\in\mathcal{Y}\ \exists x\in\mathcal{X}\ :\ r(x,y)\,)\,]\ \wedge$$
$$[\,(\forall y\in\mathcal{Y}\ \exists! x\in\mathcal{X}\ :\ r(x,y)\,)\ \wedge\ (\forall x\in\mathcal{X}\ \exists y\in\mathcal{Y}\ :\ r(x,y)\,)\,]$$

This can be simplified to $A \wedge B$ where:

$$\begin{aligned} A &:= \forall x\in\mathcal{X}\ \exists! y\in\mathcal{Y}\ :\ r(x,y) \\ B &:= \forall y\in\mathcal{Y}\ \exists! x\in\mathcal{X}\ :\ r(x,y) \end{aligned}$$

Recall the requirements for a relation r between the elements of two sets $\mathcal{X}$ and $\mathcal{Y}$ to be bijective:

$$\begin{aligned} Bij(r) &:= Func(r) \wedge Surj(r) \wedge Inj(r) \qquad \text{(B.1)} \\ Func(r) &:= [\,\forall x\in\mathcal{X}\ \forall y, y'\in\mathcal{Y}\ :\ r(x,y) \wedge r(x,y') \Rightarrow y = y'] \wedge \\ &\quad\ [\,\forall x\in\mathcal{X}\ \exists y\in\mathcal{Y}\ :\ r(x,y)] \\ Surj(r) &:= \forall y\in\mathcal{Y}\ \exists x\in\mathcal{X}\ :\ r(x,y) \end{aligned}$$

[9]`one_more'` is a subset of the set of all numbers equal to one or more: not all numbers greater or equal to one are the number of donkeys someone owns as the $\exists!$ would imply.

$$Inj(r) \;:=\; \forall x, x' \in \mathcal{X}\ \forall y, y' \in \mathcal{Y} \;:\; r(x,y) \wedge r(x',y') \wedge x \neq x' \Rightarrow y \neq y'$$

Recall the definition of $\exists!$:

$$\exists! x \in \mathcal{X} \;:\; P(x) \;:=\; \exists x \in \mathcal{X}\ [\, P(x) \wedge \forall x' \in \mathcal{X} \;:\; P(x') \Rightarrow x = x'] \qquad \text{(B.2)}$$

Consider A:

$$\begin{array}{lll} & \forall x \in \mathcal{X}\ \exists! y \in \mathcal{Y} \;:\; r(x,y) & \text{(A)} \\ \Leftrightarrow & \forall x \in \mathcal{X}\ \exists y \in \mathcal{Y}\ [r(x,y) \wedge (\forall y' \in \mathcal{Y} \;:\; r(x,y') \Rightarrow y = y')] & \text{(using (B.2))} \\ \Leftrightarrow & \forall x \in \mathcal{X}\ [\, \exists y \in \mathcal{Y} \;:\; r(x,y) & \\ & \qquad \wedge\ (\forall y, y' \in \mathcal{Y} \;:\; r(x,y) \wedge r(x,y') \Rightarrow y = y')] & \\ \Leftrightarrow & Func(r) & \end{array}$$

A is therefore equivalent to the requirement for r to be a function.

Consider B:

$$\begin{array}{lll} & \forall y \in \mathcal{Y}\ \exists! x \in \mathcal{X} \;:\; r(x,y) & (B) \\ \Leftrightarrow & \forall y \in \mathcal{Y} \;:\; [\, \exists x \in \mathcal{X} \;:\; r(x,y)] \wedge & \\ & \qquad [\, \forall x, x' \in \mathcal{X} \;:\; r(x,y) \wedge\ r(x',y) \Rightarrow x = x'] & \text{using (B.2)} \\ \Leftrightarrow & \forall y, y' \in \mathcal{Y} \;:\; [\, \exists x \in \mathcal{X} \;:\; r(x,y)] \wedge & \\ & \qquad [\, \forall x, x' \in \mathcal{X} \;:\; r(x,y) \wedge r(x',y') \wedge y = y' \Rightarrow x = x'] & \\ \Leftrightarrow & [\, \forall y \in \mathcal{Y}\ \exists x \in \mathcal{X} \;:\; r(x,y)] \wedge & \text{by reordering and} \\ & [\, \forall y, y' \in \mathcal{Y}\ \forall x, x' \in \mathcal{X} \;:\; r(x,y) \wedge r(x',y') \wedge x \neq x' \Rightarrow y \neq y'] & (a \Rightarrow b \equiv \neg a \vee b) \\ \Leftrightarrow & Surj(r) \wedge Inj(r) & \end{array}$$

B is therefore equivalent to the requirement for r to be surjective and injective.

By (B.1), $A \wedge B$ is bijective, so r is bijective.

B.3 Implementation of the basic representation

Other sections detail how the representation works in theory. In practice it must also be efficient for search. This section outlines some features of an efficient implementation.

B.3.1 Events

In a semantic network, search proceeds through the links. As described in 3.4.1.2 *(p. 37)*, determinism of search is a critical factor. Many forms of search involve finding a particular type of event connected to a node. For instance, searching up the inheritance hierarchy involves finding any `inst_` or `spec_` event connected to the particular node. However, every node is connected to a myriad of events. Since the type of an event is given by the target of its `action_`, for each event connected to a node, the search must proceed through two arcs to find its type. For n events connected to a node, $2n$ arcs must be traversed. This is inefficient, since only the events searched for need be accessed: in a real system, the large network means that large parts of it will be paged out of memory. Each traversal of an arc is associated with a likelihood of time consuming swapping. Clever organisation of the network in memory cannot avoid this, since locality in a graph is impossible to map perfectly in the general case to locality in a linear memory scheme such as computer memory.

The solution lies in grouping `subject_of` and `object_of` arcs of each node by the type of event that they lead to[10]. All the arcs connected to a node are expressed on the node, so such a change requires grouping the arcs by the action of the event they are connected to for the `subject_of` and `object_of` arcs. For the other four basic arcs, `subject_`, `action_`, `object_` and `action_of` this is not necessary since it is only the events nodes are connected to that are of import to search. A naive implementation illustrates:

[10]a little like frames are associated with slots on them

```
> data NodeArcs
>    = NodeArcs [Arc] [Arc] [Arc] -- Subject, Action, Object arcs
>               [(Node,[Arc])]    -- Subject_of arc
>               [Arc]             -- Action_of arc
>               [(Node,[Arc])]    -- Object_of arc
```

The $[\cdots]$ denotes a list, and (a, b) denotes the tuple formed of first element a and second element b. The lists of arcs correspond to the arcs, sorts and quantification with arc type given by their position in the data structure `NodeArcs`. The `Nodes` as first elements of the tuples are the actions of the events the node is connected to. Since the second element of the tuple is a list of `Arcs`, each tuple groups together all arcs connected to an event with the type given by the first element.

This solution incurs a cost in the size of the network, but given the improvement in search it provides, overall it is advantageous.

It is because of this scheme that `spec_` and `inst_` events have different action types `spec_`, and `inst_`: `spec_` could be written in terms of an `inst_` event as follows:

(E_0,R): {$\Delta\forall$-`subject_`-IO: X; $\Delta\forall$-`action_`-IO: `inst_`; ΔF-`object_`-$F\Delta$: Y}
Maintaining the distinction is however beneficial for certain processing such as search. Maintaining the distinction requires however that the alternative `inst_` form for `spec_` is always normalised to the `spec_` form. However in some forms of processing which rely on quantification, `spec_` is converted back to its base `inst_` form. This occurs for instance in the preprocessing of conversion to antonym.

Indeed, `inst_` itself is unnecessary if arbitrary quantification is implemented natively. If originally,

(E_0,R): {ΔI-`subject_`-IO: X; ΔI-`action_`-IO: `inst_`; ΔI-`object_`-$I\Delta$: Y}
then, this can be rewritten
(E_0,R): {ΔI-`subject_`-IO: X; ΔI-`action_`-IO: `spec_`; ΔI-`object_`-$I\Delta$: Y}
(E_1,R): {ΔA-`subject_`-IO: Y; $\Delta\forall$-`action_`-IO: `size_`; $\Delta\forall$-`object_`-AO: `1`}
where E_1 has many other `subject_` arcs of the form ΔA-`subject_`-IO. However, using an `inst_` event reduces the search space various searches down the hierarchy need consider, for instance semantic integration 6.7 *(p. 193)*.

B.3.2 Templates and `action_of` arcs

Determining the template event associated with a particular action or event type is a common task. For instance, it is necessary when building new events in the network to ensure that they are built as instances of their template. Similarly, it is used to check that the subject and object they are attached to is legal for their literal meanings (type-checking: 6.6 *(p. 188)*). Finally, important features such as pre- and postconditions (see 7.2.3 *(p. 264)*) are expressed on the templates. Thus, for many reasons, an efficient method of accessing an event's template is desirable.

It is very rare to traverse an `action_of` arc, as it is very rare to wish to find all the events with a particular action. Moreover this same information can be gained by searching down the inheritance hierarchy from the template event for all its instances, and those of its specialisations which have that action, and their instances. Instead, the `action_of` arc can be used to point only to the template event. This makes finding the template of any event, whether or not it is yet connected into the hierarchy simply a matter of traversing an `action_` arc to the relevant action, then traversing the `action_of` arc to the template event.

B.3.3 Families

In certain regions, the topology of the inheritance hierarchy is more tree-like than graph-like, in that one concept is partitioned into many others mainly, if not only, by specialised events. For instance, the node human is partitioned into many different kinds – owners, lecturers, managers, Greeks, ... –, yet most of these partitions are obtained through the use of events which apply only to humans. This topology reflects that a region of knowledge is of particular interest to LOLITA. But it also states that the top concept of the region has properties that distinguish it very sharply from the rest of the network. Much reasoning involved in the region it dominates will depend on these features in particular. Thus, a lot of inheritance will search up to it to check these features. For instance, if one wished to know whether some concept could be the `subject_` of an owning event, one must check whether it has all the definitional features of owners. This is simple, if it is a specialisation of humans, but if it is not, the search must consider all the definitional events it can inherit: it will proceed up the hierarchy until all the paths to `typeless` have been investigated.

Instead of searching upwards to determine whether some concept is a specialisation of some important often referred to concept, families can be used. In principle the idea is to provide a direct link between every concept and the concept that dominates it. For instance all concepts which refer to some kind of person have "human" as family.

This could be implemented by a `spec_` or `inst_` event from the concept **human** to all its elements. However this would cause a few problems. It would increase the network size dramatically because almost each concept would be assigned an additional `spec_` or `inst_` event; it would not by itself solve the problem of searching upwards to `typeless` if a concept being type-checked were not of the correct type, as there is no difference between its normal and family `spec_` and `inst_` events; and it would lower topological determinism if one wished to determine the least restricted of the specialisations of the dominating concept, as the dominating concept would not be connected only to them via `spec_` or `inst_` events, but also to all the other concepts it is the generalisation of.

Instead it is implemented as a special "family" control which takes a value corresponding to the dominating concept. This does not increase the network size significantly, nor does it decrease topological determinism, and its difference to normal `spec_` or `inst_` events prevents a search upwards. However, it does break uniqueness, so it requires additional machinery to ensure that it corresponds to the state of the hierarchy. In practice this is not a big problem since it is rare for concepts to change dominating concept.

The use of family controls reduces the need for search dramatically. Not only can processing jump directly from concepts to their dominating concepts, but it can also use the family values themselves: the set of dominating concepts, and their corresponding family values form a (small) inheritance graph. This means that algorithms such as the frequently used type-checker can determine that a concept cannot be involved in a particular event. For instance, a horse cannot own anything since `owners` has family type "human", and `horses` have type "(non-human) animal", and these two types do not intersect. Thus the search up to `typeless` has been avoided. However, not all search through the network is eliminated since the reasons for rejecting an event may involve information of finer granularity than that provided by family controls. For instance, an event may require as `subject_` "rich humans", whereas a potential `subject_` John is human but not rich. Overall, the benefits in search reduction that family controls provide for many algorithms justify the cost of maintaining them.

For non-literal concepts, the family value is the most specific value which is, nevertheless, the ancestor of the possible dominating concepts. For instance, Mickey mouse, Dumbo and Caligula's horse would all take as value "animalOrHuman", since even if one is not sure whether it is human or animal, one does know that it is one of them. Similarly if a concept changes nature over time, the most specific value which is the ancestor of the possible dominating concepts is chosen. Thus if a concept appears too general to be involved in some event, the type-checking mechanism must resort to using a search for its defining properties.

B.4 Sorts

B.4.1 The need for sorts at each end of each arc

The need for sorts at each end of each arc is demonstrated by a richness argument. A succession of simple schemes are presented, each to be shown insufficient. The manner in which each is insufficient is corrected until the final scheme is reached.

B.4.1.1 A Control on Events

The most obvious representation scheme would be to associate a new control with events. This would state whether or not the event is restricting the sense-domain of the concept to which it is attached. When an event is restricting the scope of a concept, it is defining it, and the control should be **definitional**; whereas when a concept is stating some fact about the concept, it is making an observation about the concept, and the control should be **observational**. Thus for instance if one were trying to define the concept green plants, the event *"is green"* would be attached to the node representing *"green plants"* and given a definitional control. On the other hand, the event *"Jane likes green plants"* would have an observational control.

Once a concept has been defined, observational events may be used to express facts about its elements. For instance, we may wish to state that all books are made of paper. This is not part of books' definition, since a book made of animal skin may at first seem strange, but not completely alien to the idea of a book. This does not however mean that only books are made of paper. Therefore the concept of those things made of paper and are books is a subset of the concept of things made of paper. Similarly observed statements can be made about the concepts themselves: for instance the number of 1 dollar notes in circulation is not part of a dollar's definition, but is an observed statement which would be expressed as the size of a set.

B.4.1.2 Problems with Sorts on Events: Transitive Events

The scheme of associating a definitional control with events appears to work well with intransitive events above. However, it needs to also work with transitive events, where two concepts are involved in the event. For instance *"Every one of Lolita's husbands had a fast car"*:

(E_0, Δ): { $\forall$-`subject_`-I: `Lolita`; $\forall$-`action_`-I: `wives`; F-`object_`-F: `Lolita_Husbands`}
(E_1, Δ): { F-`subject_`-F: `Lolita_Husbands`; $\forall$-`action_`-I: `own`; F-`object_`-F: `Fast_Cars1`}

E_0 has a definitional control since it defines the concept `Lolita_Husbands`, LOLITA's husbands. Similarly, E_1 has a definitional control since it defines the set of fast cars, each of LOLITA's husbands owned.

However consider the concept "LOLITA": Because E_0 is definitional, it also defines `Lolita`. Similarly, E_1 also defines `Lolita_Husbands`. This is not what was intended.

B.4.1.3 Associating a sort to each arc

Since each concept participating in an event may be or may not be restricted by it, two options are available. The first involves assigning a definitional control to the concepts themselves. This removes the constraint that all concepts participating in an event need either be restricted or not by it. However it results in that all events attached to a concept are either definitional or observational. As shown in the examples above, we want to be able to attach both definitional and observational events to the same concept. The second alternative involves associating something similar to a control to the arcs: a sort. Now that each arc may have a different sort, every node be it an entity or an event may be associated with definitional and observational statements.

Sorts are a form of information expressed on the arcs. They state the role each arc is playing with respect to its source and target nodes. Thus they are used to express whether an arc is restricting a node it is connected to, or stating something about it. In the first case the arc has a "definitional" sort with respect to that node, in the second it has an "observational" sort.

B.4.1.4 Problems with associating only one sort to each arc

Because SemNet is propositional, one event s can have another t as `subject_` or `object_`. This means that the source event (s) may be definitional or observational with respect to the target event (t). Clearly, this is the case, since one can say *"Every time the moon rises, John hides under his bed"*, where *"Every time the moon rises"* defines the set of events involving John hiding under his bed. Similarly, the source event can be observational with respect to the target event: *"John believes Jack lives on Mars"* Here John's belief does not restrict the set of events involving Jack living on Mars.

This raises the question: are the event's own arcs definitional or observational with respect to it? It might seem obvious that they are always definitional with respect to it: surely an event is defined by its arcs, which specify the various parts of the relation it expresses. However, this is not the case, as the example *"The only thing John ever did was lie"*. This means that the only event which has as `subject_` John is a lying event. That is to say, that the set of events defined by the `subject_` John is observed to have as `action_` lie:

E_2: { ΔI-`subject_`-IO: `John`; OI-`action_`-IO: `lie`}

If both arcs were definitional, the event would only say that John lied.

B.4.2 Why sorts?

Indeed, if an observational arc with respect to a concept indicates that all instantiations of the concept form a subset of the set defined by the observed arc, why should this not be expressed directly using sets and `spec_` relations?

In an example of such a network, all explicitly expressed events would be definitional, and all inherited ones would be observational. In this way, all events are definitional with respect to the nodes they are connected to, but not with respect to their subsets. This means that if a concept is defined by many events, it must be connected to all of them. If it is involved in many observational events, they all define supersets of it and should therefore be inherited.

This scheme results in a lot of repetition. Consider the conceptual hierarchy. For each concept defined, all the definitional events of its ancestors must be repeated. Thus the concept "man" is not only connected to the events which distinguish men from other animals, but is also connected to the events defining animals, living beings... and every other concept up to `typeless`. As a result, it is difficult to determine which concepts are of particular relevance to men, rather than to animals. This type of information is needed for instance for semantic distance.

Observed events are determined by searching up the inheritance hierarchy and virtually inheriting down all events which are not explicitly connected to the concept.

Virtually inheriting events means that although processing must be made aware of them in order to use them, these events cannot be physically copied down to the concept's level as they would then define it, and no longer be observational. This search has a very bad topological determinism, since all `spec_` and `inst_` arcs up the hierarchy to `typeless` must be searched in order to determine that a piece of information is lacking, and half of them on average to find a piece of information.

Similar difficulties occur when one wishes to refer to an observed event o, for instance as the cause of some other event c. An example would be *"John died"* which is observational with respect to `John`: o. *"His death caused a mutiny"* would be an example of c, as it should refer to o. But o would be implemented as superset $\mathcal{O}$ of all people who died defined by a dying event e. The node `John` would be an instance of $\mathcal{O}$ and would not have an explicit dying event attached to it since the death was observational. Since no such event is available, c must be connected somewhere else. It cannot be connected to e, since this refers to all people dying, so has to be connected to the `inst_` relation between $\mathcal{O}$ and `John`. In a complex hierarchy determining exactly which `inst_` or `spec_` event should be connected or if a new one should be built is difficult, and later changes are rendered more complex since they must not affect the meaning of the causal event.

Finally, this scheme results in many unwanted supersets. For instance, if one wishes to state that there are only 3 farmers in New York, one must build the set of all sets of three elements, and state that the farmers in New York are an instance of this set. A lot of the time, one only wishes to state that a fact is observational, but one would never wish to refer to the corresponding observational superset.

Obviously, this is not the most efficient of schemes, and more efficient variants can be found. But all suffer from the lack of locality due to the necessity of building the observational sets explicitly. This results not only in a large number of unnecessary sets, but also in an inheritance hierarchy which is difficult to maintain: adding or deleting events may affect the meaning of other statements, since it is the structure of the hierarchy which determines whether an event is observational or definitional. This fragility deprives one of the possibility of implementing algorithms which approximate the result when manipulating the hierarchy, as is desirable if the the algorithm producing an exact result is inefficient. Similarly, the lack of locality increases the dependence on search, and decreases in many other schemes distributedness with respect to the role an event is playing. All these features make sorts desirable since they make an implicit global quantity explicit locally and thereby simplify its processing.

Appendix C

Further Reasoning Results

Although depth reduces the complexity of semantic integration, semantic integration can be done on a non-annotated KB. Furthermore, semantic integration was found to be near-linear in terms of the number of concepts they explored. This appendix derives a mathematic estimate of this number in the average case.

C.1 A depthless algorithm to determine the orange node

See 6.7.2.2 *(p. 206)*.

Since the orange node is the only one through which all paths from the black node to `typeless` pass, when searching for the orange node, one wants to be sure that all paths to it have been traversed. But, one does not want to traverse each independently, since that causes high complexity. Thus one wants to traverse each arc of any path only once, while somehow maintaining tally of which paths that arc corresponded to. To do this, one needs an identifier for each path.

$p :: PathId$ is a unique identifier for each path. It is represented as a list of traversed nodes ($PathId = [Node]$). The full list of traversed nodes is unnecessary, since retaining only the node at which a path split, and the direction the path took is sufficient to uniquely identify any path. Thus, every time a node n is traversed which has more than one parent, the node n is appended to the path p, and then each of the parents is appended to a copy of p. Each copy of p uniquely identifies a path through one of the parents. The path is also compressed so that if a node n is to be appended to a list, the last element of which is also n, n is not duplicated.

Since not all paths to the orange node will be of equal length, in the ideal case, one would like a lowest-first search, as described in 6.7.2.2 *(p. 206)*. The closest thing is a breadth-first search: every node in the search space $\mathcal{S}$ is replaced by its parents at the next iteration. This replacement is called expanding a node.

One also wants to ensure that arcs are only traversed once: this will form the basis of linear complexity. To ensure this, a set $\mathcal{V}$ of visited nodes is used. If a node is met while traversing the network, which was already visited, one knows that it has already been expanded. Thus it does not need to be expanded again. By removing all visited nodes from the search space after the expansion phase at each iteration, one ensures that each arc will only be traversed once.

Finally, one wants to be sure that all paths to the orange node are traversed. Clearly, to do this, one will use the path identifiers: if all paths reach the orange node, they will all have in common the node where they split above it. The idea is thus to maintain a list of paths as one searches up that hierarchy, and test it on the fly for a node common to all paths. Since the search is a gradual process, the path list only contains the paths from the black node to the just expanded nodes.

C.1.1 Ensuring all paths reach the orange node

How can one maintain a list of all paths without retraversing each arc? One needs a complete set of all paths to know that the orange node has been traversed by each. If the search is breadth first and the paths have differing lengths, when a path reaches a node, another path may already have left it. Stopping at this already visited node would mean that not all paths were fully traversed, so one cannot be sure that the orange node was detected. Searching all paths from this already visited node would blow up the complexity since only ensuring each arc is traversed once keeps it down. The only apparent solution to this conundrum is to delay the search at any node that still has paths leading to it that have not been traversed. Then once all such paths had been traversed, they could be packaged together in a packet to be included in the path identifier of all paths leading from the node. In effect this would mean that they all share the same search beyond the node, avoiding the multiplication of paths leading to combinatorial behaviour. In this manner a full list of paths would be built up which could be tested for a common node.

However it is not possible to delay a node, since that would require either knowing that another path will visit it, or that a path cannot visit it to provide a continual supply of nodes to expand. The first is not possible since in essence it is the problem: each packet represents all the paths from one node to another, which is the same problem as determining all the paths from the black node to the orange one. The second is only possible if nodes are assigned a depth. It might seem therefore that one is condemned to combinatorial complexity. But, an implicit assumption was made: that the packets are needed in the determination of the common node. If they are not, they can be discarded, removing the need for a delay.

The packets represent all the paths from one node to another. In effect, they correspond to local expansions of the search space. Because of this, only the nodes at either end of them could conceivably be common to all paths. But the nodes at either end of them will be included in the shortest path's identifier [1], so it is unnecessary to search paths again, resulting in the preservation of linear complexity.

C.1.2 A data-structure to determine the common node

Determining the common element of a set of strings is expensive. A dedicated data-structure can go some way towards alleviating this problem. The common node candidate structure (CNCS) achieves this by expressing nodes and paths in

[1]Actually, for the top node, the recorded node is not that at which all paths come together, but the first split node above it

a graph, where each path is represented as a CNCS-node associated to the nodes in its identifier, also represented by CNCS-nodes, by a set of is_a arcs. There are 3 types of directed arc: duplicate_, source_ and is_a. Only split nodes are added to the CNCS as the orange node will be detected as a descendent of one of them.

There are three basic operations: adding split nodes, deleting split nodes and extending paths. Adding split nodes is simply a matter of building a new common CNCS-node to represent the split node, and connecting it to a specified previously existing path. Extending a path involves taking a previously existing path CNCS-node, cloning it, and assigning each copy the relevant path identifier. Cloning it means to copy the node's existing arcs as duplicate_ arcs, and to add a source_ arc to the node from its clones. Deleting a split node involves deleting its CNCS-node, and removing every reference to it in path CNCS-nodes. If another node were the source_ of the node being deleted, this other node is visited to check if it is the source_ of more than one resulting nodes: if only one arc is left, it is deleted.

The number of arcs connecting each split node CNCS-node to a path CNCS-node expresses the number of paths to which that node is common.

The set of common node candidates is implemented as an ordered circular list of pointers to each split node CNCS-node, representing all the common node candidates. The list is sorted with respect to the number paths to which each node is common. In effect the CNCS graph operations are used only to maintain this count. The list is sorted from smallest to largest number, so each time the number increases, the node's reference in the circular list is pushed back towards the end. When the number becomes equal to the number of active paths, one knows that all paths have the given node in common.

There are two cases in which the number of paths incorporating some node may become equal to the number of active paths: a path may suddenly reach the common node, and thus the path is added to the common node's CNCS node, or a path may be deleted from the set of active paths, reducing the required number of paths. This latter case could occur if most paths of the breadth first search had passed the orange node, but one was still below it, and just met a previously visited node. In both these cases, it is important to test whether there is a node that is involved in the same number of paths as the number of current active paths. To avoid multiple searches to the last element of the list, references to the list can be made to point to its last element rather than its first. Since the list is circular, the next element will be its smallest. Thus, testing the greatest element of the list – the number of paths the most common node belongs to – is achieved by reading the first element of the list. Every time the list is sorted, or an element from it deleted, an additional step will be made, but unless this occurs far more often than testing the number of common nodes, the effect will be beneficial.

C.1.3 Finding the common node

Now that the groundwork has been established, the issue is to put everything together. As before a breadth first search traverses all the black node's definitional explicit ancestors, maintaining $\mathcal{V}$ to ensure that previously traversed nodes are not investigated more than once.

The algorithm searches through all of the black node's explicit ancestors in a breadth first manner. Initially the set of visited nodes $\mathcal{V} = \emptyset$. Each element of $\mathcal{V}$ is of the form $(p :: \mathit{PathId}, n :: \mathit{Node}, c :: \{\mathit{Node}\})$, where n is the visited node being searched, p is the path that first reached it, and c is the set of n's children though which a path has reached n. Initially the black node's parents form the initial path identifiers, and the initial paths. They are added to the CNCS graph and list, and to $\mathcal{P}$ the set of active paths. Each element of $\mathcal{P}$ is of the form $(p :: \mathit{PathId}, n :: \mathit{Node}, m :: \mathit{Node})$, where n is the current node being searched and m is the previous node the path traversed. To obtain the breadth first search behaviour, $\mathcal{P}$ is implemented as a circular list similarly to the CNCS graph, for which elements are added to the end of the list (the element the list is referred to by), and read from the front (the element after that the list is referred to by). At each iteration, a path (p, n, m) is taken out of $\mathcal{P}$:

- If there is a (p', n', m') in $\mathcal{V}$ the set of visited nodes such that $n = n'$, the path node p is deleted from the CNCS and m is added to m' in the entry of

n' in $\mathcal{V}$. The number of paths of $\mathcal{P}$ is compared with the maximal number of paths with a node in common of CNCS. Unless equal, the search iterates.

- Otherwise, $(p, n, \{m\})$ is added to $\mathcal{V}$, and the set of its definitional parents $\mathcal{A}$ is found. If $\mathcal{A}$ has only one element x, (p, x, m) is placed in $\mathcal{P}$, and the search iterates. Otherwise, n is a new split node: it must be combined with every element of $\mathcal{A}$ to produce a new set of path identifiers which are associated with the elements of $\mathcal{A}$ and put into $\mathcal{P}$. The CNCS is also updated: First n is added, and connected to the path node corresponding to p. This path node is then extended by the number of new path identifiers, and each resulting path node is assigned one of these path identifiers. The search then iterates.

When the iteration stops, the node common to the most paths in the CNCS is the split node above the orange node. The closest descendent of this node which has more than one children in $\mathcal{V}$ is the orange node. The set of nodes traversed by paths is retraversed to colour them brown. This is achieved using the list of children associated with each visited node in $\mathcal{V}$, and using another set $\mathcal{T}$ to restrict the search to uncoloured nodes. This step costs $O(V)$ where V is the number of nodes between the black and orange nodes.

C.1.4 Optimising the search

The search can be optimised by changing the first case to:

- If there is a (p', n', m') in $\mathcal{V}$ the set of visited nodes such that $n = n'$, the split CNCS-nodes in p' not in p are deleted from the CNCS. The path node p is deleted from the CNCS and m is added to m' in the entry of n' in $\mathcal{V}$. The number of paths of $\mathcal{P}$ is compared with the maximal number of paths with a node in common of CNCS. Unless equal, the search iterates.

The additional step, the nodes in p' not in p are deleted from the CNCS, uses the fact only the extremities of packets need be considered. These deletions substantially reduce the number of common node candidates, improving efficiency: for instance it shortens the CNCS list, reducing the distance elements are pushed back. It also reduces the number of associations between split nodes of SemNet and nodes of the CNCS graph, which is important if the associations are implemented as a tree. A tree might be used if it is inefficient to associate information temporarily directly on the nodes of SemNet, say using controls: for instance, this might require increasing SemNet's size in memory, which might be undesirable and the reason depth information was not used.

The use of BIB trees can further improve the efficiency of this algorithm. Associating split nodes of SemNet with their counterparts in the CNCS graph with a BIB tree is beneficial as in a BIB tree failure to find information is cheaper than success: deletion of already deleted nodes in the CNCS graph fails (early) to find the relevant node. This is useful if the algorithm just presented is changed so that when a path p reaches a previously traversed node, not only are the nodes of p not in p' (the first path to reach the node) are removed from CNCS, but also the nodes of p' not in p are removed from p'. Doing this speeds up lookup failure on p' (if all paths associated with nodes in $\mathcal{V}$ are implemented as BIB trees), so that determining the nodes not common to the paths of a packet is sped up: the lower the depth of the BIB tree, the faster the failure. It is unlikely that some of the paths of a packet share many nodes, denied to others, so the increased deletion traffic on nodes of the CNCS is likely to be low. If a BIB tree associates split nodes with the CNCS, this small increase is absorbed by early failures.

C.1.5 Overall complexity

For simplicity, the worst case calculation is achieved by combining the worst case of each component, even if each worst case would not practically co-occur. Thus a higher bound for the worst case is obtained.

The use of $\mathcal{V}$ limits the search complexity to only one traversal for each of the E arcs traversed. Each arc thus traversed reaches a node, for which a test is made to see if it is already visited, and if not to add it. The worst case would be if all V nodes were in the visited node tree data-structure: $O(log(V))$. Then, if the node was already visited, the two paths must be compared involving at worst imaginable O($V.log(V)$) operations (each path has every node as split node). This would result in at worst V deletions from the CNCS, each requiring at worst V steps through source_ arcs, and V steps to delete the relevant entry from the CNCS list. The total complexity in this case is O($log(V) + V.log(V) + 2V^2$)=O(V^2). If the node were not already visited, at worst imaginable, it would be a split node involving $V - 1$ ($\approx V$) parents, which would require adding V new paths to the CNCS. At worst, the extension process would require copying V is_a or duplicate_ arcs V times, assuming all V nodes are already in the CNCS. Similarly, the reordering of the CNCS list required would be at worst V for each of the V nodes of the CNCS: V is the maximum possible length of the CNCS list, making a total of V^2. Adding all V paths to $\mathcal{P}$ is achieved in V steps. The total complexity for this case is O($2V^2 + V$)=O(V^2). Since this process is repeated for every one of the E arcs, the total complexity is bounded by O($E(V^2 + V^2)$)=O(EV^2).

O(EV^2) seems rather good, being low polynomial. But what are E and V? E and V are the number of arcs and nodes respectively that the search traversed, and herein lies the catch. Consider the worst case, where one of the black node's parents is the orange node, but the other parent splits a lot leading though very long paths before reaching the orange node. Also assume that every ancestor of the orange node has more than one parent, thus leading to many many paths. In this situation, a lot of the space above the orange node may be traversed before all the paths from the black node have reached the orange node. This overshooting of the orange node can incur an exponential cost with respect to the maximal difference between the shortest and longest paths of each packet, minus the shortest distance of the lower extremity of the packet to the orange node. This exponential effect comes from the splitting above the orange node and is due to the breadth first search assigning equal chance to every path of being about to reach the orange node.

Limiting this worst case is clearly a priority.

C.1.6 Minimising the worst case

One additional piece of information that can be used to minimise overshooting are the family controls: if the parents of the black node are of a set of families $\mathcal{F}$, then the orange node must be of the smallest super-family f including all the families of $\mathcal{F}$. This means that the search can be stopped at any node with as family a super-family of f. It can also be used for a large grained lowest depth type algorithm which ensures that the lowest families get searched first. This is achieved by the normal breadth first search, but $\mathcal{P}$ being divided into generations to search: nodes above the current family level are placed in a later generation to be searched later. This only proves useful when the black node's parents are of different families, usually leading to a very large search space, but overall a rather rare case.

A finer grained method is to use the information in the CNCS to direct the search. The problem with the breadth first search was that it accorded equal likelihood to each path of encountering the orange node. What is needed is a way of ensuring that in the worst case the search space traversed does not explode, while not penalising too harshly the average case. Two points help devising such a method:

- If a path has suffered many splits it is likely to be far from the black node. If other paths have suffered far fewer splits than it, they are worth expanding first, since all paths must reach the orange node before it is detected.
- If most paths from a given split node have reached an already visited node, it is likely that most of the paths left will too: paths are unlikely to be of equal length (otherwise breadth first search would have been ideal), so most paths will reach a previously visited node: in SemNet nodes tend to have more children than parents.

Why are the path splits taken rather than raw lengths? At every point in the breadth first search, the raw lengths are equal, but that does not prevent potential combinatorial explosions: the main source of problems are the splits, not the raw length.

These two observations from the basis for the following work allocating algorithm. Each time a path p is taken from $\mathcal{P}$, the fraction of resources to be devoted to it is obtained by looking up p in the CNCS graph: the set $\mathcal{N}$ of split nodes in its path identifier are determined uniquely by first finding all the paths $\mathcal{S}$ source to p by traversing the source_ arcs in one direction from p. The set $\mathcal{N}$ is obtained by traversing the is_a arcs from $\mathcal{S}$. The number of paths connected to a node n is $p(n)$. The fraction of resources is then

$$\prod_{n \in \mathcal{N}} \frac{1}{p(n)}$$

Because only <u>active</u> paths that are connected to a given node are included in the calculation – all other paths are deleted from the CNCS –, a path gets a higher proportion of resources when others sharing a split node with it reach a previously visited node: they are deleted from the CNCS. This satisfies the second observation.

The cost imposed by this scheduling algorithm is limited to V [2] for each arc traversal: $O(EV)$. The deletion of path nodes which are the source_ of only one (rather than none or many) path, further reduces this cost. The amount of effort to be put into a path, can be translated into the number of arcs that path (or all paths derived from it put together) should traverse by maintaining a "smallest yet encountered resource fraction", and dividing the calculated fraction for each new path by it.

Consider again the worst case: the black node has only two parents, one of which is the orange node, all other paths to the orange node are very long, and there is a lot of path splitting above the orange node. Because the black node has two parents, each path from it will be given half the resources. This means that if N' paths are traversed by paths on one parent's side, only N' will be on the other's. As a result, at most $N = 2N'$. This contrasts with the breadth first search where N' paths may be traversed on one side, and $e^{N'}$ on the other, because of bad splitting. Similarly, at most N' different nodes are reached through N' arcs. Thus the worst case is bound by $O(N^3)$, which $O(8N'^3)$. The bound on the worst case is thus still

[2] at worst the path identifier length is V, making V source_ arcs to traverse

polynomial instead of exponential. If the paths of packets are of similar lengths, there will be little overshooting, and the division of the resources will be similar to that of breadth first search.

The family method and the probabilistic method can be combined, further limiting some bad cases.

C.1.7 Finding the orange node: a conclusion

Determining the orange and brown nodes without depth is possible in polynomial complexity, but is not simple. It is also more expensive than the linear complexity given by depth information. To its advantage is a small reduction in SemNet memory size, and eliminating the depth information maintenance procedures.

C.2 A Formal Examination of the Complexity of Semantic Integration

At present SemNet does not include enough information to provide the test-bed for an empirical study of complexity. The only available alternative is to use a formal model. This is only an approximation to the problem, and may ignore salient information. It should therefore be taken more as a proof of concept, rather than as a cast iron proof of behaviour.

Classification in KL-ONE has been found NP-hard, co-NP-complete, and a variety of other unappealing worst-case complexities. This lead many researchers to devise increasingly weak, and thus increasingly useless, logics to express concepts' definitions. SemNet did not make the same choices. This section attempts to show the average complexity of downwards classification.

All the results of 6.7 *(p. 193)* were near-linear in terms of the number of concepts they explored. This section attempts to show what to expect this number to be in the average case.

Again it should be stressed that the complexity of subsumption has not been investigated. However the use of subsumption is minimised in LOLITA to testing possible implicit ancestors, descendents of the ancestor search space determined by upwards integration above the black node.

C.2.1 A model

C.2.1.1 The virtual inheritance lattice

The graph $\mathcal{E}$ is formed from SemNet's inheritance hierarchy. For simplicity, $\mathcal{E}$'s only edges will correspond to the set relations (spec_ and inst_) of the hierarchy: the edges are directed and labelless. $\mathcal{E}$ thus expresses all the structure of LOLITA's inheritance hierarchy. The relations "child", "descendent", and so on, when applied to $\mathcal{E}$ correspond to the meanings they have for SemNet's inheritance hierarchy.

$\mathcal{E}$ is a subgraph of the full possible inheritance hierarchy. The full possible inheritance hierarchy is a lattice since every concept has a most specific subsumer (the orange node) and a most general subsumee. To see this, recall that each definitional event (or action_) defines a concept and every concept can be expressed purely as the intersection of these definitional events. $\mathcal{E}$ is the explicit hierarchy and only expresses part of this lattice $\mathcal{V}$, the virtual hierarchy.

The difference between $\mathcal{E}$ and $\mathcal{V}$ falls into two categories:

- The vertices of $\mathcal{V}$ which have no instantiations in $\mathcal{E}$ because they have no descendents occurring in $\mathcal{E}$. For instance, LOLITA may know of no *"hungry peasants"*.
- The vertices of $\mathcal{V}$ which have do not have instantiations in $\mathcal{E}$ because there is no information about them, although they do have descendents in $\mathcal{E}$. For instance, LOLITA might know that *"every small French tractor is reliable"* but nothing about small tractors, an implicit ancestor of small French tractors.

Vertices of the second kind will be called dummy vertices. In essence they are compressed out of the explicit hierarchy, resulting in arcs between non-subsequent layers of the graph. This contrasts with lattices, where each generation (layer at depth d from the root) is connected only to the next (layer at depth $d+1$ from the root). For instance if a node is connected to its parent and its grandparent, it is expressed in the lattice as a vertex connected to its parent and a dummy vertex. The dummy vertex is connected to the vertex's grandparent.

$\mathcal{V}$ less vertices of the first kind forms $\mathcal{I}$, the implicit inheritance lattice. Thus $\mathcal{E} = \mathit{compress}(\mathcal{I})$, where *compress* is a simple compression function.

C.2.1.2 Lattice Properties

A lattice can be usefully described by 3 values:

- c: The average number of children each vertex with children has.
- p: The average number of parents each vertex with parents has.
- d: The depth of the lattice.

From these values the number nodes in a balanced partial lattice such as $\mathcal{I}$ can be derived. A lattice has layers. Consider the gap between two layers: every vertex of the higher level has c children, whereas every vertex of the lower layer has p parents. Overall there are $c * |\mathcal{L}_n|$ children edges for the layer $\mathcal{L}_n$. This is equal to the number of parents in the next layer $\mathcal{L}_{n+1}$ since vertices have parents only of the previous layer, and children of the next. This makes $p * |\mathcal{L}_{n+1}| = c * |\mathcal{L}_n|$ so, each new layer has $\frac{c}{p}$ more nodes than the previous layer. Since the lattice has a root node (`typeless`), the total number of nodes is:

$$\sum_{n=0}^{n=d} \left(\frac{c}{p}\right)^n = \frac{1 - \left(\frac{c}{p}\right)^{d+1}}{1 - \frac{c}{p}}$$

This could have also been derived more directly: in the whole lattice, there are as many arcs from vertices to their children as their are from vertices to their parents: $c\left(N - \left(\frac{c}{p}\right)^d\right) = p\,(N-1)$. The $N - \left(\frac{c}{p}\right)^d$ portion of the formula comes from the fact that there are $\left(\frac{c}{p}\right)^d$ nodes without children, and the $N-1$ portion comes from the fact the root node (`typeless`) has no parents. We obtain $N = \frac{c\left(\frac{c}{p}\right)^d - p}{c-p}$ which is equivalent to the above result.

