Department of Computer Science

University of Manchester

Manchester M13 9PL, England

Technical Report Series

94-2-1

Michael Brown

A Memory Model for Case Retrieval by Activation Passing

A Memory Model for Case Retrieval by Activation Passing [1]

Michael Brown

Department of Computer Science
University of Manchester
Oxford Road, Manchester, UK.
michaelb@cs.man.ac.uk

Technical reports issued by the Department of Computer Science, Manchester University, are available by anonymous ftp from ftp.cs.man.ac.uk in the directory pub/TR. The files are stored as PostScript, in compressed form, with the report number as filename. Alternatively, reports are available by post from The Computer Library, Department of Computer Science, The University, Oxford Road, Manchester M13 9PL, UK.

Abstract

This thesis is concerned with the development of an under-lying model of memory to support selective case retrieval for case-based reasoning. The major requirements are that retrieval should be highly flexible yet efficient. The traditional approach of "indexing" is rejected as being too restrictive while more flexible approaches in analogical reasoning are generally too computationally expensive.

Several important organisational principles are developed in the memory model. A network representation is advocated with a number of required extensions; such as multi-granular representation, context-based segregation and a statistically-based grading of paths. The organisation of memory offers the potential for the serial performance of a number of retrieval tasks that have previously only been addressed by assuming a massively parallel implementation. The retrieval mechanism developed is a novel activation passing technique that creates a gradation of stored cases during retrieval.

Empirical evidence gained using a prototype implementation of the memory model supports the feasibility of the technique. A meaningful grading of stored cases is consistently achieved without a reliance on an exhaustive search of memory. Evidence that the memory model will scale to use on large-scale case bases is also presented.

Chapter 1

Introduction

The purpose of this chapter is to provide the necessary background to the work described in this thesis and to establish the main aims of the work. In particular, the process of retrieval is focussed upon and its central role in the operation of knowledge-based systems identified. Two general requirements are established; that retrieval be highly flexible and computationally efficient. Case-based reasoning is subsequently described as a promising new paradigm for implementing knowledge-based systems. However, the need is recognised for an underlying model of memory to support the general framework of case-based reasoning. In particular, the common "indexing" retrieval strategy for case-based reasoning provides limited flexibility whereas more general models from the related discipline of analogical reasoning are typically too computationally expensive. It follows that the goal of the thesis is to provide a memory model that enables selective case retrieval to fulfill both the flexibility and efficiency requirements.

This thesis is concerned with the general field of Artificial Intelligence (AI)[1]. Of specific interest are Knowledge-Based Systems (KBSs). A KBS achieves its reasoning power through the representation of knowledge in a way that is explicit and accessible to the system itself. The use of KBSs is now wide-spread but many technical problems persist in terms of their construction and maintenance. Case-Based Reasoning (CBR) has recently emerged as a KBS paradigm and is rapidly gaining popularity because it offers the potential for overcoming many of these traditional problems.

The central issue of this thesis is memory; specifically the development of a model that facilitates the selective retrieval of stored cases to support CBR. This work is put in context in section 1.1 where the general role of memory for KBSs is considered and some fundamental requirements for memory retrieval are established. Section 1.2 then goes on to provide an introduction to CBR. A general description is given of what CBR entails and of why it is advantageous as a KBS paradigm. The aims of the thesis with respect to CBR are then described. In section 1.2.3, a brief explanation is given of the motivations behind the development of the memory model to support CBR. The problem of case

[1] Note that in order to minimise confusion of terminology, appendix A summarises the definitions for of all technical terms, such as *artificial intelligence*, used in this thesis.

retrieval is characterised in section 1.2.4 and related knowledge representation issues are discussed in section 1.2.5.

1.1 The Role of Memory in a KBS

This section introduces a general model for the operation of a KBS. Arguments are put forward for the importance of memory retrieval within the model and the fundamental requirements of a retrieval mechanism are established.

1.1.1 The Basic Architecture of a KBS

The basic architecture of a KBS is illustrated in figure 1.1. The most important data structure inside the KBS is the permanent memory which provides a long-term store for the explicit knowledge of the system. Attached to this memory store is an access engine which receives memory prompts and sends appropriate recollections to the second major data structure; the working memory.

Working memory typically stores a state-based dynamic history of the task being performed. Through interaction with the working memory a reasoning process can use the recalled knowledge to help satisfy the goals that are currently established for the KBS by some external problem environment. The accomplishment of these goals can be used to drive a process that updates permanent memory in order to bring about some form of learning in the KBS.

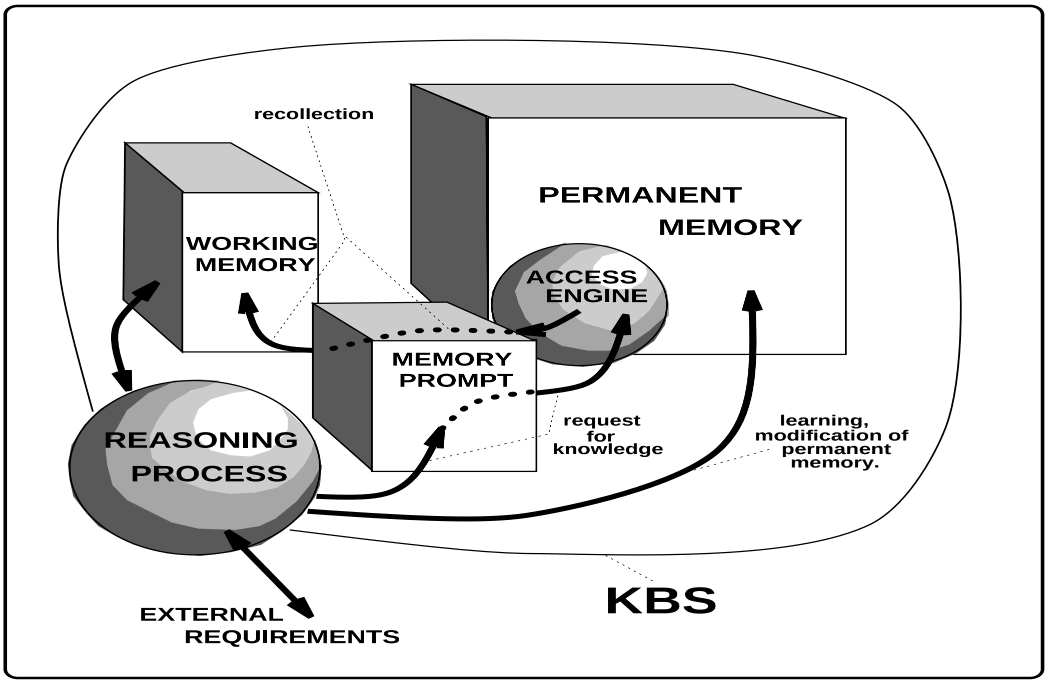


Figure 1.1: The Architecture of a Knowledge-Based System

1.1.2 The Importance of Knowledge Retrieval

The model of operation for a KBS suggested by figure 1.1 involves a three process cycle:

Process 1 Utilise Retrieved Information (Reasoning Process)
$Recollection + State_{old} \longrightarrow State_{new}$

Process 2 Generate Memory Prompt (Reasoning Process)
$State_{current} + ExternalRequirements \longrightarrow MemoryPrompt$

Process 3 Retrieve Recollection (Access Engine)
$MemoryPrompt + PermanentMemory \longrightarrow Recollection$

These processes are applied iteratively until a state is generated that satisfies the external requirements of the KBS. Of the three, Process 1 is usually the most straightforward. Once a knowledge item has been selected from memory, actually utilising it is generally a relatively simple interpretation process. For instance, in a rule-based system, using a selected rule only requires that the current dynamic state stored in working memory be augmented by the conclusions of the rule.

Whether it is Process 2 or Process 3 that carries the burden of performing most of the problem-solving work is a matter of philosophy. If the emphasis is placed on Process 2 then a *constructive* approach to problem solving is adopted. In this case, most of the work done is in comparing the current problem state to the goal state and in so doing establishing appropriate constraints for the next KBS cycle, such that there is a tendency to converge towards a solution within the vast space of possible problem states. An example of this type of KBS operation would be traditional operator-based approaches to planning [137].

An alternative approach is that of Memory-Based Reasoning (MBR) [130]. In MBR it is assumed that memory is organised such that the act of retrieval (i.e. Process 3) performs most of the problem solving. For example, the DMAP system [88] presents a model of memory that is so designed as to cause memory access to perform natural language parsing. More generally, MBR involves the storage and retrieval of reasoning traces, the aim being to convert a highly difficult constructive search problem into a more constrained recollection task. The fact that memory is bounded and can be designed to facilitate selective retrieval means that the search space that memory defines is inherently more tractable than the search space of all possible states that a KBS could generate. This principle is epitomised in CBR and is neatly summarised by a quote from Hammond[2]:

> "Every time you run into a problem which in the standard AI technology would be solved by searching a space of possibilities, in case-based what you do is search the space of actuality; you search the space of what *does* happen, what *has* happened, rather than the space of what *could* happen; because the space of what could happen is *outrageously* huge, and the space of what has happened is nice and tiny and tractable in comparison."

[2]From an address, "Reasoning as Remembering; The Theory and Practise of CBR", given at the AAAI-92 conference.

It is the MBR philosophy that is adopted in this thesis, hence the central interest of the thesis is in the process of retrieval.

1.1.3 General Requirements for Memory Retrieval

Memory retrieval can be viewed as a constraint satisfaction problem, as will be clarified later in section 3.1.1. Two potentially conflicting requirements can be established for a general model of memory to support MBR:

- **Flexibility:** There are no inherent restrictions in terms of the circumstances under which a particular knowledge item in memory can be potentially recalled, or in terms of the type of knowledge that can be held in memory.

- **Efficiency:** The computational effort required to find an appropriate item of knowledge in memory is not prohibitive.

It is the development of a memory model that satisfies *both* these requirements that is the main contribution of this thesis. The memory model is specifically designed to support the CBR paradigm. Accordingly, an introduction to CBR is provided below.

1.2 An Introduction to Case-Based Reasoning

This section starts by establishing what is meant by CBR. The interest in CBR is then justified in terms of its advantages as a KBS technique. However, the general framework for CBR is considered to be too abstract. A central goal of the thesis is therefore established; to provide an underlying model of memory that supports CBR. In accordance with this goal, the requirements for retrieval and representation of knowledge are assessed.

1.2.1 What Is Case-Based Reasoning?

CBR is a relatively new KBS paradigm. Its origins lie in work on frame representation [118]. A brief description of what CBR entails is given below.

"Reasoning is Remembering"

"Reasoning is remembering" is the slogan that most researchers in CBR adhere to. The basic principle behind CBR is that once a solution to a particular problem has been produced (by whatever means), the problem and its solution should be stored in memory as an experience (i.e. a *case*). Stored cases can be recalled to supply a solution to *similar* problems encountered in the future. Furthermore, for many applications the recollection of previous mistakes (i.e. failed cases) may also be highly advantageous as this can prevent future repetition of a serious error.

For an example of CBR in action imagine that, in her early days of crime, Helen Jones was a shoplifter[3]. One of her early experiences is recounted in figure 1.2.

[3]The problem domain of performing a robbery has been chosen for purely illustrative purposes within

```
At the tender age of 15, Helen was caught attempting to steal some
clothes  from a shop in  the  ''Trendi-clobber''  high street chain.
Although Helen had successfully concealed the plundered items, she
was unaware that they were electronically tagged. This tagging
caused a hidden alarm system to trigger when Helen left the store,
hence causing her to be detected and apprehended.

While  languishing in  custody, Helen  had time to  consider her
mistake. Through a conversation with another prisoner, she realised
that she should have checked the store more thoroughly. She also
realised  that to  perform a  successful  robbery, as well as
concealing her bounty she would have to remove the tags from the
stolen clothes prior to leaving the store.

Fortunately, due to her youth, Helen was not prosecuted and was
let off with a severe warning. On furthering her shoplifting career
Helen was always reminded of her harrowing experience whenever she
robbed a store. Consequently, she always looked for signs of a
hidden alarm on the shop entrance. She always carried a small pair
of scissors and, assuming she could do so without being observed,
used the scissors to remove any electronic tags. As a result her
career flourished.
```

Figure 1.2: An Illustration of Case-Based Reasoning as a Means of Avoiding Previous Mistakes

A CBR Framework

7 Stages for Case-Based Reasoning

1. **Encode** a description of the target case in memory (perhaps identifying salient aspects)

2. **Retrieve** source cases, guided by the features of the target case

3. **Determine a relevancy grading** for the retrieved source cases with respect to the target case (generally based on some measure of 'similarity')

4. **Select** the most salient source case

5. **Map additional knowledge** from the source to the target case

6. **Evaluate** the mapped knowledge to verify that it is valid for the target case. If failures detected:-
   - Attempt to **adapt** the source case to remove the clash, goto 6
   - If adaptation fails, **reject** the source case, goto 4

7. **Update memory for learning.** stored knowledge may include: a completed target case, any major source case adaptations, noteworthy failures, and a generalisation of the successful mapping

Figure 1.3: A Framework for Case-Based Reasoning

The preceding section gave a taste of what CBR is, via a fictitious example. In figure 1.3 a more thorough general framework for CBR is presented[4].

1.2.2 The Advantages of Case-Based Reasoning

Having given a brief introduction of how CBR operates this section is concerned with explaining the potential benefits of the technique.

---

this thesis (this stands as a novel contrast to the frequent use of CBR in legal reasoning applications!). All examples are completely fictitious. It should also be stressed that the system developed in this thesis hasn't been practically used for any criminal applications.

[4]The term *target case* will be used throughout this thesis to refer to the description of a current problem and the term *source case* used to refer to any previously completed problem solving episode that is stored in memory.

The Principle of Cognitive Economy

The most fundamental assumption in CBR is that it usually takes less effort to retrieve and adapt an old problem solution than to derive a solution to a new problem from first principles. For example, CBR can be used to anticipate problems by referring to previous errors [58, 80]. The assumption of reduced effort is borne out in many applications but must be treated cautiously; as discussed in [17], many of the stages of CBR are themselves complex problems that may only be soluble after significant effort.

The Use of CBR in Weak Theory Domains

Highly deductive KBS techniques, such as theorem proving or explanation-based reasoning, depend heavily on the possession of a complete causal model of the problem domain. In weak theory domains, such as law [6, 16, 7, 125] the causal model will, at best, be incomplete. In such domains the use of precedent cases (i.e CBR) may be the most suitable form of reasoning.

Ease of Construction

A major problem generally associated with the construction of a KBS is the so called *"knowledge acquisition bottleneck"*. This bottleneck is caused by a mismatch between the form of the raw data that can easily be extracted from typical knowledge sources such as a human expert and the more structured and precise format that is generally required for the representation of knowledge within a KBS. The problem has lead to much work in the development of efficient knowledge acquisition methodologies, such as [97].

For CBR systems, the work required for knowledge acquisition may be significantly less because there is no need to amalgamate raw knowledge, in the form of a collection of problem solving anecdotes, into a single coherent model. Rather, knowledge can be represented as a collection of largely independent (possibly even contradictory) episodes and hence a minimal amount of knowledge translation is required [25].

Robustness and Maintenance Through Incremental Learning

Another common problem with KBS is the *"plateau of expertise"* problem [85]. This pertains to the fact that KBSs generally exhibit high expertise with respect to the specific task that they are designed to perform yet are totally ignorant beyond the bounds of this expertise. This means that, at the periphery of its expertise, a KBS's reasoning can become 'brittle' in the sense that it cannot deal with unforeseen situations. The root cause of this fragility is that unlike human problem solvers, a KBS is unable to fall back on 'common sense' when its specific expertise is not applicable.

Common sense has been described in AI as the ability either to apply generalised (domain independent) knowledge [103, 70] or to adapt knowledge from other domains to the current situation [85]. There exist projects, notably [86], that are attempting the massive task of encoding sufficient knowledge for a KBS to begin to exhibit something like common sense. However, such large scale approaches are clearly not viable for most practical applications.

This problem of bounded expertise is also relevant to the problem of maintaining a KBS system within a volatile or weak-theory domain. In such domains it cannot be expected that the KBS possesses a complete model of the domain at any one time. Within such domains the KBS exhibits fragmented expertise with respect to the application task. Machine learning [68] is frequently cited as a necessary component for overcoming this problem.

CBR has great appeal in terms of robustness and maintainability due to the relative ease with which new knowledge (in the form of cases) can be added to a system's memory. The system can incrementally learn so as to cater for any exceptional and unforeseen circumstances simply by being provided with an appropriate case that tackles that situation. The robustness that this incremental learning provides has been empirically demonstrated elsewhere [129].

1.2.3 A Memory Model For CBR

One of the great appeals of CBR is that people identify with it as representative of what *they* actually do when *they* solve problems. This is encouraging in that it suggests that CBR may be related to something fundamental about human cognition. However, the subjective appeal of CBR may also be misleading. The framework of figure 1.3 is very abstract and should not be misconstrued as an algorithm. Several of the stages of the framework are themselves difficult problems [17]. Much of the real interest lies in determining techniques, such as the memory model of this thesis, that permit accomplishment of the various CBR stages.

Another important point is that, although the various steps of the framework are usually performed in the order presented in figure 1.3, there is often a high degree of interaction between them [40]. It will be argued in chapter 5 that the approach taken in this thesis, which emphasizes the role of memory and the process of retrieval, is sufficient to subsume most of the stages of the CBR framework.

Finally, the desire for generality in the model of memory developed here means that some non-CBR issues must also be addressed in its development. In particular, the process of analogical reasoning also requires encompassment, as explained below.

A Comparison of Case-Based to Analogical Reasoning

The differences between CBR and Analogical Reasoning (AR) are largely pragmatic [17]. The primary difference is that CBR usually operates within a single knowledge domain where as AR, by definition, must link cases in separate domains. Not surprisingly, this makes AR the more difficult type of reasoning to perform; in particular, there is a necessity to comply with the flexibility requirement for retrieval given earlier. Although several admirable implementations of analogical retrieval have recently been developed [139, 34, 45, 48, 82] none of these systems achieve their flexibility without a potentially high computational cost. The main contribution of the model developed in this thesis is, therefore, to explain how flexibility in case retrieval may be achieved for relatively low effort.

As an example of analogical reasoning, consider the response by Jane Smith, the cell-mate of Helen Jones, on hearing of her misfortune (see figure 1.4). The analogy between Jane's anecdote and Helen's experience connects two instances of a more abstract principle, namely; *"If you are trying to avoid some unwanted occurrence then simply attending to one possible cause of the occurrence does not protect you from other possible causes."* Hence, Helen's thoroughness in concealing stolen goods is analogous to Frank's thoroughness in checking for traffic. Helen's failure to notice the alarm system is analogous to Frank's failure to notice the open manhole. Finally, Helen's desire not to be caught is analogous to Frank's desire not to be killed.

```
''You see Helen, you made the same mistake as my uncle Frank. Frank
  had always worried about being killed by a car, so was always
  meticulous in checking for traffic when he crossed a road.
  Unfortunately, he never thought to check where he was stepping
  and one day he fell down an open manhole.''
```

Figure 1.4: An Example of Analogical Advice

Another difference between CBR and AR is in the nature of the linkage between source and target cases. For CBR this linkage is often made simply through the sharing of so called *"surface features"*. As is demonstrated by the above example, AR often requires cases to be linked via more abstract, less easily inferable, aspects. AR also often involves matching cases based on their overall descriptive structure [51]. It is much more difficult to organise a case base with respect to the factors required for AR than to organise a case base with respect to the values of readily observable features, which explains the difficulty of the task of analogical retrieval [120]. Nevertheless, this is one of the problems tackled in this thesis.

1.2.4 The Problem of Case Retrieval

The discussion given so far in this chapter has identified case retrieval as the problem of primary interest in this thesis. Accordingly, this section starts by establishing a more formal definition of the search problem that case retrieval entails. The requirements for case retrieval are then reassessed in the light of this definition.

The Search Space for Relevant Case Retrieval

The knowledge base for a CBR system can be thought of as a set of n independent source cases, $Cases = \{case_1, \ldots, case_n\}$. Generally in CBR, the task of case retrieval is to find the source case that best matches the current target case. The standard assumption is that the measure of the most relevant source case is some function of a one-to-one mapping between the features of the source and target cases. Generally, this function is some measure of the size of the largest coherent feature mapping.

Assume that the target case description has f_t features and the source case description has f_s features and that $f_t < f_s$. The space of mapping can be defined as that in which all possible permutations of the target case features are tried and for each of those permutations the target case features are bound one at a time to distinct features in the source case. This gives a mapping space size of:

$$MappingSpace = \frac{f_t! f_s!}{(f_s - f_t)!} \tag{1.1}$$

If, for simplicity, it is assumed that the n cases in the case base and the target case all have the same number of features, f_{ave}, then the search space for retrieval of cases based on feature matching has a size of:

$$RetrievalSpace(case_t, Cases) = n * f_{ave}! * f_{ave}! \tag{1.2}$$

The Requirements of Case Retrieval

The search space for case retrieval (as defined above) is in theory proportional to the number of cases in memory. It follows that for serial implementation, the speed of exhaustive search of all previous cases is inversely proportional to the size of the knowledge base. For some small scale CBR applications this is acceptable [129]. However, if CBR is to be used in a large scale application, or if the matching of each source case to the target case is computationally expensive, then exhaustive search becomes prohibitively slow. One possible solution is to use parallel techniques so that many previous cases can be compared to the current target case in a single time step [34, 33]. However, as will be discussed in chapter 3, parallelism alone does not solve the most complex and general retrieval problems. It follows that a refinement of the efficiency requirement for case retrieval is:

- **Efficiency:** The retrieval of a relevant source case should avoid exhaustive search of all cases in memory.

A means of avoiding exhaustive search during retrieval is the systematic and purpose-directed organisation of memory. This is the so called *"indexing"* approach to case retrieval. Indexing has been the most heavily researched area in CBR. A host of techniques exist [26] and general purpose indexing tools, such as $Re - Mind$[5], have recently become commercially available. However, the fact that an application specific structuring of memory is required means that indexing generally does not fulfill the flexibility requirement laid down on page 5.

A serial exploitation of a highly organised memory is, in fact, the approach taken in this thesis. However, the intention is that there should be a minimal restriction as to

[5] $Re - Mind$ is a trade mark of Cognitive Systems, Inc.

how a source case may be used in the future. In particular, analogy, as indicated by the example of figure 1.4, often causes a previous experience to be used in a novel way that would be hard to predict prior to encoding the case in memory. Indeed, even for single domain applications with a relatively strong domain theory, prior determination of what case features are relevant for retrieval may only be partially possible [8].

It is argued in this thesis that *similarity* should be an integral part of the case retrieval process (see also [18]). The notion of similarity can be subdivided with respect to what factors are involved in the similarity comparison between cases. Three general types of similarity are recognised here:

1. **Semantic Similarity** A correspondance in terms of an overall match between the set of features involved in two descriptions [143]. Loosely speaking, this is a correspondence due to like meaning.

2. **Pragmatic Similarity** A correspondance with respect to particular features of the two descriptions such that the selected features are relevant to the purpose of making the comparison [75, 5, 105]

3. **Structural Similarity** A correspondance between the pattern of relationships that connect components in each description [51].

The approach taken in this thesis is to replace indexing with a retrieval process that is *biased* towards the selection of past cases, based on a number of factors that relate to the above similarity types (see [19] for a summary of these factors). In order to achieve this the relationship between knowledge representation and knowledge retrieval must be established. The next section gives a brief introduction to some of the representation issues that need to be addressed.

1.2.5 The Requirements for Case Representation

The process of case retrieval is intimately linked to several aspects of case representation. These include: the knowledge that is stored in a case; the additional, supportive knowledge that needs to be stored; the structure of a case representation; and how memory as a whole is organised. Some general requirements with respect to these issues are outlined below.

Representing Individual Cases

For many applications it is useful to define general case description formats to simplify the characterisation and comparison of cases [145]. Unfortunately, for complex applications, treating cases as standardised and monolithic structures detracts from the flexibility with which the knowledge held in the case can be used. The more flexible approach, and hence the one adopted in this thesis, is to have a decompositional case representation that permits case fragments to be used in isolation and allows multiple cases to be combined in generating a new problem solution [34, 25].

The Need for Supporting Domain Knowledge

As has already been noted, the achievement of CBR is a non-trivial task. In particular, a host of additional knowledge, other than cases, may be required to support CBR. It follows that a complete memory-based approach to CBR must be sufficiently general to permit knowledge of various types to be stored and retrieved from the system's memory.

One aspect that may require domain specific knowledge is the measurement of similarity. In a general model of CBR, it should be possible to pair source and target case features that are themselves merely similar. This requires a sufficient domain model to characterise how the features are similar. Furthermore, the measure of similarity may be idiosyncratic to the particular feature type. For example, the numbers "100" and "150" may be judged quite similar as cricket scores, both being broadly classed as *"centuries"*. In contrast, "100" and "150" would be considered highly dissimilar if they were the values of two IQs. It will be shown in this thesis, via a number of examples, that through the explicit encoding of *"world knowledge"* several components of CBR, such as similarity assessment and adaptation, can be recast as indirect retrieval searches.

1.3 Thesis Overview

The structure of the rest of the thesis is as follows. In chapter 2, the issue of knowledge representation and memory organisation is considered, taking into account the general requirements for a memory model to support case retrieval. In chapter 3, it is argued that the proposed structuring of memory is sufficiently rich to allow a simple marker passing process to deal with a number of difficult and potentially intractable retrieval problems without resorting to exhaustive search of memory. The discussion of marker passing for retrieval is continued in chapter 4 where an activation passing mechanism is introduced and refined to suit the requirements of selective case retrieval. In chapter 5, the embodiment of the memory model in a prototype system is described. The general model for case retrieval is compared to existing work on the subject. In addition, arguments are put forward for the centrality of case retrieval in the CBR process. In chapter 6, the initial experimental work using a prototype of the memory model is described. In particular, the selective power of the implementation of case retrieval is demonstrated. Finally, chapter 7 provides the conclusion for the thesis, providing a summary of the ideas presented and indicating profitable directions for future development.

Chapter 2

Knowledge Representation

This chapter develops a knowledge representation tailored to the general requirements of case retrieval. A local network representation is chosen as it provides a means of explicitly representing the indirect connectivity between cases. A number of extensions to a basic network representation are required. A simple method for supporting multi-granular representation is introduced and is seen as a necessity for flexible case representation. A general model of "context" is developed. This allows for the focusing on relevant parts of the network in a manner that subsumes traditional "indexing" approaches to selective retrieval. Modularity is introduced to the network via a frame-like construct. The organisation of the frame modules is more general than that of related work and is orthogonal to any particular type of knowledge held by the memory. Again, this is seen as necessary if flexibility in memory usage is to be preserved. Finally, the rigid organisation of memory is augmented by a statistical relation weighting scheme. The usefulness of relation weighting is seen in terms of improving both the accuracy and efficiency of retrieval by feature-based search.

In the development of most AI systems, the issue of how to represent the knowledge required by the system must be addressed [14]. Most knowledge representations are tailored to emphasise those aspects of the domain that are most useful for the particular system's application [29]. Accordingly, the knowledge representation developed in this chapter is tailored to the task of case retrieval.

The representation requirements are clarified in section 2.1. In section 2.2 it is argued that a network representation is the most appropriate for the proposed memory model. The main body of this chapter is concerned with describing a number of extensions to the basic network representation that increase its suitability. The need for the support of multi-granular representation is discussed in section 2.3. Section 2.4 introduces the important notion of *"context"* and presents a means for including context in a network representation. Section 2.5 briefly discusses the need to embed frames within the network representation and section 2.6 provides a description of a novel weighting scheme that usefully augments the more rigid structuring of memory.

2.1 General Requirements

In many CBR projects the knowledge representation is highly application specific. By contrast, it was established at the end of chapter 1 that, in order to be suitable for use in a general model of memory supporting CBR, the required knowledge representation must be able to express knowledge that is complementary to the cases held in memory. In addition, knowledge should be in a format that can be actively used in the retrieval process (for example, to allow similarity to be appropriately assessed). These requirements equate to the need for knowledge to be represented *explicitly* in the memory model; this is explained below.

2.1.1 The Need to be Explicit

Explicit knowledge is that which can be interpreted in terms of the syntax of the particular knowledge representation. For example, a knowledge representation that supports the use of rules might include, for the robbery domain:

```
IF ``The safe is locked'' AND
   ``The room the safe is in is soundproof''
THEN  ``Use explosives to open the safe''
```

The pseudo-English style used in the symbolic representation of many KBSs can lead to miscomprehension as to what is actually 'known' by the system itself. For example, all that is explicit in the above rule is an inference from two internal states to a third state in a system presumably designed to reason in the domain of bank robbery. Knowledge such as the underlying causal justification of the rule (that explosions cause noise, which could lead to the detection of the robbery, etc) is not represented.

The main advantage of explicit knowledge is that it is a resource that can be used by the KBS itself. In particular, for memory-based reasoning techniques such as CBR, the issues of explicit knowledge representation and retrieval are inherently linked. It is only practically feasible to retrieve that knowledge which is represented explicitly within memory. More importantly, the organisation of the explicit knowledge representation plays an influential role in dictating what knowledge will be returned for a particular retrieval. A network representation is described in the next section as an appropriate form of explicit representation for the goals of this thesis.

2.2 Network Representation

The use of network representations is widespread in AI and has been popular since its emergence as an academic discipline. The equivalence between various network representations and formal logics is well established [127, 128]. Less formal network representations are also common. There is also much psychological evidence that the representation of knowledge in the brain possesses a network organisation [132, chapter 2]. The indication is that a network representation possesses the desired expressive power to support the representation of a range of knowledge types. The next section justifies the use of a network representation by discussing how its *metaphorical* character relates to the requirements of retrieval.

2.2.1 A Metaphor for the Real World

For any knowledge representation there exists a mapping between the primitives of the knowledge representation and knowledge items of the real world. For a metaphorical representation this mapping is so defined that structural aspects of the real world are mimicked in the structure *of* the representation, rather than being described *in* the representation. For example, in a metaphorical representation of visual information, geometric dimensions and orientation of phenomena in the real world would map to equivalent characteristics of the representation; in other words, the representation would have a pictorial basis [132, chapter2]. Metaphorical representations have the potential for high expressive power [144] and are appealing because relatively simple operations upon them can permit strong predictions about actions in the real world [24].

For a network representation, there are two basic primitives, *"nodes"* and *"links"* (or *"arcs"*). A *"local"* network representation, such as the one proposed in this chapter, is metaphorical in that knowledge items of the real world map directly onto nodes in the network and the associative connectivity that relates knowledge items has a one-to-one mapping to the relations of the network. Adopting the philosophy that a knowledge item gains its meaning from how it relates to other knowledge items [23], it follows that paths linking nodes in a network representation can be ascribed meaning. This notion has been exploited by researchers in analogical reasoning in an attempt to make analogical retrieval a syntactic graph-matching process [33, 148, 51], though it is highly debatable how central a role graph-matching alone plays.

A related idea is the *proximity principle* which states that the network distance between two nodes in the representation is indicative of the *similarity* of the two knowledge items they represent. The proximity principle can be exploited for case retrieval. For example, assume a certain knowledge item is associated with a case as one of its features. In the representation, the case will be explicitly connected to all other cases possessing the same feature via the single node representing that feature.

Generally, the connection between cases will be more indirect, via a path through memory. Recall from chapter 1 that a requirement for flexibility is that indirect mapping between case features be supported. The role of a local network representation can therefore be seen as constraining the search space that separates two case descriptions such that any path that can be forged between the two descriptions is in some sense meaningful. Furthermore, the shorter the network path the stronger the inter-case relationship it indicates.

2.2.2 A Description of a Basic Network Representation

Figure 2.1 illustrates the basic network representation proposed here[1] (with a description of why the greedy John Smith turned to crime). In particular, it is worth noting that relationships are treated as first class objects and are represented as *relation* nodes[2], rather than links as is more common. This is necessary because, as will be shown in chapters 3 and 4, knowledge about relationships is of primary importance during retrieval.

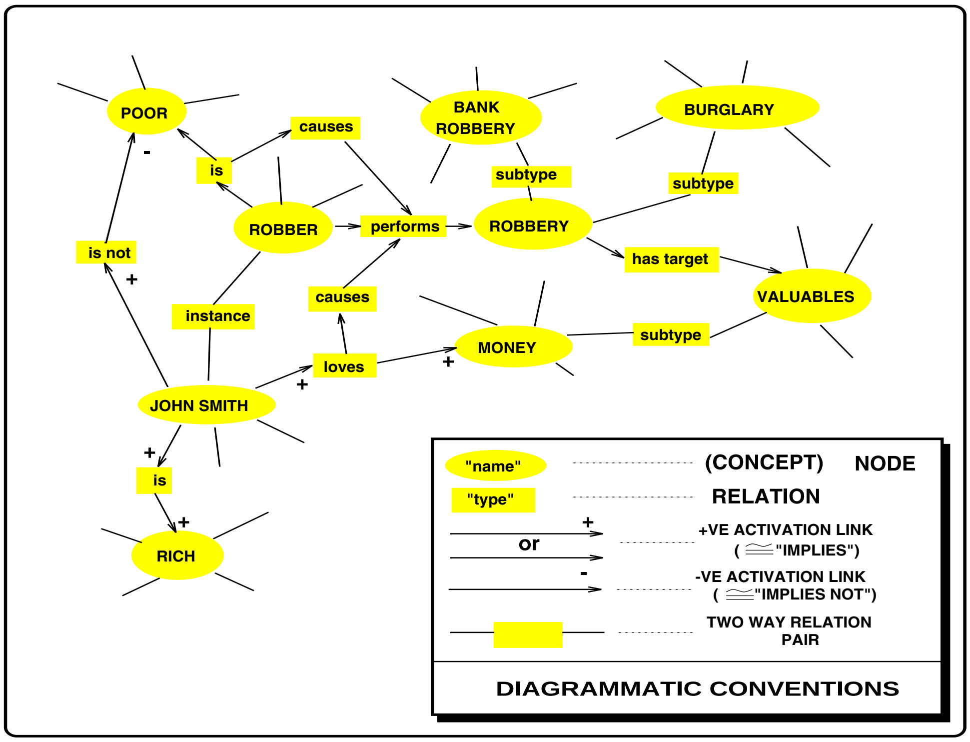


Figure 2.1: An Example of the Basic Network Representation

Any node can have any number of activation links[3] coming into and going out of it. Each of these activation links leads from/to a relation node, representing a specific relationship. The most basic syntactic structure in the network is a proposition of the form $Node_1 \longrightarrow relation \longrightarrow Node_2$, (see appendix B). $Node_1$ will be referred to as the *source* of the proposition and $Node_2$ the *destination*.

The proposed network representation lacks some of the formal rigour of semantic

[1]Note that the network of figure 2.1 is simplified for illustrative purposes. In particular, labelling relations by their type is a shorthand for a complex network structure connecting each relation node to a node representing its generic type (see appendix B). Similarly, labelling nodes by their name is a shorthand for a network connecting each node to nodes representing its denotations.

[2]In this thesis, the term *relation* will be used to denote the network primitives used to represent knowledge items of type *relationship*. The two terms should not be confused, the former existing at the implementation level, the latter at the conceptual level.

[3]The name activation link pertains to the role these links play in the retrieval process, see chapter 4 for details.

network representations. The knowledge representation proposed here is more akin to "scruffy" AI approaches to network representation, such as [41], or to the propositional network models of psychology [132]. A variety of types of knowledge may need to be represented in the network. For example, in figure 2.1 there are some relations, such as those that represent topological relationships, whose associated truth is invariable. Other relationships are much weaker, such as the suggestion that being poor causes most robbers to commit their crime. The issue of limited relationship validity is discussed further in section 2.4, where *context* will be considered as an alternative to preserving global consistency in a knowledge base.

2.3 The Need for Multi-Granular Representation

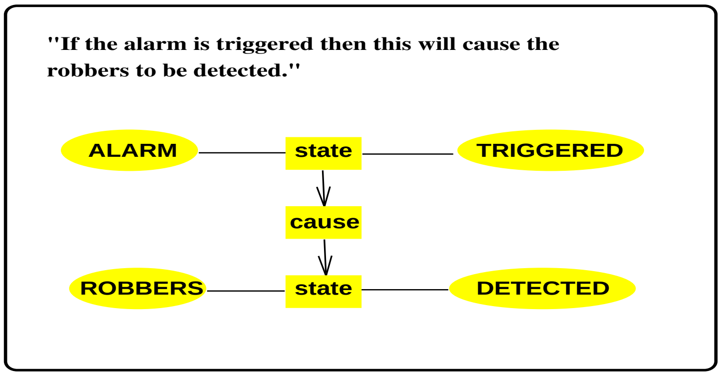


Figure 2.2: A Possible Representation of Higher-Order Relationships

To be adequately expressive, a representation needs to allow statements to be made about composite descriptions, as well as individual knowledge items. A straightforward way to represent higher-order relationships is to permit relations to connect to other relations, as shown in figure 2.2. However, this solution does not permit elegant representation of statements about complex descriptions. Such statements may be represented by multiple 2-arity relations, but a more straightforward solution is to permit many-to-one equivalence relationships to be represented within the network, as shown in figure 2.3. These relationships will be referred to as *structure links* because they permit multi-granular representation within the knowledge base.

Multi-granular representation is important for a general model of CBR. Firstly, it is linked with the need for a decompositional case structure (see section 1.2.5) in that it provides a straightforward way of tying together the distributed case components. Secondly, it allows the decision as to the appropriate level of abstraction for case representation to be left partially unresolved. This is important as the appropriate granularity is often task specific; for example, nodes that represent whole statements may be useful for story comprehension [53], whereas nodes representing more primitive *"micro-features"* may be required for more subtle similarity-based reasoning [60]. It follows that if flexibility in how a case is used is to be supported then multi-granular representation is a requirement.

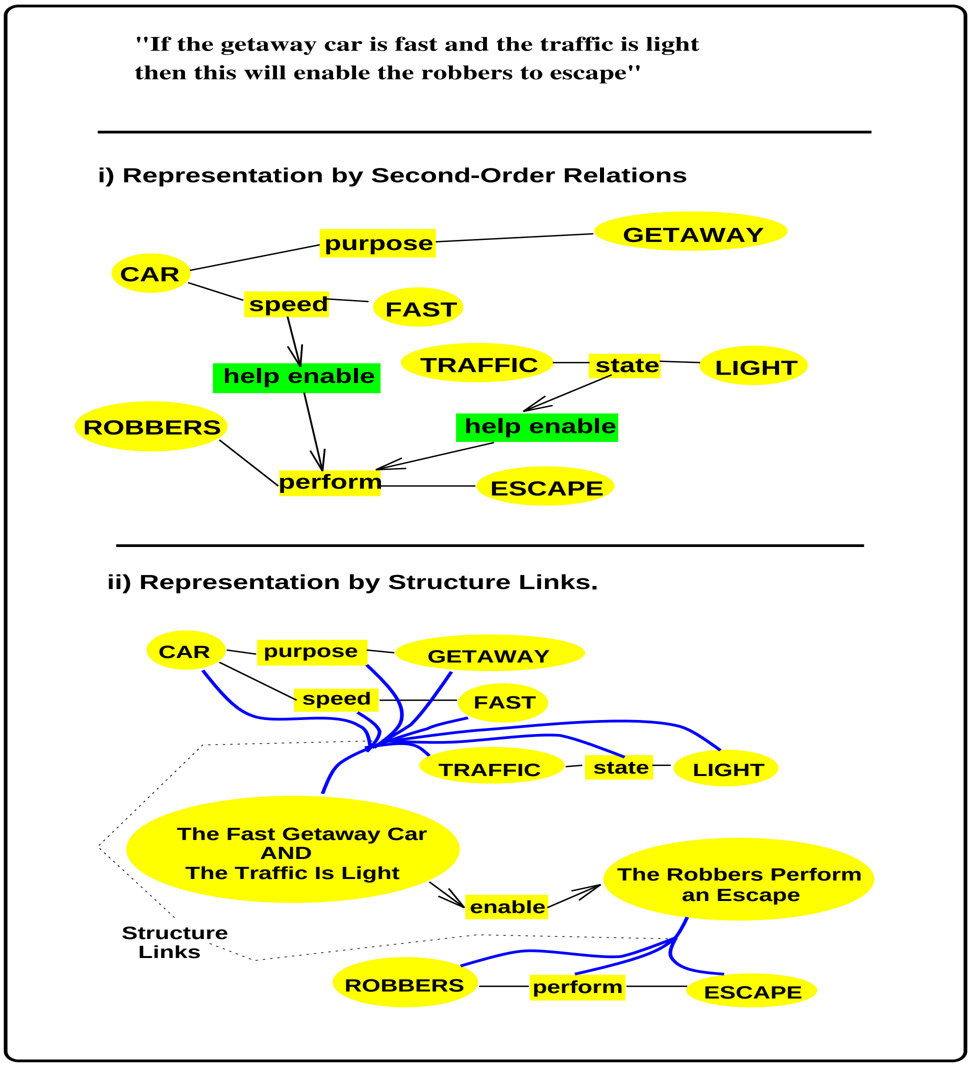


Figure 2.3: Representing Relationships About Complex Descriptions

2.4 Context

The model presented so far in this chapter is of a knowledge base consisting of a single and continuous network representation of the world. Significantly, there is no modularity within the representation. This section describes an augmentation of the basic representation with a modular structuring based on a general notion of context. Before describing the details of this enhancement, section 2.4.1 explains why it is a necessary component of the proposed memory model.

2.4.1 The Role of Context

The notion of a *"context"* is widely used in AI in order to select *relevant* information from that which is available. Two distinct types of context can be generally identified:

- **Context as a Dynamic History.** This type of context is a representation of the evolution of a reasoning process up to the current point in time. It is commonly used in natural language processing to support a range of tasks: from the low-level use of letters and letter features in the identification of other word letters [90]; to the use of words in a sentence to disambiguate the meaning of a subsequent word [136]; through to the high-level use of inferred user models [91] and recorded dialogue histories [93, 92] in processing large textual passages.

- **Context as a Static Viewpoint.** This type of context involves a precompiled organisation of the knowledge base such that sets of related knowledge items are bundled together [52, 107, 54]. Selection of a particular context determines a collection of knowledge items that are of particular interest. These viewpoints are generally orthogonal to any object hierarchy organisation of the knowledge base [91] and typically provide a means of selecting features relevant to a particular purpose [10, 107]. For example, goal-based indexing within CBR systems is a variation of this type of context.

There are a number of reasons why context needs to be represented in the memory model of this thesis. Firstly, being able to circumscribe when a particular knowledge item is valid increases the flexibility of the knowledge representation as it allows for partially true statements to be represented. Secondly, by focusing on relevant knowledge, context may be used to increase the efficiency of retrieval. More generally, supporting the influence of context on retrieval provides the required substitution for indexing in that the same selective role is performed but there is less in-built restriction as to how a case *may* be retrieved. The means for representing context that fulfills these objectives is described below.

2.4.2 Towards a Representation of Context

In terms of the model of memory being developed here a *"context"* should be treated like any other type of knowledge and be ascribed a single node in the representation. What is, therefore, required is a mechanism that permits any node in the network to influence the selection[4] of whole areas of the rest of the knowledge base.

An insight into how context selection could be represented comes from considering the role of context in assumption-based reasoning. A popular model for assumption-based reasoning is the *"possible worlds"* model [132, chapter 10] which is suitable for representing knowledge such as agents' beliefs [147]. For this model, no statement is considered to be absolutely true. Instead, it is assumed that a set of disjoint worlds, or contexts, are represented within the system and a given statement will only be true within

[4]It is perhaps worth noting that context can also act to emphasise (rather than discriminatively select) certain features of a description [102]. *"Contextual priming"* may have a significant effect on the ease of performing a successful analogical retrieval [63, 81]. However, in this thesis, contextual priming effects are best considered as part of an activation-spreading process, such as the one discussed in chapter 4, rather than being inherent in the knowledge representation itself.

a subset of all possible worlds. This type of knowledge representation is appropriate here as a fundamental assumption in CBR is that the knowledge held in a case is specific to the circumstances of that case and therefore *may* not be valid if applied, through the process of CBR, to other cases.

2.4.3 A Model of Context in a Network Representation

Any knowledge item can in principle be *referred to* in any context, regardless of whether the knowledge item actually exists within the context itself[5]. What varies from context to context is the validity of the relationships the particular knowledge item has with respect to other knowledge items. For example, the statement that *"John Smith is carrying a gun"* may be true within the context of a particular bank robbery. However, at the subsequent trial of John Smith, the statement will presumably be false even though any of: John Smith; the gun; the statement as a whole; may be *referred* to in the evidence presented at that trial[6]. It follows that the model of context defined here, should represent a focussed selection of relations within the network.

For a particular relation node there will generally be many contexts, both *positive* and *negative*. A positive context is a node that represents a knowledge item that indicates the *relevance*[7] of the relationship represented by a particular relation. A negative context indicates the invalidity of a relationship. The relationship between a context node and the relation it selects/deselects is represented by a *context link*.

An example of the use of context links is given in figure 2.4, which shows a portion of domain knowledge about bank robbery. Assume that there is some mechanism for selecting which nodes form the current context. If, for example, the node **"ACME SECURITY SYSTEM"** is included in the current context, this causes selection of the nodes representing the knowledge that it has a part that is a silent alarm and that it prohibits the neutralisation of an alarm. The **"SILENT ALARM"** node is a candidate for the next context, which would, if selected, cause the problem of the robbers being taken by surprise to be selected. Possible solutions to this problem could be selected by making the node that corresponds to the problem the next context or alternatively, by selecting the node representing the general concept of a solution.

More interestingly, several nodes can be used in unison to form a current composite context. For example, the general purpose of a retrieval session might be to anticipate problems and suggest solutions for a particular robbery. This might make the nodes

[5]Note that although it is often easier to think about contexts in terms of containment relationships, the model proposed here is intended to refer to any type of relationship that determines the relevance of knowledge items.

[6]Indeed, John Smith and the gun in question may both be present at the trial, nevertheless the "carrying" relationship between them will no longer hold.

[7]Note that a positive context does not indicate validity as, within any one context, the number of valid statements that can be made about a single knowledge item is in principle unbounded. Hence positive contexts focus on relations that represent relationships that are both valid and important. Similarly, negative contexts are only used for exceptional circumstances where a default positive context must be overridden. A number of relevancy conditions for the insertion of context links are listed in appendix G.

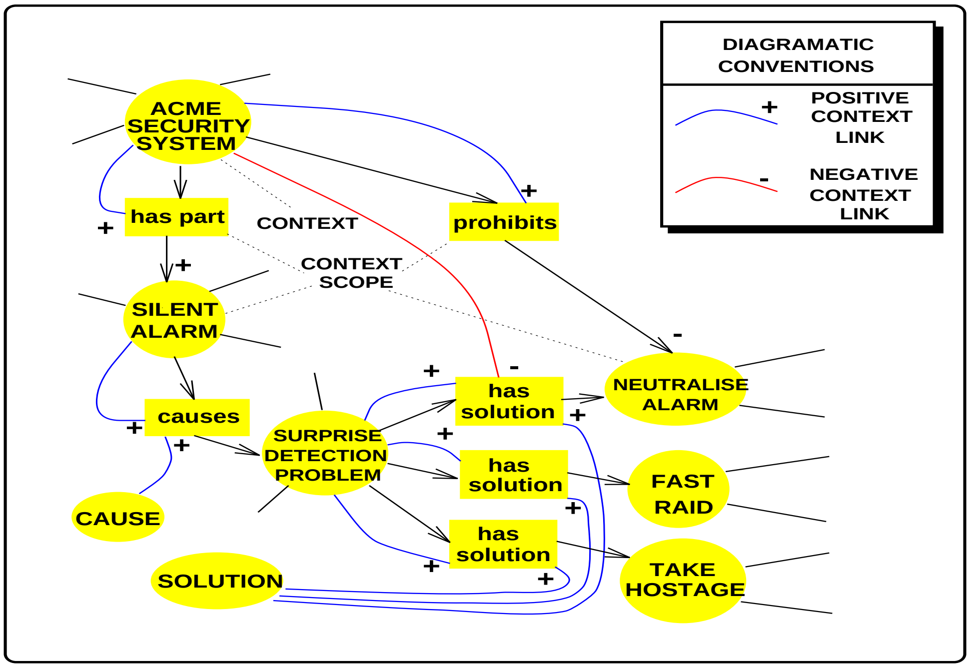


Figure 2.4: Contextual Selection of Relations

"CAUSE" and **"SOLUTION"** permanent members of the current composite context. It follows that as soon as a particular aspect of the robbery is activated in memory its causal consequences and the suggested solutions to any detected problems will also be automatically selected.

The above example illustrates the flexibility of the proposed context representation. This context representation is more fine-grained than the precompiled viewpoints of most other systems. However, the facility for the creation of arbitrary composite contexts makes the representation capable of subsuming more coarse-grained dimensional organisations of the knowledge base. Automating the dynamic selection of contexts is beyond the scope of this thesis (see section 7.3.1) but an illustration of the use of precompiled context sequences to guide retrieval can be found in appendix O.

2.5 The "Template"

In a flat network representation there is no principled way of bounding the knowledge associated with any one knowledge item represented by a node in that network; retrieval of knowledge by searching out from the associated node will eventually lead to access of the entire network. Although the representation of context described in the previous section goes some way to solving this problem, it would prove a cumbersome way of identifying what knowledge was specifically associated with each knowledge item represented in the network. Nevertheless, for a satisfactory model of memory, if a particular knowledge item is recalled then not only should the node associated with that knowledge item be retrieved but also the portion of the network that constitutes its description.

A second problem pertains to the fine-grained nature of the proposed network representation. The nodes and links that are its primitives contain little in-built semantics so that even fairly basic information, such as the name of a knowledge item, must be represented as a composite network pattern (see appendix B). Unfortunately, in accordance with the proximity principle given on page 16, the semantics that are inferred for the network may be sensitive to variation in the network structure. It is therefore desirable for consistency to exist in the way knowledge is represented in the network.

An obvious construct for satisfying the requirements of modularity and representation standardisation is the frame [94]. Accordingly the flat network representation proposed so far is enhanced to include frame-like structures which represent frequently occurring link patterns within the network. These frames will be referred to as *templates*.

2.5.1 The Description of a Template

Informally[8], a template is a collection of *role* nodes and relations. Each template constitutes the description of some general knowledge item. The relations are conjoined to form the characteristic network structure through their connection of the internal nodes of the template[9]. The network pattern will be considered as a collection of *features*. Each feature has the basic propositional structure identified on page 17, i.e. is a 3-arity tuple of the form (source-node, relation, destination node). Importantly, the relations of a template can also link one of its internal nodes to a node external to the template. In this way templates are *intimately* embedded as part of the global network of memory.

The role nodes of a template act as place-fillers that are substituted for more specific nodes[10] by an *instantiation link*. A single instantiation link generates pairings between a set of specific nodes and the role nodes of the template[11]. The effect is that each specific node inherits the links connected with its associated role node and the combined effect is that the entire network pattern of the template is inherited by the set of specific nodes. An example of the use of a template is described in the next section.

2.5.2 An Example of Template Use

Figure 2.5 illustrates a template that represents the general concept of a robbery. The definition of a robbery represented by the template is roughly:

[8]For a formal definition of a template refer to section C.1 of appendix C.

[9]Other network primitives such as the structure link and the context link are also typically used in the constitution of a template's characteristic network pattern.

[10]The nodes instantiated to a template are typically normal nodes, but role nodes can also instantiate other role nodes in the construction of a template hierarchy, such as that detailed in appendix B.

[11]An instantiation link that associates a set of specific nodes with the role nodes of a template will be said to *use* the template. Note that an instantiation link can indirectly use a template if a downward path through the template hierarchy from the instantiation link reaches the template.

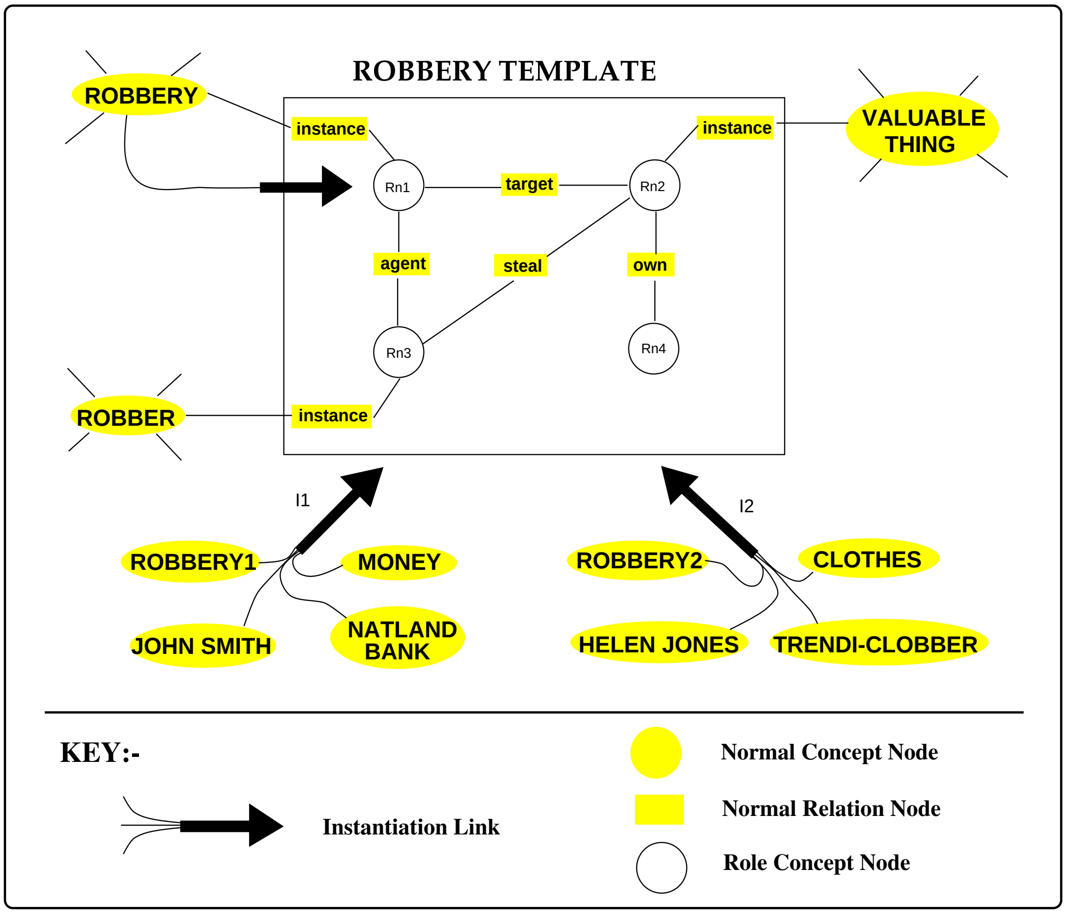


Figure 2.5: Concept Representation Using "Templates" Embedded in a Network Memory.

> A robbery has a target which is an instance of something valuable. The valuable thing is owned by something. A robbery also involves an agent who is an instance of a robber. That robber steals the valuable thing.

Note that the **"ROBBERY"** node instantiates[12] its own template (being linked to role node R_{n1}). This is a general principle; a node representing a knowledge item should be linked to the template that constitutes the description of that item. Another point to note is that the robbery template is defined in terms of knowledge items that are represented by nodes that lie outside of the scope of the template. For example, **"VALUABLE THING"** and **"ROBBER"**, despite being *referred* to in the definition of **"ROBBERY"**, in fact represents concepts in their own right and hence are not confined to the

[12]In terms of the use of instantiation links, no distinction is drawn between a node representing a general concept and a node representing an occurrence of that concept. In this sense the term "instantiation link" might be thought of as a misnomer. However, instantiation links are intended to represent a more general type of inheritance than traditional topological inheritance. The act of inheriting a network structure is the same no matter what specific type of knowledge is represented by the nodes that are associated with a template by an instantiation link; the terminology is intended to reflect the fact that the set of nodes is an instance of the particular network pattern.

robbery definition.

Figure 2.5 also illustrates a pair of instantiations of the robbery template with the details of two specific robberies. For example, instantiation link I_1 effects the following mapping: { **"ROBBERY1"**$\rightarrow R_{n1}$, **"MONEY"**$\rightarrow R_{n2}$, **"JOHN SMITH"**$\rightarrow R_{n3}$, **"NATLAND BANK"**$\rightarrow R_{n4}$}. Through this instantiation relational knowledge is inferred, such as that John Smith stole money from the Natland bank.

2.5.3 Related Work on Frames

The use of frames is of course nothing new in CBR research, indeed it formed the central principle in Schank and Abelson's pioneering work [116, 118]. The use of templates proposed in this chapter owes much to this work, but there are some notable differences.

There is no strong organisational principle that governs the structure of memory in terms of templates. This contrasts strongly with the central idea of Schank's work; that memory is organised around the servicing of goals and around expectation failures relating to those goals [117]. Because of this strong organisational principle, Schank is able to identify a number of different classes of memory structure, such as **Scenes**, **Scripts**, **MOPs** and **TOPs**. Conversely, the template described here is a more general descriptive structure that can be used to represent anything from low level syntactic conventions (see appendix B) to highly abstract concepts. The only organisational principle that connects templates is the inheritance of network patterns.

The lack of a strong organisational principle for templates is quite deliberate. The view taken is that any such organisation of knowledge is itself knowledge and should be appropriately and *explicitly* represented in the network representation, rather than implicitly in the global structure of memory. The context mechanism described in section 2.4 is sufficiently powerful to permit the dynamic selection of specific types of relation (such as those representing goal-centered relationships) as required. That this selection is superimposed on memory, rather than inherent in its structure, increases the flexibility of memory usage (see section 5.2.2).

2.6 The Relation Weighting Scheme

CBR application domains are frequently weak (in the sense described on page 8) and hence the causal relationships between observed features and problem solutions cannot be fully represented. In the face of this type of uncertainty, probability-based techniques are frequently employed in AI. For the memory model of this thesis the natural way of representing probability measures is as a weighting on the relations in the network.

Relation weighting also relates to the problem of modifying memory in order to implement learning. Case memories may explicitly represent large numbers of individual experiences and hence there is the potential for employing inductive clustering techniques, such as [140, 67, 35, 4], to build a hierarchy of abstract concepts on top of the cases. The issue of the automated restructuring of memory receives little more than a terse consideration in this thesis (see section 5.3.3). However, to be successful any inductive system needs to be guided by an appropriate biasing [95]. The relation weighting scheme put forward in this section could be of benefit in this respect as it provides a default gradation of memory that emphasises the *useful* paths within it.

The proposed relation weighting scheme involves two statistical measures, namely *typicality* and *idiosyncrasy*. These factors are similar to factors identified in the psychology literature [87, 132] and are defined in sections 2.6.1 and 2.6.2 respectively. Each factor is also described in terms of its meta-level role for knowledge representation and retrieval. Finally, in section 2.6.3 a propagation technique is proposed that overcomes problems concerning the localised effect of relation weighting.

2.6.1 Typicality

The typicality factor is inspired by the prototypical model of human concept representation [112, 113, 131]. Typicality is defined here in terms of the hierarchical organisation of templates in memory. Figure 2.6 presents an example template hierarchy representing a topology of businesses (potential targets for a robbery). This hierarchy is used for illustrative purposes in the description of typicality given below, and in section 2.6.2 for the description of idiosyncrasy.

A Description of Typicality

Typicality is defined with respect to a particular feature within a template and a particular instantiation link that (possibly indirectly) uses that template. Roughly speaking, typicality is a measure of the proportion of all uses of the template that assign the same value (i.e. a specific normal node) to the destination role node of the feature in question. A more thorough definition of typicality is found in section C.2 of appendix C.

Value assignment can be direct; as in the assignment of **"JEWELS"** to $node_{r5}$ by instantiation link I_7 (figure 2.6); or indirect via a path down the template hierarchy; as in the assignment of **"UTENSILS"** to $node_{r5}$ via I_5 and I_8. For the calculation of typicality ratio for a given feature, only template uses that make a specific value assignment to that feature are considered. Non-assigning uses (such as I_{11} with respect to rel_5 in figure 2.6) cause a default typicality to be assigned to the relation that is an average of all specified typicality values for that relation.

As an illustration of typicality assignment, consider the values that can be attributed to instantiation link I_1 of figure 2.6. With respect to rel_1 no typicality value can be ascribed as there exists no destination node at this level in the hierarchy. With respect to rel_3, there are 7 uses[13] of template T_3 that assign a value and 4 of these[14] assign the same value (namely **"JEWELS"**) as I_1. The typicality is, therefore, $\frac{4}{7} = 0.57$. With respect to rel_5 of the 10 value assigning uses of T_5, again 4 assign the same value as I_1, hence the typicality at this level is $\frac{4}{10} = 0.4$.

[13]Namely: $I_1, I_2, I_3, I_4, I_5, I_7, I_9$.

[14]Namely: I_1, I_2, I_3, I_7.

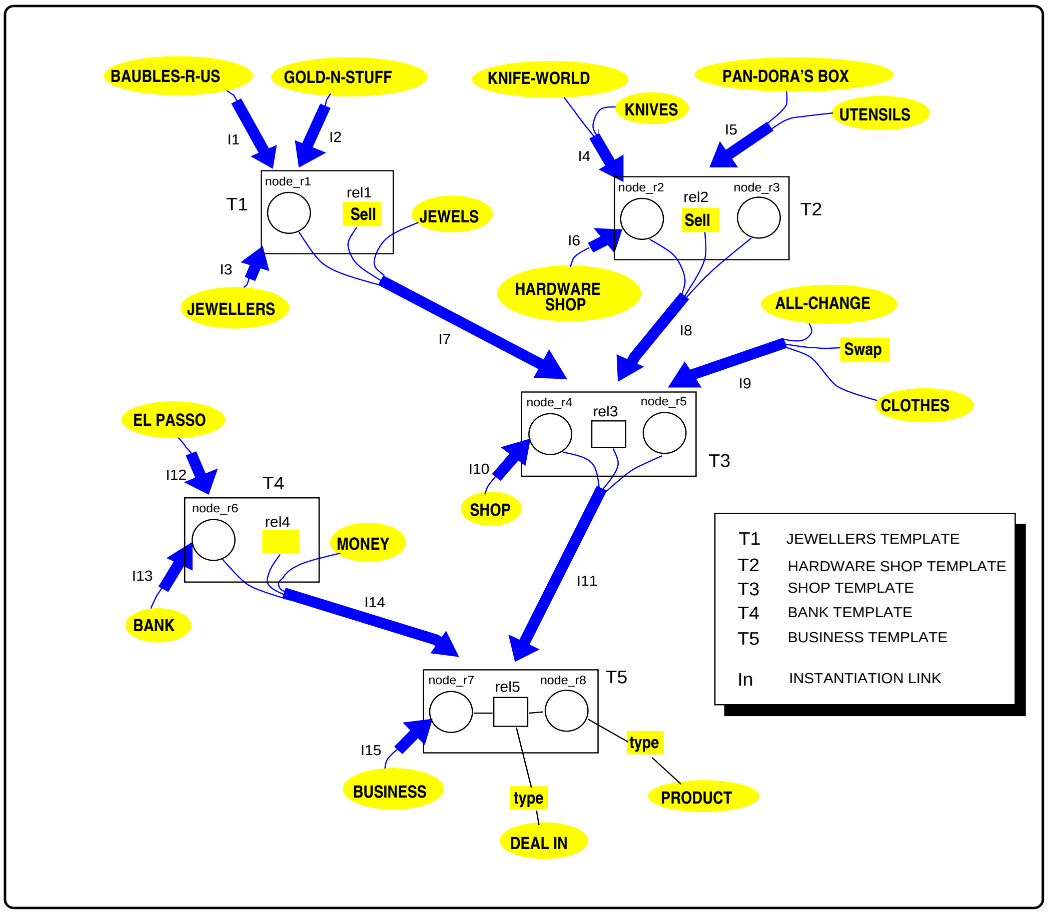


Figure 2.6: An Example Template Hierarchy

Typicality is an expensive measure to calculate [18] because it is a statistic based on the knowledge base as a whole. However the calculation need only be performed occasionally and may be reserved for times when the system is otherwise not in use. The justification for the calculation of typicality is given in the next section where the purpose is explained for the gradation of memory produced by typicality.

Structural Implications

In order to define typicality it has been assumed that the template hierarchy is predefined. This is contrary to psychological evidence that suggests people develop a prototypical concept representation *before* establishing a more theory-based representation of knowledge [77]. However, it is generally true that most AI applications will be based on some preconstructed conceptual domain model encoded within the knowledge base.

The issue being questioned in this section is whether typicality is useful. A definition of $Utility(Typicality)$ is given in section C.3 of appendix C. Basically, this utility measure is defined in terms of how well typicality guides the selection of a value from the bag [15] of all previous assignments in order to abductively infer an unknown feature value for a new case. The selection of a possible value assignment is equivalent to selecting one possible outward relation from a node when the current context provides no further guidance; in other words, it is synonymous with an under-constrained search of the network of memory.

$Utility(Typicality)$ equates to the extent to which a selection of the most frequently encountered value for some unassigned feature improves the probability that that value is correct, compared to a random value selection. The magnitude of $Utility(Typicality)$ is a function of the frequency distribution of the bag of all value assignments. Appendix D contains a number of proofs about the properties of $Utility(Typicality)$. These are summarised below:

- $Utility(Typicality) = 1$ if all possible assignments occur with equal frequency.
- $Utility(Typicality) \rightarrow 1$ as the dominance in the frequency of assignments for one particular value increases with respect to all other values.
- $Utility(Typicality)$ tends to increase as variance in the frequence distribution decreases.
- $Utility(Typicality)$ has a maximum when the frequency of the most prevalent value assignment is equal to $C_{tot}\frac{1}{\sqrt{l}}$; where C_{tot} is the total number of individual value assignments that have occurred and l is the number of distinct values that exist.
- $Utility(Typicality)$ tends to increase as the number of distinct values (i.e. l) increases. For high values of l, $Utility(Typicality) \propto \sqrt{l}$.

These findings are summarised in figure 2.7. The conclusion to be made is that typicality provides a significant and monotonically useful gradation of memory that improves the accuracy of unknown value selection.

2.6.2 Idiosyncrasy

The idiosyncrasy relation weighting factor described in this section is complimentary to the typicality measure defined in section 2.6.1. A characterisation of idiosyncrasy is given in terms of the template hierarchy and its utility explained as a gradation of memory that promotes more efficient feature-based retrieval.

A Description of Idiosyncrasy

Idiosyncrasy, like typicality, is a statistical measure based on the organisation of memory as a whole. It is defined with respect to a particular use of a template and represents how unusual is the assignment made to a particular template feature by that use. In

[15] That is, *"bag"* in the mathematical sense.

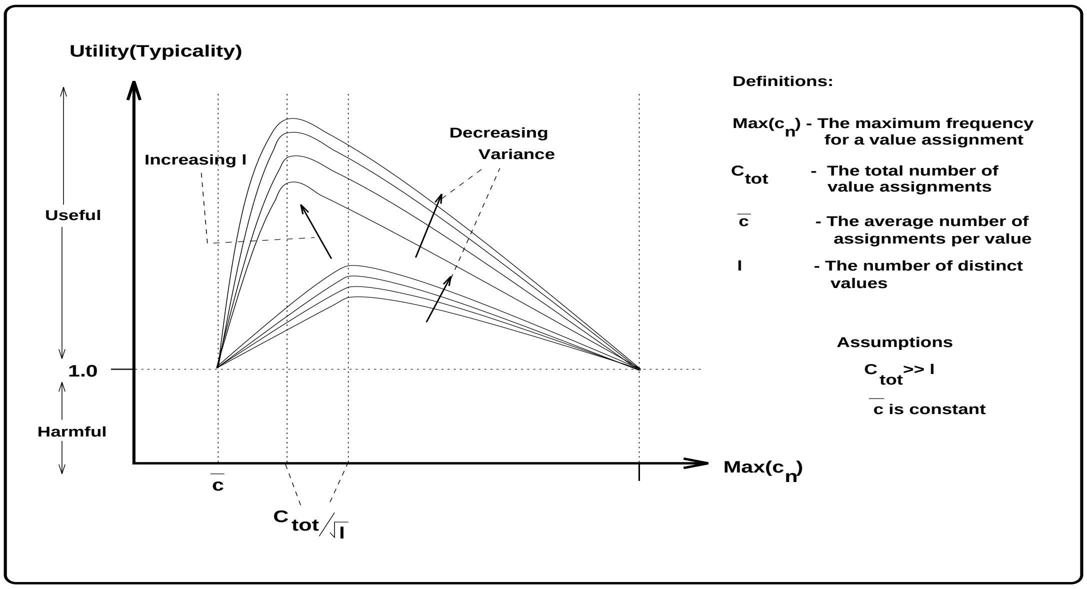


Figure 2.7: The Dependence of Typicality Utility on the Nature of the Value Frequency Distribution

loose terms, idiosyncrasy is a measure of the proportion of *other* uses of a template that make a differing assignment for a given template feature. A more rigorous definition of idiosyncrasy is found in section C.4 of appendix C.

The difference between feature assignments can occur with respect to two criteria:

1. The assignments associate a different value (i.e. normal node) with the destination role node of the feature. As an example of this, taken from figure 2.6, the (indirect) use of template T_3 by instantiation links I_4 and I_5 cause different value assignments to $node_{r5}$, namely **"KNIVES"** and **"UTENSILS"** respectively.

2. The assignments make differing specialisations of the relation of the feature. As an example of this, taken from figure 2.6, instantiation links I_8 and I_9 ascribe differing relation types to rel_3, namely **"SELL"** and **"SWAP"** respectively.

The instantiation link that a particular measure of idiosyncrasy is attributed to is discounted when determining the ratio of all template uses that make a matching feature assignment. In addition, for intermediate instantiation links, used to define some more specialised template in terms of a general one (for example, I_7, I_8, I_{11} and I_{14} of figure 2.6), not only is that instantiation link ignored in the calculation but also all those that make up the template hierarchy accessible above it.

As a worked example of calculating idiosyncrasy consider what value would be calculated with respect to I_7 and rel_3 in figure 2.6. The assignment made by this particular use of template T_3 involves associating the value **"JEWELS"** with $node_{r5}$ and the type specialisation **"SELL"** with rel_3. The instantiation links with respect to which idiosyncrasy can be calculated are $\{I_4, I_5, I_6, I_8, I_9, I_{10}\}$[16]. Of these, $\{I_4, I_5, I_9\}$ make value assignments to $node_{r5}$ that differ. In addition, I_9 causes a different type specialisation to rel_3. The combined set of differing uses is, therefore, $\{I_4, I_5, I_9\}$, which leads to the calculated idiosyncrasy value being $\frac{3}{6} = 0.5$.

Structural Implications

In section C.5 of appendix C a discussion is given leading to the definition of $Utility(Idiosyncrasy)$, a measure of how useful is the idiosyncrasy factor. This measure is concerned with the efficiency of feature-based search of memory. In section C.5, case retrieval is characterised in terms of the intersection of a number of search spaces based on single target case features. The argument is that the measure of idiosyncrasy allows the intersection to be formulated by considering the search spaces in increasing order of size. $Utility(Idiosyncrasy)$ is defined in terms of how much this ordering reduces the amount of effort that needs to be performed with respect to considering the features in a random order.

The lower bounds of $Utility(Idiosyncrasy)$ cannot be greater than 1 as there is always a possibility that a random ordering of features will be the same as that determined with respect to idiosyncrasy. However, the *expected* value for $Utility(Idiosyncrasy)$ can be defined in terms of the average amount of work that is required for a random ordering of case features. An empirical approach to this issue is described in appendix E and the results are summarised in the scatter plot of figure 2.8[17].

Similar conclusions to those reached for typicality can be drawn. In particular, as is illustrated in figure 2.8, idiosyncrasy is a factor that is generally beneficial in terms of reducing the amount of work done in classification or identification.

Measures similar to typicality and idiosyncrasy have been recognised elsewhere, notably [87], as a useful way of augmenting a knowledge representation. Where the relation weighting scheme defined here has most novelty is in combining the weight factors of typicality and idiosyncrasy via a propagation algorithm. Weight factor propagation is discussed in the next section.

2.6.3 Weight Factor Propagation

Typicality and idiosyncrasy play different and complementary roles in providing a useful gradation for individual relations within the memory network. Unfortunately, for a fine-grained network, the effects of individual weights will be highly localised, hence they will provide limited guidance to a retrieval that involves an indirect search through memory. This section describes a propagation method for overcoming this problem. For illustrative purposes the description of the El Passo Bank given in figure 2.9 is used. A complex description such as this one will generally involve the instantiation of several templates representing domain concepts; both general (such as **"BANK"** and **"SECU-**

[16] For the calculation, I_7 itself and the three instantiation links above it (i.e. I_1, I_2 and I_3) are discounted.

[17] The values plotted in this figure are taken from table E.3.

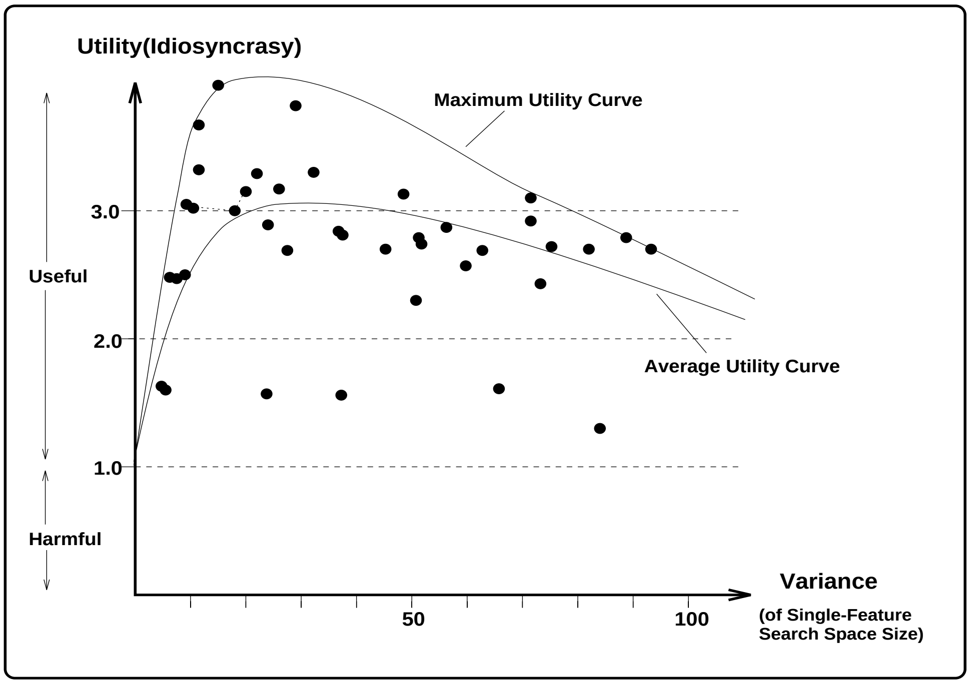


Figure 2.8: The Nature of Idiosyncrasy Utility with Variance in Feature-Based Search Spaces

RITY SYSTEM") and more specific (such as **"IMPREGNABLE BANKS, INC. MANAGER"**).

Generally, a relation will be inherited via several templates and will have several weight factors assigned. Figure 2.10.i represents a stylised view of how the El Passo Bank description might be graded after localised relation weight assignment[18], (see appendix F for a discussion of the plausibility of these weights and table F.1 for the specific values assumed).

Although some interesting default selectivity can be observed in figure 2.10.i (such as focusing on the fact that Louis D Crust owns an alligator) the graded structure is relatively coarse. This is likely to be true of more realistic knowledge bases as it may be expected that many relations will be assigned full (1.0) typicality and zero idiosyncrasy, reflecting a feature possessed by all members of a particular concept and no other concepts in memory. It follows that localised relation weighting may provide only limited guidance to a retrieval search through the network of memory.