The formula for N gives the following for d:

$$d = \frac{ln\left(1 + \frac{c}{p}N - N\right)}{ln(c) - ln(p)}$$

C.2.1.3 Assumptions about compression of $\mathcal{E}$

The previous section determined the number of nodes in the lattice from c, p and d. The corresponding $\tilde{N}$, $\tilde{c}$ and $\tilde{p}$ are known for any knowledge base expressed in SemNet, but $\tilde{N} \neq N$, $\tilde{c} \neq c$, $\tilde{p} \neq p$ since $\mathcal{E}$ is not a lattice. This section relates the values of $\mathcal{E}$ with those of $\mathcal{I}$, which is a partial lattice.

Since $\mathcal{E} = compress(\mathcal{I})$, some assumptions must be made about *compress*. Compression cannot remove the elements of $\mathcal{I}$ about which something is being said: the set of concepts $\mathcal{O}$ connected to observational events (or arcs). It might seem that removing all other concepts would result in better compression. However, this need not be the case, since removing them would require moving their restrictions down to the concepts about which something is said. This duplication of restriction information (and of the set relations connecting the restrictions to their templates) often outweighs the compression benefit. As a result, the compression assumed here will emphasise sharing restrictions common to more than one concept of $\mathcal{O}$ where the restrictions cannot be inherited from an ancestor of $\mathcal{O}$. This sharing will occur on nodes of $\mathcal{E}$ not in $\mathcal{O}$. Such nodes $\mathcal{C}$ correspond to the vertices of $\mathcal{I}$ which have more than one child in $\mathcal{O}$ or in $\mathcal{C}$ and which are themselves not in $\mathcal{O}$. $\mathcal{E} = \mathcal{O} \cup \mathcal{C}$

This might be sub-optimal in some cases, but more powerful schemes reduce sharing in some cases, resulting in a degradation of the inheritance hierarchy, and a decrease in topological determinism of some algorithms: upwards integration, for instance, must search downwards to test the black node's ancestors' descendents for subsumption with the black node. The more restrictions shared between different children of a concept are expressed on a shared child, the fewer subsumption tests need be made.

Because only those vertices of $\mathcal{I}$ which neither have more than one child in $\mathcal{C}$ nor are qualified by an observational event are compressed out of $\mathcal{E}$, their number can be calculated from SemNet. Every vertex of $\mathcal{I}$ corresponds either to the intersection of restricted concepts or to the restriction of a concept. Take the vertices of $\mathcal{I}$ corresponding to the restriction of a concept, not in $\mathcal{E}$: there are as many of these concepts as there are uninherited definitional restrictions on a node of SemNet, excluding those on nodes which have only one definitional restriction, and which are not definitional intersections of concepts. Call such restrictions I-restrictions. Consider now the vertices of $\mathcal{I}$ corresponding to intersections of restricted concepts: if they are in $\mathcal{I}$, they have descendents in $\mathcal{O}$ since $\mathcal{I}$ contains only vertices that are in $\mathcal{O}$ or have descendents in $\mathcal{O}$. There are $\sum_{n=2}^{n=r} \frac{r!}{n!}$ such intersections above a concept with r I-restrictions. Since $\sum_{n=1}^{\infty} \frac{1}{n!} \rightarrow e$, $r < \infty$ and all $\frac{1}{n!}$ are positive, this sum is smaller than $(e-1) * r!$. The total, including the key nodes, makes $r + (e-1) * r!$ vertices in $\mathcal{I}$ for each node of $\mathcal{E}$ with r I-restrictions.

Calculating this is useful, since the number of nodes of the lattice $\mathcal{I}$ corresponding to $\mathcal{E}$ (some instance of SemNet) can be calculated. In particular if SemNet has N nodes, restricted by at most ri I-restrictions, then $\tilde{N} < (R+1)N$, where $\tilde{N}$ is the number of nodes of $\mathcal{I}$, and $R = ri + (e-1) * ri!$.

$\tilde{c} \neq c$, for c the average number of children a node with children has, since the R new nodes for each node of $\mathcal{I}$ with respect to $\mathcal{E}$ must have children and parents. All new R nodes introduce $\sum_{n=1}^{n=ri}(n^2 - n) = \frac{ri(ri+1)(2ri+1)}{6} - \frac{ri(ri+1)}{2}$ children arcs. Since there R such nodes, the average is $\frac{ri(ri+1)(2ri+1)-3ri(ri+1)}{6R}$. Overall there are RN such new nodes with $\frac{ri(ri+1)(2ri+1)-3ri(ri+1)}{6R}$ children on average, and N nodes with c children on average. Hence,

$$\begin{aligned}\tilde{c} &= \frac{1}{(R+1)N} * \left(\frac{ri(ri+1)(2ri+1) - 3ri(ri+1)}{6R} * RN + cN\right) \\ &= \frac{1}{(R+1)} * \left(\frac{ri(ri+1)(2ri+1) - 3ri(ri+1)}{6} + c\right)\end{aligned}$$

Similarly, $\tilde{p} \neq p$, for p the average number of parents a node with children has. As there are as many parent arcs as there are new children arcs, since all of the new RN nodes have children (recall they all are ancestors of nodes in $\mathcal{O}$), the formula is much the same:

$$\begin{aligned}\tilde{p} &= \frac{1}{(R+1)(N-1)} * \left(\frac{ri(ri+1)(2ri+1) - 3ri(ri+1)}{6R} * R(N-1) + p(N-1)\right) \\ &= \frac{1}{(R+1)} * \left(\frac{ri(ri+1)(2ri+1) - 3ri(ri+1)}{6} + p\right)\end{aligned}$$

N is replaced by $N-1$ since the root node has no parents.

C.2.1.4 The resulting model

$\mathcal{I}$ shall be assumed a balanced lattice. This need not be the case, as SemNet need not contain equal amounts of information about everything. However if LOLITA is to be used within certain domains, it is not a great assumption to make that most of SemNet will be devoted to information in these domains. That information would be expected to result in a lattice since LOLITA organises similar information topologically close. Thus, even if all of $\mathcal{I}$ is not balanced, in large-scale applications, one can expect most of it to be. The results derived from making such an assumption should not be far wrong. Although only experience will tell, at this point it makes sense to determine the complexity (and hence whether semantic integration is worth further developing, or whether it should be seen as a nail in SemNet's coffin) making this assumption.

$\mathcal{I}$ has the values $\tilde{c}$, $\tilde{p}$, and $\tilde{N}$ determined above, and since $\mathcal{I}$ is assumed balanced, it has: $\tilde{d} = \frac{ln\left(1+\frac{\tilde{c}}{\tilde{p}}\tilde{N}-\tilde{N}\right)}{ln(\tilde{c})-ln(\tilde{p})}$

C.2.2 Downwards Integration

C.2.2.1 The model

Downwards integration in LOLITA involves searching for all descendents of the red nodes, and finding those that are descendents of all red nodes. Since in SemNet,

$c > p$, this search space looks exponential in the worst case. However, ways of taming the worst case were investigated in 6.7.3 *(p. 231)*, so what is of interest is the expected average complexity.

Instead of considering the descendents of all red nodes, only the descendent of one red node will be considered. Having determined the complexity $\mathcal{C}_d$ of determining the descendents of one red node, the total complexity is bounded $R_n\mathcal{C}_d$ where R_n is the number of red nodes. A pessimist can choose R_n to be the number of maximal number of red nodes of any node in SemNet. Choosing R_n to be the average number is probably more realistic since $R_n\mathcal{C}_d$ is pessimistic: it assumes the worst case where none of the red nodes share any common descendent. The algorithm only traverses any piece of the network once, so if the red nodes share a common descendent, fewer nodes will be traversed.

It will be assumed that any vertex of $\mathcal{I}$ stands an equal chance of being a red node. This is an approximation, due to the fact calculations are being performed on $\mathcal{I}$ rather than $\mathcal{E}$, and that the shape of $\mathcal{E}$ is unknown. In essence it means any node of $\mathcal{I}$ has an equal chance of being in any given instantiation of $\mathcal{E}$. It may seem strange that nodes with no children may be considered red nodes, but they model well the behaviour of instances: instances in $\mathcal{E}$ have no children, so no search will be performed. Similarly, the vertices of $\mathcal{I}$ without children have no children, so no search downwards can be performed.

Thus the question is what, on average, is the number of vertices below a vertex of $\mathcal{I}$. The expected value is simply the sum for each vertex of $\mathcal{I}$ of the probability of that vertex times the number of vertices below it. The probability of a vertex is simply the likelihood it is chosen at random from all others of $\mathcal{I}$. The calculation is simpler if all vertices of a particular layer of the lattice are considered together: k is the layer, p_k is the probability of that layer, and c_k is the number of vertices below it:

p_k corresponds to the proportion of the vertices of $\mathcal{I}$ that are in layer k, since all nodes of $\mathcal{I}$ have equal chance of being chosen:

$$p_k = \frac{\left(\frac{\tilde{c}}{\tilde{p}}\right)^k}{\sum_{n=0}^{\tilde{d}} \left(\frac{\tilde{c}}{\tilde{p}}\right)^n}$$

c_k is the number of vertices below a node of layer k:

$$c_k = \sum_{n=0}^{\tilde{d}-k} \left(\frac{\tilde{c}}{\tilde{p}}\right)^n$$

Thus the expected number of nodes below is

$$\begin{aligned} E &= \sum_{k=0}^{\tilde{d}} p_k * c_k \\ &= \sum_{k=0}^{\tilde{d}} \left[\frac{\left(\frac{\tilde{c}}{\tilde{p}}\right)^k}{\sum_{n=0}^{\tilde{d}} \left(\frac{\tilde{c}}{\tilde{p}}\right)^n} * \sum_{n=0}^{\tilde{d}-k} \left(\frac{\tilde{c}}{\tilde{p}}\right)^n \right] \end{aligned}$$

C.2.2.2 The derivation

Deriving a simple form for E, is a little long...

$$\begin{aligned} E &= \sum_{k=0}^{\tilde{d}} \left[\frac{\left(\frac{\tilde{c}}{\tilde{p}}\right)^k}{\sum_{n=0}^{\tilde{d}} \left(\frac{\tilde{c}}{\tilde{p}}\right)^n} * \sum_{n=0}^{\tilde{d}-k} \left(\frac{\tilde{c}}{\tilde{p}}\right)^n \right] \\ &= \frac{1}{\sum_{n=0}^{\tilde{d}} \left(\frac{\tilde{c}}{\tilde{p}}\right)^n} * \left[\sum_{k=0}^{\tilde{d}} \left\{ \left(\frac{\tilde{c}}{\tilde{p}}\right)^k * \sum_{n=0}^{\tilde{d}-k} \left(\frac{\tilde{c}}{\tilde{p}}\right)^n \right\} \right] \\ &= \frac{1-\frac{\tilde{c}}{\tilde{p}}}{1-\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}} * \left[\sum_{k=0}^{\tilde{d}} \left\{ \left(\frac{\tilde{c}}{\tilde{p}}\right)^k * \sum_{n=0}^{\tilde{d}-k} \left(\frac{\tilde{c}}{\tilde{p}}\right)^n \right\} \right] \\ &= \frac{1-\frac{\tilde{c}}{\tilde{p}}}{1-\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}} * \left[\sum_{k=0}^{\tilde{d}} \left\{ \left(\frac{\tilde{c}}{\tilde{p}}\right)^k * \frac{1-\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}-k+1}}{1-\frac{\tilde{c}}{\tilde{p}}} \right\} \right] \\ &= \left(\frac{1-\frac{\tilde{c}}{\tilde{p}}}{1-\frac{\tilde{c}}{\tilde{p}}}\right) * \frac{1}{1-\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}} * \left[\sum_{k=0}^{\tilde{d}} \left\{ \left(\frac{\tilde{c}}{\tilde{p}}\right)^k * \left(1-\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}-k+1}\right) \right\} \right] \\ &= \frac{1}{1-\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}} * \left[\sum_{k=0}^{\tilde{d}} \left\{ \left(\frac{\tilde{c}}{\tilde{p}}\right)^k - \left(\frac{\tilde{c}}{\tilde{p}}\right)^k * \left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}-k+1} \right\} \right] \end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{1-\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}} * \left[\sum_{k=0}^{\tilde{d}}\left\{\left(\frac{\tilde{c}}{\tilde{p}}\right)^{k} - \left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}\right\}\right] \\
&= \frac{1}{1-\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}} * \left\{\left[\sum_{k=0}^{\tilde{d}}\left(\frac{\tilde{c}}{\tilde{p}}\right)^{k}\right] - \left[\sum_{k=0}^{\tilde{d}}\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}\right]\right\} \\
&= \frac{1}{1-\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}} * \left\{\left[\frac{1-\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}}{1-\frac{\tilde{c}}{\tilde{p}}}\right] - \tilde{d}\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}\right\} \\
&= \frac{1}{1-\frac{\tilde{c}}{\tilde{p}}} - \tilde{d}\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}\frac{1}{1-\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}} \\
&= \frac{\tilde{p}}{\tilde{p}-\tilde{c}} - \tilde{d}\frac{\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}}{1-\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}} \\
&= \frac{\tilde{p}}{\tilde{p}-\tilde{c}} - \tilde{d}\frac{\left(\frac{\tilde{c}}{\tilde{p}}-1\right)\tilde{N}+1}{-\left(\frac{\tilde{c}}{\tilde{p}}-1\right)\tilde{N}} \\
&= \frac{\tilde{p}}{\tilde{p}-\tilde{c}} + \tilde{d}\frac{\frac{\tilde{c}-\tilde{p}}{\tilde{p}}\tilde{N}+1}{-\frac{\tilde{c}-\tilde{p}}{\tilde{p}}\tilde{N}} \\
&= \frac{\tilde{p}}{\tilde{p}-\tilde{c}} + \tilde{d} + \frac{\tilde{d}}{\frac{\tilde{c}-\tilde{p}}{\tilde{p}}\tilde{N}} \\
&= \frac{\tilde{p}}{\tilde{p}-\tilde{c}}\left(1+\frac{\tilde{d}}{\tilde{N}}\right) + \tilde{d} \\
&= \tilde{d} - \left(\frac{\tilde{p}}{\tilde{c}-\tilde{p}}\right)\left(1+\frac{\tilde{d}}{\tilde{N}}\right)
\end{aligned}
$$

$\tilde{N}$ appears above, when it is substituted for an equation equivalent to its definition:

$$\left(1-\frac{\tilde{c}}{\tilde{p}}\right)\tilde{N} = 1-\left(\frac{\tilde{c}}{\tilde{p}}\right)^{\tilde{d}+1}$$

C.2.2.3 Conclusion

Surprisingly, downward integration on average requires

$$\tilde{d} - \left(\frac{\tilde{p}}{\tilde{c}-\tilde{p}}\right)\left(1+\frac{\tilde{d}}{\tilde{N}}\right)$$

steps for each red node. Since the typical number of red nodes is bounded, at worst it only multiplies the above number of steps by a constant: if no intersection between any of the red node's descendents is found [3]. Since $O(\tilde{d}) \approx O(c.ln\tilde{N})$ for c, some constant, the overall cost of downwards integration is less that $O(ln(\tilde{N}))$. Now $\tilde{N} < (R+1)N$, so the order is less than $O(ln(N(R+1)))=O(ln(R+1)+ln(N)) \approx O(ln(N))$ since R is a constant. In other words, the average time complexity of downwards integration in LOLITA has been shown to be less than logarithmic in the size of the knowledge base.

This is a good result, and together with the real time integration methods developed in 6.7.3 *(p. 231)*, a hybrid method can be considered to achieve good average runtime (without the overhead of real time integration for most concepts), yet avoiding the worst case by using real time integration. Such work resides beyond the scope of this thesis, but clearly, the cost of semantic integration downwards is not a flaw in SemNet's foundations.

[3]The search space below shared descendents is only traversed once by the downwards integration algorithm, so this is a worst case

Appendix D

Extended Representation: Detailed Issues

For SemNet's representation to express the content of natural language texts, its richness must be further enhanced. This appendix demonstrates the increase in richness of the extended representation, and includes motivations for its design.

D.1 Values

This section motivates the design of the values representation and demonstrates its richness.

D.1.1 Representation Design and Basic Richness

D.1.1.1 Assumptions

In its most abstract form, some phenomenon is being mapped to some notion of intensity. The phenomenon itself, be it weight, size of a set or niceness must have various properties for it to be captured by this notion of intensity. This idea of intensity encapsulates the notion of a total order, for which each value can be described as greater or smaller than any other.

Many of the phenomena to be described can display variations over a possibly continuous range of states. The values representation must be able to specify uniquely any particular state if required. This can be done using a mapping from the state of the phenomenon to a value. The range of the mapping is the full range of states the phenomenon can display, and its domain constitutes the value type: all the values associated with a particular phenomenon constitute a value type.

The mapping will be assumed to be a bijection: each state must be represented uniquely by a value. The transitions of the phenomenon's state are mirrored by transitions in the corresponding values. One would like the values to vary continuously when reflecting the phenomenon's continuous transitions of states – if indeed the phenomenon involves such transitions. This reflects the notion that intensity encapsulates the notion of total order in a meaningful way: the total order reflects some aspect of the phenomenon, in particular the intensity of its states. This condition on the mapping function has the consequence that independent degrees of freedom must be expressed by different value types.

A degree of freedom corresponds to an independent variable used in an equation to model a phenomena's behaviour. In essence, each value type used to describe the phenomenon plays the same role as an independent variable. The mapping corresponds to the equation. For instance one might wish to describe the state of a (monochrome) light. This can be described in terms of its intensity and its colour. There are two degrees of freedom involved here, since one can vary the intensity and the colour of the light independently. Therefore two value types should be used. If only one were used, not only would this notion of independence of the two quantities be lost, but also if the bijective nature of the mapping were maintained, the continuous variation of either quantity could not be translated into the continuous variation of the corresponding value type.

D.1.1.2 Design

To start the ball rolling, things that are well known will be considered first: quantifiable values are represented as numbers. As required, a total order can be defined on numbers, and as their extensive use in the sciences demonstrates, they are suitable for modeling the intensity of many phenomena.

- **Numerical values**

Consider first the counting of discrete entities. This corresponds to normal scalars: numbers. Internal addition and multiplication operators are defined for this type of value. These correspond directly to addition and multiplication for any two numbers. In general this value type is used whenever numbers need to be referred to independently of any unit.

For simplicity, this value type is assumed to be the set of positive real numbers (here denoted by $\mathbb{R}^+$). This corresponds to the fact that people can refer to concepts such as fractions or even irrational transcendentals such as π which are real. Not all forms of numbers have been included though, since it seems unlikely that the extension of the set of real numbers to include infinitely small and infinitely large numbers is of interest to any but specialized number theorists. The choice of positive real values as most basic form in no way is limitative, and allows the construction of negative numbers if needed. Not including them in the basic value form will be shown to simplify the treatment of all values in the long run. It also corresponds to the intuition that negative numbers are an abstraction and that one cannot count a negative number of entities, such as sheep.

The purpose is not to define real numbers formally as is done in number theory, starting from natural numbers built with the successor function and zero, and building from this the set of integers, of rational numbers and finally of real numbers. Instead, an external definition is assumed. Thus the set of all quantifiable values is simply defined by a "real number" event, it in turn defined by the "real number" action which is not defined in the network.

Addition and multiplication are not defined within the semantic net. The behaviour they must be associated with is to be implemented by the reasoning algorithms using them. Defining them in the semantic net would be inefficient to the extreme, and would require very complex definitions, which is unnecessary for an NLE system.

Since they have been defined to correspond to addition and multiplication, the addition and multiplication operators can be used to define useful values. In particular they can be used to define zero, one, infinity and negative numbers.

∘ Expressing order

It is often necessary to express a order relation corresponding to the intensity one value being higher than another. For instance, *"John owns more chicken than Giles"*. This relation could be represented by a separate ">" relation. However, such a relation breaks uniqueness since it is possible to express the relation in terms of $a < b \equiv \exists \delta \in \mathcal{V}. a + \delta = b$ as long as all elements of $\mathcal{V}$ are positive. Since expressions of this form may occur in order to define, say, the variance in chicken owned by farmers in Essex, it is useful for them also to be recognised as expressing order, which would not happen if an independent order relation were used.

∘ Defining useful values

There are two values of $\mathbb{R}^+$ which have a distinctive behaviour when involved in an addition: 0 and infinity. Since addition is defined outside the network, it can be used to define them: $\forall v \in \mathcal{V} v + 0 = v$ is used to define 0, and $\forall v \in \mathcal{V} v + \infty = \infty$ defines ∞. Similarly, one value has a distinctive behaviour with respect to multiplication: 1. It can be defined by $\forall v \in \mathcal{V} v * 1 = v$.[1]

Once 0 has been defined, it can be used to define the set of positive real numbers not including it (here denoted by $\mathbb{R}_0^+$) using an absence of occurrence synonym. It can also be used to define the set of negative numbers $\mathbb{R}^-$: $\forall p \in \mathbb{R}^+ \ \exists m \in \mathbb{R}^- \ . \ p + m = 0$. The full set $\mathbb{R}$ can be expressed with these two sets using an exclusively ored synonym:

(E_0, R): $\{\Delta F$-`subject_`-FO: $\mathcal{V}$; $\Delta\forall$-`subject_`-$I\Delta$ 0; $\Delta\forall$-`action_`-IO: $\oplus$; ΔF-`object_`-FO: $\mathcal{V}\}$

(E_1, R): $\{OF$-`subject_`-$F\Delta$: $\mathbb{R}^+$; $\Delta\forall$-`action_`-IO: `real_positive`$\}$

(E_2, R): $\{\Delta F\forall$-`subject_`-$F\Delta$: $\mathbb{R}_0^+$; $\Delta\forall\forall$-`subject_`-IO: 0; $\Delta\forall\forall$-`action_`-IO: `synonym_`$\}$

(E_3, R): $\{\Delta F$-`subject_`-FO: E_2; $\Delta\forall$-`action_`-IO: `size_`; $\Delta\forall$-`object_`-IO: 0$\}$

[1] $\forall v \in \mathcal{V} v * a = a$ applies both to $a = 0$ assuming $v \neq \infty$, and to $a = \infty$ assuming $v \neq 0$

(E_4, R): $\{\Delta F$-`subject_`-FO: $\mathbb{R}^+$; ΔF-`subject_`-$F\Delta$: $\mathbb{R}^-$; $\Delta\forall$-`action_`-IO: $\oplus$; $\Delta\forall$-`object_`-IO: 0$\}$

(E_5, H): $\{\Delta\forall$-`subject_`-$\exists!$O: $\mathbb{R}_0^+$; ΔF-`subject_`-$F\Delta$: $\mathbb{R}$; $\Delta\forall$-`action_`-IO: `synonym_`$\}$

(E_6, H): $\{\Delta\forall$-`subject_`-$\exists!$O: $\mathbb{R}^-$; ΔF-`subject_`-$F\Delta$: $\mathbb{R}$; $\Delta\forall$-`action_`-IO: `synonym_`$\}$

(E_7, R): $\{$ ΔF-`subject_`-FO: $[E_5, E_6]$; $\Delta\forall$-`action_`-IO: `xor_`$\}$

where $\mathcal{V}$ is the set of all values, and $\mathbb{R}^+$ and $\mathbb{R}^-$ are definitional `spec_`s of $\mathcal{V}$.

o Internal and external operators

Two terms that will appear often throughout this section are "internal" and "external" operators. Internal operators correspond to those whose arguments and result belong to the same set. Some (but not all) of an external operator's arguments have a different value-type than its result. For instance an external multiplication operator could have as one argument a number, and another as a temperature, with as result a temperature.

o Representing the Addition and Multiplication operators

Internal addition and multiplication are represented by two special actions $\oplus$ and $\otimes$ respectively. External multiplication is represented by the action $\odot$. An addition event corresponds to $\sum s_n = o$ where s_n are all the partial `subject_`s of the event, and o is its `object_`. A multiplication event corresponds to $\prod s_n = o$ where again s_n are all the partial `subject_`s of the event, and o is its `object_`.

o Expressing numerical values

Once the values 0 and 1 have been defined, it is possible to use the fact that the addition operator states that its `object_` is the sum of all its partial `subject_`s. For instance, an addition event which has the value 1 as both its `subject_`s defines its `object_` as the value 2. This scheme can be extended to the number 10. Then using a combination of multiplication and addition, any value can be expressed. For instance 0.5 would be expressed as the value which when multiplied by 10 is 5. Since $0.5 \equiv \frac{1}{2} \equiv \frac{5}{10} \equiv ...$, uniqueness is lost. However this is an unavoidable property of numbers themselves. If uniqueness is of paramount importance it is possible to require all numbers to be normalized to their simplest form. However this is not always advisable. For instance 0.500 can be used to state that the error on the value is of the order 10^{-4}, and has a different meaning to 0.5. In the net the first could be expressed as $\frac{500}{1000}$, whereas the second could be expressed as $\frac{5}{10}$. Alternatively both could be expressed as $\frac{1}{2}$ and the error expressed explicitly. The merits of these options depend on the type of value being considered, so cannot be determined for all values.

In addition to the values defined in this way, some values such as π whose numerical value cannot be defined explicitly in the net can be defined by their names, or by their properties.

- **Unquantified values**

Quantifiable value types have been presented as numbers. It was argued in 7.1.2 *(p. 248)* that it is unnatural to represent unquantifiable value types as numbers, so they should not be represented in this way. However, it was also argued that as similar a representation should be adopted for both kinds of value types: both are used in a very similar way. How can such a circle be squared?

The difficulty lies in the fact the representation of unquantified value types wishes to preserve some of the properties of numbers, but not all. The question to ask is therefore which properties are not wanted and which are.

By definition, unquantifiable values must stay unquantifiable. However, as established previously all values must have a total order. Moreover, even if unquantifiable value types do not have definite numbers assigned to them, one can talk about multiples of them: *"John is twice as happy as Mary"* and *"John is twice as tall as Mary"* share the same structure. It is desirable to express them in the same way, without for that matter requiring them to share each other's properties. Even addition between unquantifiable value types appears possible *"John is nicer than the rest of us put together"*.

For values expressed numerically, it was established that the total order could be expressed in terms of the addition operator in order to preserve uniqueness: $a < b \equiv a + \delta = b$. One would like to express total order in the same way for unquantifiable value types. Thus, an internal addition operation is postulated for unquantifiable value types.

One also wants to represent notions such as *"John is twice as happy as Mary"*, which involve one undefined value expressed as the multiplication of another by a scalar. This corresponds to an external multiplication operator: it takes an argument from the set of scalars, in this case real numbers, and a value of the unquantifiable value type, and returns another such value. It should be noticed that the multiplication is not internal, so strange concepts such as the intensity of happiness squared do not appear.

For values expressed numerically, it was shown that for any number to be defined internally in the net, all that was needed was the number one. Moreover it was shown that this value could be uniquely defined by internal multiplication. Since values of the unquantifiable kind cannot be associated with a number in any way, internal multiplication cannot be used with them.

The internal operation of addition also allows two values to be defined: zero and infinity. However, these cannot be used to express all other values numerically. At worst, they can be used to express the limits of the range of the unquantified value type corresponding to the range of states a phenomenon can display. Assigning a zero value to unquantifiable value types could make sense in certain cases: *"John is not angry"* could be understood as implying that the intensity of John's anger is nil.[2].

o Relating Unquantifiable values to Quantifiable values

Unquantifiable value types have an internal notion of addition, and an external notion of multiplication. They therefore appear to correspond to numerical val-

[2]This would be the case in particular if a model of human emotions assumed that all humans have some values expressing their emotional state at all times: the traditional form of negation by non-existence could not be applied since it would contradict the model in saying that John has no anger value at a particular time although all humans have an anger value at all times. See 7.3.2 *(p. 267)*

ues which are unknown. For instance, *"John is twice as nice as Mary"* would be expressed in the network as the niceness value associated with John x and the niceness value associated with Mary y are in the relation $x = 2 * y$. Since no unit value can be defined internally without multiplication, all values of the unquantifiable type cannot be associated to any particular numerical value. In this sense there is no difference between unquantifiable values and variables which cannot have a number substituted for them. Exactly what is contained by the variables is of no concern. Thus it is possible for them to contain real numbers.

What the numbers are is not known, except that they must be positive: if they could be negative, it would be impossible to establish an order, since this is expressed by $a < b \equiv \exists \delta \in \mathcal{V} a + \delta = b$, but this equivalence is only if $\mathcal{V}$ only contains positive values. Similarly, *"John is twice as nice as Mary"* should not be understood as *"Mary is nicer than John"*, which would be the interpretation if Mary's niceness was negative. This applies for notions one might think of expressing by a negative value too: *"Today is twice as cold as yesterday"* should not be understood as *"Today was less cold than yesterday"*, which would be the interpretation if coldness was expressed as a negative value. Hence, the requirement for values to be by default positive values.

Only allowing positive real numbers to correspond to the states of the phenomenon is in no way limiting, since any range of states can be mapped bijectively to the set of real positive numbers. Not only does it capture the behaviour of natural language, but it also corresponds intuitively to the notion that negative numbers are an additional abstract construction having more to do with the notion of lack or absence than to do with the presence of some sensation/perception, which is the usual thing unquantified values are used for. Finally it allows the same algorithms to work with quantified values as unquantified ones without special treatment for unquantified values.

- **Quantified values involving units**

Quantified values which must be expressed in terms of units sit between numerical values and unquantifiable values: although they can be expressed in terms of numerical values, this depends on the choice of a unit, and for each phenomenon there are many possible choices of unit. No unit is more natural in general than any other. Just as was the case for knowledge 2.6 *(p. 15)*, the choice of most natural unit depends on the problem being considered.

The representation must allow values to be expressed in terms of more than one unit. At its most basic, the requirement is that an NLE system should allow statements such as *"John believes* $32°F$ *to be* $0°C$*"* to be represented. However this point has deeper implications. In many cases conversions from one unit to another are very complicated. For instance, the Romans divided the day into 10 hours of sunlight. This was useful since it allowed them to use sundials. However, converting that representation of time into our current 24 hour system requires more knowledge than may actually be available: the time of year, the latitude of the place where the time was determined, the position of the earth in the solar system. Such conversion is unnatural at the very least, and completely unnecessary. In general, units of are used very loosely, and conversion leads to a more precise meaning than that originally intended: *"He weighs 80 kgs"* does not mean *"He weighs 80000 g"*, and *"The house is 100 yards to your left"* does not mean *"The house is 91.44 metres to your left"*.

Representing statements in their original unit allows values associated with non-literal behaviour to be detected, and their normal loose value to be simulated: In general, 5,10 and its multiples of 10 and so on may be used more loosely than others such as 7. But this also depends on the unit being used, and whether one can easily express the same value in terms of another unit: 15 minutes is more precise than $\frac{1}{4}$ of an hour, similarly 60 minutes is more precise than an hour, but $\frac{1}{3}$ of an hour is not used so 20 minutes is vaguer than either 15 or 30 minutes. Terms used in a looser way are given a non-literal control to ensure that reasoning algorithms do not take them too literally, and send them off to be interpreted by the relevant non-literal interpretation algorithm, so that the resulting interpretation can be used.

Just as for unquantifiable values, internal addition is used to express total ordering of values which can be numerically expressed in terms of some unit. Also, as for unquantifiable values, there is no internal multiplication since it would allow the value 1 to be defined independently from any unit. This would indicate that there is a natural unit for the value type, which contradicts the idea that units are an artificial notion mapped upon a continuum[3]. This also means that units are not defined internally to be equal to one, but are defined purely externally.

◦ Units for values having an agreed zero point

Some value types have an agreed zero value. For instance distance, duration, velocity and weight have an agreed zero value. Whatever the unit chosen for each, if the value has a zero value in one unit it will have a zero in all others. These value types usually are those for which a zero point is directly perceivable using human senses. Other value types do not have an agreed zero value. For instance, the units for temperature differ: $0°C \equiv 32°F \equiv 273.15°K$. Another example is time, where a date in the Gregorian Calendar differs from one in the Jewish or the Muslim one. As units that have an agreed zero point are easier, they shall be considered first.

Units of a value type are special values of the type which are defined externally to the network. For instance, a metre is a unit of the distance type, and it is defined externally to the network, in this case simply by the word "metre". By using external multiplication, the unit can be multiplied by a scalar to express numerically the number of units any value of its type corresponds to:

(E,R): { ΔI-`subject_`-IO: [`1 metre`,5]; ΔI-`action_`-IO: $\odot$; ΔI-`object_`-$I\Delta$: v}

states that v is 5 metres. v can also have its numerical value determined in terms of other units too such as yards.

If a value type has an agreed zero point, this point can be expressed using internal addition: $\forall v \in \mathcal{V}'.v + 0 = v$. The correspondence of this point with the unit value zero can be made explicit as follows

(E_1,R): { ΔI-`subject_`-IO: [`1 metre`,0]; ΔI-`action_`-IO: $\odot$; ΔI-`object_`-IO: x }

(E_2,R): { $\Delta\forall$-`subject_`-$I\Delta$: x; ΔF-`subject_`-FO: $\mathcal{V}$'; $\Delta\forall$-`action_`-IO: $\oplus$;
ΔF-`object_`-FO: $\mathcal{V}$'}

[3]Having a defined method of determining the number of units, such as using a ruler, does not change the fact that the unit itself is not intrinsic to the phenomenon being measured.

where x is the zero value for that particular value type $\mathcal{V}$'.

o **Units for values without an agreed zero point**

Units for values without an agreed zero point are a little more complex: the notion of unit here refers not only to the interval between two values required for them to be separated by a unit, but also to the starting value from which these units are counted. For instance, centigrade and kelvin differ in their starting points, or zero points, but not in the interval between two temperatures corresponding to a unit. The value of a value type expressed in a particular unit depends therefore not only on the interval corresponding to the unit, but also to its zero point. In general, a value v corresponds to n units u when $v - z = n * u$ where z is the zero value. The case for units with an agreed zero point is special in that $z = 0$.

For these values to be expressed, the formula $v - z = n * u$ must be expressed literally. Thus to state that the temperature t is $25°C$, the following representation is needed:

(E_1,R): { ΔI-`subject_`-IO: [1°C,25]; ΔI-`action_`-IO: $\odot$; ΔI-`object_`-$I\Delta$: t' }
(E_2,R): { ΔI-`subject_`-IO: [0°C, t']; ΔI-`action_`-IO: $\oplus$; ΔI-`object_`-IO: t }

This is the general form of unit expression. If a new unknown form of unit must be dealt with, it will be assigned its own "0-point-unit" and "1-unit" values. These can then be equated with the corresponding values in other unit scales if LOLITA comes to learn the equivalences. If the "0-unit" turns out eventually to be the agreed zero point of the value type, all the additional $\oplus$ events can be pruned.

The "0-point-unit" corresponds to a point of the scale, and does not necessarily correspond to a value equal to zero. In particular, multiplying the "0-point-unit" by a scalar will not result in the "0-point-unit". Similarly, the "1-unit" corresponds to an interval, a difference between two values, and not to the value "0-point-unit+1-unit".

Units are an additional artificial external mapping on top of what is essentially an uncountable quantity. This means for instance that when someone says that it is twice as hot today as it was yesterday, it does not mean that if it was $10°C$ yesterday, today it is $20°C$. It depends on the zero point he implicitly took when he made the statement. In general, if a value type has an agreed zero point, it will usually be taken as the zero point of any statement. But in other cases the scale must be determined from the context.

o **Defining units in terms of others**

In some cases a unit corresponds to the product of two others. This is allowed since it corresponds to external multiplication. Thus, the concepts of square and cubic metres are expressed in terms of the unit one metre: each of these correspond to a different value type. Relations between different units of the same value type are also possible, such as 1 metre is 100 cm. However, such conversions are only performed on literal uses of the term metre.

D.1.1.3 Phenomena's ranges

In the abstract design (D.1.1.1 *(p. D-2)*), it was stated that "The range of the mapping is the full range of states the phenomenon can display, and its domain constitutes the value type: all the values associated with a particular phenomenon constitute a value type." This implies that unquantified values may only make sense within a certain range of values. In particular this poses the problem of what should be done, say when a value is defined to be ten times the maximum bound of the range: is this meaningless? The short answer is yes. If a value is greater or smaller than that which can be mapped to and from the phenomenon it is representing, then it is no longer describing the phenomenon. For instance if the sensation of heat is considered, one cannot discuss literally the sensation of heat experienced by a person when he touches the surface of the sun, since no such sensation exists.

Another problem lies at the smaller bound of a value type's range. For addition to express ordering, the range's lower bound must at least be the smallest difference between any two unequal values: if a and b are very close, t must still be within the value's type range for an order to be expressible in the semantic net since addition is internal.

D.1.2 Network realization

D.1.2.1 Basic design

The value scheme is implemented as described above, with the $\oplus$, $\otimes$ and $\odot$ events. The "real number" event required to restrict values to positive real values has as action `real_positive`. These 4 events suffice to express all relations between values. However for values to be useful, a fourth event is required: `has_value`. This associates a concept with a value, corresponding to the state in which the concept is.

`has_value` is the general form of associating a concept with a value. It takes the concept as `subject_`, and the value as `object_`. Because each concept may be associated with many different values, each expressing an independent aspect of its state, the `has_value` event has many specializations. Each expresses state in terms of a different phenomenon. For instance, the degrees of belief and certainty an agent accords a statement are two such phenomena, as are the amplitude, tone and timbre of a musical instrument. Each specialization enforces typing on the value which expresses the state of the phenomenon it represents, thereby ensuring different value types are not confused.

The set of all possible values is a restriction of `typeless`, defined by $\oplus$, the internal form of addition. Values are therefore anything which obeys a total order. This fits in with the notion that they correspond to gradation. A subset of this is the set of real numbers $\mathbb{R}^+$, defined by `real_positive`. Another is the set of values which have internal multiplication: the set of `numeral_` which is defined by $\otimes$. $\mathbb{R}^+$ has two subsets `countable_` and `uncountable_` which are antonyms with respect to it. `countable` is defined as the intersection of $\mathbb{R}^+$ and `numeral_`, whereas `uncountable_` is defined by being the antonym of `countable`. Finally, $\odot$, the external form of multiplication is defined as having one `subject_` and `object_` arc with `uncountable_` targets, and more than one `subject_` of `values_` type.

(E_0, R): $\{\mathrm{O}\boxed{\exists!}F\text{-}\mathtt{subject\_}\text{-}\boxed{\forall}\forall F\Delta\colon \mathcal{V}_{P'};\ \Delta\forall\!\!\forall\text{-}\mathtt{action\_}\text{-}I\mathrm{O}\colon \oplus;$
$\mathrm{O}\boxed{\forall\!\!\forall}\text{-}\mathtt{object\_}\text{-}\boxed{\exists!}\Delta\colon \mathcal{V}\}$

(E_1, R): $\{\Delta\boxed{\exists!}\text{-}\mathtt{subject\_}\text{-}\boxed{\forall\!\!\forall\!\!\forall}\mathrm{O}\colon \mathcal{V}_{P'};\ \Delta\forall\text{-}\mathtt{subject\_}\text{-}\exists!\mathrm{O}\colon \mathcal{V};$
$\Delta\forall\text{-}\mathtt{action\_}\text{-}I\mathrm{O}\colon \mathtt{synonym\_}\}$

(E_2, R): $\{\mathrm{O}F\text{-}\mathtt{subject\_}\text{-}F\Delta\colon \mathbb{R}^+;\ \Delta\forall\text{-}\mathtt{action\_}\text{-}I\mathrm{O}\colon \mathtt{real\_positive}\}$

(E_3, R): $\{\mathrm{O}\boxed{\exists!}F\text{-}\mathtt{subject\_}\text{-}\boxed{\forall}\forall F\Delta\colon \mathtt{numeral\_}_{P'};\ \Delta\forall\!\!\forall\text{-}\mathtt{action\_}\text{-}I\mathrm{O}\colon \otimes;$
$\mathrm{O}\boxed{\forall\!\!\forall}\text{-}\mathtt{object\_}\text{-}\exists!\Delta\colon \mathtt{numeral}\}$

(E_4, R): $\{\Delta\boxed{\exists!}\text{-}\mathtt{subject\_}\text{-}\boxed{\forall\!\!\forall\!\!\forall}\mathrm{O}\colon \mathtt{numeral}_{P'};\ \Delta\forall\text{-}\mathtt{object\_}\text{-}\exists!\mathrm{O}\colon \mathtt{numeral\_};$
$\Delta I\text{-}\mathtt{action\_}\text{-}I\mathrm{O}\colon \mathtt{synonym\_}\}$

(E_5, R): $\{\Delta I\text{-}\mathtt{subject\_}\text{-}I\mathrm{O}\colon \mathtt{countable\_};\ \Delta I\text{-}\mathtt{subject\_}\text{-}I\Delta\colon \mathtt{uncountable\_};$
$\Delta I\text{-}\mathtt{action\_}\text{-}I\mathrm{O}\colon \mathtt{antonym\_};\ \Delta I\text{-}\mathtt{object\_}\text{-}I\mathrm{O}\colon \mathbb{R}^+\}$

(E_6, R): $\{\mathrm{O}\boxed{\forall\!\!\forall}\text{-}\mathtt{subject\_}\text{-}\boxed{\exists!}\mathrm{O}\colon \mathtt{uncountable\_};\ \mathrm{O}\boxed{\exists!}F\text{-}\mathtt{subject\_}\text{-}\boxed{\forall}\forall F\Delta\colon \mathcal{V}_{P''};$
$\Delta\forall\!\!\forall\text{-}\mathtt{action\_}\text{-}I\mathrm{O}\colon \odot;\ \mathrm{O}\boxed{\forall\!\!\forall}\text{-}\mathtt{object\_}\text{-}\boxed{\exists!}\mathrm{O}\colon \mathtt{uncountable'\_}\}$

(E_7, R): $\{\Delta\boxed{\exists!}\text{-}\mathtt{subject\_}\text{-}\boxed{\forall\!\!\forall\!\!\forall}\mathrm{O}\colon \mathcal{V}_{P''};\ \Delta\forall\text{-}\mathtt{subject\_}\text{-}\exists!\mathrm{O}\colon \mathcal{V};$
$\Delta\forall\text{-}\mathtt{action\_}\text{-}I\mathrm{O}\colon \mathtt{synonym\_}\}$

where $\mathcal{V}$ is the set of values; $\mathcal{V}_{P'}$ and $\mathcal{V}_{P''}$ are different subsets of the power-set of values; `numeral_` and $\mathbb{R}^+$ are definitional `spec_s` of $\mathcal{V}$; `countable_` and `uncountable_` are definitional `spec_s` of $\mathbb{R}^+$; and `countable_` is also a definitional `spec_` of `numeral_`.

The way in which values are organized in the inheritance hierarchy allows new number sets, such as positive and negative real numbers ($\mathbb{R}$) or complex numbers to be defined as subsets of `numeral_` if required, as they are entities with internal multiplication. Moreover it defines unquantifiable values as those for which no internal multiplication exists, since only internal multiplication can assign numbers to values.

The $\oplus$, $\otimes$ and $\odot$ events also can have specialized forms for particular value types. This allows the type checking mechanism to test whether different value types are being confused. For instance a $\oplus$ event with `subject_s` of value types "distance" and "temperature" is meaningless.

D.1.3 The use of values

D.1.3.1 Set Cardinality

• **Size**

The `size_` event specifies the cardinality of the set which is its `subject_`. It is a specialization of the generic `has_value` event. It takes as `object_` a positive integer, of the set $\mathbb{N}$ which is defined by:

(E_0,R): $\{\Delta F$-`subject_`-FO: $[E_1, E_2]$; $\Delta\forall$-`action_`-IO: `xor_`$\}$
(E_1,H): $\{\Delta F$-`subject_`-$F\Delta$: $\mathbb{N}$; $\Delta\forall$-`subject_`-IO: 1;
  $\Delta\forall$-`action_`-IO: $\oplus$; $\Delta\forall$-`object_`-$\exists!O$: $\mathbb{N}$' $\}$
(E_2,H): $\{\Delta F$-`subject_`-$F\Delta$: $\mathbb{N}$; $\Delta\forall$-`action_`-IO: `synonym_`; $\Delta\forall$-`object_`-IO: `0` $\}$
(E_3,R): $\{\Delta I$-`subject_`-IO: $\mathbb{N}$; ΔI-`subject_`-$I\Delta$: $\mathbb{N}$'; $\Delta\forall$-`action_`-IO: `synonym_`$\}$
(E_4,R): $\{\Delta I$-`subject_`-IO: $\mathbb{R}$; ΔI-`action_`-IO: `spec_`; ΔI-`object_`-$I\Delta$: $\mathbb{N}$ $\}$

This states that every number n of $\mathbb{N}$ is either a successor of some other number m of $\mathbb{N}$ ($n = m + 1$) or is zero. It also states the $\mathbb{N}$ is a subset of $\mathbb{R}$. $\mathbb{N}$ and $\mathbb{N}$' are used to allow quantification to refer to different elements of the same set.

The `size_` event can be used either to restrict a set to a certain size, or to define the size a set has. This can be used to state that a set $\mathcal{A}$'s subset $\mathcal{B}$ represents a very small portion of it[4]:

(E_0,R): $\{\Delta I$-`subject_`-IO: $\mathcal{A}$; ΔI-`action_`-IO: `spec_`; ΔI-`object_`-$I\Delta$: $\mathcal{B}\}$
(E_1,R): $\{\Delta I$-`subject_`-IO: $\mathcal{A}$; ΔI-`action_`-IO: `size_`; ΔI-`object_`-$I\Delta$: `A`$\}$
(E_2,R): $\{\Delta I$-`subject_`-IO: $\mathcal{B}$; ΔI-`action_`-IO: `size_`; ΔI-`object_`-$I\Delta$: `B`$\}$
(E_3,R): $\{\Delta I$-`subject_`-IO: `A`; ΔI-`subject_`-AO: $\mathcal{S}$; ΔI-`action_`-IO: $\otimes$;
  ΔI-`object_`-IO: `B`$\}$

Here $\mathcal{S}$ is defined as the set of numbers for which a subset is considered very small with respect to its superset. This type of information proves useful in some forms of reasoning such as reasoning by analogy [Long et al. 93]. More generally it can also be used to state that a set is smaller that its superset, for instance to state that John did not eat every apple which ever existed or will ever exist in the sentence: *"John ate some apples"*.

[4]`size_` depends on the frame of existence under consideration.

- **Averages**

The definition of the `size_` event allows average values to be constructed. For instance, every person has a weight. Thus the set of people's weights can be defined. The average weight is then defined as the ratio of the sum of all the weights and the number of people:

(E_0,R): $\{\Delta F$-`subject_`-FO: `people`; $\Delta\forall$-`action_`-IO: `has_weight`;
ΔF-`object_`-$F\Delta$: $\mathcal{W}\}$

(E_1,R): $\{\Delta I$-`subject_`-IO: `people`; ΔI-`action_`-IO: `size_`; ΔI-`object_`-$I\Delta$: $N\}$

(E_2,R): $\{\Delta I$-`subject_`-$\forall$O: $\mathcal{W}$; ΔI-`action_`-IO: $\oplus$; ΔI-`object_`-$I\Delta$: $T\}$

(E_3,R): $\{\Delta I$-`subject_`-$I\Delta$: A; ΔI-`subject_`-IO: N; ΔI-`action_`-IO: $\otimes$;
ΔI-`object_`-IO: $T\}$

A is an average person's weight, T is the total weight of all people, and N is the number of people.

D.1.3.2 Sensations

Sensations can be expressed by values. The way in which they are used in natural language will however determine the precise ranges chosen. This section discusses the features of their usage which will determine the choice of representation.

As shown by the example *"Today is twice as cold as yesterday"* not meaning *"Today is less cold than yesterday"*, the sensation of cold is considered as a positive phenomenon. The same is true for the sensation of heat. Intuitively this corresponds to the notion that both heat and cold can be perceived as intensities of different sensations, especially when dealing with their extreme forms.

Values which express sensations are agent dependent. Thus it is possible for Paul to feel hot, whereas Chris feels the same room chilly. Such values are subjective values, and also apply to subjective qualities such as beauty. If John finds Mary beautiful, Jacob may find her ugly. These subjective notions are to be contrasted with objective values, such as *"a pink elephant"*. Objective values correspond to those for which disagreement between two people is considered a contradiction: *"The elephant is grey"* and *"the elephant is green"*. An agreement between agents is expected: one does not say *"I find that elephant grey"*, but one can say *"I find that elephant friendly"*.

The two sensations can be mapped to part of the range of an indirect notion of external temperature. This notion of temperature is defined independently from the sensations of cold and heat, but with respect to phenomena identified with other senses (for instance the sight of frozen water). Temperature is therefore not directly perceived, and includes values that cannot be perceived directly such as absolute zero, or the temperature of the surface of the sun.

The sensation of the agent of neither heat or cold can be mapped to a certain temperature: the "sensation zero point". The sensation of cold is then mapped to the negative side of the sensation zero point in the temperature scale, and the sensation of heat is mapped to the positive. The zero sensation point need not correspond to the temperature zero in the scale associated with any particular unit. Indeed since it corresponds to the temperature the agent feels neither warm or cold at, it depends on each agent and the state the agent is in. For instance if the agent has a fever, he will feel cold at the external temperature at which he usually feels comfortable. The ranges of cold or heat sensation are therefore mapped to different intervals of the range of temperature.

Although people's definition of concepts such as *"quite hot"*, *"hot"*, *"very hot"*, *"extremely hot"* are usually defined with respect their sensation of heat, their use of it may not make sense as a direct perception. For instance, one might say *"the surface of the sun is very hot"*. This sentence refers to the direct sensory perception of heat experienced day to day, as range of experience: the meaning is the same as *"Boiling water is very hot"*. But the temperature considered lies outside of this range of perception. Thus the ranges considered must be extended when mapped to the temperature range. Thus the notions of *"hot"*, *"very hot"* (and so on) are extended to the maximal temperature, which intensionally is infinite[5].

The expression of values for certain ranges depends on the object being described.

[5]Even if it may be finite according to current cosmological theories. Similarly, intensionally the minimal temperature could be negatively infinite, eg: a parallel universe with temperature -3000°C

For instance, a small elephant would still be expected to be bigger than an enormous mouse. Such concepts are relational. In some cases the object with respect to which the measurement is being made is stated explicitly, sometimes implicitly. For instance *"The sun is quite a cold star"* is an explicit usage, whereas *"Julio is tall"* is implicit: not only is Julio being compared with a set of people, but also he is being compared with the specific set of Spanish people. With respect to Dutch people, Julio may be rather short. The same accounts for *"The coffee is cold!"* and *"I don't like warm cola!"* not implying that the coffee is colder than the cola.

It would be rather simple minded to assume that the sentence *"John is twice as nice as Mary"* is meant to mean that John is exactly twice as nice as Mary. For instance this can happen when one says *"John is three times her age"*, and one knows that John is 70. LOLITA would then infer that her age must be $23\frac{1}{3}$ exactly. However this case is dealt with by considering as non-literal the usage of the concept *"three times"*.

The same observations hold for sensations which are not so obviously different. For instance, *"This room is brighter"* and *"That room is twice as dark"* refer to different intensity scales derived as the result of the partitioning of the same sense. Other such pairs include *"quiet"* and *"loud"*, *"beautiful"* and *"ugly"*, and *"fast"* and *"slow"*. In general, the choice of scale will reflect those in language. Just as for heat and cold, all these scales can be combined into a unique one, stating that something cannot literally be quiet and loud simultaneously.

○ Objective Ranges

◇ Ranges of values

Intervals of values can be expressed using the total order relation: $>$ or $\geq$. In the network, the δ is expressed by referring to the sets $\mathbb{R}^+$ or $\mathbb{R}_0^+$, if the relation is $\geq$ or $>$ respectively. This solution works both for quantifiable and unquantifiable values, since $\mathbb{R}^+$ includes both types of values.

Intervals between two values a and b ($[a,b]$) can be expressed using two events representing $\geq$. Similarly, $>$ can be used for open intervals: $]a,b[$. For instance, the set of adolescents would be defined by:

(E_0,R): {ΔF-subject_-$F\Delta$: Adolescents; $\Delta\forall$-action_-IO: has_age;
ΔF-object_-$F\Delta$: Ages }

(E_1,R): {ΔF-subject_-$F\Delta$: Ages; $\Delta\forall$-subject_-$A\exists!\Delta$: $\mathbb{R}_0^+$_powerset;
$\Delta\forall$-action_-IO: $\oplus$; $\Delta\forall$-object_-IO: 18 }

(E_2,R): {$\Delta\forall$-subject_-IO: 13; $\Delta\forall$-subject_-$A\exists!\Delta$: $\mathbb{R}^+$_powerset;
$\Delta\forall$-action_-IO: $\oplus$; ΔF-object_-$F\Delta$: Ages }

which defines adolescents as the subset of people whose age is in $[13, 18[$.

◇ Objective Absolute Ranges

Determining the minimum and maximum points of an absolute range is needed for expressions such as *"John is the tallest man I ever met"*. A first attempt would be, if $\mathcal{V}_1$ is a type of positive value:

(E_3,R): {ΔI-subject_-IO: $\mathcal{V}_1$; ΔI-subject_-$I\Delta$: $\mathcal{V}_2$; ΔI-action_-IO: synonym_}

(E_4,R): {$\Delta\forall$-subject_-$\exists!$O: $\mathcal{V}_2$; $\Delta\forall$-subject_-$I\Delta$: minimum_;
$\Delta\forall$-action_-IO: $\oplus$; ΔF-object_-FO: $\mathcal{V}_1$}

(E_5,R): {$\Delta\forall$-subject_-$\exists!$O: $\mathcal{V}_2$; ΔF-subject_-FO: $\mathcal{V}_1$;
$\Delta\forall$-action_-IO: $\oplus$; $\Delta\forall$-object_-$I\Delta$: maximum_}

In other words, minimum_ is the lowest value of the range:

$$\forall v \in \mathcal{V}_1\ \exists \delta \in \mathcal{V}_1\ .\ \texttt{minimum\_} + \delta = v$$

Similarly, maximum_ is the highest value of the range:

$$\forall v \in \mathcal{V}_1\ \exists \delta \in \mathcal{V}_1\ .\ v + \delta = \texttt{maximum\_}$$

The problem is that δ may not be small: minimum_+δ may actually be greater than maximum_ (if maximum_ < minimum_+minimum_). What is needed is a δ which is not within $\mathcal{V}_1$, corresponding to the smallest difference between any two values of $\mathcal{V}_1$. Such a value in itself would be ridiculous, if a concept were assumed to have that value. For instance a person-height of 0.2mm. However, such things are imaginable, if not realistic. Frames of existence (D.6.4 *(p. D-83)*) deal with precisely this distinction. Thus δ can belong to the frame-less set of values (in this case the heights of people I have met – including in my dreams), while $\mathcal{V}_1$ is its subset in the real frame of existence.

Another objection one might have, is that all values are specializations of $\mathcal{V}$, which is defined by $\oplus$'s template event. Should this not mean that all value ranges must include 0 and ∞? However, just as one can define $\mathbb{R}_0^+$, by stating that 0 is not in $\mathbb{R}_0^+$, one can state that some value range does not include ∞ – so all values are between `minimum_` and `maximum_`.

If $\mathcal{V}_1$ can have some negative values, the δs must all be positive. If $\mathcal{V}_1$ does not include zero, then the statements become:

$$\forall v \in (\mathcal{V}_1 - \{\texttt{minimum\_}\})\ \exists \delta \in \mathcal{V}_1\ .\ \texttt{minimum\_} + \delta = v$$

$$\forall v \in (\mathcal{V}_1 - \{\texttt{maximum\_}\})\ \exists \delta \in \mathcal{V}_1\ .\ v + \delta = \texttt{maximum\_}$$

○ Subjective Values

◇ Network Realization

Some ranges of values correspond to sensations or perceptions which depend on the agent which experiences them. Such values cannot be interpreted as existing on some absolute scale, since this would mean that if two people disagreed about, say, the beauty of some object, one of them must be wrong. Instead these events must be interpreted with respect to the agent perceiving them.

There are two ways the agent perceiving some value can be referred to. The first is when the agent is believed to experience the perception, as stated by his believing the value judgement. The second is when the agent communicates the value judgement. In this second case he need not believe it, as shown by the sentence *"Margaret says Susan is nasty"*. Other stative events taking as `subject_` an agent, may relate that agent and the value judgement. For instance, *"John pretends Susan is nasty"*.

The set of stative events on which subjective value events can depend is a set of events with generalized action `subjective_value_connector`. Since all subjective value events depend on a connection to the agent making the value judgement to have any meaning, they are defined always to be connected to such an event: the template of all subjective value events E_0 is observed always to be associated with such an event (E_1). This means that if LOLITA believes in some subjective value, she must also be the `subject_` of a stative event. This additional cost can be reduced by sharing the "LOLITA `subjective_value_connector`" event using inheritance. Notice that E_1 is also a template event.

(E_0,R): $\{\mathrm{O}\boxed{\forall}\forall$-`subject_`-$\boxed{\exists!}\Delta$: $\mathcal{W}$; $\Delta\forall\forall$-`action_`-$I\mathrm{O}$: `has_subjective_value`;
$\mathrm{O}\boxed{\forall}\forall$-`object_`-$\boxed{\exists!}\Delta$: $\mathcal{V}\}$

(E_1,R): $\{\mathrm{O}\forall$-`subject_`-$\exists!\Delta$: $\mathcal{A}$; $\Delta\forall$-`action_`-$I\mathrm{O}$: `subjective_value_connector`;
$\mathrm{O}F$-`object_`-$FF\mathrm{O}$: $E_0\}$

where the set of agents that can be the `subject_` of a `subjective_value_connector` template event is $\mathcal{A}$, $\mathcal{W}$ is the set of things that can have some subjective value, and $\mathcal{V}$ is the set of such values. All the elements of E_0 are observational `spec_s` of `has_value`.

Although `subjective_value_connector` events are stative, not all stative events should be its descendents: this would imply that all stative events could only take `has_subjective_value` events as `object_s`, since the template event E_1 has as observational `object_` E_0 ($\mathrm{O}F$-`object_`-$F\mathrm{O}$). Thus, `subjective_value_connector` must be a superset of subsets of the relevant stative events.

There could be more than one stative event connected to a subjective value event, for instance via a chain. Since stative events describe the event they are connected to, only the events connected directly to the subjective value event should be considered. If more there are more than one such events, the subjective value event is shared by many agents. That is to say many agents expressed or believed the same value judgement.

◇ Subjective Absolute Ranges

Just as an objective value can have an absolute range, so can a subjective range. The subjective values are defined by the relevant event below `has_subjective_value` associated with a particular agent: From this, the minimum and maximum value for that agent can be defined.

D.1.3.3 The use of values in language

Many adjectives and adverbs are naturally expressed as values. In particular all those that have a comparative or superlative form, either as a single word or in a construction involving adverbs of intensity such as *"more"*, can be represented using values: values represent the notion of gradation adjectives and adverbs express. However, some nouns and verbs are also easily represented using values.

• Nouns, adjectives, adverbs and verbs

∘ Adjectives and adverbs

Adjectives and adverbs are both used to qualify other concepts. Adjectives qualify concepts expressed as nouns at the grammatical level, whereas adverbs qualify concepts expressed as verbs. But this grammatical distinction is not mirrored at the semantic level. An example illustrates this well: *"The bomb exploded loudly last night, and broke my window"* and *"The loud explosion last night broke my window"*. Although the two sentences differ grammatically, it is clear that their meanings do not. Both forms are represented by an event with `action_` explode and `subject_` bomb, and are the `subject_` of the breaking of my window event. It would be rather strange therefore to represent *"loud"* and *"loudly"* differently. They both express the intensity of sound associated with the event, and are represented by the appropriate value type and specialization of the `has_value` event.

Although adjectives and adverbs may correspond to the same concept when they qualify an event, adjectives may have other meanings when qualifying other things. For instance, *"Jane has a loud voice"* does not mean that her voice is associated with a large intensity of sound, all the time whether or not she uses it. Instead it means there is a sound associated with Jane talking, and this always has a strong intensity. The meaning of the adjective loud therefore differs when it qualifies an event (loud_1) and a voice, instrument or piece of machinery (loud_2).

∘ Nouns and verbs

Although one major use of values in language involves adjectives and adverbs, verbs may also express values. For instance *"John weighs 80kgs"* is directly represented as a `has_weight` event mapping concrete objects[6] to the weight value type. Although grammatically this construction differs significantly from *"John is 2 metres tall"*, it shares the same semantic representation. Indeed, the choice of grammatical

[6]Note that it is not the node representing John that is the `subject_` of a `has_weight` event, but the node representing his body.

construction depends on the language: in French the sentence would be *"Jean mesure 2 mètres"*. Such statements also allow comparative forms *"John weighs more than I do"*.

Verbs also may also include implicit references to values, for instance *"to lack"*, *"to grow"*, or *"to multiply"*. Many nouns also involve values, for instance *"a lack of"*, *"richness"*, *"a surplus"*, *"growth"* and so on.

- **Net representation**

As described previously, adjectives and adverbs are represented using events which are specializations of `has_value`. These associate the concept being qualified with the value qualifying it, and provide the appropriate type restrictions. This section discusses particular constructions in natural language which must be represented.

∘ **Relational and extensional adjectives and adverbs**

As discussed in D.1.3.2 *(p. D-16)*, the values expressed by certain adjectives depend on the expected values for the concept they qualify. An example of such a relational adjectives is *"tall"*: *"John is a tall man"* states that John's height is greater than men's average height. Such adjectives therefore correspond to a segment of semantic net:

(E_0,R): $\{\Delta I$-`subject_`-IO: `John`; ΔI-`action_`-IO: **`has_height`**; ΔI-`object_`-$I\Delta$: $h\}$

(E_1,R): $\{\Delta F$-`subject_`-FO: `Men`; $\Delta\forall$-`action_`-IO: **`has_height`**; ΔF-`object_`-$F\Delta$: $\mathcal{H}\}$

(E_2,R): $\{\Delta I$-`subject_`-IO: `Men`; ΔI-`action_`-IO: `inst_`; ΔI-`object_`-$I\Delta$: `John`$\}$

(E_3,R): $\{\Delta I$-`subject_`-IO: `Men`; ΔI-`action_`-IO: `size_`; ΔI-`object_`-$I\Delta$: $s\}$

(E_4,R): $\{\Delta I$-`subject_`-IO: x; ΔI-`subject_`-$A\Delta$: $\mathbb{R}_0^+$; ΔI-`action_`-IO: $\oplus$; ΔI-`object_`-IO: $h\}$

(E_5,R): $\{\Delta I$-`subject_`-$I\Delta$: x; ΔI-`subject_`-IO: s; ΔI-`action_`-IO: $\otimes$; ΔI-`object_`-IO: $y\}$

(E_6,R): $\{\Delta I$-`subject_`-$\forall O$: $\mathcal{H}$; ΔI-`action_`-IO: $\oplus$; ΔI-`object_`-$I\Delta$: $y\}$

Note that the subset of tall people can be built, so that if John and Mary both happen to be tall, these events need not be repeated for each. The cost of building a new tall person is thereby limited to one instance event.

Some adjectives simply state that the concept they qualify is associated with a certain value independently from the expected value for that concept. An example of such adjectives is *"red"*: *"The book is red"* does not refer to any expected value, but only to the sensation of red that sight of the book provokes. Such adjectives are called extensional, and are represented by a single event and the corresponding value:

(E_0,R): $\{\Delta I$-`subject_`-IO: `Elephant`; ΔI-`action_`-IO: `pinkness`; ΔI-`object_`-$I\Delta$: $p\}$

It is of particular interest that extensional adjectives can be inherited from supersets. This means that concepts such as pink elephants can be built only with subset relations: it is the intersection of pink things and elephants. This saves network space as only one node is used instead of four.

◦ **Adverbs of intensity**

Adverbs like *"quarter"*, and *"half"* reduce the amount the value represents: *"I'm half happy about it"*, *"Half the bottle is left"*, *"Half the cows ran away"*. They are represented using multiplication: if v is the amount of liquid in the bottle, then $v\times$`half` expresses *"Half the bottle"*[7] External multiplication is used if the values are uncountable: either unquantifiable, or have no natural units. Otherwise internal multiplication is used.

And adverbs like *"very"* that change the range to which the qualifying value belongs: *"That's a very red tie!"*. Whereas a red tie may be any reddish colour, a very red tie is a particularly intense colour of red. Such intervals are represented by dividing the range of values into intervals corresponding to sensations which could be described by the adverbs *"very"*, *"quite"*, *"a little"*, etc. Such a scheme requires the minimal and maximal limits of the perceptual range to be known. As discussed in D.1.1.3 *(p. D-12)* the minimal bound is zero. Similarly, as discussed in D.1.3.2 *(p. D-16)* there is a maximum value, simply defined in a particular value range as the value greater than all others. The ranges of the adverbs of intensity are defined using special external unquantified values, and multiplication: `very`, `quite`, etc...

[7]See D.1.1.2 *(p. D-8)* for more information on conversions

These values are defined externally to the net.

Because of the way in which the multiplier values are used to build new range from an old one, they can be applied more than once, as in *"I saw a very very small elephant today"*. Here the *"very very"* restricts the range to an even more extreme form. Similarly, the order in which the adverbs are placed can be accounted for: *"I saw a very short tall man"* and *"I saw a very tall short man"* correspond to different height ranges.

It might be thought that defining the maximal value perceivable would be problematic for non-literal uses of sensations, such as *"the sun is very hot"*, where the notion of heat lies outside the range of sensory perception. However it should be remembered that such events are non-literal, and when interpreted are represented in terms of the abstract sister concept of temperature. Very hot in normal conversation would refer in this domain to any temperature greater than the one defined as lower limit for the perception of very hot.

∘ **Comparatives and superlatives**

Comparative constructions relate two values in a total order: *"John is older than Mark"* would be expressed as:

(E_0,R): $\{\Delta I$-`subject_`-IO: `John`; ΔI-`action_`-IO: `has_age`; ΔI-`object_`-$I\Delta$: $x\}$

(E_1,R): $\{\Delta I$-`subject_`-IO: `Mark`; ΔI-`action_`-IO: `has_age`; ΔI-`object_`-$I\Delta$: $y\}$

(E_2,R): $\{\Delta I$-`subject_`-IO: y; ΔI-`subject_`-$A\Delta$: $\mathbb{R}_0^+$; ΔI-`action_`-IO: $\oplus$; ΔI-`object_`-IO: $x\}$

`has_age` is a specialization of `has_value`.

Superlatives on the other hand express a total order between a object's value and that of all other objects of that kind. For instance *"John is the oldest man in the laboratory"*. The set of men $\mathcal{M}$ in the laboratory other than John, and the set of their respective ages $\mathcal{A}$ is defined. Then the superlative can be expressed by stating that it's value is greater than any of the set of values of the objects with which it is compared:

(E_0,R): $\{\Delta I$-`subject_`-IO: `John`; ΔI-`action_`-IO: `has_age`; ΔI-`object_`-$I\Delta$: $x\}$

(E_1,R): $\{\Delta F$-`subject_`-FO: $\mathcal{M}$; ΔI-`action_`-IO: `has_age`; $\Delta\forall$-`object_`-$\exists!\Delta$: $\mathcal{A}\}$

(E_2,R): $\{\Delta F\text{-}\texttt{subject\_}\text{-}FO\text{: } \mathcal{A};\ \Delta\forall\text{-}\texttt{subject\_}\text{-}A\exists!\Delta\text{: } \mathbb{R}_0^+\texttt{\_powerset};$
$\Delta\forall\text{-}\texttt{action\_}\text{-}IO\text{: } \oplus;\ \Delta\forall\text{-}\texttt{object\_}\text{-}IO\text{: } x\}$

(E_3,R): $\{\Delta\forall\forall\text{-}\texttt{subject\_}\text{-}IO\text{: } \texttt{John};\ \Delta F\forall\text{-}\texttt{subject\_}\text{-}F\Delta\text{: } \mathcal{M};$
$\Delta\forall\forall\text{-}\texttt{action\_}\text{-}IO\text{: } \texttt{synonym\_}\}$

(E_4,R): $\{\Delta F\text{-}\texttt{subject\_}\text{-}F\Delta\text{: } E_3;\ \Delta\forall\text{-}\texttt{action\_}\text{-}IO\text{: } \texttt{size\_};\ \Delta\forall\text{-}\texttt{object\_}\text{-}IO\text{: } \texttt{0}\}$

Note that this states that there is no one in the laboratory of the same age as John. If this is not known, a milder form must be used which allows for scenarios such as *"joint oldest"*: identical to the scheme above, except without E_3 and E_4, and $\mathbb{R}_0^+$_powerset is replaced by $\mathbb{R}^+$_powerset.

Similarly, superlatives involving an order can be built, as in *"The third oldest building in Cambridge"*. This fits the general pattern "nth x-est y". To represent this, first the values $\mathcal{V}$ corresponding the characteristic x are considered. All instantiations of y have an associated value in $\mathcal{V}$. $\mathcal{V}$ is then split into two parts $\mathcal{G}$ and $\mathcal{L}$ using the antonym_ event, where $\mathcal{G}$ is defined as having n elements, and all its elements must be greater than any of $\mathcal{L}$'s. This in effect creates the set of values corresponding to the "n x-est ys". Obtaining the "nth x-est y" is done simply by defining it to be the smallest element of $\mathcal{G}$.

(E_0,R): $\{\Delta F\text{-}\texttt{subject\_}\text{-}FO\text{: } y;\ \Delta\forall\text{-}\texttt{action\_}\text{-}IO\text{: } \texttt{has\_}x;\ \Delta\forall\text{-}\texttt{object\_}\text{-}\exists!\Delta\text{: } \mathcal{V}\}$

(E_1,R): $\{\Delta I\text{-}\texttt{subject\_}\text{-}I\Delta\text{: } z;\ \Delta I\text{-}\texttt{action\_}\text{-}IO\text{: } \texttt{has\_}x;\ \Delta I\text{-}\texttt{object\_}\text{-}IO\text{: } v\}$

(E_2,R): $\{\Delta\forall F\text{-}\texttt{subject\_}\text{-}F\Delta\text{: } \mathcal{L};\ \Delta\boxed{\forall\forall}\text{-}\texttt{subject\_}\text{-}A\boxed{\exists!}\Delta\text{: } \mathbb{R}_0^+\texttt{\_powerset};$
$\Delta\forall\forall\text{-}\texttt{action\_}\text{-}IO\text{: } \oplus;\ \Delta F\forall\text{-}\texttt{object\_}\text{-}F\Delta\text{: } \mathcal{G}\}$

(E_3,R): $\{\Delta I\text{-}\texttt{subject\_}\text{-}I\Delta\text{: } \mathcal{G};\ \Delta I\text{-}\texttt{action\_}\text{-}IO\text{: } \texttt{size\_};\ \Delta I\text{-}\texttt{object\_}\text{-}IO\text{: } n\}$

(E_4,R): $\{\Delta I\text{-}\texttt{subject\_}\text{-}IO\text{: } [\mathcal{G},\mathcal{L}];\ \Delta I\text{-}\texttt{action\_}\text{-}IO\text{: } \texttt{antonym\_};\ \Delta I\text{-}\texttt{object\_}\text{-}IO\text{: } \mathcal{V}\}$

(E_5,R): $\{\Delta\forall\text{-}\texttt{subject\_}\text{-}I\Delta\text{: } v;\ \Delta\forall\text{-}\texttt{subject\_}\text{-}A\exists!\Delta\text{: } \mathbb{R}^+\texttt{\_powerset};$
$\Delta\forall\text{-}\texttt{action\_}\text{-}IO\text{: } \oplus;\ \Delta F\text{-}\texttt{object\_}\text{-}FO\text{: } \mathcal{G}\}$

The spec_ events from $\mathcal{V}$ to $\mathcal{L}$ and $\mathcal{G}$ have not been included above. Also note that the scheme above allows two joint *"third oldest buildings"*. Here z corresponds to the n'th x-est y, and v is its x-value.

- **Interpretation**

Little work has been done in this area, but initial findings suggest that many values are expressed in a variety of standard grammatical constructions such as comparative forms, forms using intensity adverbs and value related nouns followed by *"of"* and then the value itself or the concept being described by the value: *"a surplus of money"*, *"too much food"* or even *"The engine burned two tons of coal"* where the interpretation process must determine that it is the coal and not the tons that are being burned[8]. If such standard constructions for values exist, they will allow a few general rules to interpret a wide variety of expressions. However showing this by experimental work with corpora is beyond the scope of this thesis.

D.1.3.4 Belief and certainty

As discussed in sections A.2.3.1 *(p. A-12)* and A.2.3.3 *(p. A-15)*, statements are associated with a certain degree of belief and certainty. These degrees are represented using values. This allows an order to be expressed between the degrees of certainty and belief of various events: *"I would believe more that John's death was accidental than that it was suicide."*; *"I would be more certain that I did not offend him, if he hadn't been so stiff afterwards"*. Thus, each event – other than the `belief_value` and `certainty_value` events themselves – is associated with its own belief and certainty value. These values are unquantified. Since belief and certainty are based on personal experience they are agent-dependent values, or subjective values.

But values also allow belief and certainty to be classified into broad levels: low, medium, high... Such levels prove useful when plausible reasoning algorithms assign the degree of confidence they have in statements they have derived. For an example, see [Long et al. 93], where the analogy algorithm determines the certainty for all the statements it derives. Each such level is represented by a set of the values included in that level. These sets are defined as intervals, where the interval boundaries can be unquantified values defined externally. Thus for each event in the network there are belief and certainty values, but they need not all be assigned a separate node.

Alternatively, certainty values can be partitioned into n levels by determining the

[8]unlike *"I like the dog of my mother"*, where it is not the mother being liked

minimum m and maximum of their range (see D.1.3.2 *(p. D-21)*). The difference δ between these two values can then be multiplied by $\frac{1}{n}$, to obtain the spacing s of values. This spacing can then be used to create sets of values fitting within the ranges $[m+0\ldots m+\frac{\delta}{n}[$, $[m+\frac{\delta}{n}\ldots m+\frac{2\delta}{n}[$, $[m+\frac{2\delta}{n}\ldots m+\frac{3\delta}{n}[$, …, $[m+\frac{(n-1)\delta}{n}m+\delta]$. Belief and certainty currently have 3 levels: low, medium and high.

Having a `belief_value` and a `certainty_value` event for each other event would lead to an unacceptable increase in net size. The solution adopted is to use arbitrary quantification. If E_1 is an event that has a high belief value, then it can share E_2 with all other high belief value events:

(E_2,R): $\{\Delta A$-`subject_`-IO: $[E_1, \ldots]$; $\Delta\forall$-`action_`-IO: `belief_value`;
$\Delta\forall$-`object_`-$\exists!O$: $\mathcal{HBV}\}$

E_2 has many `subject_` arcs connected to it, but none of them are partial since arbitrary quantification never refers to the same element of a set more than once. $\mathcal{HBV}$ is the set of high belief values.

D.2 Negation

D.2.1 Use of Absence of Occurrence

D.2.1.1 Naturalness of Absence of occurrence

Using non-existence allows a symmetrical treatment of sentences such as *"John thinks there are five apples in the basket, but Jane thinks there are none"*. This is represented as:

(E_0,R): $\{\Delta\forall$-`subject_`-IO: `basket`; $\Delta\forall$-`action_`-IO: `contains`;
ΔF-`object_`-$F\Delta$: `apples_in_basket`$\}$

(E_1,H): $\{\Delta I$-`subject_`-IO: `apples_in_basket`; ΔI-`action_`-IO: `size_`;
ΔI-`object_`-IO: `0`$\}$

(E_2,H): $\{\Delta I$-`subject_`-IO: `apples_in_basket`; ΔI-`action_`-IO: `size_`;
ΔI-`object_`-IO: `5`$\}$

(E_3,R): $\{\Delta I$-`subject_`-IO: `Jane`; ΔI-`action_`-IO: `think`; ΔI-`object_`-IO: $E_1\}$

(E_4, R): {ΔI-subject_-IO: John; ΔI-action_-IO: think; ΔI-object_-IO: E_2}

Notice that the size events are both hypothetical, dependent on John's and Jane's respective beliefs. This symmetry has to be contrasted with the form that would occur if a *"does not contain"* action were used instead of the size events: John believes that the basket contains a subset of apples, and that the number of these apples is five, whereas Jane believes that every apple that exists, has existed or will exist is the object_ of a *"does not contain"* event whose subject_ is the basket. This asymmetry requires a lot of processing to maintain, and is therefore unnatural:

(E_0, H): {$\Delta\forall$-subject_-IO: basket; $\Delta\forall$-action_-IO: contains;
ΔF-object_-$F\Delta$: apples_in_basket}

(E_1, H): {$\Delta\forall$-subject_-IO: basket; $\Delta\forall$-action_-IO: does_not_contain;
ΔF-object_-$F\Delta$: apples}

(E_2, H): {ΔI-subject_-IO: apples_in_basket; ΔI-action_-IO: size_;
ΔI-object_-IO: 5}

(E_3, R): {ΔI-subject_-IO: Jane; ΔI-action_-IO: think; ΔI-object_-IO: E_1}

(E_4, R): {ΔI-subject_-IO: John; ΔI-action_-IO: think;
ΔI-object_-IO: [E_0,E_2]}

D.2.1.2 Defined Absence and Uniqueness

All the forms of absence of occurrence discussed so far referred to events which were observed not to have happened. However, one also wants to be able to represent sentences that define concept by the absence of some feature. For instance, *"The people that John never met"* refers to all people who were never involved in a meeting event involving John.

Consider first the positive form of the statement, *"The people that John met"*:

(E_5, R): {$\Delta\forall$-subject_-IO: John; $\Delta\forall$-action_-IO: meet;
$\Delta\forall$-object_-$\exists!\Delta$: people$_1$}

This is equivalent to:

(E_6, R): $\{\Delta\forall\forall$-`subject_`-IO: `John`; $\Delta\forall\forall$-`action_`-IO: `meet`;

$\Delta\boxed{\forall}\forall$-`object_`-$\boxed{\exists!}\Delta$: $\texttt{people}_1\}$

(E_7, R): $\{\Delta F$-`subject_`-$F\Delta$: E_6; $\Delta\forall$-`action_`-IO: `size_`; $\Delta\forall$-`object_`-IO: `1`$\}$

The definitional E_7 ensures only instantiations of E_6 with one element are selected from E_6's parent E_8. Throughout this thesis, we have talked somewhat loosely about sets, and sets of sets. Strictly speaking SemNet's concepts are not sets, since they are defined intensionally. Similarly, rather than speaking of sets of sets, one should refer to groupings of instantiations. Usually, this does not matter, but in this case the distinction is critical. Indeed, just as E_7 could define E_6 by selecting only instantiations with cardinality one from E_8, so E_{10} can select only instantiations of E_8 with cardinality zero: each element of E_9 is a concept which happens not to have any instantiations:

(E_9, R): $\{\Delta\forall\forall$-`subject_`-IO: `John`; $\Delta\forall\forall$-`action_`-IO: `meet`;

$\Delta\boxed{\forall}\forall$-`object_`-$\boxed{\exists!}\Delta$: $\texttt{people}_2\}$

(E_{10}, R): $\{\Delta F$-`subject_`-$F\Delta$: E_9; $\Delta\forall$-`action_`-IO: `size_`; $\Delta\forall$-`object_`-IO: `0`$\}$

$\texttt{people}_2$ is thus defined to be the set of people whose meeting relation with John is an empty concept. Admittedly the notion that "something" can be restricted by "nothing" appears strange at first, but this stems from identifying concepts with mathematical sets: empty concepts still are "something", in that they are distinct, and in that cardinality just happens to be one of their properties; whereas cardinality is tightly bound up with the definition of sets, and empty sets are equal. Moreover it appears easy to build an empty set in mathematics, but in SemNet a concept will only have empty concepts as instantiations if such empty concepts occur – just like a concept will only have instantiations of concepts such as cat, if such instantiations occur in the agents environment: in SemNet, an empty set is not a construction.