The graded structure of a description can, however, be significantly enhanced by propagating weight factors within the bounds of that description. A propagation expression is described in section C.6 of appendix C. This expression merges the initial weight assignment for a relation with those of its successor relations within the network. The expression includes within it a biasing towards preserving those weight factors that are

[18]Note that a "fuzzy" logic conjunction is used to combine the typicality and idiosyncrasy weight factors to produce a single relation weighting.

highest. The iterative application of the expression allows for the delocalisation of relation weighting and for prolonged, highly weighted, paths through memory to be highlighted.

The propagation results for the example of figure 2.9 are described in appendix F and pictorially represented in figure 2.10.ii. From figure 2.10 it can be seen that weight factor propagation emphasises certain paths in memory that, intuitively, lead to the more 'interesting' information. In particular, the selected paths tend to be those that lead to highly typical components of the bank description that have subsequent highly idiosyncratic aspects. That this type of grading of memory is potentially *useful* is shown in chapter 3, where the proposed relation weighting scheme is discussed as a means of overcoming some classic retrieval problems.

2.6.4 Related Work on Probablistic Structuring of Memory

The use of a set of weighted preferences as a means for flexibly selecting useful source cases has been recognised elsewhere; notably in the PARADYME system [79, 80]. In PARADYME a mixture of traditional indexing criteria are combined with psychologically inspired factors, such as the recency with which a particular source case was last used. The weakness of PARADYME is that it only uses these preference criteria in a post-retrieval sorting process, retrieval itself is a relatively crude feature matching process.

The use of probablistic techniques to superimpose a "fuzzy" structure upon the organisation of memory is a technique that has been employed comparatively rarely in CBR research. Noteworthy exceptions are Langley et al's LABYRINTH [140] and DÆDALUS [83] systems.

In the DÆDALUS system, a frame hierarchy is induced for the problem domain, including a typicality measure for each slot of a frame. The leaves of the hierarchy are plan operators. For a new problem, an operator is selected by filtering the problem description down the hierarchy. Unlike other top-down approaches to retrieval, such as Schank's model, the typicality measure guides the downward search and this is shown [140] to improve operator selection accuracy. LABYRINTH is similar to DÆDALUS except that the cases are decomposed into part hierarchies, rather than being flat sets of attribute-value pairs. This leads to a more sophisticated classification scheme where the description components are filtered down the frame hierarchy prior to classification of the description as a whole.

Neither DÆDALUS nor LABYRINTH include a measure equivalent to idiosyncrasy. From the arguments of section 2.6.2 this is a possible source of improved classification efficiency. Similarly, because descriptions have a decompositional structure in LABYRINTH, the propagation technique of section 2.6.3 could be used to enhance the description structure by allowing weights at the higher levels in a hierarchical description to partially reflect the distribution of possible subcomponents at lower levels in the hierarchy.

Finally, in neither DÆDALUS nor LABYRINTH are the frames of the knowledge representation embedded within a more global network of world knowledge. This has some computational implications. For example, an expensive task in LABYRINTH is the mapping of features of a new problem description to those of the existing structures in memory [140]. In contrast, it is argued in section 5.3 that feature mapping (amongst other things) is an automatic by-product of the retrieval mechanism of the model of memory developed in this thesis.

2.7 A Summary of the Relationship of Knowledge Representation to Retrieval

This chapter has described a local network representation for the memory model of this thesis. The network representation relates to retrieval because retrieval is constrained by the structure of the network itself. The metaphorical characteristics of a local network representation means that search paths through the network are in some sense meaningful.

A number of extensions to the representation have been developed and the role that each extension plays is summarised below:

- **The structure link.** This allows for multi-granular knowledge representation. This is important for preserving the flexible in the potential use of a case.
- **The context link.** This connects a relation in the network to a node representing knowledge pertaining to the relevance of that relation. The use of context links and the dynamic selection of contexts allows for strong organisational principles, such as indexing, to be superimposed on memory without introducing any inherent inflexibility in memory usage.
- **The template.** This is a frame-like structure that allows characteristic network patterns to be captured and inherited. Templates allow a principled segregation of the network into descriptive modules, but are intimately embedded in the network so as not to detract from the flexibility of memory usage.
- **Relation weighting.** This augments the rigid structure of memory based on the following considerations.
  - **Typicality:** a statistical measure that can be used to guide the selection of unknown feature values.
  - **Idiosyncrasy:** a statistical measure that highlights unusual features of a specific description and hence can be used to focus feature-based retrieval.
  - **Propagation:** a mechanism that allows the influence of the above two measures to extend across whole paths through the network, rather than being localised to individual relations.

Having established a knowledge representation for the proposed model of memory, the associated retrieval mechanism will be described in chapters 3 and 4. It will be argued that the power of this retrieval mechanism is gained through exploiting the above characteristics of the representation, rather than by relying on more "brute-force" methods.

BANK DESCRIPTION

IMPREGNABLE BANKS, INC. possess a chain of banks. Each bank has a security system with many of the standard features for modern banks, such as; video cameras, door alarms, infra red beams and an alarmed safe. The banks are also manned 24 hours a day by security guards. Unusually, IMPREGNABLE only employ security guards trained in at least one martial art, a policy adopted by only one other banking company. The particular bank at El Passo is even more secure as its manager, one Louis D. Crust, keeps his pet alligator chained in the vault over night.

NETWORK REPRESENTATION

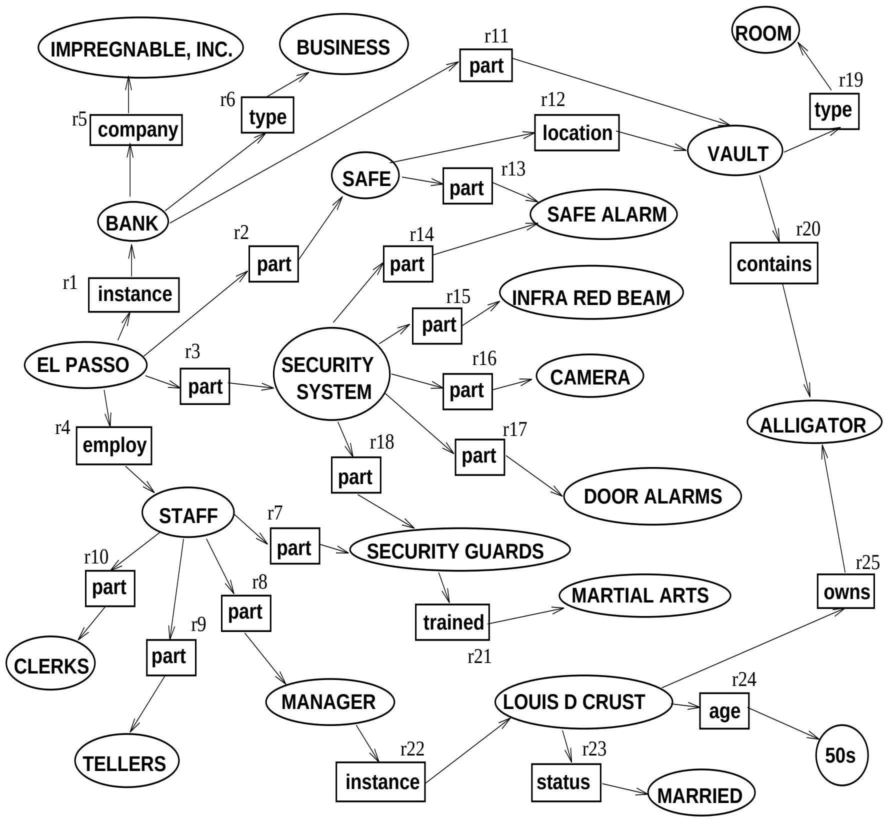


Figure 2.9: An Example for Weight Factor Propagation

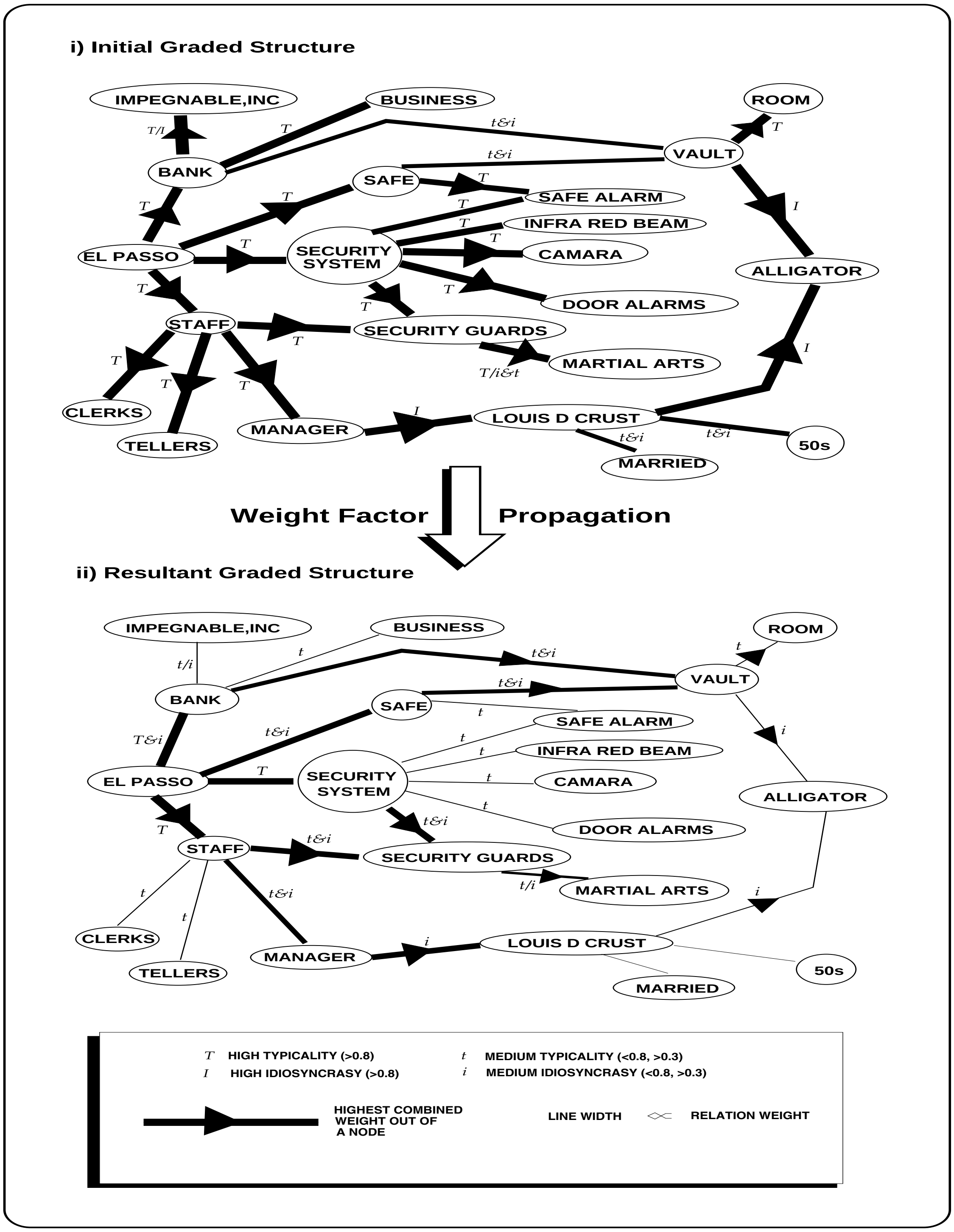


Figure 2.10: The Effect of Weight Factor Propagation on the El Passo Bank Description

Chapter 3

Retrieval and Marker Passing

This chapter is concerned with a discussion of marker passing as a technique to be applied to the problem of performing retrieval from a network model of memory. The notion of a non-deterministic retrieval is introduced. The chapter goes on to argue against the popular belief that massively parallel implementation provides a satisfactory means of solving such retrieval problems. Rather, via a series of illustrative examples, a 'brute force' massively parallel approach alone is exposed as being neither necessary nor sufficient for many types of retrieval. Instead, it is argued that the organisational principles developed thus far in the thesis, specifically the context and relation weighting schemes, have sufficient power to allow many non-deterministic retrievals to be achieved without resorting to an extensive search of memory.

3.1 When is Retrieval Difficult?

Data base technology is such that quite complex queries can be made on very large data bases and the correct information be retrieved with no appreciable delay. Therefore, the question is why should a general model to support case retrieval be at all problematic? The answer is that although recollection is often a straight forward 'look up', true intelligence often involves *creative* retrievals that generate useful recollections for unfamiliar input [117]. Accordingly, it is argued below that many retrievals may require extensive inferencing to be performed in the formation of a valid recollection.

3.1.1 Deterministic Memory Prompts

Retrieval can be thought of as a constrained search from a set of identified knowledge items to a set of required knowledge items held elsewhere in memory. In accordance with this general description, let a memory prompt be defined as the tuple $P = (S, Q, R)$. The set $S = \{s_1, \ldots, s_l\}$ corresponds to the *"seed"* nodes that represent the initially known facts. The *"query set"* $Q = \{q_1, \ldots, q_m\}$ represents the requests for additional information made in the prompt. The set $R = \{r_1, \ldots, r_n\}$ represents the constraints on the retrieval. A valid recollection description is one that assigns a value to all elements of Q and for which all relationships in R hold[1].

A memory retrieval can be considered *deterministic* if the route through memory from the seeds to the queries is fully specified by the constraints. For the knowledge representation put forward in chapter 2 a deterministic memory prompt will give a complete characterisation of the paths through the memory network that needs to be followed. For example in part i) of figure 3.1 the retrieval simply involves finding the **"detected by"** relation out of John Smith then transitively following the attached causal chain. This is the type of retrieval that can be achieved by data base look-up.

3.1.2 Non-deterministic Memory Prompts

For non-deterministic search there is some uncertainty because the memory prompt under-constrains the retrieval search space. Either the set of constraints is incomplete or some of the specified relationships between the seeds of the memory prompt are too *vague* to map in a one-to-one way to relations in memory. For example, in part ii) of figure 3.1, no direct representation of why John Smith was unhappy is encoded in the case description. There are a multitude of reasons why he might be upset. Most people, armed with world knowledge, can make the *inference* that he was unhappy because he had triggered the alarm and that this would lead to his arrest. Indeed, there may be a path in the knowledge base of the form;

> *Being arrested can lead to prosecution. Prosecution can lead to punishment. Punishment generally causes people to be unhappy.*

Finding this information requires an indirect search through a network of world knowledge. If this search is unguided, the search space of possible inferences is vast and it is the navigation of such a search space that is considered in this chapter.

3.2 Marker Passing

Marker passing is a technique that performs search through a network using only local rules contained within the nodes of the network. Reasoning is performed by passing markers from node to node; a process that, if properly controlled, can permit meaningful paths through the network to be extracted [84, 81]. Markers typically carry some rudimentary information ranging from a simple type tag through to a more complex description of the path so far taken by the marker. Generally, on each cycle a node will base the decision on whether to send a marker to each of it's successors on three criteria:

- The markers the node currently possesses.
- The markers its neighbouring nodes currently possess.

[1] The definition of a memory prompt is similar to many formal approaches to question answering, such as [1]. Most of the examples of memory prompt given in this chapter are in fact phrased as questions.

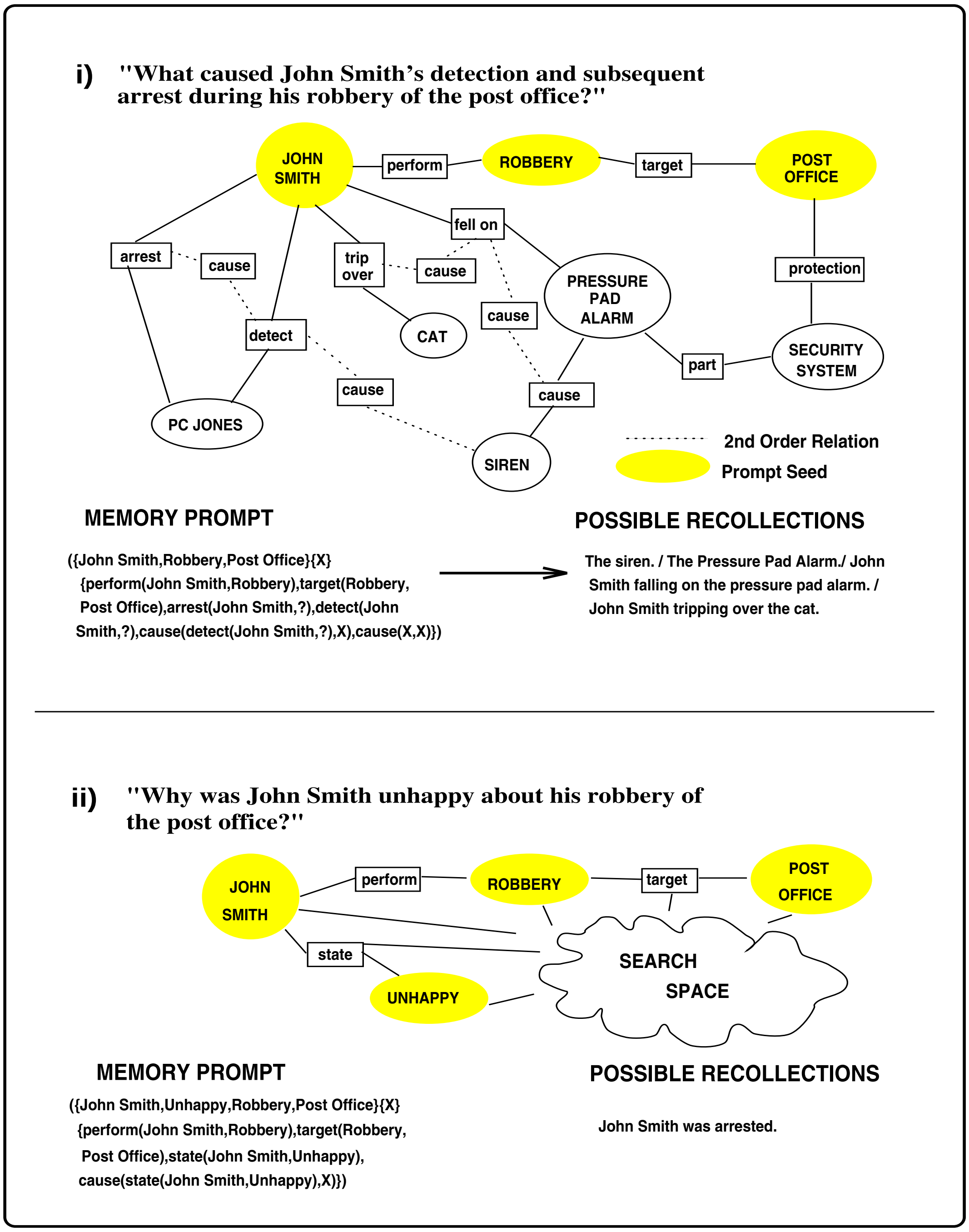


Figure 3.1: i) An Example of a Deterministic Memory Prompt. ii) An Example of a Non-Deterministic Memory Prompt

- Generic global instructions that state which sort of markers should go where.

The remainder of this section explains the appealing properties that marker passing has with respect to the general requirements of retrieval.

3.2.1 Marker Passing for Retrieval

Marker passing can be compared to heuristic search algorithms, such as A* [100, chapter 2] and its derivatives. The performance of heuristic search is governed by an *evaluation function* that estimates the 'distance' of attained states from the desired goal state. One attraction of marker passing is that it is the knowledge embodied in the network structure itself that guides the search. It was stressed that the knowledge representation developed in chapter 2 contained no inherent restrictions in terms of what knowledge is represented and how that knowledge is organised. It follows that employing marker passing to exploit the generality of the proposed network representation leads to the fulfillment of fundamental requirement that retrieval be highly flexible (see page 5).

But what of the efficiency requirement for retrieval? For heuristic search efficiency is governed by the power of the heuristics built into the evaluation function. The main argument of this chapter, presented in section 3.4, is that the various structuring techniques introduced in the preceding chapter provide sufficient constraints to make a number of types of non-deterministic retrieval tractable. The memory structuring techniques are put forward as an alternative to the prevailing approach of assuming a massively parallel implementation for marker passing.

3.2.2 Massively Parallel Marker Passing

The appeal marker passing has for supporters of massively parallel AI is obvious [78]. The expensive serial task of graph traversal is converted into many simple tasks, one per node in the network. Further, the complex data structure required to store the dynamic search tree of nodes and their respective costs is converted into small amounts of local storage within each node. It follows that each node can be thought of as, and implemented by, a simple processing element [84, 42].

Massive parallelism allows the intersection of two large sets of nodes to be performed as a relatively trivial operation taking approximately constant time [42, 41]. This one operation may lie at the heart of a number of reasoning tasks related to the process of retrieval, as will be described in section 3.3. It follows that massive parallelism is commonly seen as a means of satisfying the efficiency requirement of retrieval as it allows for a speed up that is of the order of the number of individual processing elements that are available. A number of anecdotal examples that have been used in the literature to support this claim are presented in section 3.3.2.

A point worth noting is that massive parallelism does not solve the efficiency problem for genuinely intractable (i.e. *NP-complete*) problems. The 2 to 3 orders of magnitude speed-up that is currently plausible using massive parallelism merely delays the point at which an NP-complete problem suffers combinatorial explosion. Conversely, arguments put forward in section 3.4 are that massive parallelism is not *necessary* for tackling the example problems presented in section 3.3.2. For each example a discussion is given as to how the proposed model of memory put forward in this thesis may be sufficient to permit a serial retrieval mechanism to solve the problem in a reasonable amount of time. In addition, in section 3.5 it is argued that the brute force of massive parallelism alone is *insufficient* to tackle the general problem of case retrieval as it does not embody sufficient selectivity.

3.3 Intersecting Very Large Sets

For a general model of feature-based case retrieval, a recollection can be defined as any element of the intersection of search spaces from all seed nodes in the memory prompt. Fahlman [41, chapter 2] identifies the ability to perform rapid set intersection as the major characteristic of tasks that are readily achieved in human intelligence yet highly difficult for serial AI applications. Such tasks include:

- Type inheritance.
- Anomaly detection.
- Instance classification.
- Nested context maintenance.
- Causal chain inheritance and assumption-based reasoning.

The general scheme for set intersection by parallel marker passing is illustrated below.

3.3.1 Arguments for Massively Parallel Set Intersection

On average the serial intersection of two sets takes a time proportional to the sum of the set sizes. For massive parallelism, provided that there are a sufficient number of processing elements available, the time is effectively a constant. To illustrate this, consider the task of answering the question *"Is Daisy warm blooded?"* where Daisy is a Jersey cow represented in a knowledge base including extensive taxonomic knowledge. The type of solution adopted in systems, such as NetL [41], is shown in figure 3.2.

The key assumption of massively parallel marker passing is that traversing all relations of a specified type out of all nodes possessing a selected marker type takes only one unit of time[2]. In particular only in step 4 of the example (figure 3.2), when markers are propagated up a type hierarchy, is more than one time step required. More generally, parallel implementation takes a time proportional to the longest path in the search space. A number of examples of this type of massively parallel information retrieval, taken from the litertaure, are presented below.

[2]This assumption has proven valid for Single-Instruction Multiple-Data (SIMD) machines [78].

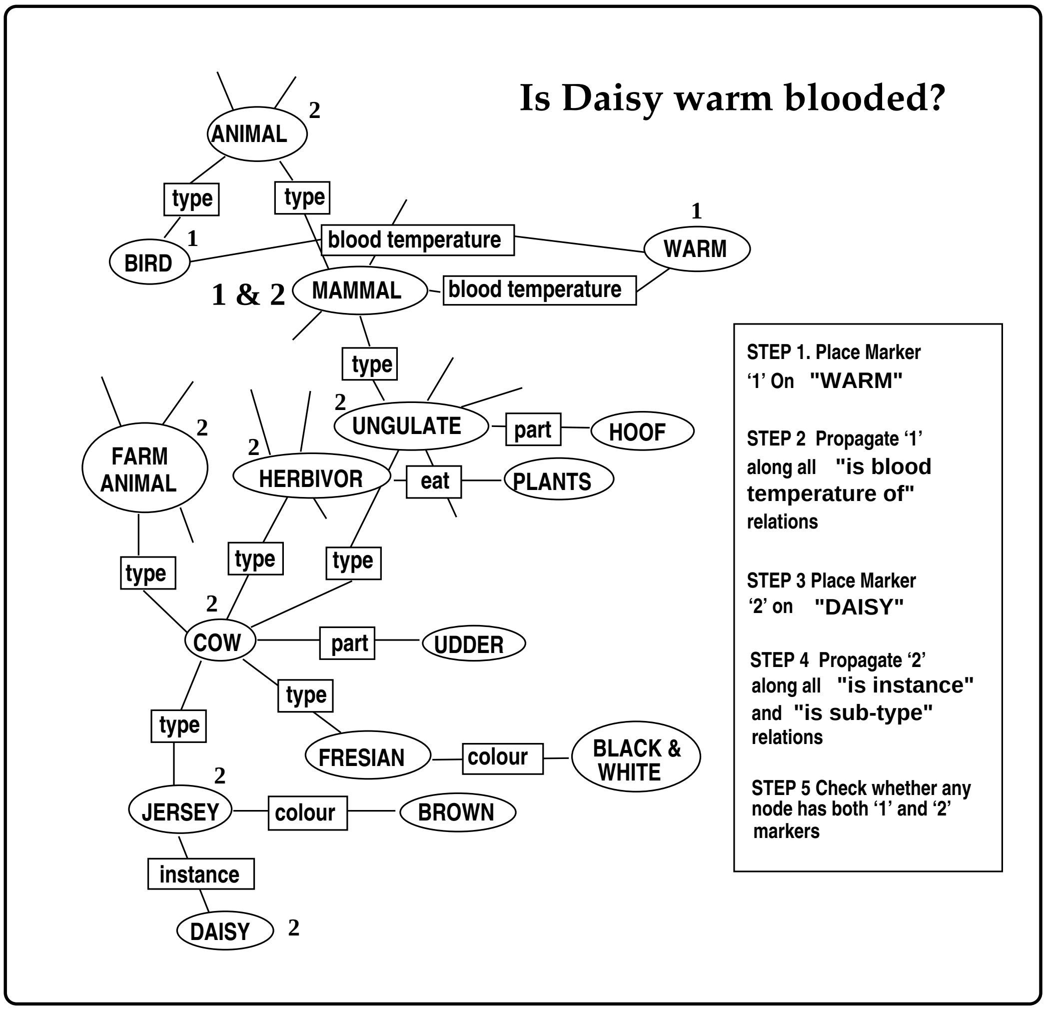


Figure 3.2: An Illustration of the Use of Massively Parallel Marker Passing to Perform Question Answering

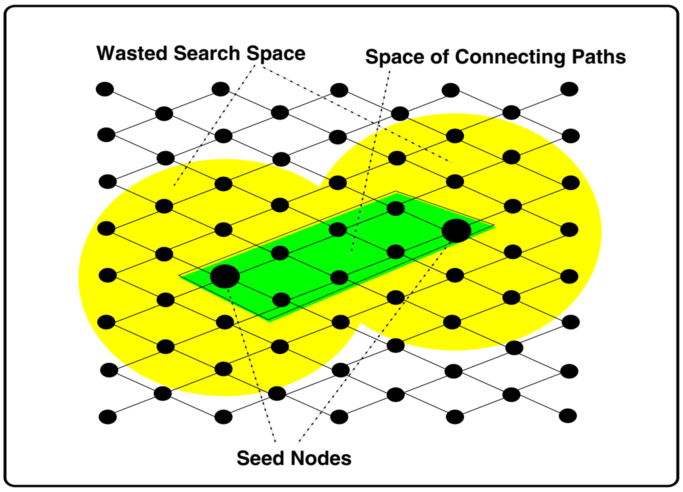


Figure 3.3: The Search Space for Connecting Path Inferences

3.3.2 Some Examples

An example taken from [41] that is similar to the above is *"does any royal person own a weapon?"*. In this case the query is answered by traversing down the type hierarchy to find all instances of royalty and all instances of known weapons. Finally, the 'owns' relations between instances of the two resultant sets are traversed. The time needed to perform this search is low (proportional to the sum of the type hierarchy depths) despite the fact that a large area of the knowledge base may receive markers.

Parallel marker passing can also be used to make additional or connecting inferences for incomplete memory prompt descriptions. This relies on a form of assumption-based reasoning which is similar to the process of CBR. In particular, the task of generating additional inferences has similar requirements to those of case retrieval. The known features from which the assumption-based inference stems will form the seeds of the memory prompt and the query set for the prompt establishes a body of knowledge that should be inferred through the retrieval.

An example of assumption-based inference used in [41, page 25] is *"John wanted to be chairman of the department. He bought some arsenic"*. The claim is that the implicit connecting theme between the two sentences is *"assassination"*. Similarly Charniak [30] uses the example of *"John wanted to commit suicide", "John got a rope."*. Here the most obvious implication is that John is going to hang himself.

Both the above example inferences may be made using exhaustive marker passing to perform set intersection. Generating the intersection of all uses of arsenic and all possible ways of gaining a promotion might indeed intersect at a node representing *"assassination"*, provided that the network contains the relevant world knowledge. Similarly *"hanging"* may be an intersection of all possible uses of rope with all possible methods of suicide.

In general what is required is a path that can be generated between two vaguely[3] related seeds in a memory prompt. The path may involve several intermediate nodes[4]. Marker passing is governed by the consideration of immediate links only. It follows that the formation of a path that includes a number of intermediate nodes is prone to involving a large and wasteful search space, as shown schematically in figure 3.3. Two distinct ways in which the knowledge representation of chapter 2 can help reduce this wasted search effort are discussed below.

3.4 An Alternative Memory-Based Approach to Set Intersection

In section 3.4.1 it is argued that the context mechanism introduced in section 2.4 can be used to restructure memory to make partially explicit certain set intersections that have proved useful. In addition, it is argued in section 3.4.2 that the relation weighting scheme described in section 2.6 plays an important role in the initial creation of paths to solve the type of retrieval problem illustrated in the previous section.

3.4.1 Context Links as a Means of Strengthening Useful Memory Paths

A direct way of avoiding exhaustive search for a set intersection is to represent the intersection explicitly. This is similar to the generation of a sub-class representing the specialisation of a concept. For example, a specialisation of the concept *"weapons"* would be *"weapons that are also owned by a royal person"*. The representation of such a concept would make the example described on page 42 a trivial problem even for serial implementations. However, there are an unlimited number of such *"ad hoc"* categories [9] so this solution can only be applied to intersections that have proved to be of particular importance.

The representation scheme developed in chapter 2 provides a more subtle way of making an explicit segregation amongst the members of a set through the use of the context link. Recall that a context link connects a node representing knowledge about the validity of an individual relationship to the appropriate relation node. For example knowing that *"suicide"* is a currently important concept signifies that, unlike many other uses of rope, forming a noose is currently significant. This focusing effect may be represented using the context link notation, as shown in figure 3.4. The circumstances under which it is valid to restructure memory in this way are considered below.

The general role context plays is to prune irrelevant paths from a search space, as schematically shown in figure 3.5. Adding a more specialised context to a relation can make retrieval more directed. However, for a path involving n relations $n-1$ context

[3] Vague in the sense described in section 3.1.

[4] Paths that involve no intermediate nodes will be referred to as *trivial* because it can be assumed that finding a connecting relation between two directly linked nodes requires negligible search.

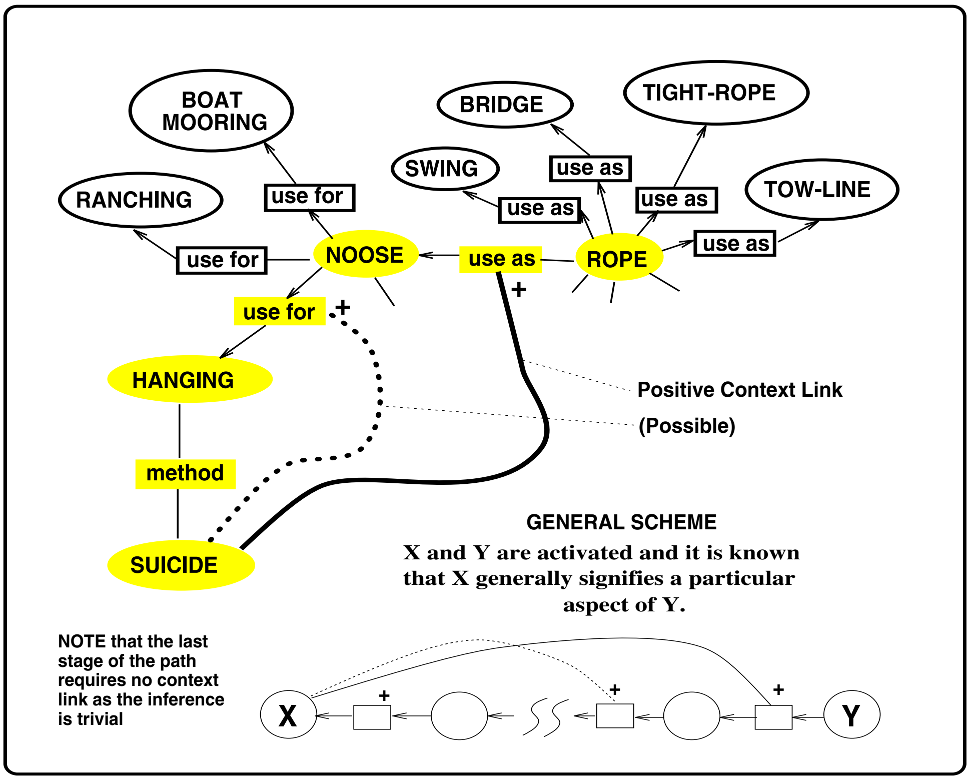


Figure 3.4: Contextual Strengthening of Intersection Paths

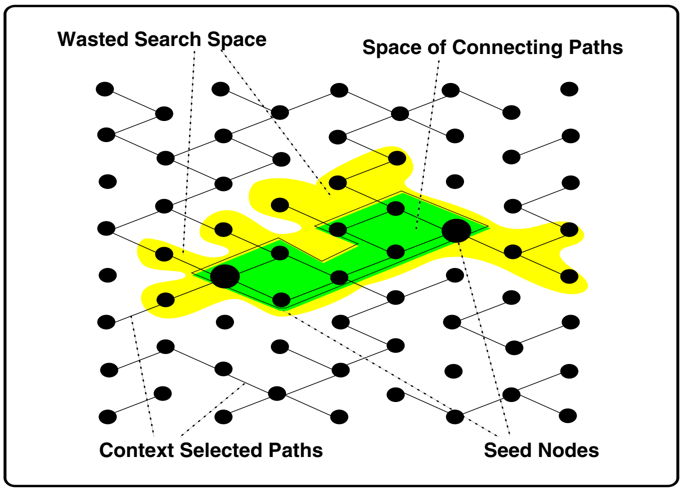


Figure 3.5: Context Prunes Irrelevant Relations from the Search Space (c.f. Figure 3.3)

links are required to allow it to be traversed without irrelevant branching. To represent all contexts for all valid paths would lead to a memory usage explosion similar to that for explicit representation of all set intersections. However, rather than expanding the search space defined by memory, context links act as an extra constraining dimension that restricts the paths through memory. In addition, as discussed below, not all the context links may need to be created.

Consider again the rope and suicide example depicted in figure 3.4. Going out from **"ROPE"** there may be a great number of possible uses stored in memory. It follows that the insertion of a context link from **"SUICIDE"** to select **"NOOSE"** as the relevant use of **"ROPE"** greatly reduces the proportion of memory that must be searched. Conversely, once **"NOOSE"** has been reached there may only be a small number of possible uses represented in memory. It follows that the insertion of a context link to select **"HANGING"** as the relevant use when **"SUICIDE"** is a context makes a relatively small reduction in the search space required to complete the path from **"ROPE"** to **"SUICIDE"**.

There is a general trade off between the number of context links used (space efficiency) and the directness of the search patterns they produce in memory (computation efficiency). Let the path from Y to X, $Path(X,Y) = (Y, rel_1, \ldots, rel_n, X)$, have a factor $Utility(Path(X,Y))$ associated with it. $Utility(Path(X,Y))$ is a measure of how important is it to traverse $Path(X,Y)$ given that X and/or Y are known; generally this can only be determined empirically from previously successful retrievals. For a stage of the path, rel_i, let there be a factor $Selectability(rel_i)$ that represents the *a priori* probability that the relation will be selected from all relations out of the relevant source node[5]. It follows that given limited space and time resources, a decision to create a new context

[5]For the selection mechanism described in chapter 5, $Selectability(rel_i)$ is proportional to the weight of rel_i and inversely proportional to the number of relations of the same type that lead outwards from the source node.

link in order to direct the selection of rel_i should be made according to the following rule:

$$CreateLink(X, rel_i) = \begin{cases} TRUE & \text{if } \frac{Utility(Path(X,Y))}{Selectability(rel_i)} > \eta \\ FALSE & \text{otherwise} \end{cases} \tag{3.1}$$

where η is some threshold constant. In other words, the creation of context links to overcome set intersection problems associated with memory retrieval should be a process that is driven by the actual use of memory.

3.4.2 Relation Weighting as an Evaluation Function in the Search for Useful Information

A path can only gain a utility factor after being found and so the strengthening of paths by the insertion of context links does not explain how paths can be initially selected. The relation weighting of section 2.6 can greatly aid in this process.

Consider again the arsenic example of section 3.3. Intuitively the use of arsenic as a poison is the one people recognise as the most *typical*. Relation weighting can capture this typicality provided that the knowledge base has a representative collection of experiences involving the use of arsenic. Hence the typicality measure could be used to select the use of arsenic as a poison by default. Once this step has been made the inference of assassinating the boss becomes very much easier.

A similar argument can be given for the suicide example. There are undoubtedly a large number of ways of committing suicide. A knowledge base with a macabre domain of application might represent many of these methods. Again the assumption is that the system has a representative set of instances of suicides from which it can derive meaningful relation weightings. If this is so then *"hanging"* may be assigned a high typicality as a method of suicide. Figure 3.6 shows how typicality might be used to guide the creation of a connecting path in the suicide example.

Typicality guides knowledge along the paths that represent the greatest proportion of the real world. The usefulness of typicality is therefore dependent on the inductive principle that what has been most prevalent in the past is most likely to recur in the future. Typicality is no aid to retrieval involving the connection of atypical aspects within the prompt description. Violations of expectation play a central role in prompting retrieval for failure-driven models of memory epitomised in [116, 117]. Fortunately the complementary relation weighting factor of idiosyncrasy provides guidance towards exceptional information.

Consider again the problem of determining why John was unhappy while robbing the post office, (figure 3.1). The fact that he was arrested and the preceding causally related incidents are the unusual aspects of the case *with respect to all other robbery cases*. Idiosyncrasy will tend to direct an otherwise unguided search to these unusual aspects hence biasing retrieval to consider expectation violation. Once the concept of being arrested is focussed upon the search to determine why John was upset is made more straightforward.

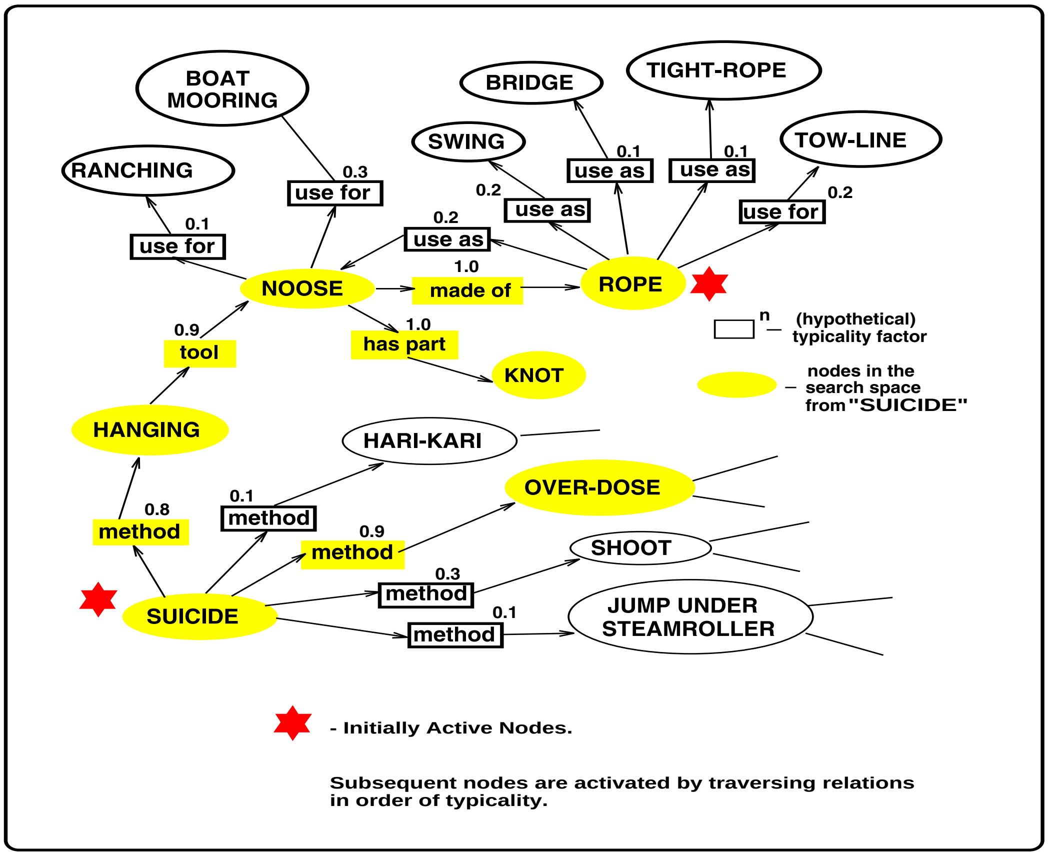


Figure 3.6: Typicality as a Heuristic Guide to Path Generation

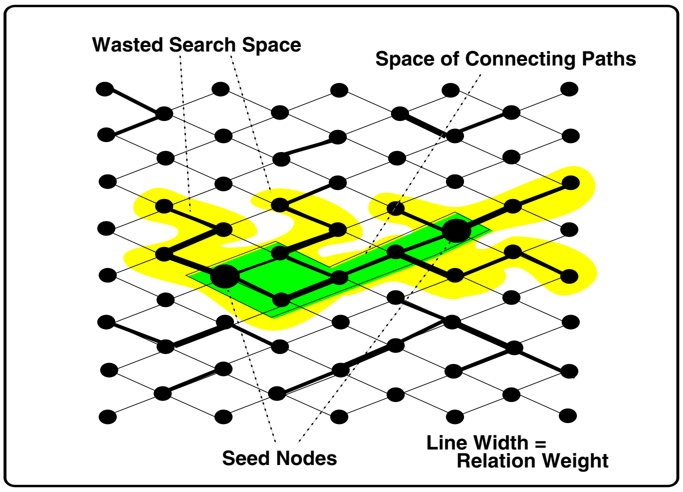


Figure 3.7: Relation Weighting Prioritises Paths Through Memory. (c.f. figure 3.3)

More generally, the use of idiosyncrasy to guide retrieval in order to explain an expectation violation is based on a similar assumption to the inductive principle embodied in typicality. The assumption is that, if a specific experience (i.e. case) includes in its description a number of unusual characteristics, frequently a single root cause will explain all the anomalies.

The conclusion from this section is that a bias towards typical *or* unusual information can indeed often aid in the search of paths from a memory prompt. In effect the typicality and idiosyncrasy weighting factors act as heuristics of an in-built evaluation function which estimates the likelihood that paths through memory will lead to useful information. The resultant graded structure of memory, stylised in figure 3.7, can be exploited to produce a heuristic retrieval strategy as a viable alternative to massively parallel marker passing. This suggests that massive parallelism may often be *unnecessary* in order to perform efficient and selective retrieval. The criticism is further strengthened in the next section where it is argued that exhaustive search may be *insufficient* for some memory retrievals.

3.5 The "Embarrassment of Riches" Problem

Many memory prompts, deterministic and non-deterministic, are open-ended in that the constraints given in the memory prompt can be satisfied by a great number of items in memory. A good example is the problem of recalling a member of the royalty that owns a weapon, given on page 42. Historically, many members of the aristocracy have owned weapons, either for defensive or recreational purposes. It follows that satisfying this memory prompt by massively parallel marker passing in a knowledge base containing a large body of historical knowledge is likely to cause many instances to be retrieved indiscriminately.

An external system does not necessarily benefit by being provided with a large and exhaustive set of all solutions to a memory prompt. Given such an abundance of information the external system may have to either select a few of the many recollections at random, or more seriously, do a secondary search to select which recollections are of most interest.

The danger of retrieving too many cases is particularly pertinent for CBR. The PARADYME system [79] is representative of the problem. Although PARADYME uses a massively parallel approach to retrieve all partially-matching cases, it's success relies heavily on the subsequent selection of retrieved cases based on a variety of preference heuristics. Similarly the MAC/FAC system [45, 48] employs a two stage process whereby the first stage involves a cheap and consequently inaccurate measure for relevance which acts as a filter on the cases in memory prior to employing the second and more rigorous stage for final selection of cases.

Exhaustive marker passing alone does not provide the selectivity required to retrieve a small corpus of pertinent cases. As has been illustrated in this chapter, the proposed memory model is designed such that as many selective constraints as possible can be employed as early as possible in the retrieval process. The goal is to minimise the amount of wasted effort that is considered unavoidable in systems such as PARADYME and MAC/FAC. The next chapter furthers the discussion on marker passing by describing the proposed mechanism for allowing selective constraints to influence retrieval in an intimate way.

3.6 Concluding Comments on Memory Organisation versus Massive Parallelism

Massively parallel marker passing is one technique for rapidly traversing the large search spaces involved in many KBS retrievals. Conversely, the goal of this thesis is to produce a rich enough structuring of memory to permit tractable serial retrieval.

Producing a fine-grained partitioning of the network using context links can drastically reduce the size of the search spaces. However the placing of context links can only be achieved manually[6] or by some process that monitors the history of previously useful memory retrievals.

Heuristics are generally used in AI in order to conquer intractable search spaces. The measures of typicality and idiosyncrasy, it has been argued, are general heuristics operating within the domain of knowledge that can be used to make retrieval tractable. It is important to clarify the limits of the usefulness of these measures.

3.6.1 Limitations of the Memory Organisation Approach

Some memory prompts will inevitably require neither highly typical nor highly unusual features. Trying to determine whether Daisy is warm blooded, (figure 3.2), is a good example of this. The main task of the search is the upward traversal of the type hierarchy

[6] As is done for the example knowledge bases referred to in chapter 6.

from the **"COW"** node. All superordinate classes for **"COW"** are equally typical because all instances of **"COW"** must be instances of any given superordinate. Similarly it is unlikely that any superordinate is particularly idiosyncratic as each superordinate will have several sub-class descendants other than **"COW"**. The relation weighting scheme given in the previous chapter is therefore no help for this particular search. Ultimately it has to be conceded that massively parallel implementations may be the only way to successfully achieve some retrievals.

Nevertheless a memory organisation approach for flexible, efficient and selective retrieval may be well suited to case retrieval. Firstly, the basic framework for CBR is tolerant to initial failure of retrieval. If non-exhaustive retrieval search fails to find an adequate source case, this will be detected by the evaluation phase of CBR and appropriate action (such as a re-retrieval) taken. Hence it is not critical that case retrieval always produces an optimum solution to the constraints of a memory prompt.

Secondly, the utility of the relation weighting factors has some relevance to a generalised model of case retrieval. It has already been argued (pages 30 to 30) that idiosyncrasy is useful in focusing the type of feature-based retrieval generally performed from a target case. In addition, typicality will be a useful criteria by which to select which knowledge to transfer from a retrieved source case to the original target case.

Chapter 4

Activation Passing

This chapter describes a novel activation passing mechanism that is designed for the requirements of case retrieval. Activation passing is chosen as the retrieval mechanism as it allows a quantitative assessment of preference constraints to be carried out during retrieval itself. However, traditional approaches to activation passing are not sufficient for the task of case retrieval. In particular, this chapter provides three main contributions: a means of representing a 'fuzzy' disjunction of simultaneous activation distributions; a method for preserving target case structure during retrieval by activation passing and a novel form of competitive activation propagation that allows a corpus of several pertinent cases to be retrieved. The resultant system is a hybrid one that intimately conjoins symbolic and numeric reasoning. This is seen as an improvement on a number of similar approaches.

In the preceding chapter it was argued that some form of marker passing is suitable as the required retrieval mechanism. In this chapter the discussion is taken one stage further by analysing the usefulness of passing simple numeric markers, or *activation*. Accordingly, this chapter starts with a general description of what is meant by activation passing and an explanation of why it is particularly well suited for application to the task of case retrieval.

4.1 Activation Passing for Retrieval

In chapter 3 retrieval was described as a process that satisfies the constraints imposed by a memory prompt. More specifically, marker passing is a suitable technique because the search through memory it implements can be guided by the constraining factors built into the structure of memory itself.

The act of searching memory by the propagation of markers is a form of symbolic reasoning. However, finding paths to potentially applicable cases only partially satisfies the requirements of case retrieval laid down in section 1.2.4. In addition, the cases that are ultimately retrieved from memory should be selectively graded in terms of their 'similarity' to the target case that prompted their retrieval. In order to achieve this a *hybrid* retrieval mechanism is required that combines the symbolic characteristics of memory search with an appropriate quantitative evaluation of the combined effect of all in-built memory constraints.

Activation passing is appropriate because it transports simple numeric values through memory. Arithmetic calculations, built into the marker passing operations of the nodes of memory, can be designed such that the amount of activation possessed by a particular node is indicative of the current importance of that node [81]. It is the design of the low level mechanics of activation passing, such that the process is suited to case retrieval, that is the main focus of this chapter. Prior to the detailed discussion, an introduction to the basic activation passing mechanism is given in section 4.1.1 and an overview of the proposed retrieval process given in section 4.1.2.

4.1.1 General Description of Activation Passing

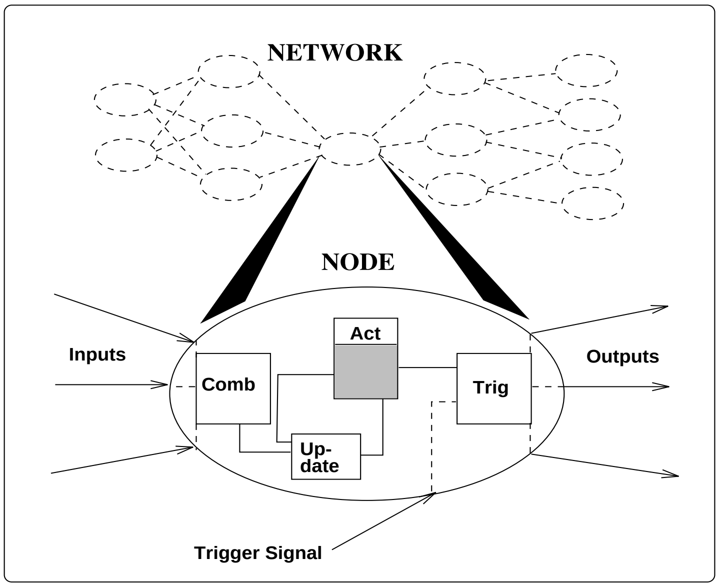


Figure 4.1: A General Activation Passing Node Unit

A general illustration of the basic process of activation passing is given in figure 4.1. Each node, n, has a set of inputs, $I(n) = \{I_{n,i}\}$, along which activation arrives periodically. The incoming activation is combined by some function $Comb(I(n))$ and is used to update $Act(n)$, the current amount of activation possessed by the node. Finally, there is a trigger function, $Trig(n)$, possibly associated with a specific input, that will cause the node to pass a proportion of it's stored activation to it's outputs, $O(n) = \{O_{n,i}\}$, and hence on to other nodes.

The update function, $F(Act(n), Comb(I(n)))$ is of great importance. Ordinarily it is a second-order function [126] causing a node's activation to asymptotically approach maximal/minimal limits on repeated receipt of activation. This has two consequences:

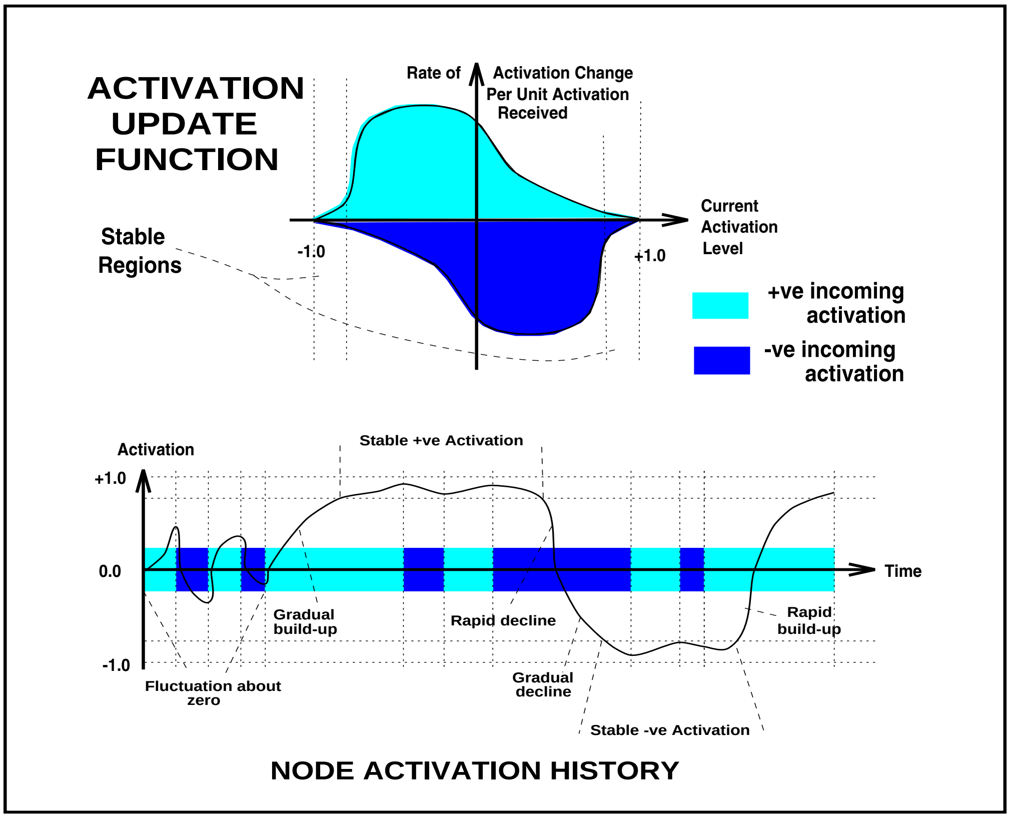


Figure 4.2: The Characteristics of Node Activation Change in Response to New Activation

the activation levels of nodes tend to become stable near the maximum magnitudes which allows the network as a whole to settle into a stable state and the nodes possessing the most activation do not necessarily dominate the activation passing process. This relates to the needs of feature-based case retrieval as it is generally important that a characteristic combination of features, rather than any one isolated feature, be the determining criterion for source case selection.

An asymmetric update function is employed here (see figure 4.2) which produces a greater change in $Act(n)$ for receipt of incoming activation of opposing polarity than for incoming activation of the same polarity. This tends to keep the node's activation at near zero if activation of both polarities is received. This means that to a first approximation there are three states for the amount of activation that nodes possess; near zero, near maximum and near minimum. This can be seen in the annotated history of figure 4.2. Nevertheless, within the bounds of these fuzzy distinctions there can still be variation in the amount of activation possessed by each node and hence it is still possible to produce an appropriate gradation of cases during retrieval.

4.1.2 A Three-Phase Activation-Passing Retrieval Mechanism

The retrieval mechanism to be detailed in this chapter can be viewed at the top level as involving three distinct phases, as shown in figure 4.3. These three phases are:

- **Phase 1:** The target case description has activation placed on all of its component nodes[1].
- **Phase 2:** Activation is propagated from each individual target case node, through the network representing world knowledge, to similar component nodes of source case descriptions.
- **Phase 3:** Activation is propagated upwards in each source case description such that the collection of individual component retrievals is amalgamated to produce a resultant residual activation in its root node.

4.2 General Problems with Activation Passing and Some Solutions

There exist some serious difficulties when attempting to employ activation passing to the task of case retrieval, caused by the local nature and low knowledge content of activation passing rules. Firstly, the retrievals from individual target case features need to be distinguished from each other yet, by the arguments of chapter 3, important intersections of retrieval paths should be detected. Secondly, there is the problem of implementing a bias towards preserving the overall descriptive structure of the target case in the retrieval

[1]Note that it is assumed that cases are represented as a hierarchical decomposition.

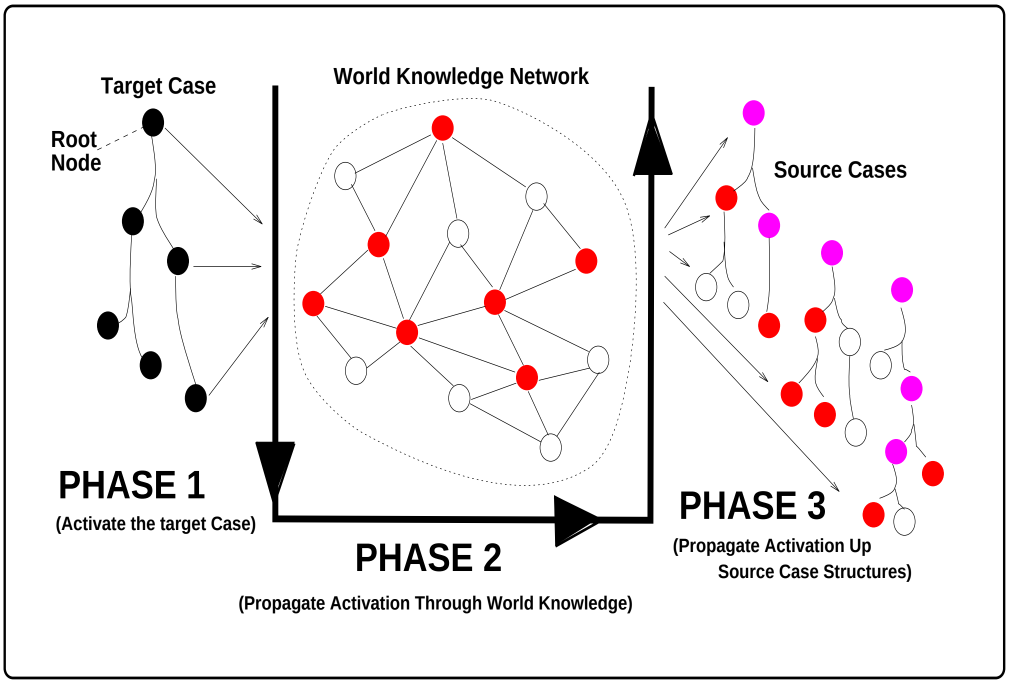


Figure 4.3: A Three-Phase Model of Retrieval

of source cases. In this section these problems will be explained and appropriate solutions put forward.

4.2.1 Implementing Disjunctive Activation Passing

During phase 2 of retrieval it can be expected that more than one feature of a target case description will be used simultaneously as a retrieval prompt. Further, several possible paths from each feature may be followed. For activation passing the problem of *cross-talk* arises if the sections from two disjoint retrieval paths are interchangeable. This can lead to ambiguous or erroneous retrieval of knowledge, as shown in figure 4.4.

Interference is a phenomenon similar to cross-talk but involving the meeting of two independent retrieval paths at a single node. For example, in figure 4.5 activation of opposing polarities reaches **"GUN"** from two separate features of a robbery case description. As described in section 4.1.1, the resultant net activation on **"GUN"** is likely be of low magnitude (i.e. destructive interference) in which case retrieval paths from two important aspects of the robbery problem will have been prematurely terminated.

The problems arise from not being able to distinguish between activation from different sources. One way of avoiding the problem is to perform activation passing sequentially [37] such that the activation distribution from at most one target case feature is present in memory at any one time. However, In chapter 3 it was shown that it is often the intersection of two independent retrieval paths that is required. Path intersection may also be vital in order to determine intermediate knowledge during a complex retrieval, such as in the *anticipation* of potential planning problems [58]. Specifically,

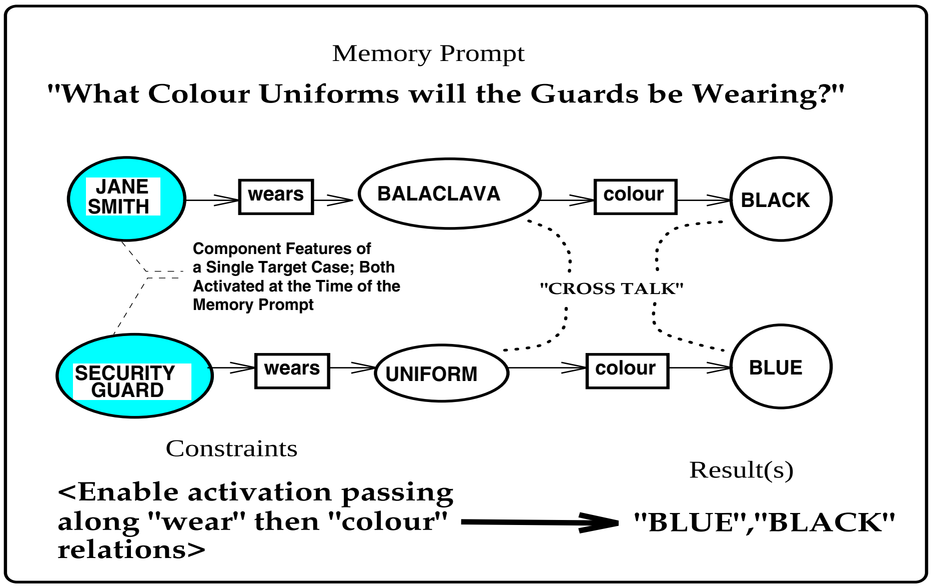


Figure 4.4: An Example of Activation Cross-Talk

for the illustration of figure 4.5, the relevant problem is that of being faced with armed guards *and* not having equivalent fire power. The *combination* of these features may suggest avoiding rather than over-powering the guards.

Disjunction by Colour Tags

For disjoint activation distributions to exist simultaneously each must in some way possess its own unique 'copy' of the network. The most efficient way this can be achieved is to add a tag that uniquely identifies to which distribution an activation marker belongs. This approach has been used frequently in marker passing systems [41, 96, 31, 81]. The *colour* tagging solution advocated in this thesis is described below.

Activation is assumed to proceed via a series of constrained steps. The problem of cross-talk occurs if, at a single step in the activation propagation, two nodes send activation yet earlier in the history of the propagation the nodes had received their activation at different steps (signifying that the nodes represent differing types of knowledge with respect to the constraints of the retrieval). Similarly, interference occurs if two of the nodes that received activation in the current step had already received activation at an earlier step in the activation propagation. A more thorough definition of cross-talk and interference in terms of the basic characteristics of an activation propagation history is given in section H.1 of appendix H.

The definition of cross-talk and interference in terms of the meta-level properties of an activation distribution can be used to support a general procedure for the detection of ambiguity problems in activation propagation. For example, in figure 4.4 **"JANE SMITH"** presumably received activation at a different propagation step to **"SECURITY GUARD"** and hence the potential for cross-talk is detectable. Similarly, in

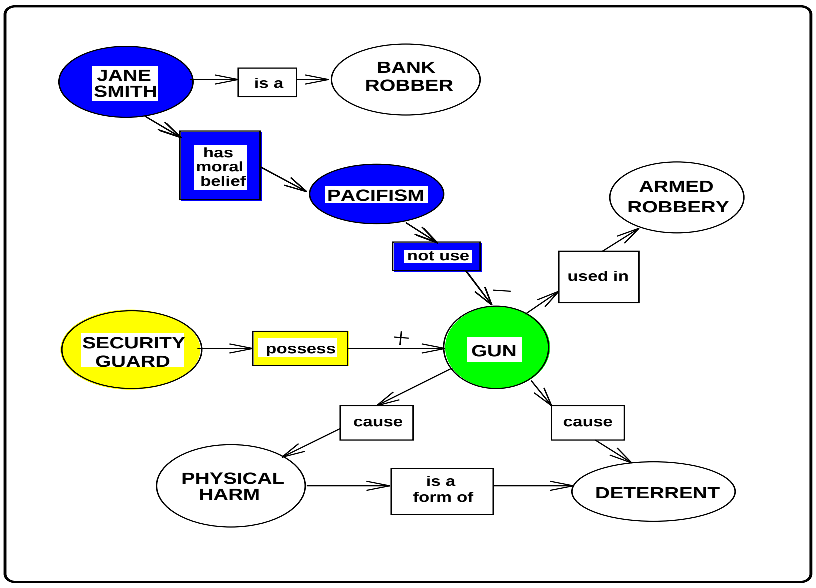


Figure 4.5: An Example of Activation Interference

figure 4.5 the activation propagation to **"GUN"** from the two distinct sources will arrive at different steps in a constrained activation propagation and hence the potential for interference is detectable.

Generally, whenever a cross-talk or interference problem is detected, it can be avoided by changing the activation colour of one of the conflicting sources of activation. An in depth example of using activation colours in this way is given in appendix I. As the propagation continues a hierarchy of activation colours will be generated, rooted at the original colour of the activation distribution with sibling colours being generated from a parent colour whenever cross-talk or interference needs to be avoided. The data structure that holds the information about how activation colours interrelate will be referred to as the *colour dependency hierarchy*.

The colour dependency hierarchy is a global data structure for activation passing and its use detracts from the localised nature of the propagation process. However, this relatively simple global data structure provides significant enhancements to the reasoning power of activation passing.

In many purely distributed approaches to marker passing each node of the network has a unique signature which is used as the tag that identifies the single activation distribution originated at that node [81, 96]. In contrast, by holding the activation distribution tags as a separate data structure to the network itself it becomes possible for multiple activation distributions to be rooted at the same node. In addition, as discussed below, by preserving a notion of dependency between the different activation colours it is possible to permit a form of *constructive interference* to be implemented where disjoint activation distributions over-lap. The notion of colour dependency will also be seen later (section 4.2.2) to allow for the activation passing mechanism to preserve the structure of a target case during retrieval.

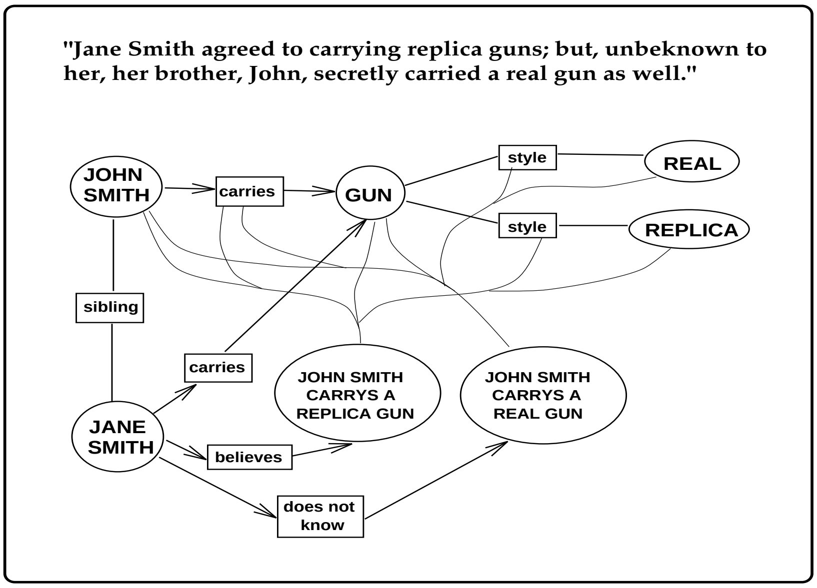


Figure 4.6: An Example of the Need for Constructive Interference

As for sequential activation propagation, the use of separate activation colours to avoid interference seems to contradict one of the central requirements of marker passing retrieval; that the intersection between separate retrieval paths be detected. An example is shown in figure 4.6. In this example it can be expected that the node **"GUN"** receives activation of at least three different colours. Precisely because **"GUN"** is multiply referenced it should be considered the most central and important concept of the example.

A function needs to be defined to characterise how the differently coloured activation distributions interact to implement a 'fuzzy' form of conjunction. Assume a node n already possesses activation, $Act_{c_a}(n)$, of colour c_a. Suppose n receives incoming activation, $Comb_{c_b}(I(n))$, of colour c_b. Let $Dist(c_a, c_b)$ be a measure of the distance between two colours measured in the descendancy/ancestry direction of the colour dependency hierarchy[2]. This receipt of new activation, in addition to causing activation of colour c_b to be stored in n, also causes an amount of activation of colour c_a to be received due to *constructive interference*. The magnitude of this interference-generated activation is given by:

$$InterAct(Comb_{c_b}(I(n)), c_a) = \begin{cases} Comb_{c_b}(I(n)) * (\beta \frac{1}{1+\lambda Dist(c_a, c_b)}) & \text{if} \quad Dist(c_a, c_b) \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

Assuming $\lambda, \beta > 0$, this equation represents a decrease in the strength of interaction between differently coloured activations with increase in topological distance of the colours. In addition, sibling colours, generated to represent disjunction, appropriately

[2]The terms "ancestor" and "descendant" are used in a similar sense to their common usage when referring to topologies. Referring to the example shown in figure I.1 of appendix I, $Dist(c_3, c_4) = 0$, $Dist(c_3, c_2) = Dist(c_2, c_3) = 1$ and $Dist(c_1, c_3) = 2$.

have zero interaction.

4.2.2 Preservation of Structure in Activation Passing

Generally, the description of a knowledge item is more than a list of features, the description possesses a structure. Seeking source cases with a similar descriptive structure to that of a target case is problematic for retrieval by activation propagation. The reason for this is that the localised rules that govern activation passing are not well suited to performing calculations that involve consideration of large areas of the network. In this section the solution to the problem is related to activation passing within the structure link network primitive, introduced in section 2.3.

In general, the representation of a specific case will involve the instantiation of many templates where each template provides a description of a specific component of the case. In this sense cases are represented in a distributed manner and it is the primary role of structure links to tie together the separate component descriptions.

Activation within a structure link is designed to preserve case structure during retrieval in two ways. Firstly, a probablistic measure is defined for the inferential strength from an incomplete set of detail nodes to an abstract node. This provides a general bias towards source case retrieval across the breadth of the case description as opposed to on isolated features. Secondly, a scheme for a more fine-grained preservation of target case structure is considered. The scheme relies on the use of activation colour dependencies during the recombination of activation during phase 3 of retrieval, hence permitting the more fine-grained structure of a target case to be preserved.

Activation Passage within a Structure Link

There generally exist a number of *covariance* relationships for the description of some knowledge item that reflects the fact that certain sets of features have a high probability of occurring together. This relates to the role of a structure link in case representation. A structure link represents the fact that for a particular case a set of specific components or features *did* co-occur as a description of a higher level component; the structure link has the semantics of a many-to-one equivalence.

A problem arises because, for the retrieval scheme outlined in figure 4.3, phase 2 will generally be a non-exhaustive search of memory (in accordance with the need for retrieval efficiency). It follows that only a fraction of reachable nodes within the descriptions of source cases will receive activation. Accordingly, during phase 3 of retrieval, when activation is propagated up the structure link hierarchies of the source cases, there is a need to take account of the *proportion* of detail nodes that possess activation (i.e are known) for each structure link.

The requirement is to assess the strength of inference of an abstract node given only a partial set of its associated detail nodes. The problem is similar to that tackled in the work by Russell on determining the strength of validity of an analogical inference [114]. He argues that the strength of inference should drop exponentially as the proportion of unknown details increases, and this claim is supported by empirical data. Intuitively, a similar decay function is required for the strength of inference in a structure link. However there are some subtle differences with respect to the problem tackled by Russell. In particular, Russell assumes that not all features of a description are relevant for the generation of a correct analogical inference where as for a structure link all detail nodes can be assumed to be relevant. An appropriate measure of inferential strength is developed below.

For a structure link sl let the collection of associated detail nodes be $Dets(sl)$. Assume there are N_a structure links for which the set of detail nodes overlap with $Dets(sl)$. These structure links constitute the set of all possible structure links that are confusable with sl. Assume that between them these abstract nodes have a total of d_{tot} different detail nodes and, on average, each structure link involves d_{ave} detail nodes. For sl, let there currently be a proportion of p missing details[3]. It is shown in section H.2 of appendix H that the inferential strength from the known detail nodes to the abstract node of a structure link is:

$$IS(Dets(sl)) = \frac{1}{N_a \gamma^{(1-p)f} + 1} \tag{4.2}$$

Where $\gamma = \frac{d_{ave}}{d_{tot}}$, and $0 < \gamma \leq 1$. The function is represented graphically in figure 4.7[4]. It is used to govern activation passage up a structure link according to the following equation:

$$InAct(abst) = \eta IS(Dets(sl)) \times (Mean(TotAct(Dets(sl))) - Act(abst)) \tag{4.3}$$

where η is the *upward spread coefficient* and is a real number parameter that can be set to determine the amount of activation magnification that occurs during activation propagation up a structure link hierarchy (see section 6.5). From the graph shown in figure 4.7 it can be seen that $IS(Dets(sl))$, and hence the amount of activation passed to a structure link's abstract node, reduces as the proportion of missing details increases. This means that during phase 3 of the proposed retrieval scheme source cases that have many features activated during phase 2 tend to be favoured over those that have a few activated, irrespective of the amount of activation actually received.

Fine-Grained Preservation of Structure by Activation Passing

Measuring the upward inferential strength for a structure link with missing details provides only a crude preservation of the structure of a description during retrieval. It takes into account the number of single feature retrievals made between a source and target case during a retrieval but does not consider fine-grained structural isomorphism. That is to say it does not account for subtle variations in the arrangement of the features within the two descriptions. As for natural language [93, chapter 2] subtle syntactic variations of this form can lead to major differences in meaning. For example the sentences *"the guard threatened the unarmed robber with his gun"* and *"the robber threatened the unarmed guard with his gun"* possess quite different connotations. However, if they were

[3] A missing detail is one that does not possess activation.

[4] Appendix H also provides an explanation of the basic features of the graph of figure 4.7.

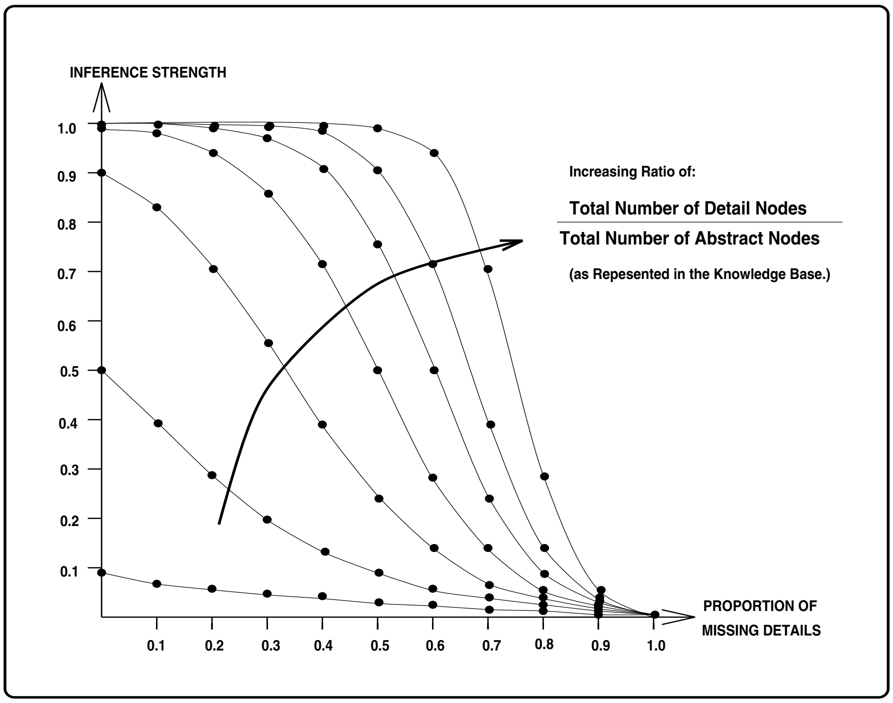


Figure 4.7: The Strength of Inference from an Incomplete Set of Details to the Abstract Node of a Structure Link.

represented as part of two robbery cases, it can be expected that phase 2 of the proposed retrieval scheme will create many links between the nodes representing components of the respective case descriptions (see figure 4.8). Only through the detection of the relative difference in positions of two paired nodes within their respective structure link hierarchies can the important distinction between the two cases be made.

The basis for the solution to this retrieval problem is to generate during phase 1 of retrieval a colour dependency hierarchy that replicates the hierarchical structure of the target case itself. In particular, for each structure link of the target case there will be a *colour dependency rule* generated that maps a parent colour for the structure links abstract node to a set of distinct child colours, one for each detail node of the structure link[5].

Generally, for a structure link in the representation of a source case, at the end of phase 2 of retrieval the detail nodes associated with that structure link will have received activations of various colours originating in the target case. These multi-coloured activations are recombined during upward activation propagation in a way that tends to place a single colour of activation on the structure link's abstract node. The recombination of colours is guided by the colour dependency rules generated in phase 1. Specifically, the following three rules are employed:

[5]The collection of colour dependency rules can be considered equivalent to a single, acyclic and non-tangled colour dependency hierarchy.

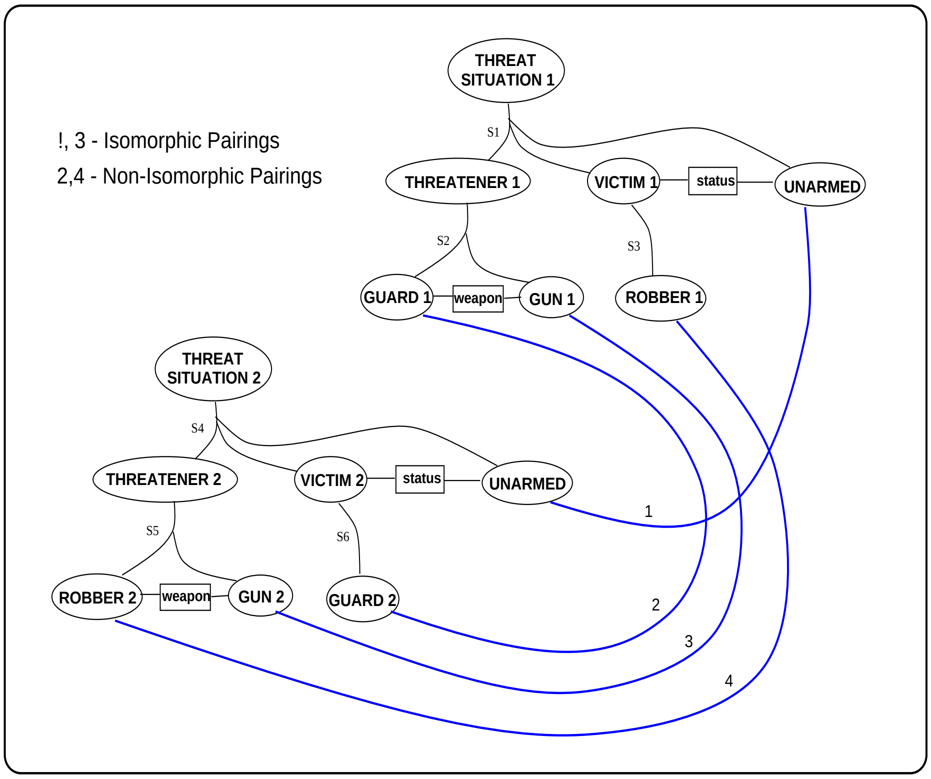


Figure 4.8: An Example of Non-Isomorphic Case Descriptions.

Structure Link Colour Merge Rule 1 *Select the colour dependency rule that incorporates the greatest number of detail node activation colours. Where more than one rule applies, choose all such rules.*

Structure Link Colour Merge Rule 2 *To measure upward inferential strength, only detail nodes that have activation of a colour specified in the selected colour dependency rule can be used.*

Structure Link Colour Merge Rule 3 *A second measure of inferential strength should be made, based on the proportion of colours of the selected colour dependency rule that are actually present in the structure link detail nodes.*

As a simple example of how this scheme may help prevent non-isomorphic retrievals consider again the example of figure 4.8. Taking **"THREAT SITUATION 1"** as the target case it is reasonable to assume that **"ROBBER 2"** receives activation from **"ROBBER 1"** and **"GUN 2"** receives its activation from **"GUN 1"**. However, although **"GUN 2"** and **"ROBBER 2"** are both details of the same structure link (S_5) of the source case the nodes from which they receive their activation are detail nodes of differing structure links of the target case. Hence, when activation propagation up S_5 occurs, according to the second and/or third rule above the upward inferential strength must be reduced and hence the strength of retrieval as a whole is reduced.

A more detailed example of how the proposed scheme for detailed structure preservation works can be found in appendix J. An important point to note is that it is a *bias*, rather than a restriction, on retrieval. It is compared to more rigorous approaches below.

More Rigorous Approaches to Preserving Structure

The use of colour merging rules can be contrasted to Russell's formal approach to analogical reasoning [115]. In Russell's work, structure is imposed upon the otherwise independent features of a case by the use of *"determinants"*. A determinant states that the values of a collection of features on the left-hand side of a rule are *sufficient* to determine the value of the single feature on the right-hand side. This idea has recently been extended by Huang [65, 66, 64] so that the determinant also identifies on its left-hand side *necessary* features for making an analogical inference. In either case the use of determinants makes analogy more principled as there is a reduced risk of inferring an unknown value based on irrelevant and coincidental features.

The scheme proposed in this thesis uses colour dependency rules to bias retrieval towards source cases with a similar structure to that of a target case. It follows that the usefulness of this bias (with respect to increasing the accuracy of information transferral from source to target case) stems from how valid are the hierarchical decompositions of cases in terms of not representing irrelevant features for each component description.

The problem of deriving a suitable decomposition of a case is equivalent to deriving a suitable determinant rule. Huang identifies three possible approaches [66] namely inductive learning, logical deduction, and direct user input. He focuses primarily on the process of induction and his work might possibly be modified to provide a means of automatically generating case component templates. Currently, however, in the prototype of the memory model of this thesis case representations are hand-coded. It follows that the validity of the hierarchical decomposition of cases is a knowledge engineering issue (see section 6.3.2 for an example of this).

Another interesting comparison can be made between the scheme proposed here for structure preservation and that proposed by Gentner et al [49, 51, 44]. Her *"systematicity principle"* states that:

> "Analogy conveys a system of connected knowledge, not a mere assortment of independent facts. Preferring systems of predicates that contain higher-order relations with inferential import is a syntactic expression of this tacit preference for coherence and deductive power in analogy." [43]

The memory model of this thesis does not treat the pattern of relationships between two descriptions as the fundamental constraint on retrieval (though as argued in appendix L a bias towards preserving relational structure may also exist in retrieval). Rather, this factor is just one of a number of similarity-based constraints that act in conjunction to guide the selective process of retrieval.

The scheme proposed here is unusual in that it embodies some of the spirit of systematicity within an activation passing system. At the same time, the proposed scheme is less open to the common criticisms of Gentner's theory; that it requires one-to-one feature correspondence, that it is highly sensitive to representation variations and that it takes no account of semantic similarity factors. In particular, a necessity of the flexibility requirement for retrieval is that preservation of structure be a preference rather than requirement during retrieval.

Consider again the example depicted in figure 4.8. Retrieving one of the *"threat"* cases, based on a strong feature correspondence and a weak structural correspondence, may be an appropriate retrieval; some knowledge about threatening behaviour is likely to be applicable to both the described scenarios and therefore exchangeable between them. For sure, another case concerning a robber being threatened may be more informative with respect to **"THREAT SITUATION 1"** than is **"THREAT SITUATION 2"** but if no such case exists then **"THREAT SITUATION 2"** despite its variation in descriptive structure, may be the best possible recollection.

4.3 Competition in Activation Spreading Retrieval

So far in this chapter activation passing has been put forward as a mechanism for performing selective retrieval because it allows the various in-built biases of the memory organisation to be quantitatively assessed during retrieval. In this section the argument is taken one step further by considering the nature of *competition* during retrieval. Again, traditional activation passing techniques will be seen to be inadequate for the task of case retrieval and a novel approach will be developed.

The need for competition in activation passing relates to both the efficiency and flexibility requirements of retrieval. The general effect of competition is to direct continuing activation propagation to areas of the network that have already received an appreciable amount of activation. Without some form of inhibition an activation distribution would quickly diverge to cover all nodes in the network. It follows that competition has two practical advantages :

- a heightening of accuracy by focusing attention on those nodes that receive activation from multiple sources. In particular this allows 'contextual priming' effects to influence the course of a retrieval [132, chapter3]. This is important for flexibility as it means that a recollection that was initially prompted by some unusual feature will tend to be reinforced, rather than swamped, by continuation of the retrieval process from additional features.

- a reduction in the amount of computation performed in activation passing. This relates in an obvious way to the desire for retrieval to be efficient.

A common strategy for activation passing is to implement a *"winner takes all"* strategy where eventually a single highly activated node will emerge as dominant over its competitors. A common technique for implementing competition is to use explicit inhibitory links in the network. As is explained below, neither a "winner takes all" inhibition strategy nor the use of explicit inhibitory links is adequate for the goals of this thesis.

4.3.1 Objections to a "Winner Takes All" Strategy and the Use of Explicit Inhibitors

"Winner takes all" in activation passing refers to the biasing of retrieval towards the nodes currently possessing most activation. The most highly activated nodes inhibit, in some way, the receipt of activation by their rivals. Eventually a single node will emerge as dominant over all its competitors. For a local network representation this type of discrimination is often achieved by the explicit use of *inhibitory* links in the network.

The objection to this normal model for activation passing is twofold. Firstly for CBR it is usually more useful to retrieve a corpus of several relevant previous cases rather than a single most relevant case [80, 33, 133]. Grading of relevance within this corpus is of course useful and is indeed a central goal for the proposed memory model. However, the combination of information from multiple source cases should not be precluded. For this reason it is not desirable for a "winner takes all" strategy to be acting between cases in memory. An alternative strategy for competition is proposed in section 4.3.3.

An inhibitory link negates the activation that passes along it. If excitatory activation links can be interpreted as representing logical implication then it follows that an inhibitory link represents "implies not". The competition arises because the more activation a node possesses the stronger it can inhibit its competitors which in turn reduces the competitors' inhibition of that node. In particular, it is often postulated that an explicit inhibitor should exist between all nodes of a mutually exclusive set [37, 90, 139].

The second objection to traditional approaches is that, unfortunately, explicit inhibition is generally inefficient, if not impractical. For a set of n mutually exclusive nodes, n^2 inhibitory links are required. For example, for McClellands letter and word recognition system [90] it is proposed that inhibitory links exist between all possible words, all possible letters, between a letter feature and all letters not possessing that feature and between each letter and all words not containing that letter. McClelland's system was successful, but only feasible because the data set it was applied to was relatively small (just over 1000 four letter words). It is doubtful that the system would scale up for significantly larger lexicons.

As a result of these qualms, inhibitors are reserved in the proposed model of memory only for the representation of exceptional negation, where some default features must be suppressed. This means that some *virtual* mechanism for implementing inhibition between competitive memory elements is required. This is discussed below.

4.3.2 Virtual Inhibition

As an alternative to the most highly activated nodes sending negative activation to other nodes, virtual inhibition simply directs further positive activation to the activated nodes. This principle was first suggested by Reggia [111]. In his scheme two factors influence how much activation a destination node receives; the proportion of activation it already possesses, (with respect to all other destination nodes), and the magnitude of the weight of the outward link with respect to all other outward link weights. This is expressed more

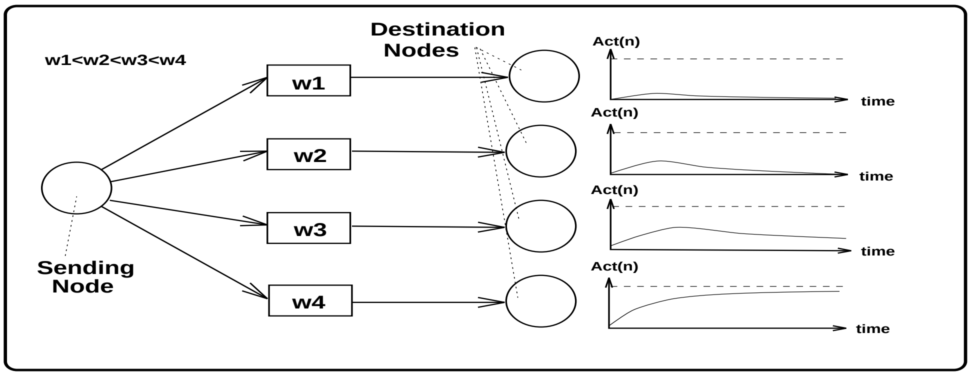


Figure 4.9: The General Effect of "Winner Takes All" Competition for Virtual Activation Inhibition

formally below.

Assume that a fraction of the activation possessed by a node n is to be shared amongst its outputs, $O(n)$. Let d_i be the destination node reached by following the ith outward link, $O(n)_i$. Let the weight w_i be associated with $O(n)_i$. In Reggia's scheme the amount of activation sent along the output link is characterised by:

$$OutAct(O(n)_i) \propto Act(n) * (\frac{w_i}{\sum_{j=1}^{Card(O(n))} w_j}) * (\frac{Act(d_i)}{\sum_{j=1}^{Card(O(n))} Act(d_j)}) \qquad (4.4)$$

As illustrated in figure 4.9, the above equation can quickly lead to one of a set of competing nodes becoming dominant[6]. Hence, the scheme provides a "winner takes all" activation propagation equivalent to the use of explicit inhibitor links and, by the arguments of the previous section, is not well suited to the task of case retrieval. An alternative virtual inhibition strategy is discussed below.

4.3.3 A "Leaders Share Spoils" Model

The question that is addressed in this section is how a mechanism for virtual inhibition can in general permit a restricted set of *several* alternatives to be simultaneously considered during retrieval. The key is for each node that is sending activation to select only a subset of its outward links to be receivers. The proposed equation for activation sending is:

$$OutAct(O(n)_i) \propto Act(n) * (\frac{IsSelected(O(n)_i)}{\sum_{j=1}^{Card(O(n))} IsSelected(O(n)_i)}) \qquad (4.5)$$

[6]The form of activation receipt shown in figure 4.9 is assumed to include a time dependent decay of the amount of activation possessed by a node, and also asymptotic increase towards a maximum magnitude of activation. Note that the system can settle into a state where more than one node has a significant amount of activation, though generally only one node can have near maximum magnitude activation.

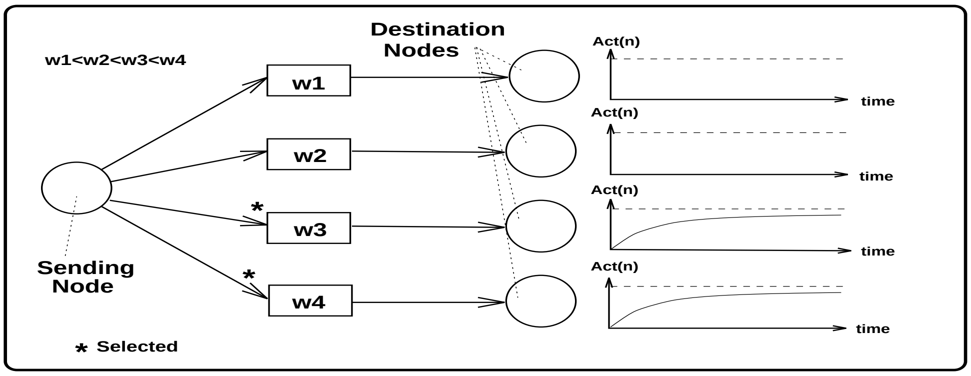


Figure 4.10: The Proposed Model for Activation Passing

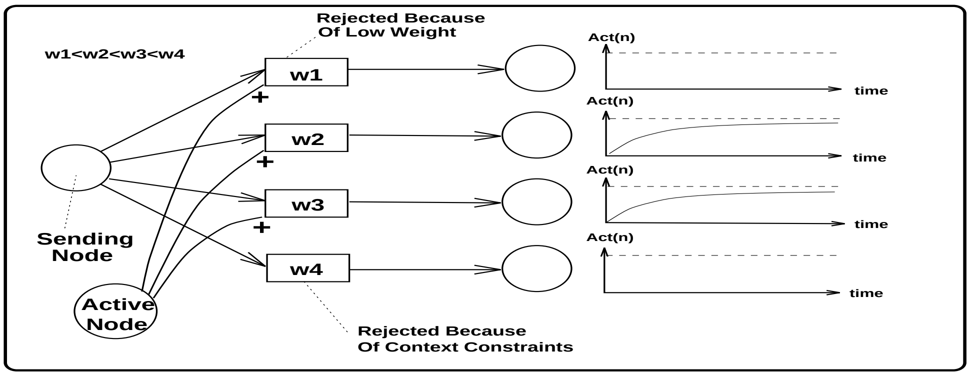


Figure 4.11: The Selection Criteria for Receiver Links

where $IsSelected(O(n)_i)$ takes the value 0 or 1. Note, as shown in figure 4.10, that the activation sent is shared *equally* amongst the selected receivers. This is important because it adds to the flexibility of the proposed scheme by enhancing its context sensitivity; should an unusual (low weighted) path through memory be selected due to an unusual context then a relatively high amount of activation will flow along that path. The power of the proposed scheme rests with the criteria used to select the current receivers. The rules for selection are described below. As will be seen, within the bounds of context the selection is biased towards those nodes already possessing activation and for this reason it can be thought of as a "leaders share the spoils" model.

Selection of Receiver Links

It is at this point that the representation of context and of probablistic relation weighting play their crucial role in the memory model. In section 2.4.3 context is defined to select which relations are currently relevant. A relation r_i associated with $O(n)_i$ is deemed traversable if and only if it has at least one *active* positive context and no active negative contexts. The status of active is independent of the current activation distribution and is bestowed upon a node by an external control process (see section 5.1.1).

At any one time only a subset of all outward activation links will pass through a traversable relation. Links through non-traversable relations are not candidates for selection as receivers (as shown in figure 4.11) hence the currently active contexts act as the primary constraints on activation propagation.

Each relation also possesses a weight as defined by the relation weighting scheme of section 2.6. As indicated by equation 4.5 this weighting has no effect on the amount of activation passed to receiving destination nodes. However, relation weighting acts as a secondary selection criteria acting within the bounds set by the active contexts.

An activation sending node is limited to select only B_c receiver links[7]. The selection takes place from all outward links that pass through traversable relations and is governed by two rules. Firstly, any outward link that leads to a destination node, d_i, that already possesses activation will be automatically selected. Secondly, if the selection of all such receivers does not exhaust the B_c limit, the remaining traversable links are selected in the order of the weights attached to their associated relations, until the limit on the number of receivers is reached. These two rules are defined below:

Receiver Selection Rule 1

$$Act(d_i) \neq 0 \wedge IsTraversable(r_i) = 1 \rightarrow IsSelected(O_i) = 1$$

Receiver Selection Rule 2

$$Act(d_i) = 0 \wedge (\sum_{j \neq i} IsSelected(O_j) < B_c) \wedge IsTraversable(r_i) \wedge$$
$$\neg(\exists k.(k \neq i) \wedge (IsSelected(O_k) = 0) \wedge IsTraversable(r_k) \wedge (w_k > w_i)) \rightarrow$$
$$IsSelected(O_i) = 1$$

Because rule 1 takes priority over rule 2 there is an inherent bias towards nodes already possessing activation. In this way the rules encourage competition in the process of activation propagation. The tendency is to preserve an existing activation distribution, rather than extending it to new nodes. Furthermore, localised exhaustive searching for all relational paths between those items already retrieved is encouraged. As is discussed in appendix L this type of focus aids in the retrieval of coherent bodies of knowledge rather than the sporadic selection of isolated knowledge items.

4.4 Towards a Symbolic/Analogue Hybrid System

In the above discussion an activation passing mechanism has been described that is appropriate for the requirements of a retrieval process as laid down in this thesis. In fact, what has been described is an example of a hybrid system where symbolic and analogue processing is conjoined in a very intimate way.

The requirements of a retrieval may generally be explicitly stated as symbolic constraints (see section 3.1.1). Accordingly, symbolic reasoning, manifested in terms of the selection of active contexts, forms the primary constraint that guides activation propagation in the proposed model. However, the recovery of useful knowledge items from memory also requires a more quantitative assessment of potential recollections. For

[7] In figure 4.11 $B_c = 2$.

multi-feature retrieval via indirect paths between cases the combined strength needs to be evaluated for all traversed routes to a given source case. In addition, the various preference biases built into memory should be quantitatively accounted for during retrieval. These numeric calculations are precisely what the three-phase activation passing scheme developed in this chapter has been designed to achieve.

The retrieval process that has been modelled is potentially very powerful because the symbolic and numeric reasoning occur simultaneously. This point is illustrated below by a comparison to similar hybrid systems.

4.4.1 A Comparison to Similar Hybrid Systems

Charniak was one of the first to postulate that natural language processing requires a combination of symbolic reasoning about syntax and a more probablistic judgement of plausible word semantics [30]. In his WIMP system [31] an activation passing process is used to find potential semantic paths between words in a sentence. Once retrieved, these paths are vetted by a more deductive reasoner to extract those that can be used to make valid inferences.

Problems were encountered with WIMP. Firstly it finds many relatively useless paths that have little inferential power. Charniak recognises that probabilities could provide a utility gradation for the possible paths retrieved. This is exactly the role played by relation weighting in the model of memory proposed in this thesis. The second problem Charniak identifies is that activation passing can produce many non-meaningful paths. This can be recognised as due to the lack of a focusing mechanism acting *at the time* of marker passing. The retrieval model proposed in this chapter allows control knowledge, in the form of active contexts, to constrain activation propagation and hence is potentially much less susceptible to finding irrelevant paths.

James Hendler's SCRAPS system [60] is similar to Charniak's. Its goal is to predict possible conflicts from the basic constraints of a plan and it achieves this by symbolic marker passing in a semantic network. SCRAPS is however unable to create deductive paths that include a similarity comparison between two objects. Hendler proposes [59, 60] a lower level network of definitive (or *"micro"*) features connecting the concepts in the semantic network. When the symbolic marker passing reaches a node of the semantic network that possesses definitive features, an activation passing process through the micro-feature network can find *similar* nodes in the semantic network. In this way the SCRAPS system involves a more intimate hybridisation than the Wimp system. However it still requires posterior path evaluation and there is no attempt to partition the semantic network such that context selects relevant paths.

For both SCRAPS and WIMPS an attenuation mechanism is employed to limit the extent of marker passing. Generally both systems will exhaustively search short bushy paths or long narrow paths but will not search long bushy paths. Conversely, the approach of limiting branching at individual nodes, as proposed in section 4.3.3, allows for the partial investigation of the long bushy search spaces at the cost of not exhaustively searching short bushy search spaces. The justification (as discussed in chapter 3) is that, within the bounds of context, a probablistic selection of a few possibilities from a bushy path often forms a sufficient recollection.

A more recent hybrid system to be designed for the task of knowledge retrieval is Lange et al's REMIND system [82], based on the earlier work on ROBIN [81]. REMIND interprets simple natural language scenario descriptions by the propagation of activation through a network containing the distributed description of a variety of previously experienced situations.

Unlike SCRAPS and WIMPS, REMIND is akin to the model of memory proposed in this thesis in that the activation propagation scheme used is designed to produce a similarity-based gradation of cases from memory. However, the network representation of REMIND contains many nodes that represent rather contrived knowledge items. For example, in [82] a case about hiding marijuana is related to a case about a boy hiding a pornographic magazine under his bed because the case components **"Police-See-Illegal"** and **"Guardian-See-Naughty"** are directly linked through a more general node **"See-Object"**. The representation of this type of *ad hoc* knowledge is in itself not a sufficient organisational basis for memory (see also section 5.2.2). In particular, if REMIND is used for large scale applications involving a multi-purpose knowledge base then it is likely to run into similar problems to those of WIMPS; retrieval by unguided activation propagation will be swamped by the production of spurious knowledge from memory.

In this respect there are two notable advantages for the memory model proposed here with respect to REMIND. Firstly, the structure link allows any representation of composite knowledge items to be broken down into their constituents, and hence a more selective retrieval by more specific features of a case description is possible. In addition, as for the case of WIMPS, the use of context as a primary constraint on the activation propagation scheme for the proposed model enables a much more focussed retrieval than is possible in REMIND.

Finally, Thagard, Holyoak, et al have done some significant work in applying connectionist techniques to the problem of analogical retrieval particularly in the development of the ARCS system [139]. This system performs case retrieval by a relaxation technique within an inhibitory/excitatory constraint network. The construction of this constraint network allows the same type of biasing on retrieval to be achieved in ARCS as for the model of memory proposed in this thesis.

The constraint network restricts possible matches between different case components and must be constructed for each retrieval. It follows that although the relaxation technique converges quickly to produce psychologically plausible results the actual construction of the network is where most work is done in ARCS. Furthermore this construction task is clearly dependent on the size of memory, as a network of constraints must be built for the set of components of each previous case in memory. In addition, the space inefficiency problem of using explicit inhibitors (see section 4.3.1) applies to ARCS.

These problems associated with ARCS have been avoided in the memory model proposed here because the organisation of memory is sufficiently rich to allow the constraints relied upon by ARCS to be dynamically superimposed at retrieval time. This imposition

	WIMP	SCRAPS	REMIND	ARCS	**CRASH**
Retrieval Includes Structural Constraints	NO	NO	NO	YES	YES
Retrieval Performs an Assessment of Similarity	NO	NO	YES	YES	YES
Point of Application For Pragmatic Constraints	Post Retrieval	Post Retrieval	During Retrieval	During Retrieval	During Retrieval
Search of Memory is Non-Exhaustive	YES	YES	NO	NO	YES

Table 4.1: A Comparison of Numeric/Symbolic Hybrid Systems for Knowledge Retrieval

occurs through a combination of active context assignment, the competitive nature of activation passing, and through the scheme proposed in section 4.2.2 for incorporating structural biasing into phase 3 of retrieval.

A summary of the comparison between the various hybrid systems is presented in table 4.1. Note that CRASH[8] is the prototype implementation of the model of memory developed in this thesis and will be described in more detail in the remaining chapters.

4.5 Chapter Summary: Activation Passing for Case Retrieval

This chapter has been concerned with the development of an activation passing model tailored to the needs of a case retrieval system. Disjunction is crucial within retrieval and has been implemented using activation colouring. In addition, colour dependency rules, in conjunction with a probablistic measure of inferential strength, have been shown to allow the inclusion of a bias towards preservation of the descriptive structure of a target case during retrieval of source cases.

A virtual inhibition model has been developed for the activation propagation based on the selection of receiver links out of individual nodes. Context plays the primary role in this selection with relation weighting providing a secondary selection criteria. The result is a "leaders share the spoils" competitiveness during retrieval. The resultant activation passing system has been seen as a potentially potent hybrid combining in an intimate way symbolic constraints and numerically-based evaluation during the retrieval of knowledge from memory.

Chapter 5 focuses once again primarily on the process of case-based reasoning. In particular, how well the model of memory developed so far satisfies the requirements of case retrieval will be considered.

[8]**C**ase **R**etrieval by **A**ctivation passing **SH**ell.

Chapter 5

A More Unified Model of Case-Based Reasoning

This chapter focuses once more on CBR. The aim is to show that the model of memory developed thus far is sufficient to explain the underlying mechanism of CBR. In particular, the emphasis is on the centrality of the process of selective case retrieval. The CRASH prototype system that implements the memory model is introduced. The flexibility of the retrieval mechanism of CRASH is shown; it subsumes both similarity-based and indexing models of case retrieval. In particular, various index transformation strategies are cast as indirect retrieval via the appropriate knowledge represented in the network of memory. Mapping between two case descriptions is seen to be unified with the process of retrieval and hence gains the same flexibility at relatively low computational cost. In addition, the evaluation and adaptation of source cases is explained via an iterative retrieval mechanism. Finally, retrieval is viewed as a creative process that may enable various modes of learning.

In this chapter the memory model developed in the preceding chapters is considered in terms of its role within the CBR framework. The chapter starts with a description of CRASH, the prototype system that provides a practical realisation of the model.

The body of the chapter is concerned with a comparison of the memory-based approach embodied in CRASH to more traditional approaches to CBR. In section 5.2, a number of issues concerning the flexibility of retrieval are considered and related to the retrieval mechanism of CRASH and in section 5.3 several traditionally post retrieval stages of CBR are cast in terms of retrieval. The issues that are linked to retrieval include: mapping (section 5.3.1), case adaptation (section 5.3.2) and learning (section 5.3.3).

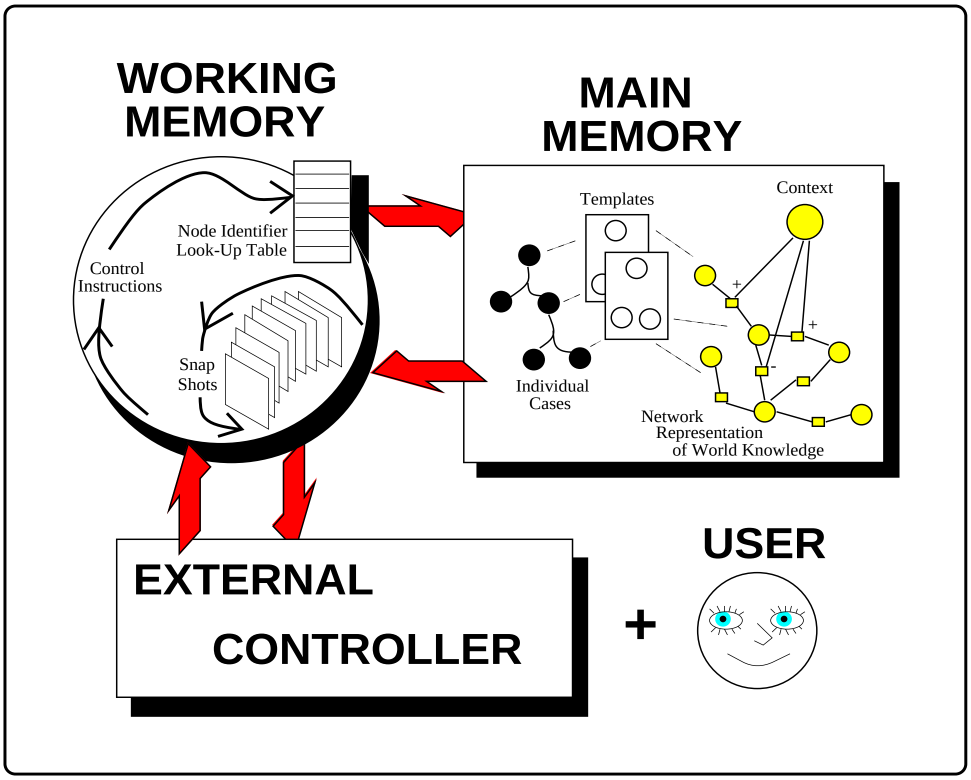


Figure 5.1: The Tripartite Structure of the CRASH System

5.1 The CRASH Implementation

CRASH is a full-blown prototype of the memory model, implemented using the ProKAPPA[1] KBS development tool. The components of the memory model, such as the various types of nodes and links, are represented using the object-orientated facilities of ProKAPPA. This provides a convenient implementation medium by which the specific details of the memory model can be tested. This section gives a brief overview of the CRASH prototype system. In particular, its general algorithm for case retrieval is presented.

5.1.1 An Overview of CRASH

CRASH comprises a tripartite architecture (as shown in figure 5.1); the *main memory*, the *working memory* and the *external controller*. Chapters 2 to 4 have already extensively detailed the main memory module, so the primary concern in this section is to explain the roles played by the other two components.

The External Controller (EC) performs reasoning tasks that are not easily built into the network primitives of main memory. For instance, the construction and modification of the network are responsibilities of the EC. The EC also determines when the activation passing for retrieval terminates and ultimately how the recollections produced by retrieval

[1]ProKAPPA is a trademark of Intellicorp, Inc.

are utilised. Finally, the EC exhibits an important influence on the activation spreading process. In order that control may be administered during activation propagation the process is quantised into discrete, time ordered segments, referred to as *cycles*. Before the initiation of each cycle, the EC establishes various conditions that will be imposed for its duration[2], such as which nodes will be the active contexts.

Working memory provides the interface between main memory and the EC. This includes a set of protocols by which the EC can influence activation passing. Working memory also contains a history of the status of main memory. This history records both the control instructions administered by the EC and the resultant activation *"snap-shot"* that is a log of the amount of activation possessed by each activated node at the end of a cycle.

5.1.2 A General Retrieval Mechanism

In general a 3 phase model for case retrieval is proposed in this thesis (see section 4.1.2). A framework for the mechanics of CRASH is given in figure 5.2. This framework reflects the emphasis given to the process of retrieval in this thesis. In particular, little interest is paid to how the recollection is subsequently utilised; as noted previously in section 1.1.2, this is assumed to be a relatively straightforward process.

A Comparison of the CRASH framework to the General CBR Framework

The general framework for case retrieval as implemented in CRASH is compared in figure 5.3 to the framework for CBR given in section 1.2.1. Some interesting many-to-one mappings are highlighted. In particular, the CBR stages of relevancy judgement and mapping are placed in correspondence with the three phases that make up the CRASH retrieval process. The propagation of activation through a network of world knowledge (phase 2 of CRASH's retrieval) is also seen to partially relate to the tasks of evaluation, adaptation and learning that are inherent to CBR. These relationships are explained in the remainder of this chapter.

5.2 On Flexibility in Retrieval

One of the main claims of this thesis is that designing the model of memory so as to include several key types of biasing constraint during the process of retrieval can facilitate the efficient recollection of *relevant* information. In section 5.2.1 this is seen as akin to providing a similarity-based model of retrieval; the nature of the relationship between 'similarity' and relevance is briefly discussed and how 'similarity' is manifest in CRASH is described. In section 5.2.2 the flexibility of CRASH's retrieval mechanism is given

[2]Each activation cycle is made up of a prespecified number of activation *pulses*. Notionally, the cycle is the most basic step of an activation propagation. However, breaking a single cycle down into a number of pulses allows for a closer approximation to an analogue process by the numeric operations that constitute the activation passing mechanism.

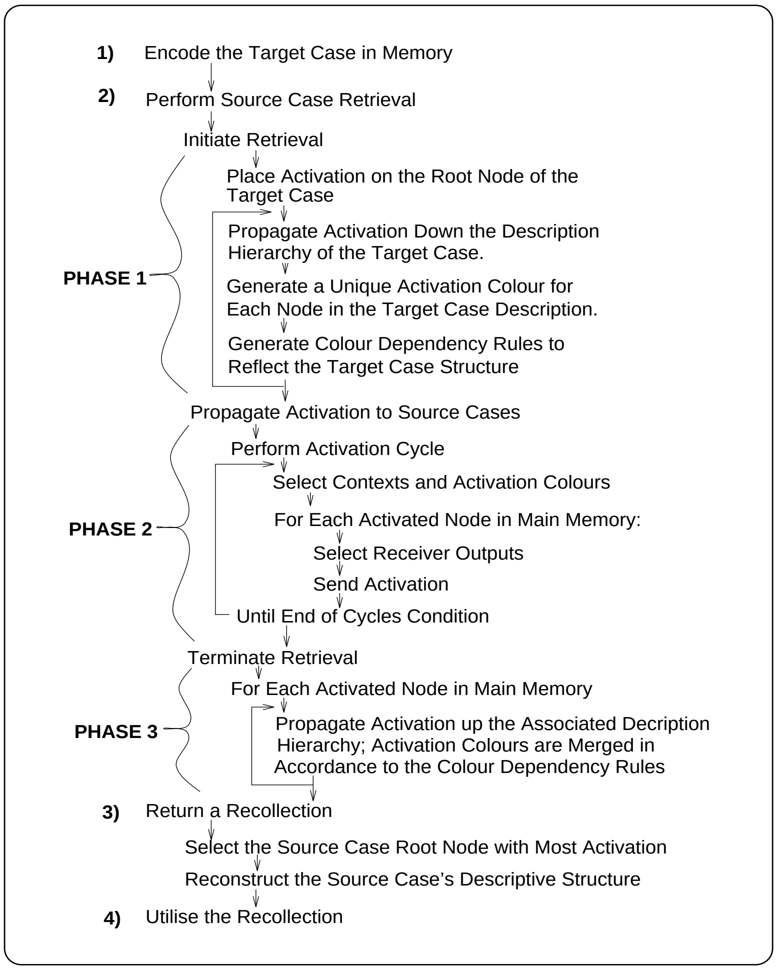


Figure 5.2: A General Description of the Mechanism of CRASH

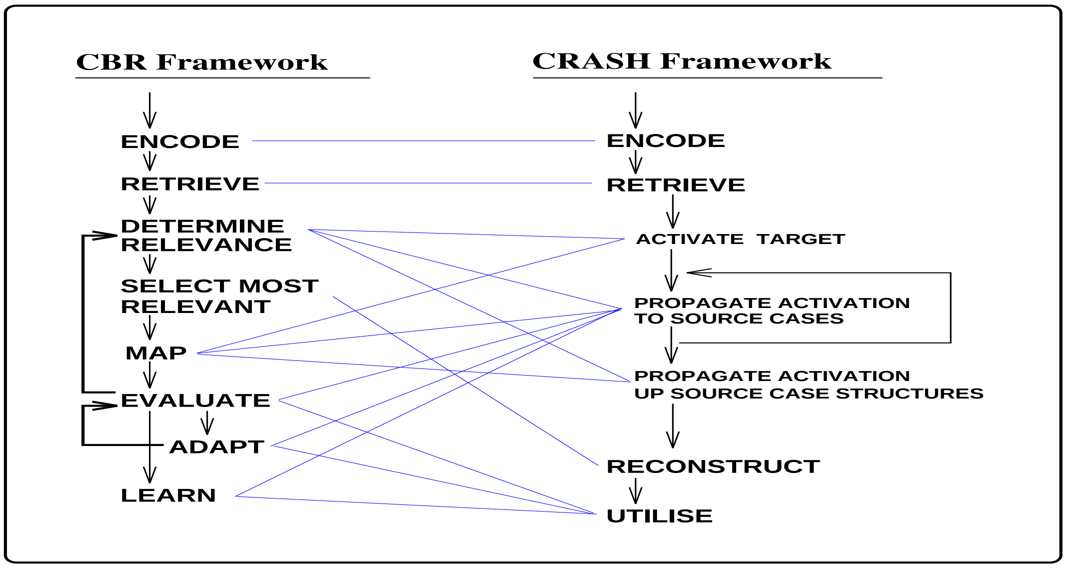


Figure 5.3: The Correspondence Between the CRASH Memory Algorithm and the General CBR Framework

further support by argument that the proposed memory model is capable of subsuming traditional approaches to providing flexible "indexing" in CBR.

5.2.1 Judging Similarity and Relevance During Retrieval

Similarity relates to flexibility in that for a model of memory to be completely flexible requires that previous knowledge be accessible even when there is only a partial match between the criteria expressed in the memory prompt and the features represented in the stored knowledge. For people, this type of flexibility seems a necessity as the complexity of the real world means that a new experience rarely complies fully with our preconceived model of the world. Hence, psychological theories of memory are often centered around a similarity-based model for knowledge representation and retrieval [49, 63, 112, 101, 94]. By an extension of this argument, for an artificial implementation of a memory for which flexibility is a major goal, it is reasonable to suppose that similarity assessment is an important function to be performed by the retrieval mechanism.

As described earlier (page 12) the general notion of similarity can be subclassified in terms of a number of criteria. Two knowledge items may be compared in terms of: their overall meaning (semantic similarity), their role with respect to a particular purpose or goal (pragmatic similarity), or in terms of the pattern of relationships in the description of the knowledge item (structural similarity). It will be argued in this section that CRASH's model of memory allows all three types of similarity to be incorporated into case retrieval. However, it must first be established why each type of similarity might be in some circumstances *useful* in terms of producing relevant cases from memory. This is discussed below.

```
Memory Prompt:
  During the raid on the El Passo bank the gang of bank robbers
  accidentally pass through an infra-read beam and in so doing
  trigger the alarms. The security guards will arrive imminently
  and the gang must rapidly plan their escape. Relevant
  recollections will be triggered from the salient features of
  a triggered alarm and arriving guards and must provide a
  possible solution that satisfies the goal of escape.

Recollection 1.
  John Smith remembers that during a previous raid on an
  ''Amiable Banks'' branch, when the alarm was triggered the gang
  was able to escape by overpowering the guards.
```

Figure 5.4: An Example of Necessary Requirements for a Retrieval

The Relationship Between Relevancy and Similarity

It is important to define what it means for a source case to be *relevant* to a given target case. Recall from section 3.1.1 that a memory prompt generally establishes a set of seed nodes from which the search of memory proceeds and a set of constraints that have to be satisfied by any recollection returned from memory. Establishing this type of predetermined constraint on retrieval is what is meant by pragmatic similarity; in other words pragmatic similarity relates to the *necessary* requirements of a retrieval. An example is given in figure 5.4, based on a description of a failed raid on the El Passo bank.

However, by the arguments of section 3.1.2, if a memory prompt is either open-ended (i.e. many cases in memory will satisfy the necessary requirements) or non-deterministic (i.e. the necessary requirements do not fully specify a path through memory) then predetermined constraints alone are not *sufficient* for finding a recollection. Additional constraints, inherent in the structure of memory itself, are required to guide the search.

Semantic similarity is one such type of additional constraint. Semantic similarity favours previous cases that share the greatest number of features with the current target case. The justification for using semantic similarity as a means of selecting relevant recollections is based on the principle of induction. Collections of features that have occurred together in previous cases (of the same type) are *assumed* to have done so due to some underlying reason, hence may be expected to occur together in future cases [67, 35, 95, 38]. It follows that a solution from a source case is more likely to be applicable to a target case if the two cases are semantically similar (assuming that the source case satisfies the necessary requirements of the target case).

For example, from the general description of the El Passo bank (see figure 2.9) it is known that the security guards are trained in martial arts. The preselection of features

```
Recollection 3.
  Helen Jones remembers that during a previous raid on a ''Swank
  Bank'' branch, when the alarm was triggered the gang were able
  to escape by hiding behind a door. When the guards  (whom the
  robbers knew to be trained in martial arts) rushed into the room
  the robbers made a run for it.
```

Figure 5.5: A Possibly Sufficient Recollection for the "Escape" Example Problem

that relate directly to the necessary requirements of the retrieval may omit this fact from the memory prompt generated for the problem described in figure 5.4. However, knowledge of the guards martial arts skills is not irrelevant[3]. If it is included as part of a secondary selection of semantically similar previous cases then the recollection given in figure 5.5 may be produced. The solution offered in figure 5.5 is more appropriate than that of figure 5.4 when all aspects of the problem faced by the robbers are taken into account.

It is precisely because the guards are trained in martial arts that the solution suggested by the recollection of figure 5.4 is insufficient, because it negates the possibility of the gang overpowering the guards. Now suppose that the gang had never encountered a situation of being confronted by more powerful guards before (i.e. the recollection of figure 5.5 is not possible). Then they would want to recall any knowledge concerning an escape when cornered by a more powerful adversary. A possibility is given in figure 5.6.

The point about the recollection in figure 5.6 is that it has little semantic similarity with respect to the initial problem description. The commonality lies in the pattern of relationships between the entities. It has been strongly argued elsewhere [148, 75, 44, 63] that preservation of relational structure in a match between a source case and a target case is important if a sound inference of knowledge from the former to the latter is to be carried out.

In summary, the relationship that emerges between the various types of similarity and relevance is as follows. Pragmatic similarity is synonymous with the identification of *necessary* requirements for a retrieved case to be relevant. Semantic and/or structural similarity may also be required in order to increase the likelihood that a retrieved case is *sufficient* with respect to the requirements of the memory prompt.

[3]It is not generally possible to completely separate the relevant from irrelevant features of a target case, which explains why reliance on semantic similarity has a role to play in determining relevance. Reasons for not being fully able to isolate the relevant features include the lack of a complete causal model of the domain or, in real time applications such as that of figure 5.4, a lack of time to complete the task.

```
Recollection 3.
  Jane Smith was reminded of the time that her pet cat was cornered
  by next-door's Alsatian. Jane had been amused by this incident
  because the cat had made itself look very fierce and bared its
  claws. Although this was mainly bluff, it had been sufficient to
  make the dog back away.

  Acting on this recollection Jane put her fingers in her pocket
  and, in time honoured fashion, pretended to be aiming a gun at the
  guards when they entered the room. The guards retreated and the
  robbers escaped.
```

Figure 5.6: Another Possible Solution to the Problem

The Realisation of Similarity Measures in CRASH's Retrieval

This section reviews how each type of similarity is manifested in CRASH (see also [18]). Firstly, because of the metaphorical nature of the network representation (see section 2.2.1), semantic similarity between any two features of a source and target case can be equated to two characteristics of the interconnecting network of world knowledge:

- The length of paths between connected nodes of the cases.
- The number of paths between connected node of the cases.

This idea is illustrated in figure 5.7. The two network properties can be measured using the activation passing mechanism of chapter 4 in a natural way. Firstly, a node only sends a fraction of its activation to its immediate successors in the network. It follows that the amount of activation that arrives at an eventual destination node tends to drop off exponentially with increasing length of path between it and the original source node[4]. Secondly, the preference of semantic similarity for multiple retrieval paths is accounted for by the function embedded within each node for combining activation from multiple sources.

Two additional aspects of the memory model effect the biasing towards semantic similarity it produces. Firstly, relation weighting distorts memory in a way that makes some paths (those that are potentially most useful) more likely to be traversed by activation than others. Hence, the strength of a connecting path between two cases is not just a measure of its indirectness but also of its default utility. Secondly, the preference of the proposed activation passing model for finding routes to already activated nodes (see page 68) increases the bias towards cases that are semantically similar due to the large number of connections between them and the target case.

[4]This exponential decay also has some psychological plausibility as it is in agreement with empirical evidence concerning the confusability (i.e. similarity) of perceived stimuli [121, 39, 122].

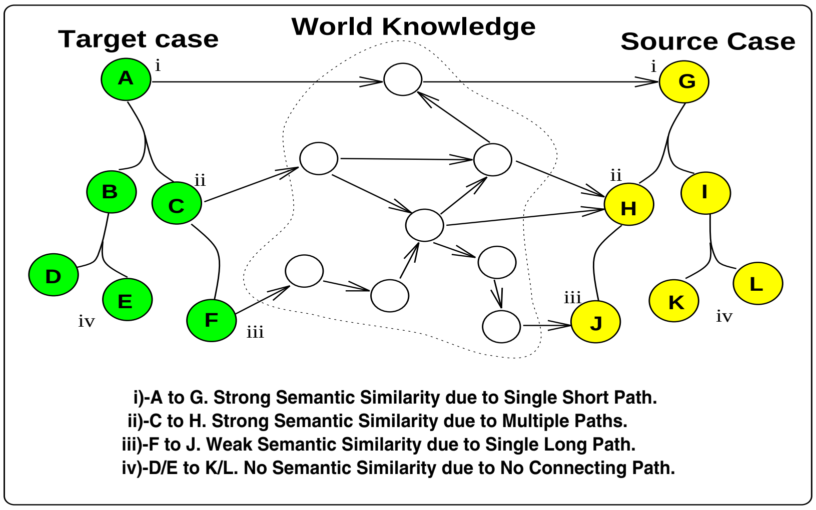


Figure 5.7: Equating Semantic Similarity to the Properties of a Local Network Representation

Pragmatic similarity corresponds to the use of context to select relations in CRASH. By the arguments of section 4.4, semantic and pragmatic similarity are intertwined in phase 2 of CRASH's retrieval. This provides a more flexible model of retrieval than the more standard approach of applying pragmatic criteria *before* measuring semantic similarity. In particular, it allows for the reaction to intermediate knowledge uncovered *en route* to a relevant recollection, as in the avoidance of anticipated planning problems (figure 2.4).

Finally, the notion of structural similarity, discussed in section 4.2.2, has already been related to the use of colour dependency rules to guide the merging of activation during phase 3 of CRASH's retrieval.

5.2.2 Providing a Generalised Model of Indexing

From the above discussion, the retrieval of CRASH can be seen as potentially highly flexible because it has built into it the means of biasing towards pragmatic, semantic and structural similarity. The alternative to a similarity-based approach to retrieval often used in CBR is indexing. This relies on an analysis of the application domain so as to predetermine those case features that pertain to the way in which cases will be used [5, 56, 58]. The indexing principle is that if memory is structured about purpose-directed features, relevant source cases may be retrieved with minimal search.

In this section the memory model implemented by CRASH is compared to indexing techniques and seen to be sufficiently general to cover a number of different approaches to indexing. The proposed memory model is seen as a compromise between the unreliable approach of using *surface level* features and the more principled but costly and inflexible use of *abstract features*. In addition, several approaches to increasing the flexibility of indexing, particularly in terms of index transformation strategies, will be seen as representable within the proposed model of retrieval.

Abstract or Surface-Level Features? A Compromise Position

One of the most hotly debated issues in CBR research has been what constitutes an *index* [57]. On the one hand, if the features that are readily available to initially describe a target case can be used to directly access source cases, the process of retrieval may require little effort or additional knowledge. On the other hand, reliance on these so called surface features can lead to the retrieval of a large number of superficially similar previous cases. The alternative is to expend some effort in deriving abstract features from the surface features [104]. Abstract features are generally more powerful in terms of the selection of relevant source cases, but are costly to deduce. This is the widely recognised dichotomy of indexing in CBR [28].

Assuming that abstract indices are required, two questions must be addressed:

- How are the abstract indices derived?
- What additional knowledge is required to derive abstract indices?

Generally, surface features are *interpreted* to form more abstract features as a preliminary stage of retrieval. For example, in the JUDGE system [7] a surface level description, such as *"Ted slashed at Al with a knife"*, is interpreted in terms of general legal terms, such as *"Unprovoked-Violence"*. Unfortunately, given a collection of surface features the space of possible abstractions that may be produced is vast [104]. A common approach is to use explanation-based techniques to extract the features of a case that are relevant to a particular goal [105, 104, 56]. This implies that a detailed causal model of a domain is required in order to automatically extract abstract features.

Even supposing the problem of forming abstract-feature indices can be solved, a purely *top-down* approach to case retrieval is flawed as it relies on source cases being encoded in terms of the same abstract features that are derived for a new target case. The purpose for the future retrieval of the source case must be predicted at the time of its encounter and this prohibits the formation of creative analogies[5].

As an example of this, Schank's model of memory involves organisation around goal-orientated hierarchies [117, 116]. Unfortunately this prohibits access to a case via features that are not connected to the central goal associated with the case. For example, a case about eating a burger in a fast food restaurant might include some description of the food itself. This information might be useful for a later problem to do with, say, planning relishes to be served at a barbecue. The goals of eating in a restaurant and planning a menu for a social occasion are sufficiently dissimilar to suggest that this retrieval cannot be achieved via the goal-based indexing alone. The retrieval needs to occur via the apparently incidental feature of the type of relish used to enhance the particular burger that was consumed in the restaurant.

[5]A creative analogy is taken to be a newly derived mapping between two case descriptions in separate knowledge domains, where no such mapping existed *a priori*.

A related problem with the use of abstract indices is that, in isolation, they are usually too general to allow for selective retrieval of source cases. For example, in some planning systems cases are modelled as instances of proverbial advice, such as *"many hands make light work"* [103]. However, a single proverb will apply to a great number of cases across a range of domains [70]. For example, the above proverb applies equally well to a group of bank robbers attempting to rapidly remove the contents of a bank vault and a school of piranhas attacking a wallowing antelope. Analogical recollections that are this abstract will rarely provide useful information from memory.

Generally, it is necessary to use a combination of surface-level and abstract features to perform analogical case retrieval [11]. What is proposed here is that the analytical knowledge required to link surface features to more abstract principles should, where possible, be represented as world knowledge in memory. It follows that phase 2 of the retrieval mechanism will then potentially be able to access source cases directly via any of the following routes: shared surface features; predetermined abstract features associated with the target case; or indirectly via abstract features stored in memory that are accessible from the surface features of the target case.

Because of the preference built into the activation passing mechanism for reinforcing the existing activation distribution, a retrieval by any one of these types of path will tend to encourage the other types of path to be formed (appendix L gives an example of the mechanics of this reinforcement). It follows that in CRASH a successful source case retrieval will generally involve paths via both surface-level features and more abstract features. This makes retrieval in CRASH more principled than retrieval based on surface-features alone while avoiding some of the inflexibility and ambiguity of the purely top-down approach.

Index Transformation for Flexible Retrieval

An interesting approach to flexibility is to systematically transform an initial set of indices, causing a corresponding change in the set of source cases that can be retrieved. Several classes of index transformation exist [134, 62] including:

- **Elaboration.** Add details to an existing abstract index.
- **Abstraction.** Find a generalised concept to which an existing index belongs.
- **Mutation.** Make a substitution or modification to components of a source case, (c.f. "tweaking", in [119, 117]).
- **Generation.** The scope of the target case is increased by using deductive techniques to infer additional knowledge about the case.

It is proposed here that each of these forms of index transformation can be redefined as a general strategy for network traversal, provided that the relevant knowledge is held in memory. This is illustrated in figure 5.8 and explained below.

Elaboration, in CRASH, is equivalent to traversing down a structure link. This provides a set of detail nodes that may be used as indices in place of the abstract node

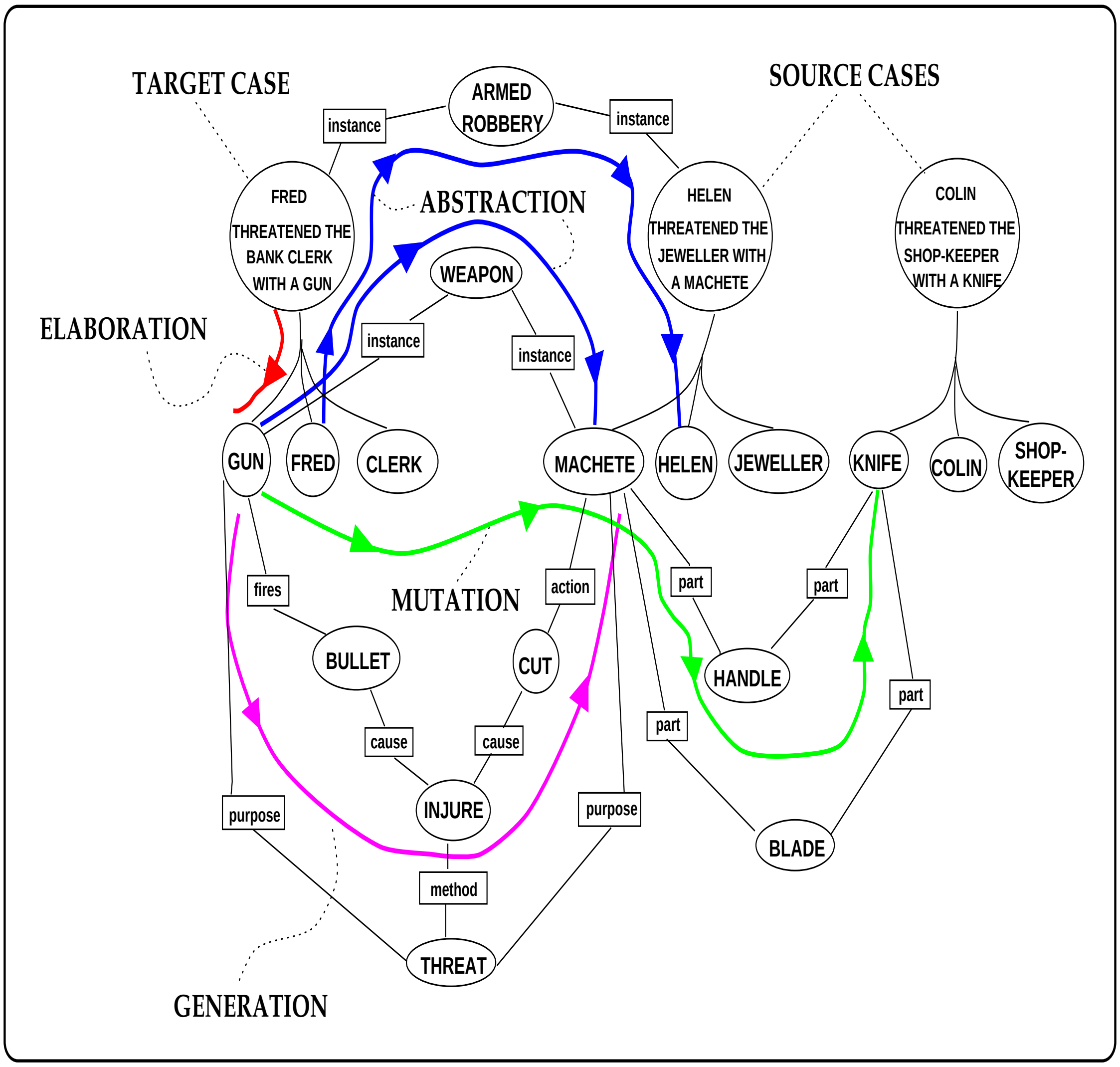


Figure 5.8: Index Transformation as Network Traversal

they represent. Hence, for the target case **"FRED THREATENED THE BANK CLERK WITH A GUN"** of figure 5.8, elaboration generates nodes, such as **"GUN"** and **"CLERK"**, as alternative indices.

In CRASH, there are two possible means for performing abstraction on a given target case node. The first is the opposite to elaboration, the upward traversal of structure links. In addition, the type hierarchies represented as part of the systems world knowledge may be used. Hence, **"WEAPON"** may be a useful abstract index generated from **"GUN"**, see figure 5.8.

Mutation is a more complex two step process in CRASH. Assume, a target case node has been linked, through the passage of activation, to a node, $n_{s,1}$, in an intermediate source case. If activation passage then continues from $n_{s,1}$ to find a second node, $n_{s,2}$ in another source case, the node $n_{s,1}$ can be thought of as a substitute index for the initial target case node and hence the initial source case has been mutated. For example, in figure 5.8 **"MACHETE"** may be considered a substitution index for **"GUN"**. Using **"MACHETE"** increases the chance of finding the **"KNIFE"** node. Note, this type of replacement could operate at any level in the case decomposition and it is possible to exchange quite large case components.

Generation in CRASH corresponds to the access of intermediate portions of the system's world knowledge *en route* to retrieving source case nodes. For example, the node **"INJURE"** may be thought of as an index generated for the target case of figure 5.8 through the traversal of domain knowledge pertaining to the functionality and purpose of guns.

5.2.3 Concluding Remarks on Flexible Retrieval

The discussion in this section has shown that the proposed memory model is capable of supporting both similarity-based and indexing models of retrieval. With respect to similarity-based retrieval three distinct categories of similarity (semantic, pragmatic and structural) have been identified and the relationship each plays in selecting relevant cases from memory briefly explained. CRASH is capable of manifesting all these types of similarity during the process of case retrieval and this provides some justification for the claim that the memory model has been designed to give highly flexible case retrieval.

An alternative view of retrieval is indexing. Indexing can generally be made flexible by in some way inferring new indices from the original target case description. For the proposed memory model inferencing is performed by the propagation of activation markers through the network representation. It follows that the various strategies for index transformation and generation may be recast as activation propagation through the appropriate knowledge represented in the network of memory. The feasibility of this has been illustrated.

5.3 Closing the CBR Loop

The research described in this thesis has focused on the organisation and functionality of a model of memory, designed to facilitate selective case retrieval. However, no account of CBR would be complete without at least a brief discussion of the 'post-retrieval' processes, namely; *mapping*, *evaluation*, *adaptation* and *learning*. The relationship of CRASH's memory model to these processes is stressed in this section.

5.3.1 Mapping in Retrieval

Mapping is the process of placing pairs of features from a source and target case in correspondence. As stated in section 1.2.4, in the absence of constraints, there are a factorial number of ways of mapping between two sets of features. More generally, there may exist many different criteria that determine how features should be paired [148], and generally mapping interacts in an intimate way with the other stages of the CBR process [40].

The view taken here is that the process of retrieval determines feature mappings. Given that a mapping can be thought of as a set of individual feature-to-feature pairings, this requires that for each source case node that is retrieved the target case node that instigated its retrieval must be uniquely identified. This is realised through each feature node of a target case being given a unique colour of activation during phase 1 of CRASH's retrieval process. All source case feature nodes that map to a given target feature node can, therefore, be identified at the end of retrieval through their possessing activation of the same colour as that node.

It is likely that, for a given target case node, an indirect path through the network of world knowledge will exist to most, if not all, nodes of a source case. However, one of the main arguments of chapter 2 is that memory can be organised so as to tend to select *'useful'* paths first. Given that a limit is generally placed on the amount of effort expended during phase 2 of retrieval, the tendency is for only the more useful paths between target and source case nodes to be traversed. In particular, the in-built constraints on memory influence which paths will be selected and hence restrict which feature mappings are possible. This approach is compared below to related work on the problem of mapping between a source and target case description.

Standard Approaches to Feature Mapping

Mapping in CBR is quite often highly simplified in practise. A common approach is to use standardised frames for case representation [124]. In such systems the factorial search space of possible feature mappings collapses to a linear comparison. A similar approach to the use of frames is to use a specialised, task specific vocabulary for describing cases. Features are matched by being expressed in identical terms. This relates to the basic philosophy behind indexing as the pre-selection of salient features not only permits efficient case retrieval but also establishes preliminary and important correspondences between source and target case descriptions [79].

A highly constrained approach to case mapping is implementable in CRASH, for instance by using the template facilities. However, in order to preserve the generality of the proposed model of memory analogical mapping must also be be possible in CRASH.

In contrast to CBR, in analogical reasoning the task of generating a mapping between two descriptions remains open-ended. Analogical mapping generally creates an indirect connection between paired case components and is often context sensitive [22]. Generally the problem can be characterised as a complex constraint satisfaction problem [40, 138]. Because mapping is not necessarily tied to retrieval, it is often implemented via an expensive and exhaustive search of all possible mappings [34]. This is a violation of the desire for efficiency in the model of memory. To illustrate how CRASH is capable of producing the required flexibility in mapping with improved efficiency it is compared in the remainder of this section to two admirable analogical mapping systems.

The Structure-Mapping Engine (SME) SME [44] is based on the systematicity principle, (see section 4.2.2). The emphasis is on preserving relational structure during mapping. SME proceeds via four main stages:

1. **Local Match Construction**. All possible feature pairings are found.
2. **Global Map (Gmap) Construction**. Local mappings are combined to produce maximally sized, globally consistent collections of feature pairings.
3. **Candidate Inference Construction**. The inferences that can be generated from each Gmap are determined.
4. **Match Evaluation**. From evidence attached to each local mapping, an evaluative score for each Gmap is constructed.

The generation of a local mapping in SME is comparable to the generation of a path between a source and target case node in CRASH. In SME the process is governed by a set of heuristic rules [49]. In particular (in the analogical mode) SME creates mappings between pairs of predicates with the same functor, then attempts to create a mapping between the arguments of the two predicates.

There is a tendency to preserve relational structure during mapping in CRASH (for example see appendix L) but this is not a requirement. In particular, CRASH cannot map higher-order relationships first, as is the case with SME. The reason for this is that in CRASH mapping is unified with the retrieval process and there is no obvious way that memory can be structured that permits focussed retrieval of source cases via abstract features alone (see section 5.2.2).

Stage 2 of the SME operation, the creation of Gmaps, is similar to phase 3 of CRASH's retrieval. CRASH is geared towards preserving the target case structure during phase 3 of the retrieval. However, a potential weakness of CRASH is that, during phase 3, it applies colour dependency rules simply by considering each source case structure link in isolation. It follows that, for highly nested case descriptions, CRASH may not preserve the most global structure mapping between a given source case and the target case. On the other hand the structure-preserving process of phase 3 is an efficient single pass process, hence there is a direct trade between mapping accuracy and computational effort. Furthermore, through being less rigorous than SME, CRASH is more tolerant of slight variations in the structures of two mapped cases.

Stage 4 of SME's operation combines evidence scores which are assigned to individual map hypotheses by the same rules that created them in Stage 1. Again, this is comparable to what occurs in CRASH during phase 2. However, compared to the subtle combination of various criteria that is performed during CRASH's activation passing, the assignment of weights to mappings in SME seems somewhat crude and arbitrary (see [44, appendix A]).

The Analogical Constraint Mapping Engine (ACME) Holyoak and Thagard's ACME system [138] provides the closest comparison to the CRASH model of mapping. Both systems incorporate pragmatic, semantic and structural similarity as constraints on mapping. However, like SME, ACME does not attempt to unify mapping and retrieval[6].

Both SME and ACME can generate a mapping based purely on syntactic considerations. Conversely, in CRASH, for a local mapping there must be at least some, possibly indirect, semantic similarity between the paired features. This is not considered a weakness as there exists a host of evidence that suggests that, in human analogical retrieval, some semantic similarity link is required [63, 49, 120]. Admittedly, structural similarity is indicated as being more important when subjects, presented with two cases, are asked to assess the soundness of the mapping [50]. However, this is a different problem to that being tackled in CRASH. The assumption that retrieval and mapping are fundamentally linked means that structural similarity cannot be considered as the only criteria by which mappings are generated.

Finally, ACME, like SME, initially generates *all* plausible single-feature correspondences between a source and target case. This exhaustiveness is avoided in the memory model of CRASH by applying constraints at the earliest possible point, namely during the retrieval of cases themselves. Efficiency is achieved because the search space of retrieval (and hence mapping) is greatly restricted and flexibility is achieved because the actual constraints that are applied are determined for each individual retrieval.

5.3.2 Evaluation and Adaptation

The evaluation of a successful source to target case mapping has received relatively little attention in the literature, considering the crucial role it plays in the CBR process. Several evaluation techniques can, however, be identified:

- **User Evaluation**. The easiest approach is to delegate the responsibility for determining how useful a source case is to the user of the CBR system [108].

[6]It should, however, be noted that the connections between ACME and its sibling retrieval program, ARCS [139], are strong, suggesting the possibility of unification.

- **Simulation**. In applications where a *"suck it and see"* approach can be applied to retrieved source cases, a simulation program may be used to automatically fabricate the results of applying the source case.

- **Knowledge-Intensive Verification**. If sufficient causal knowledge of the domain is available, it may be possible to use deductive techniques to assess the validity of a source case [124, 75, 58].

Ideally, in CRASH, evaluation should be included in retrieval as part of the index generation process described on page 84. Any causal knowledge that can be identified for automated evaluation should be included in the world knowledge held in main memory. The passage of activation through the network representing world knowledge then has the potential for uncovering appropriate portions of this knowledge and hence providing more deep-rooted criteria by which to retrieve source cases. This idea was illustrated earlier in the thesis in figure 2.4 where the explicit representation of causal domain knowledge was shown to allow a potential problem for a particular bank robbery to be detected and an appropriate type of solution to be identified during the search for a source case.

Unfortunately, it is unrealistic to expect a fully applicable source case to always be retrieved. Reasons for this include: incompleteness of the causal knowledge held in main memory, the non-deterministic and non-exhaustive nature of the proposed activation passing process, and the simple lack of a completely satisfactory case in memory. It follows that the external controller is still required to perform some evaluation of the retrieved case, though this issue is not addressed in this thesis.

Adaptation as Recursive Retrieval

Generally, the process of evaluation identifies two things:

1. Those requirements of the target case which remain unsolved *after* the transfer of knowledge from the source case.

2. Those aspects of a source case that cannot be transferred compatibly to the target case.

Adaptation techniques have been highly researched in CBR and a host of possible techniques exist [17]. The general principle behind adaptation is either to modify the incompatible aspects of the source case or to modify any non-crucial requirements of the target case [62]. The aim of the modification is to attempt to alleviate any constraints that stop the source case being wholely applicable to the target case.

A strong link between adaptation and retrieval is suggested here. A common approach to adaptation is for the incompatible features of the source case to be *"tweaked"* [74] in a way that is similar to the indexing transformation strategies discussed at the end of section 5.2.2. Another common approach is simply to replace incompatible components of a source case with an equivalent component from a different source case [20, 61, 73, 33, 110]. This latter approach effectively merges multiple source cases to create a composite source case that more completely matches and satisfies the requirements of the target case[7]. These two approaches to adaptation, it is proposed, can be unified in a single model recasting adaptation as iterative retrieval.

Assume that at the end of evaluation, the external controller has identified those target case requirements not yet satisfied by mapping from the source case. Using the terminology of section 3.1, these requirements can form the *query set* for a new memory prompt. The constraints on the memory prompt will determine the type of adaptation that is to be implemented by the retrieval. The seed nodes for the new memory prompt will generally come from both the original target case and the partially utilised source case that is to be adapted.

If an alternative to an incompatible source case component is retrieved from the nodes of the target case description then this can be thought of as a direct replacement. However, if a new source case component is retrieved from the incompatible source case component itself then, by arguments similar to those given early for index transformation, this can be thought of as modification. Hence, no clear distinction is made between the processes of modification and substitution of target case features. Both are implemented by a secondary retrieval and the only distinguishing factor is what part of memory constitutes the seed nodes for the secondary memory prompt.

Typically the cycle of memory prompt generation, retrieval and re-evaluation will iterate several times until all requirements of the target case are met. The achievement of adaptation is thus based primarily around the process of retrieval. For a worked example of this process the interested reader is referred to appendix K.

5.3.3 Learning

A commonly used definition of machine learning is:

> A learning machine is one that is able to modify itself in such a way as to improve its future performance, either by increasing the efficiency or accuracy by which it performs one of its current tasks, or by allowing the system to perform some new task that it was unable to perform previously.

There are virtually no CBR systems that do not involve learning in one form or another. Indeed, in some systems the two processes are inseparable [116, 21]. The lack of attention given to learning in this thesis is, therefore, somewhat unusual. This is redressed in this section, where various modes of learning are related to the memory model developed in the preceding chapters. In particular, the notion of creative retrieval is discussed.

Learning in CBR can generally be characterised in terms of the amount of restructuring of memory it requires [117]. The most straightforward form of learning is the

[7]Note, the creation of a composite source case requires a distributed case representation, as is supported in CRASH. Furthermore, the fact that CRASH is designed to retrieve a corpus of several relevant source cases, (see section 4.3.3), means that alternative case components may be found without the external controller necessarily having to initiate a second retrieval.

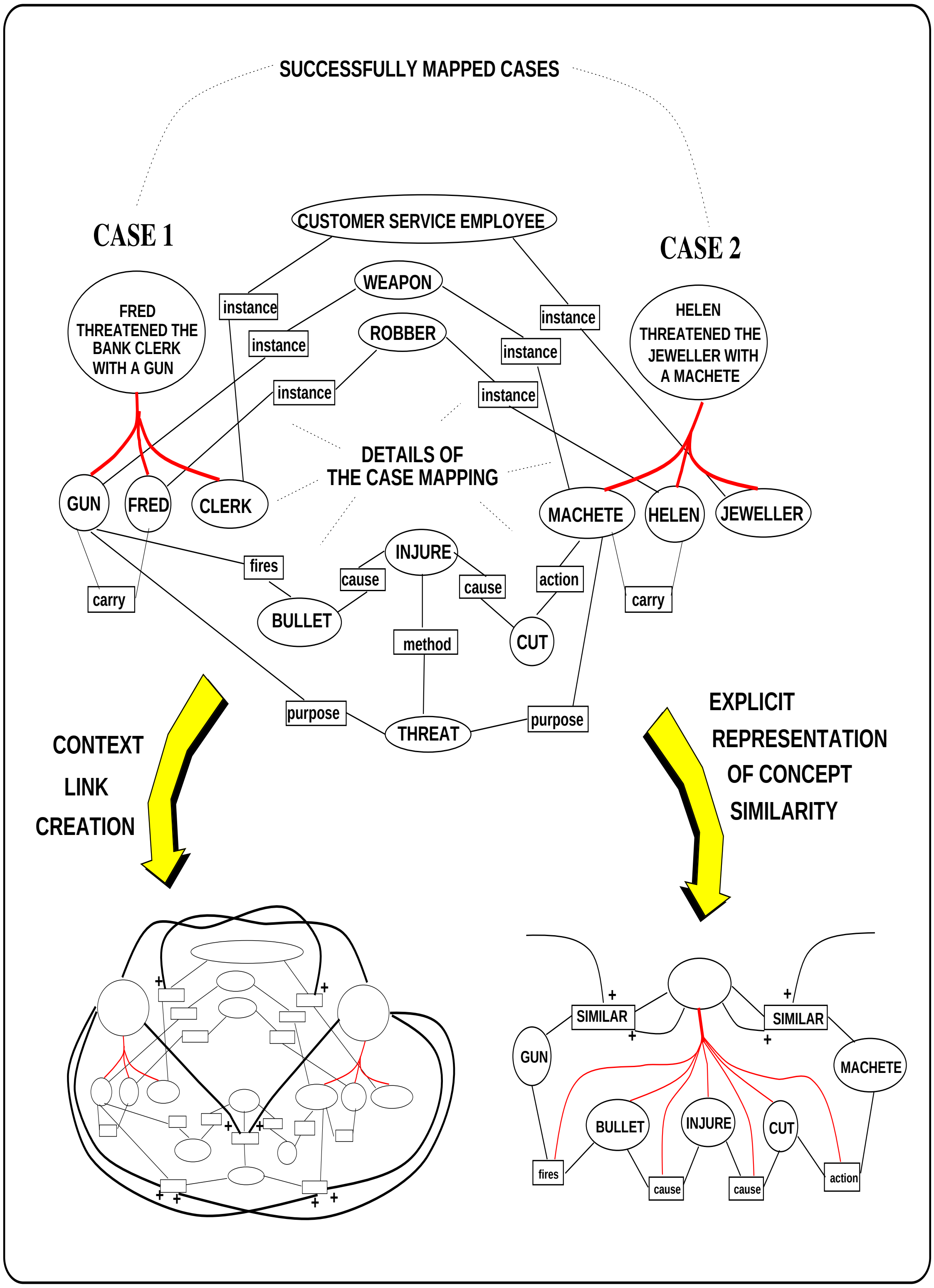


Figure 5.9: Restructuring Memory to Strengthen Successful Mappings; Two Possible Approaches in CRASH

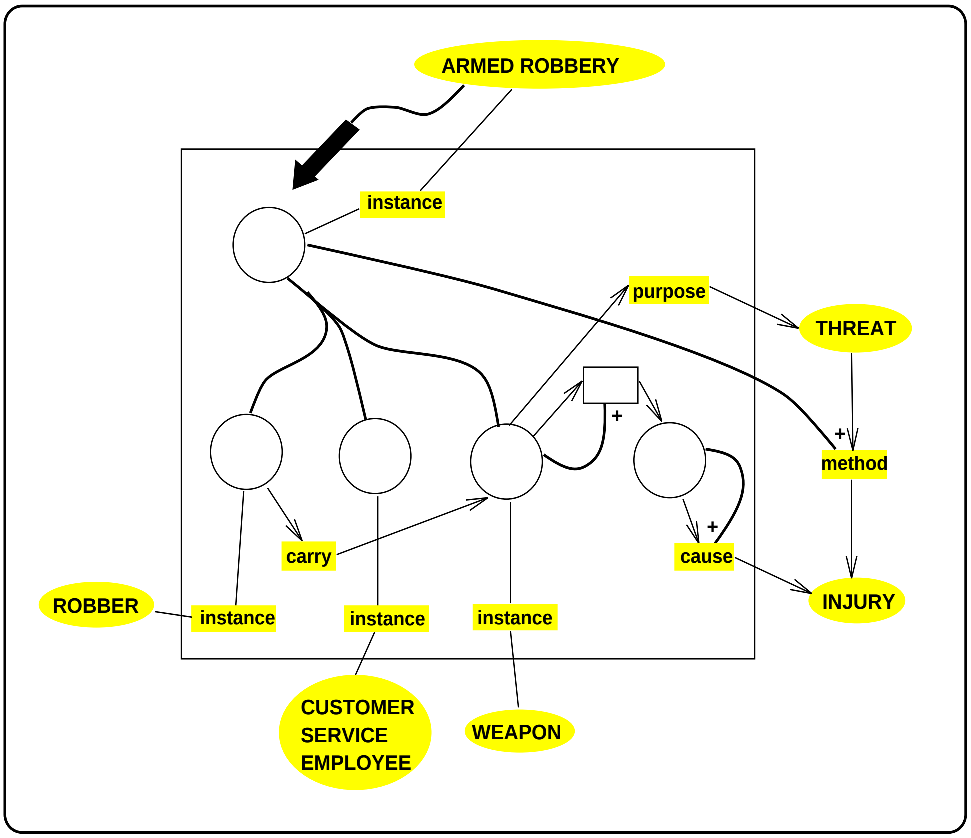


Figure 5.10: An Example of a Template Representing the Generalisation of Multiple Cases

simple incremental addition of new cases. More complex forms of learning involve more radical restructuring of the case base. Generally, the restructuring is based around the preservation and/or generalisation of successful case mappings and the recording of any failed case applications.