This treatment proves natural as the same process can deal with concepts expressing multiple level groupings. Thus, for instance, the same negation process is involved the negation *"John believes the basket contains no apples"* as in "John believes all of the baskets in that room contain no apples".

There is currently another way of defining $\texttt{people}_2$: $\texttt{people}_2$ is the `antonym_` of `people`$_1$ with respect to the concept of all people. This is due to `antonym_` breaking uniqueness: see D.6.3.3 *(p. D-80)*.

D.2.2 Problems in Performing Negation

D.2.2.1 Conversion to antonym depends on quantification and sorts

Conversion to antonym in many cases requires changes to the quantification of the arguments of the event being converted. This in turn can interact with sortal information, rendering conversion to antonym a non-local phenomenon. First the interaction with quantification is illustrated. An example leads to the revelation that sortal information can be affected. Other difficulties are illustrated.

The purpose of this section is to illustrate the problem, but not provide an actual algorithm to solve it. The actual algorithm for negating statements is a form of reasoning and therefore outside the scope of this thesis. However the ability to express negation is an important aspect of any representation, and should be discussed.

- **Conversion to antonym and quantification**

The conversion to antonym of an event requires a change of quantification of its arguments. For instance the conversion to antonym of *"a man likes a fish"* is *"all men dislike all fish"*. If the conversion were only *"a man dislikes a fish"*, the sentence *"a man likes a fish"* could still be true. This intuition forms the basis of the treatment of negation in classical logic, which corresponds to conversion to antonym in LOLITA.

In classical $\mathcal{FOPL}$, predicates can take three types of argument: constants, existentially quantified variables, and universally quantified variables. When negated, the existential and universal quantifications associated with variables are swapped, but the constants do not change. This swap accounts not only for the example given above, but also for the invariance of the term *"Sengan"* in the negation of such statements as *"Sengan likes a fish"*, which becomes *"Sengan dislikes all fish"*. A similar distinction between the types of arguments of LOLITA's events is needed for successful conversion to antonym.

In LOLITA, the arguments of events are qualified by quantification and by sorts. To negate events, these arguments must be resolved into their equivalent logical type. One might think that this could be inferred from the three forms of quantification, individual, existential and universal. A suitable example can falsify this hypothesis. The sentence *"Sengan likes a fish"* is chosen as it contains an example both of a logical constant (Sengan) and an logical existentially quantified variable (a fish):

(E_0,R): $\{\Delta I$-`subject_`-IO: `Sengan`; ΔI-`action_`-IO: `like`;
ΔI-`object_`-$I\Delta$: $\texttt{fish}_1\}$

(E_1,R): $\{\Delta I$-`subject_`-IO: $\texttt{fish}_0$; ΔI-`action_`-IO: `inst`;
ΔI-`object_`-$I\Delta$: $\texttt{fish}_1\}$

As both Sengan and the fish are qualified by an individual quantification, quantification alone does not differentiate between logical constant and logical variable.

- **Conversion to antonym and sorts**

If LOLITA's quantification does not distinguish between logical constants, and logical existentially quantified variables, perhaps the other qualification of events' arguments will: sorts. Indeed, in the example *"Sengan likes a fish"*, the event E_0 is observational with respect to `Sengan`, but is definitional with respect to the fish ($\texttt{fish}_1$). This makes sense: if a constant is defined, it will be defined outside the formula in which it is used. A variable on the other hand is restricted by the formula in which it is used. Sengan is not defined by his liking a fish, so with respect to the sentence he is a constant. The fish on the other hand is defined by the sentence – it is the fish that Sengan likes, as is therefore dependent on it for its meaning. Conversion to antonym must therefore use sortal information.

- **Conclusion**

Conversion to antonym must take both quantificational and sortal information into account when negating an event.

D.2.2.2 Preservation of event arity

Event arity denotes the number of `subject_s` and `object_s` an event may have. The notion of full and partial arcs has already been introduced in 5.1.2.4 *(p. 123)*: *"John carried the piano"* has a full `subject_` arc, and `subject_` arity one; whereas *"John and Bill carried the piano"* has a partial `subject_` arc and `subject_` arity two; and *"John, Bill, and George carried the piano"* has a partial `subject_` arc but `subject_` arity three. Further, as discussed in 5.4.3.2 *(p. 151)*, one cannot always infer an event of lower arity from an event of higher arity. For instance, one cannot infer from the sentence *"Roberto, Rick and Russell own the 3R company"* that Roberto owns the 3R company. This means that event arity should be preserved by conversion to antonym.

Requiring event arity to be preserved makes conversion to antonym a non local phenomenon. For instance, consider the negation $\neg E_0$ of the event *"Sengan dislikes all fish"*. This would be written:

$$(\neg E_0, R)\text{: } \{\Delta\forall\text{-}\mathtt{subject\_}\text{-}IO\text{: }\mathtt{Sengan}\text{; }\Delta\forall\text{-}\mathtt{action\_}\text{-}IO\text{: }\mathtt{dislike}\text{;}$$
$$\Delta F\text{-}\mathtt{object\_}\text{-}FO\text{: }\mathtt{fish}_0\}$$

Whereas in the natural language form, only one argument suffers a change of quantification (*"all fish"*), the quantification of all arguments was changed by the negation of E_0. This corresponds to the fact there is now for each fish one event of Sengan disliking it. Had the change been restricted to the `object_` arc, only one event of Sengan disliking all the fish together[9] would have resulted. This would have contravened the preservation of arity.

Conversion to antonym must preserve the arity of its events. This results in non-local changes of quantification which preclude treatment of negation based solely on individual arcs.

D.2.2.3 The treatment of defined individuals

The conversion to antonym of the event *"Sengan likes a fish"* was the event *"Sengan dislikes all fish"*, $\neg E_0$. Assume now that one wants to obtain the conversion to

[9] which is meaningless

antonym of the latter event. One would expect to obtain the event: *"Sengan likes a fish"*. Indeed this is sufficient to render $\neg E_0$ false, and it incorporates the other cases which render $\neg E_0$ false: *"Sengan likes two fish"*, *"Sengan likes three fish"*... One would however be mistaken to represent it by E_0. Indeed, E_0 defines $\texttt{fish}_1$ to be the (one and only) fish that Sengan likes. This is because expressing concepts as individuals defined by some arcs is a shorthand. It avoids writing that a concept is a set observed to have size one. Using E_0 would therefore have been a stronger statement than intended.

Since the problem lies in the definitional sort, one might want to try an event *"Sengan likes a fish"* which is observational with respect to the fish. However this breaks the rules ensuring conceptual uniqueness. The fish would only be defined by a single definitional event. Thus a definitional sort must be used, but not an individual quantification. This leads to the arity preserving solution:

(E_0,R): $\{\Delta\forall$-`subject_`-IO: `Sengan`; $\Delta\forall$-`action_`-IO: `like`;
ΔF-`object_`-$F\Delta$: $\texttt{fish}_1\}$

(E_1,R): $\{\Delta I$-`subject_`-IO: $\texttt{fish}_0$; ΔI-`action_`-IO: `spec`;
ΔI-`object_`-$I\Delta$: $\texttt{fish}_1\}$

(E_2,R): $\{\Delta I$-`subject_`-IO: $\texttt{fish}_1$; ΔI-`action_`-IO: `size`;
ΔI-`object_`-$I\Delta$: `fish_size`$\}$

where `fish_size` is stated to be greater or equal to one by an observational values event.

D.2.2.4 Difficulties associated with unique existentials

Unique existentials prove to be another cause of difficulty in the conversion to antonym. For instance, take *"Every man likes a woman"*, where each man likes one woman only. One would expect its conversion to antonym to be *"There is a man who dislikes all women"*, but in fact it is *"Either there is a man who dislikes all women, or there is a man who likes more than one woman, or both"*. In other words, the postulated uniqueness and the liking can both be negated independently. This means that negation of such events can become very complicated. For instance:

(E_0,R): $\{\Delta F$-subject_-FO: $\texttt{men}_0$; $\Delta\forall$-action_-IO: like;
$\Delta\forall$-object_-$\exists!\Delta$: $\texttt{women}_1\}$

where $\texttt{women}_1$ is a definitional spec_ of $\texttt{women}_0$, becomes (without some of the size_ events)

(E_1,H): $\{\Delta F\forall$-subject_-$F\Delta$: $\texttt{men}_1$; $\Delta\forall\forall$-action_-IO: dislike;
$\Delta\forall F$-object_-FO: $\texttt{women}_0\}$

(E_2,H): $\{\Delta F$-subject_-$F\Delta$: $\texttt{men}_2$; $\Delta\forall$-action_-IO: like;
$\Delta\forall$-object_-$\exists!\forall\Delta$: $\texttt{women}_2\}$

(E_3,R): $\{\Delta I$-subject_-$\forall\forall$O: E_1; ΔI-subject_-$\forall$O: E_2;
ΔI-action_-IO: or_$\}$

(E_4,R): $\{\Delta F$-subject_-FO: $\texttt{women}_2$; $\Delta\forall$-action_-IO: size;
$\Delta\forall$-object_-$\exists!\Delta$: $\texttt{sizes}_1\}$

where $\texttt{sizes}_1$ are observed to be greater or equal to two and are a definitional spec_ of set_sizes; $\texttt{men}_1$ and $\texttt{men}_2$ are definitional spec_s of $\texttt{men}_0$; $\texttt{women}_2$ is a definitional spec_ of $\texttt{women}_0$. Notice that the further complication due to the definitional sort for $\texttt{men}_1$, results in E_1 being a set of sets of events (set level 2).

Further, conversion to antonym must ensure that its results are legal. This might occur when negating $\exists!\forall$. In the representation this is equivalent to $\exists$. A naive implementation might convert *"Everyman likes at least one woman"*, where the set of women is referred to by a $\exists!\forall$, to *"There is a man who dislikes all women, or there is a man who likes more than one set of women, or both"*. This conversion occurs because it is the set that was stated to be unique. Writing the second part of the statement in the network is however illegal: definitional sets are maximal, so each of the two sets (instantiations) would be maximal. But concepts do not include the same instantiation twice. Thus only the first part of the statement may legally be written. This first part would also have been obtained had an $\exists$ quantification been used instead of the $\exists!\forall$, assuming that SemNet included it.

D.2.2.5 Effect of the network being propositional

SemNet is propositional. This adds two more difficulties to conversion to antonym. Events which refer to the event being negated constitute the first problem. The second problem occurs when the event being negated defines an argument which is an event.

○ Events referring to an event to be negated

Events that are negated are often qualified by other events. What happens to these events? The events may either be definitional or observational with respect to the event being negated. The problem can be broken down into two parts, with this distinction.

Events are little different to arcs with respect to the event. Indeed, as pointed out in B.1.2.4 *(p. B-6)*, an event's arcs behave like just like other events to it. The key point is that an arc behaves like an event as far as the event of which it is the source is concerned: it states that the event is in a particular relationship with another concept. However, for arcs' destinations, they are only parts of events, and it is the full event that must be considered. Because of this, definitional events are treated just as definitional arcs are: they are included in the events' negation. The only exception being that the definitional action arc has its destination replaced by the antonym action. Because of this, the definition of the event changes, and the observational events and arcs are not connected to the event converted to antonym.

○ Events defined by an event to be negated

The problem of events which are negated and define others is no different to entities defined by an event which is negated. For instance, an event which is defined as *"the things John believes Roberto did on the night of the 15th June"* is extensionless, if John disbelieves every event involving Roberto doing anything on the 15th of June: an extension would create a contradiction. The actual negation process is no different from any other node completely or partially defined by an event which is being negated. Few of the events presented in this thesis taking events as arguments have antonyms. "Believe" is one of the few exceptions.

(E_0,R): $\{\Delta\forall$-`subject_`-IO: `John`; $\Delta\forall$-`action_`-IO: `believe`;
ΔF-`object_`-$F\Delta$: $E_1\}$

(E_1,H): $\{\Delta\forall$-`subject_`-IO: `Roberto`;$\}$

(E_2,R): {ΔF-subject_-$F\Delta$: E_1; $\Delta\forall$-action_-IO: at_time;

ΔF-object_-$F\Delta$: times_1}

where times_1 are also defined to be the set of times that occurred on the 15th of June. Negating E_0 results in that E_0's object arc becomes observational with respect to an event E_1', defined only by an event identical to E_2 and a subject_ arc with target Roberto.

∘ **Conclusion: effect of propositionality**

Propositionality is not a source of great difficulties in conversion to antonym.

D.2.2.6 Implications of negating a definitional event

Very few events are purely observational with respect to all of their arguments. This means that most conversions to antonym will negate an event which defines some other concept. This raises the question of what should be done with this concept: its existence depends on an event which is no longer believed true. A truth maintenance system must solve this problem by deciding whether to believe the event or its conversion to antonym. This is necessary in order to ensure consistency of the knowledge base. It raises the issue of belief, since at first, two agents may believe something, and later one of them might change his mind and believe in the conversion to antonym of the event. Both the event and its conversion to antonym must then be maintained within the network.

Instead of discussing the truth maintenance system, which is clearly an issue of reasoning, the rest of this section will concentrate on the effect of conversion to antonym on observational events or arcs. For simplicity, the discussion will be phrased as if conversion to antonym were a destructive operation on the event being converted, and associated concepts. The general case cannot be achieved by simply duplicating the concepts manipulated in this way, and then performing the conversion to antonym on them: each original concept is associated by a set relation to the corresponding changed concept.

When an event E_0 which defines a concept C is converted to its antonym, $\neg E_0$, the resulting event refers observationally to the concept's closest parent, P. A concept which is defined by many events is equivalent to the definitional intersection of as many sets each defined by one of the definitional events. This means that if the concept C is defined by events additional to E_0, the closest parent P may have to be built as the set defined by all the definitional events of C other than E_0, and inserted appropriately into the set hierarchy. $\neg E_0$ will then refer to P observationally. Because P is not C, C's observational events will not copied to P: C's observational events express facts about the concept C, that are not necessarily true of P[10].

There is however another aspect. Some of the definitional events originally connected to the concept C, and now connected to its closest parent P, may have been mutually defining events. These would also have defined other concepts $\mathcal{C}_1$, which in turn may be mutually defining with other concepts $\mathcal{C}_2$, and so on. All the concepts which are directly or indirectly mutually defining with C form the set $\mathcal{C}$. Since all these concepts were mutually defining with C, the definition of C constituted part of their own definition. Thus, when they were disconnected from C, and connected to P, their definition changed. In other words, the same nodes now correspond to different concepts. This means that any observational events these nodes were connected to, are no longer correct, since the observational events referred to different concepts. As a result, when E_0 is converted to its antonyms, a search must be performed over all the concepts that were mutually defining with any concepts that E_0 happened to define. This search must remove all observational events connected to the mutually defining concepts.

For instance,

(E_0,R): {$\Delta\forall$-`subject_`-IO: `John`; $\Delta\forall$-`action_`-IO: `like`;
ΔF-`object_`-$F\Delta$: `fish`$_1$}

(E_1,R): {ΔF-`subject_`-$F\Delta$: `fish`$_1$; $\Delta\forall$-`action_`-IO: `has_part`;
ΔF-`object_`-$F\Delta$: `red_tails`}

(E_2,R): {ΔF-`subject_`-FO: `fish`$_1$; $\Delta\forall$-`action_`-IO: `eat`;
ΔF-`object_`-$F\Delta$: `food`$_1$}

[10]Since C is no longer defined uniquely, it should be deleted from the network, as it is illegal.

(E_3,R): {ΔF-subject_-$F\Delta$: red_tails; $\Delta\forall$-action_-IO: is_red}

where $\texttt{fish}_1$, red_tails, $\texttt{food}_1$ are specializations of fish, tails, and food respectively becomes

(E_0,R): {$\Delta\forall$-subject_-IO: John; $\Delta\forall$-action_-IO: dislike;
ΔF-object_-FO: $\texttt{fish}_1$}

(E_1,R): {ΔF-subject_-$F\Delta$: $\texttt{fish}_1$; $\Delta\forall$-action_-IO: has_part;
ΔF-object_-$F\Delta$: red_tails}

(E_3,R): {ΔF-subject_-$F\Delta$: red_tails; $\Delta\forall$-action_-IO: is_red}

where $\texttt{fish}_1$ and red_tails are specializations of fish and tails respectively.

Similarly, since the conversion of antonym changes the meaning of E_0 to $\neg E_0$, any observational events or arcs connected to E_0 must be deleted. This has a consequence that conversion to antonym is limited to events that are defined by an action_ arc. Since this is usually the case, it poses no problem.

D.2.2.7 Conclusion: conversion to antonym

Conversion to antonym is a complex process, involving many separate problems. However it occurs far more rarely that the other form of negation, so cannot be considered severely detrimental to efficiency. Moreover, its scope is mainly limited to the event being negated, although the consequences of that negation affects any concept the event defines directly, or through mutual definitional dependencies.

D.2.2.8 Negating events by absence of occurrence

- **The treatment of defined individuals**

The same difficulties occur with defined individuals for absence of occurrence as for conversion to antonym. This means that the same sort of preprocessing that is applied to this problem for conversion to antonym is used for negation by absence of occurrence: D.2.2.3 *(p. D-33)*.

- **The actual negation**

Negating individual events is performed by raising the level of quantification by one: the individual is transformed into a set of one element[11]. This is then negated by setting its observational `size_` to zero. The actual process of doing this involves changing the quantification of all arcs whose source is the event by adding a final (independent) $\forall$. Thus, where the event was once an individual, it is now a set ($I\forall \equiv \forall$): another level of grouping has been added at the bottom of the grouping structure. The events that refer to the negated event must also be adapted so that they now refer to its instantiations. Because of event arity (D.2.2.2 *(p. D-33)*), this process is non-local.

Negating events with many instantiations is very similar. Since events may be in one-to-one, many-to-one or one-to-many relations with their arguments, negation cannot be performed simply by adding an observational `size_` zero to the event. For instance, negating in this way the observational event that farmers beat the donkey (that they own) to produce farmers do not beat the donkey that they own would result in stating that there are no farmers who own a donkey because of the $F - F$ quantification. The solution is again to raise the level of quantification by one as previously by adding a new grouping at the bottom level. Again an observational `size_` zero event, with quantification referring to the bottom level but one, is used to state that there are no events which follow the quantification scheme. Now the size zero event has no implication on the number of farmers or donkeys, since the F quantification is not dependent on the lower level $\forall$.

There are two ways of raising the quantification: one can add a level of quantification either above all the existing levels or below. To see the difference, take as example the negation by absence of *"Each of the salesmen sold all his cars"*: *"Each of the salesmen sold none of his cars"*. This could be represented by E_0 (quantification above) or E_1 (below), without the relevant `size_` events:

(E_0, R): $\{\Delta\underline{\forall}F\forall$-`subject_`-$FO$: $\mathcal{S}$; $\Delta\underline{\forall}\forall\forall$-`action_`-$IO$: `sell`; $\Delta\underline{\forall}FF$-`object_`-$FF\Delta$: $\mathcal{C}\}$

(E_1, R): $\{\Delta F\forall\underline{\forall}$-`subject_`-$FO$: $\mathcal{S}$; $\Delta\forall\forall\underline{\forall}$-`action_`-$IO$: `sell`; $\Delta FF\underline{\forall}$-`object_`-$FF\Delta$: $\mathcal{C}\}$

(E_2, R): $\{\Delta\underline{\forall}\forall$-`subject_`-$IO$: `John`; $\Delta\underline{\forall}\forall$-`action_`-$IO$: `sell`; $\Delta\underline{\forall}F$-`object_`-$F\Delta$: `cars`$\}$

(E_3, R): $\{\Delta\forall\underline{\forall}$-`subject_`-$IO$: `John`; $\Delta\forall\underline{\forall}$-`action_`-$IO$: `sell`; $\Delta F\underline{\forall}$-`object_`-$F\Delta$: `cars`$\}$

[11] see 5.4.2.3 *(p. 140)*

Assume that John is one of the salesmen. Inheritance must now map E_0 or E_1 to John. For E_1 this is easy, since (E_3) *"John sold none of his cars"* is an instance of E_1: inheritance or semantic integration can build the required set hierarchy easily. However this is not the case for E_0 and the corresponding E_2, since although E_2 is inherited from E_0, it is not an instance of E_0. Thus the preferred solution is to add a new grouping at the bottom level.

- **Events that define their arguments**

As for conversion to antonym, negating events which define other concepts changes their definition. As for conversion to antonym, this aspect must be treated by a truth maintenance system.

D.3 Relating concepts to their possible parts

D.3.1 Parts of concepts

D.3.1.1 Motivation

Dividing an object into its constituent parts is a very frequent process. Indeed, it is related to the partitioning process described in 2.4 *(p. 13)*, and serves the same purpose. Once the parts have been identified, the object as a whole can also be described by the relations between them. The purpose is to determine easily identifiable, yet also significative parts in the sense that they participate in many internal relations with other parts of the object. The value of partitioning as before is judged on the reasoning power it provides. For instance, one could divide a clock into small cubes, but the relations between the cubes would not provide any power for reasoning. If instead, the clock were divided into cogs, springs, and other pieces that resulted from it being taken apart gently, then the relations between the pieces would be useful.

Although definitional events and the conceptual hierarchy implicitly partition the "world" into pieces, a means to state that one such piece has other pieces as parts must be provided. Concrete objects obviously have parts, but so do other concepts. For instance, typing a text can be divided into small events each involving pressing a key, which can in turn be subdivided into events involving pressing the key, keeping it there, and releasing it. Many events involving interactions between physical objects can be modeled in this way.

D.3.1.2 Properties of division

Objects may be divided into parts in many different ways. For instance the geographic region of Europe can be subdivided either into countries, or into geographic features (plains, rivers, mountains, islands). A piece resulting from one way of dividing the object may have areas in common with pieces resulting from another division. For instance, the Alps straggle the borders of Switzerland, France and Italy. These pieces may be referred to, for instance as the Swiss, French and Italian Alps. This means that the representation must provide some way of grouping the results of each division if desired. Similarly, objects may be divided into other objects which in turn may be further subdivided into other objects. Thus hierarchies of parts can be formed. The representation must allow such hierarchies to be expressed.

An object may share some of its properties with its parts. For instance, if a table is made of wood, the table, its legs and its top all share the property of spontaneously burning at the wood's flash point temperature. However some of its other properties are not inherited, such as its weight. If the table were broken, each piece would not weigh the same as it did. This difference between those properties that are known to be shared by an object and its parts, and those that are known not to, is the difference between intrinsic and extrinsic properties. The substance from which an object is made provides many such intrinsic properties, whereas extrinsic properties commonly correspond to properties of the object that the substance was made into. By annotating the properties appropriately, a scheme similar to inheritance can be introduced so that properties intrinsic to a substance can be inherited to the object itself.

D.3.1.3 The need for a new relation

This notion of parts is different to the `inst_` and `spec_` events, although they both provide means of partitioning LOLITA's world. Because the relation between an object and its parts distinguishes between intrinsic and extrinsic properties, it can relate an object to a seemingly unrelated object used in its construction. As a result, the same object can be divided in different ways into different parts. The resulting parts have little to do with their source. For instance, if one were given a bolt, it would be hard to identify whether it were part of a building, a tractor, a lift or the Mir space station. Indeed, bolts' properties are defined independently of the use to which they are put.

`inst_` and `spec_` on the other hand inherit all the properties true of the elements of the superset, so they are constrained to relating concepts and their specializations. However, `inst_` can also be used to relate individual concepts, such as a group of men, to its elements, such as John and Mark. No properties true of the group are inherited to its elements, so again the meaning differs.

D.3.1.4 Substances

Some concepts corresponding to physical objects have the property that they are not thought of as identifiable individual objects. These are substances, which have the particularity that when you divide them into parts, the parts too are instances of the same concept. For instance a piece of meat, divided into pieces forms more pieces of meat. In natural language such concepts usually are expressed as count nouns: people talk of pieces of metal, or some metal, but not of metals in the sense of individual objects.

D.3.1.5 Granularity of division

Although intrinsic properties are inherited by the parts of a given object, there is a limit to how far the process of division can be taken. If one divides pieces of meat up sufficiently, the result is no longer meat, but various proteins. Similarly, if one divides an alloy, the result is no longer an alloy, but the metallic atoms involved in its composition. For substances this can be associated with the notion that a substance not only corresponds to a thing, but to a particular organization of other more basic things, which have relations between them, and it is these relations or internal organization which result in the properties of the substance perceived at a different scale.

Subdividing events, such as walking also involves a notion of granularity. Once the division has reached a certain stage, one cannot recognise the sub-event concerned as walking. The sub-event could be part of walking, or of falling, for example.

Thus the properties defining the concepts which behave as substances must be associated with a range of application.

This notion that there is a limit after which dividing a substance results in individuals can be turned the other way: if one is dealing with a large enough number of individuals, they start behaving like a substance with its own properties. For instance, a landslide is composed of stones, gravel and so on, but their movement as a group has many properties in common with a liquid. Similarly crowds may display similar types of behaviour whatever the individuals forming them. Here however, one must decide whether the notion of partition involves the inheritance of intrinsic properties or not. If so, one would use the notion of division, otherwise one would use `inst_` events, as described above.

D.3.2 The has_part event

In order to relate a concept to its parts, an event is needed: the `has_part` event, which takes as `action_ has_part`.

A requirement for the `has_part` event is that it be able to group the parts resulting from applying a particular division scheme to some object. Thus, for instance, division of the human body into trunk, arms, legs and head should distinguished from its division into organs. If each part were simply to be the `object_` of a separate `has_part` event, it would be impossible to distinguish the results of the two division schemes, unless their relations were enumerated exhaustively: the brain is part of the head, part of the spinal cord is in the head, another part is in the trunk, and so on... In general exhaustive enumeration is unwise since the relations between elements resulting from different division schemes may not be known. LOLITA might not know that there are geographic regions inhabited by German-speakers in the state of Czechoslovakia. Nor indeed, should she need to.

A solution which satisfies this requirement is a `has_part` event with as `subject_`, the divided object; as `action_`, `has_part`; and as partial `object_s`, the results of a particular division. This also allows the `has_part` event to relate a single part of an object to the object itself with no reference to the division scheme itself: if a `has_part` event only has one `object_`, then it could be thought to state that the object has only one part. But this part would be the object itself. Thus, if a `has_part` event only has one `object_`, it will be taken to mean that its `object_` is but one of its parts. This scheme is essential, for without it LOLITA would need to know the full result of the partitioning scheme before an event could be expressed.

D.3.3 Using has_part

D.3.3.1 Representing substances and intrinsic properties

Substances are represented as the set of objects with the properties of that particular substance. Their behaviour with respect to division is expressed using the `has_part` event:

(E_0, R): $\{\Delta F\forall$-`subject_`-FO: `metals`; $\Delta\forall\forall$-`action_`-IO: `has_part`;
$\Delta\boxed{\forall\forall}$-`object_`-$\boxed{\exists!}\Delta$: `metals'`$\}$

(E_1, R): $\{\Delta I$-`subject_`-IO: `metals`; ΔI-`action_`-IO: `synonym_`;
ΔI-`subject_`-IO: `metals'`$\}$

(E_2, R): $\{\Delta F$-`subject_`-$F\Delta$: `metals`; $\Delta\forall$-`action_`-IO: `is_metal`$\}$

where `is_metal` corresponds to the definitional properties of metals.

Thus a gold ring is a piece of gold with additional properties, restricting it further to being a ring[12].

If `metals` is observed to have certain intrinsic properties, such as only applying to instantiations of a minimal weight, these intrinsic properties will be definitional for `metals'`. It is this part of the statement which not only states the substance's behaviour with respect to division, but also which ensures that inheritance of intrinsic properties works as desired. Take as example of a substance, metal. If only a part of an object is made of metal, then the object will not be an instance of metal, preventing it from acquiring intrinsic properties it does not have. However, if an object is part of a piece of metal, it will fit one of the requirements of E_0, a definitional event of `metal'`. Assuming one knows that it fits the other requirements such as minimal weight, it is an instance of `metal'`. Thus, by E_1, it is a piece of metal.

Intrinsic properties are represented in the same way. Their template defines as `subject_` a substance which has as one of its observational properties that its parts are also of the same substance. Other observational properties may state the property's range of application. For instance, for notions such as colour or melting point to apply, a minimal amount of substance is required. Substances can then be defined as combinations of the appropriate intrinsic properties, using bi-implicature[13].

Other properties of objects or substances may be known to vary in predictable ways when an object is subjected to a **has_part** event. For instance, the sum of the weights of an object's parts is equal to its weight. Similarly, every part of an object is smaller than it. For events, an event's parts all take less time than it did. Such common-sense knowledge can be encoded as above, and proves essential in many reasoning tasks such as, say, working out whether someone's alibi stands up.

[12]See 7.6.2.1 *(p. 291)* for information about **synonym_s**
[13]see 5.4.3.3 *(p. 153)*

D.3.3.2 has_part for events

Just as a substance can be divided into other instances of the same substance, so certain events can be divided up. For instance "existing": if John existed the whole of yesterday, he existed at any time yesterday. However, just as with substances, for which there is a conceptual difference between *"the substance gold"*, and *"a piece of gold"*, there is a conceptual difference between the full event and its parts: *"I ran to school today"* includes all the running performed from leaving home to reaching school. Thus, there are two types of running action: `run` and `run_substance`. The former has the latter as parts (except perhaps at the beginning and end).

This difference also appears in N.L., for instance between *"I ran a mile"* which cannot be quantified by *"for twenty minutes"* (*"I ran a mile for twenty minutes"*) and *"I ran towards the church"* which can (*"I ran towards the church for twenty minutes"*).

- **Granularity**

Some events can only be divided until they reach a certain granularity. For instance, the "walking" event can be divided to individual steps, but further subdivisions would not be recognised as walking. The granularity is thus determined by the minimal extent an event must have to be recognised as an instance of that event: taking a few steps for instance. This may often be associated with a time, but it should be noticed that unlike substances where a precise size might be found, events' granularity is not so much determined by their span as by what actually happens. For instance, it might take longer to determine that a sloth is walking than a person, because sloths move slowly. Similarly, it might take a longer time to determine some viscous material is flowing, than it does water. Glass is a prime example of this. As a supercooled liquid it flows, but incredibly slowly.

If it is necessary to specify a minimal time, it must be associated with a sort of template event which is the superset of all events with a given `subject_` or `object_` type: e.g. all events with a person as `subject_`. Like a template event, it describes attributes of a vast class of events, such as all events involving people as `subject_s` and walking as `action_`. Unlike a template event, it is not only defined by its `action_` but also by its `subject_` and `object_`.

Other events can be subdivided infinitely. For instance, "existing" is such an event. If an object existed from 1992 to 1994, it existed at every instant during that period, where an instant is an infinitely small time. Existing is expressed as non-zero set size, and is an instance of a `has_value` event. Many other `has_value` events share this property of infinite division.

- **Aside: event density**

Associated with the notion of granularity is the notion of event density. This corresponds to non-literal uses of events, where an event is said to occur during a particular time although it did not occur during every instant of that time. For instance, *"He walked from Inverness to Penzance"* does not mean that he spent every minute during that event walking. He might have done, but that was not necessary for the statement to be made. This is one of the cases where homogeneity theory ([Garigliano 89]) is expected to provide a solution in the longer term. Currently, the walk from Inverness to Penzance would be considered non-literal.

- **Pre- and post- conditions**

The parts of events are also events, and thus also have their own pre- and post-conditions. If the division of the event into parts is complete, it is expected that all the preconditions of the sub-events, less the postconditions of the sub-events that preceded them, will be the preconditions of the whole event. The same is expected for the postconditions.

For instance, *"to eat"* can be viewed as consisting of cutting food, putting it in the mouth, chewing it, swallowing it, and so on. Cutting food has as precondition that the food can be cut, and as postcondition that the food is smaller than it was before. Putting food in the mouth has as condition that the piece is small enough, and as postcondition that the food is in the mouth. Chewing food has a precondition that there is food in the mouth and is not so hard it cannot be chewed, and as postcondition that the food is small and wet enough to be swallowed. Swallowing has as precondition the food is small and wet enough to be swallowed, and as postcondition that the food is in the stomach. From this, cutting is seen to be optional: one does not need to cut pieces of rice. Similarly, chewing is optional: one does not need to chew yogurt. Since cutting food satisfies the preconditions of putting it into one's mouth, the precondition that the food is small enough to fit in the mouth does propagate to being a precondition of the *"to eat"* template event. However, the postcondition that the food ends up in the stomach is not the precondition of any other subpart of *"to eat"* so does get propagated to *"to eat"*'s template event.

- **Part's times**

The time of the parts of events can be specified with respect to the full event, allowing the initial, intermediate and final stages of an event to be specified. This is particularly important to specify the difference between full events and substance-like parts of events:

(E_0, R): {O∀-subject_-∃!Δ: Walking_Animals; Δ∀-action_-IO: walk}

(E_1, R): {ΔF-subject_-FO: E_0; Δ∀-action_-IO: at_time;
Δ∀-object_-∃!Δ: $\texttt{times}_1$}

(E_2, R): {ΔF-subject_-FO: E_1; Δ∀-action_-IO: has_part;
ΔF-object_-FΔ: E_3}

(E_3, R): {Δ∀-subject_-∃!O: Walking_Animals; Δ∀-action_-IO: leave}

(E_4, R): {ΔF-subject_-FO: E_3; Δ∀-action_-IO: at_time;
Δ∀-object_-∃!Δ: $\texttt{times}_2$}

(E_5, R): {ΔF-subject_-FO: [$\texttt{times}_1$,$\texttt{times}_2$]; Δ∀-action_-IO: starts_;

Every walk event starts by a leaving event.

D.4 The representation of Time

D.4.1 Time and language: A new paradigm

Just as space, time appears to the agent to be a continuous phenomenon. Any conceptualization of specific times is the result of partitioning. Thus it can be represented by many models.

D.4.1.1 Scientific model

A starting point for the representation of time could be the scientific model of time. Classically, time is regarded as one of the axes of a coordinate system: a continuum of points isomorphic to the real line. Every instant is represented uniquely by one of these points, which can be expressed in terms of any other, by a unique number of units. The units themselves are precise, and any uncertainty is expressed as a range: the error. Events are assumed instantaneous, but if they take any length of time, they are associated with an interval on the real line, which has a start and end point.

Time is thus an independent variable, indeed the independent variable par excellence, on which the functions in scientific models often depend. This is because science models processes, which by their very nature always involve time.

Although this model suits scientific tasks, it does not model naturally many NL statements.

D.4.1.2 Forcing the Distinction between Intervals and Instants

The scientific model of time represents instants by a single value, and intervals by two values defining the interval's endpoints. There is no way of representing some constraint on the time of an event which could be an instant or could be an interval. Thus, one is forced to decide whether the event was instantaneous or not, in order to represent statements about its time.

For instance, to represent the sentence *"I woke up this morning"* using the scientific model, one would end up stating that the event of my waking up took place between $t_0 \pm \delta t_0$ and $t_1 \pm \delta t_1$, and t_0 and t_1 would both be further defined to have taken place this morning. This is because the scientific model forces one to specify whether the event is an interval or an instant. In the case of waking up, one can specify this because one knows what waking up is. But one might not know whether the action designated by a verb is instantaneous or not. Suppose that one does not know the meaning of *"to laud"*, one is still knows that *"I lauded him this morning"* means that the lauding occurred this morning, without knowing whether it took an instant or an interval.

In other words, the scientific representation forces one to say things one does not know. There is no reason why one should have to state whether *"I woke up this morning"* is instantaneous or not, when all one wants to state is that the the event occurred this morning. Being forced to make distinctions unnecessarily is the sign of a bad partition of the problem, that the representation is not adapted to the problem for which it is being used: natural language.

To conclude, the representation of time for SemNet must allow statements to be made about any event without forcing one to decide whether or not the event was instantaneous.

D.4.1.3 Precise times

The scientific model of time forces the times of events to be expressed precisely, when precise values are not known, not meant or simply irrelevant.

For instance, *"I'll see you in five minutes"* does not mean that precisely five minutes – 300 seconds – after the utterance, the people will meet again. Nor does it mean that precisely between four minutes – 240 seconds – and six minutes – 360 seconds – after the utterance, the people will meet again, if an error of one minute is used. It means something far vaguer: something around five minutes. As was discussed in 7.1 *(p. 247)*, certain expressions like *"five minutes"*, *"half an hour"* indicate a far vaguer level of precision than *"four minutes"* or *"thirty minutes"*. This behaviour means that the representation of time must allow vague times to be expressed.

A consequence of this is that conversions between units of time is delicate. *"five minutes"* is often not equivalent to *"300 seconds"*. Similarly, conversions between dates in different calendars are not trivial, and should not be forced by the representation. Indeed, the equivalence between different calendars may not be known, yet the representation should be able to express such dates as *"On the fifth day of the third tenial, ..."* [14]. Indeed, imposing the requirement of conversion to some standard calendar is unnecessary for tasks that require no reasoning. It may even be counter-productive for tasks such as translation, where keeping the original form may be desirable, and the requirement that each calendar be convertible reduces the robustness of the system when it encounters a new text. In other words, the representation of time should support the expression of date and time in a wide variety of calendar systems or time units.

For instance, the Roman day was divided into ten hours of sunlight as measured by a sundial. Converting this time into modern western units would require knowing the location (latitude) of the event. Moreover if all times had to be converted into the time system used today, it would be virtually impossible to express partially defined times such as *"the fifth hour of some day during the forth lunar month 434"*. Indeed, even if all the precise day and latitude were known, the *"fifth hour"* might turn out to be *"11:45:03"* which seems to indicate a precision the original text did not convey.

Simply because the representation should not force a level of precision greater than that meant does not mean that the representation should prevent precise values being represented: the representation must be able to express whatever level of precision is meant flexibly.

[14]During the French Revolution, weeks were converted to metric form: each week consisted of ten days.

D.4.1.4 Time as an axis

The scientific model proves clumsy when representing such day-to-day sentences as *"After meeting the chairman yesterday, Jack was seen dancing in the car park for more than five minutes."*: Jack met the chairman between v_0 and v_1, and $yesterday\_start <= v_0$ and $v_0 < v_1$ and $v_1 <= yesterday\_end$. Someone saw Jack dancing between v_2 and v_3, and $v_3 - v_2 = v_4$ and $v_1 <= v_2$ and $v_2 < v_3$ and $v_4 > 5$ minutes. Five variables and seven ordering relations are needed to represent two times and three relations: the time when Jack met the chairman was yesterday (one time and one relation); the time when Jack was seen dancing was after his meeting (one time, one relation) and lasted more than five minutes (another relation).

These difficulties stem from the use of constrained variables to express uncertainty of knowledge. The representation in the scientific model maintains a sharp distinctions between those times which can be assigned a date, and those which cannot. The former are represented by some number, whereas the latter are represented by a possibly constrained variable. The need for variables stems from the view of instants being points on the time axis. This choice of point means that one immediately knows the instant's relation to all other instants: for each one, one knows whether it is earlier or later than the instant. In effect one is dealing with a total order. The variables provide a means of escaping this, and expressing a partial order with constraints on the variables.

However, in NL it is very rare to know the exact relation between the times of all the events in the text being analysed. Thus in the general case, it is impossible to achieve the total order required to map all these times to a single axis. Instead, what emerges is a graph of mutual constraints on times. The graph includes the temporal relations the speaker considered important, or non obvious, but will neglect many others. Thus the position of many times with respect to others will not be known. Representing this by constrained variables proves very complex.

Instead times could be considered as any other concept. What information is known about them, qualifies them. Their representation does not include any references to unknown information, while maintaining the ability to represent the full range of temporal information. Times considered as concepts are thus constants, rather than variables, about which more or less may be known. There is no difference in representation between a time for which a date is known and a time which is defined only as being the time when an event occurred.

Furthermore, the representation of times as a graph reflects their usage. In the scientific representation of unknown times as variables, there are implicit relations between variables which make up the partial order. Similarly, the total order is implicitly in the nature of the constants assigned to the times. However, in the representation of times as concepts, the relations between them from a graph, since the concepts are nodes in the graph SemNet. The implicit partial order is thus explicit in the graph.

D.4.1.5 Perceptual and Real Instants

Events which take a certain amount of time may be presented as instants in NL sentences. For instance, *"Sengan woke up, ate breakfast and went to work"* refers to events as if they were instants. On the other hand *"While I was waking up, someone threw cold water at me"*, refers to the waking-up as an interval, although both may refer to the same event. NL often expresses intervals as instants in this way, only referring to the internal nature of events when needed. Henceforth, intervals referred to as if they were instants will be called "perceptual instants". "Real instants" will refer to times that are conceptually instants, independently from a text's viewpoint. For instance, just such an instant is the time when a ball thrown into the air is stationary.