In CRASH there are several ways by which a case mapping may be preserved. One method is simply to strengthen a successful target to source mapping path via the insertion of context links (see page 43). A stronger link can be forged through 'shortening' the path between two nodes by explicitly representing their mutual similarity. These two techniques are illustrated in figure 5.9. An even more radical restructuring of memory is to explicitly represent a generalisation of the entire source to target case mapping [123, 148]. For example, figure 5.10 shows a possible generalisation of the cases shown in figure 5.9. Generally, the decision as to when and by what means memory is restructured must evaluate the potential benefits of the restructuring with respect to any practical cost that it incurs, (see equation 3.1 for an example).

A contentious point is where creativity enters into a case-based learning system. In failure-driven models the emphasis is often on the post-retrieval analysis of cases. In contrast, in the same way that relevancy judgement, mapping and adaptation have been seen here as highly dependent on the process of retrieval, it is proposed that most of the creativity in learning occurs through the retrieval of source cases. This stance is argued below.

Creative Retrieval

The first important point to clarify is the nature of the knowledge stored in the CRASH knowledge base. The knowledge is not factual, in the sense of classical logic, but is more akin to the knowledge held in non-monotonic systems, such as that described in [89]. Most relationships represented in the CRASH knowledge base are only valid in a limited number of contexts and with a probability less than unity.

Because CRASH's memory provides a *probablistic* model of the world there are important implications in terms of what is achieved during retrieval. Retrieval in CRASH is *abductive* by nature. The mapping paths generated between the nodes of a target and source case description can be thought of as constrained *hypotheses* as to how the associated case features *may* validly correspond. It follows that, if the evaluator of a CRASH retrieval accepts the mapping, this enables learning in that knowledge that was by definition tentative can be made more concrete (at least in the context of the current cases).

CRASH is also capable of representing definitive domain knowledge. If retrieval is performed via such knowledge, the process may be considered as *deductive* rather than *abductive*. As such, simply strengthening the retrieval path does not qualitatively affect CRASH's model of the world. Nevertheless, the strengthening of deductive paths may be thought of as a form of learning, in that it emphasises *useful* valid paths (as opposed to less useful valid paths) and hence will improve the future performance of memory.

Finally, as for any other CBR system, any generalisation that is made of a source to target case mapping in CRASH will invoke a form of *inductive* learning. Because CRASH's retrieval is biased to include a measure of relevance, the retrieval itself will be a useful guide in generating useful generalisations from induction [95].

5.4 A Summary of CRASH as a Tool for CBR

This chapter has attempted to shed some light on how the model of memory developed earlier in the thesis ties in with a broader view of CBR. It is emphasised that the standard CBR framework is not adhered to closely in CRASH. Rather, the process of memory retrieval is seen to be dominant. It has been argued that, through the incorporation of similarity constraints, CRASH manages to permit a high degree of flexibility in retrieval without incurring any insurmountable computational costs.

Contentiously, it has been proposed that retrieval lies at the heart of other stages of CBR. A unified model of mapping and retrieval has been put forward. It has also been argued that explicit representation of causal domain knowledge in memory can reduce the amount of external evaluation and adaptation that is required by partially performing these processes during retrieval. In addition, an external evaluator may exploit CRASH's memory by performing case adaptation as an iterative retrieval process. Finally, it has been argued that most of the creativity required for a CBR system to perform interesting learning may come from the flexibility in the source case retrieval.

The model proposed here can be seen as more fundamental than the commonly advocated CBR framework, as it unifies the various stages of the framework through the retrieval process. The following chapter adds some credibility to the proposed model through a presentation of empirical results achieved through the use of CRASH.

Chapter 6

Bin-Packing: A Case Study in Retrieval

Chapter 5 argued for the importance of the role of the three-phase retrieval model of CRASH within a more complete CBR system. The aim of this chapter is to provide supporting evidence that the proposed memory model can be practically fulfilled and implements its requirements. The chapter describes a number of experiments using the CRASH prototype on a simplistic "bin-packing" problem. Among the conclusions indicated by the results are: CRASH's retrieval shows a bias towards semantic and structural similarity that is robust to factors such as the size of case base; the similarity-biased retrieval provides a useful selection of cases despite attempts to avoid purpose-directed tailoring in the cases; individual CRASH retrievals do not require an exhaustive search of memory to achieve a high quality; and finally, the amount of effort required to achieve a high quality of retrieval increases at a rate that is slower than a linear dependency on the size of the case base.

This chapter describes the application of the CRASH prototype to an example knowledge domain, namely the *"bin-packing"* problem. The goals for this experimentation are specified below.

6.1 Aims of the Case Study

The purpose of this case study is to provide evidence for some of the pivotal aspects of the memory model developed in this thesis. At the most basic level there is a need to show that the proposed knowledge representation and retrieval mechanism can actually be implemented.

The next step is to demonstrate that the retrieval mechanism is selective in an appropriate way. In particular, in accordance with section 5.2.1, retrieval should exhibit a bias towards semantic and structural similarity and the extent to which this biasing results in the selection of 'useful' source cases must be investigated. Note that a key requirement of the experimentation is to investigate the potential power of retrieval based on similarity assessment alone, hence the experimentation is explicitly designed to avoid the type of inherent tailoring that is associated with indexing techniques.

It is also important to test how robust is the selective biasing implemented in retrieval. This is the motivation behind the experimentation described in section 6.5 which involves an investigation of phase 3 of CRASH's retrieval and its effect on the over-all retrieval result. In addition, section 6.6 includes the testing of CRASH over a range of differently sized case bases.

Finally, it is important to gain some insights into the relationship between retrieval effort and retrieval quality. The experimentation into this aspect is described in section 6.6. In particular, the results achieved with CRASH are compared to those using a more conventional sorting algorithm to order the cases in memory.

6.2 A Description of the Problem

In this section an introduction to the bin-packing case study is given. This comprises both a description of what the problem entails and reasons for chosing the problem for demonstrating the feasibility of the CRASH memory model.

6.2.1 The Bin-Packing Problem

The bin-packing problem involves the tessellation of shapes in a 2-dimensional grid. The initial state of the problem is characterised by a set of unplaced shape instances and an empty rectangular grid. The task is to place the shapes within the grid so that they exactly cover its area. The problem is decomposed into a number of steps (corresponding to individual bin-packing cases) each involving the placement of one or more[1] shapes in the remaining space. An example of a complete solution to a bin-packing problem is shown in figure 6.1.

To simplify the problem, only rectangles and L-shapes are used in this case study. Rectangles come in three sizes; 1-by1, 1-by-2 and 1-by-4. L-shapes come in two sizes; 2-by-2 and 2-by-3.

6.2.2 The Suitability of Bin-Packing as a Test Problem

The bin-packing problem suffers from the common yet unavoidable criticism of many AI example applications; that it is unrealistically simple. However, this type of tiling problem is worthy of AI research because it can be seen as an abstraction of more realistic problems, such as cell placement in VLSI design [135].

From an experimental perspective, an important advantage of tackling a simplistic problem as a means of empirical evaluation is that the method for solving the problem

[1] In certain circumstances, such as near completion of the bin-packing problem or when interlocking two L-shapes, it makes sense to think of several shapes being placed as a single problem solving step.

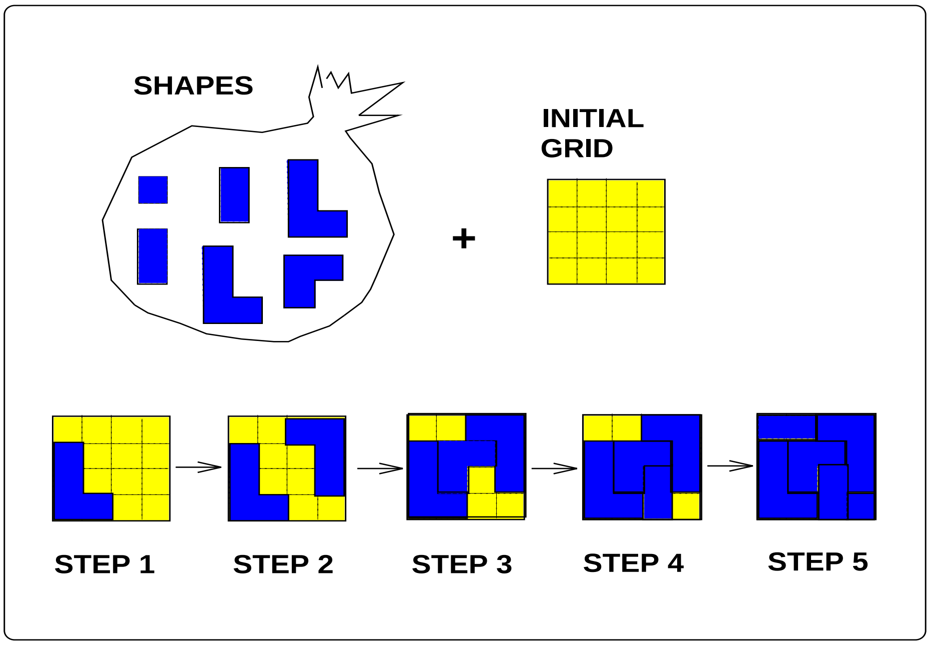


Figure 6.1: The Process of Solving the Bin-Packing Problem

can be clearly understood. For example, knowledge elicitation revealed that the following collection of 6 heuristics (listed in a rough order of priority) is highly effective for solving the bin-packing problem:

- **Heuristic 1.** Place L-shapes before rectangles.
- **Heuristic 2.** Place the largest shapes first.
- **Heuristic 3.** Maximise the extent of contact of a placed shape with the remaining space's periphery.
- **Heuristic 4.** Chose a shape placement that maximises the rectangularity of the remaining space.
- **Heuristic 5.** The bounding box for the remaining space must be able to contain the bounding box of any of the remaining shapes.
- **Heuristic 6.** Given two remaining L-shapes of the same type, place them as a pair such that they are tessellated to form a rectangle.

It is not obvious, given these heuristics, that CBR is an appropriate solution technique. For CBR to be directly applicable the cases would need to be defined in terms of abstract and purpose directed features, such as *"the rectangularity of the remaining space"*. This type of tailoring in the case representation is deliberately avoided in the experimental work described in this chapter (see section 6.3.2). This is because the interest is in determining to what extent a purely similarity-based process, as opposed to a more traditional indexing approach, can prove useful for case retrieval.

6.3 Building the Knowledge Base

A brief description of the techniques used to acquire the bin-packing cases is given in section 6.3.1. This is followed in section 6.3.2 by a discussion of representation issues that arose in the construction of the knowledge base.

6.3.1 The Knowledge Elicitation Procedure

The knowledge required to perform bin-packing was acquired by the recording of the author's own performance on a series of example problems[2]. The conscious thought processes experienced at the time of problem-solving were faithfully recorded using audio tape and the transcripts used to generate the source cases to be encoded in CRASH. A total of 22 cases were generated representing each individual problem solving step for 4 completed bin-packing problems. These cases are used as the source cases in the experimentation described in this chapter and are pictorially summarised in appendix M. In addition, a single target case is used as the basis of all retrievals in this experimental work and is also presented in appendix M [3]

6.3.2 Representing Cases and Domain Knowledge

Cases are represented as a decompositional hierarchy, the general form of which is shown in figure 6.2. This hierarchical structure is in fact represented by a collection of templates, one per major component of the case description. Each case is encoded by multiple instantiations of these templates. The overall structure of each case is preserved using structure links.

It was intended to avoid unnecessary purpose-directed biasing in the representation of cases. For example, the shapes to be placed are represented as a randomly ordered list[4]. From the set of heuristics given on page 96 shape type has relevance with respect to problem solving decisions, hence in order to be pragmatically neutral the structure of the case representation should not discriminate between shape types.

Similarly, the placement of shapes and the partially filled grid are both represented by lists of coordinates. It follows that any abstract geometric features that might be strongly related to the solution of the bin-packing problem remain implicit within the representation. Finally, the case description has some noise deliberately included through the inclusion of at least one irrelevant feature; namely the **"PARITY"** of the remaining

[2]Thanks to Jonathan Cox for providing help in setting the example bin-packing problems.

[3]Only one target case was used in the experimental work so as not to introduce an additional source of variation into the experimental procedure. The experimental work described in this chapter is largely concerned with the basic mechanics of the retrieval process. Experimental work into how the features of the prompting target case can influence the result of retrieval is an interesting area for future work, (see also section 7.2.1).

[4]Note that a list representation is chosen rather than a set representation for aggregate components of the case representation because it gives the case representation a more restricted structure. This is advantageous because it supports experimentation into the bias of retrieval towards structural similarity, (see section 6.4 for a definition of the structural similarity bench-mark).

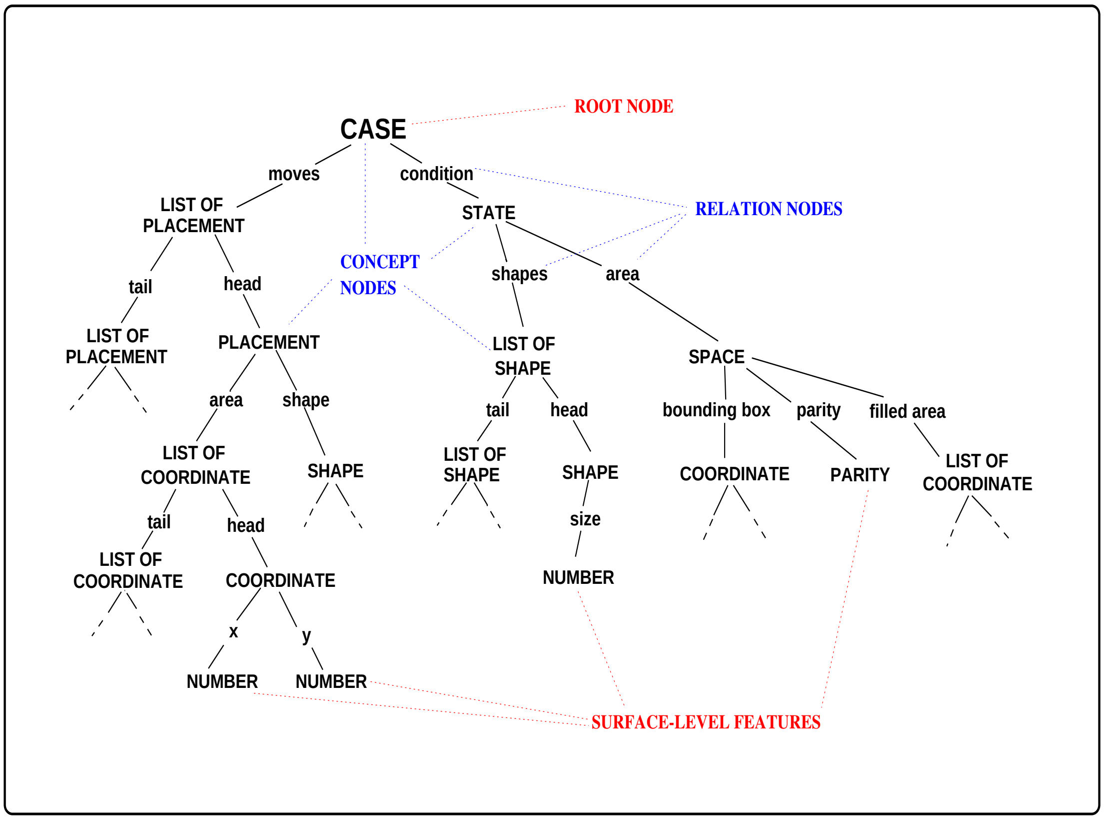


Figure 6.2: The Decompositional Structure for Bin-Packing Cases

coordinate space to be filled. An example of the representation of one of the cases from figure 6.1 is depicted in figure 6.3[5].

The network of world knowledge in which the bin-packing cases are embedded is fairly limited. It mainly involves topological information concerning each case component. For example, the shape instances within each case are all leaves of a single type hierarchy which includes, at an intermediate level, the concepts of *"L-shape"* and *"rectangle"*. This hierarchy allows non-identical types of shape (e.g. small and large L-shape's) to be linked during case retrieval.

6.4 Bench-mark Case Comparisons

In this section, three test orderings for the bin-packing source cases are established. These orderings involve three theory-based comparison relationships between the target and source cases: heuristic-based assessment, semantic similarity and structural similarity. The three orderings of cases that are produced are used to evaluate how significant is the bias towards each relationship for the retrieval mechanism of CRASH. The description

[5]Note that the duplication of numeric values in figure 6.3 is for clarity of illustration; in the actual implementation of the case base each value is represented by a single node.

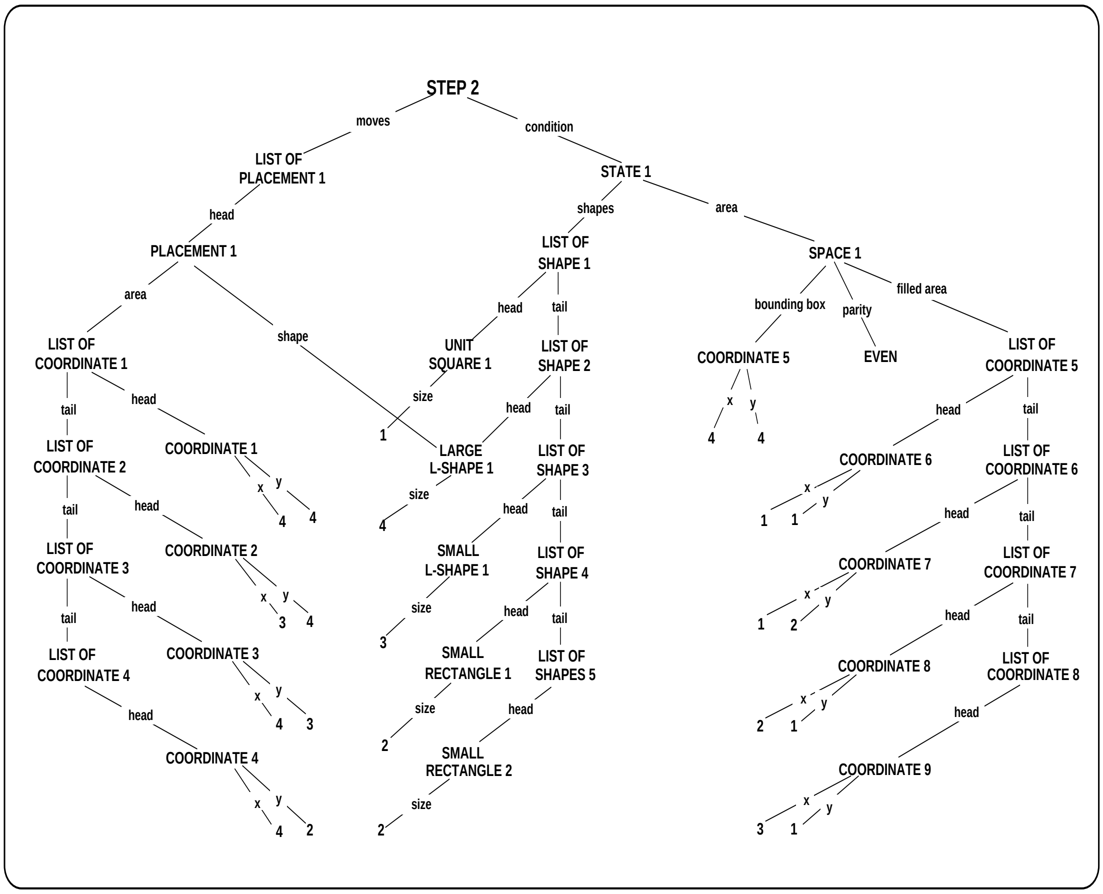


Figure 6.3: An Example of a Bin-Packing Case

here for each bench-mark is fairly terse and the interested reader is referred to appendix N for a more detailed account.

6.4.1 The Heuristic-Based Ordering of Cases

The basis for the heuristic-based ordering is to transfer the parts of each source case that relate to the placement of shapes to the target case and then determine *post hoc* which of the heuristics of section 6.2.2 support the completed target case. A score is determined for each source case in terms of the number of heuristics that are applicable[6]. This score is, in turn, used to derive an ordering of the source cases that reflects how *useful* they are in terms of proposing a problem-solving step for the target case.

[6]The summation of applicable heuristics is weighted so as to take into account the recognised prioritisation of the heuristics.

6.4.2 Similarity-Based Orderings of Cases

Pure semantic similarity treats the case as being represented by a *bag* of features, with no account being given to the relative positions of two paired features in their respective case description structures. The definition of semantic similarity used here for the semantic ordering of cases is described in section N.2 of appendix N. It is based on finding the set of compatible pairings between individual case features that maximises a summation of the match scores for the individual pairings. The relative difference in sizes of the two case descriptions is also taken into account and the resultant similarity measure is normalised.

Structural similarity is akin to semantic similarity in so much as it is based on a summation of individual pairings of case features. However, the choice of which features from the two case descriptions can be placed in correspondence is constrained by the structure of each case description. Given that all bin-packing cases are encoded according to the standard hierarchical decomposition given in figure 6.2, structural similarity can be measured by matching the components of the two case descriptions in a top-down manner, as described in section N.3 of appendix N.

6.4.3 Measuring Case Ordering Correlation

The main metric that is used to compare the strength of correspondence of a retrieval to a bench-mark factor is the standard deviation, σ, in their respective case orderings (see section N.5). This metric provides a measure of how good is the over-all[7] gradation of source cases produced by a retrieval.

The greater the correspondence between two ordered lists of cases the lower is the value of σ. However, the value of σ is clearly dependent on the size of the lists of cases involved. In order to permit comparison between the results for different sizes of case base the statistical significance of a given value of σ must be considered. Accordingly, in section N.5 of appendix N a *biasing factor* is defined that is normalised to lie in the range -1.0 (complete contradiction) to 1.0 (complete match). In addition, in section N.6 four statistically-based categories of retrieval quality are established, namely; strong positive bias, medium positive bias, weak positive bias and negative bias.

Bench-mark Correlation

In section 5.2.1 arguments were put forward that structural and semantic similarity are generally required in order to fully determine the relevance of a particular case to some new problem situation. The implication in terms of the bench-mark factors defined in this section is that the heuristic-based ordering of cases ought to be partially coupled to the semantic and structural orderings.

Table 6.1 represents the correlation that exists between each bench-mark relationship for each size of case base used in the experimentation. The results show that there

[7] A more focussed measure of retrieval quality based on only the three most strongly retrieved source cases is also introduced in section 6.6.

Case Base Size	Compared Bench-Marks	Ordering Deviation (σ)	Biasing Factor	Statistical Significance (Weak,Med,Strong)
8 CASES	Semantic to Structural	1.93	0.930	Strong +ve
	Semantic to Heuristic	1.85	0.940	Strong +ve
	Structural to Heuristic	2.73	0.680	Med +ve
13 CASES	Semantic to Structural	2.58	0.995	Strong +ve
	Semantic to Heuristic	3.62	0.950	Strong +ve
	Structural to Heuristic	4.45	0.735	Med +ve
22 CASES	Semantic to Structural	3.23	1.000	Strong +ve
	Semantic to Heuristic	6.19	0.985	Strong +ve
	Structural to Heuristic	6.34	0.980	Strong +ve

Table 6.1: Correlation Between Bench-Mark Orderings

is a particularly strong connection between semantic and structural similarity. This is not surprising given that a standardised case decomposition is used for representing the source cases. Importantly, the hypothesis that a statistically significant correlation exists between either similarity-based ordering and the heuristic-based ordering is also borne out.

6.5 An Investigation of the Nature of Feature-Based Retrieval

This section describes a preliminary experiment to confirm that the retrieval process performed by CRASH produces meaningful results and that the results of the retrieval mechanism are not unduly sensitive to variation in some key parameters.

6.5.1 Experiment 1: Motivations and Method

The motivation behind this experiment is twofold. A basic goal is to establish that a biasing towards both semantic and structural similarity is exhibited by the CRASH retrieval mechanism. In addition, the experiment investigates how variation of the parameters that control phase 3 of the retrieval (i.e. the amalgamation of isolated single feature retrievals, see section 4.2.2) influences the quality of retrieval as a whole. Two key parameters were systematically and independently varied in the experiment:

- The sensitivity of upward activation passage to missing details.
- The amplification of activation during upwards traversal.

For this experiment, phase 2 of CRASH's retrieval process is guided by a precompiled sequence of active contexts (see section O.1.1 of appendix O for details). This sequence defines a retrieval search space that involves the direct access of source case nodes from every node in the target case description. This enables some observations to be made as to role played by individual case features in determining retrieval quality. The extent of retrieval search during phase 2 is kept at a constant and relatively low value, in order that the influence of phase 3 be emphasised.

The experiment is carried out using the case base of 13 source case (see section M.1 of appendix M). A more detailed description of the experiment can be found in section O.1 of appendix O.

6.5.2 Experiment 1: Results

Sensitivity to missing details is governed by the *proportionality exponent*, (γ of equation 4.2). As γ is increased so the sensitivity to missing details increases. Figure 6.4 depicts the results in terms of the effect of varying γ on the quality of retrieval, as measured with respect to each of the three bench-marks.

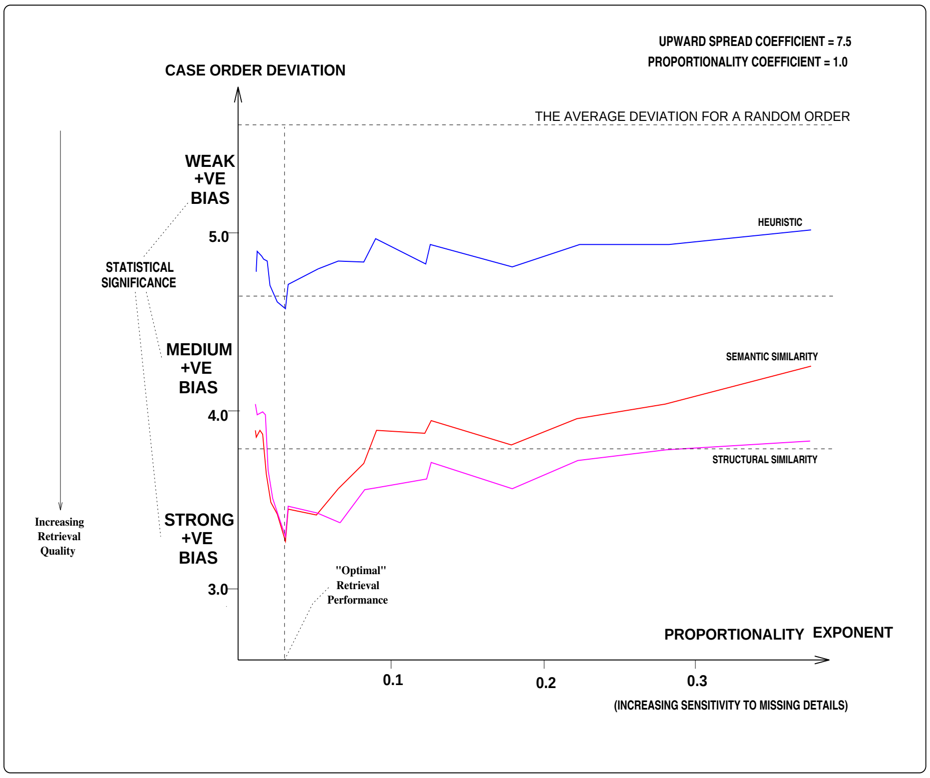


Figure 6.4: The Effect of Sensitivity to Missing Details on the Quality of Case Retrieval

The amplification of activation during upward traversal is directly proportional to the *upward spread coefficient*, (η of equation 4.3). The effect of varying η on the resultant ordering of retrieved cases is shown in figure 6.5.

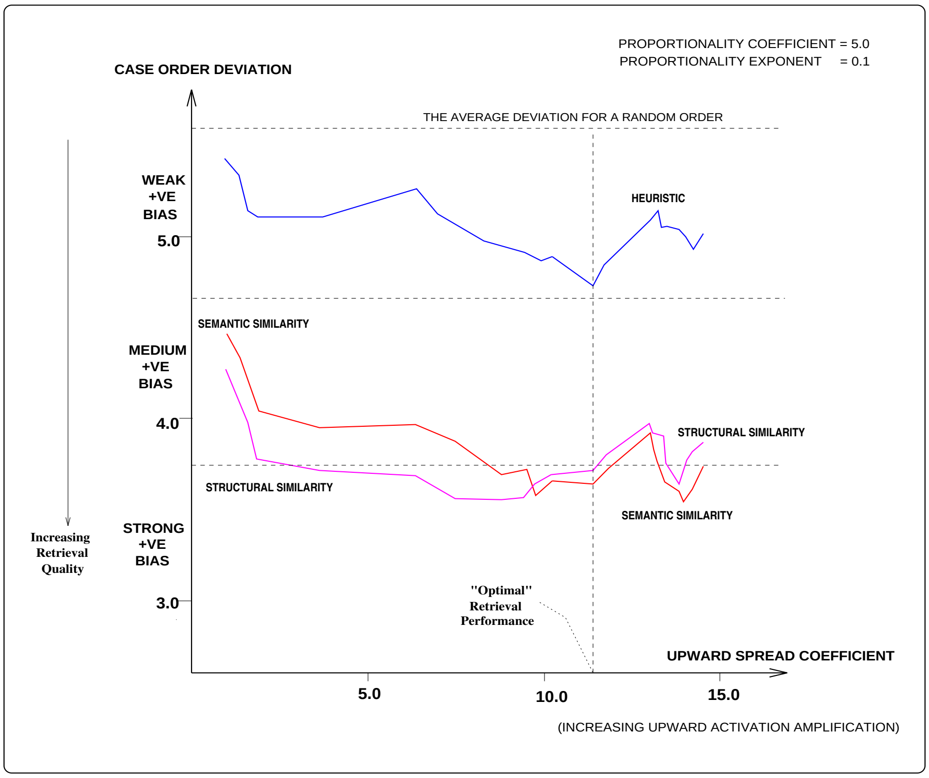


Figure 6.5: The Effect of the Upward Spread Coefficient on the Quality of Case Retrieval

6.5.3 Experiment 1: An Analysis of the Results

With respect to the first goal of this experiment it can be clearly seen from figures 6.4 and 6.5 that there is generally a strong bias in CRASH's retrieval towards both semantic and structural similarity. In addition a bias (albeit somewhat weaker) persists towards the heuristic-based ordering of source cases. In addition, the relative variation in retrieval quality is low for variation in both parameters (γ and η). This suggests that CRASH is robust to the exact details of the mechanism employed during phase 3 of retrieval.

The effect on retrieval caused by varying both parameters suggest that the relative influence of both surface-level and more high-level case features must be balanced. If surface-features are allowed to dominate (low sensitivity to missing details and/or high activation amplification) then isolated surface-features become highly influential and the retrieval becomes 'noisy'. However there is also evidence that important information is contained in the surface features of a case; if the emphasis on surface features is diminished too far (either through high sensitivity to missing details or low upward activation amplification) then the quality of retrieval correspondingly decreases.

6.6 The Scalability of Retrieval

This section describes experimental work aimed at showing that the retrieval process of CRASH has the potential to be successfully used in very large case bases.

6.6.1 Experiment 2. Motivations and Method

From experiment 1 it was shown that semantic similarity, structural similarity and to a lesser extent heuristic relevance are all preferences manifested by the retrieval mechanism of CRASH. The first motivation for experiment 2 is to strengthen these results by showing that they persist over a range of case base sizes.

An additional motivation for this experimentation is to determine the relationship between the resultant quality of retrieval and the amount of effort expended during retrieval as performed by CRASH. In particular the goal is to provide some support for one of the central claims of this thesis; that exhaustive search may be avoidable if memory is suitably well structured.

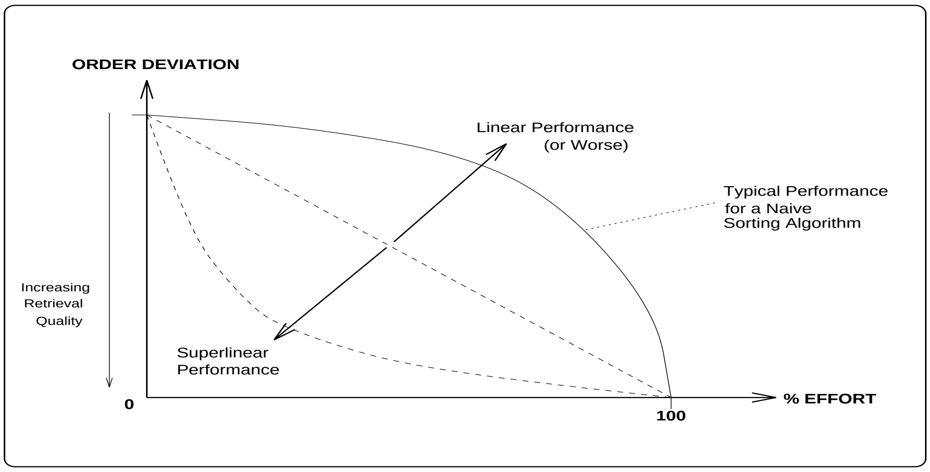


Figure 6.6: A General Interpretation of Quality Versus Effort from the Perspective of Order Deviation

In order to enable the results of CRASH to be meaningfully interpreted section O.2.1 of appendix O defines the performance of a generic sorting algorithm. The conclusion that is reached is that a linear decline in order deviation per percentage effort marks the boundary for linear performance (see figure 6.6). In general, a naive sorting algorithm that traverses the retrieval search space in an exhaustive and unguided way will produce a quality (measured by order deviation) versus effort curve that lies on or above this boundary line. Conversely, a sorting algorithm that generates a curve that lies below the boundary line can be said to be usefully *selective* in that the retrieval results it produces for a fraction of total effort have a disproportionately high quality. Such performance is referred to as *superlinear*.

The final motivation for this experiment is to demonstrate that CRASH's retrieval results are useful not only in terms of the over-all gradation of source cases that is produced but also in terms of the ability to select a small number of the most salient cases from memory. This is important as a useful case retrieval mechanism generally needs to provide a small corpus of cases for a case-based reasoner to work from. Accordingly, the experimental results are also analysed in terms of the three most highly activated source cases produced at the end of each retrieval.

The experiment involved performing a large number of retrievals with varying degrees of effort performed during phase 2. The procedure was carried out on three separate case bases containing 8, 13 and 22 cases. The retrieval search space of phase 2 is defined by the same context sequence as in experiment 1. This context sequence treats all case features equally during retrieval; there is no purpose-directed filtering of the target case description. In addition, the relation weighting scheme of section 2.6 is not employed on the case bases. It follows that the performance of the activation passing scheme of CRASH is being tested on the basic, unconstrained network representation (c.f. figure 3.3).

For a more detailed description of this experiment refer to section O.2 of appendix O.

6.6.2 Experiment 2: Results

This section summarises the results produced by CRASH during experiment 2. The results are viewed first in terms of order deviation and then in terms of the quality of the top three cases produced by each retrieval.

Order Deviation Results

The combination of case base size and bench-mark factor produces 9 possible dimensions for the results of this chapter. A comprehensive summary of the experimental results is presented via a series of graphs in appendix P. With respect to order deviation the results generally conform to the pattern presented in figure 6.7.

A description of the features of this graph can be found in section O.2.2. The most important feature is the *mean performance curve*. This tends to a maximal retrieval quality with increasing retrieval effort but at a rate that reduces as effort increases. The implications are as follows:

- At low degrees of effort CRASH's retrieval generally performs better than would be anticipated for a linear, unguided search of the retrieval search space.

- There exists a *cross-over point* above which the performance of CRASH's retrieval becomes worse than that of unguided search.

- The *memory bias* reflects the maximal (sub-optimal) performance that may be achieved by CRASH's retrieval mechanism.

That the mean performance curve initially lies below the naive sorting threshold suggests that CRASH may offer a potential reduction in the amount of effort required to

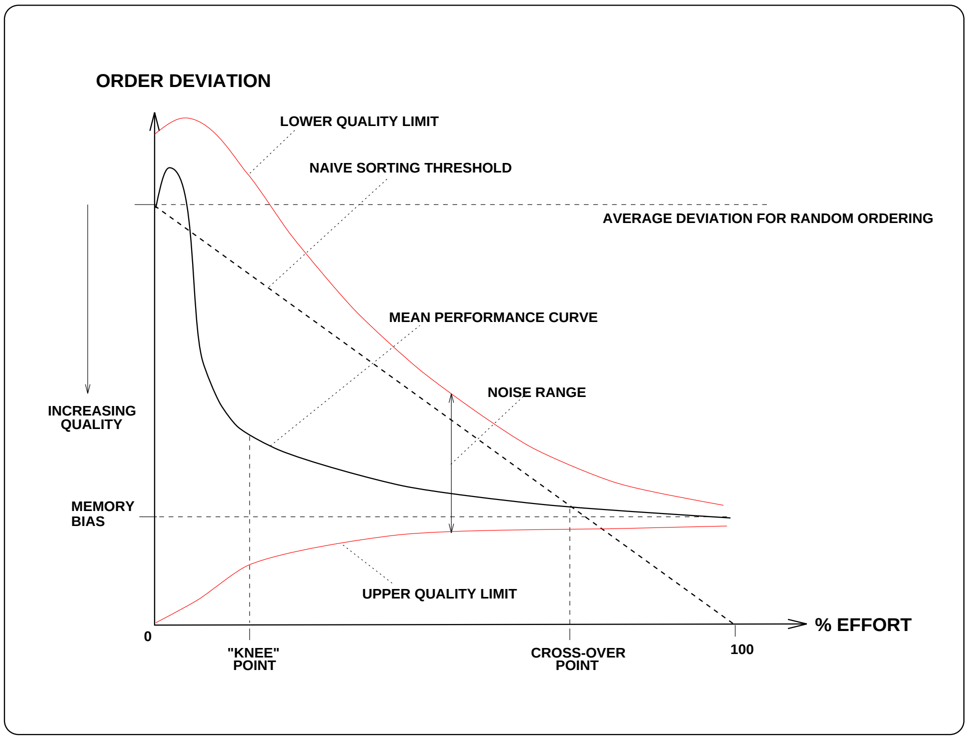


Figure 6.7: The General Relationship Between Retrieval Quality (Order Deviation) and Retrieval Effort

partially order the source cases in memory. Whether a net gain in performance is achieved in reality also depends on how the costs of the basic computational operations of CRASH compare to those of a given sort algorithm. Furthermore, if the mean performance curve has the same shape for all sizes of case base, CRASH will at best provide a constant factor speed up; the amount of effort required to attain a given level of quality will still be dependent on the number of cases in memory as the size of retrieval space grows proportional to the number of cases.

It is, therefore, important to compare the mean performance curves for each size of case base used in this experiment. This is done by defining the *"knee"* point to be the point at which the mean performance curve 'levels off'[8]. For the graphs presented in appendix P the following estimates can be made:

- 8 Cases: "Knee" point ≈ 19.5 %
- 13 Cases: "Knee" point ≈ 16.0 %
- 22 Cases: "Knee" point ≈ 7.0 %

[8]For the graphs of appendix P the "knee" point of each mean performance curve is taken to be the first major peak in quality as effort increases from zero. In the more ideal situation, where the mean performance curve is noise-free, the "knee" point must be defined in terms of cost-effectiveness; by reference to the first derivative of the mean performance curve a point can be defined at which the rate of change in quality per unit effort has reduced to some cut-off threshold.

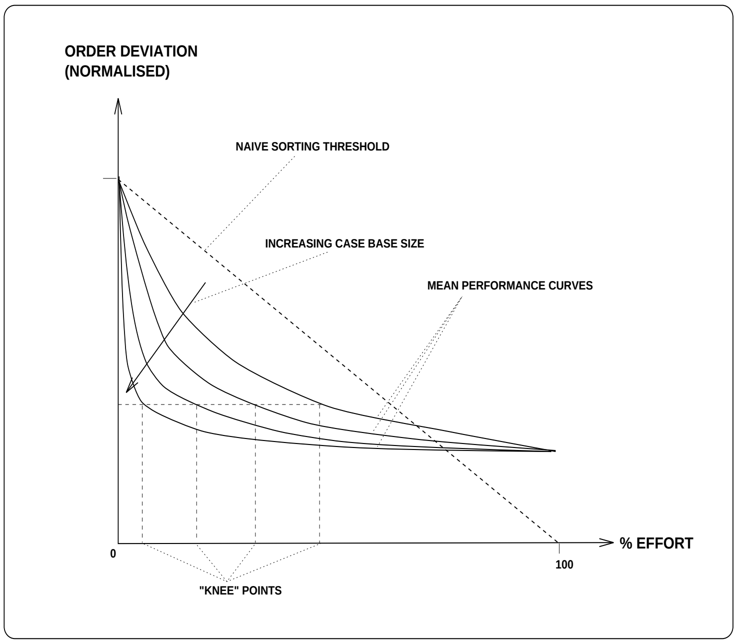


Figure 6.8: The Variation of the Mean Performance Curve with Case Base Size

The trend shown is that the "knee" point is, as desired, inversely dependent on case base size. This is schematically shown in in figure 6.8. The simplest relationship that would explain this is:

$$KneePoint = \frac{l}{N_c^r} \tag{6.1}$$

where N_c is the number of cases in a case base and l and r are positive constants. Determining the exact values for this relationship requires experimentation over a range of case bases that is far more extensive than was possible for the work done to support this thesis. However, a preliminary estimate from the three case bases used gives r somewhere between 0.57 and 2.5.

"Top 3" Results

In order to analyse the results of experiment 2 in terms of the quality of the *"top 3"* retrieved cases the metric Q_3 is defined in terms of how much the three most highly activated cases deviate from their respective positions in the appropriate bench-mark ordering, (see section O.2.2). The range of Q_3 is $-1 \leq Q_3 \leq 1$ where 1 corresponds to an exact match with the bench-mark and -1 corresponds to a complete contradiction.

The general pattern found when applying Q_3 to the results of experiment 2 is shown in figure 6.9. The quality versus effort relationship is highly similar to that as measured by order deviation. Again the mean performance curve tends to the maximum attainable quality at a rate that generally declines as retrieval effort increases.

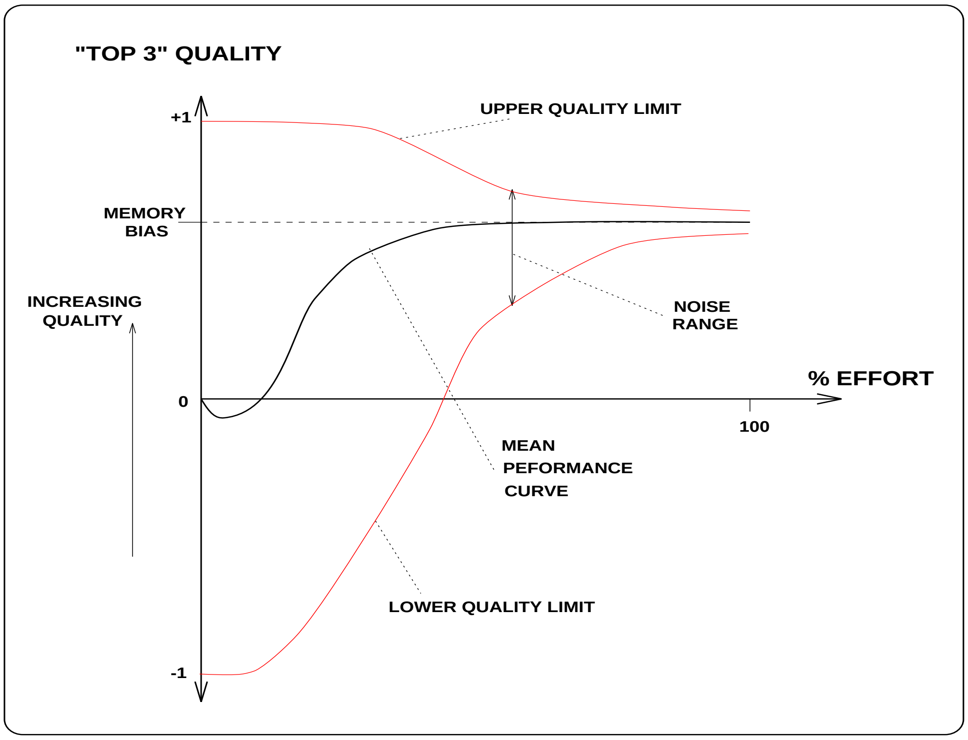


Figure 6.9: The General Relationship Between Retrieval Quality (Top 3) and Retrieval Effort

6.6.3 Experiment 2. Analysis of the Results

This section provides some further discussion on the key observations made from the above experimental results.

The Memory Bias

It has been observed that the quality of CRASH's retrieval tends to a sub-optimal maximum with respect to any given bench-mark. The simplest explanation of this is that, in order for retrieval to provide a compromised biasing towards the competing constraints it cannot fully comply with any one individual bench-mark.

Evidence for Scalability

As mentioned above, the general form of the mean performance curve for CRASH retrievals suggests that for an individual retrieval a high quality of retrieval is attainable at well below exhaustive effort. In addition, of importance is the observation that the *"knee"* point for the curve appears to be in some way inversely dependent on case base size.

The size of the retrieval search space defined for experiment 2 grows in a way that is linearly proportional to the number of source cases held in memory (see page 235).

However the proportion of the search space that needs to be searched to retrieve a high (if sub-optimal) quality of retrieval appears to have an inverse relationship proportional to the number of cases in the case base. Combining these two observations the following prediction can be made:

> The *absolute* amount of effort required for a CRASH retrieval to reach a high level of quality is a slowly growing function of the number of source cases held in memory.

Defining the exact nature of the relationship between the number of cases and the required effort is, as noted previously, beyond the scope of this thesis. What has been established is that it is expected to lie somewhere between being constant and being directly proportional to the number of cases in memory. That the relationship is predicted to grow slowly with increased case base size is encouraging evidence that the memory model embodied in CRASH will scale-up for large applications.

A question that arises is why low levels of retrieval effort produce disproportionately high levels of retrieval quality. Part of the answer lies in the fact that it is not only the mechanism of retrieval but the representation and structure of memory that is designed to bias retrieval towards selecting 'similar' source cases.

As argued in section 5.2.1, semantic similarity between two cases can be cast in terms of the relative number of connective paths between their features. For retrievals involving a low amount of effort the paths from the target case are selected effectively at random[9]. Probability dictates that most of these paths will lead to source cases that have a relatively high semantic similarity to the target case. The inherent biasing of CRASH's retrieval towards already activated parts of memory (see page 68) will amplify this effect as once one path to a source case node has been traversed the probability of traversing subsequent paths is greatly increased.

Robustness of CRASH

A similar pattern of results is observed for quality judged both in terms of the broad measure of order-deviation and the more focussed measure with respect to the three most highly activated cases of each retrieval. This is of particular importance as it suggests that the retrieval mechanism of CRASH is robust in terms of producing a useful gradation no matter how large a corpus of source cases is required by the post-retrieval stage of a CBR system. In this respect CRASH is a significant improvement on systems, such as described in [139, 82], which are geared towards selecting a single case from memory.

[9] For the experimental case bases relation weighting was not performed. By the arguments of chapter 2, it is reasonable to suppose that in suitably complex case bases the default graded structuring of memory produced by relation weighting will further increase the disproportionately high quality achieved by low effort CRASH retrievals.

The Bias Towards Useful Source Cases

It was stated in section 6.6.2 that the quality produced by CRASH's retrieval tends towards a maximal level (the memory bias) as retrieval effort tends towards 100 percent. As far as the evidence of the graphs in appendix P will allow, this statement seems to hold true for the semantic and structural similarity bench-marks. However, the statement is less clear-cut when it comes to considering the heuristic-based ordering of cases. The results seem to suggest a possible decline in retrieval quality with respect to this bench-mark beyond a certain degree of effort (for example, see figures P.6 and P.7 on page 247).

Part of the explanation is that the representation of bin-packing cases was designed so as not to include features that relate directly to the problem-solving purpose of the case. However some case features have more relevance to the problem than others. For example, from the heuristics given on page 6.2.2, *large L-shapes* are more important to consider when solving the bin-packing problem than *small rectangles*.

Amongst the collection of all source cases the frequency and relevancy of case features do not coincide. For example, there are approximately half as many *large L-shapes* in the 22 source cases as there are *small rectangles* (see appendix M). It follows that, as retrieval effort increases there may be a tendency to base retrieval on an increasing proportion of irrelevant case features. For example, when retrieval effort exhausts all paths to *large L-shapes*, further effort will be based mainly on *small rectangles* (and other types of shape) and hence further effort will tend to degrade retrieval with respect to the heuristic ordering.

The conclusion is that, while it may be expected that the bias towards semantic and structural similarity will generally strengthen with increased retrieval effort, the bias towards heuristic ordering will generally involve a series of local maxima and minima reflecting the mismatch between the frequency and relevancy of case features. This problem is avoidable by basing retrieval only on features that are recognised as relevant to the purpose of the retrieval. This is achievable in CRASH through the use of a more focussed sequence of context selections than that employed in the experimental work of this chapter (see section 7.2.1).

6.7 Results Summary

The experimental results described in this chapter have provided preliminary empirical evidence that the memory model implemented in CRASH satisfies some of the the key requirements laid down in this thesis. In particular, it has been confirmed that, as proposed in chapter 4, activation propagation can be a suitable practical technique by which an assessment of similarity can be formulated and transported *during* retrieval.

CRASH has been shown to have a consistent preference towards semantically and structurally similar cases. Evidence has been presented that suggests this biasing is robust, both to variation in some of the parameters that characterise CRASH's retrieval and to variation in case base size. Furthermore, this biasing has been seen to apply across the range of all source cases that are retrieved and also in identifying the few most salient cases in memory.

The argument that similarity between cases can be used to help judge the problem-solving usefulness of a source case has also received some support. For all experiments retrieval has shown some bias towards the heuristic ordering bench-mark. However, the limitations of a purely similarity based approach have been recognised. In particular, as might be expected, the biasing has been seen to decline as the proportion of irrelevant features used during retrieval increases. The conclusion is that pragmatic similarity, in terms of the selection of *relevant* features by context, also has an important role to play in CRASH.

Finally, and perhaps most importantly, evidence has been provided that the memory model proposed in this thesis is scalable. Exhaustive feature-based search is shown to be unnecessary to produce a high standard of retrieval quality for a given case base. The preliminary experiments also provide suggestive evidence that the required amount of effort to perform high quality, selective retrieval may grow at a rate significantly less than a linear dependency on case base size.

Encouraging though these preliminary results are, there remain many open areas for investigation using CRASH. The future use and development of CRASH is discussed in the next chapter after a short and concluding summary of the thesis.

Chapter 7

Conclusions and Future Work

In this concluding chapter the themes of the thesis are drawn together through a restatement of the goals of the research and a summary of the major contributions of this work. These contributions are put into context by a comparison to the few most closely related research projects. In addition, the use of alternative techniques, specifically neural networks and genetic algorithms, is considered for the task of case retrieval.

Possible future directions for this research are then detailed. A number of experiments using the existing CRASH system are outlined, concerning the additional testing of constraints that influence the search of memory and moves towards the use of CRASH for more large scale applications. Finally, the dynamics of context selection and the automation of input parsing are identified as two important areas for future development in a move towards a complete model of memory.

This chapter draws the thesis to a conclusion by establishing what has been achieved in the development of the memory model of CRASH and by identifying a number of promising directions for future development.

7.1 Modelling Memory

This section starts with a brief overview of the work described in this thesis on the development of a model of case memory. This is followed by a consideration of alternative approaches to the problem of case retrieval, both in terms of the most closely related systems to CRASH and more broadly, in terms of alternative AI methodologies that may provide interesting new approaches to the problem.

7.1.1 Thesis Summary

It is first important to restate the objectives of this thesis, as laid down in chapter 1. A synopsis is then given of the major contribution made in this thesis.

The Goals of the Thesis

This thesis has been concerned with the AI paradigm of knowledge-based systems. Despite their widespread popularity, KBSs still suffer from a number of technological difficulties including; fragility of expertise, inapplicability in weak theory domains, and difficulty of construction (see section 1.2.2). It is partly because case-based reasoning offers a solution to these types of problem that its popularity as a KBS technology continues to grow rapidly.

Traditional approaches leave a lot to be desired of CBR in terms of a model of reasoning; systems are typically *ad hoc* in nature and generally applicable principles are hard to extract. Indeed, many exponents of indexing-based CBR view this type of domain specific tailoring as a fundamental necessity [2, 5, 58, 116, 145]. However, in order that CBR may be based on more solid theoretical grounds, there is a need to establish an underlying model that is both widely applicable and moves towards a unification of the various identifiable components of the CBR process.

The approach taken here has been to view CBR as a manifestation of a more basic model of memory. The general requirements on this memory model are: that it is non-restrictive, in terms of the type of knowledge that can be stored and retrieved; that the access of cases (and other types of knowledge) be highly flexible and that the retrieval process avoids an exhaustive search of memory. A model of memory has been developed and, as was described in chapter 6, its feasibility has been practically demonstrated.

Major Contributions

A number of issues concerning knowledge representation have been addressed in the course of this thesis. Firstly, there is the problem of selecting granularity, i.e. what should constitute the 'basic unit of knowledge' for the representation of a case. This question is considered undecidable for a general model of memory (see section 2.3). The adopted solution was to introduce the *structure link* as a means of supporting multiple levels of granularity within a single network representation. In addition, the structure link has proven to have an important role in retrieval as it allows constraints concerning the morphology of case descriptions to be incorporated by a simplistic retrieval mechanism (see section 4.2.2).

The type of knowledge used in CBR is, by nature, limited in its scope of applicability. Accordingly, the need for segregation in the chosen network representation has been satisfied. Firstly, the *context link* has been introduced to represent a generalised notion of context (see section 2.4). The role identified for context in retrieval is to provide a dynamic constraint that is superimposed on the network representation (see section 4.3.3) in a way that can be viewed as a generalisation of traditional "indexing" techniques.

Secondly, the *template* has been introduced as a frame-like structure for bounding the description of individual knowledge items in the network (see section 2.5). Templates are embedded in an 'intimate' way in the network of memory. In addition, the inheritance hierarchy that links templates is highly general and is orthogonal to the knowledge represented in memory. These characteristics make the use of templates more flexible than previous frame-based approaches to CBR.

In order to 'fine-tune' the structuring of memory, a novel relation weighting scheme was introduced in section 2.6, based on two statistically derived factors (*typicality* and *idiosyncrasy*) and an iterative propagation technique that allows these factors to be combined over an area of the network. Proof of the utility of these weighting factors has been given.

In general, marker passing through a local network representation can be considered to be a form of symbolic reasoning. One of the most fundamental operations of this type of reasoning, and one that is pertinent to feature-based case retrieval, is the intersection of two disjoint search spaces (see section 3.3). It has been argued in chapter 3 that the the enhancements to the basic network representation, particularly context and relation weighting, play an important role in retrieval because they allow set intersection to be performed in a much more selective way than is possible in a flat network representation alone.

The task of case retrieval is one that cannot generally be performed by symbolic reasoning alone; source cases should be graded with respect to a number of constraining factors established by the target case. In accordance with this requirement, a novel activation passing mechanism has been developed in chapter 4. A number of technical innovations were made, driven by the need to marry the activation passing process with the general requirements for case retrieval. These innovations include: a means of providing disjoint activation distributions that includes *constructive interference* (see section 4.2.1); the use of the structure link to preserve both the coarse-grained and fine-grained structure of a target case during retrieval (see section 4.2.2) and a form of virtual inhibition that aids in the competitive selection of a corpus of source cases (see section 4.3). It has been argued (see section 4.4) that the resultant activation passing scheme provides a hybrid symbolic/numeric reasoning system that avoids some traditional problems concerning reliability and inefficiency. The reason identified for this is that the two types of reasoning act in intimate cooperation.

It has been argued in chapter 5 that the memory-based approach of this thesis is suitable as a basis for unifying the CBR framework. Its power stems from the symbolic reasoning that is performed at the time of retrieval by marker passing through the network representation. Accordingly, a number of traditionally post-retrieval processes can be recast as types of retrieval search, provided that the appropriate knowledge is held in memory. In this way, the proposed memory model is capable of subsuming both similarity-based and flexible indexing approaches to case retrieval (see section 5.2). In addition, mapping and retrieval are seen in section 5.3.1 to be synonymous. Finally, case adaptation has been considered as a manifestation of an iterative retrieval mechanism that, in general, uses fragments from a number of separate sources to fully satisfy the target case constraints (see section 5.3.2).

7.1.2 Alternative Approaches to Case Retrieval

This section is concerned with a consideration of possible alternative techniques to those developed in this thesis for application to the task of case retrieval. Firstly, a final comparison is made between the model of memory developed here and the most closely related research projects. This is followed by a brief consideration of how some more general techniques might be applied to the retrieval problem.

Comparable Models for Flexible Case Retrieval

There are currently three main research projects that have a close relationship to the work of this thesis, in terms of the type of constraints that are applied during case retrieval and the techniques used to achieve retrieval. The systems produced by these research projects are: the Structure Mapping Engine (SME) [43, 44, 50, 51] and its successor MAC/FAC [45, 47, 48]; ARCS [139] and its sibling ACME [138]; and REMIND [82] and its predecessor ROBIN [81]. These projects have already been considered at various points in the thesis. In this section, the related systems are briefly compared with respect to the two key requirements for a model of memory as laid down at the start of this thesis, namely *flexibility* and *efficiency*.

Flexibility has been achieved in the memory model proposed here by the use of a network representation that explicitly contains supporting world knowledge for cases, and by an activation-passing mechanism that is guided both by the static structuring of memory and by the dynamic selection of contexts.

Comparable flexibility is achieved in ARCS by a similar use of activation propagation as a means of evaluating the various types of constraint that can be imposed on memory. In ARCS the comparable process to dynamic context selection is that which generates a custom-built constraint network for each new retrieval [139].

In REMIND, flexibility is again achieved by the propagation of activation through a network of inter-connecting domain knowledge. Indeed, the flexibility of the mechanism is demonstrated in [81] as it is seen to provide high context sensitivity with regard to the exact knowledge content of the initial memory prompt. However, the pragmatic and structural constraints that are used in CRASH and ARCS to guide retrieval are lacking in REMIND, hence its flexibility may be achieved at the cost of generating many spurious retrievals.

MAC/FAC achieves flexibility by coupling a cheap and crude filter of memory, based on the use of content vectors, with the more rigorous selectivity performed by SME in 'literal similarity' mode [47]. However, MAC/FAC is fundamentally a two-stage process; the semantic constraints of its MAC phase are not conjoined with the primarily structural constraints of the FAC stage. In addition, the FAC stage is driven purely by a direct comparison of source cases to the initial memory prompt. It follows that contextual guidance and the guidance of intermediate world knowledge are lacking from the MAC/FAC retrieval mechanism. Hence, though the MAC system is flexible enough to produce psychologically plausible results [48], the delayed application of structural constraints detracts from the selectiveness of its preliminary retrieval stage (see page 119 for a clarification of this point).

With respect to efficiency, it is interesting to note that, of the four approaches, only the memory model proposed here does not assume a reliance on massive parallelism in retrieval. For ARCS performance is particularly poor because of the need to construct the constraint network for each retrieval. In the worst case, the amount of work that needs to be done for this process is $O(N_c^2)$ complexity (where N_c is the number of cases in memory) and the very least that can be expected is $O(N_c)$ complexity [139]. Similarly, the retrieval effort of MAC/FAC is linearly dependent on the number of cases in memory, though the amount of effort expended in the matching per case is very much less than for ARCS. In REMIND, activation propagation is performed until the network of memory settles in a stable state, therefore, the amount of retrieval effort required in REMIND is also directly related to the size of memory.

The issue of the amount of effort required for retrieval is fundamentally different in CRASH as retrieval effort is parameterised and hence the amount of effort actually expended can be predetermined. The knowledge representation of CRASH has been specifically designed to facilitate selective retrieval and this removes the need for an exhaustive search of memory. In CRASH, the issue is no longer one of how rapidly the retrieval process can search through all of memory but how good are the retrieval results achieved at a specified fraction of exhaustive effort. The preliminary empirical investigations of this issue are encouraging, high quality is generally achieved at low degrees of retrieval effort and the absolute amount of effort required to achieve a given level of quality appears to grow less rapidly than a linear dependency on case base size.

Promising General Techniques

In this section a brief consideration is given of the suitability of alternative AI paradigms for the task of case retrieval.

If each source case in memory is viewed as a category in its own right, the task of case retrieval can be thought of as a feature-based classification problem[1]. For the classification to be performed, a number of potentially conflicting constraints (such as maintaining semantic and structural similarity while also responding to any specific purpose-directed requirements) must be satisfied as far as is possible. This is the type of problem for which the use of neural networks is generally highly beneficial.

Research into the potential use of neural networks for the task of case retrieval is surprisingly scarce, with [141] providing a notable exception. A reason for this may simply be a reluctance to break from the legacy of the traditional approaches to CBR, where indexing has been the dominant retrieval paradigm and post-retrieval processes have been predominantly viewed as knowledge-intensive, symbolic reasoning tasks.

A rejection of neural networks because of their inability to perform complex symbolic reasoning may only be founded on practical grounds. There is a growing trend towards research into neural networks that exhibit some symbolic reasoning capabilities [142, 55].

[1]Some evidence for the psychological relevance of this assumption is given in [101].

In addition, there is no fundamental reason why neural networks cannot be embedded as a component working in conjunction with KBS modules of a hybrid system [109].

A more serious difficulty prohibiting the direct use of neural networks for case retrieval is that for most CBR systems the collection of cases in memory will evolve as the system is used. This severely detracts from the possibility of using a neural network to directly map from a target case description to a selected source case in memory because each time the stock of source cases changes the network will require retraining. However, a potential niche for neural networks is identified here in terms of the interpretation of "surface" features to produce more "abstract" features on which to base retrieval.

In the discussion of pages 81 to 82, it was seen that abstract features, such as proverbial advice, are potentially powerful for guiding source case retrieval. However, for a flexible retrieval mechanism it is necessary for the memory itself to associate the surface features of a case with the appropriate abstract features. The recognition of the abstract features applicable for a target case is, again, a difficult feature-based classification task. However, the vocabulary of abstract features associated with a given knowledge domain is much more static than the stock of domain cases held by the system. It follows that it is more feasible to build and train a neural network that is capable of recognising appropriate abstract features for a new target case. An over-view of the potential role of neural networks in case retrieval is depicted in figure 7.1.

Another recent, innovative technique for classification is the use of genetic algorithms [71, 72]. A genetic algorithm approach to case retrieval might treat all source cases in memory as an initial population that progressively evolves to produce a population of source cases most applicable to the target case. The constraints of the target case, therefore, provide the fitness function used to selectively establish each successive generation of source cases.

At first sight the genetic algorithm approach is unappealing because of the computational costs involved; for a pure genetic algorithm approach to retrieval the initial population may need to encompass the whole of memory. However, the technique has appeal in terms of the high flexibility it may provide. In particular, the genetic operators that mutate and merge members of the population will cause the final 'retrieved' source case population to be made up of members that are adaptations and composites of the original source cases [36]. By similar arguments to those of section 5.3.2 it can be seen that a potential benefit of a genetic algorithm approach is the unification of case adaptation and case retrieval.

7.2 Future Experimental Evaluation of CRASH

The experiments described in chapter 6 mark a beginning to the empirical investigation into case retrieval that may be performed using CRASH. This section gives a brief description of some possible areas for future experimentation.

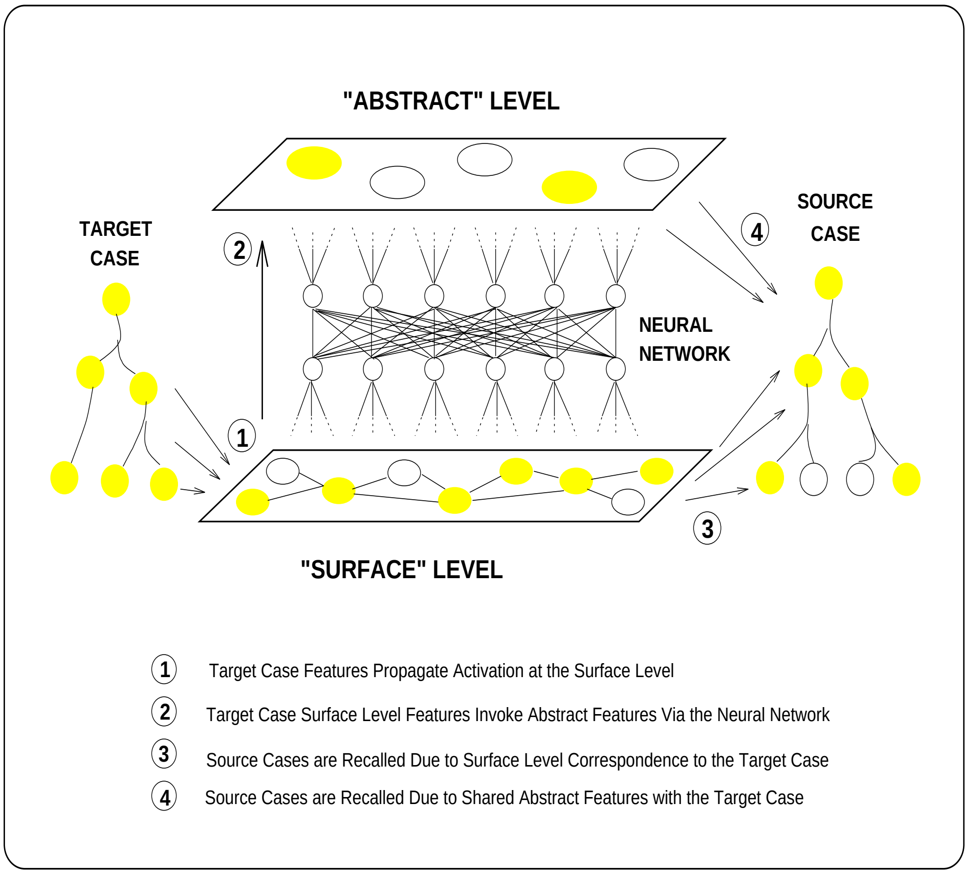


Figure 7.1: The Use of a Neural Network to Recognise Abstract Features so as to Facilitate Case Retrieval

7.2.1 Experimentation in the Short Term Future

A number of directions for further investigations are feasible using the existing bin-packing application.

Defining the Role of Pragmatic Knowledge

Due to its intimate combination of contextual constraints and numeric processing, the CRASH prototype provides a unique opportunity for gauging the role that pragmatic knowledge plays in the process of retrieval.

Recall that for the experiments described in chapter 6 a precompiled context sequence is used to guide phase 2 of retrieval and that the sequence used does not discriminate against any of the target case features. In other words, the retrieval performs an unfiltered measure of semantic similarity. In contrast, a tacit assumption of indexing strategies is that a large amount of purpose-directed knowledge is required in order to facilitate efficient retrieval.

The need for purpose-directed indexing may be qualified using CRASH through a systematic variation of the context sequence used to control phase 2 of retrieval. For each variant a different type of target case feature may be omitted from consideration during retrieval. The results produced can be compared to those of chapter 6 in order to establish the effect of the omitted feature in determining the quality of the resultant retrieval.

Iterative Retrieval

As presented in this thesis the three phases of CRASH's retrieval process are distinct. In particular, the amalgamation of feature-based retrievals during phase 3 occurs after the completion of the search of the network of memory in phase 2. In this sense, CRASH suffers from the same weakness as other systems, such as MAC/FAC [47], ANAGRAM [33] and PARADYME [80], that treat retrieval as a process that is fundamentally multi-staged. By delaying the application of the constraints applied in the more selective later stages, a lot of wasted effort is expended in the earlier stages.

The fundamental difference which sets CRASH apart is that the preliminary selection step takes place by forging feature-to-feature paths through the network of world knowledge rather than selecting whole cases. This distinction is important as it offers a potential improvement in CRASH's performance by running phase 3 and phase 2 in a lock-step mode.

Assume that activation is propagated up a source case description whenever one of its component nodes receives activation. Given that the propagation of activation through the network of world knowledge is biased towards nodes already possessing activation, a disproportionate number of subsequent paths will lead to the initially activated source cases. This process needs to be controlled so that the consequences of initial single-feature retrievals are not overly exaggerated. Nevertheless, in principle the lock-step mode of operation will allow structural constraints to provide feedback to the activation spreading process that searches memory[2].

It follows that one area of future experimental work is to test this lock-step mode of operation and compare it to the results presented in chapter 6. In particular, it will be of interest to determine whether structurally-based feedback can speed the relative rate at which CRASH's retrieval process produces a high quality gradation of source cases.

7.2.2 The Need for More Complex Test Applications

The test application of chapter 6 is (by necessity) relatively simple, so that the behaviour of the most basic aspects of the proposed memory model can be understood. However, this type of simplified problem domain is insufficient to test all aspects of CRASH. For example, forming a clear distinction between the roles of structural and semantic similarity in retrieval requires a more heterogeneous knowledge base than is offered by the bin-packing problem. An interesting test-bed for future experimentation with CRASH may be provided by the "Karla the Hawk" stories [46] used by Gentner et al in the testing of SME and MAC/FAC. These stories form a set of short descriptive passages which differentiate from each other along the separated dimensions of surface-level (i.e. semantic similarity) and structural similarity.

There is also a need to test CRASH on applications where the search space between target and source cases is more indirect than for the bin-packing problem of chapter 6. A retrieval that needs to search a network of world knowledge to find source cases must traverse a much more extensive search space than that defined for the bin-packing problem. The memory model proposed here relies on the combination of contextual constraints and relation-weighting to handle such massive search spaces in a tractable manner, hence an application must be sought that allows these factors to be fully tested. This ties in with the long term aim of applying CRASH to a real-world problem.

7.3 Towards a Complete Model of Memory

The work of this thesis has provided some important insights into the operation of a memory designed to implement selective case retrieval. However, the model of memory is capable of being expanded and in this section two important areas for future developments are considered.

7.3.1 Towards a Methodology for Context Selection

The dynamic selection of contexts has been advocated in this thesis as an alternative to the inflexible organisation of a case base with respect to predefined indices. Context

[2]This type of cyclic retrieval process, whereby the initial retrieval of some knowledge item focuses the attention of memory towards the additional features of that knowledge item, may provide a general memory-based reasoning approach to the problem of feature-based recognition [41, pp 51-58].

provides an extra dimension of control by which a network representation can be segregated. Precompiled context sequences, such as used in the experimentation of chapter 6, can be applied by an external controller so as to superimpose a selective retrieval strategy upon the memory model. However, the use of externally defined context sequences partly contradicts with a purely memory-based reasoning stance because some knowledge, namely the control knowledge embedded in the context sequence, is held externally to main memory.

An extremely interesting and open research issue is whether the selection of active contexts can be performed internally by the memory itself. This requires two related issues to be investigated: how control knowledge should be represented *within* memory and what the relationship is between the transient distribution of activation within memory and the selection of active contexts.

Whether it is even possible to define a simple mathematical relationship that relates contexts to the activation distribution is not clear. Some tentative experimentation has been performed using CRASH for context selection based on a combination of factors such as:

- Permitting activation to flow from the most highly activated node(s) in memory.
- Permitting as much activation as possible to flow.
- Permitting activation to flow from as many nodes as possible.

There is a fundamental problem in using a combination of this type of context selection criteria; invariably, the activation distribution eventually reaches a state in which it continually supports the selection of a single context or a repetitive cycle of contexts. The indication is that, to avoid memory becoming 'preoccupied' in this manner, a second-order dependency of context selection on the activation distribution is required, in order to discourages the incessant selection of the same contexts. One potential direction for this future research is to attempt to adapt research into the dynamic behaviour of neural networks, such as [98, 76, 13, 146, 99], for the task of context selection.

7.3.2 Memory Based Parsing

In [82], systems such as CRASH that rely on the hand-coding of cases are rightly criticised for their incompleteness as a model of memory. Ideally, CRASH should be able to receive input in some restricted natural language format and automatically assimilate the description into the network representation of memory. Systems such as REMIND [82] and DMAP [88] provide impressive models of how a memory-based approach may be used for this parsing problem. Such approaches are particularly appealing because they explain the way in which the extensive knowledge required to disambiguate the meaning of a natural language phrase can naturally be brought to bear as an integral part of the parsing process.

Exactly how the parsing process can be implemented for the memory model developed in this thesis is an area for future investigation. However, two general and complementary processes can be identified as required [81, 82, 88]:

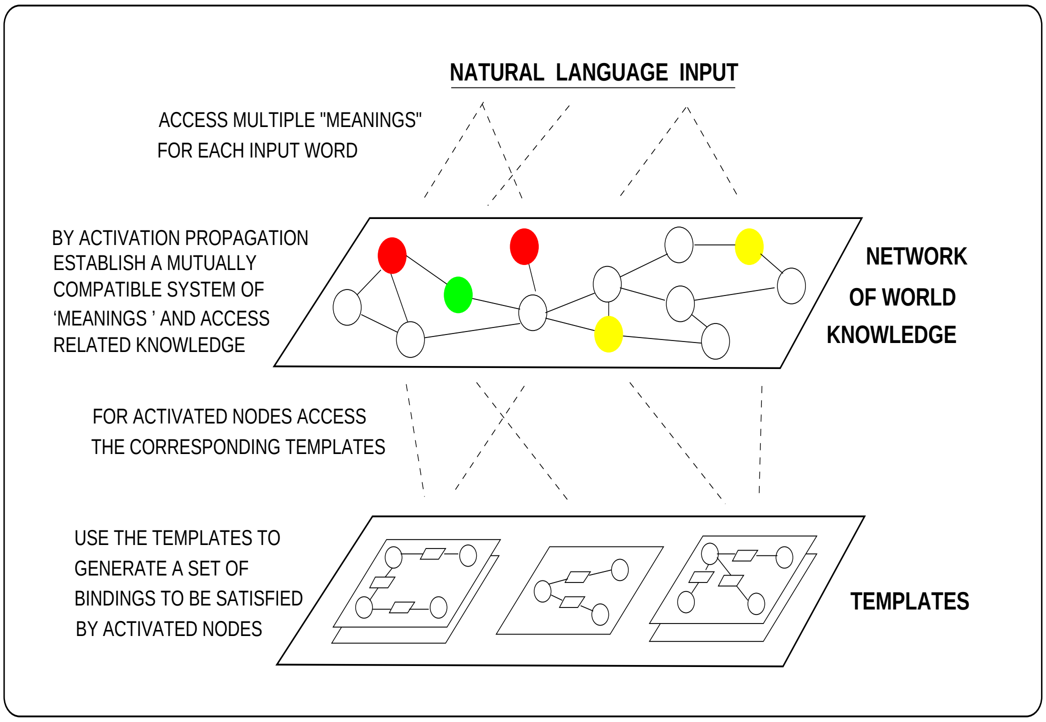


Figure 7.2: A Scheme for Automated Input Parsing in CRASH

- The system must be able to find a mutually compatible set of meanings for the individual input words. This requires the ability to search for indirect associations between the meanings of words.
- The system must be able to generate and satisfy binding-constraints as to how the retrieved knowledge items in memory should interact to form a coherent description.