Since intervals referred to as perceptual instants are expressed in the same way as real instants, they should not be represented differently. Indeed, different representations for perceptual instants (intervals) and real instants, increase the complexity of the NL analysis and generations processes. For instance, there is no way of knowing whether a previously unknown event type is an interval or an instant if it is expressed as an instant in a sentence. *"I lauded him"* may be instantaneous, but need not be. If real instants are represented differently from intervals, there must be three different representations of time: one for intervals and perceptual instants, one for real instants, and one for times which may be instants or intervals. Later, if the same event type is referred to in a way that made it clear that it takes an interval, all its instances will need the times at which they occurred to be converted to interval form.

Instead of three different representations for real instants, intervals, and unknowns, the difference between intervals and real instants can be expressed as additional information. If one knows that a time is an interval or an instant, one knows more than if all one knows is that it is a time. Thus all times can be referred to as instants in NL, depending on the viewpoint adopted by the text: *"The British Empire collapsed in the 20th century."*

In other words, the representation of an interval, a perceptual instant and a real instant should only be distinguished by additional information.

D.4.1.6 Event density

In the scientific model of time, properties have well-defined values throughout the time under investigation. Thus, properties that hold during an interval are implicitly understood to hold at each instant during that interval. However, in NL this is not necessarily the case. For instance, *"I walked from Durham to Newcastle in eight hours"* does not necessarily imply that I was walking at every instant along the way: I may have stopped in between. Indeed, as discussed in 7.4 *(p. 269)*, events may not even make sense for intervals under a certain size. Thus, the statement that an event held during some time must not be taken as meaning that it held at every instant during that time. However the representation must also be able to express that the event held non-stop if so desired.

D.4.1.7 The need for instants and for intervals

Representing instants and intervals may appear to be unnecessary, and to break uniqueness.

Indeed, it might be argued that instants never occur, so it is sufficient only to represent intervals. However, people can conceptualize the moment at the top of its trajectory when a ball is stationary after being thrown in the air. The fact that this may only be an abstract concept is irrelevant. It is clearly a concept people can discuss. Similarly, concepts such as instantaneous velocity refer to the notion of real instant. Thus, even if it were only a conceptual device that allows people to think about certain phenomena, SemNet must be able to represent it.

Alternatively, since instants are needed, it might be argued that intervals are unnecessary. They could simply be represented as all the instants lying between their endpoints. However, this would imply that all events made sense at the temporal granularity of an instant. Thus when *"Bob Dole campaigned to become the next president of the USA"*, he must have campaigned at every instant of that range of time. Moreover, it must make sense to state that *"Kevin ran a mile in three minutes for a nanosecond"*, since it makes sense to state that *"Kevin ran a mile"* during every instant of the three minutes it took him to run it. Again this is dubious, since running a mile definitely does not take an instant. As was explained in 7.4 *(p. 269)*, events have a certain temporal granularity under which they no longer make sense.

Since no coherent view can be obtained either only using instants or only using intervals, SemNet needs both.

D.4.1.8 The ends of intervals

Another issue of concern in the scientific model, when dealing with intervals is whether the event holds at the endpoints of the interval. The reason for this concern is that if an event holds during the interval $[t_0, t_1]$ and does not during the interval $[t_1, t_2]$, it could hold and not hold simultaneously at t_1 if the intervals are closed – if they include both of their endpoints.

However it is very rare in NL that details such as whether an event holds during an interval's endpoints are stated. For instance, *"The train trip lasted an hour"* does not state whether it includes the endpoints, since this is irrelevant. Indeed, the notion of some events holding for an instant may be disputed. It makes therefore sense to make the conservative assumption that events that occur during an interval do not occur at it's endpoints. If they do, this must be specified additionally: *"The examination started at 3 o'clock precisely ended at 4 o'clock precisely"* could be interpreted as saying that the examination held from 3 o'clock included to 4 o'clock excluded. The difficulty in creating credible examples reflects the problem's artificiality.

D.4.1.9 Representing the time of an event

- **Quantificational dependencies**

Events can depend quantificationally on the times at which they occur. For instance, in the sentence *"Every day the sun rises"*, there is an event of the sun rising associated with a time each day. Similarly, the times at which events occur can depend quantificationally on the events themselves. For instance, *"The executions always take place at midday on Fridays"* states that the time of every execution is Friday at midday.

- **Sortal dependencies**

Events can define the times at which they occur: *"'When the economy has met the Maastricht conditions, we shall reinstate the welfare system" said Helmut Kohl'.* The time when the welfare system is to be reinstated is defined by the condition *"when the economy has met the Maastricht conditions".*

But times can also restrict a concept to the events which occur during them: *"The economy will improve next year"* or *"John and Mary will marry tomorrow"*. This restriction also allows absence of occurrence to be expressed: *"I did not see Mary yesterday".*

- **Epistemological issues**

Statements can be made about the time at which events occurred. For instance, the time an event occurred is subject to belief: one may believe that *"John bought a car"*, but not believe that he bought it in 1900. Similarly, different people may have told one the event and when it occurred: *"Jack said 'John bought a car'; Bill said 'Ah yes, that was in 1979'."* The certainty of the event's time may be in expressed: *"I'm pretty sure that my mother's birthday is on the 9th of June"*. Or, the time at which an event did not occur can be stated: *'Hercules Poirot said "I know that John could not have committed the murder at midday, because I dined with him."'*

D.4.2 Example Templates

The template event for the meaning of after_ (*"The start of the second event is after the end of the first event."*) is E_0 in

(E_0,R): {O∀-subject_-∃!Δ: $\mathcal{A}$; Δ∀-action_-IO: after_4 ;
O∀-object_-∃!Δ: $\mathcal{B}$ }

(E_1,R): {ΔF-subject_-FO: E_0; Δ∀-action_-IO: duration_;
Δ∀-object_-∃!Δ: $\mathcal{C}$}

(E_2,R): {ΔF-subject_-FO: $\mathcal{A}$; Δ∀-subject_-∃!Δ: $\mathcal{D}$;
Δ∀-action_-IO: ends_}

(E_3,R): {ΔF-subject_-FΔ: $\mathcal{D}$; ΔF-subject_-FΔ: $\mathcal{F}$;
Δ∀-action_-IO: starts_}

(E_4,R): {ΔI-subject_-IO: instants; ΔI-action_-IO: spec_;
ΔI-object_-IΔ: $\mathcal{D}$}

(E_5,R): {ΔF-subject_-FΔ: $\mathcal{F}$; Δ∀-action_-IO: has_duration;
ΔF-object_-FO: $\mathcal{C}$}

(E_6,R): {ΔF-subject_-FO: $\mathcal{B}$; Δ∀-subject_-∃!Δ: $\mathcal{E}$;
Δ∀-action_-IO: starts_}

(E_7,R): {ΔI-subject_-IO: instants; ΔI-action_-IO: spec_;
ΔI-object_-IΔ: $\mathcal{E}$}

(E_8,R): {ΔF-subject_-$F\Delta$: $\mathcal{F}$; ΔF-subject_-$F\Delta$: $\mathcal{E}$;
$\Delta\forall$-action_-IO: ends_}

This particular meaning states that the startpoint of the later interval is $\mathcal{C}$ later than the endpoint of the earlier interval.

The template event for follows_ (*"The start of the first event precedes the start of the second event."*) is E_0:

(E_0,R): {O$\forall$-subject_-$\exists!\Delta$: $\mathcal{B}$; $\Delta\forall$-action_-IO: follows_;
O$\forall$-object_-$\exists!\Delta$: $\mathcal{A}$ }

(E_1,R): {ΔF-subject_-FO: E_0; $\Delta\forall$-action_-IO: duration_;
$\Delta\forall$-object_-$\exists!\Delta$: $\mathcal{C}$}

(E_2,R): {ΔF-subject_-FO: $\mathcal{A}$; $\Delta\forall$-subject_-$\exists!\Delta$: $\mathcal{D}$;
$\Delta\forall$-action_-IO: starts_}

(E_3,R): {ΔF-subject_-$F\Delta$: $\mathcal{D}$; $\Delta\forall$-action_-IO: has_duration;
ΔF-object_-FO: $\mathcal{C}$}

(E_4,R): {ΔF-subject_-FO: $\mathcal{B}$; $\Delta\forall$-subject_-$\exists!\Delta$: $\mathcal{E}$;
$\Delta\forall$-action_-IO: starts_}

(E_6,R): {ΔI-subject_-IO: instants; ΔI-action_-IO: spec_;
ΔI-object_-$I\Delta$: $\mathcal{E}$}

(E_7,R): {ΔF-subject_-$F\Delta$: $\mathcal{D}$; ΔF-subject_-$F\Delta$: $\mathcal{E}$;
$\Delta\forall$-action_-IO: ends_}

D.4.3 Time and the Inheritance Hierarchy

Introducing time has consequences for the inheritance hierarchy. This section discusses these, starting with the notion that entities may have a limited life-span. This leads to qualifying the set relations by at_time events, which requires an extension to the inheritance algorithms. The implications for antonym_ events is then reviewed.

D.4.3.1 Time and size_

Entities exist for a limited time. Men, for instance, are born, live, then die. Behind this triviality lies an important problem: the combination of time and the inheritance hierarchy. The first occurrence of this problem is the variation of size of a set. For instance, in 1992 there were 10 players in the departmental AI football team, whereas in 1993 there were 12 and in 1990 there were none. The concept of AI departmental football team does not vary, but it can have different sizes at different times. This is expressed by attaching many `size_` events to the team, but qualifying each by a different time.

In the same way individual entities have a life-span. Until the introduction of time, they were represented by constants in SemNet. Recall that constants in SemNet which are defined by events are in fact equivalent to sets defined by the same events which are observed to have `size_` 1. Representing directly the fact entities have a limited life-span would be represented by qualifying the `size_` event by an `at_time` event. The `size_` event could therefore no longer be implicit, requiring all constants with a limited life-span to be represented by a set of `size_` 1 for the life-span, and `size_` zero otherwise. This solution is cumbersome. The next section presents an alternative.

D.4.3.2 Time and the inheritance hierarchy

Many sets only vary in `size_` because individual elements belong to different sets at different times. Indeed, people may change nationality, becoming a Frenchman after having been a German. Similarly, someone may be a painter, a fireman and a journalist at different points of their life. This means that the set relations must be qualifiable by `at_time` events.

This has implications for the inheritance algorithms: How is inheritance over set relations qualified by `at_time` events performed? The answer is much in the same way as it is performed for other events qualifying the set relations, such as epistemic events. The inherited events are restricted to the times qualifying the set relations over which they were inherited.

Since events may be qualified by a time and then be inherited through multiple set relations, each qualified by its own time, the resulting inherited event may be qualified by many times. A similar situation arises when a value inherits many `has_value` events. Either these value events are compatible, in that some value(s) can be found that satisfies them all, or there is a contradiction. For instance, if one agent believes a set has both `size_` 0 and 1 (in the same frame of existence), there is a contradiction.

To be defined, all concepts other than `typeless` must have at least one definitional set relation to a parent: 5.4.2.2 *(p. 139)*. Consequently, if a concept is connected to definitional set relations themselves restricted by times, it is only defined for those times. That is to say, it cannot be applied to other times, so cannot be discussed. For instance, the concept `Roberto` cannot be applied to times when Roberto is not defined. The only problem with this scheme is if hypothetical statements about what Roberto would do if he were alive are to be expressed. However such statements would be made in a different frame of existence, so a different concept of Roberto would be involved.

Definitional events can be qualified by a time, just as observational events are. Thus the concept *"The football team that won Euro'96"* can be defined. Similarly, if a concept is connected to its parents by definitional events qualified by different times, the inherited definitional events will have these different times. Although this is allowed, it is extremely rare that a concept does not have some definitional events true throughout its life span since concepts are stable partitions of the agent's environment: 2.4 *(p. 13)*.

D.4.3.3 The singular role of time

Events are usually defined by an `at_time` event. Indeed the same sort of event may occur more than once. For instance, the event *"Sengan eats a meal"* usually occurs three times a day. Individual events are equivalent to sets of events observed to have `size_` 1. Thus any individual event *"Sengan ate a meal"* must be distinguished from all other such events by a definitional at_time event. Otherwise, the individual event would state that there is only one event with the feature of *"Sengan eating a meal"*. In some cases this is warranted, for instance for *"Sengan was born on the 1st of March 1971"*.

This feature distinguishes time from other independent variables, which might be thought similar. For instance, events can occur in different locations, introducing three new independent variables. However most events are observed to have occurred in a place, rather than defining the place in which they occurred. This is because the partitioning process resulting in stable concepts implicitly depends on time: a concept is a configuration of the agent's environment that he can recognise when ever it occurs. Indeed even its purpose to allow the agent to interact more successfully with his environment has a temporal basis.

D.4.3.4 Representing the Gregorian calendar

- **Basic calendar**

The Gregorian calendar represents dates by counting years, months and days. The counting starts at one. Years are counted either backwards or forwards from an origin point attributed to be the day Jesus Christ was born. Thus year 1 BC is followed by year 1 AD, without an intervening year zero. Months are counted from the start of the year, and days from the start of the month.

This would seem easy enough to represent, but years, months and even days do not have fixed durations. The duration of leap years differs from other years. Similarly, months may be 28, 29, 30 or 31 days long. Days may be 23, 24 or 25 hours long, to allow the changes to and from summer time. And, every so often, an additional second is added to an hour, to take account of the slowing rotation of the Earth. In a representation with precise values for durations, this would cause difficulties. In SemNet however, the treatment of values in different units is fuzzier. For instance, it is understood that one hour need not be 3600 seconds. This means that to refer to the year 2001, it suffices to refer to the 1 year long interval 2000 years after the origin of the Gregorian calendar. Months can be referred to in a similar way, but the origin will be the start point of the year in which they occur. Days are then indexed with respect to the start of the month. Thus the following is obtained:

(E_0,R): {ΔI-subject_-IO: Years_AD; ΔI-action_-IO: spec_;
ΔI-object_-$I\Delta$: $\mathcal{Y}_1$ }

(E_1,R): {ΔI-subject_-IO: Years_AD; ΔI-action_-IO: spec_;
ΔI-object_-$I\Delta$: $\mathcal{Y}_2$ }

(E_2,R): {ΔI-subject_-IO: $\mathcal{Y}_1$; ΔI-subject_-IO: $\mathcal{Y}_2$;
ΔI-action_-IO: antonym_; ΔI-object_-IO: Years_AD }

(E_3,R): {ΔI-subject_-IO: $\mathcal{Y}_1$; ΔI-action_-IO: size_;
ΔI-object_-IO: 1 }

(E_4,R): {ΔF-subject_-$F\Delta$: $\mathcal{Y}_1$; $\Delta\forall$-subject_-IO: BJC ;
$\Delta\forall$-action_-IO: starts_}

(E_5,R): {ΔF-subject_-$F\Delta$: Years_AD; $\Delta\forall$-action_-IO: follows_;
$\Delta\forall$-object_-IO: BJC }

(E_6,R): {ΔF-subject_-$F\Delta$: E_5; $\Delta\forall$-action_-IO: duration_;
ΔF-object_-FO: $\mathcal{V}_1$ }

(E_7,R): {ΔF-subject_-$F\Delta$: $\mathcal{Y}_2$; $\Delta\forall$-action_-IO: follows_;
ΔI-object_-IO: BJC }

(E_8,R): {ΔF-subject_-$F\Delta$: E_7; $\Delta\forall$-action_-IO: duration_;
ΔF-object_-FO: $\mathcal{V}_2$ }

(E_9,R): {ΔF-subject_-FO: $\mathcal{V}_1$; $\Delta\forall$-subject_-IO: 1_Year ;
$\Delta\forall$-action_-IO: $\oplus$; ΔF-object_-$F\Delta$: $\mathcal{V}_2$ }

(E_{10},R): {ΔF-subject_-$F\Delta$: Years_AD ; $\Delta\forall$-action_-IO: has_duration;
$\Delta\forall$-object_-IO: 1_Year }

where $\mathcal{V}_1$ is the set of integer year durations, greater or equal to zero, and Years_AD is a specialization of the set of times. Months are then expressed with:

(E_{13},R): {$\Delta F\forall$-subject_-FO: Years_AD ; $\Delta\forall\forall$-action_-IO: follows_;
ΔFF-object_-$F\Delta$: Months_AD }

(E_{14},R): {ΔFF-subject_-$FF\Delta$: E_{13} ; $\Delta\forall\forall$-action_-IO: duration_;
$\Delta F\forall$-object_-FO: $\mathcal{V}_3$ }

where $\mathcal{V}_3$ is the set of integer month durations, greater or equal to zero but smaller than twelve. Months_AD is a specialization of the set of times. A similar treatment deals with days, hours, minutes, and so on.

There is however still a problem: the notions of duration are fuzzy in SemNet. This proved useful because it allowed years to be referred to as year long intervals occurring so many years after the origin of the Gregorian calendar, without being concerned with the variation of duration of individual years. However it also means that LOLITA does not know that years are consecutive sharing only endpoints. This must be written explicitly:

(E_{11},R): {ΔF-`subject_`-$F\Delta$: $\mathcal{I}_1$; ΔF-`subject_`-FO: `Years_AD` ;
$\Delta\forall$-`action_`-IO: `ends_`}

(E_{12},R): {ΔF-`subject_`-FO: $\mathcal{I}_1$; ΔF-`subject_`-FO: $\mathcal{Y}_2$;
$\Delta\forall$-`action_`-IO: `starts_`}

and similarly for months, etc.

$\mathcal{Y}_1$ is defined by BJC the origin of the Gregorian calendar; and defines $\mathcal{Y}_2$ – the set of years after 1AD. $\mathcal{V}_1$ are integer multiples of a year duration, starting at zero. Thus the above states that every year y of `Years_AD` that occurs Y years after BJC, is associated with a year y' of $\mathcal{Y}_2$ which occurs $Y+1$ years after BJC, and that there is an instant i of $\mathcal{I}_1$ which ends y but starts y'. The same scheme must be applied to months, days, hours and so on. Every year will be an instance of `Years_AD`, and usually also of $\mathcal{Y}_2$ because of the `antonym_` event. Thus every year which occurred Y years after BFC will inherit the fact it shares its endpoint with a year starting $Y+1$ years after BFC and if $Y > 0$, it shares its startpoint with a year starting $Y-1$ years after BFC.

The calendar for dates BC is defined in a similar manner, except that the `follows_` event takes as `object_s` the years, since they precede BJC. Further, the set of durations qualifying the `follows_` event takes integer year durations of one year or more, rather than zero years or more. This means that the year's date is exactly the number of years between the start of the year and BJC.

• Improving Efficiency of access

The representation of date described above does not allow efficient retrieval of events by date. Since this is a relatively common task, efficiency must be improved.

The inefficiency stems from the low determinism of search in finding a date: starting from the calendar's origin, there are many paths that could be followed to reach the desired dates. By using a specialization of `follows_` say `follows_1`, only to express dates, the search space can be limited to date information. However, each event with `action_ follows_1` connected to the calendar's origin must be traversed to determine its duration, to decide whether it is a good candidate for further search. This is still bad in a large database.

The natural solution is to structure the search space further. Given the nature of dates, it would seem quite easy to build a tree to direct search. Unfortunately the way in which people refer to the Gregorian calendar is asymmetrical which makes the building of a linguistically useful hierarchy difficult. One speaks of the 1980s (meaning 1980 to 1989 included), and of the 20th century (meaning 1901-2000 included). This difference in whether one starts counting at zero or at one means one cannot include the representation of centuries and decennia in the the same tree, despite the benefits to search which would result. The solution adopted is to build a tree corresponding to decennia and to link all but the first decennia of the century to it by an `is_in` event, and to include all the missing years in much the same manner. The hierarchy is a binary tree using specializations of the `starts_` and `ends_` events to give a unique search path down from the root node. The date of the root node can be expressed in the usual way, although the path from it to the calendar's origin is expressed by specialized `follows_` events, say `follows_2`.

The main search tree includes years, months and days. The secondary tree includes centuries associated with the relevant years as described previously. There are obviously many choices of binary tree that could be made for the search tree. In particular, a balanced tree, or a tree encoding the date in binary. The first choice has the advantage of the keeping the tree depth to a minimum. It has two disadvantages: it requires the tree to be balanced each time a new date is inserted or deleted. It also requires the date of each node to be determined as the search proceeds from the root, to determine which branch should be taken. Although this is a task with a unique search path, it requires quite a few events to be traversed. The alternative choice is to use the binary pattern representing the date's number for the search: 4 is 100 in binary, which for a tree of depth 3 would mean the following search path: `ends_` – `starts_` – `starts_`. The depth of the tree thus depends on the range of dates to encode, although it can always be increased by introducing a new root extra branches above the current root. Although this creates deeper search trees, the date of each node down the tree from the root does not need to be checked, which increases speed. However it could decrease robustness: The binary pattern used to search the tree is implicit, whereas the nodes of the tree are all defined by their date expressed by $\texttt{follows}_{-1}$ events. These dates could contradict the information implicit in the order of the `starts_` and `ends_` events forming the tree.

Thus the depth of the search for the first tree is $\log_2(N)+\log_2(12)+\log_2(30)$, rounded up, where N is the year component of the latest date known to the system. As the hierarchy is built on the fly, memory is saved by only building the paths necessary to reach the dates known to the system.

D.4.4 Aspect

D.4.4.1 Aspect and Tense

Statements in the past and the perfect tenses both refer to events that have already occurred with respect to some reference time. However there is a subtle difference in meaning between them: *"John ate an apple"* and *"John has eaten an apple"* are not equivalent. This difference is not in the ordering of events, but in the truth of the event's postconditions. The past does not state whether or not the state resulting from the event is still valid. The perfect states that it is. *"I have bought a car"* means that I own the car now, whereas one could say *"I bought a car"* long after one had sold it. It the event is itself a state or process, rather than resulting in a change of state, there are no postconditions so the perfect refers to the continuation of the state itself. Contrast *"Jane lived in London for two years"*[15] and *"Jane has lived in London for two years"*[16]. Similarly *"I paced the*

[15]from 1965 to 1967 for instance

[16]until now

room the whole day" and *"I have paced the room the whole day"*. In general, the perfect emphasizes the end-point of its statement, often the post-conditions. This difference in the meaning of tense is often called a difference of aspect.

Statements in the progressive tense refer some interval within an event. For instance, *"While John was making breakfast, Mary got dressed"*. However they do not state that the full interval during which they occurred started after the reference event, or that they were over before its end: *"John phoned his mother. The sun was setting, and the light lit the sea stretched out in front of him"* Indeed, they often do not even imply that the event expressed in the progressive actually reached completion. For instance one can say *"I was building a wall, but I ran out of stones"*, but not *"I built a wall, but I ran out of stones"*. As argued in 7.4 *(p. 269)*, the progressive often refers to a different concept than the non-progressive tenses, distinguished for instance by their postconditions: after *"I was washing the floor"*, there is a more or less clean floor, but after *"I washed the floor"*, the floor should be clean. Not all progressives refer to a different concept: *"I was seeing the moon"* implies *"I saw the moon"*. These differences between continuous and non-continuous tenses are often classed as pertaining to aspect.

D.4.4.2 Aspect and Classification of Verbs

A related issue is the classification of verbs[17]. Verbs can be classified into four groups.[18]

The first distinction is between verbs that do not possess a continuous tense, such as to know: **"I am knowing"*. These are state verbs. The other verbs have a continuous tense, such as to learn: *"I am learning"*, which can further be divided into three categories.

Activities are verbs which take time but have no completion point. For example, *"I lived in Paris"*, or *"I was running"*. This does not indicate that they lack postconditions, but that the postconditions do not change suddenly to true at

[17]See [Kenny 63], [Vendler 67], [Mourelatos 81] for more details

[18]There are variations on this scheme, such as [Steedman 77]

some point. Usually, they involve continuous changes. They can be qualified by a "for duration" but not by an "in duration" construction: *"I swam for two hours"* but not *"I swam in two hours"*.

Accomplishments are verbs which take some time and achieve a completion point when their postconditions are satisfied: *"I built a wall"*. These verbs can be qualified by an *"in duration"*, or *"take duration to"* construction, which states the time taken to complete the accomplishment. They cannot be qualified by a *"for duration"* construction. For instance *"I ran a mile in a minute"* but **"I ran a mile for an hour"*. However they are associated with an equivalent activity: *"I ran for an hour"* or *"I was building a wall for an hour"*.

Achievements are verbs that they have an instantaneous character: *"John reached the summit."* They can be qualified by a "in" construction: *"John reached the summit in three hours"*, making them similar to accomplishments. Unlike accomplishments, however, they do not have equivalent activities.

Accomplishments and their equivalent activities are different concepts, but are represented in English by the same words. They are distinguished grammatically, as usually activities are intransitive, and the corresponding accomplishments are transitive. If the intransitive form does not exist, the activity concept is expressed by the progressive.

Sentences can also be classed into an aspect in the same way. Tense is only one component in the choice of a sentence's aspect. Quantification may also play a role. For instance, *"crossed the bridge"* would appear to be an accomplishment. However in the sentence *"Cars crossed the bridge"* is an activity, since the preferred quantification is the distribution over the set. In contrast, *"Students wrote a letter"* is an accomplishment since the preferred reading is not the distribution over the set.

D.4.4.3 Language dependence of Aspect

The distinction between aspect and temporal ordering appears well founded. For instance, whereas the ordering of events expressed by the present in English and French does not vary, the aspect does. In English, *"I eat an apple"* usually conveys a impression of repetition, as it usually occurs in sentences such as *"I eat an apple every day"*. Instead the progressive is used: *"I am eating an apple"*. No such inference applies in French, where the non-progressive form is used: *"Je mange une pomme"*.

In English, the simple present has a habitual meaning: *"John eats a lemon"* (...every day, to prevent scurvy). Or it can be used to express a permanent truth: *"Tigers eat meat"*. These are issues of language usage but are sometimes referred to as aspect.

The differences in aspect and temporal ordering of same tenses between different languages suggest that different tenses express different information by the conjugation of verbs. Indeed, classical Greek provides an example of this using a third voice to the habitual active and passive, the medium, to express that the speaker did something for his own interest: *"I do something for myself"*.

D.5 Linguistic and Conceptual Nodes

The labels associated with each node, presented in 5.1.1.4 *(p. 117)*, allow concepts to be associated with a word corresponding to the concept they refer to. This scheme is sufficient for one unambiguous language. However in practice, LOLITA is to process ambiguous natural languages, in order, say, to translate text. This section presents a solution to this problem.

D.5.1 Linguistic nodes

D.5.1.1 The problem

The words in natural language do not correspond one-to-one to the concepts that prove useful in the inheritance hierarchy. Some concepts cannot be expressed by a single word, others correspond to many words. Similarly, the same word may refer to more than one concepts. If more than one language is introduced, the problem is compounded by the need to distinguish between labels used in different languages. The "one node label per node" scheme is insufficient and too inflexible to provide solutions to all these problems.

D.5.1.2 The solution: linguistic nodes

Concepts are expressed by different words in different natural languages. However, they may all designate the same concept: *"man"*, *"Mann"*, and *"homme"* refer to the same idea. Thus on the one hand there is a linguistic label *"man"*, and on the other there is the concept itself `man`. But the label *"man"* is also a concept: the concept of the label of the concept `man`. As such, it can be thought about, and added to LOLITA's network. Indeed, this is necessary if LOLITA is to be able to represent sentences such as *"Some people call them freedom-fighters, others call them terrorists"*. These label concepts are called linguistic nodes, and the concepts they refer to are conceptual nodes.

Now that these two forms of node have been introduced, one would like to state that one refers to the other. This can be done by an event whose `action_` states the language of the linguistic node. `in_english`, `in_french`, `in_italian`, `in_spanish`, and `in_chinese` are examples of the generic type `in_language`. This takes as `subject_` a conceptual node, and as `object_` its corresponding linguistic node.

D.5.1.3 The use of linguistic nodes

• Ambiguity

One word may be ambiguous and refer to many concepts. For instance, the word "strip" can refer to the concepts of air-strip, strip of land or striptease. Such words are the `subject_s` of many `in_language` events, linking them to all the relevant `object_` concepts.

The frequency of occurrence of a particular word being associated with one of its meanings can be expressed using values[19]: the `in_language` event is the `subject_` of a `has_linguistic_frequency` event, which has as `object_` the relevant frequency. The information thus provided can help in ranking possible candidates meanings for a word, during disambiguation. Since values are used, all the normal value reasoning methods can be used.

- **Linguistic synonyms**

Synonyms are words which refer to the same concept. For instance, *"cab"* and *"taxi"* refer to the same concept. They can be expressed by two `in_language` events with different `subject_s` but the same `object_`. This avoids using two conceptual nodes to represent the same concept, and maintains uniqueness.

As for ambiguous words, the `in_language` events can be qualified by value events stating the frequency of occurrence of a particular word in a particular language. This can be used by a natural language generator to choose its words given the style of the text it is producing. For instance the word with the higher usage frequency would be used in a text for children.

- **Wordless concepts**

Many concepts do not correspond to a word in natural language. For many of these the reason is that they correspond to a phrase in natural language. For instance there is no single word for *"large grey-coloured fish"*. This means that such concepts must be built during the interpretation process, and if one wishes to know their meaning in natural language, a generator must build the appropriate natural language expressions for them.

In some cases, this is due to one language having a word for a concept, but another

[19]For more information on values, see 7.1 *(p. 247)*

lacking one. For instance, in Italian there is no word for *"to mow"*[20]. Instead the expression *"to cut the grass"* is used. Similarly, in English there is no single word to express the Italian *"mollo"* which means disgustingly soft. Here too, the generator should use the definition of the concept to produce a natural language expression for it in the relevant language.

Thus in general, when the natural language word or expression for a concept must be given in a particular language, and there is no word for that concept in that language, the concept is described by using it's definitional properties by a generator. Otherwise, the relevant word is used by traversing the relevant `in_language` event, such as the `in_french` event.

- **Partial and full definitions of concepts**

  - **Concepts for unknown words**

Although one would like all concepts in the database of a large NLP system to be uniquely defined, so that the computer knows the difference between lead and iron, in practice such a well-defined knowledge base is unlikely. Indeed, even if all the concepts in her network were uniquely defined, one would like LOLITA to display robust behaviour while processing a text containing a word unknown to her. This however poses a problem, in that all concepts must be uniquely defined in the network.

Little is known about the concept expressed by an unknown word w. However, it is known is that it is referred to by the word w in the language being processed. Thus, although its meaning is unknown, it should be available for further processing. For instance, LOLITA might know what galligaskins are in the sentence "John wore his galligaskins to work". However she should be able to build a concept for them, and then infer from the is sentence that galligaskins are some form of clothing[21]. For this to be possible, the concept corresponding to the word w must be available in the network, and therefore defined uniquely. This is achieved by using a definitional `in_language` event, as follows:

[20]For further information, see [Morgan et al. 94]

[21]The template event 'wearing' has as `object_` clothing

(E_0,R): $\{\Delta I$-`subject_`-$I\Delta$: `galligaskins_concept`; ΔI-`action_`-IO: `in_english`;
ΔI-`object_`-IO: `galligaskins_word`$\}$

Needless to say, the amount known by LOLITA about a particular concept will limit her ability to reason about it.

∘ **Full and partial definitions**

Even if the definition of a concept is sufficient to distinguish it from all other concepts in the network, it may be a definitional `subject_` of an `in_language` event. This means not all its definitional properties are explicitly stated in the network: the concept is partially defined.

Sometimes all the definitional properties of a concept are explicitly stated within the network. This is for instance the case for the word *"owner"* which is defined as all those people who own some object. Such fully defined concepts are observed to correspond to a particular word in some language:

(E_0,R): $\{\Delta I$-`subject_`-IO: `owners_concept`; ΔI-`action_`-IO: `in_french`;
ΔI-`object_`-IO: `propriëtaire`$\}$

∘ **Aside: unknown meanings**

The scheme presented for unknown words worked when the word was completely unknown. However, sometimes it is a meaning of a word that is unknown. For example, LOLITA might not know the American meaning of "submarine" in the sentence *"The sailor ate the submarine"*[22]. However she can be told that there is another meaning of *"submarine"*. Adopting the previous scheme fails since the word *"submarine"* is not sufficient to define the new concept uniquely. However LOLITA also knows that the new meaning does not refer to the other meanings of the word. She can use this to define the new concept uniquely: it is defined as the concept corresponding to the word *"submarine"* and antonymous with respect to `typeless` to all other meanings of that word [23]. Once this information is available, she can infer that submarine refers to some form of food. This mechanism is limited by uniqueness to one new undefined meaning per word.

[22]Taken from [Hirst 87]

[23]If `antonym_` is eliminated, the same effect can be achieved using the same method as antonyms of events are without `antonym_`: see D.6.3.3 *(p. D-80)*

D.5.2 Expressing language dependent information

The word for a concept is not the only language dependent feature of concepts. Independently of abstractions such as grammar, individual features of words may vary. This is the case for grammatical information such as gender, varying from English to French to German. The same word may be masculine in one language such as *"der Wald"*, masculine in German and *"la forêt"*, feminine in French, both designating the concept *"forest"*. Such information should be expressed at the level of linguistic nodes.

D.5.2.1 Controls on linguistic nodes

Most language dependent information such as gender, the grammatical nature of words (noun, verb, adverb, adjectives...), transitivity of verbs, count or mass nouns, or what information a particular irregular word form is expressing (*"men"* is the irregular plural of *"man"*) and so on is expressed as linguistic controls. The reason for this is that expressing this data as events provides no advantages: unlike other information it is not reasoned about, but only used as immutable facts.

Of special interest is the linguistic control which specifies whether a node is linguistic or conceptual. This can be used to ensure both kinds of nodes are not confused. Also worthy of mention, linguistic and conceptual gender must be distinguished: the German word *"das Mädchen"* signifies *"the young girl"* and is conceptually feminine, but is linguistically neuter.

D.5.2.2 Relations between linguistic nodes

Some words have irregular forms that are not subject to regular morphological inflection. These forms can be represented in the network using relations between linguistic nodes. One such relation is the `root_of` event which takes as `subject_` the root form of the word and as `object_` the irregular form. The particular grammatical features of the irregular form can either be expressed by other relations at the linguistic level, or by linguistic controls.

(E_0,R): {ΔI-`subject_`-IO: `man`; ΔI-`action_`-IO: `root_of`; ΔI-`object_`-IO: `men`}

D.5.3 Sorts and linguistic nodes

Linguistic nodes are linguistic entities corresponding to concepts, and are defined by the linguistic information they carry. This means that there is only one node per combination of linguistic information. Thus, linguistic nodes other that events expressing linguistic relations, are never referred to as definitional by an arc. It is the linguistic control that states that a node is linguistic, and is therefore defined by its controls expressing linguistic information.

D.6 Intension, Extension and Frames of Existence

D.6.1 Intensionality and Extensionality

Statements may either be made about concepts or about the things to which concepts refer. We need to distinguish these cases. For instance, *"the morning star"*, the *"evening star"* and *"Venus"* are different concepts. The morning star is the last point of light in the sky to disappear at dawn; the evening star is the first point of light in the sky to appear at dusk; and Venus is a particular planet of the solar system. An Egyptian slave would have known about the first two concepts quite well, but not about the third. Thus they are different concepts, and refer to different situations. This conceptual level where relations are expressed about the concept is the intensional level. We also now know that the last point of light in the sky at dawn is the same object as the first point of light in the evening, and as the planet Venus. Thus the three concepts refer to the same thing, or have the same extension.

The representation is intensional, in that each concept is represented by a different node. Each concept is defined as a restriction of typeless, using events connected to it either directly, or indirectly via inheritance, by arcs with definitional sorts.

Concepts are different if they are restricted by different events, regardless of what objects satisfy all their restrictions in a particular world. Although each concept is restricted by different events, it is possible for two concepts to happen to have the same extension in the world. That is to say that the same set of objects of the world satisfy the restrictions of both concepts. However this is not so much a consequence of their definition, as a result of the particular world being described. There is no reason derived uniquely from the definitions of the concepts themselves, for the planet Venus and the first point of light in the sky in the evening to be the same object. This happens to be a quirk of the world we live in.

Distinguishing intensionally different concepts is important. For instance, it might be that the set of people who play departmental football, and the set of people from the department who work on LOLITA consists of the same people. It might also be known that these people are fit. However, one would not want some causal reasoning algorithm to derive that one reason for these people to be fit is that they all work on LOLITA. This could happen if the two concepts were not distinguished intensionally, and expressed as one node and is avoided by having a node for each concept.

Intensionality is also involved in different views of the same objects. For instance an astrologer and an astronomer may have very different concepts of the planet Venus. Similarly, flowers can be thought of as beautiful arrangements of petals on top of a green stem, or as the reproductive organs of plants. Choosing the right conceptualization is important in trying to understand why people give each other flowers, or in giving a biology lecture.

D.6.2 Intensional and Extensional equality

Since the representation represents different concepts by different nodes, there must be a means to state that two concepts refer to the same object. Similarly, there are situations where one wishes to state that some node is intensionally equal to another, or that both nodes refer to the same concept. To preserve uniqueness, this is only allowed when there is some doubt as to whether the nodes represent the same concepts. An example of this is the representation of ambiguity: nodes representing ambiguous concepts are intensionally equal to one of their possible meanings.

Equality is expressed using the synonym event[24]. Using this event in conjunction with the definitional and observational sorts on the arcs proves sufficient to represent both varieties of equality: if the synonym event has observational sorts for both its `subject_`s, it states that the `synonym_` event is in no way responsible for the `subject_` nodes being synonymous, since in no way it restricts them. Similarly if the `synonym_` event has definitional sorts for both its `subject_`s, it restricts both in such a way that they are synonymous. This corresponds to stating that each node is defined by all the definitional events connected directly or indirectly via the inheritance hierarchy to both nodes. The first kind of `synonym_` event expresses extensional equality, whereas the second expresses intensional equality.

A third variety of equality is possible: the `synonym_` has a definitional sort on one of its `subject_` arcs, and an observational sort on the other. Here the node connected to the definitional `subject_` is defined by the restrictions of the node connected to the observational `subject_`. This variety of synonym is used for instance to build powersets of a set.

It should perhaps be mentioned that there can be a difference between a statement of synonymity between the concepts $\mathcal{A}$ and $\mathcal{B}$, and the statement that for every instantiation of the concept $\mathcal{A}$ there is an instantiation of the concept $\mathcal{B}$ which is synonymous to it, and vice-versa: in the latter case, statements about the concept itself will not be mirrored over the `synonym_` event to both nodes. For instance, if a concept under consideration $\mathcal{A}$ is the team that won the departmental football trophy, and even if all its elements are definitionally synonymous with those of $\mathcal{B}$, $\mathcal{B}$ will not be inferred to be the team that won the departmental football trophy.

[24]This implies that no processes should compare noderefs to determine whether they are dealing with the same concept.

D.6.3 Synonyms and Uniqueness

Synonyms are necessary to express extensional equality, and prove useful for building powersets, and the representation of ambiguity. However, their use has a price: they can break uniqueness. This section describes the way in which this cost is minimized.

D.6.3.1 Definitional and Observational Sorts

The first situation where synonyms can be used to break uniqueness involves definitional and observational sorts. A synonym event could state that all instantiations of the concept X occur in the concept Y and vice-versa. If Y has the same definitional events as X but also a few more, its additional definitional events are effectively observational events, despite being marked definitional. Such situations are prevented by rendering illegal any `synonym_` event between a node and its ancestor in the inheritance hierarchy.

D.6.3.2 Set Union: `union_`

Similarly, synonyms break the uniqueness of the definition of the `union_` event which can be represented in three different ways.

The first alternative uses quantification only:

$$(E_7,R):\quad \{\mathrm{O}F\text{-}\texttt{subject\_}\text{-}F\forall\Delta:\ \mathcal{D};\ \Delta\forall\text{-}\texttt{action\_}\text{-}I\mathrm{O}:\ \texttt{union\_};\ \mathrm{O}\forall\text{-}\texttt{object\_}\text{-}\exists!\Delta:\ \mathcal{A}\}$$

$$(E_{10},R):\quad \{\Delta F\forall\text{-}\texttt{subject\_}\text{-}F\exists!\mathrm{O}:\ \mathcal{A};\ \Delta F\boxed{\exists!}\text{-}\texttt{subject\_}\text{-}F\boxed{\forall\!\forall}O:\ \mathcal{D};\ \Delta\forall\!\forall\text{-}\texttt{action\_}\text{-}I\mathrm{O}:\ \texttt{synonym\_}\}$$

which simply states that for every set a in $\mathcal{A}$ (the sets which are unions of others) there is a set e of events in E_{10} and a set d of subsets of a in $\mathcal{D}$: for a given d, to every distinct element x of each subset in d corresponds an event of e which states that x is equal to the corresponding element of a. If an element is shared between some of the subsets of d, the corresponding event will have more than 2 `subject_s`[25].

[25]Strictly speaking, this and the next examples do not define `union_` – but state truths about

The second alternative uses belief events to form a concept defined by a disjunction of properties[26].

(E_8,H): $\{\Delta F\forall F$-`subject_`-FFO: $\mathcal{A}$; $\Delta FF\forall$-`subject_`-$FF\exists!O$: $\mathcal{D}$;

$\Delta\forall\forall\forall$-`action_`-$IO$: `synonym_`$\}$

(E_9,R): $\{\Delta FF$-`subject_`-$F\forall FO$: E_8; $\Delta\forall\forall$-`action_`-IO: `or_`$\}$

Here quantification is also quite subtle: E_8's structure mirrors that of $\mathcal{D}$. For each set of `synonym_` events chosen by the `or_` event, one or more are believed, allowing elements to be shared between the subsets of each d (d defined above). Each `or_` event takes many `subject_`s because of the $\forall$ in $FF - F\forall F$.

For completeness a third form is presented, where cardinality and not `synonym_` proves sufficient to express observational `union_`. This form cannot be expressed as an inheritable expression, but does not require the use of logical disjunction. Let $\mathcal{S}$ be the superset and $\mathcal{P}_x$ to be its subsets. For $\mathcal{S}$ to be the union of the subsets $\mathcal{P}_x$, the subsets $\mathcal{P}_x$ must contain every element of $\mathcal{S}$. Let $\mathcal{X}_\backslash$ be the subsets of $\mathcal{S}$ which have as defining property that their elements lack at least one of the properties required for them to be one of the elements of $\mathcal{P}_x$. Every $\mathcal{X}_\backslash$ must be empty since every element of $\mathcal{S}$ must satisfy one of the conjunctions of properties defining the subsets $\mathcal{P}_x$. Thus if each set $\mathcal{P}_x$ is defined by properties $p_{x,y}$, then $\mathcal{X}$ is defined by $\bigwedge_x \bigvee_y f(p_{x,y})$ where $f(p)$ denotes the absence of property p[27] This expression still contains belief connectives (`and_` and `or_`). However, by applying $(a \vee b) \wedge c \equiv (a \wedge c) \vee (b \wedge c)$ the statement can be transformed to an outer `or_` qualifying many `and_`s: $\bigvee_t \cdots \bigwedge_u f(q_{t,u})$. The outer `or_` can be eliminated by replacing the single subset $\mathcal{X}$ by as many concepts $\mathcal{Y}_z$ as the `or_` has arguments. Each $\mathcal{Y}_z$ is defined by the relevant group of absent properties (obtained for each t from $\bigwedge_u f(q_{t,u})$) and each $\mathcal{Y}_z$ has observed `size_` 0: only the subsets $\mathcal{P}_x$ contain elements of $\mathcal{S}$ since $\mathcal{S}$ is their `union_`. Obviously this does not work for properties, such as arcs, which cannot be qualified by `size_` 0.

it. The practical consequence of this is that `synonym_` statements which correspond to `union_` will not be integrated under `union_`'s definition by semantic integration. Although `union_` can be defined in this manner, the integration thus required is beyond the current capacity of semantic integration.

[26]This may be depreciated in the future since defining concepts by a logical disjunction of properties may render reasoning and semantic integration intractable.

[27]Expressed in SemNet by the property's event having observed `size_` zero: see 7.3.2 *(p. 267)*.

The first two forms win on compactness: the third form can require very many empty sets if the subsets of the union are defined by many properties. But the third form does not require additional machinery: no union_ to recognise, no complex quantificational or logical reasoning. Thus union_ is only worthwhile if its cost in code complexity is justified by its reduction of network size. Currently, no decision has been taken, and union_ is maintained as a precaution. The second definition of union_ is preferred, since it involves simpler inheritance. Normalization is assumed to deal with the first case.

D.6.3.3 Antonyms: antonym_

The antonym_ event can be defined by belief events to defined a concept by a disjunction of properties[28].

(E_5,H): $\{\Delta F\forall F$-subject_-FFO: $\mathcal{A}$; $\Delta FF\forall$-subject_-$FF\exists!O$: $\mathcal{D}$;
$\Delta\forall\forall\forall$-action_-$IO$: synonym_$\}$

(E_6,R): $\{\Delta FF$-subject_-$F\forall FO$: E_5; $\Delta\forall\forall$-action_-IO: xor_$\}$

Here quantification is quite subtle: E_5's structure mirrors that of $\mathcal{D}$. For each set of events chosen by the xor_ event, only one is believed, which – since everything is a matter of belief – states that for any set being partitioned by an antonym_, each of its elements occurs in only one of the partitions. Each xor_ event takes many subject_s because of the $\forall$ in $FF - F\forall F$.

The second form cannot be expressed as an inheritable expression, but does not require the use of logical disjunction. The general case is based on the third definition of union_ above. This is augmented by stating that every subset $\mathcal{P}_x$ is a partition of $\mathcal{S}$. I.e., no element of $\mathcal{S}$ is shared by two or more $\mathcal{P}_x$. Two sets $\mathcal{A}$ and $\mathcal{B}$ lack a common element if:

(E_a,H): $\{\Delta\boxed{\forall}\forall$-subject_-$\boxed{\exists!}O$: $\mathcal{A}$; $\Delta\boxed{\forall}\forall$-subject_-$\boxed{\exists!}O$: $\mathcal{B}$;
$\Delta\forall\forall$-action_-IO: synonym_$\}$

(E_b,R): $\{\Delta F$-subject_-FO: E_5; $\Delta\forall$-action_-IO: size_; $\Delta\forall$-object_-IO: 0 $\}$

By using this for every combination of pairs of sets of $\mathcal{P}_x$, one can state the sets

[28]This may be depreciated in the future since it may render reasoning and semantic integration intractable.

$\mathcal{P}_x$ share no common element. Even though E_b can be shared using arbitrary quantification, this results in $\frac{n(n-1)}{2}$ `synonym_` events, where n is the number of partitions.

The above expression is augmented by many special cases, such as the complement set. The complement c of any set s with respect to its parent $\mathcal{S}$ is defined by a form of negation of s's definitional properties: if s is defined by the existence of property x and the absence of y, then c is defined by the absence of x or the presence of y. Any set and its complement are antonyms with respect to their parent.

The general case of the second form fails for events like antonym verbs: the defining property of $\mathcal{P}_x$ are `action_` arcs, which cannot be qualified by `size_` 0. However this case can also be dealt with: antonym verbs occur in pairs, such as *"like"/ "dislike"*. If one of the template events t is defined by its `action_` the other event u can be defined as being the subset of $\mathcal{S}$ which does not include elements of t, where S is u and t's common superset. For instance, $\mathcal{S}$ could be *"to have feeling for"*, t could be defined by `like`, and could define u which defines `dislike`.

Just as was the case for the `union_` event, the first form suffers from its additional reasoning requirements, and the second from its space consumption. The same trade-offs must be made. Similarly, the first definition is currently preferred because it involves simpler inheritance.

D.6.3.4 Powerset: `powerset_` and `infinite_powerset`

`synonym_` can define `powerset_`: For each element x in $\mathcal{X}$, there is a set y in $\mathcal{Y}$ which contains all the possible sets that one can build from the elements of x:

(E_{30},R): {ΔF-`subject_`-FO: $\mathcal{X}$; O$\forall$-`action_`-$I\Delta$: `powerset_`;
ΔF-`object_`-FO: $\mathcal{Y}$}

(E_{31},R): {$\Delta F\boxed{\forall\forall}$-`subject_`-$F\boxed{\exists!}$O: $\mathcal{X}$; ΔFFF-`subject_`-$FFF\Delta$: $\mathcal{Y}$;
$\Delta\forall\forall\forall$-`action_`-$I$O: `synonym_`}

Notice that the action `powerset_` is defined by E_{31}. The same should be done for `antonym_`, `union_`, and all other events that can be completely defined within SemNet. However, it obfuscates the presentation, so generally has not been done.

Similarly, the infinite powerset can be built:

(E_{32},R): $\{\Delta F$-`subject_`-FO: $\mathcal{X}$; $\mathrm{O}\forall$-`action_`-$I\Delta$: **`infinite_powerset`**;
ΔF-`object_`-FO: $\mathcal{Y}\}$

(E_{33},H): $\{\Delta FF$-`subject_`-$FF\Delta$: $\mathcal{X}$; $\Delta F\boxed{\exists!}$-`subject_`-$F\boxed{\forall\forall}\Delta$: $\mathcal{Y}$;
$\Delta\forall\forall$-`action_`-IO: `synonym_`$\}$

(E_{34},H): $\{\Delta FF$-`subject_`-FFO: $\mathcal{Y}'$; $\Delta F\boxed{\exists!}$-`subject_`-$F\boxed{\forall\forall}\Delta$: $\mathcal{Y}$;
$\Delta\forall\forall$-`action_`-IO: `synonym_`$\}$

(E_{35},R): $\{\Delta FF$-`subject_`-FFO: E_{34}; $\Delta F\forall$-`subject_`-$F\exists!O$: E_{33};
$\Delta F\forall$-`action_`-IO: `or_`$\}$

(E_{36},R): $\{\Delta F$-`subject_`-FO: $\mathcal{Y}$; ΔF-`subject_`-$F\Delta$: $\mathcal{Y}'$;
$\Delta\forall$-`action_`-IO: `synonym_`$\}$

However, the use of an `or_` means that the usage of this definition may become depreciated: it might render reasoning and semantic integration intractable.

D.6.3.5 No Uniqueness Break for `spec_`

There is however some good news. Synonyms might be expected to break uniqueness with respect to `spec_` relations: $\forall x \in X\ \exists y \in Y\,.\,(x = y)$ would indicate that X is a subset of Y. However it should be remembered that when an event refers to a concept with existential quantification, it refers to all of the concepts' instantiations. This prevents such schemes from working. A similar fate awaits schemes attempting to emulate `inst_` relations using arbitrary quantification: arbitrary quantification is a shorthand for an implicit `inst_` relation. However to avoid situations where arbitrary quantification is expanded out resulting in a node a being synonymous to another b which itself is an instance of a third c, synonym events are not allowed to refer to a `subject_` using arbitrary quantification.

D.6.4 Frames of existence

D.6.4.1 The need for frames of existence

• The problem: Are there more than one different extensions of a concept ?

Before attacking the meat of this section, an important point will be emphasized. It should be remembered at all times throughout this discussion that all references to the world, the real world or reality do not refer to any absolute reality, if such a thing exists. Instead they refer to the world that the agent believes to be real, and to the partitions of his sensory perception which define his understanding of that world. This notion is quite subtle, especially when discussing the beliefs other agents may hold about the world. It should be remembered that to an agent, the notion of other agents is yet another partitioning of his perception. Thus when he refers to their beliefs about the world he is not referring to what they really believe or even to their partitioning of their perception, since they may not even exist. Instead he refers to what he believes these agents as partitions of his sensory perception believe about the world which is also a partition of his sensory perception. Now that this has been clarified, the problem can be discussed.

In the previous subsections, extension was discussed in terms of some world. This notion needs clarification. Intension corresponds to the partitioning of all possible sensory input, or "meaning space" discussed earlier (2.4 *(p. 13)*). Extension on the other hand would seem to be from the previous discussion, objects that the agent has experienced appearing in his perceptual space, and therefore those that he believes to be part of the world in which he lives and acts. However, there may be other forms of extension. For instance, a fictional character in a book is referred to as an object, extensionally rather than intensionally: *"Sherlock Holmes lit his pipe, and coughed bitterly"* does not refer to the concept of Sherlock Holmes, but to the person himself. The problem is that human agents know that Sherlock does not exist physically in the world. This knowledge prevents them from making mistakes such as asking someone to interview Sherlock Holmes. In order for LOLITA not to make mistakes, she needs such knowledge, and hence the ability to represent it.

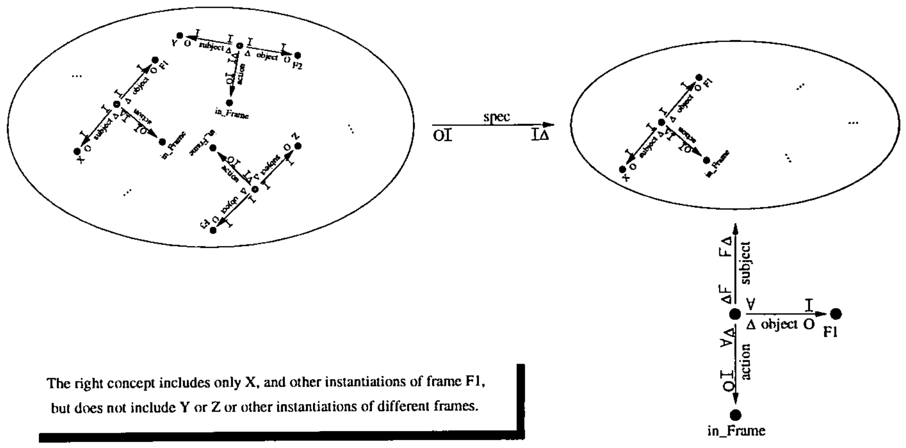

spec
OI
IΔ
subject
object
action
in_Frame
F1
F2
X
Y
Z
The right concept includes only X, and other instantiations of frame F1, but does not include Y or Z or other instantiations of different frames.

Figure D.1: Frames of existence

A less trivial example occurs when different plans are to be evaluated, for instance for their anticipated environmental impact. Although the actions and their effects can be discussed for each plan, none of them "exist" in that they have not yet been applied. Indeed, the various options discussed may involve conflicting actions which could not "exist" simultaneously: an extreme example could involve someone doing something in one of the plans, although he had been killed previously in another.

- **The solution: Partitioning extension into frames**

It seems therefore clear that the same concept can correspond to different extensions depending on the situation considered. The solution adopted involves the notion of frames of existence. These frames correspond to the different situations in which the possible extensions of a concept are encountered. For instance, in the "cartoon" frame of existence, flying elephants could have a non-empty extension including Dumbo, but in the "reality" frame of existence they would have an empty extension. The membership of a concept's instantiation to a particular frame $\mathcal{F}_1$ is simply modeled: the instantiation is the `subject_` of an `in_Frame` event, which has as `action_` "`in_Frame`" and as `object_` the node $\mathcal{F}_1$. $\mathcal{F}_1$ is an instance of the set of frame of existence markers. An object may belong to more than frames of existence, by being the `subject_` of more than one `in_Frame` events with different `object_s`. The set of extensions $\mathcal{E}$ of a particular frame of existence $\mathcal{F}_1$ is chosen by using a definitional `in_Frame` event with `object_` $\mathcal{F}_1$.

D.6.4.2 The use of frames of existence

- **Structuring frames of existence**

◦ **Structure in the inheritance hierarchy**

Frames of existence are expressed in the inheritance hierarchy. They are antonym_ to concepts to which they cannot apply with respect to typeless. These include frame of existence markers, the in_Frame event, spec_ and inst_ relations, and other things which do not occur in possible worlds.

Some frames of existence can be more specialized than others. This is expressed using the inheritance hierarchy, as for instance for:

(E_{11},R): {ΔF-subject_-$F\Delta$: $\mathcal{F}_0$; $\Delta\forall$-action_-IO: in_Frame; $\Delta\forall$-object_-IO: f_0}
(E_{12},R): {ΔF-subject_-$F\Delta$: $\mathcal{F}_1$; $\Delta\forall$-action_-IO: in_Frame; $\Delta\forall$-object_-IO: f_1}
(E_{13},R): {ΔF-subject_-$F\Delta$: $\mathcal{F}_2$; $\Delta\forall$-action_-IO: in_Frame; $\Delta\forall$-object_-IO: f_2}
(E_{14},R): {ΔF-subject_-$F\Delta$: $\mathcal{F}_4$; $\Delta\forall$-action_-IO: in_Frame; $\Delta\forall$-object_-IO: f_4}
(E_{15},R): {ΔF-subject_-$F\Delta$: $\mathcal{F}_5$; $\Delta\forall$-action_-IO: in_Frame; $\Delta\forall$-object_-IO: f_5}
(E_{16},R): {ΔI-subject_-IO: $\mathcal{F}_0$; ΔI-action_-IO: spec_; ΔI-object_-IO: $\mathcal{F}_1$}
(E_{17},R): {ΔI-subject_-IO: $\mathcal{F}_0$; ΔI-action_-IO: spec_; ΔI-object_-IO: $\mathcal{F}_2$}
(E_{18},R): {ΔI-subject_-IO: $\mathcal{F}_1$; ΔI-action_-IO: spec_; ΔI-object_-$I\Delta$: $\mathcal{F}_3$}
(E_{19},R): {ΔI-subject_-IO: $\mathcal{F}_2$; ΔI-action_-IO: spec_; ΔI-object_-$I\Delta$: $\mathcal{F}_3$}
(E_{20},R): {ΔI-subject_-IO: $\mathcal{F}_3$; ΔI-action_-IO: size_; ΔI-object_-IO: 0}
(E_{21},R): {ΔI-subject_-IO: [$\mathcal{F}_4$, $\mathcal{F}_5$]; ΔI-action_-IO: antonym_;
ΔI-object_-IO: $\mathcal{F}_1$}
(E_{22},R): {ΔI-subject_-IO: $\mathcal{F}_1$; ΔI-action_-IO: spec_; ΔI-object_-$I\Delta$: $\mathcal{F}_4$}
(E_{22},R): {ΔI-subject_-IO: $\mathcal{F}_1$; ΔI-action_-IO: spec_; ΔI-object_-$I\Delta$: $\mathcal{F}_5$}

Here the frame of existence $\mathcal{F}_2$ is more specialised than $\mathcal{F}_0$. For instance, $\mathcal{F}_0$ could be the frame of existence of fictional entities, and $\mathcal{F}_2$ that of those appearing in books by Franz Kafka. Frames of existence can be stated to involve no common elements as shown with $\mathcal{F}_3$, or be antonyms as for $\mathcal{F}_4$ and $\mathcal{F}_5$ with respect to $\mathcal{F}_1$. The first situation occur between entities appearing in Chinese mythology and those appearing in Norse mythology, whereas the second could refer to the only two books written by a particular author.

○ Storage considerations

Frames of existence need not occupy much storage space, since once they have been defined, normal set relations can be used to refer to the instantiations of a particular concept in a particular frame of existence. This node corresponds to the intersection of the intensional concept under consideration, and the restriction of typeless corresponding to all objects occurring in the frame of existence under consideration. For instance $\texttt{Dogs}_{\mathcal{F}}$ is the extension of all dogs in the frame of existence $\mathcal{F}$:

(E_0,R): {ΔI-`subject_`-IO: `Dogs`; ΔI-`action_`-IO: `spec_`;
ΔI-`object_`-$I\Delta$: $\texttt{Dogs}_{\mathcal{F}}$}

(E_1,R): {ΔI-`subject_`-IO: $\mathcal{F}$; ΔI-`action_`-IO: `spec_`; ΔI-`object_`-$I\Delta$: $\texttt{Dogs}_{\mathcal{F}}$}

Not all events need have their own `in_Frame` event. Indeed, it is rare for there to be a reason for an event to refer to an `in_Frame` event. Thus to save on net size, a degree of compression can be achieved by using arbitrary quantification:

(E_1,R): {ΔA-`subject_`-IO: [N_0, N_1, N_2 ...]; $\Delta\forall$-`action_`-IO: `in_Frame`;
$\Delta\forall$-`object_`-$\exists!$O: $\mathcal{F}_0$}

Here E_1 serves as an `in_Frame` event for N_0 and each of its other `subject_s`. These `subject_s` are not partial since no two arcs referring to the same concept with arbitrary quantification can refer to the same instantiation of that concept. Searchwise, this results in a substantial loss of determinism of search from the event to the node since their are so many `subject_s` to choose from. However the only reasons for such searches would be if the `in_Frame` event were referred to which is not the case when this shorthand is used, or if all nodes in a particular frame of existence were being searched for, in which case determinism of search is not relevant.

• Referring to non-existent entities

The `size_` event is used to state the number of entities that exist in a particular frame of existence. If there are none, the number used is zero. This allows the existence of a particular entity in a particular frame of existence to be discussed: *"The existence of Pluto was conjectured by mathematics, and proven by observation"*[29].

[29]Many of the examples in this section were taken from [Hirst 91].

Similarly *"The existence of Vulcan was conjectured by mathematics, but disproven by observation"*. Even the non-existence of things can talked about (by reference to `size_` zero events): *"It's a good thing that carnivorous cows don't exist!"*

The `size_` event can be used in belief relations. This allows *"John believes in ghosts"* to be represented as John believes that the size of the concept ghosts in the real-world frame is greater than zero. LOLITA on the other hand may not, and will connect the concept to a real status `size_` zero event. Events too can be described in this way *"John believes in the resurrection of Christ, but Mary doesn't"*. Similarly concepts such as *"Christianity is monotheistic: Christians believe there is one god"* can be expressed as stating that in the relevant frame of existence (for example, the real-world frame), the `size_` of the concept defined by the properties that gods have is one.

Every node may have a different frame of existence to its neighbours. This allows statements such as *"John believes that every dragon has a tail"*:

(E_0,H): {ΔF-`subject_`-FO: `dragon`; $\Delta\forall$-`action_`-IO: `has_part`;
ΔF-`object_`-$F\Delta$: `tail`}

(E_1,R): {$\Delta\forall$-`subject_`-IO: `John`; $\Delta\forall$-`action_`-IO: `believe`;
ΔF-`object_`-FO: E_0}

(E_2,R): {ΔAF-`subject_`-FO: E_1; ΔAA-`subject_`-IO: `John`;
$\Delta\forall\forall$-`action_`-IO: `in_Frame`; $\Delta\forall\forall$-`object_`-IO: `real_world_frame`}

If the dragon having tails and the believing events could not be in different frames of existence, the believing event's F-`object_`-F quantification would force both events to have the same number of instantiations in the same frame of existence. The frame of existence that must be considered is the real-world frame since that is where the believing event would occur. However in the real-world frame there are no dragons, (and hence no dragons having tails either), so the number of events must be zero. This would imply that the representation does not allow the statement considered, or other such statements about a concept, irrespective of whether it has real-world extension. If every node can have a different frame of existence, such beliefs about concepts can be represented. In the example, the nodes *"dragons"*, *"dragons have tails"* and *"dragon tails"* are intensional (have undefined extensionality) whereas

"John" and *"John believes that ..."* are restricted to the real-world frame.

Concepts can be defined to "exist", yet be intensional. For instance one can refer to *"the gold mountain that exists"* without for that matter believing it has an existence in the real-world frame of existence. Similarly such a concept is represented as a node of undefined extensionality corresponding to the set of golden mountains, but with definitional `size_` 1. This states that there is one golden mountain that exists intensionally, and therefore in some frame of existence, but not necessarily any frame of existence known to LOLITA.

- **Referring to objects across frames of existence**

In some situations one may want to make references between extensions of different frames of existence. For instance *"John models himself upon Sherlock Holmes"*. Here John and the modeling event are in the real world frame of existence, whereas Sherlock Holmes is fictional. Frames of existence allow this since the relevant frame of existence can be specified for each node separately. Even causal relations can occur between frames of existence: *"I dreamt John killed my wife last night, so I was rather unfriendly when I met him in the office this morning."*

In other situations, one might even want to define extensions in one frame of existence referring to extensions in another. Consider *"I had a dream of a better world, in which they are free who here are oppressed, and they are well who here are sick and lame"*. The set of people in the better world corresponding to the oppressed ones in this is defined with respect to this world. This can be expressed using the appropriate synonym event:

(E_0,R): {O∀-subject_-∃!Δ: oppressors; Δ∀-action_-IO: oppress;
O∀-object_-∃!Δ: oppressed}

(E_1,R): {Δ∀-∃!Δ: oppressors$_a$; Δ∀-IO: oppress;
Δ∀-object_-∃!Δ: oppressed$_1$}

(E_2,R): {ΔAF-subject_-FΔ: oppressed$_1$; ΔAF-subject_-FΔ: E_1;
Δ∀∀-action_-IO: in_Frame; Δ∀∀-object_-IO: real_world}

(E_3,R): {Δ$\boxed{\forall}$∀-$\boxed{\exists!}$Δ: oppressors$_b$; Δ∀∀-IO: oppress;
Δ$\boxed{\forall}$∀-object_-$\boxed{\exists!}$O: oppressed$_1$}

(E_4,R): {ΔF-subject_-FO: E_3; $\Delta\forall$-action_-IO: size_; $\Delta\forall$-object_-IO: 0}

(E_5,R): {ΔAF-subject_-FO: oppressed$_1$; ΔAF-subject_-$F\Delta$: E_3;
$\Delta\forall\forall$-action_-IO: in_Frame; $\Delta\forall\forall$-object_-IO: dream_world$_1$}

(E_6,R): {ΔI-subject_-IO: oppressors; ΔI-action_-IO: spec_;
ΔI-object_-$I\Delta$: oppressed$_1$}

(E_7,R): {ΔI-subject_-IO: E_0; ΔI-action_-IO: spec_;
ΔI-object_-$I\Delta$: E_1}

The set of people in the better world corresponding to the oppressed in this world are observed to be free[30]. This works because each instantiation of an event is also associated with one or more frames of existence. It is therefore possible for the same people to be sick in one world, and well in another, since the being sick or being well occurs in different worlds.

Although extensions in one frame of existence can be defined by referring to extensions in another, the intension of a concept in "meaning space" may only be defined in terms of restrictions on typeless, and not in terms of its extensions in any particular frame of existence.

D.6.4.3 Intension as pan-frame intension

The intensional nature of the representation means that when extension has been partitioned into frames, intension was also: there is now a concept of Venus in the dream-world, in the real world, and in Sherlock Holmes' world, but there is also a concept of Venus independent from any particular world. It is this frame independent concept which corresponds to what was referred to previously as the intension of a concept. These concepts form an intensional meaning space which corresponds to restrictions of typeless, unrestricted by any in_Frame event.

- **Antonyms in the Intensional space**

Antonyms are used in the inheritance hierarchy to partition concepts. For instance, the concept "human" is the object_ of an antonym_ event with subject_s "men"

[30]they do not correspond to all free people in the better world, but a subset

and "women". However expressing an **antonym_** event in the intensional space would preclude any person being a man in one frame of existence and a woman in another. This is clearly not desirable since it is possible for a man to dream that he is a woman. However, it is not possible for instantiations of the literal concept of human to be both male and female, or neither in the same frame of existence. Thus a special antonym relation, **i_antonym**, is used. It is restricted to having as subject_s and object_, concepts whose instantiations occur in the same frame of existence:

(E_0,R): $\{\mathrm{O}F$-subject_-$F\forall\Delta$: $\mathcal{X}$; $\Delta\forall$-action_-$I\mathrm{O}$: **i_antonym**; $\mathrm{O}\forall$-object_-$\exists!\Delta$: $\mathcal{Y}\}$

(E_1,R): $\{\Delta F$-subject_-$F\forall\mathrm{O}$: $\mathcal{X}$; $\Delta\forall$-action_-$I\mathrm{O}$: **antonym_**; $\Delta\forall$-object_-$\exists!\mathrm{O}$: $\mathcal{Y}\}$

(E_2,R): $\{\Delta FFF$-subject_-$FFF\mathrm{O}$: $\mathcal{X}$; $\Delta\forall\forall\forall$-action_-$I\mathrm{O}$: **in_Frame**; $\Delta\boxed{\forall}\forall\forall$-object_-$\boxed{\exists!}\mathrm{O}$: $\mathcal{F}\}$

(E_3,R): $\{\Delta FF$-subject_-$FF\mathrm{O}$: $\mathcal{Y}$; $\Delta\forall\forall$-action_-$I\mathrm{O}$: **in_Frame**; $\Delta\boxed{\forall}\forall$-object_-$\boxed{\exists!}\Delta$: $\mathcal{F}\}$

- **Undefined extensionality**

An issue related to intensionality is undefined extensionality. This corresponds to a reference to an extension that may or not be believed to exist in what is considered to be the real world, which is implicit in some events. For instance, consider the verb "to need", as in *"I need a hammer"*. The word "hammer" does not refer to any particular instance of a hammer. In fact, there may be no such thing as a hammer, as illustrated by the sentence *"I need a dragon"*. Rather it is some possibly non-existent[31] instance of the idea of a hammer that is referred to: *"I need something which would allow me to travel to Australia in a matter of seconds"* is an example of a concept for which no extension yet exists in the real world. It is important for the computer to distinguish objects with undefined extensionality from those with extensions in the real world. For instance, assume that the computer is told that the user needs a hammer and is then asked for the list of hammers it knows of, and their places. If it cannot distinguish defined and

[31](in this world)

undefined extensionality, it will also list *"the hammer that the user needs"* as a known hammer. Thus a representational device is needed to distinguish between the two types of statement.

Undefined extensionality is achieved by not specifying a frame of existence for the relevant concept. Thus *"I need a hammer"* has nodes which are defined to be part of the real world as `subject_` and event, whereas its `object_` node is not defined by any frame of existence. This does not prevent the event from referring to an instance of the concept which is in a particular frame of existence. For instance one can still say *"I need the hammer in the kitchen"*. Similarly, events can have undefined extensionality: *"I want to throw John into the river"* refers to an event which does not necessarily have an extension in the real world.

Another less obvious example of a relation which does not require any particular frame of existence for its arguments is "taller than": one can say *"Paul is taller than Winnie the Pooh"*. However, this does not imply anything either about the frame of existence of Paul or that of Winnie the Pooh.

Other events assume that their `subject_`, `object_` and event all belong to the same frame of existence. For instance, kissing and seeing are examples of this: *"I saw a unicorn today"* sounds strange, since unicorns are not (usually considered) real, whereas I am real. Similarly *"John kissed Mary"* would make sense if John and Mary were both real, or both fictional, but not if one were real and the other fictional.

Requirements for each event as to which frame(s) of existence (if any) it, its arguments, and the events qualifying it require can all be expressed on the template events in the usual fashion:

(E_0,R): $\{\mathrm{O}F\forall$-`subject_`-$F\forall\Delta$: `beings`$_1$; $\Delta\forall\forall$-`action_`-$I\mathrm{O}$: `need`;
$\mathrm{O}\boxed{\forall\forall}$-`object_`-$\boxed{\exists!}\Delta$: `needs`$\}$

(E_1,R): $\{\Delta FF$-`subject_`-$FF\mathrm{O}$: `beings`$_1$; $\Delta\forall\forall$-`action_`-$I\mathrm{O}$: `in_Frame`;
$\Delta F\forall$-`object_`-$F\Delta$: $\mathcal{F}\}$

(E_2,R): $\{\Delta FF$-`subject_`-$FF\mathrm{O}$: E_0; $\Delta\forall\forall$-`action_`-$I\mathrm{O}$: `in_Frame`;
$\Delta F\forall$-`object_`-$F\Delta$: $\mathcal{F}\}$

- **Template events and intensional concepts**

Template events are intensional concepts: they define the concepts an event can have as `subject_`(s) and `object_`(s), independently from any frame of existence. They must therefore be in the intensional space.

- **Belief about intensional concepts**

Another kind of problem involves preconceived ideas. For instance, Joan may wish to meet the Scottish golf champion because she believes he will be rich. However she does not wish to meet John, the actual Scottish golf champion, who is poor. In this case she believes that the observational attributes of the concept of *"Scottish golf champion"* involve being rich. Because of this, John cannot be an instantiation of a Scottish golf champion for her in the real-world frame of existence since he would contradict her fact of all instantiations of the concept of Scottish golf champion must be rich. Such situations are possible as long as Joan herself does not believe that the Scottish golf champion and John are the same person. If she did, the only recourse for the agent would be to assume that she does not mean the same thing by the concepts used (such as 'rich') as it does.

- **Intensionality and sorts**

Intensional concepts have certain limitations concepts in frames of existence lack.

Intensional concepts are not qualified by an observational `size_` event: This would mean that the number of extensions of a concept are predetermined, in any possible world one can imagine. But this number is only predetermined if `size_` is definitional, otherwise it is only a quirk of the real world, or any other frame of existence. Thus,

$$(E_0,R)\text{:}\quad \{\Delta\forall\text{-}\mathtt{subject\_}\text{-}\exists!\mathrm{O}\text{: }\mathtt{ants};\ \Delta\forall\text{-}\mathtt{action\_}\text{-}I\mathrm{O}\text{: }\mathtt{eat};\ \Delta\forall\text{-}\mathtt{object\_}\text{-}\exists!\mathrm{O}\text{: }\mathtt{food\_pieces}\}$$

means that there is an event for every combination of subject and object: There is an eating event between every ant of `ants` and every piece of food of `food_pieces`, at every time, in every location, believed by every person... The size of this set is clearly not finite. To reduce it to a subset, a definitional `in_Frame` event is connected to E_0: now only the events that (are believed to have occurred) in E_0's frame are considered.

Indeed, any observational statement about an event need be considered very carefully. For instance, if the concept *"Chinese Telephones"* is qualified the the observational event *"is black"*, it would mean that it is not possible even to imagine a Chinese telephone that were not black. Most statements in the intensional space will thus limit themselves to the definition of the terms of the network.

D.6.4.4 The choice of frames of existence

Although the notion and representation of frames of existence has been presented, this thesis will not argue for any particular ontology of frames of existence. The particular use of frames of existence depends on the particular reasoning that one wants to perform with them, and as yet there is not enough evidence to make any conclusive choice. However, possible uses of frames of existence will be suggested in order to improve understanding of their scope.

Frames of existence provide a means of structuring knowledge. In particular they can be used to contribute to reducing the search-space involved in processing. However to be used in this way, the knowledge must be structured into chunks which are as independent as possible. For instance, real-world events and fictional events can be considered as independent with respect to causal reasoning since *"Sherlock Holmes burned London down"* has no direct causal effect on the real-world London.

- **Possible usages of frames**

○ **The real-world frame**

The real-world frame corresponds to things that exist or existed physically and causally interacted with other objects of the real-world. This includes objects as diverse as the people in my lab, electric fields, the cold-war and so on. The real-world frame is shared between agents, in that John may believe that dragons exist whereas LOLITA does not. This would mean that John believes that dragons can causally interact in the same world as his wife lives in[32].

One important characteristic of the real-world frame is the requirement for internal consistency. For instance one cannot sell a cake and eat it. Any frame of existence that is hoping for real-world status must have such consistency.

The real-world frame can be divided into subframes of existence.

◇ Different civilizations

Different frames of existence can be assigned to different ancient civilizations in the past, and to a lesser extent different cultures in the present. Even if they share a common language, their ideas of what concepts such as ownership may vary greatly. In some cultures it is possible for people to be owned. In other cultures, albeit rarely, an animal can own things too. More generally customs, the way in which people interact, and expected behaviour vary from culture to culture. These differences can be considered sufficiently important for a different real-world subframes to be assigned to each.

◇ Micro Theories

As discussed in A.2 *(p. A-7)*, scientific knowledge consists of a set of theories, whose domain of application vary. However where their domains of application intersect, the results they provide must be equal (within the degree of accuracy defined by the theory itself). Thus predictions can be made within the quantum world, the classical world, the relativistic world or the relativistic quantum: reasoning occurs within one of these worlds. One could therefore assign each such theory a frame of existence which is associated with its conditions of use. Statements referring to the assumptions of a particular micro-theory are also associated to the relevant frame of existence.

However there is no need to limit this use of frames to scientific knowledge. Common sense knowledge such as "naive physics" or can also be structured in this way. For instance, naive physics can be given a domain of application: it applies to slow moving (with respect to the speed of light) earth bound objects within a certain

[32]Remember that this world need not exist, but just be a partitioning of the input of his senses.

temperature range, and so on: objects fall towards the ground only on the earth, not in space. Supercooled liquids that can syphon themselves out of containers are not included. On earth in the middle of the day a cloudless sky may be blue, or orange if polluted, but definitely not green. Similarly other forms of knowledge can be structured in this way: usual human behaviour within a particular society, animal and plant behaviour, classical versus pop music, economics, and so on.

The "real world" frame of existence has as distinguishing feature that it is associated with a stringent set of causal models that must hold for all statements within it. Practically this can help determine whether a text should be considered fictional or real: *"The statue turned its head towards me and winked"*.

o **Fictional existence**

Different stories, or series of stories can be assigned a different frame of existence. This ensures that fictional events, such as *"Sherlock Holmes burnt down London"* do not get confused with real-world events. There is no requirement for statements in a fictional frame of existence to be consistent. This allows fictional statements such as found in *"Alice in Wonderland"* to be processed, be it translation or quote finding.

Another form of fictional existence is dream existence, involving people's dreams. Like stories these need not be consistent, and should not be confused with real-world events.

o **Abstract existence**

Certain concepts exist by definition, in an abstract sense. These include mathematical objects, such as the number 841, and relations such as the square root of a number. A separate frame of existence can be devised for such objects. In particular it is possible to state that they occur only in a particular frame of existence: mathematical objects are defined among other properties as being objects of that frame of existence. The abstract frame of existence can be declared an antonym to all other frames of existence, leading to the impossibility to talk of real-world extensions of the concept.

A consequence of this would be that sentences such as *"John does not believe in the number one"* would not make sense. Sentences of the form *"John believes in X"* imply that if John is in the real-world, he believes X has a real world existence. However the number one is defined not to have a real-world extension. The sentence involves a type clash by postulating that the number one can have a real-world extension, before the issue of the size of its real world extension can be discussed. This differs from other entities which can be believed an influence on the real-world frame: *"John believes in reincarnation"*.

This structure can be useful in disambiguation too: *"This is a triangle"* really refers to the representation of a triangle, rather than the mathematical concept of it. For instance this allows the same number to be represented in many different ways, depending on people's handwriting, and so on. Whereas you can rub the physical representation of a triangle out, you cannot rub out the concept itself.

∘ Possibility and Plans

Possibility corresponds to events which could occur in the real-world but which do not necessarily occur. For instance, *"You could have done the washing"* refers to an event which could have occurred but did not[33]. Possible information can be expressed by a separate frame of existence: the possible-world frame.

The real-world and possible-world frames of existence share certain behaviours: for instance for a frame of existence to be a possible-world frame, it must be internally consistent and consistent with the real world (previous to the date at which it would have occurred in the case of past plans). For an event to occur in the real-world frame, it must be possible. Therefore all events in the real-world frame must also also have extensions in the possible-world frame. This means that the real-world frame is a subframe of the possible-world frame. Each possible-world subframe must include a substantial proportion of the real-world frame, although not necessarily all of it: in the possible-world subframe in which the last world-war

[33]Possibility must be distinguished from potentiality which refers to the ability to perform an action, a situation in which the event's preconditions were met in order for the event to be possible. Potentiality is involved in statements such as *"I can swim"*, or *"Birds can fly"*. Both forms can occur in the same statement *"I wish I saved her, but I couldn't because I can't swim"*.

was won by the Nazis, many events would be shared with the real-world frame (events outside the causal influence of this event, such as the Schumacher-Levy comet striking Jupiter, or the battle of Hastings of 1066), but many events would be inconsistent with the real-world frame (like the building of the Berlin Wall). Similarly if an extension does not exist within the possible-world frame, it cannot occur in the real-world frame: *"telepathy is impossible"*. Even statements such as *"telepathy might be possible"* are correctly analysed as not stating that it actually occurs in the real-world frame.

Another perhaps more practical example of possible-world subframes are plans: one might have various alternative plans. Each must be self consistent, consistent with the part of the real-world they depend on causally, but they can be mutually inconsistent. Expected events are also included in the possible-world frame.

Some events involve arguments that exist only in the possible-world frame and not in the real-world frame. An example is the `object_` of cancel: *"Kevin canceled the 5-aside football match"* refers to a non-existent but planned football match.