The way in which these processes may correspond to the memory model of CRASH is depicted in figure 7.2. Each term in the input is assumed to link[3] to a set of internal nodes, where each node represents a possible meaning for that term. Generally, associated with each internal node will be a template representing its descriptive structure.

It follows that establishing an internal description for the input requires processing at two levels. Firstly, activation propagation through the network of world knowledge will form associations between the possible 'meanings' of terms and will uncover intermediate nodes that represent related knowledge items (which in turn will have their own descriptive templates). Secondly, the collection of associated templates establish expectations as to network pattern that is compatible with the selection of each 'meaning'.

The main area for future investigation is to develop a methodology that can select one 'meaning' per input term from the set of all possible meanings, such that a maximally consistent description is established with respect to the constraints imposed by

[3]The "label node" knowledge representation primitive already exists within the proposed model of memory and is intended to allow external 'stimuli' to be associated with the nodes of the internal network of memory, see appendix B.

the accessed templates. The development of a successful, automated parsing mechanism is a necessity if the proposed memory model is to be applied to large scale applications as it drastically reduces the cost of knowledge entry. The relative success of similar systems, particularly DMAP, is a strong indication that such a mechanism will be a feasible extension to CRASH.

7.4 Concluding Remarks

The philosophy behind the use of knowledge-based systems as a means for achieving artificial intelligence is in some ways epitomised by the *"knowledge is all there is"* hypothesis of Lenat [85]:

> "No sophisticated, as-yet-unknown control structure is required for intelligence; control strategies are knowledge, and a standard evaluator can apply them."

However, the explicit representation of knowledge cannot by itself achieve intelligence. Knowledge must be retrieved from memory whenever (and only whenever) circumstances dictate that the particular knowledge is likely to be useful. For the success of a knowledge-based approach to reasoning, the process of knowledge retrieval is as important as the knowledge itself.

In the relatively near future massive knowledge bases, such as that of the CYC project [86], will for the first time be available to researchers of artificial intelligence. With the advent of such potentially powerful resources, the problem of managing and retrieving the knowledge therein will be greatly amplified and consequently research into these issues will grow in significance.

In this thesis the problem of storing and retrieving knowledge from memory has been considered from the perspective of case-based reasoning. The model of memory has been based around some novel organisational principles embodied in the knowledge representation and around an accompanying activation-passing mechanism that has been specifically tailored to the requirements of case retrieval. The model provides high flexibility, in terms of the circumstances under which a case may be retrieved from memory, because the retrieval process has been cast as one that is *biased* towards a number of different types of constraint. The memory has also been designed so that retrieval does not rely on an exhaustive search of memory. Providing a model that, in principle, achieves both high flexibility and efficiency is the major contribution of this thesis. The work herein described, and its continued development, should have a direct influence on future research into case-based reasoning and memory-based reasoning in general.

Bibliography

[1] Peter Achinstein. *The Nature Of Explanation.* Oxford University Press, 1983.

[2] Richard Alterman and Mary Wentworth. Determining the Important Features of a Case. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp197–202. Morgan Kaufmann Publishers, 1989.

[3] John R Anderson. *A Theory of the Origins of Human Knowledge.*, pages pp664–683. Morgan Kaufmann Publishers, Inc, 1990.

[4] P Arabie and J D Carroll. MAPCLUS: A Mathematical Programming Approach To Fitting The ADCLUS Model. *PSYCHOMETRIKA*, 45:pp211–235, 1980.

[5] Kevin D Ashley. Indexing and Analytic Models. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp197–202. Morgan Kaufmann Publishers, 1989.

[6] Kevin D Ashley and Edwina L Rissland. Compare and Contrast, A Test Of Expertise. In *AAAI-87: Proceedings of the Sixth National Conference on Artificial Intelligence.*, pages pp273–278. Morgan Kaufmann Publishers, July 1987.

[7] William M Bain. A Case-Based Reasoning for Subjective Assessment. In *AAAI-86, Proceedings of the 5th National Conference on Artificial Intelligence.*, volume 1, pages pp523–527, Philadelphia, August 1986. Morgan Kaufmann Publishers.

[8] Ralph Barletta and William Mark. Explanation-based Indexing of Cases. In *AAAI-88: Proceedings of the Seventh National Conference on Artificial Intelligence.*, volume 2, pages pp541–546, Saint Paul, Minnesota, August 1988.

[9] Lawrence W Barsalou. Ad hoc Categories. *Memory and Cognition*, 11(3):pp211–227, 1983.

[10] Chumki Basu. Organizing Multiple Points of View in Episodic Memory. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp208–212, Pensacola Beach, Florida, 1989. Morgan Kaufmann Publishers.

[11] Lawrence Birnbaum and Gregg Collins. Reminding and Engineering Design Themes: A Case Study in Indexing Vocabularies. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp47–51, Pensacola Beach, Florida, 1989. Morgan Kaufmann Publishers.

[12] E.J. Borowski and J.M. Borwein. *Dictionary of Mathematics.* Collins, 1989.

[13] Fernanda Botelho and Max Garzon. Global Dynamics in Neural Networks III. In *IJCNN: Proceedings of the International Joint Conference on Artificial Intelligence.*, pages pp341–348, Seattle, WA, 1991. Institute of Electrical and Electronic Engineers. Extended Abstract.

[14] Ronald J Brachman. The Future of Knowledge Representation. In *AAAI-90: Proceedings of the Ninth National Conference on Artificial Intelligence.*, pages pp1082–1092, 1990. Extended Abstract.

[15] Steven Bradtbe and Wendy G Lehnert. Some Experiments With Case-Based Search. In Janet Kolodner, editor, *Proceedings of the Case-Based Reasoning Workshop*, Florida, May 1988. Morgan Kaufmann Publishers.

[16] L. Karl Branting. Integrating Generalisations with Exemplar-Based Reasoning. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp224–229. Morgan Kaufmann Publishers, 1989.

[17] Mike G Brown. Case-Based Reasoning: Principles and Potential. Commercial Report, AI Intelligence, PO Box 95, Oxford, OX2 7XL, January 1992. Editor, Alex Goodall.

[18] Mike G Brown. Incorporating Similarity Measures Into Case Retrieval Using Analogue Marker Passing. In Bernd Neumann, editor, *ECAI-92, Proceedings of the Tenth European Conference on Artificial Intelligence.*, pages pp590–592, Vienna, Austria, August 1992. John Wiley and Sons, Inc.

[19] Mike G Brown. An Under-Lying Memory Model to Support Case Retrieval. In *EWCBR-93: The First European Workshop on Case Retrieval.*, Kaiserslautern, Germany, 1993. Springer-Verlag. IN PRESS: to appear in the workshop preprints.

[20] Mark H Burstein. Concept Formation by Incremental Analogical Reasoning and Debugging. In R S Michalski, J G Carbonell, and T M Mitchell, editors, *Machine Learning: an A.I. approach*, pages pp351–367. Morgan Kaufmann Publishers, 1986. vol II.

[21] Mark H Burstein. Combining Analogies in Mental Models. In D H Helman, editor, *Analogical Reasoning.* Kluwer Academic Publishers, Boston, 1988.

[22] Mark H Burstein. Analogy vs CBR: The Purpose of Mapping. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp133–135, Pensacola Beach, Florida, 1989. Morgan Kaufmann Publishers.

[23] Edward A Bushinski and Henry J Koren. *The Analogy of Names and The Concept of Being.* The Ad Press Ltd, New York, 1959. A literal translation of the work of Thomas De Vio, the Cardinal Cajetan, fifteenth century.

[24] A Camurri, M Frixione, G Vercelli, and R Zaccaria. Some Concepts on Analogic Planning in Assembly Tasks. In Luigia Carlucci Aiello, editor, *ECAI-90, proceedings of the 9th European Conference on Artificial Intelligence*, Stockholm, Sweden, August 1990. Pitman Publishing.

[25] Jaime G Carbonell. A Theory Of Reconstructive Problem Solving And Expertise Acquisition. In R S Michalski, J G Carbonell, and T M Mitchell, editors, *Machine Learning: an A.I. approach*. Morgan Kaufmann Publishers, 1986. vol II.

[26] David Waltz (chair). Panel Discussion on "Indexing Algorithms". In *Proceedings of the Case-Based Reasoning Workshop*, pages pp 25–45, Pensacola Beach, Florida, 1989. Morgan Kaufmann Publishers.

[27] David Waltz (chair), Charles Martin, Michael Pazanni, and Paul Thagard. Panel Discussion on "Indexing" Algorithms. In *Proceedings of the Case-Based Reasoning Workshop*, Pensacola Beach, Florida, 1989. Morgan Kaufmann Publishers.

[28] Lawrence Birnbaum (chair). Panel discussion on "indexing vocabulary". In *Proceedings of the Case-Based Reasoning Workshop*, pages pp46–65, Pensacola Beach, Florida, 1989. Morgan Kaufmann Publishers.

[29] B Chandrasekaran. Towards a Functional Architecture for Intelligence Based on Generic Information Processing Tasks. In *IJCAI-87 : Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages pp1183–1192, Milan, August 1987. Morgan Kaufmann Publishers, Inc.

[30] Eugene Charniak. Passing Markers: A Theory of the Contextual Influence in Language Comprehension. *Cognitive Science*, 7:pp171–190, 1983.

[31] Eugene Charniak. A Neat Theory of Marker Passing. In *AAAI-86, Proceedings of the Fifth National Conference on Artificial Intelligence*, pages pp584–588, Philadelphia, 1986. Morgan Kaufmann Publishing.

[32] Eugene Charniak and Drew McDermott. *Introduction To Artificial Intelligence*. Addison-Wesley Publishing Company, 1985.

[33] Diane J Cook. ANAGRAM, An Analogical Planning System. Technical Report UIUCDCS-R-89-1561, Department of Computer Science, University of Illinois, Urbanna-Champaign, December 1989.

[34] Diane J Cook. The Base Selection Task In Analogical Planning. In *IJCAI 91, Proceedings of the Twelfth International Joint Conference On Artificial Intelligence*, volume 2, pages pp790–795, Sydney, Australia, 1991. Morgan Kaufmann Publishers, Inc.

[35] Paul de Boek and Seymour Rosenberg. Hierarchical Classes: Models and Data Analysis. *PSYCHOMETRIKA*, 53(3):pp361–381, September 1988.

[36] Dwight Deugo and Franz Oppacher. Applications of Case-Based Reasoning Using Knowledge Base and Genetic Techniques. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp239–243, Pensacola Beach, Florida, 1989. Morgan Kaufmann Publishers.

[37] Joachim Diederich. An Explanation Component for a Connectionist Inference System. In Luigia Carlucci Aiello, editor, *ECAI-90, proceedings of the 9th European Conference on Artificial Intelligence*, pages pp222–227, Stockholm, Sweden, August 1990. Pitman Publishing.

[38] Thomas G Dietterich. Learning at the Knowledge Level. In *Readings In Machine Learning*, pages pp11–24. Morgan Kaufmann Publishers, Inc., 1986.

[39] Daniel M Ennis. Confusable and Discriminable Stimuli: Comment on Nosofsky(1986) and Shepard(1986). *The Journal of Experimental Psychology*, 117(4):pp408–411, 1988.

[40] Thomas C Eskridge. Continuous Analogical Reasoning: A Summary Of Current Research. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp253–257. Morgan Kaufmann Publishers, 1989.

[41] Scott E Fahlman. *NETL, A System For Representing And Using Real-World Knowledge*. The MIT Press, 1979.

[42] Scott E Fahlman and Geoffrey E Hinton. Connectionist Architectures for Artificial Intelligence. *Computer*, 20(1), January 1987.

[43] B Falkenhainer, K D Forbus, and D Gentner. The Structure Mapping Engine. In *AAAI-86 Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 272–277, Philadelphia, August 1986. Morgan Kaufmann Publishers.

[44] B Falkenhainer, K D Forbus, and D Gentner. The Structure Mapping Engine: Algorithm and Examples. *Artificial Intelligence*, 41(1):pp1–63, November 1989.

[45] Kenneth Forbus and Dedre Gentner. MAC/FAC: A Model of Similarity-based Retrieval, March 1993. Personal Communications.

[46] Kenneth Forbus, Dedre Gentner, and Mary Jo Rattermann. "Karla the Hawk" stories: supplement to The Roles of Similarity in Transfer: Separating Retrievability from Inferential Soundness., 1993. Personal Communications, In Press.

[47] Kenneth D Forbus and Dedre Gentner. MAC/FAC: A Model of Similarity-based Retrieval. In *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society.*, pages pp 504–509, Chicargo, Illinois, August 1991. Lawrence Erlbaum Associates.

[48] Kenneth D Forbus, Dedre Gentner, and Mary Jo Rattermann. The Roles of Similarity in Transfer: Separating Retrievability from Inferential Soundness. *Cognitive Science*, 1993. Personal Communications, In Press.

[49] Dedre Gentner. Analogical Inference and Analogical Access. Technical Report UIUCDCS-R-87-1365, Dept Of Computer Science, University of Illinois, Urbana-Champagne, USA, August 1987.

[50] Dedre Gentner, Brian Falkenhainer, and Janice Skorstad. Analogical Processing: A Simulation and Empirical Corroboration. In *AAAI 87: Proceedings of the Sixth National Conference on Artificial Intelligence*, volume 1, pages pp322–326, Seattle,Washington, 1987. Morgan Kaufmann Publishers, Inc.

[51] Dedre Gentner, Kenneth D Forbus, and Brian Falkenhainer. The Structure-Mapping Engine. Technical Report UIUCDCS-R-86-1275, Dept Of Computer Science, University of Illinois, Urbana-Champagne, USA, May 1986.

[52] Ashok K Goel, Janet L Kolodner, Michael Pearce, Richard Billington, and Craig Zimring. Towards a Case-Based Tool for Aiding Conceptual Design Problem Solving. In *Proceedings: Case-Based Reasoning Workshop.*, pages pp109–120, Washington, D.C., 1991.

[53] Arthur C Grasser and Temar Murachver. *The Symbolic Procedures Of Question Answering*, chapter 2. Lawerence Erlbaum Associates, 1985.

[54] Barbara J. Grosz. The Repesentation and Use of Focus in a System for Understanding Dialogue. In Barbara J. Grosz, Karren S. Jones, and Bonnie L. Webber, editors, *Readings in Natural Language Processing*, pages pp 353–362. Morgan Kaufmann Publishers, Inc, 1986.

[55] Robert F Hadley. Connectionism, Rule Following, and Symbolic Manipulation. In *AAAI-90: Proceedings of the ninth national conference on artificial intelligence.*, pages pp570–586, 1990.

[56] Kristian J Hammond. CHEF: A Model of Case-Based Planning. In *Proceedings of AAAI-86, the 5th National Conference on Artificial Intelligence*, volume 1, Philadelphia, August 1986.

[57] Kristian J Hammond. On Functionally Motivated Vocabularies: An Apologia. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp52–56. Morgan Kaufmann Publishers, 1989.

[58] Kristian J Hammond. Explaining and Repairing Plans That Fail. *Artificial Intelligence.*, 45:pp173–228, 1990.

[59] James A Hendler. Marker-Passing and Microfeatures. In *IJCAI 87: Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, volume 1, Milan, 1987. Morgan Kaufmann Publishers, Inc.

[60] James A Hendler. Marker Passing over Microfeatures: Towards a Hybrid Symbolic/Connectionist Model. *Cognitive Science*, 13:pp79–106, 1989.

[61] Daniel Hennessy and David Hinkle. Initial Results from Calvier: A Case-Based Autoclave Loading Assistant. In *Proceedings: Case-Based Reasoning Workshop.*, pages pp225–232, Washington, D.C., 1991.

[62] Thomas R Hinrichs and Janet L Kolodner. The Role of Adaptation in Case-Based Design. In *Proceedings: Case-Based Reasoning Workshop.*, pages pp121–132, Washington, D.C., 1991.

[63] Keith J Holyoak and Kyunghee Koh. Surface and Structural Similarity in Analogical Transfer. *Memory and Cognition*, 15(4):pp332–340, 1987.

[64] Ye Huang. *Relevancy Knowledge and Analogical Reasoning.* PhD thesis, University of Manchester Institute of Science and Technology, UMIST, 1992.

[65] Ye Huang and Alison E Adam. A Relevancy Model For Analogical Reasoning. In Bernd Neumann, editor, *ECAI-92, Proceedings of the Tenth European Conference on Artificial Intelligence.*, pages pp139–143, Vienna, Austria, August 1992. John Wiley and Sons, Inc.

[66] Ye Huang and Alison E Adam. Relevancy Knowledge in Analogical Reasoning, 1992. Personal Communication.

[67] Robert E Stepp III and Ryszard S Michalski. Conceptual Clustering: Inventing Goal-Orientated Classification Of Structured Objects. In R S Michalski, J G Carbonell, and T M Mitchell, editors, *Machine Learning, an AI Approach.* Morgan Kaufmann Publishers, inc., 1986. vol II.

[68] Robert E Stepp III and Ryszard S Michalski. Understanding The Nature Of Learning: Issues and Research Directions. In R S Michalski, J G Carbonell, and T M Mitchell, editors, *Machine Learning, an AI Approach.* Morgan Kaufmann Publishers, inc., 1986. vol II.

[69] Alan Jeffrey. *Mathematics for Engineers and Scientists.* Van Nostrand Reinhold, 1989.

[70] Eric Kimball Jones. Case-Based Analogical Reasoning using Proverbs. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp275–279. Morgan Kaufmann Publishers, 1989.

[71] Kenneth A. De Jong and William M.Spears. Learning Concept Classification Rules Using Genetic Algorithms. In *IJCAI 91, proceedings of the 12th International Joint Conference On Artificial Intelligence*, pages pp651–656, Sydney, Australia, 1991. Morgan Kaufmann Publishers, Inc.

[72] James D Kelly (Jr) and Lawerence Davis. A Hybrid Genetic Algorithm for Classification. In *IJCAI 91, proceedings of the 12th International Joint Conference On Artificial Intelligence*, pages pp645–650, Sydney, Australia, 1991. Morgan Kaufmann Publishers, Inc.

[73] Subbarao Kambhampati. Representational Requirements for Plan Reuse. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp20–23. Morgan Kaufmann Publishers, 1989.

[74] Alex M Kass and David Bleake. Case-Based Reasoning Applied to Constructing Explanations. In Janet Kolodner, editor, *Proceedings of the case-based reasoning workshop*, Florida, May 1988. Morgan Kaufmann Publishers.

[75] Smadar Kedar-Cabelli. *Towards a Computational Model of Purpose-Directed Analogy*, pages pp647–656. Morgan Kaufmann Publishers, Inc, 1990.

[76] A Kehagias. The Local Backward-Forward Algorithm. In *IJCNN: Proceedings of the International Joint Conference on Neural Networks.*, pages pp321–326, Seattle, WA, 1991. Institute of Electrical and Electronic Engineers.

[77] Frank C Keil. *Conceptual Development and Category Structure*, chapter 7. Cambridge University Press, 1987.

[78] Hiroaki Kitano, James Hendler, Tesuya Higuchi, Dan Moldovan, and David Waltz. Massively Parallel Artificial Intelligence. In *IJCAI 91, Proceedings of the Twelfth International Joint Conference On Artificial Intelligence*, pages pp557–562, Sydney, Australia, 1991. Morgan Kaufmann Publishers, Inc. (Panel Discussion).

[79] Janet L Kolodner. Judging Which is the Best Case for a Case-Based Reasoner. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp77–81. Morgan Kaufmann Publishers, 1989.

[80] Janet L Kolodner. Selecting the Best Case for a Case Based Reasoner. In *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society.*, pages pp155–162, Ann Arbor, Michigan, August 1989. Lawrence Erlbaum Associates.

[81] Trent E. Lange and Michael G. Dyer. Frame Selection in a Connectionist Model of High-Level Inferencing. In *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society.*, pages pp 706–713, Ann Arbor, Michigan, August 1991. Lawrence Erlbaum Associates.

[82] Trent E Lange and Charles M Wharton. Analysis of an Integrated Comprehension and Episodic Memory Retrieval Model. In *IJCAI-93: Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence.*, pages pp208–213, Chambéry, France, 1993.

[83] Pat Langley and John A Allen. Learning, Memory and Search Planning. In *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*, Chicargo, 1991. Lawrence Erlbaum.

[84] Wing Lee and Dan Moldovan. The Design of a Marker Passing Architecture for Knowledge Processing. In *AAAI-90: Proceedings of the Ninth National Conference on AI.*, pages pp570–586, 1990.

[85] Douglas B Lenat and Edward F Feigenbaum. On the Thresholds of Knowledge. In *IJCAI 87: Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages pp1172–1176, Milan, August 1987. Morgan Kaufmann Publishers.

[86] Douglas B Lenat, Ramanathan V Guha, Karen Pittman, Dexter Pratt, and Mary Shepherd. CYC: Towards Programs with Common Sense. *Communications of the ACM*, 33(8):pp30–49, August 1990.

[87] Iwayama Makoto, Tokunaga Takenobu, and Tanaka Hozumi. A Method of Calculating the Measure of Salience in Understanding Metaphors. In *AAAI-90: Proceedings of the Ninth National Conference on Artificial Intelligence.*, pages pp298–302, 1990. Extended Abstract.

[88] Charles Eugene Martin. *Direct Memory Access Parsing.* PhD thesis, Yale University, 1991.

[89] John McCarthy. Applications of Circumscription to Formalizing Common-Sense Knowledge. *Artificial Intelligence, an International Journal*, 28(1):pp89–116, February 1986.

[90] James L McClelland and David E Rumelhart. An Interactive Activation Model of Context Effects in Letter Perception: Part 1, An Account of Basic Findings. *Psychological Review*, 88:pp375–407, September 1981.

[91] Kathleen McCoy. Generating Context-Sensitive Responses To Object-Related Misconceptions. *Artificial Intelligence, an International Journal*, 41(2), December 1989.

[92] Kathleen R McKeown. Discourse Strategies for Generating Natural Language Text. *Artificial Intelligence*, 27(1):pp1–41, September 1985.

[93] Michael McTear. *The Articulate Computer.* Basil Blackwell, in association with Andre Deutsh, 1987.

[94] Marvin Minsky. *The Society of The Mind.* Simon and Schuster, Inc., 1986.

[95] Tom M Mitchell. *The Need for Biases in Learning Generalisations*, pages pp184–190. Morgan Kaufmann Publishers, Inc, 1990. Originally published as a Technical Report from Rutgers University, 1980.

[96] Dan Moldovan, Wing Lee, Changhwa Lin, and Minhwa Chung. SNAP: Parallel Processing Applied to AI. Technical Report PKPL 91-3, University of Southern California, Dept. of Electrical Engineering Systems, University of Southern California, University Park, LA, August 1991.

[97] Enrico Motta, Tim Rajan, and Marc Eisenstadt. A Methodology And Tool For Knowledge Acquisition In Keats-2. Technical Report HCRL tech' report no.32, OPEN UNIVERSITY, Milton Keynes MK7 6AA, March 1988.

[98] George J. Mpitsos, Robert M. Burton, and Mark O Hatfield. Analogies of Brain Functions in Neural Network Models. In *IJCNN: Proceedings of the International Joint Conference on Artificial Intelligence*, volume 1, pages pp 181–182, San Diego, California, 1990. Institute of Electrical and Electronic Engineers.

[99] Shigetoshi Nara and Peter Davis. Chaotic Wandering and Search in a Cycle-Memory Neural Network. *Progress of Theoretical Physics*, 88(5):pp 845–855, November 1992.

[100] Nils J Nilsson. *Principles of Artificial Intelligence*. Springer-Verlag, 1982.

[101] Robert M Nosofsky. Attention, Similarity and the Identification-Categorization Relationship. *The Journal of Experimental Psychology:General*, 116(1):pp39–57, 1986.

[102] Robert M Nosofsky. ON Exemplar-Based Representations: Reply to Ennis (1988). *The Journal of Experimental Psychology:General*, 117(4):pp412–414, 1988.

[103] Christopher Owens. Domain-Independent Prototype Cases for Planning. In Janet Kolodner, editor, *Proceedings of the Case-Based Reasoning Workshop*, Florida, May 1988. Morgan Kaufmann Publishers.

[104] Christopher Owens. Plan Transformations as Abstract Indicies. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp62–65. Morgan Kaufmann Publishers, 1989.

[105] Michael J Pazzani. Indexing Strategies for Goal Specific Retrieval of Cases. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp31–35. Morgan Kaufmann Publishers, 1989.

[106] Roger Penbrose. *The Emperor's New Mind; concerning computers, minds, and the laws of physics*. Oxford University Press, 1989.

[107] Bruce W Porter, James Lester, Kenneth Murray, Karren Pittman, Arthur Souther, Liane Acker, and Thomas Jones. AI Research in the Context of a Multifunctional Knowledge Base: The Botany Knowledge Base Project. Technical Report AI88-88, The University of Texas, AI Laboratory, The Uni' of Texas at Austin, USA, September 1988.

[108] Pearl Pu and Markus Reschberger. Case-based Assembly Planning. In *Proceedings: Case-Based Reasoning Workshop.*, pages pp245–254, Washington, D.C., 1991.

[109] Luis Carlos Rabelo, Sema Alptekin, and Ali S. Kiran. Synergy of Artificial Neural Networks and Knowledge-Based Systems for Intelligent FMS Scheduling. In *IJCNN-90: Proceedings of the International Joint Conference on Neural Networks.*, pages pp 359–365, San Diego, 1990. Institute of Electrical and Electronic Engineers.

[110] Michael Redmond. Distributed Cases for Case-Based Reasoning; Facilitating Use of Multiple Cases. In *AAAI-90: Proceedings of the Ninth National Conference on AI.*, pages pp304–309, 1990.

[111] James A Reggia. Virtual Lateral Inhibition in Parallel Activation Models of Associative Memory. In *IJCAI-85, Proceedings of the Ninth International Joint Conference on Artificial Intelligence.*, volume 1, pages p244–248, Los Angeles, California, August 1985.

[112] E Rosch, editor. *Principles of Categorization*, chapter 2. Lawrence Erlbaum Associates, 1978.

[113] Eleanor Rosch. Cognitive Representations of Semantic Categories. *Journal of Experimental Psychology: General*, 104(3):pp192–233, 1975.

[114] Stuart J Russell. A Quantitative Analysis of Analogy by Similarity. *Cognitive Modelling and Education*, 1986.

[115] Stuart Jonathan Russell. *Analogical and Inductive Reasoning.* PhD thesis, Stanford University, USA, 1987.

[116] Roger C Schank. *Dynamic Memory; a Theory of Reminding and Learning in Computers and People.* Cambridge University Press, 1982.

[117] Roger C Schank. *Explanation Patterns. Understanding Mechanically and Creatively.* Lawrence Erlbaum associates, Hillsdale, New Jersey, London, 1986.

[118] Roger C Schank and Robert P Abelson. *Scripts, Plans, Goals and Understanding.* Lawrence Erlbaum Associates, 1977.

[119] Roger C Schank and David B Leake. Creativity and Learning In A Case-Based Explainer. *Artificial Intelligence*, 40(1-3):pp353–385, 1989.

[120] Colleen M Seifert and Kristian J Hammond. Why There's No Analogical Transfer. In *Proceedings of the DARPA Case-Based Reasoning Workshop*, pages pp148–152. Morgan Kaufmann Publishers, 1989.

[121] Roger N Shepard. Discrimination and Generalisation in Identification and Classification: Comment on Nosofsky. *The Journal of Experimental Psychology*, 115(1):pp58–61, 1986.

[122] Roger N Shepard. On Exemplar-Based Representations: Reply to Ennis(1988). *The Journal of Experimental Psychology*, 117(4):pp412–414, 1988.

[123] Hong S Shinn. Abstractional Analogy: A Model Of Analogical Reasoning. In Janet Kolodner, editor, *Proceedings of the Case-Based Reasoning Workshop*, Florida, May 1988.

[124] Evangelos Simoudis and James Miller. Validated Retrieval in Case-Based Reasoning. In *AAAI-90: Proceedings of the Ninth National Conference on AI.*, pages pp310–315, 1990.

[125] David B Skalak. Options for Controlling Mixed Paradigm Systems. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp318–323. Morgan Kaufmann Publishers, 1989.

[126] Françoise Fogelman Soulie. *Neural Networks: From Theory to Applications*, August 1992. Tutorial T13, ECAI-92.

[127] J F Sowa. *Conceptual Structures, Information Processing in Mind and Machine.* Addison-Wesley Publishing Company, 1984.

[128] John F Sowa, editor. *Principles of Semantic Networks.* Morgan Kaufmann Publishers, inc, 1991.

[129] Craig Stanfill. Learning to Read: A Memory-Based Model. In Janet Kolodner, editor, *Proceedings of the Case-Based Reasoning Workshop*, Florida, May 1988. Morgan Kaufmann Publishers.

[130] Craig Stanfill and David Waltz. Towards Memory-Based Reasoning. *Communications of the ACM*, 29(12):pp1213–1228, December 1986.

[131] Robert J Sternberg and Edward E Smith, editors. *Concepts And Thought*, chapter 2. Cambridge University Press, 1986.

[132] Neil A Stilling, Mark H Feinstein, Jay L Garfeild, Edwina L Rissland, David A Rosenbaum, Steven E Weisler, and Lynne Baker-Ward. *Cognitive Science: An Introduction.* MIT Press, Cambridge Massachusetts, London England, 1987.

[133] Katia P Sycara. Resolving Goal Conflicts via Negotiation. In *AAAI-88, Proceedings of the Seventh National Conference on Artificial Intelligence*, volume 1, pages pp245–250, Saint Paul, Minnesota, August 1988. Morgan Kaufmann Publishers, Inc.

[134] Katia P Sycara and D Navinchandra. Index Transformation and Generation For Case Retrieval. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp324–328. Morgan Kaufmann Publishers, 1989.

[135] Yoshiyasu Takefuji and Kuo-Chun Lee. A Parallel Algorithm for Tiling Problems. *IEEE Transactions on Neural Networks.*, 1(1):pp143–145, 1990.

[136] Michael K Tanenhaus and James M Leiman. Evidence for Multiple Stages in the Processing of Ambiguous Words in Syntactic Contexts. *Journal of Verbal Learning and Verbal Behaviour*, 14(4), 1979.

[137] Austin Tate. A Review of Knowledge-Based Planning Techniques. In Marvin Minsky, editor, *Expert Systems 85. Proceedings of the 5th Conference of the British Computer Society.*, pages pp89–111, University of Warick, 1985. Cabridge University Press.

[138] Paul Thagard and Keith J Holyoak. Analogical Mapping by Constraint Satisfaction. *Cognitive Science*, 13:pp295–355, 1989.

[139] Paul Thagard, Keith J Holyoak, Greg Nelson, and Gochfeld. Analog Retrieval by Constraint Satisfaction. *Artificial Intelligence*, 46:pp259–310, 1990.

[140] Kevin Thompson and Pat Langley. *Concept Formation in Structured Domains*, chapter 5. Morgan Kaufmann, 1991.

[141] Philip Thrift. A Neural Network Model For Case-Based Reasoning. In *Proceedings of the Case-Based Reasoning Workshop*, pages pp334 –336, Pensacola Beach, Florida, 1989. Morgan Kaufmann Publishers.

[142] David S Touretzky and Geoffery E Hinton. Symbols Among The Neurons: Details of a Connectionist Inference Architecture. In Aravind Joshi, editor, *IJCAI-85: Proceedings of the 9th International Joint Conference on Artificial Intelligence*, volume 1, pages pp238–243, Los Angeles, California, August 1985.

[143] Amos Tversky and Itamar Gati. *Studies of Similarity*, chapter 4, pages pp79–98. Lawrence Erlbaum Associates, 1978.

[144] Tony Veale and Mark T Keane. Conceptual Scaffolding: Using Metaphors to Build Knowledge Structures. In Bernd Neumann, editor, *ECAI-92, Proceedings of the 10th European Conference on Artificial Intelligence.*, pages pp174–178, Vienna, Austria, August 1992. John Wiley and Sons, Inc.

[145] Rajendra S Wall, Dan Donahue, and Stan Hill. The Use of Domain Semantics For Retrieval and Explanation in Case-Based Reasoning. In Janet Kolodner, editor, *Proceedings of the Case-based Reasoning Workshop*, Florida, May 1988. Morgan Kaufmann Publishers.

[146] Xin Wang. Period-Doubling to Chaos in a Simple Neural Network. In *IJCNN: Proceedings of the International Joint Conference on Neural Networks*, volume 2, pages pp 333–338, Seattle, WA, 1991. Institute of Electrical and Electronic Engineer.

[147] Yorick Wilks, John Barnden, and Jin Wang. Your Metaphor or Mine: Belief Ascription and Metaphor Interpretation. In *IJCAI 91, Proceedings of the Twelfth International Joint Conference On Artificial Intelligence*, pages pp945–949, Sydney, Australia, 1991. Morgan Kaufmann Publishers, Inc.

[148] Patrick H Winston. Learning and Reasoning by Analogy. *Communications of the ACM*, 23(12), December 1980.

Appendix A

Glossary of Terms

This appendix provides an extensive glossary for the technical terms used within this thesis. There is a certain amount of cross-referencing within this glossary. When the definition of a term includes the use of another term from the glossary then this will be highlighted using an italic font.

ABSTRACT NODE The *node* that is decomposed into a collection of *detail nodes* by a *structure link*.

ACTIVATION A simple numeric value that constitutes the type of marker passed during *activation propagation*. The magnitude of activation is usually designed to be a quantitative measure of the importance of the *node* that receives it.

ACTIVATION DISTRIBUTION The momentary pattern of *activation* in a network; i.e. the set of *nodes* possessing activation at a given moment in time.

ACTIVATION LINK The most basic form of connection in the CRASH network representation. A single activation link connects a *relation* to/from another *node*. Activation links, as the name suggests, form the pathways along which *activation* flows in the CRASH system. A positive activation link simply transports activation whereas a negative activation link reverses the polarity of the activation that propagates along it.

ACTIVATION PROPAGATION A specific type of *marker passing* for which the individual markers are *activation*.

ACTIVATION PASSING see *activation propagation*.

ACTIVE (OF CONTEXT NODES) A status bestowed by the *external controller* to an individual *context node* so as to signify that the context node is part of the current *context* for CRASH.

ADAPTATION A stage of *CBR* that attempts to modify, in some way, either the *source case* or *target case* in order to enable the successful *mapping* of knowledge from the former to the latter.

ANALOGY A similarity-based *mapping* between two descriptions of *knowledge items* that exist in separate *knowledge domains*. By necessity, the components of the two descriptions are paired with respect to structural constraints and/or through the recognition of the two descriptions as instances of some more general concept, rather than by *surface level* commonalities.

ANALOGICAL REASONING (AR) A form of *assumption-based reasoning* that relies on the notion of *analogy*. An initially created analogical *mapping* is used as the basis of knowledge transfer from the *source case* to the *target case*.

ANALOGICAL RETRIEVAL A form of retrieval that yields an *analogy* to the *memory prompt* description. This requires that the *knowledge base* represents knowledge from more than one *knowledge domain*.

ARC see *link*.

ARTIFICIAL INTELLIGENCE (AI) A branch of cognitive science concerned with the construction of computer systems capable of performing tasks which would generally be considered to require intelligence if performed by a human. For a comprehensive introductory text refer to any of [32, 127, 106].

ASSUMPTION-BASED REASONING A generic term used for any *AI* system for which the reasoning is based upon assertions that are not logically well-founded (i.e. cannot be proven to be correct).

BUBBLE-SORT A simplistic 'in-place' sorting algorithm that operates by identifying adjacent list elements that are wrongly ordered with respect to each other and swapping them. This basic process is performed iteratively until the list exhibits the desired ordering.

CASE The basic descriptive structure that is retrieved from memory and used as the basis of reasoning in a *CBR* or *AR* system. Generally, a case will correspond to the complete description of a single problem solving episode, though in more flexible CBR systems a case is usually defined decompositionally, such that the individual steps that comprise a complete problem-solving episode can themselves be treated independently as cases.

CASE BASE The *knowledge base* of a *CBR* system in which a store of *source cases* is held.

CASE-BASED REASONING (CBR) A particular *KBS* paradigm that involves reasoning from precedent cases stored in a *case base*. The reasoning process generally involves: the retrieval and selection of a most relevant *source case* from the features of a *target case*, formation of a *mapping* between the two cases, transference of knowledge from source to target case, evaluation of the transferred knowledge, possible adaptation of the source case and finally learning from experience. A general discussion of practical issues concerning CBR can be found in [17]. See also *analogical reasoning*.

CAUSAL MODEL The explicit representation of knowledge concerning cause and effect within a particular *knowledge domain*.

COLOUR DEPENDENCY RULE A data structure used in CRASH to capture the generation of a set of 'child' *activation* colours from some 'parent' colour. This colour generation is usually prompted by a need to avoid *cross-talk* or *destructive interference*.

COLOUR DEPENDENCY HIERARCHY The collection of all *colour dependency rules*; this establishes a transitive ancestry between all *activation* colours.

COMBINATORIAL EXPLOSION The phenomenon whereby the *search space* that must be traversed in order to find a solution to a particular problem grows exponentially with increasing input complexity for the problem.

CONSTRUCTIVE INTERFERENCE An interaction between two or more differently coloured *activation distributions* that over-lap at a single *node*. Constructive interference is an interaction between the various activation distributions that leads to an increase in the magnitude of *activation* of each colour that is possessed by the node. This relies on the polarity of all interfering activations the same.

CONTEXT Two general notions of context can be generally identified in *AI* ; a dynamic collection of facts representing the history of a particular reasoning session or a precompiled viewpoint that segregates a collection of related *knowledge items* from the rest of a *knowledge base*. In CRASH, the model of context involves the dynamic formation of groups of active *context nodes* which collectively define the *paths* through memory that may currently be traversed.

CONTEXT LINK A primitive of the CRASH knowledge representation that connects a *relation* to a *node* representing a *knowledge item* that has some bearing on the relevancy of the *relationship* represented by the relation.

CONTEXT NODE A *normal node* associated with a *relation* via a *context link*.

CONTEXTUAL PRIMING A general phenomenon recognised in psychology whereby the preceding *context* for a *memory prompt* influences what knowledge is actually retrieved from memory, (see [132, chapter 3]).

CROSS-TALK An ambiguity problem that can arise in *activation propagation* when the conditions of a single *cycle* allows *activation* to be passed from two separate *nodes* yet the nodes have differing 'meanings' with respect to the constraints imposed by the *memory prompt*. Generally, two such nodes will have received their original activation in different cycles in the history of the activation distribution.

CYCLE Notionally, the basic time step for an *activation propagation*. During a cycle all conditions, such as which *context nodes* are currently active, remain invariable. A single activation cycle is made up of a number of individual activation *pulses*.

DESTINATION NODE The node of a *feature* to which the *relation* from the associated *source node* leads.

DESTRUCTIVE INTERFERENCE An interaction between two or more differently coloured *activation distributions* that over-lap at a single *node*. Destructive interference is an interaction between the various activation distributions that leads to a decrease in the magnitude of *activation* of each colour that is possessed by the node. This is caused by a difference in the polarity between the interfering activations.

DETAIL NODES The collection of nodes that in conjunction form the description of the single *abstract node* of a *structure link*.

EVALUATION A stage of *CBR* that establishes the validity of the *mapping* of knowledge from the *source case* to the *target case*. In addition, evaluation should identify the constraints of source and target cases that remain to be satisfied. Evaluation also identifies the incompatible source and target case components. Hence, evaluation acts as a guide for *adaptation*.

EXPLICIT REPRESENTATION Pertaining to the description of a *knowledge item*; a knowledge representation is explicit if the components of the description and their inter-relationships are transparent with respect to the syntactic conventions of the knowledge representation.

EXTENSION (OF CONCEPT REPRESENTATION) A *node* representing the set of instances of some general concept. This node is tied to a partnering *intention* node.

EXTERNAL CONTROLLER (EC) The component of the CRASH system that provides the control instructions for *main memory* and utilises the retrieval results.

FEATURE (OF TEMPLATES) A tuple of the form (*source node*, *relation*, *destination node*). The three *nodes* of a feature are connected by *activation links* and the construct forms the most primitive composite structure in the network representation (see appendix B).

GRANULARITY This relates to how 'large' are the *knowledge items* represented by the primitive elements of a knowledge representation. For a fine-grained representation the knowledge items associated with the primitives will in general be indivisible. By contrast, primitives in a coarse-grained representation will represent composite knowledge items, such as an entire *case*, that can be broken down into more primitive knowledge items.

IDIOSYNCRASY A relation weighting factor of CRASH. Idiosyncrasy factors are assigned to individual *relation nodes* and are caused by individual *uses* of a *template*. Roughly speaking, a particular idiosyncrasy value corresponds to the proportion of all other uses of a template that make a different *feature* assignment to that of the *instantiation link* that the idiosyncrasy value is attributed to.

INDEXING The most predominant means of *case* retrieval in *CBR*. Indexing generally relies on the predetermination of a vocabulary of salient case features, i.e. features relevant to the purpose of the CBR application. Memory is then organised with respect to these indices, which allows for efficient and goal-directed retrieval to be performed. For a discussion on different variations on the theme of indexing see [27].

INSTANTIATION LINK A primitive of the CRASH knowledge representation used to associate a collection of *nodes* with the *role nodes* that are contained within a particular *template*.

INTENTION (OF CONCEPT REPRESENTATION) A *node* that is the root of the description of some general concept. This node is tied to a partnering *extension* node.

INTIMATE (OF TEMPLATES) Pertaining to the fact that *relation nodes* contained within a *template* can cross its boundary, hence linking the template to the global network of memory.

INTERFERENCE (OF ACTIVATION PROPAGATION) The phenomena whereby two disjoint activation distributions over-lap and interact. The interaction can either be *constructive interference* or *destructive interference*.

KNOWLEDGE ACQUISITION BOTTLENECK The problem concerning the generally high cost of eliciting the required knowledge upon which a *KBS* will base its reasoning. In addition, because the raw knowledge obtained from sources such as human experts is generally not well structured, there is often a costly translation process to convert the knowledge into a computer usable format.

KNOWLEDGE BASE The part of a *KBS* system in which is stored the explicit knowledge used by the system to achieve reasoning.

KNOWLEDGE-BASED SYSTEM (KBS) A software system that performs complex reasoning through the manipulation of knowledge stored in an explicit (i.e. symbolic) format within an associated *knowledge base*. A discussion of the philosophy behind KBSs can be found in [85].

KNOWLEDGE DOMAIN Loosely defined as an 'area' of knowledge, such as the set of all *knowledge items* that comprise the expertise necessary to solve a particular problem.

KNOWLEDGE ITEM This is a generic term for any type of knowledge about the real world. It covers generalised concepts, specific entities, individual *relationships* and so forth.

LEARNING A process that enables a system to improve its future performance, either through performing a current task more efficiently and/or accurately, or through gaining the ability to perform new tasks. For a *KBS*, learning entails either the improved organisation of existing knowledge in memory or the assimilation of new knowledge into memory. See any of [3, 38, 68] for an introduction to the field of machine learning.

LINK A basic network primitive used to connect two (or more) *nodes* in a network representation. In many network representations (though not the one proposed in this thesis) a link represents a specific *relationships*, hence links are often labelled with relationship names.

LOCAL NETWORK REPRESENTATION a network-based representation of knowledge for which there is a one-to-one mapping between the *nodes* of the network and *knowledge items* in the real world.

MAIN MEMORY The component of a *KBS* that holds all the long-term knowledge used by the system.

MAPPING A correspondence between two two descriptions, generally involving a set of pairings between the respective components of the two descriptions.

MARKER PASSING A form of reasoning that is performed by the propagation of symbolic or numeric markers through a network-based knowledge representation.

MASSIVE PARALLELISM An implementation paradigm that can be applied to process such as *marker passing* that decompose into large numbers of identical and notionally concurrent operations. Massive parallelism applies computing architectures that are composed of large arrays of simple and identical processing elements to these types of problem, providing the possibility of several orders of magnitude speed-up.

MEMORY-BASED REASONING (MBR) A term applied to reasoning that relies primarily on the recollection of *knowledge items* stored in memory to directly provide a solution to a problem. *CBR* is a specialised form of MBR. See [130] for a discussion of the general philosophy behind MBR.

MEMORY PROMPT The conditions that instigate a retrieval from memory. A memory prompt is generally comprised of three components: a set of initial points in memory from which to initiate the retrieval search (*seed nodes*); a set of constraints that guide the retrieval; and a set of requirements in terms of what knowledge should be returned from memory (*query set*).

METAPHORICAL REPRESENTATION A representation of knowledge where the mapping between its primitives and the *knowledge items* of the real world is such that structural aspects of the real world are reflected in the structure of the representation. For example, by this definition, a toy model of, say, a sports car can be considered to be a metaphorical representation of the real thing.

MICRO-FEATURE A term used in [59, 60] to denote the most primitive attributes of a more abstract concept. Hence, while the linkage at the level of abstract concepts may be based on specified *relationships* at the level of micro-features the linkage is a measure of *semantic similarity*.

NEGATIVE CONTEXT A *context*, in the sense employed for CRASH, that indicates that the *relationship* represented by a particular *relation* is currently invalid.

NODE A basic primitive of a network representation. Nodes are generally related to one another via *links*. For the CRASH implementation two distinct classes of node exist; *normal nodes* and *role nodes*.

NORMAL NODE A *node* used in CRASH to represent a single *knowledge item*.

NP-COMPLETE A term applied to any computation for which the complexity, measured with respect to the time it takes the computation to complete, can be provably shown to have a relationship to the input data complexity of the computation that is worse than a polynomial dependency. It follows that an NP-complete computation may be subject to combinatorial explosion.

PATH Loosely, a route through the network of memory. That is to say a path can be thought of as a sequence of *links* connecting intermediate *nodes* that are passed through in forging a connection between the nodes at either end of the path.

PERMANENT MEMORY see *main memory*.

PLATEAU OF EXPERTISE PROBLEM This problem concerns the localised nature of the knowledge held by most *KBS*; outside of their *problem domain* they tend to exhibit complete ignorance. This tends to cause 'brittleness' in the system's behaviour as it cannot fall back on general principles to handle exceptional circumstances not covered by its problem-specific knowledge (see also [85]).

POSITIVE CONTEXT A *context*, in the sense employed for CRASH, that indicates that the *relationship* represented by a particular *relation* is currently relevant.

PRAGMATIC SIMILARITY A measure of similarity (i.e. *semantic similarity* and/or *structural similarity*) for which the *knowledge items* that comprise the two related descriptions are first filtered with respect to their relevance towards some particular purpose.

PROBLEM DOMAIN A *knowledge domain* containing knowledge concerned with solving a specific problem.

PROXIMITY PRINCIPLE This relies on the fact that a *local network representation* is a type of *metaphorical representation* in that it reflects the associative connectivity between *knowledge items* in reality. It follows that the length of a *path* connecting to *nodes* in a local network representation is indicative of the *semantic similarity* between the two knowledge items represented by those nodes.

PULSE A component of a *cycle*. For a single pulse every *node* that is enabled to do so will perform one *activation* sending operation.

QUERY SET The part of a *memory prompt* that represents the request for information to be fulfilled by the retrieval process. Generally, the query set can be thought of as a collection of uninstantiated variables that are bound to specific *knowledge items* as a result of retrieval.

RELATION An alias for *relation node*

RELATION NODE A knowledge representation primitive of CRASH. Specifically, it is a type of *node* whose purpose is the representation of *relationships*.

RELATIONSHIP The knowledge of a particular type of association between two *knowledge items*.

ROLE NODE A *node* used in CRASH to represent a place-filler in the generic network pattern captured by a *template*. A role node is generally used to define a number of *normal nodes* through association implemented by *instantiation links*.

RULE-BASED SYSTEM A type of *KBS* that achieves reasoning through the systematic application of production rules.

SEARCH SPACE A definition of the set of all possibilities that a reasoning process could generate during its operation. A search space is generally defined in terms of the set of possible states that can be generated and that represent a partially completed problem. In addition the states are interconnected by the representation of possible state transitions. For the network model of memory, a state equates to a *node* in the network and a state transition equates to the traversal of a *link* between two nodes.

SEED NODES For a network-based model of memory, these nodes form the starting state from which a search of memory proceeds in order to satisfy a *memory prompt*.

SEMANTIC SIMILARITY A correspondence between two descriptions based on an overall match between the sets of *knowledge items* that comprise each description. Generally, component knowledge items are paired by having identical or closely related 'meanings'.

SNAP-SHOT (OF ACTIVATION) A data structure produced at the end of each *cycle* of CRASH. The snap-shot captures the momentary *activation distribution*, in terms of the *nodes* that received *activation* during the cycle and the resultant activation held by each node. Snap-shots are arranged sequentially to provide a faithful history of a complete *activation propagation* process.

SOURCE CASE This refers to any previously 'experienced' case that has been encoded in memory and therefore may be the subject of retrieval. The terminology refers to the fact that for *CBR* or *AR* knowledge is transferred from a source case to a *target case*.

SOURCE NODE The node of a *feature* from which the *relation* to the associated *destination node* stems. The basic process of *activation propagation* is for *activation* to pass from a source node, via the relation, to the destination node.

STRUCTURAL SIMILARITY A correspondence between two descriptions based on the pattern of connections between the *knowledge items* of each description.

STRUCTURE LINK A knowledge representation primitive of CRASH. A structure link connects an *abstract node* to its associated collection of *detail nodes*. The link represents a many-to-one equivalence *relationship* and as such allows for multiple levels of *granularity* in the *knowledge base*.

SUPERLINEAR With respect to a computational process; implies that the relative rate at which the process converges towards a solution is greater than linearity (see [12, pp 573]).

SURFACE FEATURE The most readily apparent *knowledge items* that can be used to describe a *case* (i.e. its most primitive features). For a hierarchical decomposition of a case the surface features are the leaves of the tree.

SURFACE LEVEL *knowledge items* represented in the network of memory that possess the same *granularity* as the *surface features* of a *case*.

TARGET CASE The description of the current problem-solving situation. This description is usually incomplete in that it lacks knowledge of a solution. It is the inference of this knowledge by reference to *source cases* that is the task performed by *CBR*.

TEMPLATE A knowledge representation primitive of CRASH. Similar in principle to the traditional frames used in *AI*, a template captures a characteristic network pattern that constitutes the description of a general *knowledge item*. A template usually includes a set of *role nodes* and a set of *relation nodes* that connect the role nodes together, as well as linking the role nodes to *normal nodes* outside of the scope of the template. The network pattern of a template can be inherited by associating its set of role nodes with another set of nodes via an *instantiation link*.

TEMPLATE HIERARCHY A non-cyclic organisation of *templates*. The basis for the template hierarchy is that the set of *role nodes* of a specific template can inherit the network pattern associated with a more general template by use of the *instantiation link* (see appendix B for examples of this). It follows that a network pattern can be transitively inherited via several intermediate templates in the template hierarchy.

TRACTABLE Loosely, this is used to refer to a computational process that completes in a practical amount of time. More formally, the usual threshold applied in computer science for whether or not a computation is tractable is that any computation whose complexity is *NP-complete* is said to be intractable.

TYPICALITY A relation weighting factor of CRASH. Typicality factors are assigned to individual *relation nodes* and are caused by individual *uses* of a *template*. Roughly speaking, a particular typicality value corresponds to the proportion of all uses of a template that assign the same value to the *destination node* of a particular template *feature* as does the *instantiation link* that the typicality value is attributed to.

USE (OF A TEMPLATE) An *instantiation link* that (directly or transitively via the *template hierarchy*) associates a set of *nodes* with the *role nodes* of a *template*.

VAGUE (OF MEMORY PROMPT) An adjective applied to a *memory prompt* for which the specified constraints are either incomplete, in that they do not fully specify how retrieval should map from the *seed nodes* to the *query set*, or for which the constraints do not map in a one-to-one way with *relations* of the network of memory.

WEAK THEORY DOMAIN A *knowledge domain* for which no complete and underlying theory exists that can be used to guide problem solving. Rather, a weak theory domain generally comprises a collection of *ad hoc* principles and exemplar scenarios as the basic knowledge types.

WORKING MEMORY The component of a *KBS* that holds the dynamic data structures generated during problem solving.

WORLD KNOWLEDGE The collection of *knowledge items* and their *relationships* that comprise the model of the world that is *explicitly represented* inside a *KBS*. Typically, such world knowledge is restricted to a single *knowledge domain*.

Appendix B

The Base Template Hierarchy

The knowledge bases constructed for the prototype implementation of the model of memory rely on a set of basic templates that capture network patterns for the most primitive semantic constructs used to express more complex descriptions. This appendix details each template, giving an illustration of the network structure, and explaining the semantic interpretation. However, before being able to do this, it is first necessary to clarify the meaning of graphical notation that will be used to present the templates.

B.1 Notation

This section gives a brief explanation of the standard graphical format by which each of the knowledge representation primitives will be depicted in this appendix. Three general classes of primitive exist; normal nodes, role nodes and links.

B.1.1 Normal Nodes

Normal nodes are used to represent specific items of knowledge that exist in the real world. The general representation convention for all members of this class of knowledge representation primitive is that their graphical symbol is shaded. The symbols are depicted in figure B.1 and explained below.

The Normal Concept Node

Normal concept nodes are used to represent any type of knowledge item[1]. As shown in figure B.1, concept nodes are generally depicted as a shaded ellipse with a textual symbol within them. This textual symbol is solely as an illustrative aid; it provides a terse description of the real-world knowledge item that the normal concept represents. It should be stressed that no such descriptive information actually exists within normal concept nodes.

[1] with the exception of relationships and denotations.

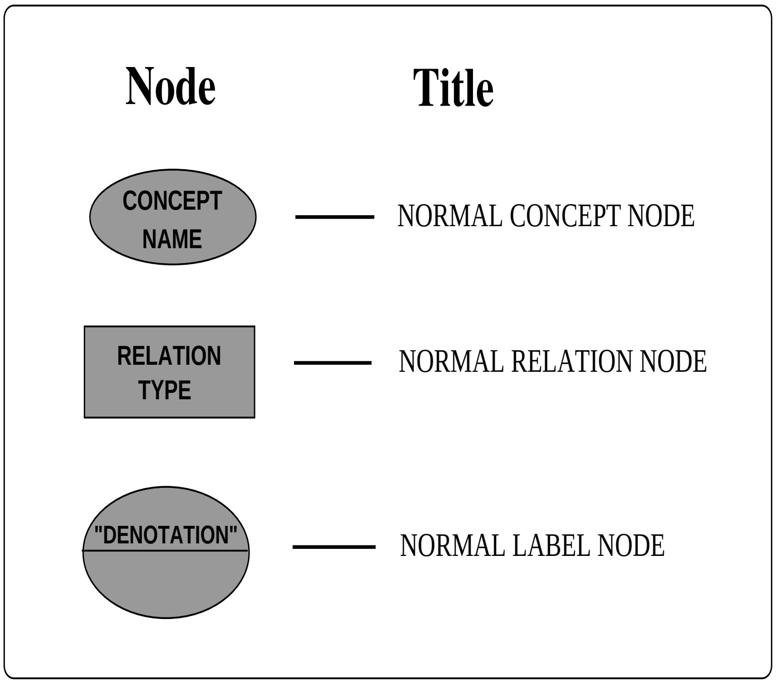


Figure B.1: The Normal Node Symbols

The Normal Relation Node

Normal relation nodes are depicted as shaded rectangles. As for concept nodes this graphical symbol is often supplemented by a textual description. In this case the description is used to specify what general class of relationship the relation node represents an instance of. Again, this knowledge is not internalised to the normal relation node; the text is used as shorthand for an explicit network pattern that should be associated with the relation to connect it with the normal concept node representing its parent type.

The Normal Label Node

The normal label node is also depicted as a shaded ellipse. The convention is that the ellipse is divided by a horizontal line and a string is contained in the top half of the symbol. This notation is intended to illustrate that normal labels lie on the boundary between the internal representation of knowledge in the knowledge base and the real world. From within the memory, a label node appears and behaves exactly like a normal concept node. However the label node *does* contain a representation of the depicted string within itself. This string is intended to represent an interface to external stimuli; it is a representation of a symbol used externally to the memory. Indeed in principle a whole family of normal label nodes could be created to deal with many different classes of external stimuli other than just a textual name for concepts.

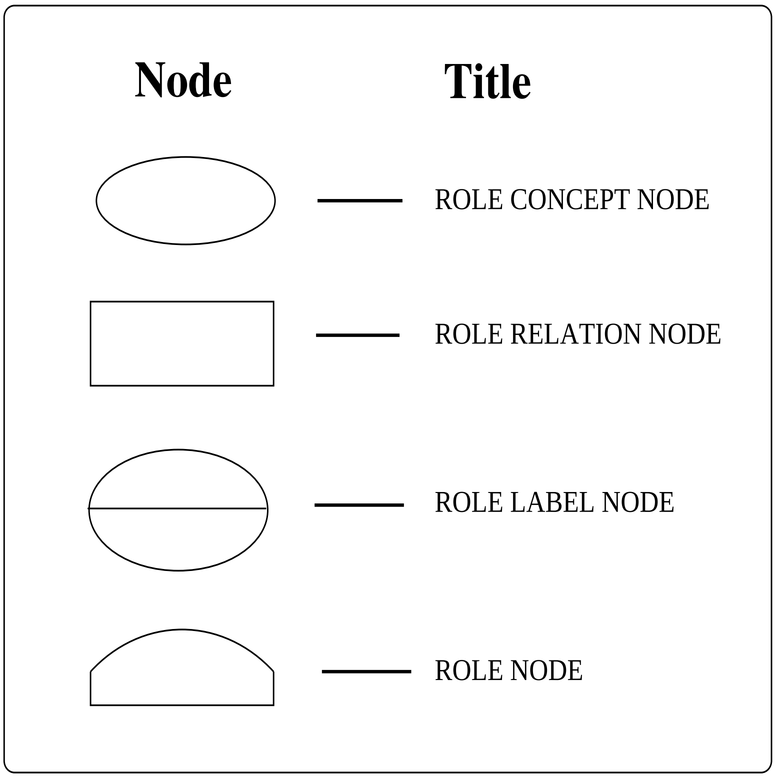


Figure B.2: The Role Node Symbols

B.1.2 Role Nodes

Role nodes act like variables in the proposed knowledge representation scheme. They do not represent any specific knowledge as such but can be instantiated by other nodes in order that those nodes can inherit the network structure surrounding the role node. Hence, role nodes are the place-holders in the network patterns captured as templates.

Role node symbols are distinguished by being unshaded (and also unannotated) as shown in figure B.2. The set of symbol shapes that are used in the presentation of role nodes mirrors that for normal nodes. This is intended to reflect constraints in terms of what type of node can instantiate each type of role node. For example, only relations can instantiate a role relation. In addition, a generic role node exists that can be instantiate by an other type of node.

B.1.3 Links

The symbols for the types of link used in presenting the base template hierarchy are shown in figure B.3.

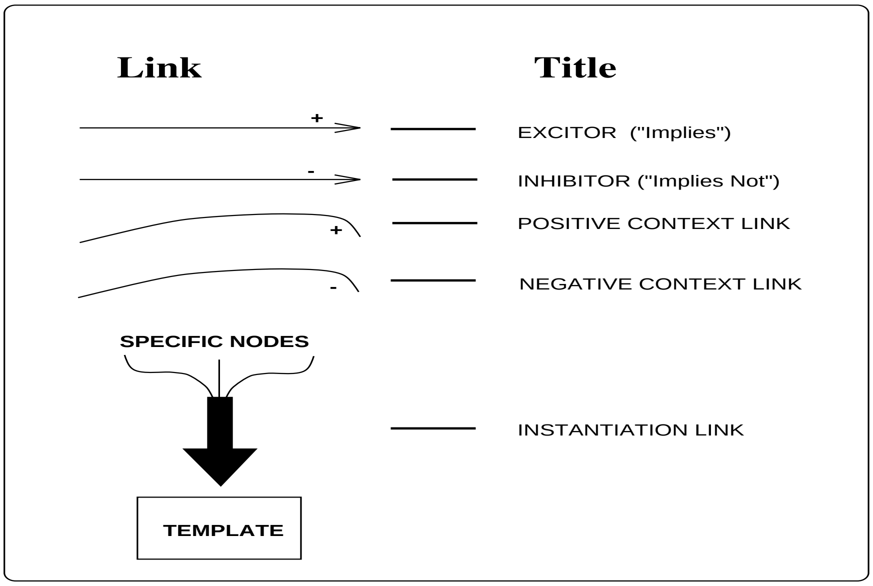


Figure B.3: The Link Symbols

The Activation Link

This type of link is used to construct the relational structure of the proposed knowledge representation. There are two varieties; excitors (that can be thought of as *implies*) and inhibitors (which can be thought of as *implies not*). The distinguishing characteristics of activation links is that they are arrowed, to represent the direction of the implication. Excitors are distinguished from inhibitors by the polarity labelling at the arrow head ("+" or "-" respectively).

The Context Link

Context links represent connections relating knowledge that determines the validity of relations to the relations themselves, (see section 2.4.3 for details about the semantics of context links). The link is depicted as a curved line between the relevant context node and a relation node. The polarity of the context link is depicted at the end that joins the relation node.

The Instantiation Link

Instantiation links pair a set of specific nodes to the nodes of a template. For succinctness of representation, the individual node-to-node pairings are not shown. Generally, these can be inferred because the presented layout of the specific nodes mirrors the layout of the role nodes in the template they instantiate.

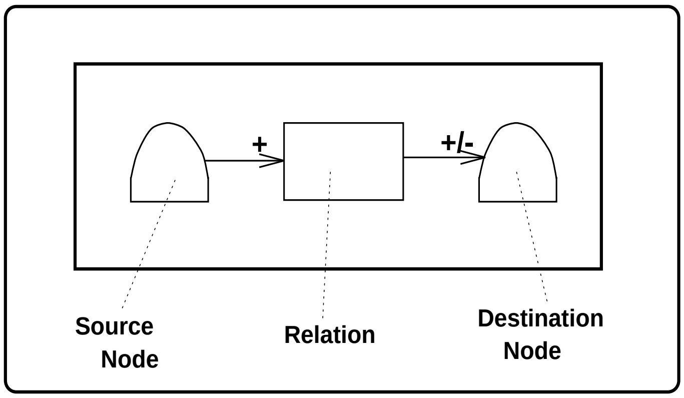


Figure B.4: The Positive/Negative Proposition Template

B.2 The Base Templates

Having established the notation to be used in presentation, the base templates are described in this section.

B.2.1 The Proposition Templates

For the construction of th network representation, the most basic semantic primitive is a relationship connecting two knowledge items. The appropriate network structure is shown in figure B.4. Note that there are two versions of this network pattern distinguishing whether a +ve or -ve activation link leads to the destination node. An example of the use of the proposition template to represent a simple statement is shown in figure B.5. Note that the concept nodes of that example represent specific instances of two general concepts, rather than the concepts themselves.

B.2.2 The Label Templates

These templates represent the relationships between a node representing a knowledge item and a label node denoting the knowledge. The internal node is defined to be the *'meaning'* of the label, and the label is defined to be the *name* of the node.

The Instance Label Template

This template is used to attach a label denoting some general concept to all instances of that concept. Concepts generally possess a great number of instances. In particular, all relation nodes in a knowledge base possess at least one general relation type that they are an instance of. It is undesirable that all nodes representing relevant instances should be directly accessible from a given label, as this would lead to too large an area of memory being accessed. Hence, only the parent concept itself should be accessible from the label, (the instances can be accessed indirectly as the *"extension"* of the concept).

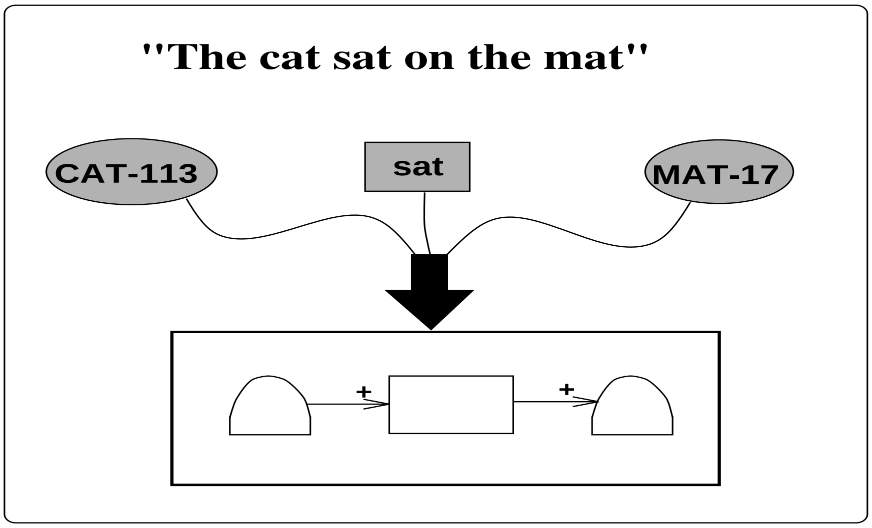


Figure B.5: An Example of the Use of the Positive Proposition Template

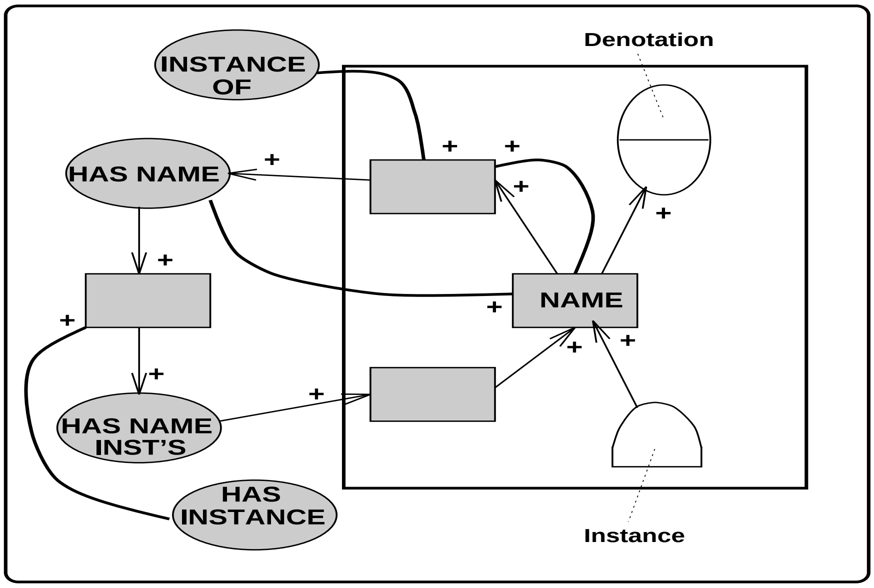


Figure B.6: The Instance Label Template

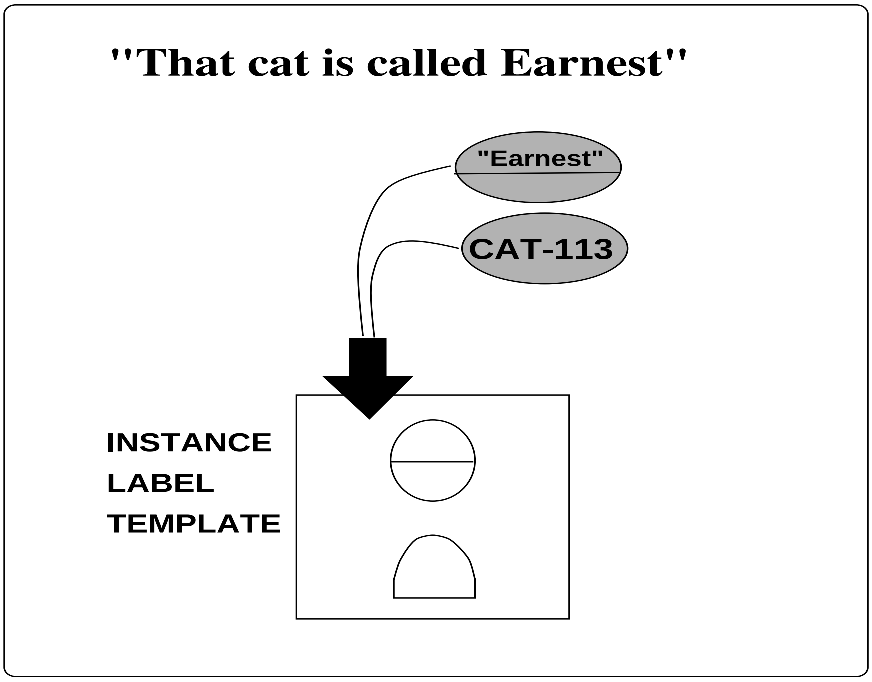


Figure B.7: An Example of the Use of the Instance Label Template

Conversely, if a particular instance is accessed during a retrieval, it may be useful to produce its denotation for use by the external controller. The instance label template of figure B.6 represents the network pattern necessary for this association. An example of the use of this template to represent the name of a specific concept instance is shown in figure B.7. Although not depicted in the example for reasons of clarity, the nodes representing the specific concept instance and its denotation will inherit the complex network structure of the instance label template and hence will be related to each other in the appropriate way; for example that **"Earnest"** is the name of the specific cat represented by node **"CAT-113"** will be made explicit.

The Concept Label Template

The concept label template is similar to the instance label template, but involves a bidirectional connection between the concept node and the label node. Also, unlike the instance label template, the relations involved in the concept label description can themselves be labelled via the instance label template, (see figure B.8).

B.2.3 Inheritance

The most common way for a knowledge item to inherit properties is by it being a sub-type/instance of some more general concept. However, inheritance can occur via many different type of relationship. For example, the sub-parts of some object inherit many of its properties; for a cat sat on a mat it is not unreasonable to assume that the cat's tail is

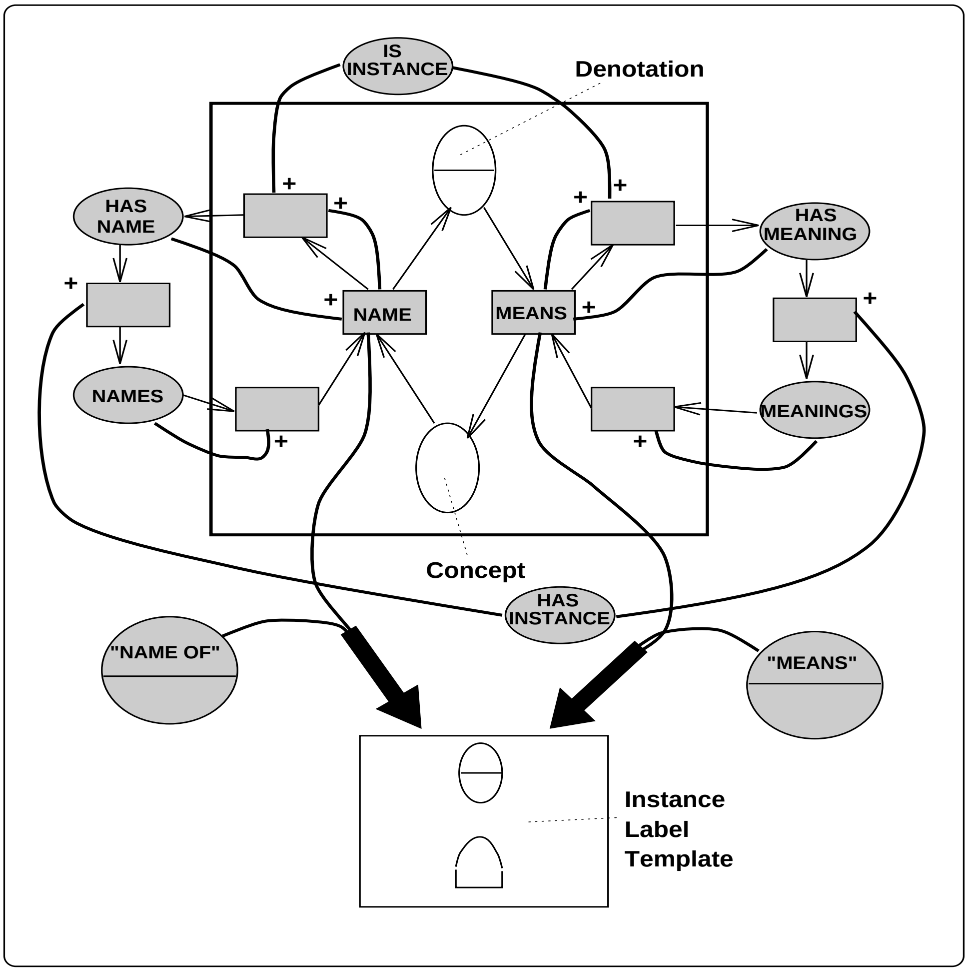


Figure B.8: The Concept Label Template

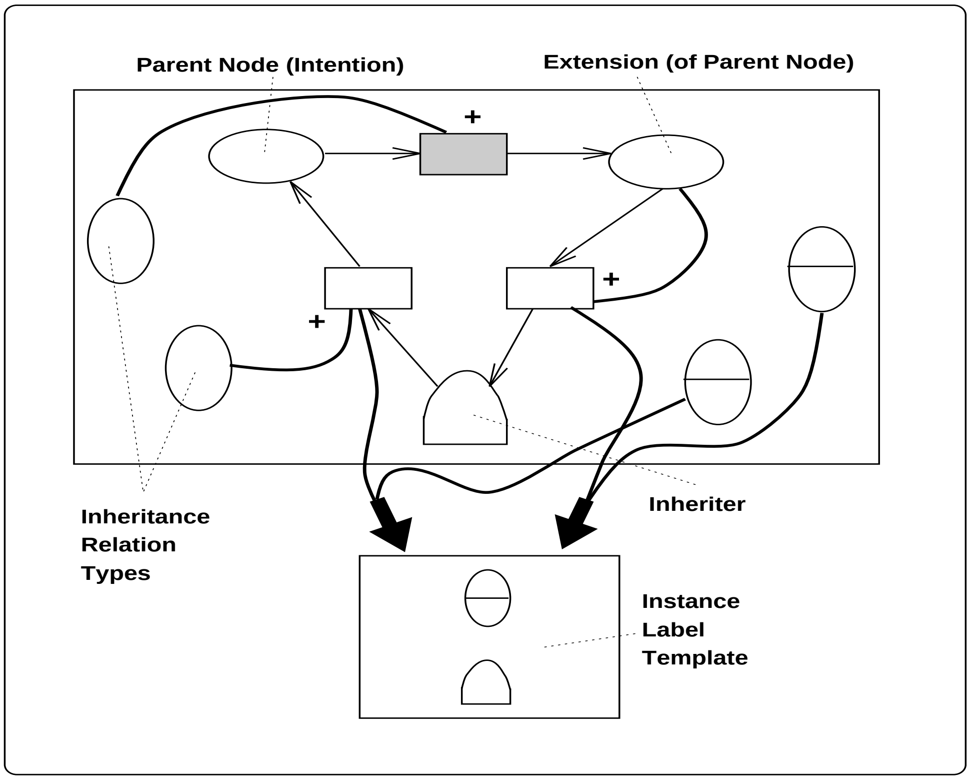


Figure B.9: The Inheritance Mechanism Template

also located on the mat. In this section two templates are defined, one representing the general network structure for any inheritance relationship, and the other utilising this template to implement type inheritance.

The Inheritance Template

Inheritance is assumed to occur from a single abstracted node (and its associated description template) to one of many specific child nodes. In this template the relationship from the parent to the child is unspecified. The parent node has an associated *extension* node that represents the set of all child nodes[2]. This indirection has practical advantages as it makes clear the distinction between concept instances and the more abstract descriptive features of the concept. In terms of practical advantages, this division allows the features of the parent node to be isolated from the selection of example children of the parent. The characteristic network for inheritance is illustrated in B.9.

The Type Inheritance Template

Type inheritance can be achieved either via "sub-type of" or "instance of" relationships. Accordingly, the type inheritance template represented in figure B.10 provides two mu-

[2]The parent node itself will be referred to as the *intention*.

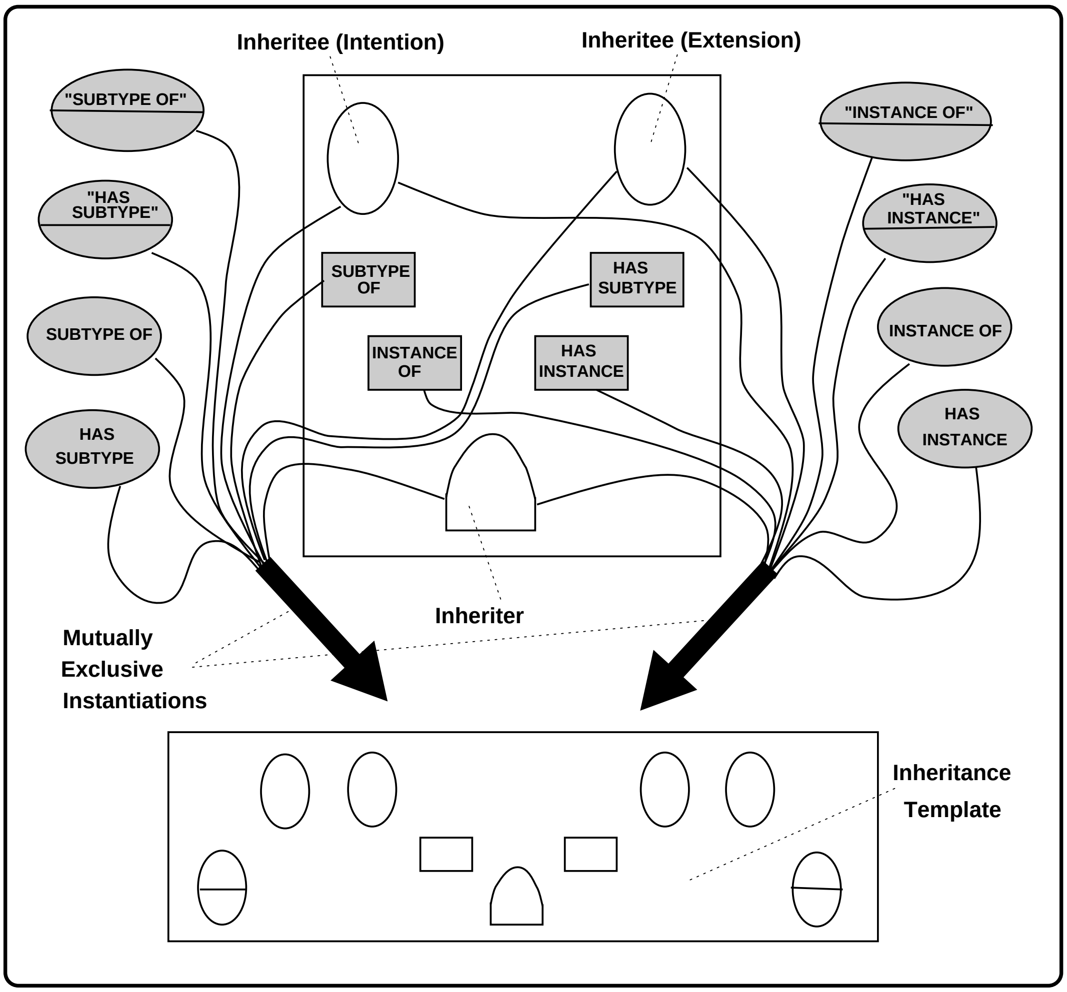


Figure B.10: The Type Inheritance Template

tually exclusive instantiations for the general inheritance template.

B.2.4 The Relationship Template

The relationship template is defined almost entirely in terms of the basic templates described above. The properties necessary for a full description of a relationship in CRASH are as follows;

- Every relation should be paired with an opposite in order to permit bi-directional search of memory.
- The activation links used to connect two related nodes can be positive or negative, (hence an alternative pair of instantiations of the proposition templates is used for both directions of the relationship, see figure B.11).

- Every relation node instance should be explicitly linked, via the type inheritance template, to a pair of nodes representing the intention and extension of its parent concept.

- The type node for a relation is one of its positive contexts.

- Every relation node instance should be labelled.

These properties can be seen as borne out in the instantiation links used to define the relationship template illustrated in figure B.11.

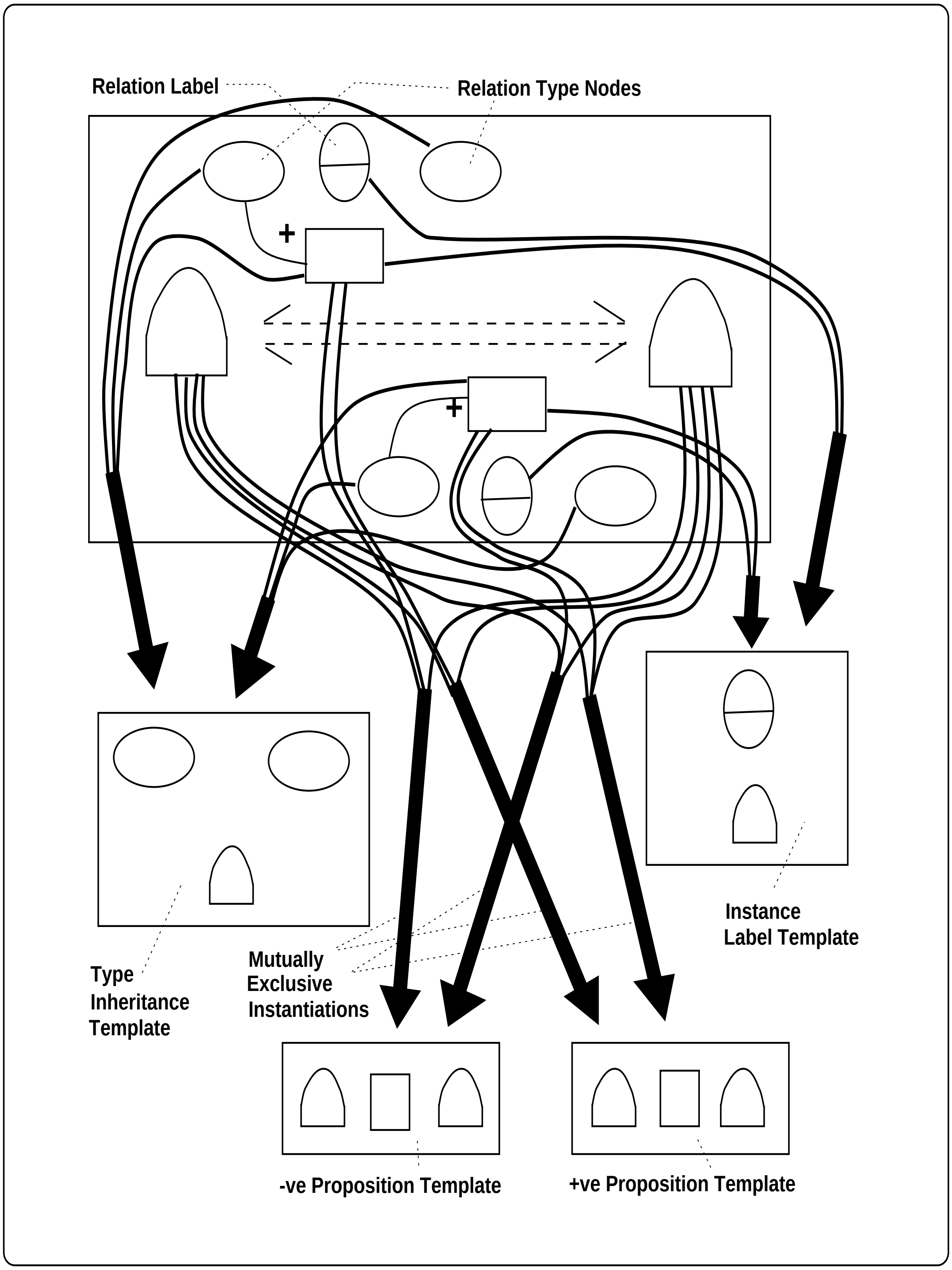


Figure B.11: The Relationship Template

Appendix C

Knowledge Representation Definitions

This appendix presents a number of definitions for some of the knowledge representation concepts more informally described in chapter 2.

C.1 The Definition of a Template

The definition of a template given in this section supplements the informal discussion on templates given in section 2.5.

Formally, let a template T be the tuple $\{N_r, R_r, R_n, P\}$. Let the set $Nodes(T)$ be $N_r \cup R_r \cup R_n$. $Nodes(T)$ constitutes the set of all nodes that are contained in T.

The set $N_r = \{node_{r1}, \ldots, node_{ri}\}$ contains the *role* nodes of the template. These are unlike *normal* nodes in that they do not represent specific items of knowledge but act as variables that can be *instantiated* by other nodes external to the template. The sets $R_r = \{rel_{r1}, \ldots, rel_{rj}\}$ and $R_n = \{rel_{n1}, \ldots, rel_{nk}\}$ represent the relations (role and normal respectively) that are contained in the template.

The set $P = \{p_1, \ldots, p_l\}$ represents the set of propositions that define the network structure captured in the template. Note that these propositions will also be referred to as *template features* in the discussion below. Each proposition, $p_i = (node_{si}, rel_i, node_{di})$, must satisfy the constraint:

Condition 1 $\forall p_i \in P.(rel_i \in R_r \cup R_n) \wedge ((node_{si} \in Nodes(T)) \vee (node_{di} \in Nodes(T)))$

In other words, for all propositions inside the template, either the source node, $node_{si}$, or destination node, $node_{di}$, is a node contained in the template. The connecting relation rel_i is also by definition in the template. However, either the source or destination node for rel_i can be a node external to the template; in other words the relations that are associated with a template can cross its boundary. In this sense the template is said to be *intimately* embedded in the rest of the network of memory.

C.2 The Definition of Typicality

The definition of typicality given in this section supplements the informal discussion given in section 2.6.1. The same example template hierarchy as found in that section is used here for illustrative purposes and is replicated in figure C.1.

Let a role node, $node_{r,j}$, inside a template, T, be the destination node for a relation, rel_k, also contained in the template. Following instantiation link paths down through the template hierarchy will cause a set of possible normal node *values* to be assigned to $node_{r,j}$. Let this set be $Values(node_{r,j}) = \{(node_{n,1}, c_1), \ldots, (node_{n,l}, c_l)\}$. The tuple $(node_{n,i}, c_i)$ represents the fact that node $node_{n,i}$ is assigned to $node_{r,j}$ in a total of c_i *assigning uses* of T[1]. An instantiation link is said to assign a specific value to a role node if all possible paths down the hierarchy, from the instantiation link to the template containing the role node, cause that value to be assigned.

As an example from figure C.1, I_7 assigns **"jewels"** to $node_{r8}$ via I_{11}. Furthermore, the same assignment is made by I_1, I_2 and I_3 as all three of these instantiation links must pass through I_7 to reach $node_{r8}$. As an example of a values set, $Values(node_{r8}) = \{$ (**"MONEY"**,3), (**"CLOTHES"**,1), (**"UTENSILS"**,1), (**"KNIVES"**,1),(**"JEWELS"**,4) $\}$.

Let $Frequencies(node_{r,j}) = \{c_1, \ldots, c_l\}$ be the frequency distribution for all value assignments, where l is the number of of distinct values. The set of typicality factors assigned to the relation rel_k is $Typicalities(rel_k) = \{typ_1, \ldots, typ_l\}$, where:

$$typ_i = \frac{c_i}{\sum_{x=1}^{l} c_x} \tag{C.1}$$

Table C.1 summarises the typicality values that would be calculated for the example of figure C.1[2]. The first column of table C.1 gives the complete group of instantiation links which have the same associated frequency, defined in terms of the value they assign to the destination of the relation specified in the second column[3]. The third column specifies the relevant number of matching assignments and the fourth column specifies the total number of assigning uses of the relevant template with respect to the particular relation. The fifth column is the resultant typicality value, given by the ratio of third column value to fourth column value. Finally, note that instantiation links that do not cause a value assignment to a particular relation's destination node cause a default typicality assignment, equal to the weighted average of all typicality assignments for that relation.

[1]The uses of a template are taken to be the set of all instantiation links transitively above it in the template hierarchy. The assigning uses of a template are those uses of a template that lead to a value assignment for the role node in question.

[2]More generally, measures of typicality are scaled by a *familiarity* coefficient that gives the statistical certainty of the typicality value. Familiarity is derived by assuming that there is a binomial distribution in terms of template uses that (do/do not) match a particular feature value assignment. A typicality value that is based on many, rather than few, uses of a template will have a correspondingly higher familiarity factor. A similar familiarity coefficient is applied to the idiosyncrasy measure defined in the section C.4.

[3]Note that this does not necessarily mean that the value assigned by each of the instantiation links is the same, merely that they cause the same typicality value to be assigned to the relation.

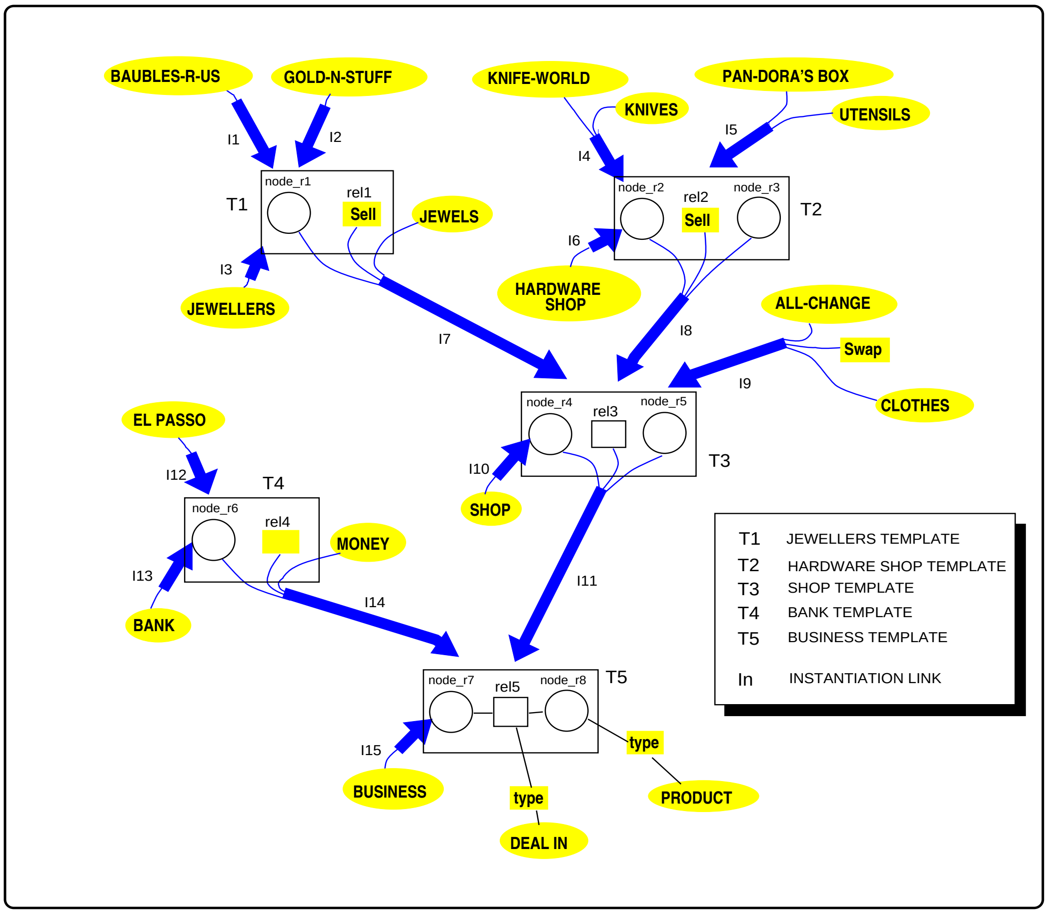


Figure C.1: An Example Template Hierarchy

C.3 The Definition of the Utility of Typicality

In this section a metric is defined for the utility of typicality, in terms of its use in improving the probability of correctly resolving an unknown feature value.

It is assumed that the set of uses of the template are *representative*. That is to say, the probability of a new instantiation link, I_{new}, assigning a value, $node_{n,i}$, to a given destination role node is equivalent (within the bounds of statistical variation) to the proportion of prior uses of the template that also assign $node_{n,i}$ to the role node. In other words, cases are spread evenly within the knowledge domain [15].

Let the total number of feature value assignments for a particular destination role node be C_{tot}, where:

$$C_{tot} = \sum_{x=1}^{l} c_x \tag{C.2}$$

Now assume that for a new case, connected to a template by I_{new}, the correct feature value is not actually known and must be selected from the *bag* of previously assigned values. One possibility is to assume that, given a free and random choice, the probability

Source Instantiation Links	Relation	Number Of Equivalent Assignments	Total Number Of Assigning Uses Of The Template	Typicality Factor
I_4, I_5	rel_2	1	2	0.5
I_6	rel_2	N/A	2	0.5 (default)
I_1, I_2, I_3, I_7	rel_3	4	7	0.57
I_4, I_5, I_9	rel_3	1	7	0.14
I_6, I_8, I_{10}	rel_3	N/A	7	0.39 (default)
I_1, I_2, I_3, I_7	rel_5	4	10	0.4
I_4, I_5, I_9	rel_5	1	10	0.1
I_{12}, I_{13}, I_{14}	rel_5	3	10	0.3
$I_6, I_8, I_{10}, I_{11}, I_{15}$	rel_5	N/A	10	0.28 (default)

Table C.1: Typicality Values for The Example Template Hierarchy

that a value $node_{n,i}$ is chosen is equal to the proportion of previous uses that have assigned $node_{n,i}$; that is $\frac{c_x}{C_{tot}}$. It follows that, the probability that for I_{new} a correct value is guessed for the destination role node, is given by:

$$Prob(CorrectGuess(node_{r,j}, I_{new})) = \sum_{n=1}^{l} \frac{c_x^2}{C_{tot}^2} \tag{C.3}$$

In terms of the retrieval mechanism presented in this thesis, the probability defined in equation C.3 is equivalent to assuming that all outward relations from a node have an equal chance of being selected during retrieval. An alternative is to assume that a typicality gradation is in place and that, if I_{new} does not indicate which value to select, the most typical relation will always be selected. Hence, the probability of a correct guess for I_{new} is equal to the probability that the the correct value for I_{new} is the most typical:

$$Prob(CorrectTypGuess(node_{r,j}, I_{new})) = \frac{Max(c_n)}{C_{tot}} \tag{C.4}$$

Typicality is said to be useful if it improves the probability of guessing an unknown value. That is to say, it is desirable that $Utility(Typicality) > 1$ where $Utility(Typicality)$ is defined as:

$$Utility(Typicality) = \frac{CorrectTypGuess(node_{r,j}, I_{new})}{CorrectGuess(node_{r,j}, I_{new})} = \frac{Max(c_n) \times \sum_{n=1}^{l} c_n}{\sum_{n=1}^{l} c_n^2} \tag{C.5}$$

The magnitude of $Utility(Typicality)$ is a function of the frequency distribution of values. Several properties of this function are proven in Appendix D.

C.4 The Definition of Idiosyncrasy

Idiosyncrasy will be defined in this section to be a measure of how unusual is the value *and* relation specialisation assigned to a particular template feature by a specific instantiation link, I_{new}. It is a measure of the proportion of *other* uses of the template that make a different assignment for the feature.

Again, let $node_{r,i}$ be the role node that is the destination of a feature of template T and rel_i, a relation in T, be the relation of that feature. Let $Assignment(node_{r,i}, I_{new})$ be the normal node assigned to $node_{r,i}$ by I_{new}. For example (from figure C.1), $Assignment(node_{r8}, I_4) =$ **"KNIVES"**. Let $Types(rel_i, I_{new})$ be the set $\{type_1, \ldots, type_n\}$ of all nodes representing the parent concept of rel_i, as assigned by the indirect path to the relation from I_{new}. For example (again from figure C.1), $Types(rel_5, I_4) = \{$ **"deal in"**, **"sell"** $\}$[4].

To aid in the definition of idiosyncrasy several sets representing different categories of instantiation link can be defined:

Definition 1 *Let* $AllUses(T)$ *be all (direct and indirect) uses of the template* T*. For example,* $AllUses(T_3) = \{I_1, \ldots, I_{10}\}$ *(see figure C.1).*

Definition 2 *Let* $AllUsesThrough(I_{new}, T)$ *be all uses of* T *that pass through* I_{new}*, including* I_{new} *itself. For example,* $AllUsesThrough(I_8, T_3) = \{I_4, I_5, I_6, I_8\}$*.*

Definition 3 *Let* $AllValueAssignments(node_{r,i}, node_{n,x})$ *be the set of all uses of* T *that cause* $node_{n,x}$ *to be assigned to* $node_{r,i}$*. For example,* $AllValueAssignments(node_{r5},$**"JEWEL"**$) = \{I_1, I_2, I_3, I_7\}$*.*

Definition 4 *Let* $AllTypeAssignments(rel_i, \{type_1, \ldots, type_m\})$ *be the set of all uses of* T *that cause* rel_i *to be assigned a set of type nodes that are a subset of* $\{type_1, \ldots, types_m\}$*. For example,* $AllTypeAssignments(rel_5, \{$**"sell"** $\}) = \{I_1, \ldots, I_8, I_{10}\}$*.*

For a given instantiation link I_{new} and feature $(?, rel_i, node_{r,i})$ of template T, the set of all *other* uses of the template that match the feature assignment is given by:

Definition 5 $Others(I_{new}, rel_i, node_{r,i}, T) =$

$$(AllValueAssignments(node_{r,i}, Assignment(node_{r,i}, I_{new})) \cap AllTypeAssignments(rel_i, Types(rel_i, I_{new}))) \ominus AllUsesThrough(I_{new}, T)$$

Where $\ominus$ is taken to denote set difference. For example, $Others(I_1, rel_3, node_{r5}) = \{I_4, I_5, I_9\}$.

Idiosyncrasy is defined as follows:

Definition 6 $Idiosyncrasy(rel_{T,i}, node_{r,i}, I_{new}) =$

$$(1 - \frac{Cardinality(Others(I_{new}, rel_i, node_{r,i}))}{Cardinality(AllUses(T) \ominus AllUsesThrough(I_{new}, T))}).$$

[4] Note that in figure C.1 a short hand notation is used such that the linkage between each relation instance and the node representing its parent type is not explicitly represented; rather it is implicit in the labeling of the relation nodes.

Source Instantiation Links	Relation	Number of Other Matching Template Uses	Number of Other Template Uses	Idiosyncrasy Factor
I_1, I_2, I_3	rel_1	2	2	0.0 (1-2/2)
I_4, I_5	rel_2	1	2	0.5 (1-1/2)
I_6	rel_2	2	2	0.0 (1 - 2/2)
I_1, I_2, I_3	rel_3	6	9	0.33 (1 - 6/9)
I_4, I_5	rel_3	3	9	0.66 (1 - 3/9)
I_6	rel_3	8	9	0.11 (1 - 8/9)
I_7	rel_3	3	6	0.5 (1 - 3/6)
I_8	rel_3	5	6	0.17 (1 - 5 / 6)
I_9	rel_3	1	9	0.89 (1 - 1/9)
I_{10}	rel_3	9	9	0.0 (1 - 9/9)
I_1, I_2, I_3	rel_5	8	14	0.43 (1 - 8/14)
I_4, I_5	rel_5	5	14	0.64 (1 - 5/14)
I_7	rel_5	5	11	0.55 (1 - 5/11)
I_8	rel_5	10	11	0.09 (1 - 10/11)
I_9	rel_5	3	14	0.79 (1 - 3/14)
$I_{10}, 1_{15}$	rel_5	14	14	0.0 (1 - 14/14)
I_{11}	rel_5	4	4	0.0 (1 - 4/4)
I_{12}, I_{13}	rel_5	5	14	0.64 (1 - 5/14)
I_{14}	rel_5	5	12	0.58 (1 - 5/12)

Table C.2: Idiosyncrasy Values for the Example Template Hierarchy

The values of idiosyncrasy generated, according to the above definition, for the template hierarchy of figure C.1 are summarised in table C.2. The first column of table C.2 gives a collection of instantiation links grouped together because they involve an equivalent calculation of idiosyncrasy with respect to the relation specified in the second column. The third and fourth columns corresponds to the numerator and denominator (respectively) for the expression given in definition 6. Finally, the fifth column presents the idiosyncrasy values calculated for the example of figure C.1.

C.5 The Definition of the Utility of Idiosyncrasy

In this section it is argued that idiosyncrasy, as defined in section C.4, is useful for reducing the classification search spaces that must be circumnavigated during feature-based case retrieval.

Let I_{new} represent the connection of a new case with existing memory structures. Generally the case will be represented by instantiating one or more appropriate templates. Assume that I_{new} uses template T, which contains within it m features. For the new case there will generally be a corresponding m feature assignments that associate normal node values with the role nodes within T.

Now consider feature-based retrieval from the new case. This can be thought of as finding the other uses of T that have greatest correspondence to I_{new} in terms of their feature assignments. Alternatively, the problem can be conceptualised in terms of a flat network structure (generated by expanding all paths in the template hierarchy) where the new case contains m origin nodes for paths searching through memory to find other cases. Again, the goal will generally be to retrieve other cases where the greatest number of retrieval paths coincide. The ideal is to find a match on *all* features, hence the general problem is one of trying to find the intersection of m single-feature retrieval spaces.

For a single feature assignment there is a specified outward relation to the appropriate value node. The search space can then be considered in terms of all possible relations of the opposite type to the original outward relation that lead outward from the value node. This is illustrated in figure C.2, which shows a portion of the flattened network structure that is equivalent to the template hierarchy of figure C.1.

For each feature i of the m total, let the search space have size $Size_i$. Assume that search proceeds serially via steps $a, \ldots, x$ where $a, \ldots, x$ is some permutation of $1, \ldots, m$.

Now, let the amount of work done to serially intersect all search spaces when the sequence $a, \ldots, x$ is generated randomly be $Work_{random}(I_{new}, T)$. Alternatively, if idiosyncrasy is used to select features then, in accordance with the definition of idiosyncrasy (definition 6), the features are selected in terms of ascending order of relative search space size. Let the work done when idiosyncrasy determines feature ordering be $Work_{idio}(I_{new}, T)$. The utility of idiosyncrasy can be defined as:

$$Utility(Idiosyncrasy) = \frac{Mean(Work_{random}(I_{new}, T))}{Work_{idio}(I_{new}, T)} \qquad (C.6)$$

$Utility(Idiosyncrasy)$ is characterised in appendix E.

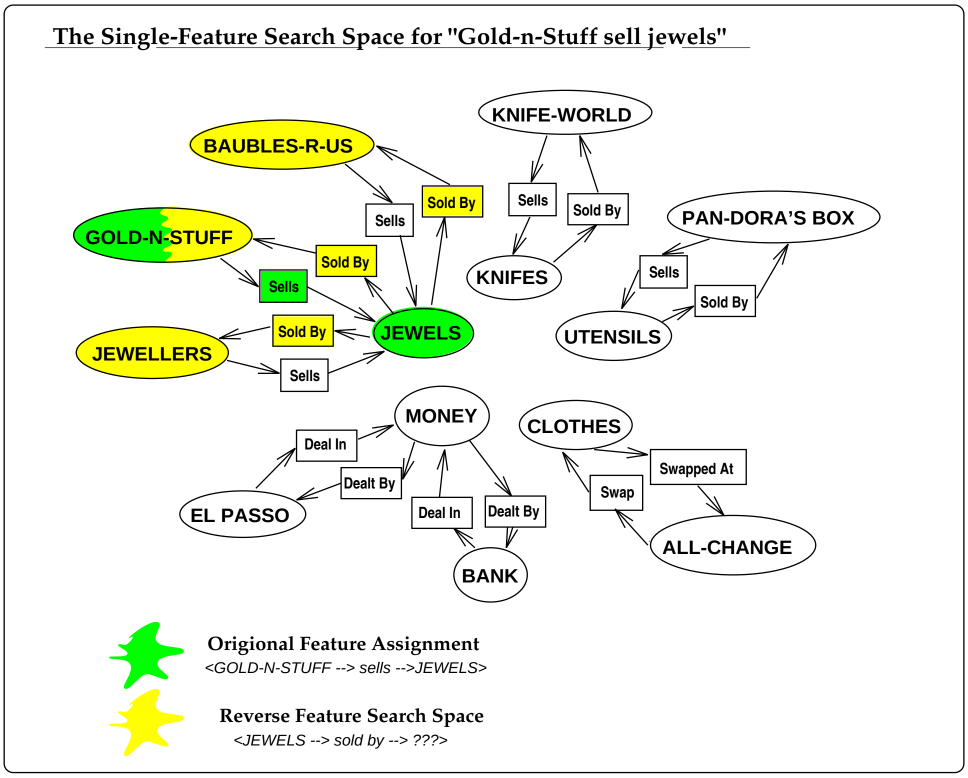


Figure C.2: An Example of the Search Space Characterised Characterised by a Single Specific Feature

C.6 The Definition of the Weight Factor Propagation Relationship

This section defines an appropriate expression that can be applied iteratively to allow the initial weight factors assigned to relations in the network of memory to have a non-localised effect.

Assume relation rel_i has a set of successor relations, $Successors(rel_i) = \{rel_1, \ldots, rel_j\}$, defined to be the relations out of the destination node of rel_i in a flattened network structure. The aim for propagation is to partially amalgamate the weight factors initially assigned to rel_i with those of its successors. This means that the resultant weight on rel_i will not only be indicative of the importance of the relation's specific destination node, but also indicative of the importance of the entire network description accessed via rel_i. The goal is to connect relations with high local utility to produce more global paths through memory. To achieve this, the aspects that should be combined in weight propagation are:

- The initial weight factor for rel_i, (a measure of the localised utility of the node directly accessed by rel_i).

- The average weight factor for $Successors(rel_i)$, (a measure of the global importance of the entire network description accessed via rel_i).

- The maximum weight factor for $Successors(rel_i)$, (a measure of *the* most important path through the network accessible via rel_i)[5].

The propagation of weight factors is an iterative process that allows a single weight factor to have a transitive effect across several relations in memory. Also, because the desire is to connect the relations with highest weight factors, the effect of propagation on a relation should diminish as its weight factor magnitude increases. Accordingly, the following iterative relationship is used:

Definition 7 $Weight_{new}(rel_i) = (C * Weight_{old}(rel_i) + (\frac{1}{Weight_{old}(rel_i)+F} *$
$(D * MaxWeight(Successors(rel_i)) + E * AveWeight(Successors(rel_i))))) \div$
$(C + \frac{D+E}{Weight_{old}(rel_i)+F})$

Where C,D,E and F are constants[6].

[5]Note that relations that lead to nodes on the periphery of a description will gain diminished importance through weight propagation simply because they have no successors.

[6]The values of $A = 30$, $B = 1$, $C = 1$ and $D = 1$ for 20 propagation cycles has proven empirically to give good results for weight propagation.

Appendix D

The Nature of the Utility of Typicality

Equation C.5 gives a definition of the utility of typicality. In this appendix the characteristics of the utility function are defined in terms of the variation in the distribution of value frequencies.

In characterising typicality, the frequency distribution for the set of values assigned to a node $node_{r,T,j}$ is $c_n = \{c_1, \ldots, c_l\}$[1], where $c_i \geq 1$, (see page 160). For describing the nature of the utility of typicality two factors of this distribution are taken to be constant, namely the number of values, l, and the sum of the frequencies C_{tot}. The mean value of the frequency distribution is given by $\bar{c} = \frac{C_{tot}}{l}$. The maximum frequency in the distribution is taken to be $Max(c_n)$. The range that can be defined for $Max(c_n)$ is:

$$\bar{c} \leq Max(c_n) \leq C_{tot} - (l - 1) \tag{D.1}$$

Table D.1 shows an example of calculations for $Utility(Typicality)$ given a frequency distribution where $l = 5$ and $C_{tot} = 12$. Note in particular that for a given $Max(c_n)$ utility decreases as statistical variance in the distribution increases. This can be attributed to the fact that, according to equation C.3, the chance of a correct random guess of a value increases with polarisation of values of c_n.

From equation C.5 conclusions can also be made about the value of $Utility(Typicality)$ at the limits of the range of $Max(c_n)$. Firstly, if $Max(c_n) = \bar{c}$ then, from the fact that all c values must equal $\bar{c}$ and from the definition of $\bar{c}$, it follows that:

$$\sum_{n=1}^{l} c_n^2 = l\bar{c}^2 = \frac{C_{tot}^2}{l} \tag{D.2}$$

Also, from the definition of $Max(c_n)$ and $\bar{c}$, it follows that

$$Max(c_n)C_{tot} = \frac{C_{tot}^2}{l} \tag{D.3}$$

[1] c_n is equivalent to $Frequencies(node_{r,j})$ defined in appendix C.

Frequency Distribution	$Max(C_n)$	$\sum_{n=1}^{l} c_n^2$	variance	$Utility(Typicality)$
(1,1,1,1,8)	8	68	9.8	1.41
(1,1,1,2,7)	7	56	6.8	1.5
(1,1,2,2,6)	6	46	4.3	1.56
(1,1,1,3,6)	6	48	4.8	1.5
(1,2,2,2,5)	5	38	2.3	1.58
(1,1,2,3,5)	5	40	2.8	1.5
(1,1,1,4,5)	5	44	3.8	1.36
(2,2,2,2,4)	4	32	0.8	1.5
(1,2,2,3,4)	4	34	1.3	1.41
(1,1,2,4,4)	4	36	1.8	1.33
(1,1,3,3,4)	4	36	2.3	1.26
(2,2,2,3,3)	3	30	0.3	1.2
(1,2,3,3,3)	3	32	0.8	1.125

Table D.1: Utility of Typicality Values for a Frequency Distribution

Recalling that $C_{tot} = \sum_{n=1}^{l} c_n$ and substituting equations D.2 and D.3 into equation C.5, the following can be concluded:

$$Utility(Typicality) = \frac{Max(c_n) \times \sum_{n=1}^{l} c_n}{\sum_{n=1}^{l} c_n^2} = 1 \quad \text{(D.4)}$$

The upper limit for $Max(c_n)$ is determined when $C_{tot} \gg l$. At this limit $Max(c_n) \longrightarrow C_{tot}$ and $\sum_{n=1}^{l} c_n^2 \longrightarrow C_{tot}^2$. Again substituting into equation C.5:

$$Utility(Typicality) = \frac{Max(c_n) \times \sum_{n=1}^{l} c_n}{\sum_{n=1}^{l} c_n^2} = \frac{C_{tot}C_{tot}}{C_{tot}^2} = 1 \quad \text{(D.5)}$$

It can be concluded at this point that, at its limits, the utility of typicality tends to 1. Also, as illustrated in table D.1, between its limits the utility of typicality is demonstrably greater than 1. Hence typicality is justified as a useful measure for the selection of unknown values. What remains is to determine at what value of $Max(c_n)$ the value of $Utility(Typicality)$ is maximal and what magnitude this maximal value is.

D.1 Determining When Utility of Typicality is Maximal

To determine what value of $Max(c_n)$ gives a maximal value for typicality utility first assume $Max(c_n) = mC_{tot}$, where $0 \leq m \leq 1$. It follows that the amount that $Max(c_n)$

exceeds $\bar{c}$ is given by:

$$\Delta_{max} = C_{tot}(m - \frac{1}{l}) \tag{D.6}$$

From the observation that $Utility(Typicality)$ is maximal when the statistical variance of the frequency distribution is minimal for a given $Max(c_n)$ it is reasonable to assume that the deficit Δ_{max} is distributed evenly amongst the remaining $l-1$ frequency values. Hence every value other than $Max(c_n)$ is given by:

$$Min(c_n) = \bar{c} - \frac{\Delta_{max}}{l-1} = \frac{C_{tot}}{l} - \frac{C_{tot}(m - \frac{1}{l})}{l-1} \tag{D.7}$$

It follows that:

$$\sum_{n=1}^{l} c_n^2 = Max(c_n)^2 + (l-1)Min(c_n)^2 \tag{D.8}$$

Substituting definitions for Max and Min into this equation it follows that:

$$\sum_{n=1}^{l} c_n^2 = m^2 C_{tot}^2 + (l-1)(\frac{C_{tot}}{l} - \frac{C_{tot}(m - \frac{1}{l})}{l-1})(\frac{C_{tot}}{l} - \frac{C_{tot}(m - \frac{1}{l})}{l-1}) \tag{D.9}$$

Expanding out the brackets and simplifying this becomes:

$$\sum_{n=1}^{l} c_n^2 = m^2 C_{tot}^2 + \frac{C_{tot}^2}{(l-1)}(m^2 - 2m + 1) \tag{D.10}$$

According to the assumption that C_{tot} is constant, maximising $Utility(Typicality)$ is equivalent to minimising the ratio $\frac{\sum_{n=1}^{l} c_n^2}{Max(c_n)}$ where, $Max(c_n) = mC_{tot}$. It follows from equation D.10 that, for maximal utility:

$$\frac{d}{dm}(mC_{tot} + \frac{C_{tot}}{(l-1)}(m - 2 + \frac{1}{m})) = 0 \tag{D.11}$$

Differentiating this gives:

$$(C_{tot} - \frac{C_{tot}}{m^2(l-1)} + \frac{C_{tot}}{(l-1)}) = 0 \tag{D.12}$$

Rearranging and simplifying this leads to the result that:

$$m = \sqrt{\frac{1}{l}} \tag{D.13}$$

From this it is concluded that the value of $Max(c_n)$ that maximises typicality utility is:

$$Max(c_n)_{Max} = \sqrt{\frac{1}{l}} C_{tot} \tag{D.14}$$

D.2 The Maximum Value for the Utility of Typicality

Substituting equation D.13 into equation D.10 the following is produced:

$$\sum_{n=1}^{l} c_n^2 = \frac{C_{tot}^2}{l} + \frac{C_{tot}^2}{(l-1)}(\frac{1}{l} - \frac{2}{\sqrt{l}} + 1) \tag{D.15}$$

This can be rearranged and simplified to give:

$$\sum_{n=1}^{l} c_n^2 = C_{tot}^2(\frac{2l - 2\sqrt{l}}{l(l-1)}) \tag{D.16}$$

Substituting this equation and equation D.14 into equation C.5, (the definition of $Utility(Typicality)$), the following expression is produced:

$$Utility(Typicality)_{Max} = \frac{Max(c_n)_{Max} \times C_{tot}}{\sum_{n=1}^{l} c_n^2} = \frac{1}{\sqrt{l}} \div \frac{2l - 2\sqrt{l}}{l(l-1)} \tag{D.17}$$

Simplifying and rearranging it can be concluded that:

$$Utility(Typicality)_{Max} = \frac{(l-1)}{2(\sqrt{l}-1)} \tag{D.18}$$

In particular, note that the maximum value of typicality utility is a function only of the number of distinct values assigned to a node. The form of this equation is shown in figure D.1.

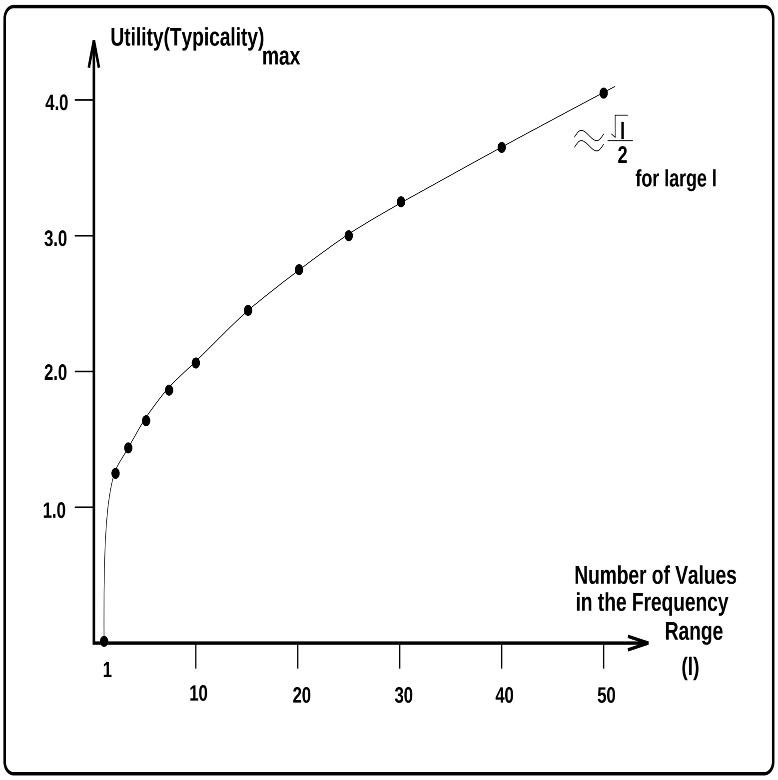


Figure D.1: The Variation of Maximum Typicality Utility with Frequency Distribution Breadth

Appendix E

The Nature of the Utility of Idiosyncrasy.

Recall that, at the end of section 2.6.2, the task of identifying a new case in terms of existing cases is defined in terms of finding the intersection of a series of search spaces, one for each new case feature. The goal of this appendix is to show that using idiosyncrasy to order the search spaces in ascending size can improve this search efficiency.

Assume that a feature i produces a corresponding search space s_i that is equated to all source cases possessing feature i. Let a be the first feature used in retrieval, with associated search space s_a. It is assumed that the resultant search space, once feature i is considered, is defined to be $s_{i,comb} = s_a \cap s_b \ldots \cap s_i$. The first problem is to characterise $s_{i+1,comb}$ in terms of the two search spaces to be intersected at the next step, namely $s_{i,comb}$ and s_{i+1}.

Let $s_{i+1,min}$ be the smaller of $s_{i,comb}$ and s_{i+1} and $s_{i+1,max}$ be the larger of the two search spaces. The most efficient search strategy is to take each member of the smaller set and check that the defining criteria[1] of the larger set applies to it. If it is assumed that cases are distributed evenly in "feature space" and that features are independent then the probability that a given element of $s_{i+1,min}$ is *not* also in $s_{i+1,max}$ is given by $\kappa^{Size(s_{i+1max})}$, where κ is a coefficient determined by the relative density of cases in feature space, the range for this coefficient being $0.0 \leq \kappa \leq 1.0$. It follows that, after step $i+1$, the resultant size of the search spaces intersection is given by:

$$Size(s_{i+1,comb}) = Size(s_{i+1,min})(1 - \kappa^{Size(s_{i+1,max})}) \tag{E.1}$$

The amount of work that requires to be performed in generating the complete search space intersection is given by:

$$WorkDone_{tot} = \sum_{i=1}^{m} Size(s_{i,min}) \tag{E.2}$$

[1]For feature based search, the defining criteria for a set $s_{i,comb}$ is that a case possesses features a to i. The defining characteristic of s_i is simply the possession of feature i.

For $i = 1$, $s_{i,min} = s_i$. For $i > 1$ the work done in producing a resultant intersection depends on a comparison in sizes of the search space generated so far and the search space based upon the ith feature. The total amount of work done is therefore highly dependent on the ordering with which features are considered. The following section summarises the results achieved for an empirical study of idiosyncrasy-based ordering versus random ordering, in terms of the required work to intersect the search spaces.

E.1 Some Empirical Findings

To empirically determine the utility properties of idiosyncrasy A number of example distributions were generated, of the form $\{Size(s_1), \ldots, Size(s_m)\}$, representing hypothetical sets of search space sizes. The distributions were varied in terms of length m, the sum total $Size_{tot} = \sum_{n=1}^{m} Size(s_n)$, and the amount of variance in the sizes. For each distribution, the work done for an idiosyncrasy ordering, namely $Work_{idio}$, was compared to the work done for random orderings of the distribution. For each sequence 100 random orderings were considered, generating the set $Works_r = \{Work_{random,1}, \ldots, Work_{random,100}\}$. The following values were calculated:

- $Max(Works_r)$ the maximum amount of work empirically measured for a random sequence.
- $Min(Works_r)$ the minimum amount of work empirically measured for a random sequence.
- $Mean(Works_r)$

Given the utility of idiosyncrasy can be estimated for each sequence according to equation the following equation (see page 165):

$$Utility(Idiosyncrasy) = \frac{Mean(Work_{random}(I_{new}, T))}{Work_{idio}(I_{new}, T)} \tag{E.3}$$

The empirical process was repeated for a set of values for κ ranging from high probability that an element is in two search spaces, $\kappa = 0.01$, through intermediate values, $\kappa = 0.1, \kappa = 0.5$, through to a low probability for an element being in two search spaces, $\kappa = 0.99$.

The results achieved are summarised in tables E.1 and table E.2. From these tables it can be seen that the assumption that chosing the smallest search space first minimises search effort holds. Hence $Utility(Idiosyncrasy)$ is generally greater than 1.

$Utility(Idiosyncrasy)$ tends to a maximum as κ tends to 1. For this limit the intersection of search spaces tends to an empty set and the actual search effort tends towards $2 * Size(s_1)$. Hence selecting the smallest search space first has maximal effect at this limit.

As variance tends to 0 then $Utility(Idiosyncrasy)$ tends to 1 because for zero variance the feature selection order has no effect on the amount of work that has to be done.

$Size_{tot}$	Variance	κ	$Work_{idio}$	$Min(Works_r)$	$Max(Works_r)$	$Mean(Works_r)$	$Utility(Idiosyncrasy)$
30	(low) 0.44	0.01	20.0	20.0	28.0	23.0	1.15
		0.1	19.8	19.8	30.0	22.4	1.13
		0.5	12.1	12.1	23.9	16.6	1.39
		0.99	4.0	4.0	8.2	5.4	1.34
	(med) 2.22	0.01	9.9	9.9	28.9	14.8	1.49
		0.1	9.1	9.1	25.9	13.9	1.53
		0.5	4.2	4.5	26.5	10.7	2.54
		0.99	2.0	2.0	10.3	4.7	2.32
	(high) 40.0	0.01	9.7	9.7	29.7	11.7	1,21
		0.1	7.1	7.2	27.7	8.7	1.21
		0.5	3.0	3.0	24.0	5,4	1.79
		0.99	2.0	2.0	22.0	3.65	1.81
50	(low) 5.11	0.01	20.0	20.0	42.0	26.6	1.33
		0.1	19.8	19.8	44.0	25.5	1.28
		0.5	13.8	13.9	41.9	22.3	1.62
		0.99	4.0	4.0	16.7	8.5	2.11
	(med) 6.67	0.01	10.0	10.0	38.0	19.5	1.95
		0.1	9.9	9.9	39.0	18.2	1.83
		0.5	6.9	7.0	36.1	17.1	2.49
		0.99	2.0	2.0	17.7	7.3	3.60
	(high) 160.0	0.01	9.6	9.6	49.7	14.5	1.50
		0.1	7.1	7.1	47.7	14.5	2.03
		0.5	3.0	3.0	44.0	6.7	2.25
		0.99	2.0	2.0	42.3	5.2	2.60
100	(low) 6.67	0.01	60.0	60.0	85.0	67.9	1.13
		0.1	60.0	60.0	86.0	68.8	1.14
		0.5	5.93	5.94	89.9	69.2	1.16
		0.99	12.4	12.4	27.7	18.8	1.51
	(med) 36.7	0.01	10.0	10.0	70.0	28.7	2.87
		0.1	10.0	10.0	76.0	29.8	2.98
		0.5	8.8	8.8	65.8	28.8	3.29
		0.99	2.0	2.0	34.6	13.6	6.70
	(high) 810.0	0.01	9.6	9.6	99.7	22.3	2.30
		0.1	7.1	7.1	97.7	12.7	1.80
		0.5	3.0	3.0	94.0	12.2	4.06
		0.99	2.0	2.0	92.6	10.2	5.05

Table E.1: Empirical Estimates for Utility of Idiosyncrasy for a Search Space Size Distribution of Length 10

$Size_{tot}$	Variance	κ	$Work_{idio}$	$Min(Works_r)$	$Max(Works_r)$	$Mean(Works_r)$	$Utility(Idiosyncrasy)$
15	(low) 0.5	0.01	10.0	10.0	15.0	11.5	1.15
		0.1	10.0	10.0	15.0	11.4	1.14
		0.5	8.6	8.6	14.3	10.5	1.22
		0.99	4.0	4.0	7.0	5.5	1.37
	(med) 2.5	0.01	5.0	5.0	14.0	8.0	1.59
		0.1	5.0	5.0	15.0	8.3	1.67
		0.5	4.0	4.0	15.0	8.3	2.06
		0.99	2.0	2.0	9.2	5.1	2.51
	(high) 20.0	0.01	4.9	4.9	15.0	8.9	1.79
		0.1	4.4	4.4	14.7	8.4	1.89
		0.5	2.9	2.9	13.7	7.3	2.54
		0.99	2.0	2.0	12.1	5.3	2.60
25	(low) 2.5	0.01	15.0	15.0	25.0	17.8	1.18
		0.1	15.0	15.0	25.0	18.3	1.22
		0.5	14.2	14.2	23.9	17.9	1.27
		0.99	6.1	6.1	13.4	9.1	1.48
	(med) 10.0	0.01	5.0	5.0	25.0	11.0	2.2
		0.1	5.0	5.0	25.0	12.3	2.45
		0.5	4.6	4.6	25.0	11.1	2.44
		0.99	2.0	2.0	16.0	6.62	3.31
	(high) 80.0	0.01	4.9	4.9	24.9	12.8	2.58
		0.1	4.4	4.4	24.7	14.0	3.16
		0.5	2.8	2.8	23.8	9.8	3.41
		0.99	2.0	2.0	22.0	8.8	4.40
50	(low) 2.5	0.01	40.0	40.0	49.0	42.9	1.08
		0.1	40.0	40.0	50.0	42.6	1.07
		0.5	39.9	39.9	50.0	42.8	1.07
		0.99	16.8	16.8	24.3	19.6	1.16
	(med) 0.5	0.01	15.0	15.0	50.0	28.8	1.88
		0.1	15.0	15.0	47.0	27.4	1.83
		29.0	14.9	14.9	46.0	25.9	1.73
		0.99	6.2	6.2	32.2	14.9	2.39
	(high) 405.0	0.01	4.9	4.9	49.9	20.2	4.13
		0.1	4.4	4.4	49.7	18.6	4.18
		0.5	2.9	2.9	48.8	19.6	6.76
		0.99	2.0	2.0	47.4	20.2	10.05

Table E.2: Empirical Estimates for Utility of Idiosyncrasy for a Search Space Size Distribution of Length 5

Variance	$Max(Size_i)$	$Work_{idio}$	$Mean(Works_r)$	$Utility(Idiosyncrasy)$
4.7	8	15.2	25.0	1.64
5.1	8	13.8	22.3	1.62
6.7	9	6.9	17.1	2.49
6.9	8	6.8	16.8	2.46
8.9	12	7.6	19.3	2.55
9.1	8	4.8	14.7	3.09
9.8	10	4.8	14.5	3.03
11.3	8	3.8	14.0	3.65
12.0	9	3.8	12.8	3.35
14.9	9	3.2	13.7	4.0
18.7	10	3.2	9.6	3.01
20.0	10	3.2	10.1	3.17
21.3	11	3.2	10.3	3.26
23.3	18	13.9	22.1	1.59
24.0	17	3.4	10.0	2.89
25.8	12	3.1	8.6	3.19
27.6	18	3.7	10.1	2.75
28.0	12	3.1	11.7	3.83
32.2	19	3.5	11.5	3.30
36.4	22	12.0	17.7	1.47
36.9	20	3.2	9.2	2.83
37.3	15	3.1	9.5	2.81
45.3	22	3.1	8.4	2.69
48.2	18	3.0	9.5	3.14
50.4	18	3.0	8.43	2.80
50.4	23	3.1	7.2	2.33
51.6	25	4.3	12.0	2.79
56.0	20	3.0	8,64	2,87
59.3	25	3.1	8.0	2.59
62.9	20	3.0	8.1	2.68
65.6	28	9.9	15.9	1.61
71.1	21	3.0	8.8	2.94
71.3	22	3.0	9.24	3.08
73.3	28	3.0	7.4	2.42
74.7	25	3.0	8.1	2.68
82.0	28	3.0	8,2	2.72
83.5	31	9.4	12.7	1.34
88.2	31	3.0	8.44	2.79
93.3	31	3.0	8.1	2.71
160.0	41	3.0	6.74	2.25

Table E.3: Variation of Utility of Idiosyncrasy with Sequence Variance

Accordingly, assuming all other factors are constant, $Utility(Idiosyncrasy)$ tends to increase as variance increases. However, the utility depends also on other factors. For instance, if there are a large number of features that give highly specific search spaces, then the random ordering of features rapidly prunes the combined search space and consequently the utility of idiosyncrasy is reduced. Table E.3 records the results for $Utility(Idiosyncrasy)$ for a variety of distributions of search space size given constant parameters; $m = 10, Size_{tot} = 50, \kappa = 0.5$. These results are plotted in figure 2.8.

Appendix F

Initial Weight Assignment; An Example

The figure 2.9 of section 2.6.3 gives a fairly complex network representation of a hypothetical bank, namely the **EL PASSO** bank. The following description of that bank was used:

```
IMPREGNABLE BANKS, INC. possess a chain of banks. Each bank has a
security system with many of the standard features of modern banks
such as; video cameras, door alarms, infra red beams and an alarmed
safe. The banks are also manned 24 hours a day by security guards.
Unusually, IMPREGNABLE only employ security guards trained in at least
one martial art, a policy adopted by only one other banking company.
The particular bank at El Passo is even more more secure as its
manager, one Louis D Crust, keeps his pet alligator chained in the
vault over night.
```

The aim of this appendix is to explain a plausible set of relation weight factors, (*typicality* and *idiosyncrasy*) that may be assigned to the description of the El Passo Bank. These weight factors are used in section 2.6.3 to illustrate the power of weight factor propagation.

The description of the El Passo bank will involve the instantiation of many individual templates representing a variety of domain concepts such as **"BANK"**, **"SECURITY SYSTEM"** and so forth. For simplicity, it is assumed that two levels of templates exist; the level of banking in general and a more specific level that pertains only to IMPREGNABLE BANKS, INC. Figure F.1 shows the assumed template structure from which the **EL PASSO** description is derived. It follows that each relation in the **EL PASSO** description will inherit two pairs of weight factors.

F.1 Initial Weights for each Bank Description Relation

This section takes each relation of figure 2.9 in turn and ascribes plausible, initial typicality and idiosyncrasy values to it. At the banking level, it is not obvious whether typicality and idiosyncrasy values should be assigned in terms of **EL PASSO** bank instance with respect to all bank instances, or in terms of *all* IMPREGNABLE bank instances with respect to all bank instances. In practise, both these sets of weights would be calculated and which ever comparison yields the highest weight factor is inherited for a particular instance.

For initial value assignment, it is assumed that the IMPREGNABLE company own 20 percent of all banks. Further, it is assumed that their are 100 IMPREGNABLE banks, (i.e. there are 500 banks all told). The initial weight factor assignments are given below for each relation. At the level of all banks, only the highest possible idiosyncrasy and typicality value that may be determined with respect to **EL PASSO** is considered. The template in which the relation resides and a brief justification of the weight factor assignments is also given.

1. rel_1 : **EL PASSO - is instance of - BANK** (from the **"BANK"** template). For all descriptions the value of this feature is **BANK**, hence, at both the *"banking"* and *"IMPREGNABLE"* level, this relation has typicality of 1.0 and idiosyncrasy of 0.0.

2. rel_2 : **EL PASSO - has part - SAFE** (from the **"BANK"** template). It is reasonable to assume that all bank instances have safes, hence at both levels typicality is 1.0 and idiosyncrasy 0.0.

3. rel_3 : **EL PASSO - has part - SECURITY SYSTEM** (from the **"BANK"** template). Similarly, all bank instances have security systems; typicality is 1.0, idiosyncrasy 0.0.

4. rel_4 : **EL PASSO - employs - STAFF** (from the **"BANK"** template). Similarly, all bank instances employ staff; typicality is 1.0, idiosyncrasy 0.0.

5. rel_5 : **BANK - has parent company - IMPREGNABLE** (from the **"BANK"** template). By definition, at the *"IMPREGNABLE"* level, typicality of this relation is 1.0, idiosyncrasy is 0.0. For the *banking* level typicality is 0.2 because this is the proportion of all bank instances that belong to the IMPREGNABLE company. The idiosyncrasy is 1.0 and is produced from a comparison of *all* IMPREGNABLE instances to all banking instances.

6. rel_6 : **BANK - is a type of - BUSINESS** (from the **"BANK"** template). This is a definitional feature of banks, hence typicality is 1.0, idiosyncrasy is 0.0 at both levels.

7. rel_7 : **STAFF - has sub-part - SECURITY GUARD** (from the **"BANK STAFF"** template). All IMPREGNABLE instances possess security guards, hence at this level typicality is 1.0, idiosyncrasy is 0.0. Now assume that a small proportion of other banks don't have security guards, (perhaps because they are too small). Arbitrarily, 11 percent of all banks (i.e 14 percent of *other* non-IMPREGNABLE banks) are taken to have no security guards. At the banking level this gives typicality of 0.89 and idiosyncrasy of 0.14.

8. rel_8 : **STAFF - has sub-part - MANAGER** (from the **"BANK STAFF"** template). All bank instances are taken to have managers, hence at both levels typicality is 1.0, idiosyncrasy is 0.0.

9. rel_9 : **STAFF - has sub-part - TELLERS** (from the **"BANK STAFF"** template). All bank instances are taken to have at least one teller, hence at both levels typicality is 1.0, idiosyncrasy is 0.0.

10. rel_{10} : **STAFF - has sub-part - CLERKS** (from the **"BANK STAFF"** template). It is assumed that a small percent (arbitrarily 6 percent) of bank instances are too small to employ clerks. However, this is not taken to be true of IMPREGNABLE bank instances which gives typicality of 1.0 and idiosyncrasy of 0.0 at the *IMPREGNABLE* level. At the banking level, typicality is 0.94 (for **EL PASSO** with respect to all bank instances) and idiosyncrasy is 0.8 (for all IMPREGNABLE instances with respect to all bank instances).

11. rel_{11} : **BANK - has sub-part - VAULT** (from the **"BANK"** template). This is not a particularly strong feature of banks. Assume only 40 percent of IMPREGNABLE banks possess vaults. At the *"IMPREGNABLE"* level typicality is 0.4, idiosyncrasy is 0.67. Assume that 50 percent of *all* banks possessing a vault. This leads to typicality of 0.5 and idiosyncrasy of 0.5 at the banking level, the latter being derived from a comparison of the **EL PASSO** instance to all bank instances.

12. rel_{12} : **SAFE - is located - VAULT** (from the **"SAFE"** template). The location of a bank's safe may have several values of which the vault is a common example. Assume that 35 percent of IMPREGNABLE banks have a safe in the vault. This gives typicality of 0.35 and idiosyncrasy of 0.65 for **EL PASSO**. Now assume that 0.45 percent of *all* banks have the safe in the vault. This gives typicality of 0.45 and idiosyncrasy of 0.55 for **EL PASSO** with respect to all other bank instances.

13. rel_{13} : **SAFE - has part - SAFE ALARM** (from the **"SAFE"** template). This is an optional extra security measure for safes. It is assumed that a disproportionately high percentage of IMPREGNABLE instances have this feature, say 80 percent with respect to 65 percent for *all* bank instances. This gives typicality of 0.8 and idiosyncrasy of 0.2 at the *"IMPREGNABLE"* level. At the banking level, typicality is 0.65 and idiosyncrasy is 0.39.

14. rel_{14} : **SECURITY SYSTEM - has part - SAFE ALARM** (from the **"SECURITY SYSTEM"** template). The initial weights for this feature are the same as for the previous feature.

15. rel_{15} : **SECURITY SYSTEM - has part - INFRA-RED BEAM** (from the **"SECURITY SYSTEM"** template). This is taken to be a common feature of security systems, possessed by 90 percent of IMPREGNABLE instances and 80 percent of *all* bank instances. Accordingly, at the *"IMPREGNABLE"* level, typicality is 0.9 and idiosyncrasy is 0.1. At the *banking"* level typicality is 0.8 and idiosyncrasy is 0.23.

16. rel_{16} : **SECURITY SYSTEM - has part - CAMERA** (from the **"SECURITY SYSTEM"** template). This is taken to be a common feature of security systems, possessed by all IMPREGNABLE instances and 90 percent of *all* bank instances. Accordingly, at the *"IMPREGNABLE"* level, typicality is 1.0 and idiosyncrasy is 0.0, where as at the *banking"* level typicality is 0.9 and idiosyncrasy is 0.13.

17. rel_{17} : **SECURITY SYSTEM - has part - DOOR ALARM** (from the **"SECURITY SYSTEM"** template). This another common feature of security systems, possessed by all IMPREGNABLE instances and 80 percent of *all* bank instances. Accordingly, at the *"IMPREGNABLE"* level, typicality is 1.0 and idiosyncrasy is 0.0, where as at the *banking"* level typicality is 0.8 and idiosyncrasy is 0.25.

18. rel_{18} : **SECURITY SYSTEM - has part - SECURITY GUARD** (from the **"SECURITY SYSTEM"** template). This feature is related to feature 7. At the *"IMPREGNABLE"* level typicality is 1.0, idiosyncrasy is 0.0. At the *"banking"* level typicality is 0.89, idiosyncrasy is 0.14.

19. rel_{19} : **VAULT - is a type of - ROOM** (from the **"VAULT"** template). True for all vaults, hence typicality is 1.0 and idiosyncrasy is 0,0 at both levels.

20. rel_{20} : **VAULT - contains - ALLIGATOR** (from the **"VAULT"** template). Vaults can contain many things, but **EL PASSO's** vault is the only one to contain an alligator! Accordingly, typicality is 0.01 and idiosyncrasy 1.0 at the *"IMPREGNABLE"* level. Typicality is effectively 0.0 and idiosyncrasy is 1.0 at the *"banking"* level.

21. rel_{21} : **SECURITY GUARDS - trained in - MARTIAL ARTS** (from the **"SECURITY GUARDS"** template). Assume that the one other banking company that employs security guards trained in martial arts, like IMPREGNABLE BANKS, INC, shares 20 percent of the market. It follows that at the *"IMPREGNABLE"* level typicality is 1.0 and idiosyncrasy is 0.0. Typicality is 0.4 and idiosyncrasy is 0.75 at the *"banking"* level.

22. rel_{22} : **MANAGER - is - LOUIS D CRUST** (from the **"BANK MANAGER"** template). Obviously, there is only one bank manager that is **Louis D Crust** so; typicality is 0.01 and idiosyncrasy 1.0 at the *"IMPREGNABLE"* level. Typicality is effectively 0.0 and idiosyncrasy is 1.0 at the *"banking"* level.

23. rel_{23} : **LOUIS D CRUST - has martial status - MARRIED** (from the **"BANK MANAGER"** template). This is a relatively incidental, and presumably fairly typical feature for bank managers. Arbitrarily, it is assumed that 60 percent of IMPREGNABLE managers are married compared with 70 percent of all bank managers. Hence, at the *"IMPREGNABLE"* level typicality is 0.6 and idiosyncrasy is 0.4. At the *"banking"* level typicality is 0.7 and idiosyncrasy is 0.3.

24. rel_{24} : **LOUIS D CRUST - has age - 50s** (from the **"BANK MANAGER"** template). Note that, without some *"chunking"* of ages into block ranges, this feature will always be completely idiosyncratic for each manager instance. It is arbitrarily assumed that 30 percent of IMPREGNABLE managers are in their fifties and that 40 percent of all bank managers are in their fifties. Hence, at the *"IMPREGNABLE"* level typicality is 0.3 and idiosyncrasy is 0.7. At the *"banking"* level typicality is 0.4 and idiosyncrasy is 0.6.

25. rel_{25} : **LOUIS D CRUST - owns - ALLIGATOR** (from the **"BANK MANAGER"** template). It is a feature of bank managers that they own many things, but only **LOUIS D CRUST** owns an alligator. Accordingly, typicality is 0.01 and idiosyncrasy 1.0 at the *"IMPREGNABLE"* level. Typicality is effectively 0.0 and idiosyncrasy is 1.0 at the *"banking"* level.

The plausible initial values for weight factors, described above, are summarised in table F.1. This table includes the "fuzzy" combination weight that can be produced from the two weight factors. This provides the initial graded structure shown in figure 2.10.i. Also shown in table F.1 are the final weight factor values that are assigned after weight factor propagation through the description network, as described in section 2.6.3. This produces a more refined graded structure represented in figure 2.10.ii.

Rel	Reference template	Initial Factors			Post-Propagation Factors		
		Typ_{init}	$Idio_{init}$	$Weight_{init}$	Typ_{final}	$Idio_{final}$	$Weight_{final}$
rel_1	IMPREGNABLE BANKS	1.0	0.0	0.5	0.81	0.33	0.46
	BANKS	1.0	0.0	0.5	0.81	0.33	0.46
rel_2	IMPREGNABLE BANKS	1.0	0.0	0.5	0.76	0.34	0.42
	BANKS	1.0	0.0	0.5	0.76	0.34	0.42
rel_3	IMPREGNABLE BANKS	1.0	0.0	0.5	0.89	0.16	0.46
	BANKS	1.0	0.0	0.5	0.89	0.16	0.46
rel_4	IMPREGNABLE BANKS	1.0	0.0	0.5	0.92	0.20	0.51
	BANKS	1.0	0.0	0.5	0.92	0.20	0.51
rel_5	IMPREGNABLE BANKS	1.0	0.0	0.5	0.47	0.0	0.11
	BANKS	0.2	1.0	0.6	0.06	0.47	0.12
rel_6	IMPREGNABLE BANKS	1.0	0.0	0.5	0.47	0.0	0.11
	BANKS	1.0	0.0	0.5	0.47	0.0	0.11
rel_7	IMPREGNABLE STAFF	1.0	0.0	0.5	0.79	0.27	0.42
	BANK STAFF	0.89	0.14	0.47	0.74	0.31	0.38
rel_8	IMPREGNABLE STAFF	1.0	0.0	0.5	0.59	0.55	0.33
	BANK STAFF	1.0	0.0	0.5	0.59	0.55	0.33
rel_9	IMPREGNABLE STAFF	1.0	0.0	0.5	0.47	0.0	0.11
	BANK STAFF	1.0	0.0	0.5	0.47	0.0	0.11
rel_{10}	IMPREGNABLE STAFF	1.0	0.0	0.5	0.47	0.0	0.11
	BANK STAFF	0.94	0.08	0.5	0.44	0.03	0.10
rel_{11}	IMPREGNABLE BANKS	0.4	0.6	0.3	0.47	0.55	0.28
	BANKS	0.5	0.50	0.29	0.51	0.52	0.28
rel_{12}	IMPREGNABLE SAFES	0.35	0.65	0.33	0.45	0.57	0.29
	SAFES	0.45	0.55	0.31	0.49	0.53	0.28
rel_{13}	IMPREGNABLE SAFES	0.8	0.2	0.4	0.35	0.06	0.07
	SAFES	0.65	0.39	0.36	0.35	0.15	0.09
rel_{14}	IMPREGNABLE BANKS	0.8	0.2	0.4	0.35	0.06	0.07
	BANKS	0.65	0.39	0.36	0.35	0.15	0.09
rel_{15}	IMPREGNABLE SECURITY	0.9	0.1	0.45	0.4	0.03	0.09
	SECURITY SYSTEMS	0.8	0.23	0.38	0.35	0.08	0.09
rel_{16}	IMPREGNABLE SECURITY	1.0	0.0	0.5	0.47	0.0	0.11
	SECURITY SYSTEMS	0.9	0.13	0.48	0.4	0.04	0.09
rel_{17}	IMPREGNABLE SECURITY	1.0	0.0	0.5	0.47	0.0	0.11
	SECURITY SYSTEMS	0.8	0.25	0.42	0.35	0.08	0.07
rel_{18}	IMPREGNABLE SECURITY	1.0	0.0	0.5	0.79	0.27	0.41
	SECURITY SYSTEMS	0.89	0.14	0.47	0.74	0.31	0.39
rel_{19}	IMPREGNABLE VAULTS	1.0	0.0	0.5	0.47	0.0	0.11
	VAULTS	1.0	0.0	0.5	0.47	0.11	0.11
rel_{20}	IMPREGNABLE VAULTS	0.01	1.0	0.50	0.0	0.47	0.11
	VAULTS	0.0	1.0	0.5	0.0	0.47	0.11
rel_{21}	IMPREGNABLE GUARDS	1.0	0.0	0.5	0.47	0.0	0.11
	SECURITY GUARDS	0.4	0.75	0.46	0.14	0.34	0.20
rel_{22}	IMPREGNABLE MANAGERS	0.01	1.0	0.5	0.2	0.75	0.32
	BANK MANAGERS	0.0	1.0	0.5	0.23	0.78	0.39
rel_{23}	IMPREGNABLE MANAGERS	0.6	0.4	0.3	0.24	0.14	0.04
	BANK MANAGERS	0.7	0.3	0.35	0.29	0.10	0.07
rel_{24}	IMPREGNABLE MANAGERS	0.3	0.7	0.35	0.10	0.29	0.06
	BANK MANAGERS	0.4	0.6	0.3	0.14	0.23	0.05
rel_{25}	IMPREGNABLE MANAGERS	0.01	1.0	0.5	0.0	0.47	0.11
	BANK MANAGERS	0.0	1.0	0.5	0.0	0.47	0.11

Table F.1: Some Values for Weight Factor Propagation

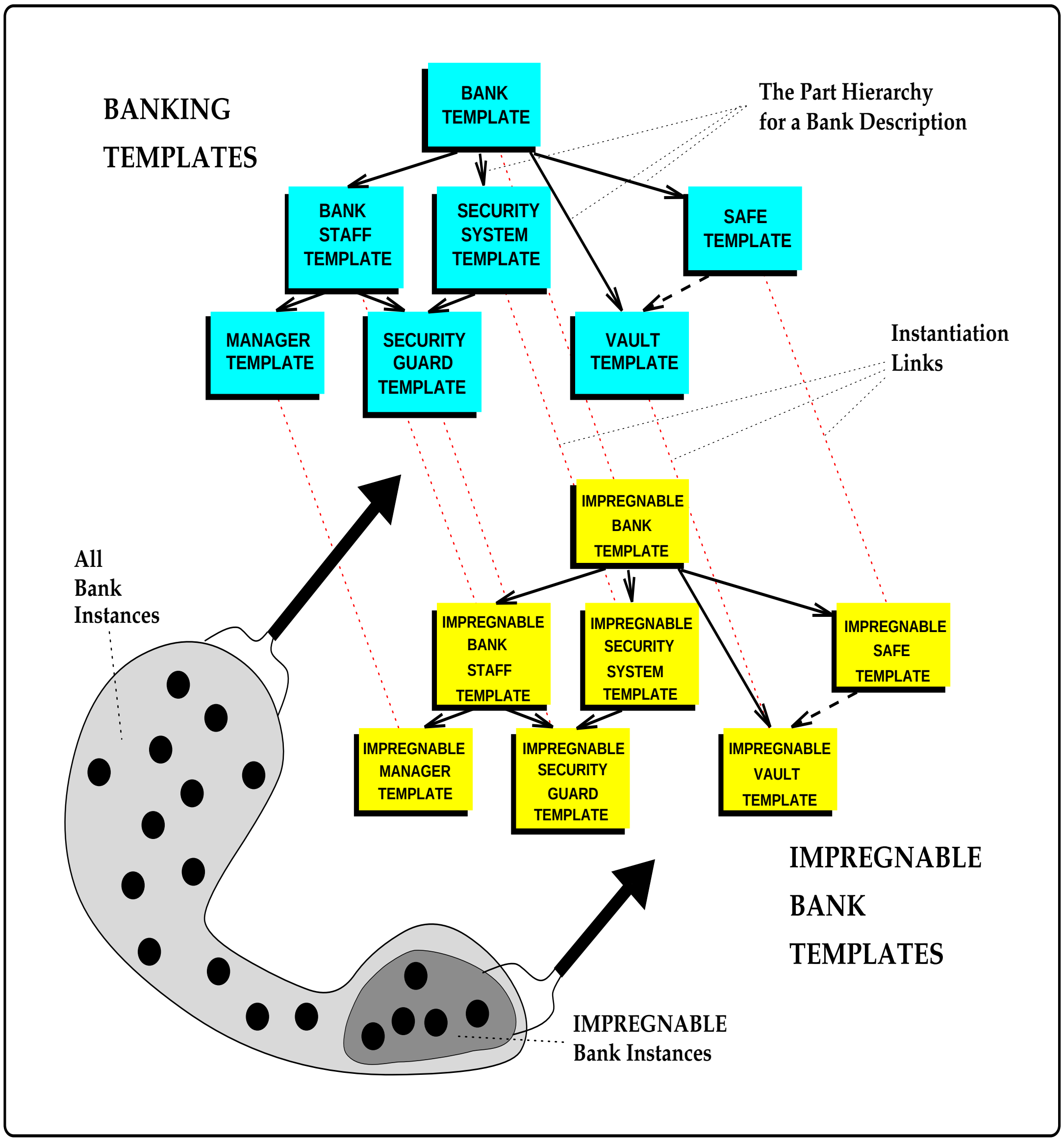


Figure F.1: Deriving a Complex Case Description from Multiple Templates

Appendix G

Conditions for Context Representation

The following is a list of the general conditions for which a positive context link is created between a context node, $node_c$, and a relation, rel, in the memory model developed in this thesis. For each condition, a brief explanation is also given as to why the link should be created.

- **If rel leads directly out of $node_c$.** In other words, every concept is a context for its own features. Hence, if attention is focused on that concept, all knowledge directly associated with it can be accessed.

- **If $node_c$ is the parent type for rel.** This is highly important because it permits the definition of retrieval paths that are akin to queries in a relational data base. That is to say, this condition allows memory retrievals to be characterised in terms of generic relational paths.

- **If $node_c$ causes a worthwhile focusing on a useful path including rel.** This follows from the arguments of section 3.3 and pertains to the strengthening of a successful, indirect retrieval path by a reduction in the potential for irrelevant branching from that path.

Appendix H

Activation Passing Definitions

This appendix contains a number of formal definitions concerned with the activation passing scheme defined in chapter 4 of this thesis.

H.1 A Definition of Cross-Talk and Interference

Cross-talk and interference are ambiguity problems that may arise if there is a lack of an explicit representation of disjunction for activation distributions. The two phenomena are defined below.

Let D_i be an activation distribution, representing all nodes possessing activation after i steps from some initial set of source nodes, N_0. Assume an activation distribution grows as a series of steps guided by a sequence of constraining conditions $C_{D_i} = C_1, C_2, \ldots, C_i$[1]. All nodes that receive activation in the same step are considered to have equivalent status with respect to the constraints on the propagation process.

For the step constrained by C_i, let the nodes that receive activation during that step be $N_i = NextNodes(D_{i-1}, C_i)$. In other words, the set N_i is formed by the effect of the constraint C_i on the activation distribution that existed immediately before the ith step. Similarly, let $S_i = Senders(D_{i-1}, C_i)$ be the set of nodes from the activation distribution immediately preceding the ith step that are enabled to propagate activation by constraint C_i.

The history of activation spreading can be defined as the sequence of sets of *all* nodes that receive activation at each conditional step in the activation propagation, $History(D_i) = \{N_0, N_1, \ldots, N_i\}$. It follows that the current activation distribution is the union of nodes from all stages in the distribution history, $D_i = N_0 \cup N_1 \cup \ldots \cup N_i$.

Cross-talk occurs in the ith step if two separate nodes send activation during that step and, at some point in the preceding history of the activation distribution, the two nodes had different statuses (i.e. received activation during differing steps). Formally, this can be expressed as:

[1]The representation of context, introduced in chapter 2, is shown in section 4.3 to provide the primary constraint on the activation spreading process developed in this thesis.

Condition 2 $CROSSTALK_i \equiv$
$$\exists n_1, n_2 \in S_i . n_1 \neq n_2 \wedge$$
$$\exists j . 0 \leq j < i \wedge$$
$$(((n_1 \in N_j) \wedge \neg(n_2 \in N_j)) \vee$$
$$((n_2 \in N_j) \wedge \neg(n_1 \in N_j)))$$

Interference occurs if activation is passed to a node that already possess activation.