D.6.5 The idea of a concept

Sometimes concepts themselves are referred to. For instance, *"The idea of freedom as a right for all is relatively recent"*[34]. This cannot be expressed directly by referring either individually or universally to an intensional concept since this would be equivalent to referring to all the instances of the concept in all possible frames of existence, at all possible times (etc...) as a group or individually. Instead a special event is used to map a symbol to its meaning, the `means_` event. The symbol referred to here is the idea itself. The example would therefore be represented by a idea symbol node as `subject_` of a `means_` event with as individually quantified `object_` the event *"all people have the right to be free"*. The idea symbol node is also the `object_` of a *"people invented relatively recently"* event.

Different people's definition of a concept may vary. For instance LOLITA may

[34]Note the difference to *"the idea of freedom is appealing"* which refers to the extension of freedom in no particular frame of existence

believe that dragons are a variety of dinosaur, whereas John might believe that they are large human-eating lizards. These are two different conceptions of the same concept. Since the semantic net is intensional, the two different conceptions will be assigned two distinct nodes. One can argue that the word "dragon" is ambiguous, referring either to John or LOLITA's between the two conceptions: John believes the word dragon refers to his sort of dragon, whereas LOLITA believes it refers to hers. However this argument must be applied to all languages known to LOLITA in that John also believes that "un dragon" and "ein Drache" refer to his sort of dragon too. Alternatively one could argue that it is the idea of dragon that varies: the words "dragon", "dragon", "Drache" all refer to the same idea of dragon, which John and LOLITA conceptualize differently. This difference in conceptualization may be explicitly stated *"John has quite a different idea of hard work to me"*.

Such a view requires a different organization of the semantic net, where each word is linked to one or more ideas. This first linkage is where the usual lexical ambiguity occurs. Additionally, each idea is linked to different conceptualizations of it. This second link corresponds to the different conceptualizations of a concept different people may have. In this framework, at first a plausible assumption would be made that each new word a person utters to LOLITA corresponds to LOLITA's conceptualization of the concept. This could later be corrected. Maintaining this difference between different people's conceptions of ideas occurs, albeit rarely, in practical situations. It occurs when people try to understand statements which do not make sense literally, but which seem to involve some words being "misused" (with respect to the hearer's understanding of them) systematically. This may occur when talking to people not speaking their native language, when learning a new field which uses words in a different way, or when talking to people who are slightly insane or senile. Once the different conceptualization has been understood it can be classified as appropriate. For LOLITA the additional storage space involved in maintaining an extra layer of "ideas" is not justified given its current applications. But this part of the representation can fulfill the need, should it ever arise.

D.6.6 Definitional and observational sorts: a matter of expression ?

There is a good objection to definitional and observational sorts which has been postponed until now because its counter-argument requires understanding of intension and extension. It runs as follows:

For the sentence *"John and Mary's house is on the corner of the street"*, it is sufficient to know that the house is on the corner of the street for it to be uniquely determined. So the house is defined by being owned by John and Mary. Similarly, for the sentence *"The house on the corner of the street is John and Mary's"*, referring to the corner of the street is sufficient to determine the house uniquely. Thus the house is defined by being on the corner of the street.

But now, two sentences that mean the same thing are represented differently! The same house is either defined by being owned by John and Mary and is observed as being on the corner of the street, or vice versa, but the representation does not seem to capture the fact that either statement is sufficient to determine the house uniquely: if both statements refer to the house with definitional sorts, it would mean that the house is defined uniquely only by both statements together; if both statements refer to the house with observational sorts, the house is not uniquely defined at all. Surely then, referring to something in a unique way is a matter of expression rather than an intrinsic property of the house? Surely, in order to determine a statement by which one can refer to an object uniquely, one checks in the database whether any other objects also satisfy the statement: if none does, one has a statement which determines the object uniquely. This has nothing to do with the concept of house, but is a matter of expression.

The first problem here is one of confusion between intension and extension. LOLITA's representation is intensional, and intensionally *"John and Mary's house"* is a different concept from *"The house on the corner of the street"*. Extensionally however, in the real-world frame of existence, and during a particular range of time, the two concepts share the same extension. However, the house itself need not change if John and Mary sell it, nor is the particular house on the corner of the street necessarily the only house that John and Mary ever owned. This shows that a concept may refer to different extensions depending on the circumstances considered.

However, the argument is correct to the extent that the amount of information required to identify a concept uniquely in a conversation depends on the context of that conversation. For instance, in the context of John and Mary's current jobs, a reference to their house would usually be understood to refer to the house they own now, not the one they owned previously. In the semantic net, however, references must determine concepts uniquely with respect to the context formed by the whole semantic net. Arcs with definitional sorts must restrict concepts uniquely with respect to this maximal context. Thus part of the process of interpretation to transform natural language input into semantic net will involve reference resolution. In other words by maintaining some notion of the current context, references determined uniquely with respect to the context must be transformed so that they are determined uniquely with respect to the whole semantic net.

D.7 Textual References

In addition to the ability of representing the meaning of texts, SemNet must be able to represent physical texts as a sequence of words formatted into sentences, paragraphs, pages and so on. This is essential for many of the tasks LOLITA is to perform. Indeed, if LOLITA is to be used as an advanced information searching tool, she must remember in what texts what she read can be found.

D.7.1 Textual references: The problem

D.7.1.1 Text Structure

- **The need for a representation of text structure**

If a text is clearly structured, it is easier to read. Indeed, the way in which a text is organized can express an overview of its important ideas and the broad lines of its argument. LOLITA can profit from such information, if she has a representation suitable for representing it. This section develops this idea.

The structure of a clear text reflects the structure of the argument, and each of the ideas is presented separately. Later ideas build on previous ones, but avoid repetitions of earlier arguments, only referring to them if necessary. Thus, ideas are presented locally. This gives a context which simplifies much of the processing. For instance, the term "dogs" is more likely to refer to the opposing team in a football fanzine, but to animals in an article about pets. Exploiting such information reduces ambiguity.

Texts are structured into chapters, sections, paragraphs and sentences. These levels of structure correspond to different levels of granularity of the argument. To benefit from the locality of the ideas presented in a text, LOLITA must be able to recognise, model, and relate these different levels of structure.

Relating sentences in a text is required by analysis. For instance, the difference in interpretation between the sentences *"He went to a restaurant. He went to Mac Donald's"* and *"He went to a restaurant and he went to Mac Donald's"* stems from a difference in structure. Similarly, only structure distinguishes *"John left. Bill had insulted him"* and *"John left and Bill had insulted him"*, yet the former is well formed, and the latter is not. Many other meta-level forms of analysis must be applied to the text to improve interpretation: stylistic, discourse or dialogue analyses require information about the structure of the text.

Information about the structure of the text is also required by many applications. Translation, for instance, requires the translated text to appear in a similar order to the original. Text retrieval systems, such as those employed by the rapid reaction units of political parties, need the ability to return precise references to incriminating quotes by opposition members, so that the relevant text can quickly be located. The MUC competition required coreference to be expressed by adding the references LOLITA determined to the original text next to the relevant words.

Table D.1: References using structure

Types of logical Structure	References
Titled Chapters/Sections/Articles	Table of Contents
Keyworded sections	Reference: Glossary, Dictionary...
Numbered sections	Grammar Book, Butterfly Guide...
Types of physical Structure	**References**
Page number	Index
Page number and word reference	Hypertext, MUC annotation, SGML...
Book volume/Tablet/Scroll/Wall...	In Index, Catalog...

- **Models of text structure**

Two main kinds of text structure can be distinguished. Logical structure is the division of the text by content. Physical structure is the division of the medium in which the text is expressed.

Logical structure depends on the type of information expressed. Newspapers and magazines, for instance, are divided into articles. Books are often divided into chapters, and further levels of sectioning. Most texts are divided into paragraphs, sentences and words. Each level of sectioning can be associated with a title, or word.

Physical structure is determined by the text's medium and the text's nature. Encyclopedias are usually divided into volumes. Books, newspapers, and magazines are divided into pages. Pages on computers, in particularly the world wide web, are divided into separate files, and file systems. Older sources of information include tablets, scrolls and walls. [35]

The structure of texts always has physical and logical components. Sometimes the two converge, as in the case of different volumes of a novel. In other cases they do not, as for pages.

- **References to text structure**

Most references within a text make use of the structure of the text. References

[35] The notion of physical structure can be the cause of some confusion: a textref corresponding to a page does not represent the physical page which can be burnt. It represents the structure given by the page to the text. Each copy of a book has the same structure, but different physical pages, taken from different trees.

often use both logical and physical structure to achieve a higher ease of use. For instance, a quote in a Newspaper can be located both by article title and page number. Table D.1 *(p. D-102)* gives a summary of common indexing methods, and the text structure they use.

- **A general model of text structure**

◦ **Salient features of text structure**

A representation of the text's structure should be capable of representing a wide variety of text structures, while requiring the introduction of as few new forms of reasoning as possible. This section considers salient features of text structure and concludes that text structure is very similar to another phenomenon already represented.

Text is structured into segments of text following each other. This segmentation is such that within a particular model of text structure, there is no intersection between segments. For instance, paragraphs do not include other paragraphs, or chapters other chapters. Pages may share paragraphs, but pages are concepts of the physical model of a text, whereas paragraphs are concepts of the logical model.

Within a particular model, there are different levels of segmentation. A section contains paragraphs containing sentences containing words... Representing the structure of the text corresponds to relating these different levels of segmentation. There are two components to the structure: the order of the segments as they appear in the text, and whether one segment is within another.

The elements of a text's structure are fully fledged concepts. For instance single words can be referred to independently of their meaning. An example is *"Every second word in that document is incorrectly spelt !"*. Similarly the ordering of segments can be discussed as in *"I think you should put that paragraph in section two"*. This means they should be represented explicitly by concept nodes in SemNet, so that any statement can be made about them.

The hierarchical nature of text structure, the use of ordered segments, and the conceptual status of segments make text structure very similar to time. This means that a similar representation can be used for both.

∘ Comparison of time and text structure

Time was modeled in terms of relations between intervals and duration. Text structure can similarly be modeled by relations between segments and segment length.

◇ Intervals and Segments

Just like segments, an interval can be contained in another, shared by a common subinterval with another, started by another. Similarly, intervals and segments share a notion of duration or length.

◇ Duration and length

The notion of segment length depends on the model of structure used.

For logical models of structure, the length of a segment can be modeled in terms of the segments it contains. In other words, valid units can be chapters, paragraphs, sentences, words or letters. A paragraph can have as length one paragraph, three sentences, twenty five words, or one hundred and fifty letters. It could be measured as quarter of a chapter, but this is not very accurate: chapters are fuzzy units, possibly varying from a few pages to few hundred pages. Notions such as half a chapter are usually expressed by stating that the paragraph has half the number of words, or the half the number of pages the chapter has. A zero length segment in this model is zero sentences, zero words or no letters long. The space between two words consists of no letters, no words. Thus, a zero length segment will be taken to be the space between two words.

For physical models of structure, the length of a segment can be modeled in terms of segments it contains. Valid units are book volumes, or pages. Similar observations hold for these units as the physical models. Zero length segments in the physical model correspond to the space between two pages, two books.

Zero length segments are defined both for physical and logical models of text structure as parts of the medium or text not actively expressing information. The availability of a zero length segment concept is essential to the similarity between the representation of time and text structure: The notion of instant is the key to providing a notion of order with `starts_` and `ends_` events.

◇ **Conclusion**

Time and Text structure are remarkably similar, and can be represented by the same basic representation. This is useful, since it means that the same reasoning processes can be applied to both.

D.7.1.2 Statements involving the text's representation

Texts have a dual nature, being physically represented as symbols and conveying meaning by being interpreted. Because SemNet focuses on concepts, most of the discussion in this thesis has focussed on the meaning aspect of texts. Statements can however also be made about representational elements of the text, such as words, or paragraphs. Examples of this were briefly discussed in D.7.1.1 *(p. D-103)*.

References to the text can address details of the representation: *"That was a short paragraph"* or *"That word is incorrectly spelt"*.

They can also specify the author of the text: *"Did you write these words?"* or *"It was Jack who uttered the rallying cry"*. Allowing reference to the words, rather than their interpretation is important as the interpretation may either be flawed or impossible: *"Why did John shout 'Donkeys!' ?"*; *"I do not understand what you mean by 'The world economy is based on the purveyance of commodities"'*; or *"I am asking you what the words of the accused were, not what you think he said to you."*.

Finally the text can be believed in, because the agent trusts that whoever said it knew what he was talking about. This type of behaviour occurs in some religions, where one must have faith, even if one does not understand what is meant. It also may occur in Science or Engineering, where people may believe statements such as *"This gives a jamming margin of 5dB and any signal greater than this will jam a dsss signal"* because they read it in a book.

These events are distinct from their relatives which refer to interpretations. Thus, for instance, there must be (at least) two meanings of "say", one of which refers to the actual words that were said, and the other to their interpretation. The distinction can be enforced using type restrictions. This might seem to be a break of uniqueness, if, say, a link between concepts and the text that expresses them were made. It is not however, since there is not a one to one correspondence between language and concepts. For instance there are more than one ways of expressing the concept "to marry": *"I wed her"*, *"I married her"*, *"I took her to be my wife"*...

D.7.1.3 Linking Concepts to Text Structure

Since text conveys meaning, references to it usually also refer to its meaning. [36] For LOLITA to be aware of this connection, the representation of a text's structure must be linked to its interpretation. This section discusses the need for such a connection in more depth, and the nature of the link.

- **The need for links between text structure and concepts**

Many of the reasons for a link between a text's structure and its interpretation follow from the discussion of the need of a text structure representation (D.7.1.1 *(p. D-100)*). However the case for expressing the link within SemNet has not been put.

For a piece of information to be expressed within SemNet, there must be some concepts in terms of which it can be expressed. Just as the structure of a text can be discussed (*"The first section is rather long"*, or *"What does 'corroborate' mean ?"*), so can the link between text structure and interpretation: *"The sentence about the dam in China is too controversial for the Prime Minister's visit"*, or *"John thinks that 'Blessed are the poor in spirit: for theirs is the kingdom of heaven' means that mentally handicapped people will go to heaven after they die. But Jack thinks that it means that people who are not attached to objects or people are in a state*

[36]As discussed in A.3.2 *(p. A-32)*, the "meaning" of a text is a loose way of referring to an interpretation of it.

of mental peace and contentment". Since the link between text structure can be a subject of belief, it must be represented as an event, ie a concept.

The result of expressing the link in SemNet is that it can be permanent. Indeed, if information about text structure and its links to concepts resulting from interpretation were considered of interest only to the interpretation process, it might be stored temporarily and locally. It would be discarded once part or all of the text had been interpreted. While this would be sufficient for many of the concerns of interpretation, it would bar LOLITA from being able to reconsider the meaning of words.

Expressing the link between structure and interpretation within SemNet simplifies processing for many algorithms, among which algorithms that have only a passing interest in the rest of the text. For instance, eventually LOLITA should be able to judge her interpretation of texts, and determine whether she believes she has misunderstood a word. Assume that she had encountered the word "phreak", and on the basis of phonetic information had decided that this was a misspelling of "freak". Since the word "phreak" is rare, she may not be able to deduce much from the context of the text in which she first encountered the word. All she will note, is that the word "freak" seems not to fit well into the context. However, later she may encounter more examples of it, and revise her previous assumption: "phreak" is a new word related to telephony. This type of algorithm will only be interested in the contexts in which the word is encountered, and perhaps the structure of the texts in which it occurred, but not the text itself.

- **Types of links**

Text structure and its interpretation can be linked in many ways. This section discusses the types of these links.

∘ **Words used**

The simplest of links between a concept and a phrase states that the concept was derived from the phrase. It is represented by a `words_used` event. As an event, it is subject to belief. Believing a words used event does not imply that one believes in the concept it links to the text, but only that one believes that the concept was expressed by the words of the text. Thus it can be used for statements such as *"I think the words he used were 'I hate the army"'* in the context *"He said he was unhappy at the base." "What were the exact words he said to you?" "I think ...".*

∘ **Phrase means**

The other important link between a text and concepts is interpretation. For instance, SemNet must be able to represent sentences such as *"I understood him to have said that three hundred year old institutions such as Barings have long traditions of pristine controls over their employees."* This task is fulfilled by the `phrase_means` event.

More formally, a `phrase_means` event states that if the phrase it is connected to is believed, then the concepts to which it is attached are also believed. Thus belief in a `phrase_means` event alone does not imply belief in the interpretation. For instance, an agent could believe that the statement "The Queen is dead" means that Elizabeth II, Queen of England is dead, yet need not believe that the Queen is dead. However, if the agent believes in the statement, a priori to understanding it, he must also believe in its meaning if he believes that his interpretation of the statement is correct. For instance, one might believe, yet not understand, the statement *"In an impurity semiconductor containing donors, the Fermi energy lies above the middle of the forbidden band"*, because the Nobel Prize in Physics uttered it. If one also believes that one has the correct interpretation for it – ie that one has understood it, then one must believe in that interpretation.

• **Nature of the link**

The fact that one concept can be represented by many words, and one word may be represented by many concepts determines the structural component of the link's representation.

An example of one word corresponding to many concepts is *"amabo"*, the latin for *"I will love"* , which refers not only to me but also to a time after now. An example of words corresponding to one concept is given by *"to fly"* which is the action of the flying event. Further, many words may refer to many concepts. For instance, "a dog" is represented by an `inst_` event and a new child of `dog`, the concept dogs.[37] Finally, SemNet represents phrases or sentences by events. Thus, the phrase "a man bit a dog" is expressed by a single event. These observations require a piece of text to be linked to many concepts.

A text can be linked to many concepts through multiple links or via a central event. Various reasons conspire to prefer the second alternative.

A unique event is clearly more economical in net space than many, and is simpler to connect meta events to, such as belief events.

Sentences express more meanings than the individual words that constitute them. For instance, order is an important factor in interpretation: *"John killed Bill"* is not equivalent to *"Bill killed John.* Each word of a phrase could be individually linked to the concept E_0 expressed by the phrase: *"The car was burning"*. However, if a pronoun were used to refer to the phrase (*"It was most upsetting to see"*), the pronoun would also be attached to the concept E_0. As a result there would be no distinction between the groups of words that express or refer to the concept. The words must therefore be grouped together into individual phrases that refer to the same concept(s). There are only two alternative ways available of performing the grouping. Either one uses text structure to assemble the phrase into a text segment that can be referred to by multiple links, or one must use a unique linking event. The first alternative fails to represent single phrases in sentences where the phrases are intertwined: in *"A man, John said, bit a dog"*, the words "A", "man", "bit", "a" and "dog" do not form a text segment, so could not be associated with the corresponding event by the first method. This leaves only the second method.

The link between concepts and text is represented by a unique event with possibly many partial `subject_s` and `object_s`.

[37] It could be represented by a reference to the concept dogs with an arbitrary quantification, but that would prevent it from being referred to by another event, even if the following text were to refer to that concept.

D.7.2 Textrefs: A Representation of Text Structure

The representation of text structure shares many features with that of time. The discussion of these will lead to a new general representation, similar to that of values: a timelike representation. This is followed by a discussion of features specific to textrefs.

D.7.2.1 Shared aspects with time

- **A mirrored representation**

The representation of text structure mirrors the representation of time very closely. There are therefore textref variants of `starts_`, `ends_`, `is_in`, and `follows_`: `T_Starts`, `T_ends`, `T_is_in` and `T_follows`. Similarly, a specialization of `has_value` mirrors the effect of `has_duration`: `T_segment_length`. Just as `follows_` could be qualified by `duration_`, so `T_follows` can be by `T_segment_length`.

Structure is expressed relative to the text in which it occurs. The beginning of the text serves as origin. This mirrors calendars for time for which dates are expressed as a duration since an origin. For instance:

(E_0,R): {ΔI-`subject_`-$I\Delta$: page4; ΔI-`action_`-IO: `T_follows`;
ΔI-`object_`-IO: `sengan_thesis_start`}

(E_1,R): {ΔI-`subject_`-$I\Delta$: E_0; ΔI-`action_`-IO: `T_segment_length`;
ΔI-`object_`-IO: 4_pages}

(E_2,R): {ΔI-`subject_`-IO: 4; ΔI-`subject_`-IO: 1_page;
ΔI-`action_`-IO: $\odot$; ΔI-`object_`-$I\Delta$: 4_pages}

(E_3,R): {ΔI-`subject_`-$I\Delta$: page4; ΔI-`action_`-IO: `T_segment_length`;
ΔI-`object_`-IO: 1_page}

defines the fourth page of this thesis.

Just as for calendars, a model of the text must be built in SemNet for it to know that the forth page follows the third and precedes the fifth: all the arguments about the fuzzy nature of units still apply. See D.4.3.4 *(p. D-62)* for more details. Two models can be created for the same text, one corresponding to the logical and one to the physical structure of the text.

• **Defining a new class of reasoning**

The use of the same type of reasoning for time and text structures, and the mirrored representation, means that a new class of reasoning should be defined. This "timelike" representation has the same generic status as the values representation and is defined in a similar manner. All events used for the time and text structure representation are specializations of their timelike equivalents. The composite events such as `follows_` or `T_follows` are defined at this abstract timelike level. The difference between the time and text structure specializations is defined by the type differences of the arguments of the template events. This ensures that times and textrefs are not related through timelike events. Reasoning algorithms that previously dealt with time events should now deal with time like events.

The result is that the abilities previously reserved for reasoning about time are now available for text structure. Thus, constraints on the position of some information can be resolved to determine where it is, in the same way as constraints on a time can be resolved. This is used for instance by *"The sentence you are looking for is on an even page, in chapter 2 or 3, and at the start of a paragraph."* Further, the analysis rules will benefit from similar sentences being represented in the same way: *"The forth page of the book"* – *"The forth day of the conference"*, *"The chapter after mine"* – *"The day after my birthday."*, or *"In the first chapter, John says that..."* – *"In the fifth year of the plan, Russia..."*

D.7.2.2 Differences to time

• **Labelling Textrefs**

∘ **Why label Textrefs ?**

Although it is sufficient to specify the location of a word to determine the word referred to, it proves useful to identify the textref by the word to which it corresponds. The first method requires retaining a copy of the document. This requires external access to the document which renders any processing referring to the words represented by the textrefs unduly complex. It also renders the textref information useless if the document were lost.

Expressing the word the textref represents within SemNet on the other hand, simplifies processing for many algorithms, among which algorithms that have only a passing interest in the rest of the text. For instance, the algorithm discussed in D.7.1.3 *(p. D-107)* is not interested in the text itself, but only the contexts in which the word is encountered, and the word's spelling. This is necessary to determine in which contexts LOLITA interpreted "phreak" as "freak", which is a prerequisite to allowing her to revise her interpretation.

∘ **Label Textrefs, how ?**

Since there is a need to refer to the words of a text, the next question to be resolved is how this can be achieved.

The standard method of qualifying a concept is to connect events to it which express the information one wishes to qualify it by. In this case, a word is represented by a string of characters. There are therefore two components to each letter: the particular character used, and its position within the word. Each would be represented by an event, leading to a mass of events to express even four letter words. Unless the independence between the types of the information expressed by separate events is actually needed for processing, this is an expensive solution. In fact, this independence proves more of a hindrance to potential applications. Take the example of the algorithm revising assumptions of misspellings (D.7.1.3 *(p. D-107)*): it only wishes to refer to full words rather than individual letters. Having to access each letter and place it in the relevant slot is inefficient and cumbersome, unless justified by the needs of some other type of processing.

An alternative, albeit non standard method is available. In the same way as linguistic nodes do not use events to represent their letters, textrefs can use the label associated with each node: see 5.1.1.4 *(p. 117)*. To avoid confusing programmers, linguistic nodes and textrefs are distinguished. The words that textrefs represent are surrounded by double quotes, whereas the corresponding linguistic nodes lack quotes, and the conceptual nodes lack labels. Thus `"cat"` is the textref for *"cat"*.

This method has the advantage of simplifying not only the expression but also the access to the words corresponding to the textrefs. As explained in D.7.3.2 *(p. D-116)*, this alternative method is not restrictive, since if the structure of the characters in a word were needed, it can be expressed.

- **Further qualifying textrefs**

Just as textrefs can be labelled, they can be qualified by specific events. This was discussed in D.7.1.2 *(p. D-105)*, involving examples such as *"Did you write these words?"*. Other specific examples are titles being associated with sections, labels of texts such as ISBN numbers, or associating the physical page or volume with the structural page.

Since the title of a text section is itself a section of text, expressing the title of a section is expressed as a relation between textrefs:

(E_0,R): $\{\Delta I$-`subject_`-IO: `title_x`; ΔI-`action_`-IO: `entitles`;
ΔI-`object_`-$I\Delta$: `sengan_thesis`$\}$

A complete model of a text would therefore include a representation of textrefs, text content, and physical objects with which texts are represented.

- **Place in the hierarchy**

Textrefs are a specialization of the timelike concepts, defined by being `subject_s` of `words_used` events. Since textrefs reflect the structure of the text, each segment is an instance of the set of textrefs. Thus if the word "cat" appeared three times in a document, there would be three textrefs `"cat"` – one for each occurrence of the word *"cat"*. This means that SemNet will include a large number of textrefs. [38]

All textrefs could be instances of the set of all textrefs, without any internal structure. However, this would result in very low topological determinism and would badly affect search. For instance, consider again the algorithm revising assumptions of misspellings (D.7.1.3 *(p. D-107)*). In order to determine the existence of a dis-

[38]For a while at least. Clearly, if LOLITA is to deal with any substantial quantity of text, she will need algorithms to eliminate less useful information. A prime target would be the common words expressed by textrefs, unless they had special significance. For instance, determiners such as "the" or "a" would usually be eliminated, unless in a sentence such as "You are only a queen. We can always replace you!"

tinct meaning for the word "phreak", the algorithm needs to access all occurrences of the word and the associated interpretation of it, to derive a context. A similar algorithm might determine that a word known to LOLITA had another meaning. It too would face the same need to access all concepts attached to the word. A way of structuring the textrefs to improve topological determinism is therefore needed.

The problem can be reduced by grouping textrefs of same word type. This is achieved by having a set of textrefs for each word type. In the "cat" example, the three individual `"cat"` textrefs would be the children of a generic `"cat"` textref set. Thus if an instance of a word is known, all other instances of it can be found simply by searching for the highest parent with the required name, and finding all its children. For instance, the word "Hamlet" may not have been known as a proper name to LOLITA. So she may have interpreted it as a tiny village in some sentences, or as a small ham in others. Only by accessing all occurrences of the word, may she be able to determine that there is a third meaning. Grouping words in this way also improves the efficiency of stylistic analysis which may wish to determine whether words rythme (for poetry) or whether a sentence is repeated for effect.

The hierarchy can be further structured to improve the topological determinism from the superset of all textrefs to the set of textrefs corresponding to any particular word type.

D.7.3 The use of textrefs

Textrefs prove useful in a wide variety of circumstances. This section reviews some of their uses.

D.7.3.1 Interpretations

The `phrase_means` event states that the concepts it links to textrefs are an interpretation of the text they represent. Since interpretation is of great relevance to LOLITA, some examples of the use of `phrase_means` are presented.

`phrase_means` can be used to present different people's interpretations of a same text: John might believe that Mary saying "I like you" means she loves him, but is shy... whereas Mary only meant that she liked him.

Because LOLITA's NL analysis phase produces an interpretation of the source text, the interpretation is connected by a `phrase_means` event to the relevant textrefs. Strictly speaking, the result of the interpretation is a set of `phrase_means` events connected to concepts. The strength of various aspects of the interpretation are expressed by belief values in the relevant `phrase_means` events. Any belief in the conceptual structure which is the object of the `phrase_means` events is dependent on belief in the attached `phrase_means` event. This means that the concepts forming the interpretation of the text are all accorded hypothetical status until LOLITA decides that she believes them. This decision is based on many (possibly interlinked) factors such as whether they contradict her previous knowledge, whether the interpretation rules accord the `phrase_means` events a sufficiently high degree of belief, and whether the source of the information is reliable.

The advantage of marking all concepts resulting from analysis as hypothetical is that the information derived from them cannot interfere with the reasoning required by other phases of their analysis. For instance, a contradiction in SemNet would occur if LOLITA believed that *"Bill Clinton is the president of the United States"*, and the text to analyse was *"Bill Clark is the president of the United States"*. Determining whether LOLITA is to believe that the speaker is mistaken, that he is fantasizing, or that he is talking about a different United States or a different time, requires reasoning which will fail if SemNet contains contradictions. Using `phrase_means` allows all information to be entered into SemNet free of such interference.

A problem arises however as to what should be done with statements that are hypothetical in NL: *"If I were king, I would abolish property"*. Both the events corresponding to "I were king" and "I would abolish property" are hypothetical, but the event expressing the "If" relation is not. Does forcing all concepts to hypothetical not destroy this distinction? Surprisingly it does not, since only the event expressing the "If" will be connected to a believed in textref by a `phrase_means` event. This illustrates the important point that believing in a textref does not imply believing in segments occurring within it.

Once a text's interpretation, and the text itself is believed, the concepts forming the text's interpretation are believed in too. Initially their belief value will be a combination of the amount the interpretation is believed, and the amount the text's source is believed, but later other information may play a part.

D.7.3.2 Keeping track of the words forming LOLITA's facts

The `words_used` event is used to keep track of the words from which a fact was derived. Its main use is therefore in the development cycle as a diagnostic tool for possible errors in the interpretation phase.

It proves sometimes useful to refer to parts of words. For instance, the ending "n't" in English corresponds to "not". For it to be linked to the resulting negation in the conceptual layer of SemNet, part of a word must be referred to[39]. In the example *"I didn't kill him!"*, part of the word "didn't" must be referenced. This is possible because of the general nature of the representation of text structure. A word too is a segment of text, which includes smaller text segments, such as syllables... Other possible such linkages between word morphology and concepts include verb conjugation, or the declension of nouns and determiners.

Other languages also benefit from the ability to refer to parts of words. For instance Turkish uses prefixes and postfixes for constructions which would use separate words in English. Chinese fits on the other end of the spectrum with no morphological variation.

D.7.3.3 Coreference tasks

`words_used` was used for coreferencing applications, such as the coreference task

[39]The alternative scheme of preprocessing all occurrences of "n't" into "not" solves this problem, but deprives later stages such as style analysis of useful information.

of the MUC 5 competition. The task is to annotate source texts with pronouns and other forms of anaphoric reference. This was achieved by using the fact that pronouns and the other forms of anaphoric reference are connected by `words_used` events to their reference. For instance, *"Humpty Dumpty was a poor student. Because of government cuts, he climbed up a wall to steal an apple. But he slipped, and cost the NHS instead".* `words_used` events connect the words "Humpty Dumpty", and "He" to the concept `Humpty_Dumpty`.

The result proves useful to keep track of LOLITA's development, and therefore serves as a development tool.

D.7.3.4 Information Retrieval

`words_used` and `phrase_means` events will form the basis of any information retrieval system based on LOLITA. Any search conducted by LOLITA will occur at the conceptual level. Most information retrieval can be modeled in terms of queries. Once the concepts satisfying the relevant queries have been found, the texts in which the they occurred must be determined. Textrefs are used at this point to give a clear reference.

A current LOLITA project, spearheaded by Drs M.Fox and D.Long, is to use LOLITA as an advanced Internet searcher. The idea is to augment the standard pattern matching methods used by current search engines, by using LOLITA's NLP analysis to reject irrelevant texts. Textrefs can express web page addresses in the same way as they allow the letters of a word to be expressed, such as: `"http://www.dur.ac.uk/~lolita"`. Textrefs can also express the structure within hypertext, capturing the notion that referring pages may provide a context to a page, in a similar manner to titles. For instance a list of software titles on a page referred to as *"pirated products"* has a different meaning to the same page with title *"Soandso's products"*.

D.7.4 Textrefs: Conclusion

Textrefs provide a general solution to the problem of representing the text's structure by exploiting the similarity between time and textual structure. This allows all the standard reasoning algorithms used for time to be used for textual structure.

D.8 Ambiguity

Natural language texts often demonstrate a considerable degree of ambiguity. Even simple sentences such as *"A bishop blesses every novice near a river"* prove highly ambiguous. The ambiguity varies from differing meanings of words to scopal or attachment ambiguity. Inference is needed to perform disambiguation. Since inference is usually performed on the net it is advantageous to use a representation where ambiguous information is freely accessible in the net to avoid replicating the inference algorithms for use by the disambiguation process. This need becomes more urgent if a lazy disambiguation technique is used, whereby some of the disambiguation of a sentence may be performed a few sentences later when further relevant information has been determined. Thus a first requirement for the ambiguity representation is that it should allow processing of the information in the net to be performed by the same routines as those dealing with unambiguous statements. As a consequence of this, the lesser the difference between the representation of ambiguous and unambiguous statements, the better.

D.8.1 The type of ambiguity to represent

D.8.1.1 Ambiguity in Natural Language

The ambiguity encountered in Natural Language texts consists of sets of alternatives. Let us consider a simple sentence: *"I saw the lady with the hat"*. Such a simple sentence has a considerable degree of ambiguity. Firstly, the words "saw", "lady", "with" and "hat" all correspond to more than one meaning. This means that each is associated with a set of alternatives. Secondly, the prepositional phrase *"with the hat"* could either be attached to the seeing event (where "with" could mean using) or to the lady herself (where "with" could mean accompanied by, wearing...): another alternative. Finally, more than one lady may fit the description, leading to a further set of alternatives, once a referent is to be chosen.

D.8.1.2 A simple attempt: Global ambiguity

The simplest method of dealing with ambiguity in the required way would be to create all the possible interpretations of the sentence. If each of these were expressed in its own copy of the semantic net, the knowledge of each such semantic net would completely unambiguous and reasoning could easily be performed in each.

Further processing of the text must occur in all these nets. Thus adding new information, and determining impossible or unlikely interpretations must be repeated for each copy of the net. Since there is an interpretation for each combination of alternatives, this scheme requires a combinatorial number of copies of the net. This in turn imposes a combinatorial processing load, since each copy of the network must be processed independently.

For instance, consider only ambiguity due to ambiguous words in the sentence *"The shortest path between two points is a straight line"*. The syntactic analysis assigned to this sentence restricts the number of meanings to: 11 [40] of 17 to "shortest", 4 of 4 to "path", 1 of 2 to "between", 16 of 23 to "points", 5 of 11 to "is", 13 of 14 to "straight", and 25 of 31 to "line". This represents a total of $11 * 4 * 16 * 5 * 13 * 25 = 1144000$ interpretations! This should be further increased to take account of other sources of ambiguity such as attachment decisions, and quantifier scoping. Clearly, a naive approach to ambiguity is not practical for large scale systems where ambiguity may not be resolved until further sentences have been processed. Obtaining a representation of ambiguity that allows efficient processing is thus essential.

[40]These numbers are taken from LOLITA's current knowledge base. However the numbers depend on the distinctions made by the chosen ontology.

D.8.1.3 Sharing and alternative arcs: Local ambiguity

The failure of the simplest method of dealing with ambiguity is that it replaces local ambiguities by global ones: to avoid representing the various meanings of a word, it replicates the net. Each such copy is distinguished only by a few arcs. Instead, if the common structures were shared between possible interpretations, all processing involving them would only need to be performed once. Therefore, it is computationally efficient to maximise sharing as long as this in itself does not incur a combinatorial or worse cost.

By comparing the structures in the various semantic nets discussed above for the different interpretations of a sentence, one observes that ambiguity resides purely in the attachment of arcs. For instance, consider the sentence *"I bought a small dog"*. Here the words "buy", "small" and "dog" are all ambiguous. By implementing a scheme where arcs may correspond to one of many alternatives, one obtains a structure with a high degree of sharing. After syntactic analysis, 4, 10 and 5 meanings remain for the concepts buy, small and dog respectively:

(E_0,R): $\{\Delta I$-`subject_`-$I\Delta$: `Sengan`;

ΔI-`action_`-IO: $\texttt{buy}_1$/ ΔI-`action_`-IO: $\texttt{buy}_2$/

ΔI-`action_`-IO: $\texttt{buy}_3$/ ΔI-`action_`-IO: $\texttt{buy}_4$;

ΔI-`object_`-IO: $\mathcal{SD}_{00}$/ ΔI-`object_`-IO: $\mathcal{SD}_{01}$/

$\cdots$/ ΔI-`object_`-IO: $\mathcal{SD}_{49}\}$

where $\mathcal{SD}_{nm}$ is the subset of the nth meaning of small and the mth meaning of dog, so there are implicitly 100 additional `inst_` events in the example above. However, this "alternative arc" solution has two major deficiencies. Firstly it requires the programmer to write additional routines to deal with the ambiguous arcs thereby violating the first requirement. This was to ensure that the standard net processing algorithms will work. Secondly, and more seriously, it does not allow information useful for disambiguation to be expressed.

The failure of the alternative arc scheme resides in that it attempts to increase sharing by enforcing too strict a degree of locality onto ambiguity. To see this, consider the sentence *"John went to the cinema the day before I met him"*. If I know more than one John, it may not be obvious to me which is being referred to. Thus there is ambiguity of the noun "John". In the alternative arc scheme, the `subject_` of the "going" event and the `object_` of the "meeting" event would both be ambiguous arcs having as alternative targets the various possible Johns. Since each alternative arc is independent from all others, it would be possible for the "going" event to involve a different John from the "meeting" event. Although the use of the pronoun "him" clearly indicates that the same "John" is involved, there is no means of expressing this information which transcends the locality of the alternative arc.

D.8.1.4 Reflecting the locality of ambiguity in the representation

Both of the previous schemes failed either in their computational complexity, or in their ability to represent all the required forms of ambiguity. They also both fix the level of locality of the ambiguity they can represent. However the types of ambiguity in NL vary from very local ambiguity, such as different word meanings, to global ambiguity, such as attachment decisions or anaphoric references spanning many sentences. By representing ambiguity flexibly at level where it occurs, sharing can be maximised without limiting the ambiguity that can be expressed.

D.8.2 The nature of ambiguity modelled

The ambiguity considered stems from the nature of the analysis phase: it is an interpretive phase. Thus its task is to take natural language input, and derive from it a meaning that is coherent with its model of the world. Coherent here means that the meaning derived is possible with respect to the facts expressed in its model of the world, and that it does not require a large change to that model to fit in consistently. The notion of "Large change" encompasses both quantitative and qualitative notions: important facts on which a lot depends cannot be revised easily. This derivation process is aided by restrictions on the meanings that can be derived: these are the linguistic rules that LOLITA is given. Note that it is neither assumed that LOLITA's conversant has the same linguistic rules, nor has the same concepts as LOLITA. This means that neither a universal form of meaning is assumed, nor language is considered as a compressed form of such meaning using standard rules. Rather all that is assumed is a sufficient degree of commonality for statement made by one agent to be understandable most of the time to the other. This is an obvious necessity, since speaking a language is an effort which must be rewarded by some practical gain.

Within this framework, it is the notion of interpretation that is fundamental. Ambiguity, seen from this viewpoint, corresponds to a set of possible alternative interpretations that have been derived from the surface utterance at a particular point in the processing. This set will be reduced by disambiguation, such as provided by discourse rules, coherence checking, and so on. Other rules increase it by mapping a surface form to many alternative meanings.

The question therefore becomes what does LOLITA believe the surface form to mean? What meaning does LOLITA assign to the utterance? As discussed in the epistemological section, meaning is fundamentally a belief about the world which will determine further actions within it. Thus the question can be reformulated as: if LOLITA believed the surface form of the statement, what belief does this imply LOLITA holds about the world? In general this applies to any agent, so the key notion is that of "what world belief does belief in the surface form of a particular statement imply?"

D.8.3 A general representation of ambiguity

Analysis of ambiguity led to two conclusions relevant to the design of a representation of ambiguity:

The ambiguity to be modelled results from the interpretation process. It corresponds to different interpretations that LOLITA can derive from the text. Disambiguation occurs when LOLITA decides she believes in one interpretation more than another.

The contradictory demands of representing all forms of ambiguity, yet maximising sharing can be satisfied by a flexible representation which captures the locality of the ambiguity.

This section develops a representation that satisfies these twin demands.

D.8.3.1 Representing interpretation

The fundamental notion that the representation must capture is that the text is interpreted. For this, a representation is needed for the text itself, for the corresponding concepts, and a mapping between them. But such a representation has already been introduced! Textrefs provided a representation of the text itself: its words, and its format. The `words_used` and `phrase_means` events provided a mapping from the concrete text to LOLITA's interpretation of it. Thus this part of the work has already been done.

It is however useful at this point to clarify the exact meaning of the `words_used` and `phrase_means` events.

`words_used` events simply state that a particular conceptual structure was derived from the words corresponding to the textrefs that are the `words_used`'s `subject_s`. Belief in a `words_used` event does not state any belief in its `object_` concept.

A `phrase_means` event is usually used to state that LOLITA believes that the phrase expressed by the event's `subject_` textref(s) means the conceptual event(s) that are its `object_`(s). More formally, a `phrase_means` event states that if the phrase expressed by the event's `subject_` textrefs is believed, then so must be the events that are its `object_`(s). Thus belief in a `phrase_means` event alone does not imply belief in the interpretation. For instance, LOLITA can believe that the statement *"The Queen is dead"* means that Elizabeth II, queen of England is dead, but she need not believe it. However, if LOLITA believes in the statement, a priori to understanding it, she must also believe in its meaning if she believes that her interpretation of the statement is correct. For instance, LOLITA might believe, yet not understand, the statement *"In an impurity semiconductor containing donors, the Fermi energy lies above the middle of the forbidden band"*, because the Nobel Prize in Physics uttered it. If she also believes that she has the correct interpretation for it – i.e. that she has understood it, then she must believe in that interpretation.

As was explained in D.7 *(p. D-100)*, all events added to SemNet during interpretation are added as hypothetical[41], and the `object_` of a `phrase_means` event (or equivalently, of an `and_` event, itself the `object_` of a `phrase_means` event).

D.8.3.2 Variable nodes

- **Variable nodes: Motivation**

Ambiguity can already be represented by the `words_used` events. Indeed, suppose that LOLITA is unsure to which of two meanings a word corresponds. This would mean that there are two possible interpretations of the word, one of which she believes is correct. This can directly be mapped into the representation by connecting the word to the two concepts with two hypothetical `words_used` events, themselves the subjects of a `xor_` event. Whether the `xor_` event is believed depends on whether the interpretation it represents is believed: if it were not for the interpretation, there would not even be `words_used` events to choose from. Thus, the `xor_` event is one of the `subject_s` of an `and_` event (representing one full interpretation), itself the `object_` of a `phrase_means` event.

The problem with this scheme is that it does not allow sharing of common conceptual structures. The alternative `words_used` events split the word into separate meanings at the conceptual level. This means that common structures cannot be shared at the conceptual level, precisely contradicting what is wanted. For instance the representation of a sentence with only two ambiguous words, *"Ross was escorted from bar to the dock"*[42] would require four similar structures: one for each of the four combinations of meanings.

Variable nodes provide a solution to this problem, by delaying the splitting of a word or sentence into alternatives until the conceptual level is reached.

[41]except events corresponding to hypothetical statements in the text: see D.7 *(p. D-100)*.

[42]This might either invoke a courtroom or a harbour scene (taken from [Hirst 87])

- **The representation of Variable Nodes**

Variable nodes represent nodes which correspond to one of a set of alternative concepts. Their alternatives are expressed using hypothetical `synonym_` events, all of which are the `subject_` of a `xor_` event, itself dependent on the relevant `phrase_means` event. Thus they are concepts defined as being synonymous with one of their alternatives. They provide a unique reference point to which the events from the rest of the text can be connected. The `synonym_` events are definitional with respect to the variable node, but observational with respect to the alternative concepts.

The variable node may be the `object_` of the relevant `words_used` event, but need not be. For instance, if the phrase "a cat" is processed, the `words_used` event would be connected to an instance of the variable node, rather than the variable node itself (a set). The instance event would however be connected to the same `phrase_means` event as the `xor_` event via an `and_` event.

Prepositions are often ambiguous. For instance, "with" may refer to accompaniment as in *"I went to the cinema with John"*, instrument as in *"I opened the bottle with a corkscrew"*, wearing as in *"The lady with the red hat"*, and many other forms of association *"The soldier with the sister"*. It is possible to express this type of information as ambiguity of the `action_` of an event.[43]

Variable nodes are also used in the treatment of anaphoric reference. Thus, one can state that there is only one John involved in the sentence *"John was sacked the day before I met him"*, although it may not be clear to which of the Johns it knows, the agent is referring:

(E_0,H): $\{\Delta I$-`subject_`-$I\Delta$: `people`$_1$; ΔI-`action_`-IO: `sack`;
ΔI-`object_`-IO: $\mathcal{VN}_1\}$

(E_1,H): $\{\Delta I$-`subject_`-$I\Delta$: `Sengan`; ΔI-`action_`-IO: `meet`;
ΔI-`object_`-IO: $\mathcal{VN}_1\}$

[43]It is not claimed that explicitly enumerating all the possibilities need be the only method used to solve this problem. A hybrid method of enumerating the common possibilities, while including an unpreferred generic alternative to be refined by other algorithms may be more appropriate.

(E_2,H): {ΔI-subject_-IO: "John"; ΔI-action_-IO: words_used;
ΔI-object_-IO: $\mathcal{VN}_1$}

(E_3,H): {ΔI-subject_-IO: [E_6, E_7, ...] ; ΔI-action_-IO: xor_}

(E_4,H): {ΔI-subject_-IO: [E_0, E_1, E_2, E_3, ...] ; ΔI-action_-IO: and_}

(E_5,R): {ΔI-subject_-IO: "John was sacked ... I met him";
ΔI-action_-IO: phrase_means; ΔI-object_-IO: E_4}

(E_6,H): {ΔI-subject_-IO: John$_1$; ΔI-subject_-$I\Delta$: $\mathcal{VN}_1$;
ΔI-action_-IO: synonym_}

(E_7,H): {ΔI-subject_-IO: John$_2$; ΔI-subject_-$I\Delta$: $\mathcal{VN}_1$;
ΔI-action_-IO: synonym_}

...

Similarly, sentences such as *"John hit Bill. He died"* can be treated: "He" may refer either to John or to Bill, but initially which is not known.

o Under-specified variable nodes

Another important feature of variable nodes is that they are integrated into the hierarchy. It would be possible to make all variable nodes children of typeless, but it proves more useful to make them into the smallest common superset which includes all of their alternatives. The variable node then has all the features common to the alternatives it represents. It is under-specified, in that it is a concept which includes all of its alternatives. Often however, the information shared by the alternatives is sufficient for useful disambiguation. Take for instance *"The frog lived near the fence"*. LOLITA has 3 meanings of frog: an adornment, a Frenchman, and an amphibian. Even if LOLITA is unable to decide whether a Frenchman or an amphibian is being referred to, she knows it is an animate being, since semantic selection of the verb "live" dismisses the adornment meaning. If the next sentence is *"It rose every morning"*, the meaning of "rise" reserved for astronomical objects such as the sun, moon and so on, can be dismissed because the frog is animate. [44] In computational terms, it is far better to access shared information than it is to consider each alternative individually.

[44] In many languages the syntactic gender of the pronoun can further refine the choice. *"it"* indicates *"the frog"*.

Maintaining the variable node as the smallest common superset requires processing each time one of its alternatives is deleted. However, the information about the features shared by the alternatives is accessed very often. When processing often performs the same operation, computational efficiency is improved by memoizing the resulting information. Seen in this light, ensuring that the variable node is always the smallest common superset is of obvious computational benefit.

By providing a clear conceptual distinction between features shared by all of the alternatives, and features true of each individually, variable nodes substantially simplify the disambiguation algorithms. Two classes of disambiguation algorithms are obtained: a simpler class applying only to variable nodes, which use less information and which are thus used as a first filter. The second class are those that apply to individual alternatives. These are used for variable nodes with few alternatives once the simpler rules have been applied.

∘ Variable node grouping

Variable nodes are further extensible by allowing their alternatives to be further grouped into smaller sets. For instance, the word "cabinet" can refer either to a piece of furniture, the people in the cabinet office, or to the cabinet office itself. By grouping these meanings into furniture and government pertaining concepts, sentences such as *"The craftsman built the cabinet"* can quickly be disambiguated. In general, the ambiguity will involve many more meanings than the above three, and grouping will prove essential when processing packed structures (see D.8.4 *(p. D-128)* below).

Grouping is obtained simply by adding new variable nodes, each connected to the alternatives forming part of its group. The main variable node is then connected to these new variable nodes, and to any alternatives remaining single after the grouping process. For instance,

(E_0,H): $\{\Delta I$-`subject_`-$I\Delta$: $\texttt{craftsman}_1$; ΔI-`action_`-IO: `build`;
ΔI-`object_`-$I\Delta$: $\mathcal{VN}_2\}$

(E_1,H): $\{\Delta I$-`subject_`-$I\Delta$: $\mathcal{VN}_1$; ΔI-`subject_`-IO: $\texttt{cabinet}_{furniture}$;
ΔI-`action_`-IO: `synonym_` $\}$

(E_2,H): $\{\Delta I$-`subject_`-$I\Delta$: $\mathcal{VN}_1$; ΔI-`subject_`-IO: $\mathcal{VN}_3$;
ΔI-`action_`-IO: `synonym_` $\}$

(E_3,H): $\{\Delta I$-`subject_`-IO: $\mathcal{VN}_1$; ΔI-`action_`-IO: `inst_` ;
ΔI-`object_`-$I\Delta$: $\mathcal{VN}_2\}$

(E_4,H): $\{\Delta I$-`subject_`-IO: E_1; ΔI-`subject_`-IO: E_2;
ΔI-`action_`-IO: `xor_` $\}$

(E_5,H): $\{\Delta I$-`subject_`-$I\Delta$: $\mathcal{VN}_3$; ΔI-`subject_`-IO: $\texttt{cabinet}_{building}$;
ΔI-`action_`-IO: `synonym_` $\}$

(E_6,H): $\{\Delta I$-`subject_`-$I\Delta$: $\mathcal{VN}_3$; ΔI-`subject_`-IO: $\texttt{cabinet}_{members}$;
ΔI-`action_`-IO: `synonym_` $\}$

(E_7,H): $\{\Delta I$-`subject_`-IO: $[E_5,E_6]$; ΔI-`action_`-IO: `xor_` $\}$

(E_8,H): $\{\Delta I$-`subject_`-IO: $[E_0, E_3, E_4, E_7, \ldots]$; ΔI-`action_`-IO: `and_`$\}$

(E_9,R): $\{\Delta I$-`subject_`-IO: `"The craftsman built the cabinet"`;
ΔI-`action_`-IO: `phrase_means`; ΔI-`object_`-IO: $E_8\}$

D.8.4 Packed Structures

As the analysis proceeds, likely combinations of meanings emerge. For instance, in the sentence *"Ross was escorted from the bar to the dock"*, the words "bar" and "dock" may invoke either a courtroom scene of a harbourside-scene. For simplicity we will only consider the 2 meanings of bar and dock. Of the resulting 4 meaning combinations, two are preferred: either both bar and dock are expected to refer to their courtroom meaning, or to their harbourside meaning. As variable nodes cannot capture this correlation, this type of information can only be expressed by unpacking the shared structures, and obtaining four structures, one for each of the possible interpretations of the sentence.

D.8.4.1 Correlations in ambiguous statements

Packed structures allow correlations between alternatives of different variable nodes to be expressed. This is achieved by realizing that the alternatives are expressed in terms of belief operators: `xor_` events. Just as `xor_` events are used to alternate between synonym_s, and_ events can be used to gather alternatives into an interpretation. The and_ event joining the alternatives into an interpretation states a belief in which alternatives the variable nodes express.

(E_0,H): {ΔI-subject_-$I\Delta$: people$_1$; ΔI-action_-IO: escort; ΔI-object_-IO: Ross}

(E_1,H): {ΔI-subject_-$I\Delta$: E_0; ΔI-action_-IO: origin_; ΔI-object_-$I\Delta$: $\mathcal{VN}_1$}

(E_2,H): {ΔI-subject_-$I\Delta$: E_0; ΔI-action_-IO: destination_; ΔI-object_-$I\Delta$: $\mathcal{VN}_2$}

(E_3,R): {ΔI-subject_-IO: "bar"; ΔI-action_-IO: words_used; ΔI-object_-IO: $\mathcal{VN}_1$}

(E_4,H): {ΔI-subject_-$I\Delta$: $\mathcal{VN}_1$; ΔI-subject_-IO: bar$_{drink}$; ΔI-action_-IO: synonym_ }

(E_5,H): {ΔI-subject_-$I\Delta$: $\mathcal{VN}_1$; ΔI-subject_-IO: bar$_{court}$; ΔI-action_-IO: synonym_ }

(E_7,R): {ΔI-subject_-IO: "dock"; ΔI-action_-IO: words_used; ΔI-object_-IO: $\mathcal{VN}_2$}

(E_8,H): {ΔI-subject_-$I\Delta$: $\mathcal{VN}_2$; ΔI-subject_-IO: dock$_{harbour}$; ΔI-action_-IO: synonym_ }

(E_9,H): {ΔI-subject_-$I\Delta$: $\mathcal{VN}_2$; ΔI-subject_-IO: dock$_{court}$; ΔI-action_-IO: synonym_ }

(E_{11},H): {ΔI-subject_-IO: [E_4,E_8; ΔI-action_-IO: and_ }

(E_{12},H): {ΔI-subject_-IO: [E_5,E_9]; ΔI-action_-IO: and_ }

(E_{13},H): {ΔI-subject_-IO: E_{11}; ΔI-subject_-IO: E_{12}; ΔI-action_-IO: xor_ }

(E_{14},H): {ΔI-subject_-IO: [E_0, E_1, E_2, E_{13}, ...] ; ΔI-action_-IO: and_}

(E_{15},R): {ΔI-subject_-IO: "Ross was ... dock"; ΔI-action_-IO: phrase_means; ΔI-object_-IO: E_{14}}

E_{11} and E_{12} express the two preferred interpretations. [45]

[45]Note that the representation given in the example is simplified: no particular commitment is made to the example representation of origin and destination, however if they were to be used they would refer to positions in a location model, rather the concepts "bar" and "dock". Also,

D.8.4.2 Belief and Interpretation

The existence of a node corresponding to an interpretation of the text allows interpretations to be referred to as concepts. Thus, alternative interpretations may be expressed, either as possibilities contemplated by LOLITA, or as believed in by different people. It also allows chains of reasoning to refer to interpretations, as in *"It was because John thought Margaret meant his wife was having an affair, that he disappeared."*

Because ambiguity is modelled in terms of belief, the `belief_value` event is used to rank alternative interpretations, in the opinion of different agents, including LOLITA. A subtle but important distinction must be made between `belief_value` events connected to `phrase_means` events and those connected to the logical connectives joining the parts of the various interpretations together. The former signify how much the agent believes that the `phrase_means subject_` textrefs mean their `object_` concepts. The latter signify how much the agent believes the statements formed by the conjunction of the interpretation's parts, independently from whether it believes the statement to be the meaning of the textrefs. For instance, one might not believe that John wanted some birthday present, but believe that the interpretation of *"Oh! Just what I always wanted!"* is that he wanted it.

D.8.4.3 Dismissing interpretations

The representation also allows impossible correlations to be expressed as such. For instance, *"The fan broke the window"* either means *"someone broke the window with a fan"* or *"A fan broke the window"*, as in a football fan. Although disambiguation may not be able to chose which of the interpretations is appropriate, it should be able to represent that there is a choice between two of them. This might seem trivial, simply being a matter of a `xor_` node between the two possibilities, but such solutions decrease sharing.

At different points throughout the disambiguation process, impossible meaning

properly E_1 and E_2 should be connected to instances of $\mathcal{VN}_1$ and $\mathcal{VN}_2$ respectively.

combinations may be found by different algorithms. Simply removing the interpretation from the packed structure often destroys the packed structure and reduces sharing dramatically. Since sharing is essential to maintaining efficiency, the representation must allow impossible meaning combinations to be represented while maintaining sharing.

The text being analysed is associated with a textref. The concepts forming its interpretation are the `subject_s` of an `and_` event, which is connected to the textref by a `phrase_means` event. Rejecting an interpretation corresponds to a disbelief that the textref means that interpretation. This can be expressed directly in the network by writing that the dismissed interpretations as `subject_s` of a `nand_` event, itself the `subject_` of the `and_` event connected to the `phrase_means` event. For instance, in *"The frog found the bugs"*, the interpretation *"the amphibian found the errors"* can be dismissed:

(E_0,H): {ΔI-subject_-$I\Delta$: $\mathcal{VN}_4$; ΔI-action_-IO: find;
ΔI-object_-$I\Delta$: $\mathcal{VN}_2$}

(E_1,H): {ΔI-subject_-$I\Delta$: $\mathcal{VN}_1$; ΔI-subject_-IO: bug_{error};
ΔI-action_-IO: synonym_ }

(E_2,H): {ΔI-subject_-$I\Delta$: $\mathcal{VN}_1$; ΔI-subject_-IO: bug_{insect};
ΔI-action_-IO: synonym_ }

(E_3,H): {ΔI-subject_-$I\Delta$: $\mathcal{VN}_1$; ΔI-subject_-IO: bug_{spy_device};
ΔI-action_-IO: synonym_ }

(E_4,H): {ΔI-subject_-IO: $\mathcal{VN}_1$; ΔI-action_-IO: inst_ ;
ΔI-object_-$I\Delta$: $\mathcal{VN}_2$}

(E_5,H): {ΔI-subject_-IO: [E_1, E_2, E_3]; ΔI-action_-IO: xor_ }

(E_6,H): {ΔI-subject_-$I\Delta$: $\mathcal{VN}_3$; ΔI-subject_-IO: $\text{frog}_{amphibian}$;
ΔI-action_-IO: synonym_ }

(E_7,H): {ΔI-subject_-$I\Delta$: $\mathcal{VN}_3$; ΔI-subject_-IO: $\text{frog}_{frenchman}$;
ΔI-action_-IO: synonym_ }

(E_8,H): {ΔI-subject_-IO: $\mathcal{VN}_3$; ΔI-action_-IO: inst_ ;
ΔI-object_-$I\Delta$: $\mathcal{VN}_4$}

(E_9,H): {ΔI-subject_-IO: [E_6, E_7]; ΔI-action_-IO: xor_ }

(E_{10},H): {ΔI-`subject_`-IO: [E_1, E_6]; ΔI-`action_`-IO: `nand_` }

(E_{11},H): {ΔI-`subject_`-IO: [E_0, E_4, E_5, E_8, E_9, E_{10}, ...];
ΔI-`action_`-IO: `and_`}

(E_{12},R): {ΔI-`subject_`-IO: `"The frog found the bug"`;
ΔI-`action_`-IO: `phrase_means`; ΔI-`object_`-IO: E_{11}}

The input to LOLITA is currently text, but SemNet could also be used for a system with speech input. Other examples of this type of ambiguity occur in NL speech: *"Each bill requires a check/cheque"*[46], is ambiguous as to whether each bill must be payed by cheque, or each new law must be checked.

D.8.4.4 Structuring ambiguity

Disambiguation algorithms can further benefit if they structure the information they process to reflect information that is salient to them. This section describes three examples of such structure and shows how the representation allows it to be represented.

- **Examples of structure in ambiguity**

The locality of ambiguity may an important consideration. If many sentences are ambiguous because of one global ambiguity, it may be worth concentrating effort on disambiguating this ambiguity, than on the many local ambiguities. Indeed, the information given by a text is useless if most of the facts derived from it are still ambiguous. For this, the representation should be able to express locality of ambiguity.

Alternatively, the disambiguation algorithm may benefit more from a structure emphasising dependencies of ambiguity. If ambiguity is grouped by dependency, once an ambiguity has been resolved, simply reading the representation tells one the dependent ambiguities which might now be solved. Clearly, a group of ambiguities may depend on another group, so the representation must allow many levels of grouping. This structure would improve the efficiency of disambiguation,

[46]Example taken from [Hirst 87]

by guiding the search through the remaining ambiguity.

The structure could also reflect the preference for certain interpretations. Some interpretations may be preferred to others for a variety of reasons. For instance, frequently used senses of words may be preferred, whereas non-literal uses of words may be dispreferred. The structure could reflect this, so that disambiguation algorithms concentrate on the current best choices, while maintaining other possibilities: a classic best first strategy.

- **Expressing the structure of ambiguity**

Different interpretations of the text are formed by combining individual ambiguities using logical connectives. Because logical connectives are used and have properties such as associativity and distributedness, different structures can be built which are logically equivalent.

This observation allows dependencies between possible alternatives to be expressed. For instance, if a and b depend on c and d respectively and vice-versa, the flat alternative structure $(a \wedge b \wedge c \wedge d)$ can be rearranged to the logically equivalent $((a \wedge c) \wedge (b \wedge d))$. This can be repeated at many levels, where each step through the tree of and_ events from the topmost event expresses an increase in the level of mutual dependency between the alternatives forming its `subject_s`.

The structure can also reflect the preference for certain interpretations. To do this, it must be possible to represent any interpretation by a node. Representing correlations between alternatives, has the same requirement. To do this, the equivalence between statements like $\texttt{xor\_}(a_1, b_1, c_1) \wedge \texttt{xor\_}(a_2, b_2, c_2)$ and $\texttt{xor\_}((a_1 \wedge a_2), (a_1 \wedge b_2), (a_1 \wedge c_2), (b_1 \wedge a_2), (b_1 \wedge b_2), (b_1 \wedge c_2), (c_1 \wedge a_2), (c_1 \wedge b_2), (c_1 \wedge c_2))$ can be used. Since this step reduces sharing, it is often desirable to keep most of the expression of ambiguity in the first form, and only expand the preferred interpretation. It is not possible to do this using pure `xor_` and and_s. Indeed, the packed structure expressing all the interpretations will include the preferred interpretation. Creating a structure that did not include it would result in less sharing. For instance if $a_1 \wedge a_2$ were the preferred interpretation, the structure not including it would be: $\texttt{xor\_}([a_1 \wedge b_2], [a_1 \wedge c_2], [\texttt{xor\_}(b_1, c_2) \wedge \texttt{xor\_}(a_2, b_2, c_2)])$

Instead if maximal packing is maintained, the top logical connective between the packed structure and the preferred interpretation must be or_. Otherwise, the repetition of the preferred interpretation outside and inside the packed structure would eliminate itself: True xor_ True is False. Thus the full example would be $(a_1 \wedge a_2) \vee$ xor_$(a_1, b_1, c_1) \wedge$ xor_(a_2, b_2, c_2) By unpacking the preferred interpretation and placing it under the top logical connective, the disambiguation algorithms can concentrate on it. They do not need to deal with the extra weight of processing less likely candidates, as would be imposed by a representation which forced maximum sharing at all times. However the alternative candidates are maintained, and can be processed if the current preferred interpretation is dismissed.

There is also structure in the textrefs, which encodes notions such as locality: some textrefs correspond to phrases, others to sentences, and others to paragraphs, etc... Each may be the subject_ of a phrase_means event, which can be associated with the ambiguity at its level. For instance suppose that ambiguity in the referent of a pronoun spans two sentences a and b. a and b are each represented by a textref. These textrefs are each connected via a phrase_means event to an and_ event, c and d respectively. The ambiguity, represented by a xor_ event x will be a subject_ of both of these and_ events. But the paragraph in which a and b are, is also associated with an and_ event, e. e will not only have c and d as subject_s, but also x. This represents the fact that the ambiguity occurs at a super-sentence level. In common with the structure expressing preference, the ambiguity to process is expressed higher up the logical connective structure: nearer the nodes corresponding to the interpretation of the whole text.

Using structure within the tree of logical connectives is complementary to using logical connectives for correlating ambiguity, or for dismissing certain interpretations. Indeed, the disambiguation algorithms can use an algebra of transformations on the logical connective tree. For instance, when certain interpretations have been dismissed, it may be possible to delete the positive expression of the interpretation if the resulting tree retains the same amount of sharing, by using a xor_ $\neg a$ is true.

D.8.5 Hidden Assumptions

The discussion of ambiguity so far has been informal to simplify it. However, two hidden assumptions must be rendered explicit.

The first assumption is that ambiguity ultimately can be resolved to a unique meaning. That is to say every statement ultimately has only one interpretation. Although the precise interpretation may not be known, the assumption is that if one interpretation is chosen, the statement has no other meaning. Even though this is false for deliberately ambiguous statements, double-ententes and puns, it is a fair assumption for the factual type of information, and day to day statements LOLITA is designed to process. The scheme presented may be extendible to cover such possibilities, but this will not be considered within the scope of this thesis. Currently LOLITA still has difficulties producing a good interpretation of many literal sentences. Much work is still needed on this question before problems such as double meanings can be addressed. This limitation is therefore based on an understanding of the current limitations of natural language analysis, and the assumption that these will take quite a long time to resolve.

The second assumption is a consequence of the first: although concepts have been argued to be fuzzy to a certain degree, they are still assumed to be distinguishable. Thus although the ambiguity may involve two very similar concepts and be very subtle, the representations considered all insist that either one or the other of the concepts was meant. This insistence will occur despite the fact that there may be no information to decide, and it really makes no difference either way. Although using variable nodes will decrease the inconvenience posed by this insistence for most types of processing, the scheme could result in a large amount of the semantic net being occupied by useless ambiguity information. This can however be avoided using a clever disambiguation algorithm which detects these cases and adds new subsuming concept where necessary. Obviously the difficulty then lies in establishing how important the difference between two concepts is. Such an algorithm is beyond the scope of this discussion.

D.8.6 Features of the representation for disambiguation

As was shown by the example *"The shortest path between two points is a straight line"*, ambiguity is difficult because of its combinatorial behaviour. A naive representation, such as multiple semantic nets, will impose combinatorial complexity on disambiguation. The only method of defeating it, is "divide and conquer" strategy. Since the complexity stems from the number of possible combinations being a product of the number of alternatives of each local ambiguity, each local ambiguity must be treated as independently from all the others as possible. Since the NL analysis algorithms must also build and read from the representation, a representation that requires each combination to be written explicitly, without allowing any sharing, will necessarily have combinatorial complexity. A representation that emphasizes sharing of common structures will not impose combinatorial complexity on the disambiguation algorithms[47], and gives ample opportunity for clever strategies exploiting common structures.

SemNet's representation of ambiguity emphasizes sharing and allows clever disambiguation algorithms to exploit sharing. This section develops a few examples of this type of processing.

D.8.6.1 "The shortest path..."

The first example is *"The shortest path between two points is a straight line"*. After syntactic analysis, this had $11 * 4 * 16 * 5 * 13 * 25 = 1144000$ interpretations, due to word ambiguity. Much of this ambiguity is due to many subtle distinctions in meaning of the words, which could be grouped into broad bands of meaning. This difficulty can be alleviated by using variable node grouping, and underspecified definitions. The resulting broader concepts still retain enough information to make useful disambiguation decisions. Since fewer, albeit vaguer, concepts are considered the number of interpretations considered by the disambiguation algorithm is reduced substantially. One of the simplest forms of disambiguation is seman-

[47]Clearly, even with a representation emphasizing sharing, it is possible to write algorithms with combinatorial complexity.

tic selection. Substantial computational costs can be saved by using a first pass to perform semantic selection at the family level, which requires only looking up controls rather than searching through the net. For this reason, and since family broadly expresses conceptual similarity, family information is used to create the variable node grouping. The number of interpretations to consider is reduced to $2 * 3 * 10 * 5 * 2 * 10 = 6000$ which is some 2 orders of magnitude less.

If the decisions used to create the variable node grouping reflected the type of information needed to quickly eliminate some interpretations, a large number of possibilities can quickly be eliminated. Once this has been done, the surviving alternative concepts can again be combined into new variable node groupings, and the process repeated: the resulting variable node groupings are more specific than those of the previous generation. Eventually, the granularity may reach the alternative meanings of the sentence themselves. In essence, the process avoids the disambiguation algorithms being swamped by too much irrelevant information at a particular stage of processing.

D.8.6.2 A detailed example: "The astronomer married the star"

Consider the example of *"The astronomer married the star"*. There are four meanings of star in Wordnet: the astronomical object; anything looking like the star shaped figure; the lead actor in a play or film; and an expert, ace or whizz. Using family controls nodes, these can be combined into three variable node groupings: one for inanimate objects, one for shapes, and one for humans. There are two meanings of marry in Wordnet, illustrated by the sentences *"John married Mary"*, and *"the priest married the couple"*. In both cases, the template `object_` has family type "human", so the interpretations of star expressed by the variable nodes "figure" and "inanimate objects" can be dismissed.

There is no further information to disambiguate further the meaning of "star", so the second stage is the disambiguation of "marry". The templates of both meanings of "marry" differ in the quantification of their `object_`. The *"John married Mary"* meaning takes one object per event as expressed by a $F - F$ quantification on the `object_` arc. The *"The priest married the couple"* meaning takes a set of people of two elements for each event: $F - F\forall$ quantification and the relevant `size_` two event. Thus, the "marry" has been disambiguated.

The remaining ambiguity can only be resolved using additional information, for instance provided later in the text. The underspecified variable grouping node will therefore be maintained. The bulk of the disambiguation was done linearly in three steps, instead of combinatorially in six.

D.8.6.3 Maintaining underspecified variable nodes

Often, it will be impossible to disambiguate between similar meanings of concepts. For instance, according to Wordnet, there are two similar meanings of lake which both refer to a pigment. Since LOLITA is Wordnet compatible, both meanings are included in the semantic net. But the difference of meaning is so small that the choice between them is rarely determined. However the knowledge that the variable node is a pigment is sufficient for most purposes. Similarly, it is often difficult to distinguish between the lead actor in a play or film and an expert, ace or whizz.

Underspecified variable nodes provide a degree of robustness, by allowing LOLITA to reason with the information she analyses despite the eccentricities of the conceptual ontology she uses. Eventually, however, they should prove useful tools to determine which conceptual distinctions prove unnecessary: if LOLITA has been repeatedly been unable to distinguish between the same groups of concepts, while analyzing large corpora, and this is due to lack of information in the text rather than to the failings of her algorithms or data, the distinction between concepts may be unnecessary.

D.8.6.4 Node count

The final example illustrates the reduction in structure that must be built in order to represent multiple forms of ambiguity: *"A bishop visits every church near a* *lake"*. The following structure expresses the ambiguity due to the words bishop (3 senses), visits (4), lakes (3), churches (3), the prepositional attachment of near (to lake or to visiting) and the ambiguity of quantification (a bishop = an arbitrary bishop, or every bishop; and a lake = an arbitrary lake or every lake):

(E_0,H): $\{\Delta I$-subject_-IO: $\mathcal{VN}_{Bishops}$; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{Bishop2}$;
ΔI-action_-IO: synonym_$\}$

(E_1,H): $\{\Delta I$-subject_-IO: $\mathcal{VN}_{Bishops}$; ΔI-action_-IO: spec_;
ΔI-object_-$I\Delta$: $\mathcal{VN}_{Bishop2}\}$

(E_2,H): $\{\Delta I$-subject_-IO: $\mathcal{VN}_{Bishop2}$; ΔI-action_-IO: size_; ΔI-object_-IO: 1$\}$

(E_3,H): $\{\Delta I$-subject_-IO: $\mathcal{VN}_{Bishop2}$; ΔI-action_-IO: size_;
ΔI-object_-$A\Delta$: more_than_one$\}$

(E_4,H): $\{\Delta F\forall$-subject_-FO: $\mathcal{VN}_{Bishop2}$; $\Delta\forall\forall$-action_-IO: $\mathcal{VN}_{visits}$;
$\Delta\forall\forall$-object_-$\exists!$O: $\mathcal{VN}_{churches}\}$

(E_5,H): $\{\Delta F\forall$-subject_-$F\Delta$: $\mathcal{VN}_{Bishop2}$; $\Delta\forall\forall$-action_-IO: $\mathcal{VN}_{visits}$;
$\Delta\forall F$-object_-FO: $\mathcal{VN}_{churches}\}$

(E_6,H): $\{\Delta I$-subject_-$\forall\forall$O: E_4; ΔI-subject_-IO: E_0; ΔI-action_-IO: and_$\}$

(E_7,H): $\{\Delta I$-subject_-$\forall\forall$O: E_5; ΔI-subject_-IO: $[E_1,E_3]$; ΔI-action_-IO: and_$\}$

(E_8,H): $\{\Delta I$-subject_-$\forall\forall$O: E_5; ΔI-subject_-IO: $[E_1,E_2]$; ΔI-action_-IO: and_$\}$

(E_9,H): $\{\Delta I$-subject_-IO: $[E_6,E_7,E_8]$; ΔI-action_-IO: xor_$\}$

(E_{10},H): $\{\Delta F$-subject_-$F\Delta$: $\mathcal{VN}_{churches}$; $\Delta\forall$-action_-IO: in_loc;
ΔF-object_-$F\Delta$: pos$_1\}$

(E_{11},H): $\{\Delta FF$-subject_-$FF\Delta$: E_5; $\Delta\forall$-action_-IO: in_loc;
$\Delta\forall\forall$-object_-$\exists!\Delta$: pos$_1\}$

(E_{12},H): $\{\Delta I$-subject_-$\forall\forall$O: $[E_5,E_{11}]$; ΔI-action_-IO: and_$\}$

(E_{13},H): $\{\Delta FF$-subject_-$FF\Delta$: E_4; $\Delta\forall$-action_-IO: is_in;
$\Delta\forall\forall$-object_-$\exists!\Delta$: pos$_1\}$

(E_{14},H): $\{\Delta I$-subject_-$\forall\forall$O: $[E_4,E_{13}]$; ΔI-action_-IO: and_$\}$

(E_{15},H): $\{\Delta I$-subject_-IO: $[E_{10},E_{12},E_{14}]$; ΔI-action_-IO: xor_$\}$

(E_{16},H): $\{\Delta I$-subject_-IO: $\mathcal{VN}_{Lakes}$; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{Lake2}$;
ΔI-action_-IO: synonym_$\}$

(E_{17},H): {ΔI-subject_-IO: $\mathcal{VN}_{Lakes}$; ΔI-action_-IO: spec_;
ΔI-object_-$I\Delta$: $\mathcal{VN}_{Lake2}$}

(E_{18},H): {ΔI-subject_-IO: $\mathcal{VN}_{Lake2}$; ΔI-action_-IO: size_;
ΔI-object_-IO: 1}

(E_{19},H): {ΔI-subject_-IO: $\mathcal{VN}_{Lake2}$; ΔI-action_-IO: size_;
ΔI-object_-$A\Delta$: more_than_one}

(E_{20},H): {$\Delta F\forall$-subject_-FO: $\mathcal{VN}_{Lake2}$; $\Delta\forall\forall$-action_-IO: in_loc;
$\Delta\forall\forall$-object_-$\exists!\Delta$: pos_2}

(E_{21},H): {$\Delta F\forall$-subject_-$F\Delta$: $\mathcal{VN}_{Lake2}$; $\Delta\forall\forall$-action_-IO: in_loc;
$\Delta\forall F$-object_-$F\Delta$: pos_2}

(E_{22},H): {ΔI-subject_-$\forall\forall$O: E_{20}; ΔI-subject_-IO: E_{16};
ΔI-action_-IO: and_}

(E_{23},H): {ΔI-subject_-$\forall\forall$O: E_{21}; ΔI-subject_-IO: [E_{17},E_{19}];
ΔI-action_-IO: and_}

(E_{24},H): {ΔI-subject_-$\forall\forall$O: E_{21}; ΔI-subject_-IO: [E_{17},E_{18}];
ΔI-action_-IO: and_}

(E_{25},H): {ΔI-subject_-IO: [E_{22},E_{23},E_{24}]; ΔI-action_-IO: xor_}

(E_{26},H): {$\Delta\forall$-subject_-$\exists!\Delta$: [pos_1,pos_2]; $\Delta\forall$-action_-IO: near[48];

(E_{27},H): {ΔI-subject_-IO: Bishops_1; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{Bishops}$;
ΔI-action_-IO: synonym_}

(E_{28},H): {ΔI-subject_-IO: Bishops_2; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{Bishops}$;
ΔI-action_-IO: synonym_}

(E_{29},H): {ΔI-subject_-IO: Bishops_3; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{Bishops}$;
ΔI-action_-IO: synonym_}

(E_{30},H): {ΔI-subject_-IO: visits_1; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{visits}$;
ΔI-action_-IO: synonym_}

(E_{31},H): {ΔI-subject_-IO: visits_2; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{visits}$;
ΔI-action_-IO: synonym_}

(E_{32},H): {ΔI-subject_-IO: visits_3; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{visits}$;
ΔI-action_-IO: synonym_}

(E_{33},H): {ΔI-subject_-IO: visits$_4$; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{visits}$;
ΔI-action_-IO: synonym_}

(E_{34},H): {ΔI-subject_-IO: churches$_1$; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{churches}$;
ΔI-action_-IO: synonym_}

(E_{35},H): {ΔI-subject_-IO: churches$_2$; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{churches}$;
ΔI-action_-IO: synonym_}

(E_{36},H): {ΔI-subject_-IO: churches$_3$; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{churches}$;
ΔI-action_-IO: synonym_}

(E_{37},H): {ΔI-subject_-IO: Lakes$_1$; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{Lakes}$;
ΔI-action_-IO: synonym_}

(E_{38},H): {ΔI-subject_-IO: Lakes$_2$; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{Lakes}$;
ΔI-action_-IO: synonym_}

(E_{39},H): {ΔI-subject_-IO: Lakes$_3$; ΔI-subject_-$I\Delta$: $\mathcal{VN}_{Lakes}$;
ΔI-action_-IO: synonym_}

(E_{40},H): {ΔI-subject_-IO: [E_{27},E_{28},E_{29}]; ΔI-action_-IO: xor_}

(E_{41},H): {ΔI-subject_-IO: [E_{30},E_{31},E_{32},E_{33}]; ΔI-action_-IO: xor_}

(E_{42},H): {ΔI-subject_-IO: [E_{34},E_{35},E_{36}]; ΔI-action_-IO: xor_}

(E_{43},H): {ΔI-subject_-IO: [E_{37},E_{38},E_{39}]; ΔI-action_-IO: xor_}

(E_{44},H): {ΔI-subject_-IO: [E_9,E_{15},E_{25},E_{40},E_{41},E_{42},$E_4$3];
ΔI-action_-IO: and_}

E_4 of E_6 is observational with respect to $\mathcal{VN}_{Bishop2}$ and $\mathcal{VN}_{churches}$ corresponding to the reading *"Every Bishop visits every church near a lake"*

E_5 of E_7 defines $\mathcal{VN}_{Bishop2}$ as those who visit a church near a lake, corresponding to the reading *"Every church near a lake is visited by a bishop"*.

E_5 of E_8 defines $\mathcal{VN}_{Bishop2}$ as the bishop who visits all churches near a lake.

Only 52 nodes (44 events, 2 positions, and 6 virtual nodes) are used, whereas without packing, 23329 nodes would have been necessary to express every possibility. The number of readings is:

[48] "near" is replacing the set of events that would model it in the representation of location.

- 3 Quantification Ambiguity of *a bishop*
- 3 Quantification Ambiguity of *a lake*
- 3 Meanings of *bishop*
- 4 Meanings of *visits*
- 3 Meanings of *church*
- 3 Meanings of *lake*
- 2 Possible prepositional attachments of *near a lake*

making a total of $3 * 3 * 3 * 4 * 3 * 3 * 2 = 1944$ readings. Each reading requires 12 nodes:

- 3 `spec_` or `inst_` or `synonym_` events from $\texttt{Bishops}_x$, from $\texttt{churches}_x$ and from $\texttt{Lakes}_x$;
- 3 instances/specializations produced by the above events;
- 2 `in_loc` events for churches and lakes;
- 2 positions for the above 2 events;
- 2 events: visiting and near.

resulting in $1944 * 12 + 1 = 23329$ where the +1 stands for the top `xor_`event.

D.8.7 Ambiguity: Conclusion

A model of ambiguity based on the existing events defined in the network has been discussed. Because of its reliance on the existing notions of belief and textual references, the introduction of ambiguity does not require any changes to forms of processing which do not use information about ambiguity. Thus algorithms that do not use information about ambiguity, such as temporal reasoning are not required to cope with the additional complication of processing information about ambiguity.

A poor representation of ambiguity, for instance such as alternative links, would require changes to all processing algorithms, because distinct mechanisms are used to express unambiguous and ambiguous statements.

The computational efficiency of the representation is ensured by always maintaining a high degree of sharing while allowing alternative interpretations to be expressed. Using the `belief_value` event, the alternatives can be ranked, and the behaviour appropriate to LOLITA's beliefs be obtained. Clearly, the `certainty_value` of an interpretation can also be modulated, depending on the extent the disambiguation algorithms used plausible or valid reasoning. This would allow LOLITA to behave in the manner appropriate to the certainty she has in her interpretation.

Index