Condition 3 $INTERFERENCE_i \equiv$
$$\exists n_1 \in N_i . n_1 \in D_{i-1}$$

H.2 Deriving the Structure Link Inferential Strength

This section gives a worked derivation of the equation for $IS(Dets(sl))$, the inferential strength from a subset of the detail nodes of structure link sl to its abstract node.

In section 4.2.2 the following factors were defined:

- N_a, the total number of different abstract nodes for all structure links in the knowledge base.
- d_{tot}, the total number of different detail nodes from all structure links.
- d_{ave}, the average number of detail nodes per structure link.
- p, the proportion of detail nodes *not* known for structure link sl.

The set of all possible detail nodes defines the dimensions of a *"feature space"*. It is assumed here that the abstract nodes are distributed evenly throughout this space. From a given set of detail nodes, an associated set of *selected* abstract nodes exists, such that the identified set of detail nodes is a subset of the detail nodes associated with each selected abstract node. The inferential strength, $IS(Dets(sl))$, is defined as the reciprocal of the size of the set of abstract nodes that are selected by the *known* details of sl.

On average, for a given detail node, the number of abstract nodes, k, that involve the detail node in their description is given by:

$$k = \frac{N_a * d_{ave}}{d_{tot}} \tag{H.1}$$

Hence, if one detail node is known for sl, the set of selected abstract nodes is of size k.

Given two randomly generated sets of abstract nodes of sizes s_1 and s_2, then the total number of pairings of abstract nodes between these two sets is given $s1 * s2$. Since each pairing has a probability of $\frac{1}{N_a}$ of linking to identical abstract nodes, the size of the intersection of the two sets is (to a close approximation) given by $\frac{s_1 * s_2}{N_a}$[2].

[2]Note that this assumes that $s_1, s_2 \ll N_a$.

It follows that, if there are two detail nodes known for sl, two sets of k abstract nodes are intersected. The resultant set of abstract nodes that contain *both* detail nodes in their description has a size of $\frac{k^2}{N_a}$. More generally, if x detail nodes are known then the number of selected abstract nodes is given by:

$$NumbSelectedAbst = k(\frac{k}{N_a})^{x-1} \tag{H.2}$$

Now it is assumed that $x = (1-p)d_{ave}$. It follows that the strength of inference can be defined as:

$$IS(Dets(sl)) = \frac{1}{k(\frac{k}{N_a})^{(1-p)d_{ave}-1} + 1} \tag{H.3}$$

From the definition of k given above this simplifies to:

$$IS(Dets(sl)) = \frac{1}{N_a(\frac{d_{ave}}{d_{tot}})^{(1-p)d_{ave}} + 1} \tag{H.4}$$

Some noteworthy points can be made about this definition;

- As the total number of detail nodes, d_{tot}, increases (for a fixed N_a) the ratio $\frac{d_{ave}}{d_{tot}}$ tends to 0, hence the inferential strength tends to 1. This can be explained by the fact that the "feature space" in which the abstract nodes are distributed is increased in size. It follows that the probability of any two abstract nodes being selected by a given set of details is effectively decreased.

- As the number of *known* details increases (i.e. $p \to 0$) the inferential strength increases.

- As the total number of abstract nodes, N_a, increases (for a fixed d_{tot}) the inferential strength tends to 0. This is because the density of abstract nodes in "feature space" is increased, which in turn increases the chance that any two abstract nodes are selected by a given set of details.

Appendix I

Colour Change, Cross-Talk and Interference

This appendix discusses a simple example of the use of colours to prevent cross-talk and interference in an activation propagation. The purpose is to illustrate the methodology for activation colour generation. The appendix is based around the example illustrated in figure I.1 [1], and explained below.

In step 1, all **"part"** relations out of **"BANK"** are traversed, reaching three component parts, namely **"SAFE"**, **"ENTRANCE"** and **"SECURITY SYSTEM"**. At this point there is a possibility of cross-talk (as defined in section 4.2.1) because all the bank parts themselves have parts. If the **"part"** relation is selected again, things that are *parts of parts of the bank* will be confused with things that are *parts of the bank*. To avoid this, the initial activation distribution is split into two uniquely coloured activations during step 1.

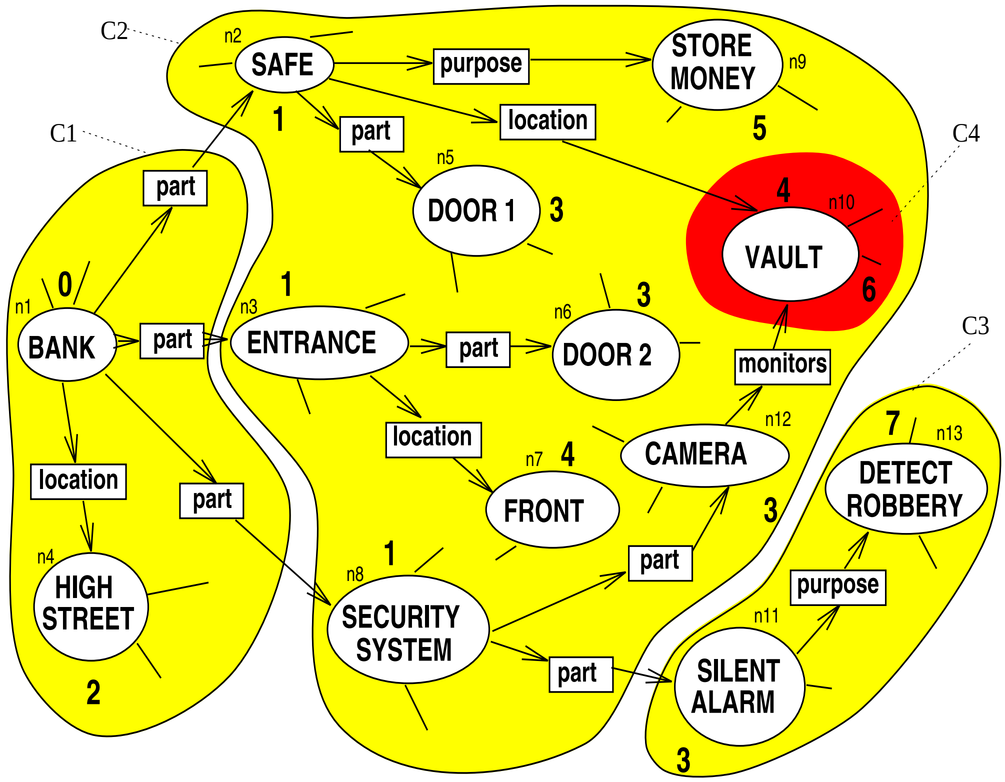
Step 3 selects the **"part"** relation for the second activation distribution, hence finding the parts of parts of the bank, namely **"DOOR 1"**, **"DOOR 2"**, **"CAMERA"** and **"SILENT ALARM"**. Cross-talk occurs again because **"SILENT ALARM"** and **"SAFE"** both have an outward **"purpose"** relation. A third coloured activation distribution is created to avoid confusion between things that are *purposes of parts of the bank* and things that are *purposes of parts of parts of the bank*.

Step 4 expands the second activation distribution by traversing the **"location"** relations. Amongst the locations of parts of the bank that are accessed is **"VAULT"**. In step 6, the second activation distribution reaches **"VAULT"** via a different path. To distinguish between these two accesses of **"VAULT"**, the second access causes a fourth coloured activation distribution to be generated. This is, therefore, an example of colour generation to avoid interference.

The rest of the example (steps 2, 4, 5 and 7) expand the four disjoint distributions without encountering cross-talk or interference. Hence, for these steps, no colour changes are required. The conclusion that can be drawn is that, generally, the granularity repre-

[1] For simplicity, in this example it is assumed that the constraints on activation passing select all relations of a particular type.

sented by the *necessary* colour changes will be more coarse than the granularity of the network itself.



Step No.	Activation Distribution (History - Colour)		Selected Relations	Result (History - Colour)	
1	(n1)	C1	"part"	(n1)	C1
				(n2,n3,n8)	C2
2	(n1)	C1	"location"	(n1)(n4)	C1
3	(n2,n3,n8)	C2	"part"	(n2,n3,n8)(n5,n6,n12)	C2
				(n11)	C3
4	(n2,n3,n8)(n5,n6,n12)	C2	"location"	(n2,n3,n8)(n5,n6,n12)(n10,n7)	C2
5	(n2,n3,n8)(n5,n6,n12)(n10,n7)	C2	"purpose"	(n2,n3,n8)(n5,n6,n12)(n10,n7)(n9)	C2
6	(n2,n3,n8)(n5,n6,n12)(n10,n7)(n9)	C2	"monitors"	(n10)	C4
7	(n11)	C3	"purpose"	(n11)(n13)	C3

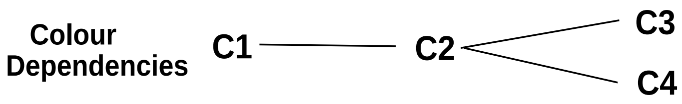


Figure I.1: Colour Changes in an Example Activation Distribution

Appendix J

An Example of Preserving Case Structure

This appendix describes a worked example of the scheme proposed on pages 60 to 63 for the biasing of retrieval towards source cases that correspond, in a fine-grained way, to the structure of a target case. Consider a simplified target case description:

> CASE 1 - John Smith, wearing a balaclava and toting a gun, robbed the hardware shop of its money.

Figure J.1 shows the initial activation distribution within the target case. This distribution is generated by propagation down the structure link hierarchy associated with the target case. Note that a unique colour of activation is generated for each node in the representation, allowing the individual feature-based retrievals to be performed disjointly. A colour dependency rule is generated for each structure link to reflect the relationship between the activation colour associated with its abstract node and the colours of activation in the detail nodes. These rules are stored and used later in phase 3 of retrieval to guide the merging of activation colours in propagation up the candidate source case representations.

The target case, CASE 1, along with all other cases in memory, is assumed to be connected to a network of world knowledge via template instantiation. Assume that activation passing from CASE 1 during phase 2 (see figure 4.3) of retrieval causes the following case descriptions to be partially activated:

> CASE 2 - Colin Brown broke into the hardware shop at night and stole its money.

> CASE 3 - Helen Jones, wearing a stocking on her head and carrying a machete, robbed the jewellers of their diamonds.

> CASE 4 - Fred McDonald went to the general store to buy a balaclava and a knife.

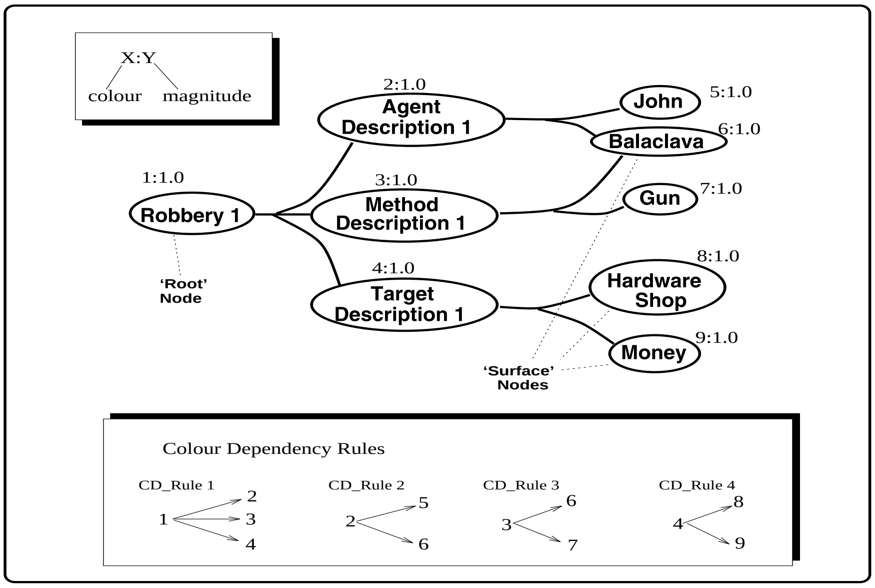


Figure J.1: The Initial Distribution of Coloured Activation in an Example Target Case Description

For simplicity, it is assumed that initial retrieval only places activation in the nodes representing the surface features of the source cases. That is to say, activation propagation is assumed to occur only from the leaf nodes of the target case hierarchy. Given that each target case node originally possesses activation of maximum magnitude (i.e. 1.0), the following simplified assumptions are made with regard to the amount of activation each source case leaf node receives:

- 1.0 - if the source and target case nodes represent *exactly* the same type of knowledge item.
- 0.5 - if there is an obvious but indirect semantic link between the type of knowledge represented by the source and target case nodes. For example, *"money"* and *"diamonds"* might be represented as types of *"valuable thing"* in the knowledge base, or *"John"* and *"Helen"* are both represented as instances of *"people"*.
- *0* - if no obvious link can be made between the knowledge item represented by the source case node and that represented by any node in the target case.

The initial placement of activation on the leaf nodes of the target case hierarchies is represented at the right-hand side of figure J.2. The next step is to allow this activation to propagate up the structure link hierarchies of each source case node. For each source case structure link, the detail node activations are merged according to the colour dependency rules and in accordance with the methodology presented on page 61[1].

[1]For simplicity and ease of understanding, it is assumed that the upward inferential strength ($IS(Dets(sl))$ as defined in equation 4.2) is equal to the proportion of activated detail nodes. This simplification does not detract from the principle being illustrated in this example.

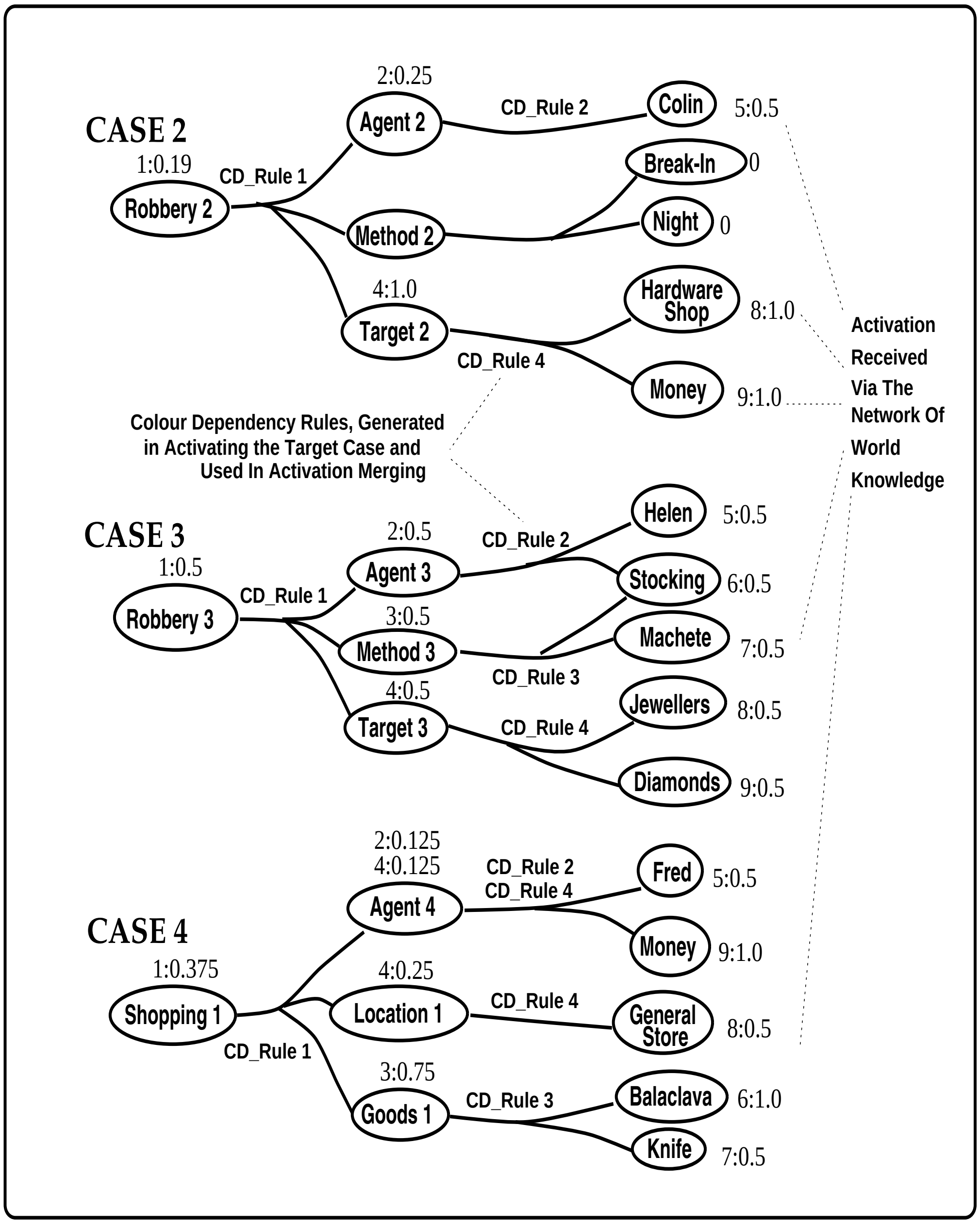


Figure J.2: An Example of the Propagation of Coloured Activation Up Source Case Structure Link Hierarchies

The result of applying these rules with respect to the example cases is shown in figure J.2. Note that CASE 3 would be retrieved with highest strength (measured in terms of the amount of activation that its root node receives). This reflects the complete structural correspondence between CASE 1 (the target case) and CASE 3. Note, in particular, that the surface level matching of CASE 3 is significantly less than that for CASE 4 but the structural dissimilarities between CASE 4 and CASE 1 are relatively high. Furthermore, comparing CASE 2 to CASE 3, it is clear that upward traversal of the structure link hierarchy gives a bias towards retrieval across the breadth of a description rather than towards retrieval on isolated features. These two observations support the arguments of pages 59 to 60.

Appendix K

Case Evaluation and Adaptation as Retrieval

In this appendix a worked example is given to illustrate the relationship between case evaluation, case adaptation and the retrieval mechanism developed in this thesis.

K.1 A Worked Example

The scenario for the example is that the gang of bank robbers have just fled the bank, carrying bulky money-bags and being closely pursued by the bank guards. To their horror, the gang find that the sports car they had intended to use for their getaway has been wheel-clamped. In addition, they can hear the sounds of police sirens in the near vicinity. It is assumed that this information is rapidly assimilated into the memories of the robbers and they are to attempt to recall past experiences of robbery escapes in order to attempt to device an alternative escape plan.

A hypothetical network representation of memory concerning robbery escapes is represented in figure K.1. A number of points about this figure need to be clarified:

- Cases are represented in a distributed manner and are held together by structure links.
- The information concerning the current robbery escape problem forms an entire target case. In contrast, the potential source cases in memory are the appropriate fragments of larger case structures concerning the description of entire robberies.
- The network of inter-connecting world knowledge appears to be somewhat unstructured. However, it is presumed that the knowledge presented in figure K.1 corresponds exactly to the portion of memory that is selected by having **"ROBBERY ESCAPE"** as a currently active context. In turn, this presupposes that the concept of a robbery escape is sufficiently important to a robber to warrant the formulation of the appropriate context links in memory.

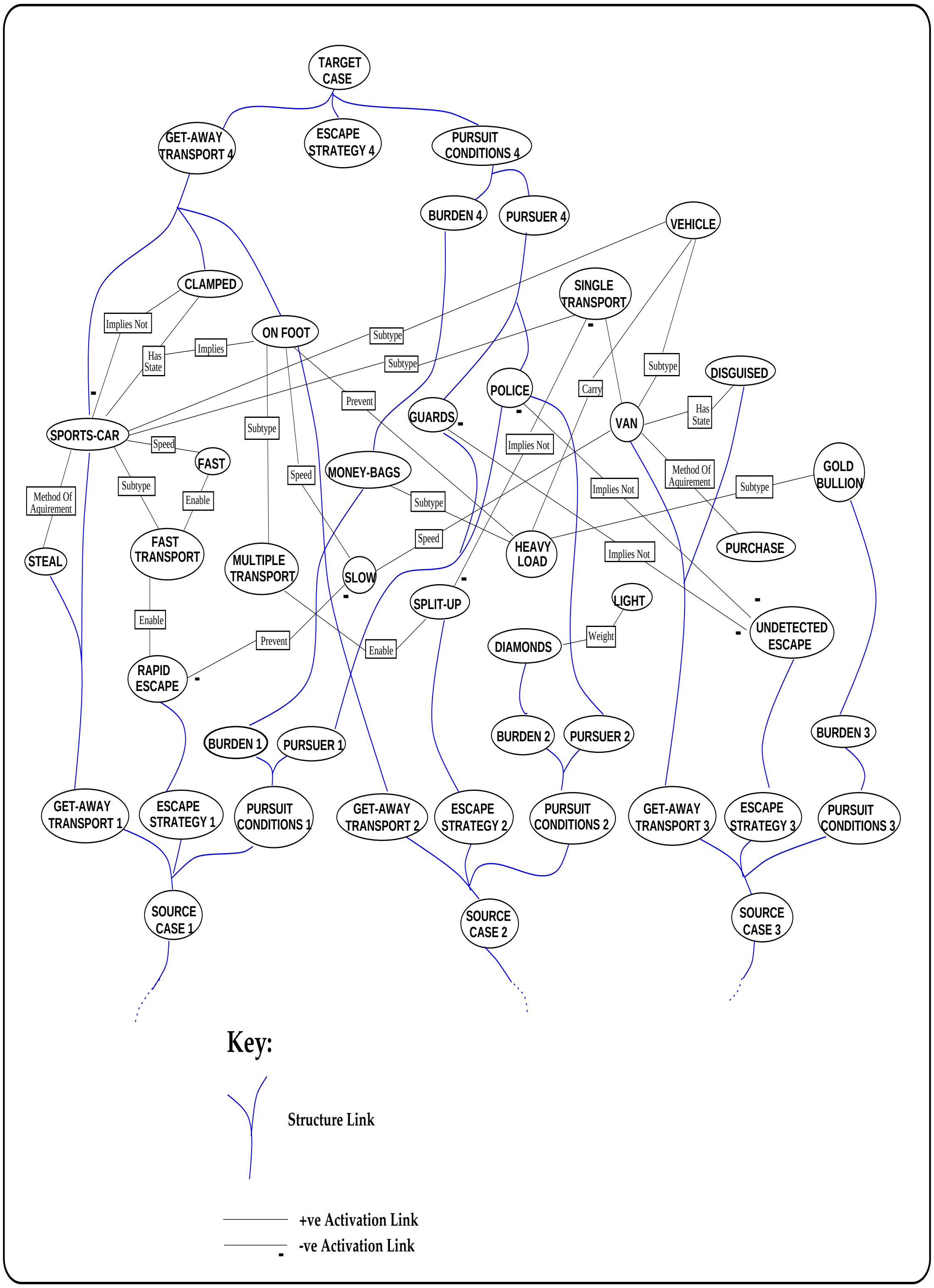


Figure K.1: An Example Portion of Memory Concerning Robbery Escapes

- The distinction between nodes representing general concepts and nodes representing specific instances is not clearly defined in the presented network. In practise, each component of a case should be represented as an instance of some more general type. However, for illustrative clarity in figure K.1 if two case descriptions have components of the same type then both components are represented by a single node corresponding to that type, rather than two separate nodes. In real implementations using CRASH the distinction between instances and general concepts should be made more correctly, hence case components are generally less directly linked.

Figure K.2 represents the initial retrieval performed from the robbery escape target case. Each node of the target case receives high positive activation during phase 1 of retrieval. During phase 2, this activation is propagated throughout the network representing world knowledge associated with robbery escapes. Finally, in phase 3, the activation placed on each node of a potential source case is amalgamated so that the amount of activation on each source case root node is a meaningful representation of the strength of retrieval for that case as a whole[1]. For the initial retrieval source case 1 is selected by retrieval.

Some important points can be made about reasoning that is performed as part of the initial retrieval process. In particular, the escape strategy of each source case is ruled out. This can be thought of as case evaluation during retrieval. What has occurred is the identification of reasons why each proposed escape strategy is inapplicable[2]. The explanation for each such evaluation is given below.

- The escape strategy for source case 1 is **"RAPID ESCAPE"**. This is positively implied by the existence of **"SPORTS CAR"** as a source case component. However, the importance of this component has been reduced because of the representation of the fact that sports car has been clamped. Moreover, the strength of inference of **"RAPID ESCAPE"** is further negated by the target case component **"ON FOOT"** which implies a slow speed, which is in turn incompatible with a rapid escape.

- The escape strategy for source case 2 is **"SPLIT UP"**. This is positively inferred from the concept **"MULTIPLE TRANSPORT"** associated with the source case component **"ON FOOT"**. However, this positive inference is reduced because there is negative evidence against **"ON FOOT"**. Specifically, the burden for the target case is a **"HEAVY LOAD"** which is contrary to the notion of escaping on foot.

[1] Note that, for the examples of this appendix, the activation possessed by the nodes is segregated into four broad bands based on its magnitude and polarity. A number of rules of thumb explain the activation distributions in figure K.2 and subsequent figures: activation drops one level for each two nodes traversed through on a prolonged path; where positive and negative activation meet at a single node, the residual is at one level below that of the positive source of activation; the activation propagated up a structure link is roughly the average of all detail activation levels (assuming all details are activated).

[2] Generally, this identification of problems corresponds to the detection of interaction between separate activation paths from the target case. It follows that the work on activation interference described in section 4.2.1 is of relevance here.

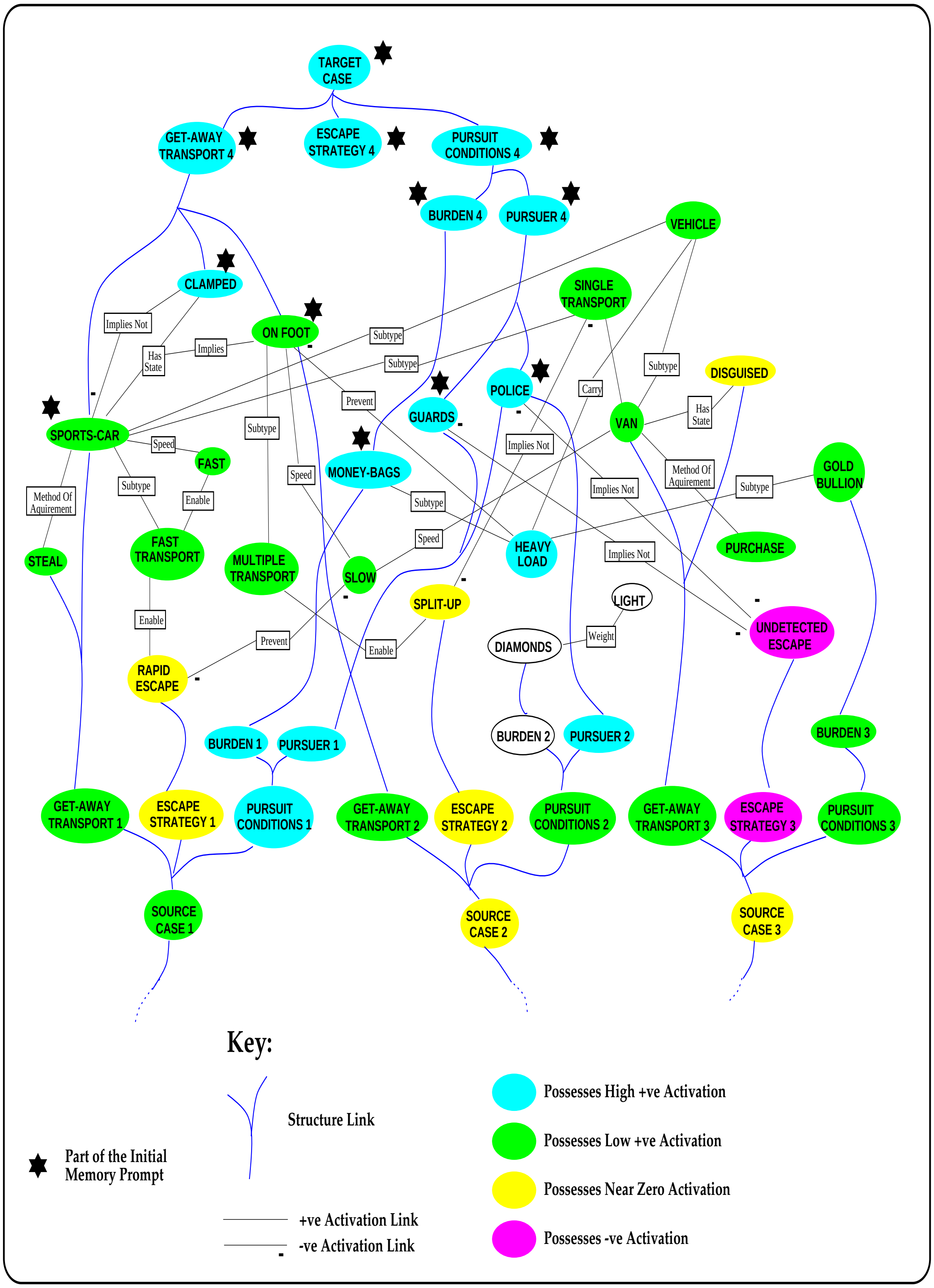


Figure K.2: An Initial Retrieval for the Robbery Escape Problem

Furthermore, the inclusion of **"SPORTS CAR"** as a target case feature provides (weak and indirect) evidence for the suggestion that the robbers escape using a single vehicle (i.e. **"SINGLE TRANSPORT"**) which in turn detracts from the evidence for the escape strategy being **"SPLIT UP"**.

- The escape strategy for case 3 is **"UNDETECTED ESCAPE"**. This is directly ruled out simply because pursuers (namely, **"POLICE"** and **"GUARD"**) are components of the target case.

At the end of the initial retrieval source case 1 in its entirety is passed to the external controller for utilisation and external evaluation. Two suggestions are made with respect to the target case; that a stolen sports car be used as the get-away vehicle, and (more weakly) that rapid escape is the appropriate strategy. Generally, the external controller should attempt to assimilate the suggestions made by retrieval with the target case on a one-by-one basis, selecting the most highly activated piece of recalled knowledge first. Hence the suggestion of using a stolen sports car as the get-away vehicle is considered first. The applicability of this suggestion can only be determined by the external controller observing whether or not a sports car is available to be stolen[3] (i.e. evaluation *cannot* be based on memory retrieval alone). Assuming that no sports car is available the advice must be *adapted*. The memory can aid the external controller in performing this adaptation through the suggestion of alternatives to the various knowledge items that constitute the suggestion.

It follows that a second retrieval takes place, for which the seed nodes are just the nodes that constitute the source case advice that has been rejected for the target case. In addition, the search path for this retrieval is constrainted to consider only "is/has subtype" relations. For the retrieval depicted in figure K.3 this implements adaptation either by seeking a suitable alternative to **"SPORTS CAR"** as something to steal or an alternative to **"STEAL"** as a means of acquiring a sports car. In particular, amongst the alternatives to **"SPORTS CAR"** that are suggested by this retrieval are **"VEHICLE"** and **"SINGLE TRANSPORT"**. This constitutes a strategy for adaptation that replaces a conflicting component with a generalisation of that component.

The second retrieval may be interpreted by the external controller to be the suggestion of *"steal any available vehicle"*. Assume that outside the bank the robbers find a lorry and two motorcycles parked. It is, therefore, possible for the external controller to formulate two feasible modifications to the target case:

- Steal the lorry (a slow, single form of transport) as the get-away transportation for the target case.
- Steal the two motorcycles (a fast, multiple form of transport) as the get-away transportation for the target case.

[3]It is assumed that the original sports-car is ruled out by the external controller and that the advice offered by memory retrieval is interpreted as meaning "steal another sports car".

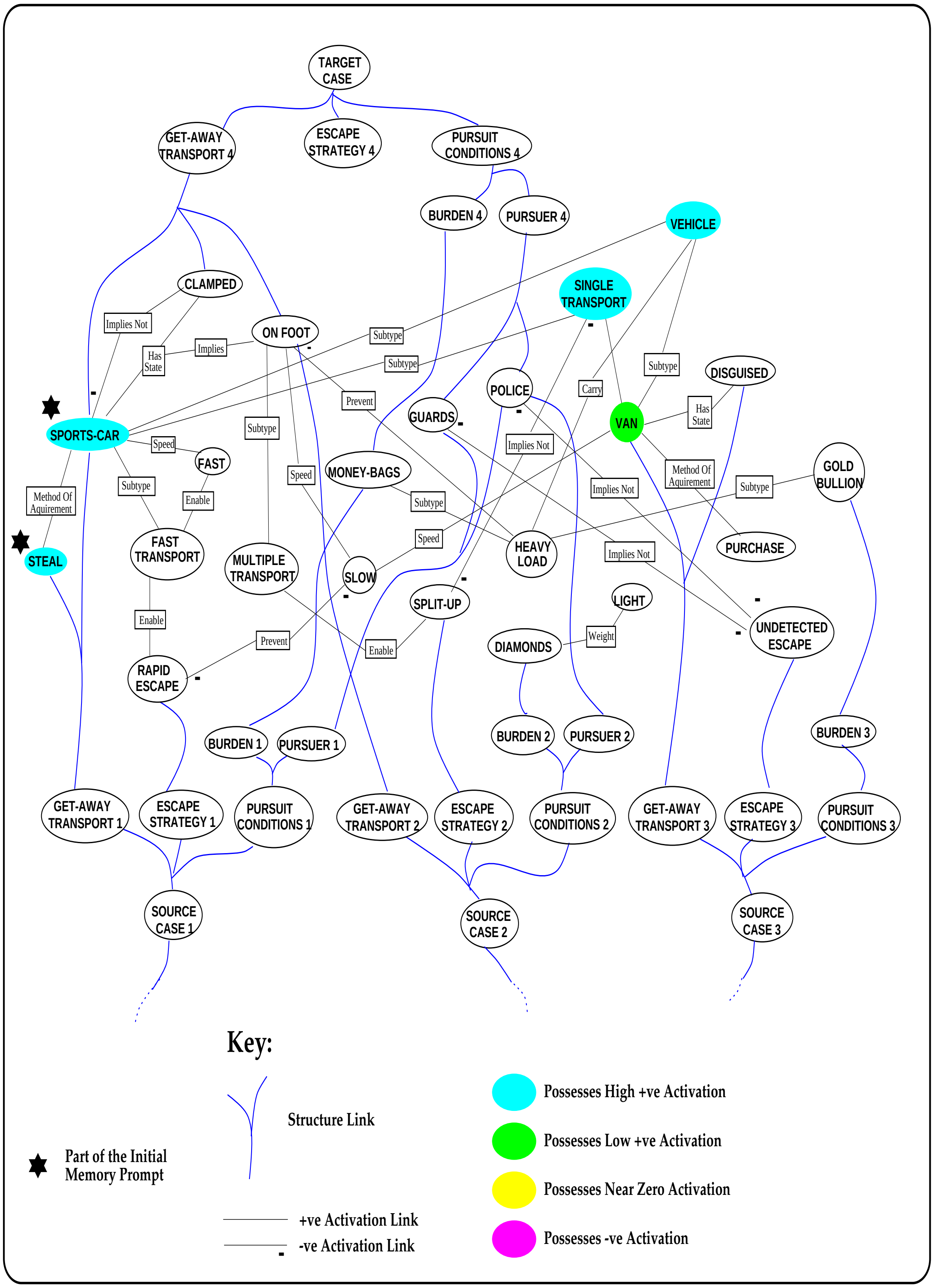


Figure K.3: An Example of Adaptation by Retrieval of an Alternative Case Component

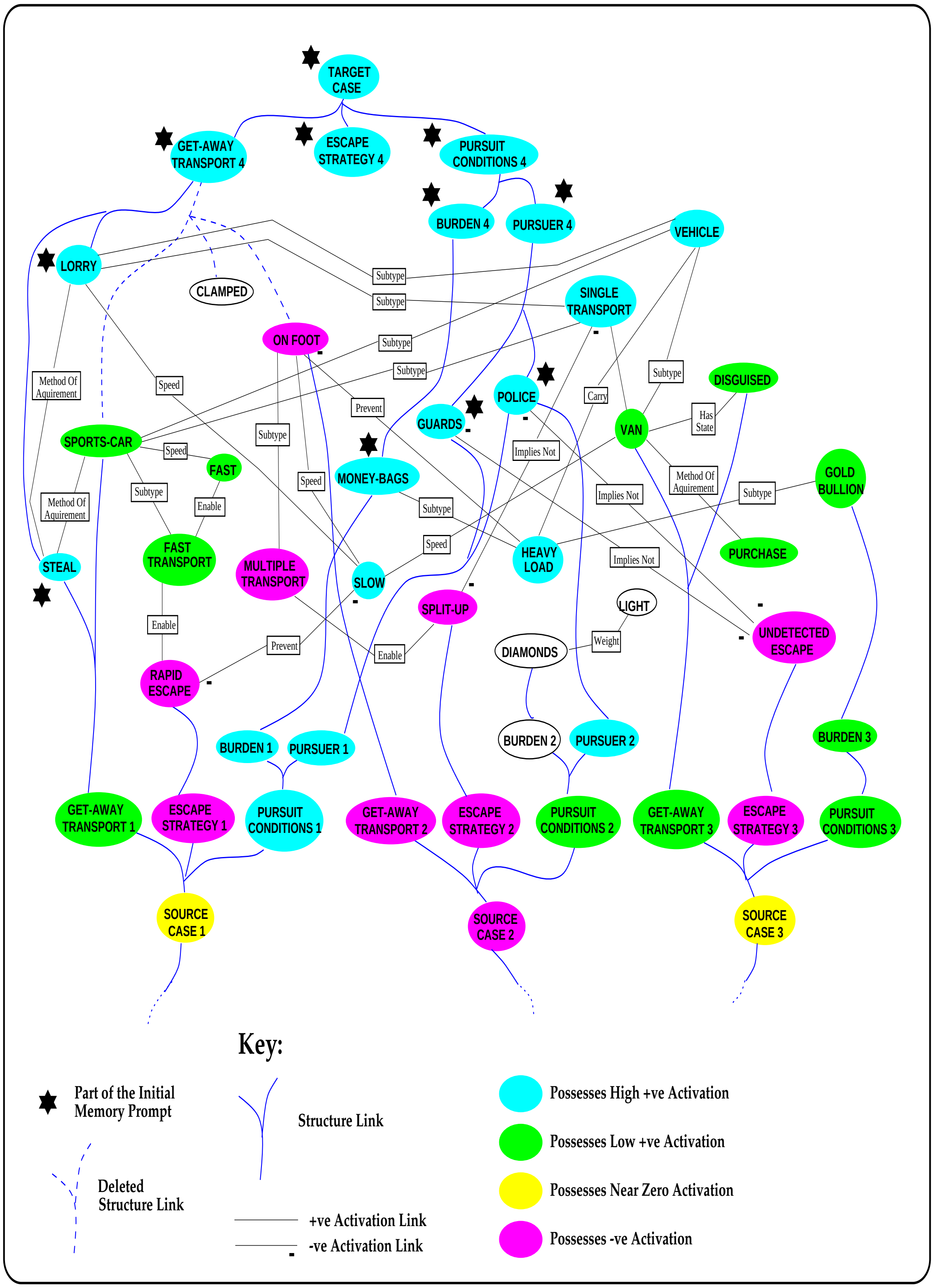


Figure K.4: A First Example of Evaluation of Transferred Information by Re-Retrieval

These two alternatives need to be evaluated. This can be done by exploiting memory as follows; for each alternative make appropriate modifications[4] to the representation of the target case and perform a retrieval similar to the initial retrieval. The results are shown in figures K.4 and K.5.

The results are that, in the case of possibly stealing a lorry (figure K.4), no source case is strongly retrieved and the escape strategy of each source case is again ruled out by evaluation during retrieval. Conversely, for the possibility of stealing the motorcycles two source cases (source case 1 and 2) are retrieved with relatively high strength. Source cases 1 suggests the feasibility of a rapid escape strategy where as source case 2 suggests that splitting-up is viable as a source case strategy. The external controller may chose to use either escape strategy or a combination of both. The point is that, by relying on retrieval, the choice of which vehicle to steal[5] has been resolved by a process of referring to precedent. The use of cases to resolve this type of evaluative decision is a potentially powerful form of reasoning [6].

K.2 Concluding Remarks

The example detailed above serves to demonstrate a number of important points about the relationship between the processes of evaluation, adaptation and retrieval. Firstly, if a sufficiently detailed representation of world knowledge exists and connects the cases in memory then some form of evaluation will generally take place *during* retrieval, in terms of the detection of interaction between the various components of the target case.

The example also demonstrates that evaluation cannot generally be completely based on retrieval. Retrieval will make suggestions as to knowledge that can be transferred from a retrieved source case to the target case. However, the initial target case description will usually not contain all information necessary for the validity of that advice to be fully assessed. Nevertheless the external controller can exploit the memory in order to make its task of evaluation and adaptation easier. For example, retrieval can suggest alternatives to the components of an incompatible body of knowledge whose application to the target case is being attempted. Moreover, any proposal that the external controller generates may be evaluated by memory simply by modifying the target case representation and allowing retrieval mechanism to reattempt the selection of source cases. The number and strength of source case retrievals provides precedent based evidence as to how problematic is the proposal.

What is suggested is a model of case adaptation by iterative retrieval. On each cycle the external controller: initiates a retrieval from the target case description, collects suggestions from the most strongly retrieved source cases and source case components, makes any modification to these suggestions based on available knowledge that is external

[4]The modifications to the target case should include deletion of the original description of the get-away vehicle and replacement with an encoding of the new possibility for a get-away vehicle; compare figure K.1 to figure K.4.

[5]Also resolved is that stealing one of the available vehicles is indeed a satisfactory solution to the get-away transportation part of the escape problem.

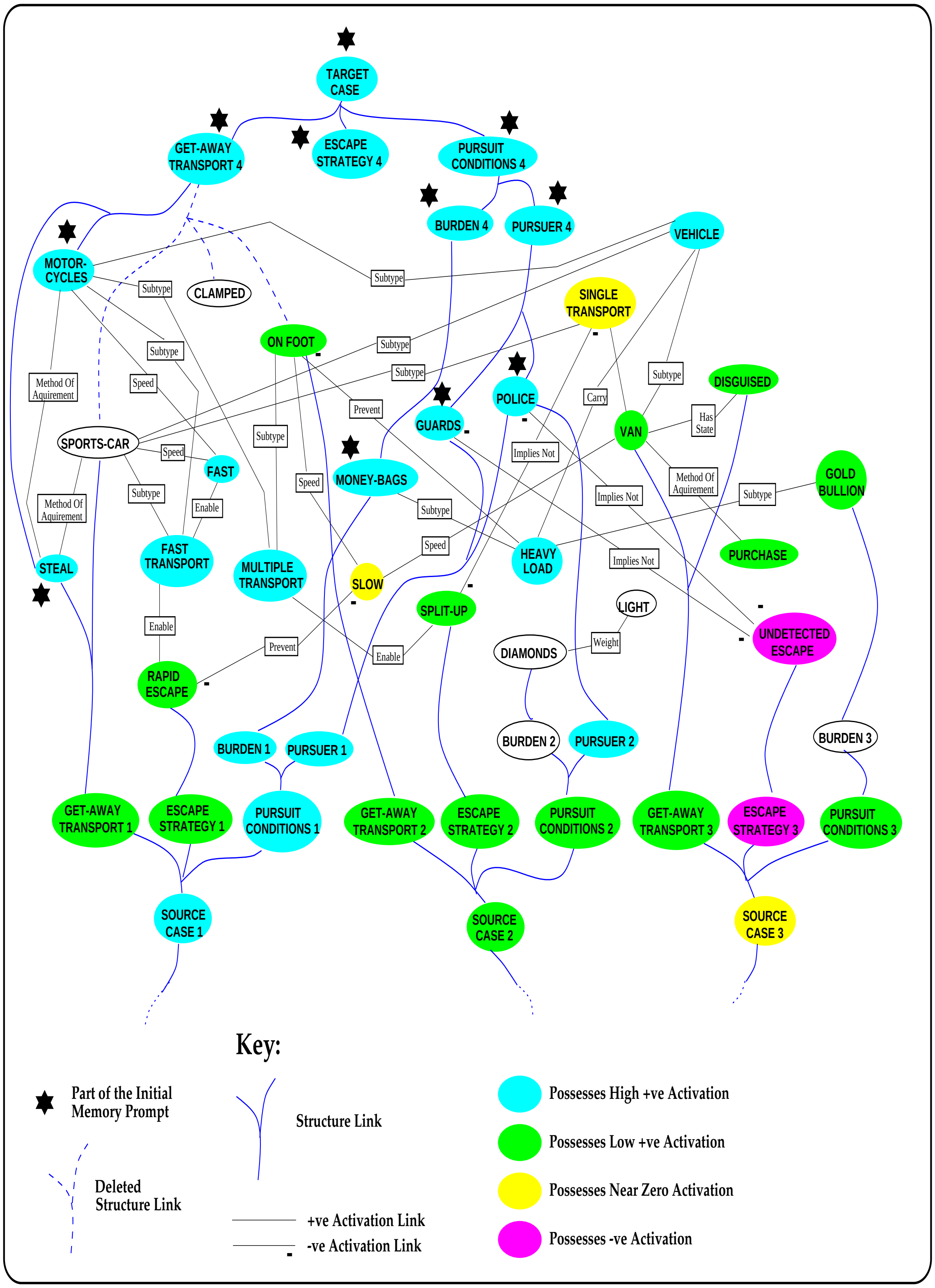


Figure K.5: A Second Example of Evaluation of Transferred Information by Re-Retrieval to main memory, modifies the target case representation to include the refined suggestion and repeats the cycle until the modified target case is sufficiently strongly supported by the set of source cases whose retrieval it prompts.

Appendix L

Preservation of Structure in Retrieval

The preservation of relational structure in a source to target case mapping has been identified as important in many similarity-based reasoning systems, particularly analogical reasoning systems [44, 25, 123, 138, 148]. However, the extensive copying of a relational structure is not the only factor that should influence a case retrieval, hence the intention in the proposed memory model is for the preservation of relational structure to act as a *bias* on retrieval. This appendix briefly explains the origins of this bias.

L.1 The Effect of Concurrent Relation Selection

Concurrent relation selection refers to an effect inherent in the memory model of this thesis, whereby the passage of activation within the relational structure of either the source or target case description will cause the corresponding structure to be simultaneously followed in the other description. Two aspects of the memory model conjoin to produce this effect:

- The facility to select a relation type nodes as an *active* contexts for the enablement of activation passing.
- The tendency of receiver link selection to chose links that lead to a destination node that already possesses activation (see page 68).

Consider the situation depicted in figure L.1. Phase 1, of retrieval scheme has completed, placing activation of a unique colour in each node of the target case. The second phase of the retrieval mechanism is underway, and a path, P, has been generated between a node of the target and source cases. Specifically, for the example of figure L.1, the path P has transferred activation of colour C_1 from the root node of the target case to the root node of the source case. It is important to emphasise that this initial pairing could have been forged between any of the case nodes; the arguments presented below only require that one such pairing exists.

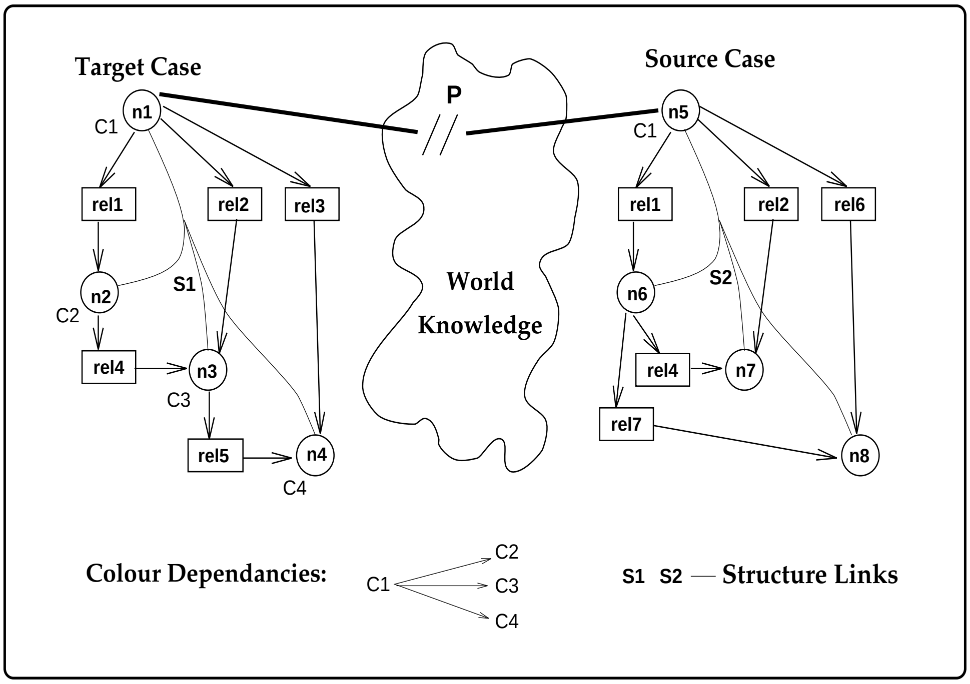


Figure L.1: Concurrent Relation Selection: Initial Status

Now assume that activation of colour C_1 is propagated further, guided by context selection. It is necessary to assume that, as part of the context selection strategy, the parent types of specific relation instances are frequently made active. Specifically, for the example, selection of the parent nodes of relations rel_1 to rel_5 will cause activation to permeate the target case description. Importantly, these context selections also cause activation to propagate along relation instances of the same type within the source case description, as shown in figure L.2. Hence, using relation parent types as active contexts tends to cause relational structure to be *'copied'* from one case description to another.

Now consider what happens if activation is permitted to propagate further. This propagation will include the spreading of activation through world knowledge from other nodes of the target description, involving activation colours other than C_1. Now, any propagation of activation that is of a colour that is either an ancestor or descendant of C_1 will, due to the receiver selection rules (page 68), tend to be directed towards those parts of the source case that have been primed with activation of colour C_1. Hence, secondary feature pairings between target and source cases can be anticipated.

Figure L.3 shows the idealised state at the end of phase 2 of activation propagation. In particular, it is assumed that two additional paths have been forged; $n_2 - n_6$ and $n_3 - n_7$. Hence, node n_6 of the source case possesses the same colour activation as does n_2 of the target case (the same applies to n_7 and n_3). The additional paths have transported the colour dependencies that originally existed for the structure link S_1 of the target case to the structure link S_2 of the source case. It is this equivalence of activation colour dependencies that is ultimately captured in phase 3 of the retrieval process, (see

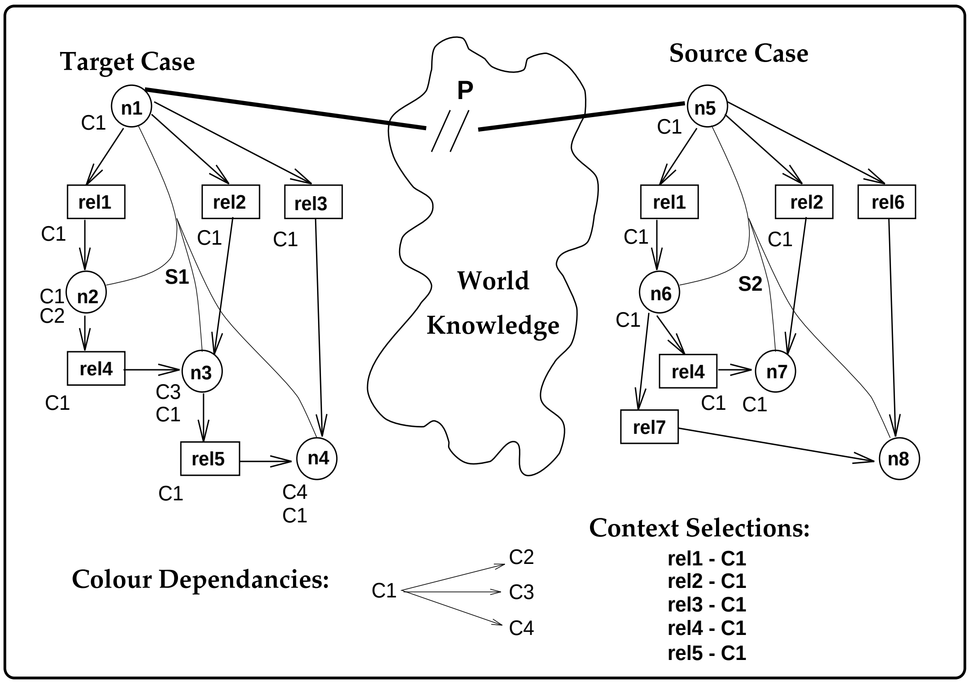


Figure L.2: Concurrent Relation Selection: After Initial Copying of the Relational Structure

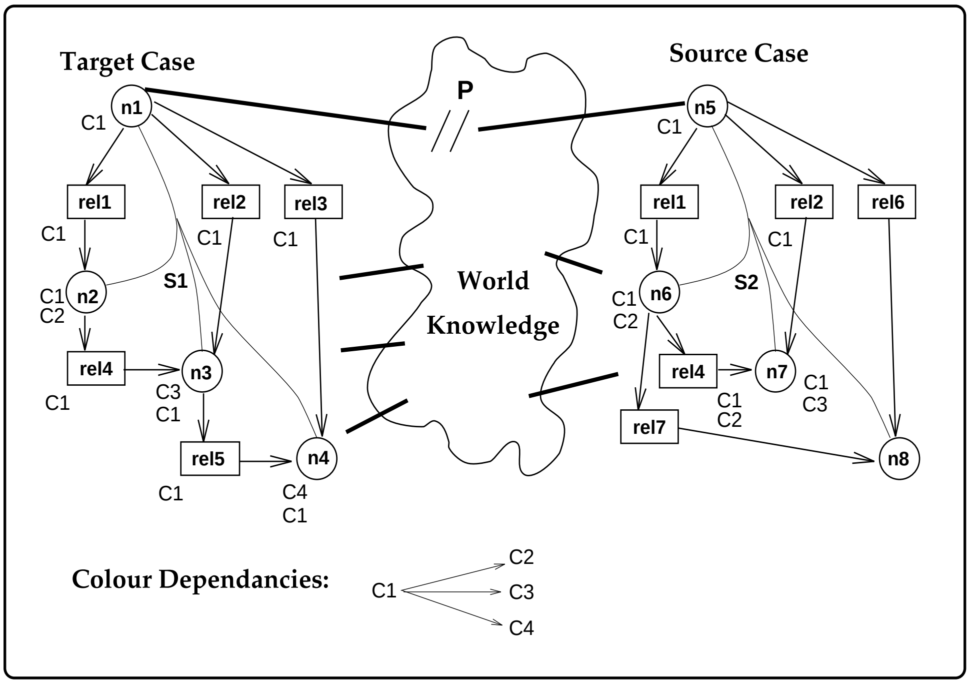


Figure L.3: Concurrent Relation Selection: Final State section 4.2.2).

It can be concluded from the above worked example that the extent to which the relational structures of two cases correspond effects two aspects of retrieval; the amount of activation that transfers between the two cases and the preservation of colour dependencies in the transferral. These two aspects have a direct baring on how strongly a source case is retrieved. Hence, the sharing of a relational structure is indeed a bias on retrieval.

Appendix M

The Bin-Packing Cases

A total of 22 individual cases comprise the collection elicited for use as source cases for the bin-packing problem. These 22 cases make up the collection of all problem-solving steps in the manual solution of four bin-packing problems. Figures M.1, M.2, M.3 and M.4 depict the cases for each of these problems.

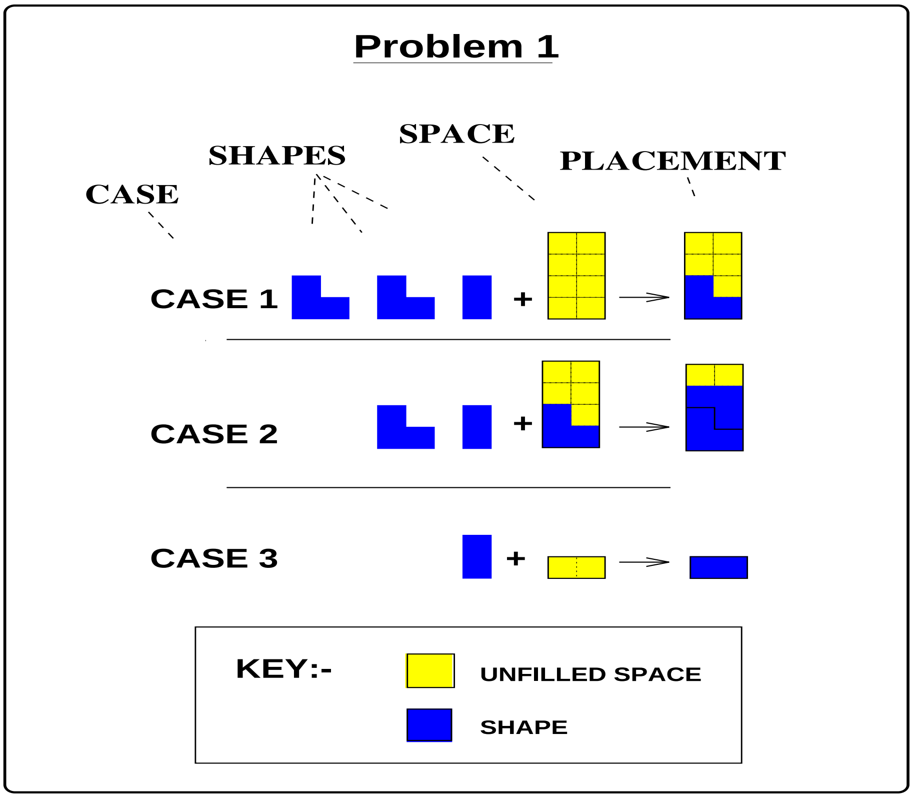


Figure M.1: Bin-Packing Cases 1, 2 and 3

The target case for the bin-packing case study is the initial state for a fifth bin-packing problem. This target case is depicted in figure M.5. Note that this description involves

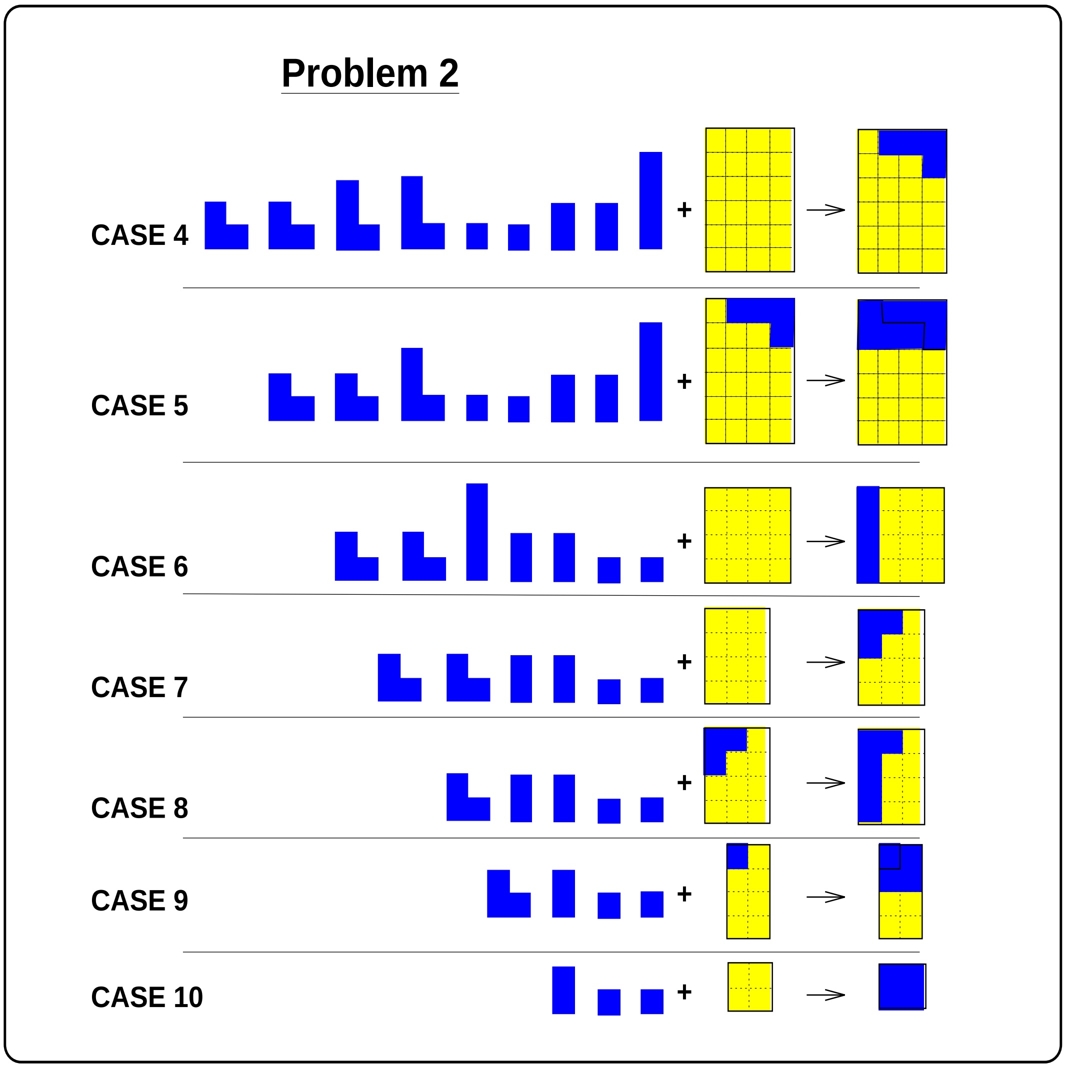


Figure M.2: Bin-Packing Cases 4 to 10

a complete characterisation of the initial state for the case but no description of the corresponding placement(s).

M.1 Three Separate Knowledge Bases

The corpus of bin-backing source cases described above are grouped by differing criteria to generate three distinct case bases for the experimental use of CRASH. This is mainly to support experiment 2 (see section 6.6) where the quality of retrieval is tested for varying case base sizes.

The cases bases are defined as follows:

- $CaseBase_{13}$: This involves cases 1 to 13. That is, it covers all problem-solving steps from the first three bin-packing problems depicted above.

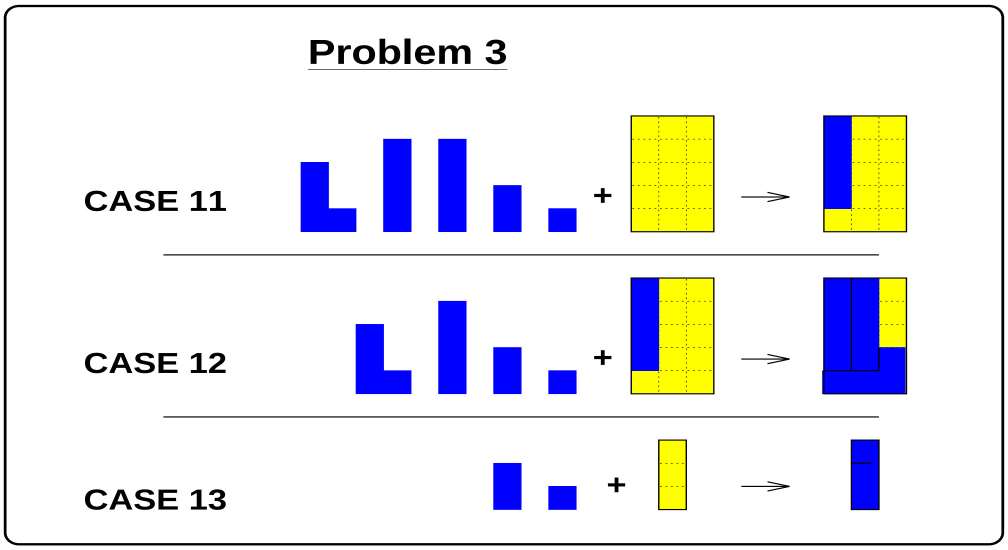


Figure M.3: Bin-Packing Cases 11, 12 and 13

- $CaseBase_{22}$: This is the full size case base and covers all 22 source cases generated from the four bin-packing problems.

- $CaseBase_8$: This is the smallest case base, being comprised of just 8 cases. The criteria for selection of cases for this case base was to include any case for which the initial unfilled area was perfectly rectangular. This leads to the inclusion of the following: case 1, case 3, case 4, case 6, case 7, case 10, case 11, case 13 and case 14.

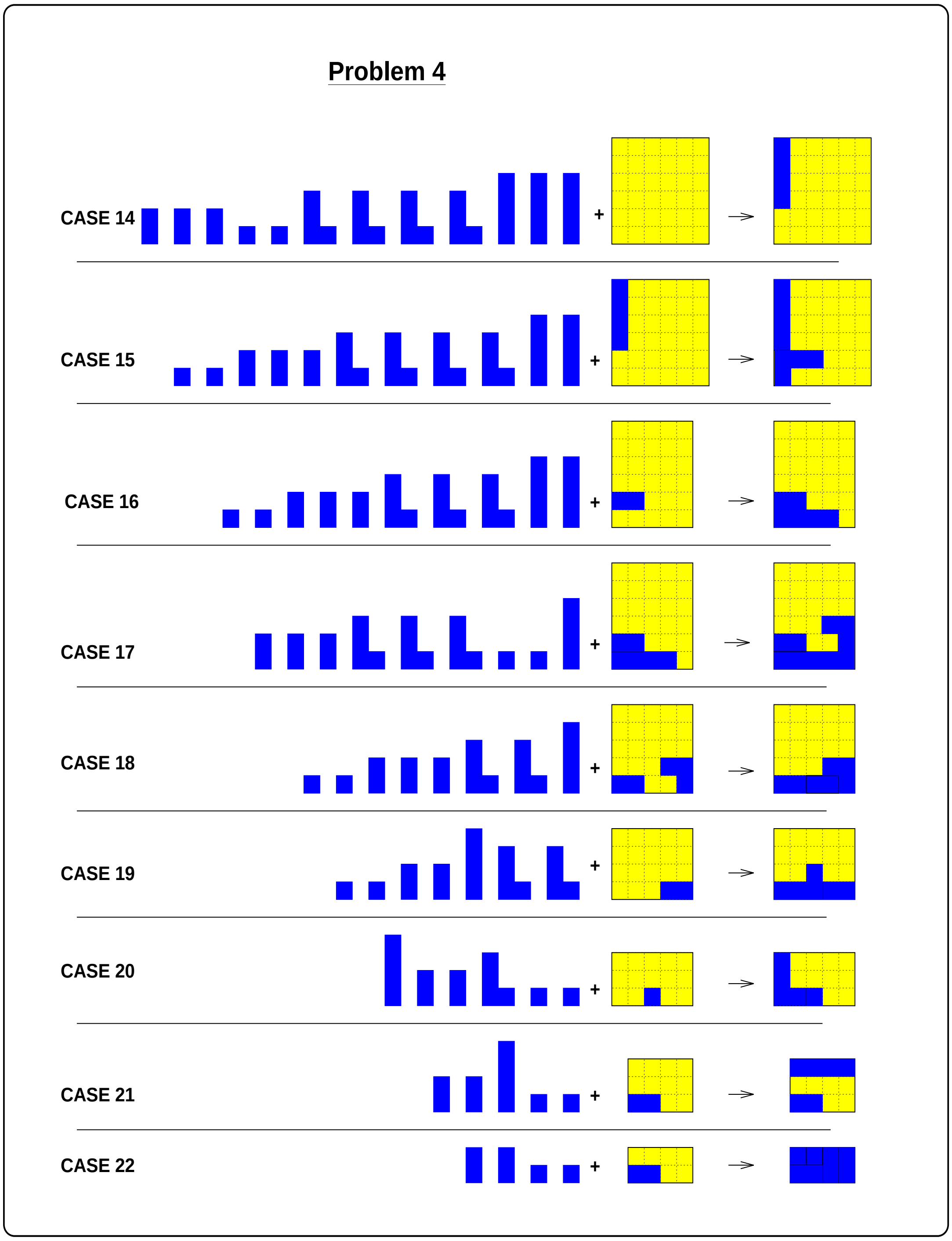


Figure M.4: Bin-Packing Cases 14 to 22

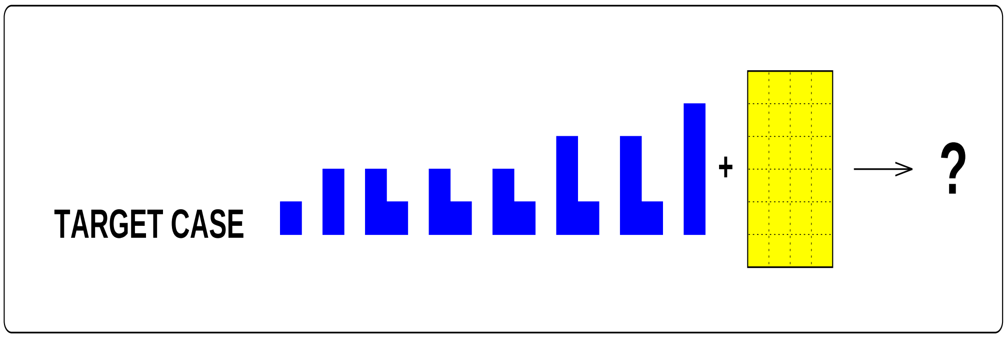


Figure M.5: Bin-Packing Target Case

Appendix N

Basis for Bench-Mark Source Case Orders

The purpose of this appendix is to establish meaningful ways of ordering the source cases depicted in appendix M with respect to the target case (also illustrated in that appendix). The rationale behind each ordering is explained in detail in the relevant sections.

N.1 Heuristic-Based Source Case Ordering

Recall that, in chapter 6, the following heuristics were recognised as providing sufficient guidance for the bin-packing problem:

- **Heuristic 1.** Place L-shapes before rectangles.
- **Heuristic 2.** Place the largest shapes first.
- **Heuristic 3.** Maximize the extent of contact of a placed shape with the remaining space's periphery.
- **Heuristic 4.** Choose a shape placement that maximises the rectangularity of the remaining space.
- **Heuristic 5.** The bounding box for the remaining space must be able to contain the bounding box of any of the remaining shapes.
- **Heuristic 6.** Given two remaining L-shapes of the same type, place them as a pair such that they are tessellated to form a rectangle.

Each source case provides one or more placements, each consisting of a specified shape and a list of *absolute* coordinates[1] within the remaining space. If each source case placement(s)

[1] The origin, $(1, 1)$, for each grid shown in appendix M is taken to be its bottom left-hand corner. It follows that for source cases such as case 7 (see figure M.2) where the placement is in a non-origin corner, on transfer to the target case the placement may become more central due to the difference in source and target case grid sizes.

Source Case	Applicable Heuristics	Heuristic Score
CASE 1	H1, H3, H5	12
CASE 2	H1, H5	8
CASE 3	H3, H5	6
CASE 4	H1, H2, H3, H5	17
CASE 5	H1, H2, H5	13
CASE 6	H2, H3, H5	11
CASE 7	H1, H5	8
CASE 8	H3, H5	6
CASE 9	H1, H5	8
CASE 10	H5	2
CASE 11	H2, H5	7
CASE 12	H1, H2, H5	13
CASE 13	H3, H5	6
CASE 14	H2, H3, H5	11
CASE 15	H1, H2, H5	13
CASE 16	H2, H3, H4, H5	14
CASE 17	H1, H2	11
CASE 18	H3, H5	6
CASE 19	H1, H2, H5	13
CASE 20	H1, H2, H3, H5	17
CASE 21	H2, H5	7
CASE 22	H5	2

Table N.1: Heuristics Associated with Source Cases After Application to the Target Case

is transferred directly to the target case (which by itself has no placement), which of the above heuristics agree with the transferred placement(s) can be determined. The results are summarised in the second column of table N.1.

The heuristics listed above are in *roughly* descending order of priority. It does not seem appropriate to model their applicability in a strict order. For example, it is not clear whether the placement of a small L-shape (heuristic 1) should take priority over the placement of a large rectangle (heuristic 2). The approach taken, in terms of modelling the heuristic-based order, is to assign a numeric weight to each heuristic according to the sequence; $Heuristic\,1 : Weight = 6, \ldots, Heuristic\,6 : Weight = 1$. Summing the total weight for the heuristics applicable for each source case produces the scores given in the third column of table N.1, from which a partial ordering of the cases can be deduced. Where two or more cases have the same heuristic score, the following analysis is used to help resolve the ambiguity:

- CASE 4 = CASE 20 : as the placement suggested by each are equivalent.

- CASE 5 = CASE 15 : as the placement suggested by each are equivalent.
- CASE 19 > (CASE 5, CASE 15) : as the placement suggested by CASE 19 comes closer to satisfying heuristic 3.
- CASE 12 < (CASE 5, CASE 15) : as the placement suggested by CASE 12 more seriously violates heuristic 4.
- CASE 14 = CASE 6 : as the placement suggested by each are equivalent.
- CASE 17 < (CASE 14, CASE 6) : as CASE 17 suggests a nonsensical placement with respect to the target case.
- CASE 11 > CASE 21 : as CASE 11 comes closer to satisfying heuristic 3.
- CASE 3 = CASE 8 = CASE 18 : as the placement suggested by each are equivalent.
- CASE 13 < (CASE 3, CASE 8, CASE 18) : as CASE 13 gives a more severe violation of heuristics 1 and 2.
- CASE 10 > CASE 22 : as CASE 10 gives a less severe violation of heuristics 1 and 2.

N.2 Semantic Similarity-Based Source Case Ordering

This section describes the measure of semantic similarity chosen to define the bench-mark ordering. A case can be thought of as a collection of sets of components, $\{S_1, \ldots, S_n\}$, where each set S_i represents all components of a particular type. The measurement of semantic similarity between two cases, C_s and C_t, can be defined as a series of set comparisons. For each set comparison, the elements of the two sets are assumed to be paired in such a way that they maximise the correspondence. Let S_i be the set of elements $\{e_{i,1}, \ldots, e_{i,m}\}$ and S_j be the set of elements $\{e_{j,1}, \ldots, e_{j,n}\}$, where each set element is an instance of a component from one of the two cases being compared. The semantic similarity measure for the set comparison is taken to be:

$$SetSim_{sem}(S_i, S_j) = ((\sum_{k=1}^{min(n,m)} PairSim_{sem}(e_{i,x_k}, e_{j,y_k})) - \frac{diff(n,m)}{max(n,m)}) \div max(n,m) \tag{N.1}$$

where $1 \leq x_k \leq m$ and $1 \leq y_k \leq n$. A restriction is that each element of the sets S_i and S_j can only be used in at most one pairing (i.e. in one value of $PairSim$).

Note that the measure of semantic similarity also takes into account the relative difference in size between the sets in two ways. Firstly, the relative difference in sizes is subtracted from the summation of the pairing similarities. Secondly, the total similarity measure is divided by the size of the *larger* set. As the total number of pairings considered is equal to the size of the smaller set, this limits the maximum value of the similarity measure to be equal to the ratio of the smaller to larger set sizes. In this way, the definition of similarity that is used includes a measure of the relative *difference* between the two case descriptions, although the measure lacks the asymmetric nature of Tversky's classic definition of similarity [143][2].

Generally speaking, each component of a case has its parent type[3] as one feature. Its other features are either links to sub-components or to surface-level values. The definition of semantic similarity for a pairing between two elements, e_x and e_y, can, therefore, be defined in a similar way to the definition given above for the similarity between two cases as a whole. Let the elements be equivalent to two sets of features, $\{f_{x,1}, \ldots, f_{x,p}\}$ and $\{f_{y,1}, \ldots, f_{y,q}\}$; the similarity measure is:

$$PairSim_{sem}(e_x, e_y) = (\sum_{r=1}^{min(p,q)} match(f_{x,a_k}, f_{y,b_k})) \div max(p,q) \tag{N.2}$$

where $1 \leq a_k \leq p$ and $1 \leq b_k \leq q$. Again, each case feature can be involved in a maximum of one pairing. The value of $match(f_a, f_b) = 1$ if f_a and f_b are themselves case components and have identical parent types *or* they correspond to identical surface feature values; otherwise the value is 0. Two exceptions are:

- If the two features are shapes for which the most specific parent types differ but which share an intermediate parent type (i.e. either **"L-shape"** or **"rectangle"**) then the match value is taken to be 0.5.
- For non-identical numeric surface feature values, the match is a measure of their relative difference.

For the decompositional case representation depicted in figure 6.2 there are nine component sets that have to be considered when defining a total semantic similarity comparison between two cases:

- $SetSim_{sem,1}$ defined in terms of the **"case"** nodes.
- $SetSim_{sem,2}$ defined in terms of the **"state"** nodes.
- $SetSim_{sem,3}$ defined in terms of the **"space"** nodes.
- $SetSim_{sem,4}$ defined in terms of the **"list of placement"** nodes.
- $SetSim_{sem,5}$ defined in terms of the **"placement"** nodes.

[2]Asymmetry could be introduced into the proposed similarity measure by replacing the *min* and *max* values with the size of the target case set. A symmetric measure of similarity was considered more appropriate for the bin-packing problem implemented in CRASH as all cases are described using the same standard decomposition.

[3]For **"shape"** components there are two parent types as the intermediate shape classes of **"L-shape"** and **"rectangle"** are present as domain knowledge in the bin-packing knowledge base.

- $SetSim_{sem,6}$ defined in terms of the **"list of coordinate"** nodes.
- $SetSim_{sem,7}$ defined in terms of the **"shape"** nodes.
- $SetSim_{sem,8}$ defined in terms of the **"coordinate"** nodes.
- $SetSim_{sem,9}$ defined in terms of the **"list of shape"** nodes.

Hence the total semantic similarity between cases C_s and C_t can be defined as:

$$SimCase_{sem}(C_s, C_t) = \frac{\sum_{i=1}^{9} W_i * SetSim_{sem,i}}{\sum_{j=1}^{9} W_j} \tag{N.3}$$

where W_i is the maximum size for $SetSim_{sem,i}$. From its definition, the bounds of this similarity measure are; $-1 \leq SimCase_{sem}(C_s, C_t) \leq 1$, 1 representing a complete correspondence and -1 representing a complete lack of correspondence. The results achieved applying this formula to the cases of appendix M are summarised in table N.2.

N.3 Structural Similarity-Based Source Case Ordering

A purely top-down measure of structural similarity is valid because all the cases used in the bin-packing case study share the same representation format. The main difference between the measure defined in this section and the semantic similarity defined in the last section is that, for structural similarity, the pairing of features is determined by the hierarchical structure of the case in a top-down manner.

Again, it is useful to consider a pairing between two case elements, e_x and e_y, that represent two paired case component instances and can be represented by two sets of features, $\{f_{x,1}, \ldots, f_{x,p}\}$ and $\{f_{y,1}, \ldots, f_{y,q}\}$ respectively. The structural similarity is defined for the pairing of these two elements as:

$$PairSim_{stuct}(e_x, e_y) = (match(e_x, e_y) + \sum_{k=1}^{min(p,q)} PairSim_{struct}(f_{x,a_k}, f_{y,b_k})) \div max(p, q) + 1 \tag{N.4}$$

where $1 \leq a_k \leq p$ and $1 \leq b_k \leq q$. The function $match(e_x, e_y)$ is the same as that defined on page 219. As for semantic similarity, each case feature can take part in a maximum of one pairing and it is assumed that the pairings, $f_{x,a} \rightarrow f_{y,b}$ are arranged so as to maximise the similarity measure. However, the measure of structural similarity cannot be localised to a direct comparison between two elements; it must also take into account the overall correspondence between the decompositional hierarchies that are rooted at each element. The measure of structural similarity between two entire cases is, therefore, equal to the structural similarity between their corresponding root nodes:

$$SimCase_{struct}(C_s, C_t) = PairSim_{struct}(Root(C_s), Root(C_t)). \tag{N.5}$$

Source Case	Semantic Similarity	Structural Similarity
CASE 1	0.187	0.733
CASE 2	0.118	0.728
CASE 3	-0.012	0.765
CASE 4	0.472	0.794
CASE 5	0.364	0.767
CASE 6	0.422	0.790
CASE 7	0.388	0.788
CASE 8	0.261	0.737
CASE 9	0.217	0.736
CASE 10	0.143	0.735
CASE 11	0.277	0.754
CASE 12	0.125	0.682
CASE 13	0.103	0.692
CASE 14	0.388	0.802
CASE 15	0.250	0.781
CASE 16	0.381	0.789
CASE 17	0.309	0.783
CASE 18	0.375	0.780
CASE 19	0.336	0.782
CASE 20	0.313	0.779
CASE 21	0.226	0.773
CASE 22	0.137	0.716

Table N.2: The Default Similarity Measures for the Bin-Packing Source Cases with Respect to the Target Case

Position	Heuristic Order	Semantic Order	Structural Order
1	CASE 4 and	CASE 4	CASE 14
2	CASE 20	CASE 6	CASE 4
3	CASE 16	CASE 7	CASE 6
4	CASE 19	CASE 14	CASE 16
5	CASE 5 and	CASE 16	CASE 7
6	CASE 15	CASE 18	CASE 17
7	CASE 12	CASE 5	CASE 19
8	CASE 1	CASE 19	CASE 15
9	CASE 6 and	CASE 20	CASE 18
10	CASE 14	CASE 17	CASE 20
11	CASE 17	CASE 11	CASE 21
12	CASE 7	CASE 8	CASE 5
13	CASE 2 and	CASE 15	CASE 3
14	CASE 9	CASE 21	CASE 11
15	CASE 11	CASE 9	CASE 8
16	CASE 21	CASE 1	CASE 9
17	CASE 18 and	CASE 10	CASE 10
18	CASE 3 and	CASE 22	CASE 1
19	CASE 8	CASE 12	CASE 2
20	CASE 13	CASE 2	CASE 22
21	CASE 10	CASE 13	CASE 13
22	CASE 22	CASE 3	CASE 12

Table N.3: The Bench-Mark Source Case Orderings

The value of structural similarity is bounded as; $0 \leq SimCase_{struct}(C_s, C_t) \leq 1$. The results of applying this recursive formula to the bin-packing cases is also summarised in table N.2. Note that all structural similarity scores for the source cases are relatively high, reflecting the uniformity in the structure of bin-packing case description structures.

N.4 A Summary of the Bench-Mark Source Case Orderings

Table N.3 provides a summary of the ordering of source cases with respect to each of the bench-mark relationships described in this appendix.

N.5 The Measurement of Order Deviation

So far in this appendix the three bench-mark relationships used to generate default source case orderings have been defined. In this section, a measure is described by which a source case ordering produced by a CRASH retrieval can be compared to these bench-marks. The metric that is used is the standard deviation, σ, in the case order. The *lower* the value of σ, the *higher* is the correspondence between two case orderings.

It is first necessary to determine the expected standard deviation between two random orderings of sets of distinct cases. Generally, for a list of length n, the average random ordering standard deviation is given by:

$$\sigma_{ave} = \sqrt{\frac{1}{n-1}\frac{1}{n}\sum_{i=1}^{n}(\sum_{j=1}^{i-1} j^2 + \sum_{k=1}^{n-i} k^2)} \tag{N.6}$$

It is also useful to determine the expected deviation from σ_{ave} for a given pair of randomly ordered case lists. Let this value be σ_σ. This value is useful for establishing the statistical significance of a measured case ordering deviation, σ, with respect to σ_{ave}.

Three sizes of case base are used in the experimentation (see section 6.6): 8 cases, 13 cases and 22 cases. For each case base the expected value of σ_{ave} is determined theoretically by equation N.6 and confirmed empirically using a large number (≈ 10000) of randomly generated case orderings. In addition, the value of σ_σ is determined empirically for each case base. The results are as follows:

- 8 Cases ($n = 8$): $\sigma_{ave} = 3.46$; $\sigma_\sigma = 0.71$
- 13 Cases ($n = 13$): $\sigma_{ave} = 5.51$; $\sigma_\sigma = 0.86$
- 22 Cases ($n = 22$): $\sigma_{ave} = 9.18$; $\sigma_\sigma = 1.04$

N.6 The Statistical Importance of Case Order Derivation Measures

The standard deviation between the case orderings of a retrieval and a bench-mark can be converted to a biasing factor, B_f, of range -1 (completely opposed) to +1 (completely corresponding) according to the following equation:

$$B_f = 1 - \frac{Prob(\sigma)}{0.5} \tag{N.7}$$

$Prob(\sigma)$ is the probability that two randomly generated case sequences will produce a standard deviation that is *less than or equal to* σ. Figures N.1, N.2 and N.3 represent the conversion from the order deviation (i.e. σ) to B_f for the three sizes of case base used in the experimentation described in this thesis. Also shown in the these figures are four distinct regions corresponding to a qualitative measure of the statistical significance of a given measure for σ. These regions are defined in terms of σ_σ as follows:-

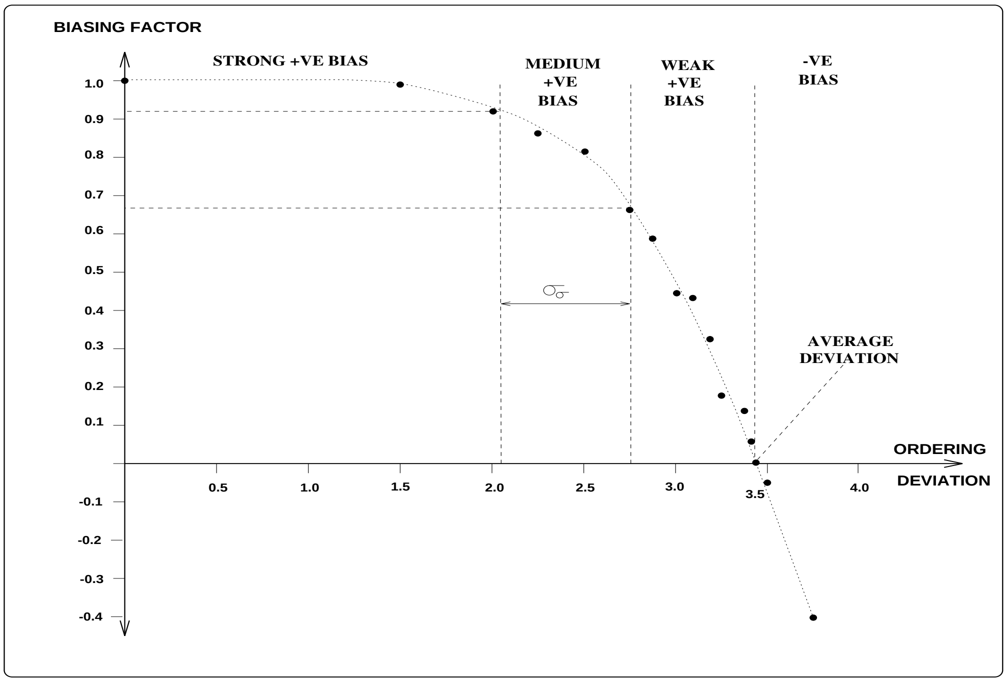


Figure N.1: Conversion of Standard Deviation of Case Orderings into a Biasing Measure (8 cases).

- **Negative Bias**. The value of σ is greater than that which is, on average, measured between two randomly generated case orderings.

- **Weak Positive Bias**. The value of σ lies between σ_{ave} and $\sigma_{ave}-\sigma_\sigma$. This has little statistical significance as, from the theory of statistics [69, chapter 18], about 68 percent of the values in a probability distribution are within one standard deviation of the mean.

- **Medium Positive Bias**. The value of σ lies between $\sigma_{ave} - \sigma_\sigma$ and $\sigma_{ave} - 2\sigma_\sigma$.

- **Strong Positive Bias**. The value of σ is less than $\sigma_{ave} - 2\sigma_\sigma$. This has high statistical relevance; there is only about a 1 in 20 chance that two randomly produced case orderings will deviate by this small an amount [69, chapter18].

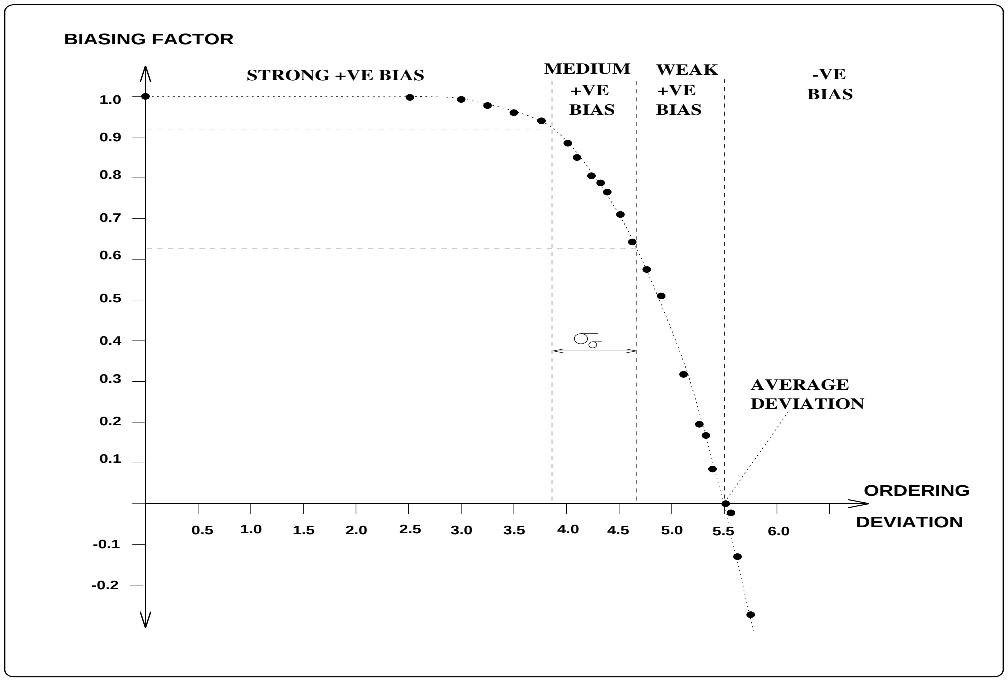


Figure N.2: Conversion of Standard Deviation of Case Orderings into a Biasing Measure (13 cases).

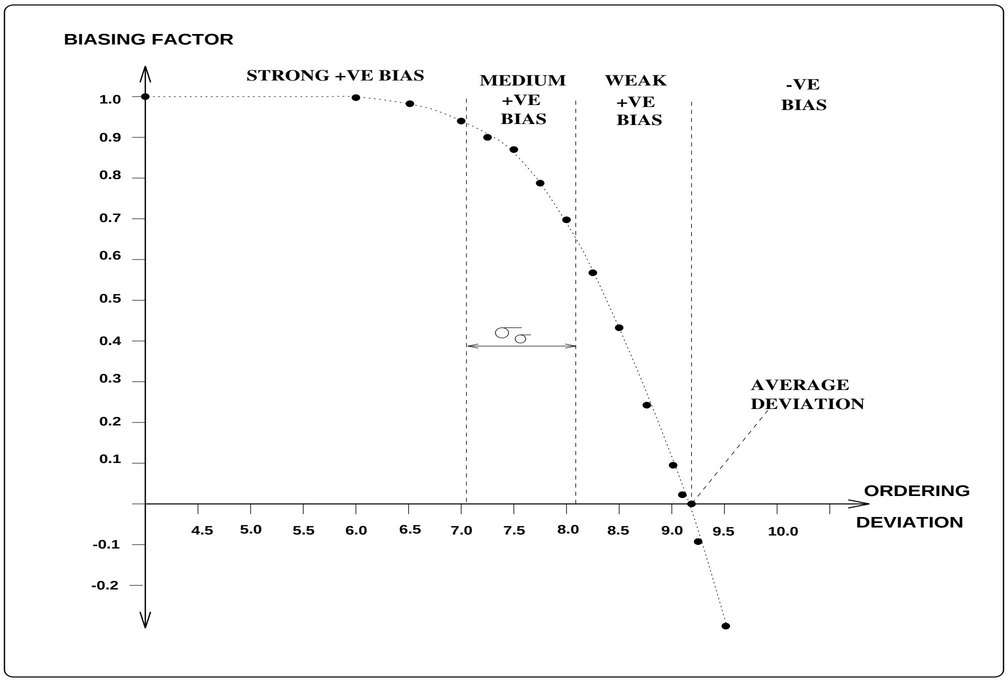


Figure N.3: Conversion of Standard Deviation of Case Orderings into a Biasing Measure (22 cases).

Appendix O

Bin-Packing Experiments

As was summarised in section 5.1.1, CRASH embodies the memory model developed in this thesis, including the 3 phase retrieval mechanism. This appendix details the experimental results achieved through the use of CRASH to solve the bin-packing problem.

O.1 Experiment 1

Experiment 1 is an investigation of the nature of feature-based retrieval using CRASH. One aim is simply to gauge the bias towards each of the bench-mark factors (see appendix N) that is realised by the retrieval mechanism. Secondly, the experiment provides a more in-depth investigation of the role of phase 3 of CRASH's retrieval.

O.1.1 Experiment 1: Description

This section starts by describing the context sequence used to control phase 2 of the experimental retrievals. This is followed by a note on other important parameter settings for the experiment.

A Predefined Context Sequence for "Bottom-Up" Retrieval

As is described in section 5.1.1, the second phase of CRASH's retrieval involves a search through its internal representation of 'world knowledge' guided by a sequence of context selections. For this experiment a predefined context sequence is used.

The strategy employed is that of direct retrieval from *all* nodes representing target case features applied in a bottom-up order. 'Direct' in this sense means that the connecting path between the target and source case nodes involves relations representing only one type of relationship. For example, by this scheme, two case components can be linked by being instances of the same general type or because they share the same feature value. By avoiding less direct paths, phenomena, such as *cross-talk* and *interference* (section 4.2.1) are avoided. In addition, because there is no discrimination as to what target case features are used to instigate source case retrieval, the context sequence allows the power of a purely similarity-based retrieval mechanism to be assessed.

The retrieval space defined by specific context sequence is characterised below and can best be understood by referring to the generic case structure shown in figure 6.2[1]:

1. For each *"shape"* of the target case, retrieve source cases via its *"size"* feature.
2. For the *"bounding box"* of the *"space"* component of the target case, perform retrieval via its *"x"* and *"y"* features.
3. For the *"space"* component of the target case, perform retrieval via its *"parity"* feature.
4. For each *"shape"* of the target case, retrieve source cases via its parent types. The order in which parent types are used is "most specific first"; first the direct parent type is used (e.g. *"small L-shape"*), next the relevant intermediate shape type is used (e.g. *"L-shape"*), and finally the general concept *"shape"* is used.
5. For all *"list of shapes"* in the target case, perform retrieval via its parent types (the concept *"list of shapes"* is used before the more generic concept *"list"*).
6. For the *"bounding box"* of the *"space"* component of the target case, perform retrieval via its parent type of *"coordinate"*.
7. For the *"space"* component of the target case, perform retrieval via its parent type.
8. For the *"state"* component of the target case, perform retrieval via its parent type.
9. For the (empty) *"list of placements"* component of the target case, perform retrieval via its parent types (the concept *"list of placements"* is used before the more general concept *"list"*).
10. For the root node of the target case, perform retrieval via its parent type, namely *"case"*.

Experiment 1: Parameter Settings

Experiment 1 is carried out solely on $CaseBase_{13}$ (see section M.1) which includes 13 source cases. The *branching constant*, B_c, is set to 3. As described in section 4.3.3, B_c limits the number of receivers that can be selected for any node that is sending activation. It follows that setting B_c to 3 causes the search that is performed during phase 2 of retrieval to be much less than exhaustive. This is important as it allows for a true assessment of the power of phase 3 in terms of the ability to amalgamate a sparse set of isolated feature-based retrievals into a more coherent whole.

[1] Note that the target case, represented in figure M.5 of appendix M does not possess all the components of the generic bin-packing case description given in figure 6.2. In particular, it has an empty list of placements and its initial state contains no filled area. The context sequence used to guide retrieval only accounts for retrieval paths traversable from the target case features that exist.

O.1.2 The Effect of Upward Activation Propagation

The result of using the context sequence detailed in section O.1.1 during phase 2 of CRASH's retrieval is a wide-ranging placement of activation in isolated nodes of the 13 source case descriptions. The series of experiments described in this section investigates how the gathering of these isolated retrievals during phase 3 can influence the ordering in the cases that are retrieved. Note that, at the end of retrieval, a case's activation is judged solely in terms of the amount of activation that reaches its root node.

Recall from section 4.2.2 that the amount of activation that an abstract node of a structure link receives is governed by the following two equations:

$$IS(Dets(sl)) = \frac{1}{N_a \gamma^{(1-p)f} + 1} \tag{O.1}$$

and

$$InAct(abst) = \eta IS(Dets(sl)) \times (Mean(TotAct(Dets(sl))) - Act(abst)) \tag{O.2}$$

Three parameters control the nature of activation passing up structure links:

- **The Proportionality Coefficient** - N_a.
- **The Proportionality Exponent** - γ.
- **The Upward Spread Coefficient** - η.

All other factors in these equations are dependent on the dynamic state of the activation distribution itself. Two experiments are proposed here; to vary γ keeping N_a and η constant, and to vary η keeping N_a and γ constant.

Experiment 1.1: Varying The Proportionality Exponent

For this part of the experiment the following settings were made; $N_a = 1.0$ and $\eta = 7.5$. Varying γ affects the sensitivity to missing details of the upward activation passing performed during phase 3 of the retrieval. As γ tends to 0 the sensitivity diminishes and as γ increases so sensitivity increases. In the experiment the range of γ is varied from 0.01 through to 0.4. The results in terms of the relative case ordering are shown in figure O.1[2]. The graph of figure O.1 can be further analysed by calculating the order deviation, with respect to the retrieval and each bench-mark, at each point where the retrieval ordering changes. The results are presented in figure O.2.

Several points are illustrated by figure O.2. Firstly, a clear and statistically significant bias is shown for retrieval with respect to both semantic and structural similarity. There is also a positive bias towards the heuristic-based ordering of cases, though this bias is weaker than for the other two bench-mark factors. This is evidence for the coupling between the three bench-mark factors which agrees with the predictions of section 5.2.1.

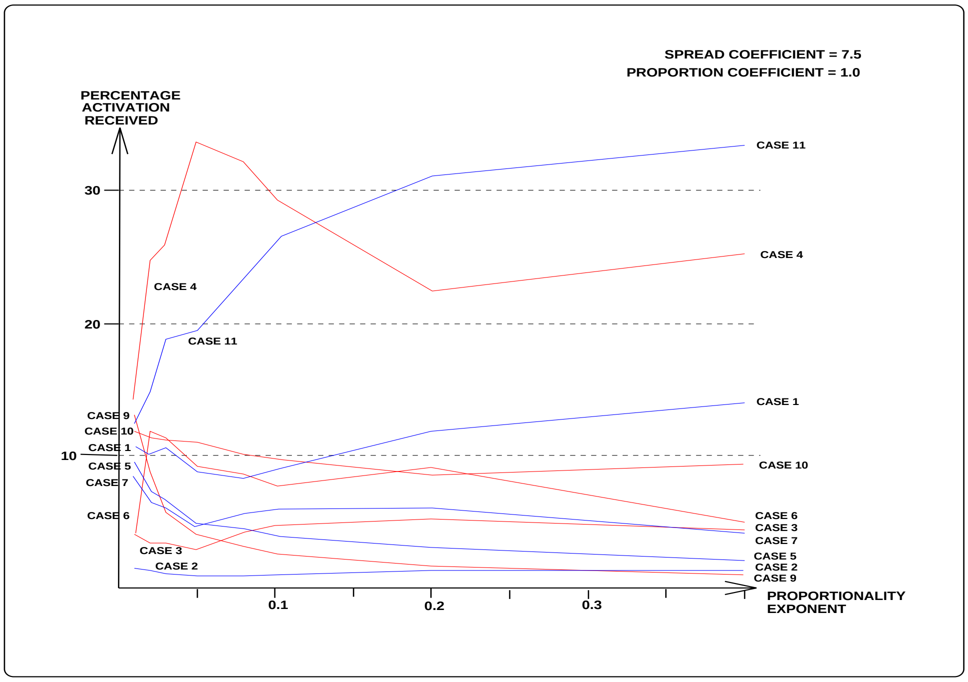


Figure O.1: The Effect on Case Ordering of Varying the Sensitivity of Upward Activation to Missing Details

Further evidence of this coupling lies in the similar morphology of all three curves of figure O.2.

An important point to note about the graph of figure O.2 is that there is relatively little variation in the order deviation with variation in γ. Using as a metric the σ_σ measure of expected variation in order deviation defined in section N.5, the variation across the experimental range for γ with respect to each bench-mark factor is as follows: semantic similarity $\simeq 1.1\sigma_\sigma$; structural similarity $\simeq 0.7\sigma_\sigma$; heuristic-based ordering $\simeq 0.55\sigma_\sigma$. These results suggest that the performance of CRASH is robust with respect to the exact setting of γ. However, some variation does persist and an optimal performance for retrieval is observed at an intermediate value for γ. This is explained below.

As γ tends to 0 from the optimal value there is an increase in the extent to which the overall retrieval is influenced by the retrieval of isolated source case nodes during phase 2. Effectively, the robustness of retrieval to coincidental feature pairings is diminished. This, in turn, produces a qualitative degradation in the result of the retrieval with respect to all three bench-marks.

As γ increases beyond its optimal value then missing details have an increasing importance in phase 3. Phase 2 of this retrieval is non-exhaustive and missing details are inevitable. It follows that an over-sensitivity to missing details will tend to cause any retrieval of surface-feature nodes during phase 2 to be subsequently ignored in phase 3.

[2]Not shown in the graph are; case 13 and case 8 which both received a small amount of activation and were always ordered 11th and 12th respectively. In this experiment case 12 received no activation.

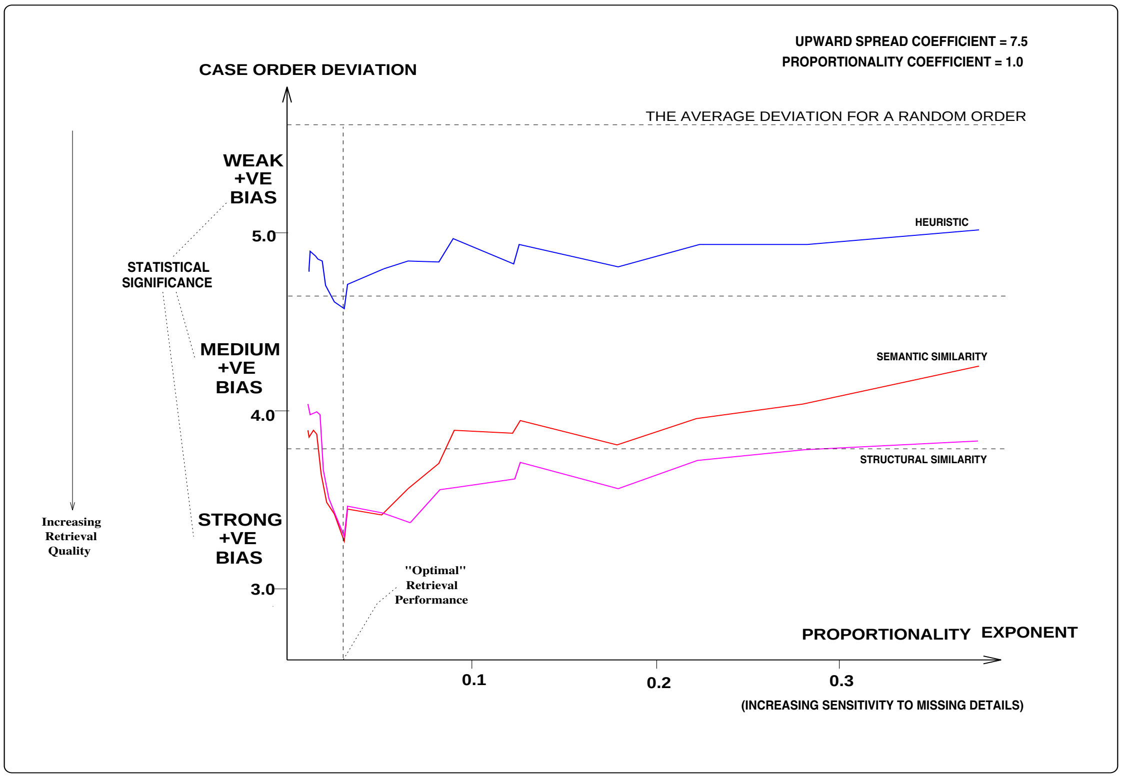


Figure O.2: The Effect of Sensitivity to Missing Details on the Quality of Case Retrieval

This is demonstrated by the increase in the relative influence of structural similarity on retrieval as γ increases. The fact that ignoring low-level features degrades the quality of retrieval is evidence that the surface level features of the cases in this case study are indeed of importance with respect to solving the bin-packing problem.

Experiment 1.2: Varying The Upward Spread Coefficient

For this experiment the following settings were made; $N_a = 5.0$ and $\gamma = 0.1$. Varying η affects the emphasis given to lower level case components during phase 3 of retrieval; as η increases the amount of activation from lower level components that is passed upwards is amplified. Note that the effect of η is more coarse grained than for γ; the sensitivity to missing details is generally localised to the consideration of a single colour dependency rule, whereas upward activation amplification applies throughout phase 3 of retrieval (see section 4.2.2 for details). In this experiment, the range of η is varied from 1.0 through to 15.0. The results are shown in figure O.3[3] and figure O.4.

The results for experiment 1.1 are similar to those of experiment 1.2. Most importantly, the performance of retrieval is again seen to be robust; for the experimental range of η that is used, the curves of figure O.4 vary as follows: semantic similarity $\simeq 1.0\sigma_\sigma$; structural similarity $\simeq 0.8\sigma_\sigma$; heuristic-based ordering $\simeq 0.8\sigma_\sigma$.

[3]Activated cases 13 and 8, along with non-activated case 12, are not shown.

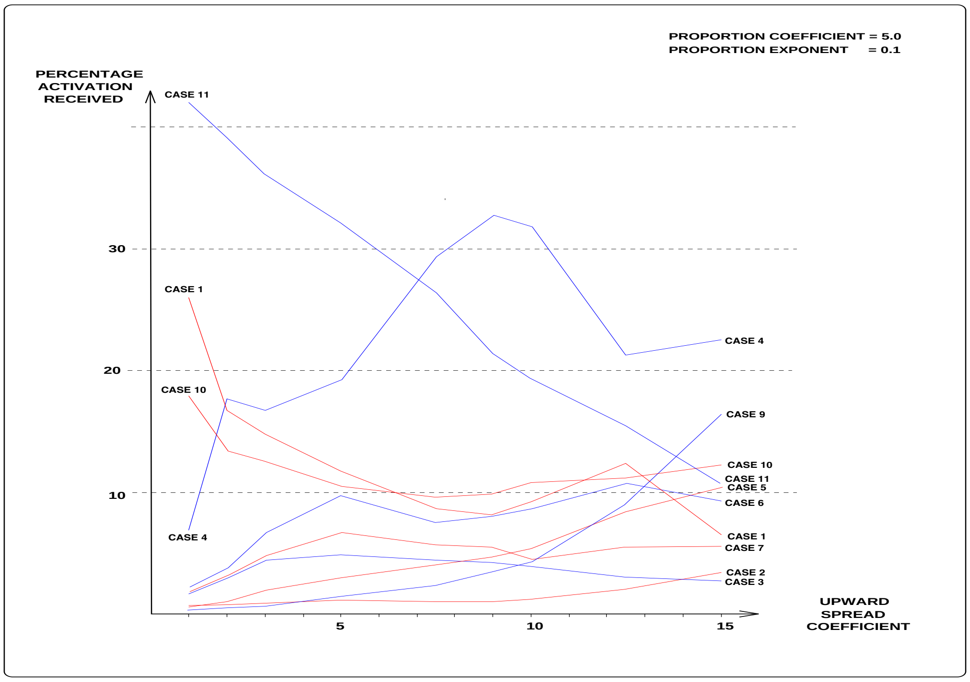


Figure O.3: The Effect On Case Ordering of Varying the Amplification of Upward Activation Passage

Nevertheless, there exists an intermediate value of η which produces optimal retrieval performance with respect to each bench-mark. As η tends to 0 surface-level features tend to be ignored. As η tends to high values so phase 3 becomes overly sensitive to isolated, surface-level phase 2 retrievals. It follows that the reasons for there being an optimal intermediate value of η are the same as for γ of experiment 1.1.

O.2 Experiment 2

Experiment 2 is primarily concerned with an investigation of the potential of CRASH to "scale up" for use on large case bases. For this purpose the relationship between retrieval effort and resultant retrieval quality is investigated. In addition, the performance of CRASH across a range of three sizes of case base (see section M.1) is also investigated. Finally, the experiment investigates the usefulness of CRASH, not only in terms of the overall grading of source cases (as is measured by order deviation) but also in terms of selecting a small corpus of source cases that are most appropriate.

O.2.1 Experiment 2: Description

This section provides a description of the experimental procedure for experiment 2. Of particular interest in the experiment is whether the memory model implemented in CRASH performs better than a linear relationship between quality and effort. It is

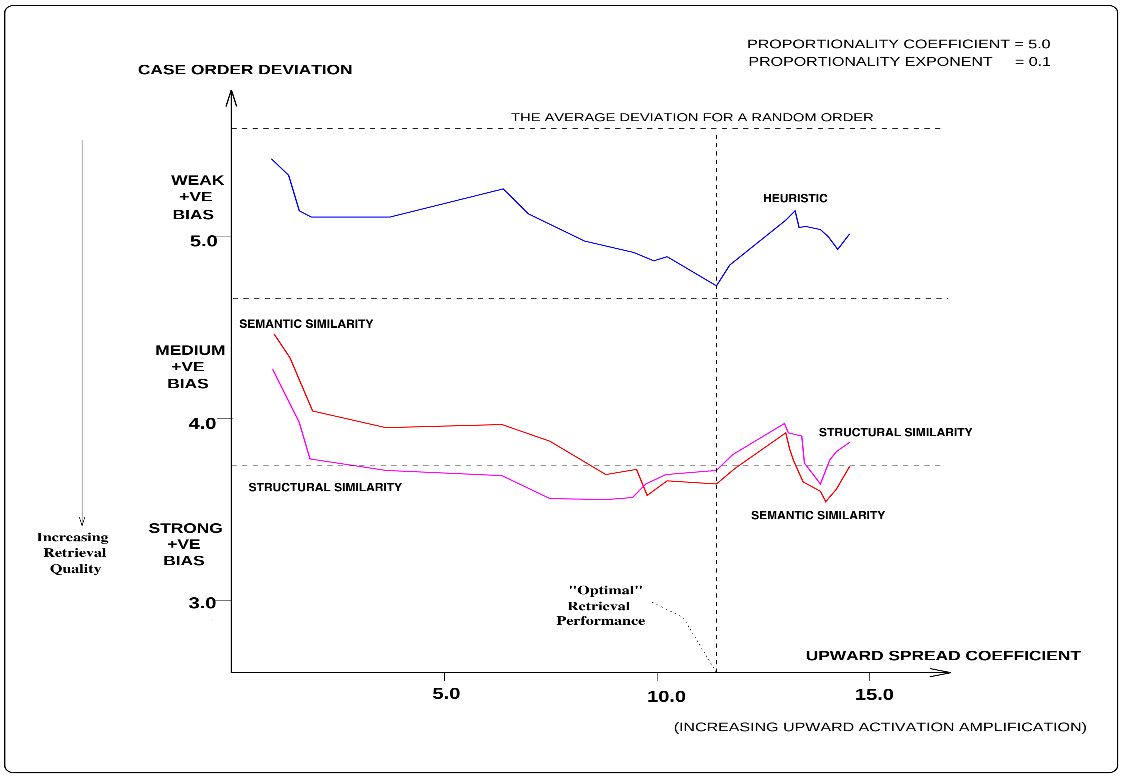


Figure O.4: The Effect of the Upward Spread Coefficient on the Quality of Case Retrieval

therefore first necessary to define what is meant by linearity in this sense; this is the focus of the following section.

A Comparable Generic Sorting Algorithm

Ideally, the retrieval mechanism of CRASH should act like a sorting algorithm, generating a list of source cases ordered with respect to the constraints on retrieval. A problem arises because, given that quality is to be measured in terms of order deviation, it is not easy to quantify what *a unit quality improvement per unit effort* actually means. The approach taken is to consider the performance of a generic sorting algorithm. The sorting algorithm chosen is a general purpose version of "bubble-sort", as depicted in figure O.5.

There are two important points about this sort algorithm. Firstly, it has a well defined criteria for the amount of effort expended; namely the number of individual element swaps so far performed. Secondly, bubble-sort is by definition an 'in-place' sort algorithm, i.e. at each step in the algorithm the complete, partially ordered list is fully defined. It follows that, for an implementation of the general bubble-sort algorithm, it is possible to record the order deviation and the amount of effort so far expended at each sorting step. At the end of the computation the total amount of effort expended can be used to 'normalise' each recorded amount of effort. The result is a curve depicting quality of retrieval (as measured by order deviation) as a function of the *relative* effort expended.

GENERIC BUBBLE-SORT

```
WHILE (not_fully_sorted(List))
    Select at random a pair of wrongly ordered and adjacent
    list elements

    Swap the ordering of the selected list pair.
```

Figure O.5: The Generic "Bubble-Sort' Algorithm

An example, based on the sorting of all 22 source cases used in the experimentation, is shown in figure O.6.

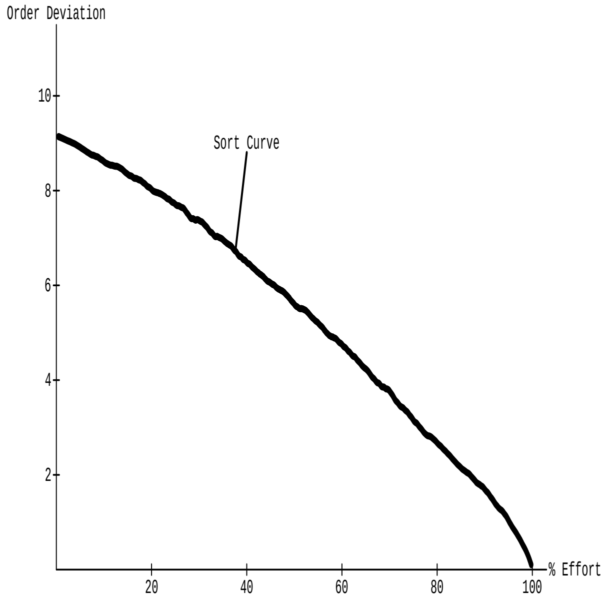


Figure O.6: An Example of the Order-Deviation versus Percentage Effort Curve for the "Bubble-Sort" Algorithm

From the graph of figure O.6 it can be seen that the quality (measured by order deviation) versus percentage effort relationship approximates to a straight line. In other words, for a naive sort algorithm, such as bubble-sort, there is (at best) a linear rate of reduction in order deviation with increasing retrieval effort.

Note that the *absolute* amount of effort performed is dependent on the type of sort algorithm used. For example, in the case of bubble-sort, the total amount of effort that may be expended is $O(n^2)$ complexity, where n is the number of elements in the list to be sorted. However, the linearity between order deviation and *percentage* of maximum effort applies to any sort algorithm that traverses its own search space in a *"naive"* (i.e.

unguided) way. Let a *superlinear* sorting algorithm be defined as one for which the partially sorted list, at some intermediate percentage of maximum effort, has a quality level with respect to its ordering that is always greater than would be achieved for a linear relationship between quality and effort. The general distinction between superlinear performance and linear (or worse) performance, with respect to quality measured by order deviation, is shown in figure O.7.

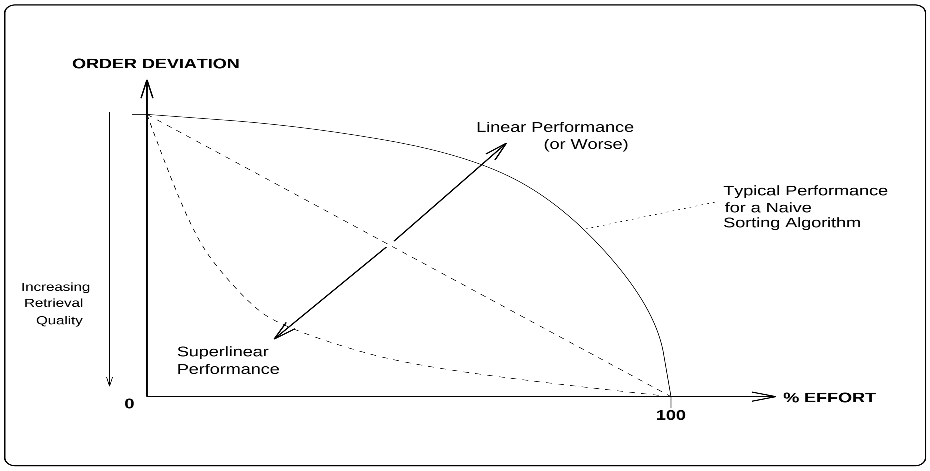


Figure O.7: A General Interpretation of Quality Versus Effort from the Perspective of Order Deviation

Experiment 2: Control of Phase 2 of Retrieval

It is during phase 2 of CRASH's retrieval that search through the space of memory is performed. The exhaustiveness of this search is controlled by the branching constant, B_c, as described on page 68. Control of retrieval search is administered in this experiment through a systematic variation of the value of B_c.

The context sequence used to perform phase 2 retrieval throughout experiment 2 is the same as that detailed in section O.1.1. The fixed context sequence means that the retrieval search space can be clearly defined for each of the case bases. For each step of the predefined context sequence the number of possible single step activation propagations is limited by two factors: only a limited number of relations are selected as traversable by the current context and only a limited number of potential nodes (originally those of the target case description) can *possibly* have received activation up to the current point in the retrieval and hence be senders. The following values were calculated for the retrieval search space sizes for the three case bases used in this experiment:

- $CaseBase_8$: Search Space Size = 2406 single step activation propagations.
- $CaseBase_{13}$: Search Space Size = 3727 single step activation propagations.
- $CaseBase_{22}$: Search Space Size = 7054 single step activation propagations.

These values represent the work done for an exhaustive retrieval. They are used to convert the actual number of single step activation propagations performed for each non-exhaustive retrieval into a percentage of maximal retrieval effort, hence allowing a meaningful comparison to the relationship depicted in figure O.7.

An interesting point to note is that, from the search space sizes determined above, there appears to be an approximately linear dependency between the number of cases in memory and the retrieval search space size. Specifically, the general relationship is assumed to be:

$$Size_{Space} = kN_c^q \tag{O.3}$$

where N_c is the number of cases in memory. The above figures suggest the following possible range of values of k and q with respect to the bin packing cases: $k \approx 310$, $0.97 < q < 1.02$.

Experiment 2: Activation Propagation in Phase 3

The parameters for phase 3 of CRASH's retrieval are set as constants for this experiment. The specific settings that were found to give good results are; $N_a = 5.0$, $\gamma = 0.1$ and $\eta = 7.5$.

O.2.2 Experiment 2: Details of Results

The three sizes of case base and three bench-marks in conjunction provide nine possible degrees of freedom to be tested in this experiment. The results for varying the extent of retrieval search effort with respect to each of these dimensions is pictorially presented in the graphs of appendix P. The general trend that can be extracted from these plots is shown in figure O.8. Several key features can be identified and are explained below:

- **Memory Bias.** Ideally, as the amount of retrieval effort increases, the deviation from a specified bench-mark ordering should tend towards zero. In practise, the ordering of cases produced by exhaustive retrieval will not correspond exactly to a specific bench-mark ordering; a maximum threshold for the degree of correspondence exists and this is what is referred to as the memory bias.

  Two factors explain why the memory bias for order deviation is greater than zero. Firstly, because the goal of CRASH's memory retrieval is to satisfy a number of differing constraints, the performance should be a compromise which does not favour one particular criterion. Secondly, the bench-mark criteria are defined subjectively with respect to the memory model of CRASH and hence it is to be expected that, without extensive fine-tuning of CRASH, they only approximately map to the actual behaviour produced by the activation passing mechanism of CRASH.

  For the plots of appendix P, the estimated value of the memory bias always lies in either the medium or high positive bias bands for order deviation. This is important as it constitutes further evidence of the robustness of CRASH's performance;

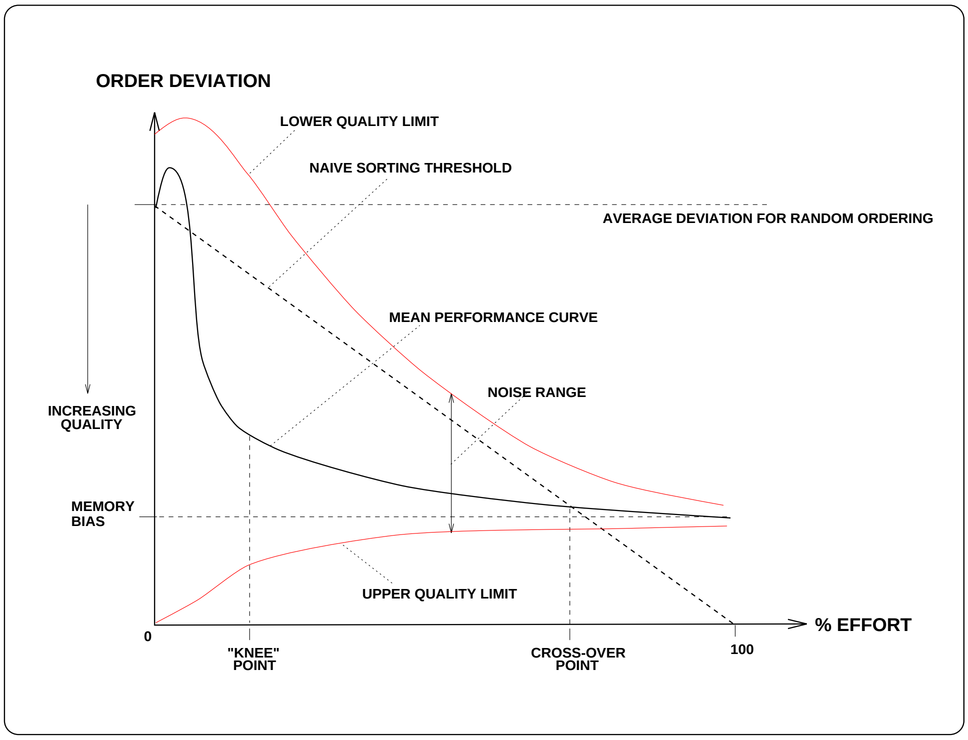


Figure O.8: The General Relationship Between Retrieval Quality (Order Deviation) and Retrieval Effort

CRASH appears capable of producing high quality results across a range of case base sizes.

- **Mean Performance Curve.** This curve represents the average order deviation that will be produced by CRASH for a given retrieval effort. From figure O.8 and the plots of appendix P, it can be seen that this curve tends to the memory bias level at a rate that generally decreases as retrieval effort increases. From the discussion of pages 232 to 234 this is sufficient to show that (at least for low-effort retrieval) CRASH's performance is superlinear. This result has important practical implications in that, for a given case base, a near optimal quality of retrieval can be achieved using only a fraction of exhaustive retrieval effort.

  That the mean performance curve for structural similarity and heuristic ordering bench-marks follows the same trend as for semantic similarity is indicative of the inter-relationship between the three bench-mark factors. This is explained below.

  If phase 3 of CRASH's retrieval is to make an accurate measure of the extent to which the structure between source and target cases over-lap, the source case should receive activation on a high proportion of its component nodes during phase 2. This is, in turn, governed by the semantic similarity between the two cases. For heuristic ordering this relationship may be less clear-cut. The results of experiment 1 suggest that features throughout a bin-packing case description are significant in governing the practical importance of that case. To a crude approximation, the more feature-to-feature paths forged in phase 2 of retrieval, the greater the bias towards heuristic ordering. However, some case features have more pragmatic relevance than others. If these highly pertinent features are not the most numerous then retrieval quality may actually degrade beyond a certain level of effort; evidence for this is presented in figure P.6 of appendix P. The general conclusion is that, in terms of heuristic ordering, the mean performance curve may involve a number of local maximums caused by a mismatch between the relative frequencies and importances of case features.

- **Cross-Over Point.** Because order-deviation with respect to a given bench-mark does not reach zero at maximum effort in CRASH, there will always be some level of effort above which a naive sorting algorithm will out perform CRASH; this threshold is referred to as the cross-over point. Moreover, that the mean performance curve of CRASH generally lies below the naive sorting threshold merely implies that CRASH offers a potential constant factor speed up at low-levels of retrieval effort. Whether a net gain in performance is achieved for CRASH with respect to some less guided sorting algorithm will depend also upon the practical factors associated with the implementation of each technique.

- **Noise Range.** The noise range represents the extent to which the quality produced by an individual CRASH retrieval can deviate from that of the mean performance curve for a given amount of retrieval effort. The noise range is a product of the in-built uncertainty that exists for the CRASH retrieval mechanism. The main source of this uncertainty is that, for low retrieval effort (i.e. a low value for B_c), only a subset of all possible receivers will be selected for each node of the network and each activation propagation step. In the absence of context or relation weighting guidance, the selection of the receivers is performed at random. For more extensive retrievals (i.e. high values of B_c), the consequences of this non-determinism will be less severe simply because less retrieval paths will be ignored. The implication is that, although the average quality of retrieval may improve little for high degrees of retrieval effort, the level of uncertainty for the quality of an individual retrieval will be reduced, as is illustrated in figure O.8.

- **Upper Quality Limit.** A threshold that indicates the maximal quality of retrieval that will be produced by a given amount of effort in CRASH.

- **Lower Quality Limit.** A threshold that indicates the minimum quality of retrieval that will be produced by a given amount of effort in CRASH.

- **"Knee" Point.** The shape of the mean performance curve indicates that a compromise point on the curve may be identifiable where the rate of quality improvement per unit effort is sufficiently low to render further effort non-cost-effective. This point is referred to here as the "knee" point.

	8 CASES	**13 CASES**	**22 CASES**
Ave. "Knee" Value	≈ 19.5 %	≈ 16.0 %	≈ 7.0 %

Table O.1: Variation in the "Knee" Point of the Mean Performance with Case Base Size

It is interesting to compare the relative positions of the "knee" point on the mean performance curve for each size of case base. Table O.1 presents the estimated "knee" points for the curves of appendix P[4]. Note that the "knee" point tends to decrease as the retrieval search space (measured in terms of the number of cases) increases. The most basic equation to explain this is:

$$KneePoint = \frac{l}{N_c^r} \tag{O.4}$$

where again N_c is the number of cases in memory and l and r are constants. From the figures of table O.1 the estimated value for the constant r lies in the range $0.57 < r < 2.5$. Now, the absolute amount of effort required to reach the quality indicated by the "knee" point is defined by combining equations O.3 and O.4. The result is:

$$AbsoluteEffort = mN_c^{(q-r)} \tag{O.5}$$

where m is a constant given by $m = k * l$. It follows that the upper bound on the complexity of $AbsoluteEffort$, as suggested by the preliminary experimentation in support of this thesis, is:

$$AbsoluteEffort \propto N_c^{0.45} \tag{O.6}$$

In other words, the absolute amount of effort required to achieve near maximal retrieval quality with CRASH appears, at worst, to be a slowly increasing function of case base size. A more accurate assessment of this relationship is an area for future empirical investigation using CRASH.

The measurement of quality of retrieval by order deviation is based on *all* source cases stored in CRASH's memory. The results achieved are encouraging in that they shows that CRASH is capable of performing a useful grading across a large number of cases during retrieval. However, for CRASH to be useful as a component of a more complete CBR system it may need to produce a small and focussed corpus of the best case in memory. There is, therefore, a need for CRASH's performance to be assessed with respect to just the most highly activated cases returned at the end of retrieval. Accordingly, the next section evaluates experiment 2 results with respect to the 3 most highly activated cases produced by each retrieval.

[4] For the graphs of appendix P the "knee" point of each mean performance curve is taken to be the first major peak in quality as effort increases from zero. In the more ideal case where the mean performance curve is noise-free, then the "knee" point must be defined by reference to the first derivative of the mean performance curve where a point can be identified at which the rate of change in quality per unit effort has reduced to some cut-off threshold.

Retrieval Results Based on the "Top 3" Cases.

For the assessment of the *"top 3"* cases, two factors must be measured:

- $Dist(\{case_1, case_2, case_3\}, S_{N_c})$: the sum of the absolute difference in position between the top three cases retrieved by CRASH and their respective positions in bench-mark ordering S_{N_c}, (N_c signifies the total number of source cases).

- $MaxDist(3, N_c)$: the maximum possible value of $Dist$ for 3 cases selected out of a case base of size N_c.

The *"top 3 quality measure"*, Q_3, can be defined as follows:

$$Q_3 = \frac{MaxDist(3, N_c) - 2Dist(\{case_1, case_2, case_3\}, S_{N_c})}{MaxDist(3, N_c)} \tag{O.7}$$

This produces a range of values for Q_3 such that $-1 \leq Q_3 \leq 1$. A value of $+1$ indicates that the correspondence between the top three cases of retrieval and the bench-mark is exact. A correspondence of -1 indicates that retrieval has selected the three worst cases in memory with respect to the bench-mark.

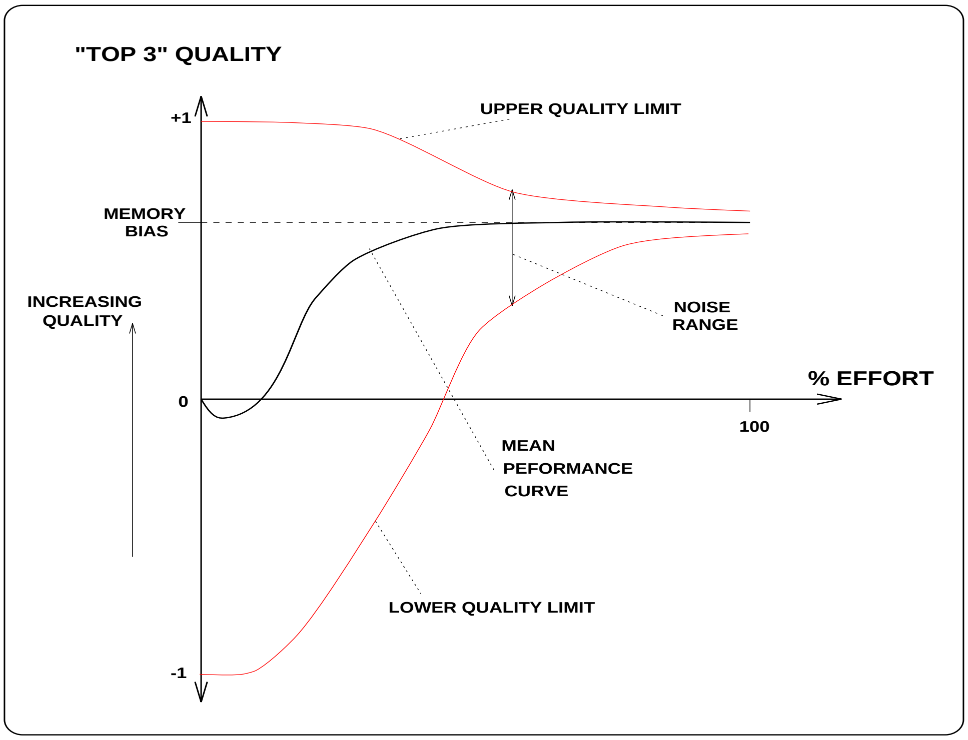


Figure O.9: The General Relationship Between Retrieval Quality (Top 3) and Retrieval Effort

The empirical results of experiment 2 are represented as plots of Q_3 versus retrieval effort in appendix P. The parity with the curves for order deviation is marked. Again, as shown in figure O.9, the mean performance curve tends to a maximal quality (i.e. the memory bias) at a rate that generally diminishes with increasing effort. Indeed, most of the above points made with respect to the order deviation curves apply to the curves for the top 3 cases. For example, the "knee" point of the top 3 mean performance curve, for all case base sizes, coincides with the same point on the order deviation mean performance curve (see appendix P). The conclusion that may be drawn is that the grading of cases performed by CRASH is robust to variation in the size of the sample of source cases taken from memory.

O.2.3 Experiment 2: Summary

This experiment has shown that the ability of CRASH to produce retrieval results biased towards all three bench-mark factors is robust across a range of case base sizes. Furthermore, CRASH's results are robust when judged both in terms of producing a global gradation of all source cases or in terms of selecting a small number of the most pertinent source cases. These results confirm that one of the central goals of this thesis has been attained; a memory model has been demonstrated that actively measures the similarity between cases as an inherent part of retrieval.

A general pattern has been observed in the relationship between retrieval effort and retrieval quality. It has been shown that the rate of quality improvement drops off with increased retrieval effort, hence suggesting that exhaustive search in a retrieval may not be cost effective. Conversely, for low retrieval effort CRASH's performance appears to be disproportionately high with respect to the performance that would be expected for a naive sorting algorithm. More importantly, there is suggestive evidence that the relative value for the "knee" point of the mean performance curves reduces as the case base size increases. This, in turn, suggests that the *absolute* amount of retrieval effort required to attain a specified quality of retrieval from CRASH increases more slowly than linearly with case base size. These results provide strong evidence for another central argument of this thesis; that a suitably structured memory can facilitate flexible and selective case retrieval without a need for exhaustive search of memory.

Appendix P

Experiment 2 Graphs

This chapter contains the set of graphs summarising the results for all retrievals performed using CRASH during experiment 2. Two graphs are presented for each of the case bases (i.e. $CaseBase_8$, $CaseBase_{13}$ and $CaseBase_{22}$ defined in section M.1) used in that experiment. The first represents a plot of retrieval quality versus retrieval effort with respect to the deviation of case order between the retrieval and each of the bench-marks defined in appendix N. That is to say, this graph measures the quality of retrieval with respect to all source cases in memory. The second graph gives a measure of retrieval quality based only on the top 3 case produced by a retrieval in comparison to the position of these cases in each bench-mark ordering.

P.1 A Note on Graph Presentation

Each graph presented in this appendix contains three curves, approximating to the *mean performance curve* for retrieval with respect to each of the specified bench-marks. This mean performance curve is derived by applying a Gaussian smoothing function to the scatter-plot of retrieval quality versus retrieval effort. An illustration of a typical scatter-plot and associated mean performance curve is given in figure P.1[1]

In addition to the mean performance curves, a line representing the

[1]The scatter-plot depicted in this figure is based on the top 3 cases retrieved from $CaseBase_{13}$ measured with respect to the semantic ordering bench-mark. Note that, as is typical of the graphs presented in this appendix, the mean performance curve summarises the results from approximately 50-80 separate retrievals.

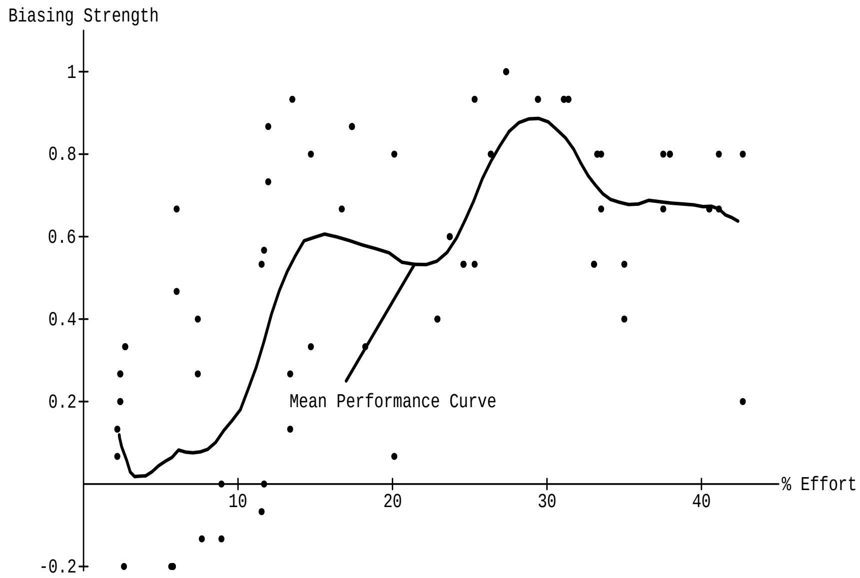


Figure P.1: An Example of the Mean Performance Curve of a Retrieval Quality vs Effort Scatter Plot

threshold of performance for naive sort is superimposed on each of the plots of quality measured as order deviation. According to the arguments of pages 232 to 234, where the mean performance curves lie below this line, CRASH's retrieval mechanism is producing superlinear results (i.e results that are better than would be anticipated for an unguided search of retrieval space).

Finally, the interpretation of the quality of retrieval, as depicted in the graphs, needs to be explained. Recall from section N.5 that the lower the order deviation, the stronger the bias of retrieval towards a bench-mark factor. To aid in the interpretation of those graphs that plot order deviation versus effort, the regions of statistical significance (as defined in section N.6) are included. For the graphs that represent the quality of retrieval with respect to just the top 3 retrieved cases, the quality of retrieval is a biasing strength that varies from $+1$ (perfect correspondence to the bench-mark) through to -1 (complete contradiction to the bench-mark). On average, this retrieval bias is 0 for a randomly generated sequence of cases compared to a bench-mark ordering.

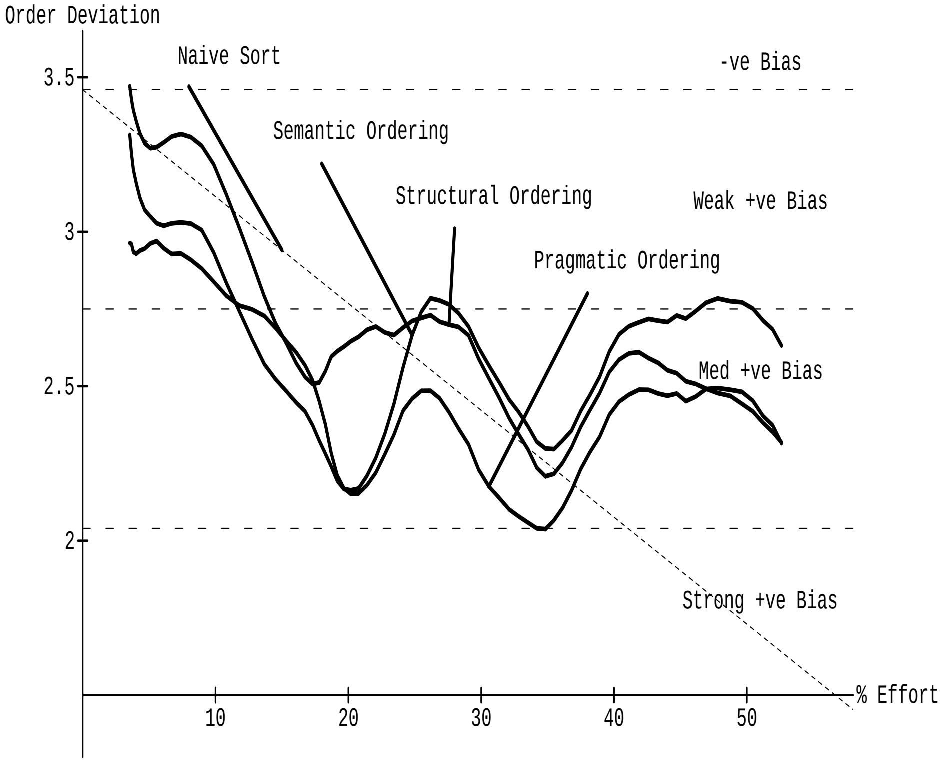


Figure P.2: Order Deviation vs Retrieval Effort for 8 Cases

P.2 Results Graphs

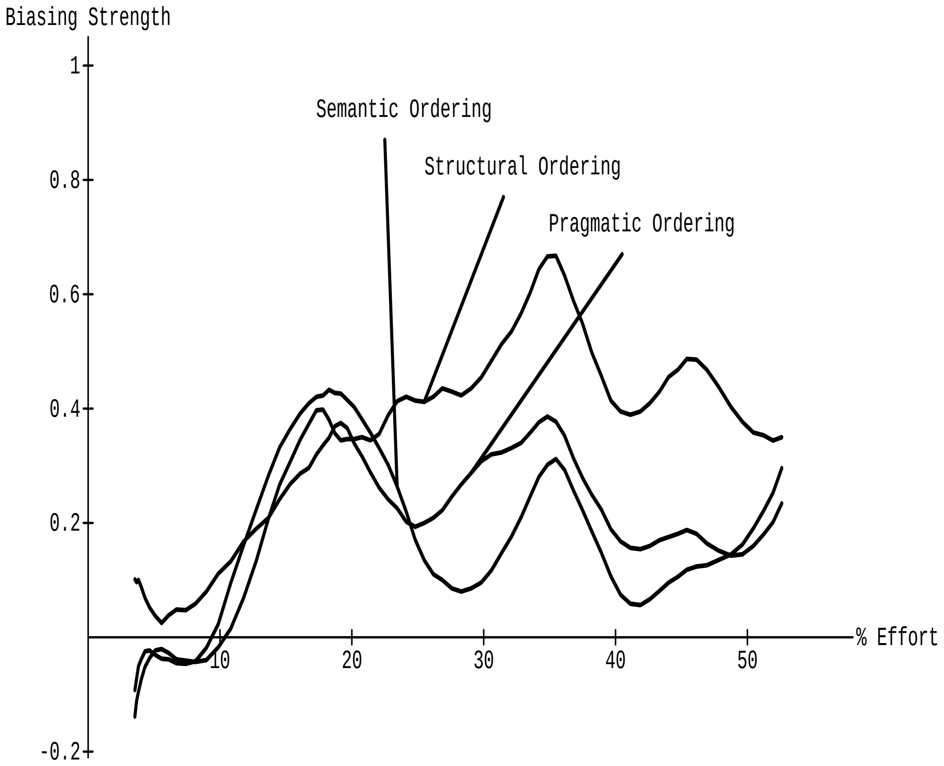


Figure P.3: Quality of Top 3 Retrieved Cases vs Retrieval Effort for 8 Cases

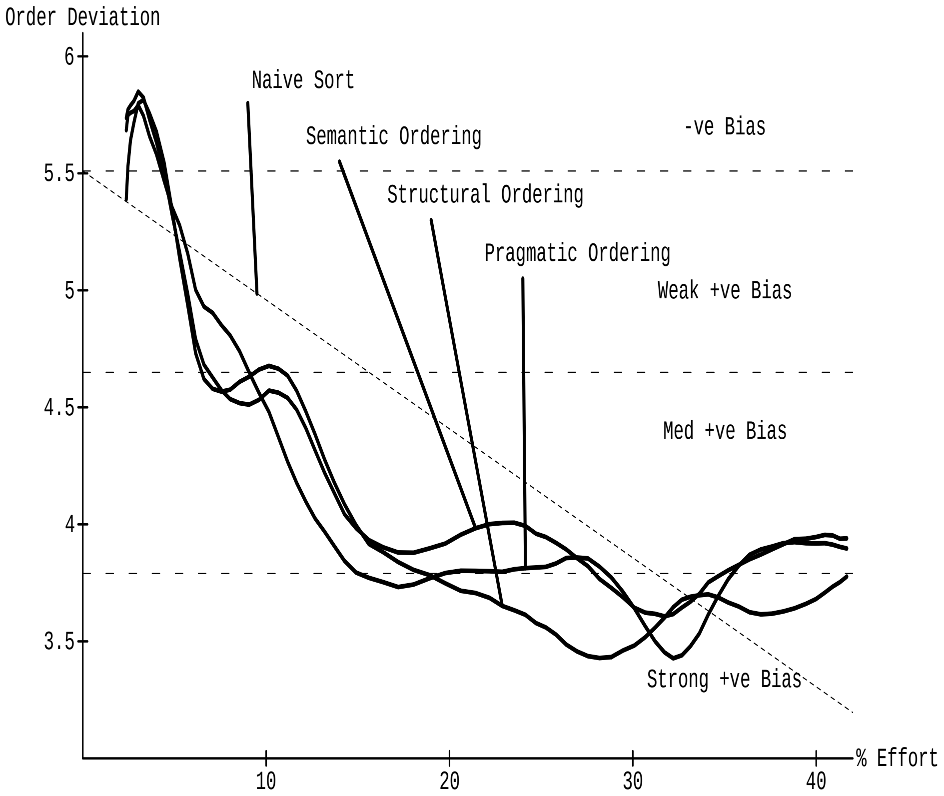


Figure P.4: Order Deviation vs Retrieval Effort for 13 Cases

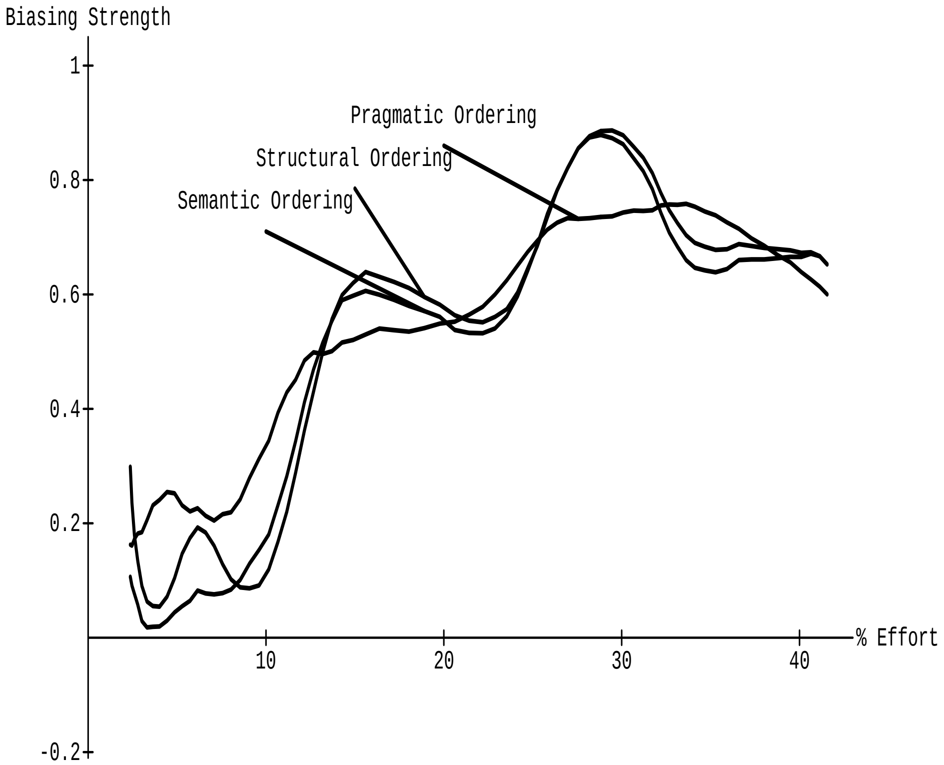


Figure P.5: Quality of Top 3 Retrieved Cases vs Retrieval Effort for 13 Cases

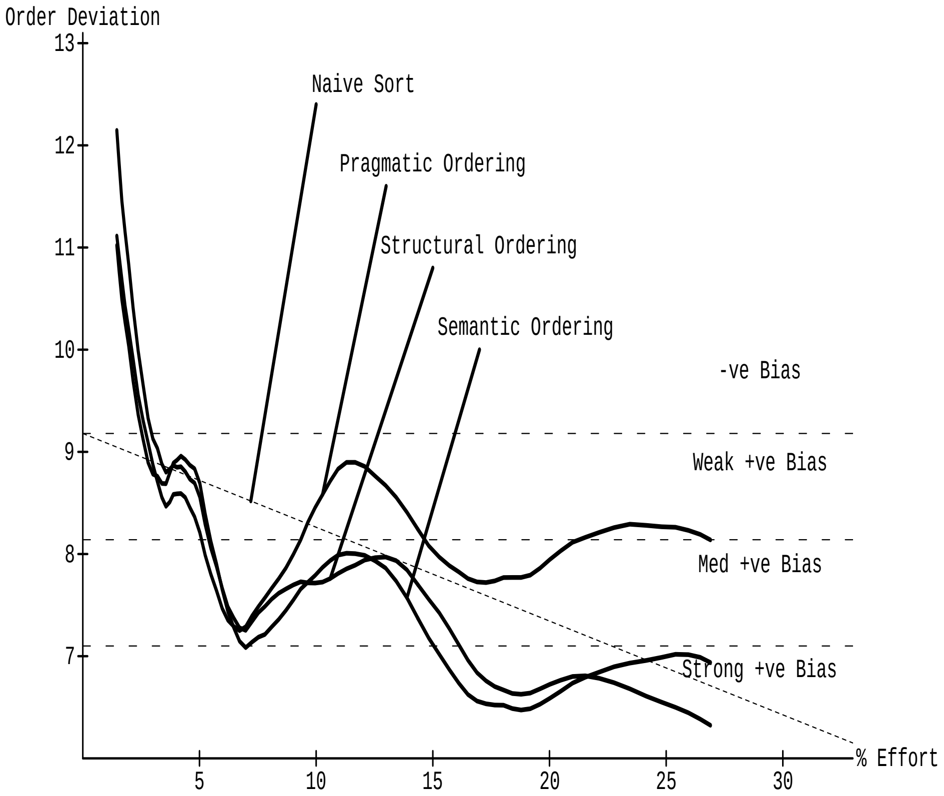


Figure P.6: Order Deviation vs Retrieval Effort for 22 Cases

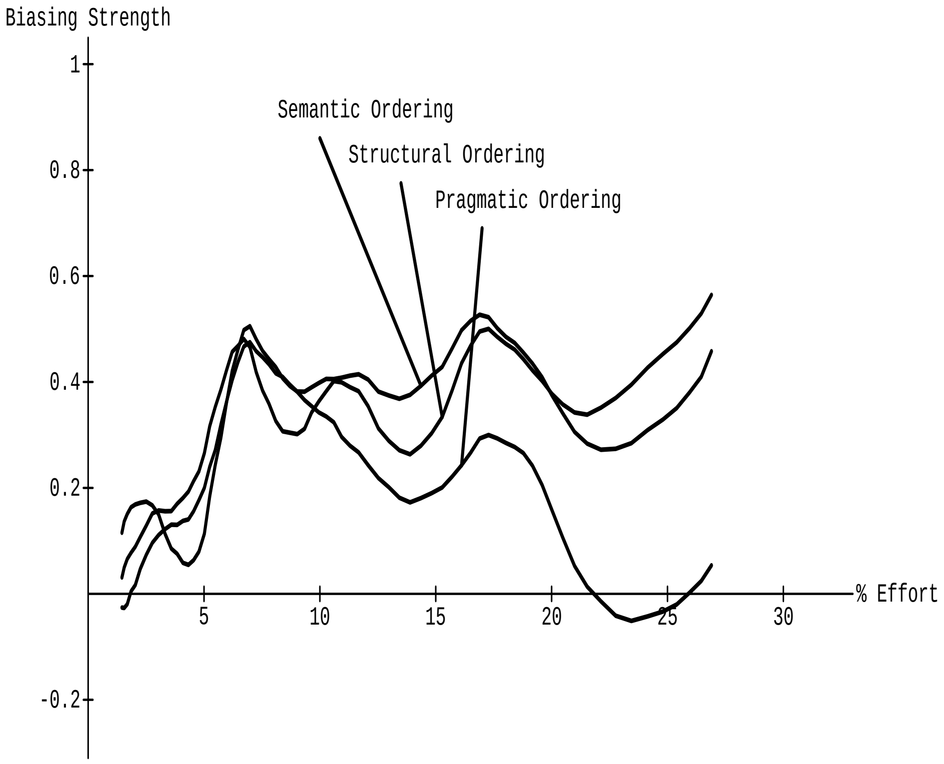


Figure P.7: Quality of Top 3 Retrieved Cases vs Retrieval Effort for 22 Cases