A Memory-based Theory of Verbal Cognition*

Simon Dennis
University of Colorado

The Syntagmatic Paradigmatic (SP) model is a distributed, memory-based account of verbal processing. Built on a Bayesian interpretation of string edit theory, it characterizes the control of verbal cognition as the retrieval of sets of syntagmatic and paradigmatic constraints from sequential and relational long-term memory and the resolution of these constraints in working memory. Lexical information is extracted directly from text using a version of the Expectation Maximization (EM) algorithm. In this paper, the model is described and then illustrated on a number of phenomena including sentence processing, semantic categorization and rating, short term serial recall and analogical and logical inference. Subsequently, the model is applied to a large scale corpus and used to extract syntactic structure and to assign syntactic representations to polysemous words within sentential context.

* This research was supported by Australian Research Council grant A00106012, US National Science Foundation grant EIA-0121201 and the US Department of Education grant R305G020027. I would like to acknowledge the many discussions that have shaped the current work. In particular, I would like to thank Michael Harrington, Michael Humphreys, Peter Kwantes, Andrew Smith, Walter Kintsch, Tom Landauer, Jose Quesada and Michael Litman for their helpful comments and suggestions.

Table of Contents

TABLE OF CONTENTS **2**

FIGURES **3**

TABLES **4**

INTRODUCTION **5**

DESCRIPTION OF THE SYNTAGMATIC PARADIGMATIC MODEL **7**

MATHEMATICAL FOUNDATIONS **12**

SEQUENTIAL MEMORY 12
The Bayesian Formulation *15*
Calculating the Memory Retrieval Component *15*
Calculating the Substitution Component *17*
Gap Probabilities *17*
LEXICAL MEMORY 18
The Generative Model *19*
The A Priori Lexical Model *19*
Updating Lexical Memory with Corpus Statistics *20*
RELATIONAL MEMORY 21
Calculating the Memory Retrieval Component *24*
Calculating the Substitution Component *26*
COMBINING SEQUENTIAL AND RELATIONAL SUBSTITUTION PROBABILITIES 26

SENTENCE PROCESSING **27**

LONG DISTANCE DEPENDENCIES 27
STRUCTURE SENSITIVITY 31
GENERATIVITY 33
SURFACE STRUCTURE INDEPENDENCE 35
SYSTEMATICITY 37
ONLINE PROCESSING: GARDEN PATH SENTENCES 39

SEMANTIC MEMORY: CATEGORIZATION AND RATING **42**

CATEGORIZATION 43
RATING 45

SHORT TERM MEMORY: SERIAL RECALL **48**

THE SERIAL POSITION CURVE 48
INTRA-LIST INTRUSIONS 51
INTER-LIST INTRUSIONS 52
SERIAL POSITION WITH GROUPED LISTS 52

INFERENCE **55**

AUTOMATIC INFERENCE 55
CONTROLLED/LOGICAL INFERENCE 57
CONTROLLED/ANALOGICAL INFERENCE 59

APPLYING THE MODEL TO NATURAL CORPORA **62**

INDUCING SYNTACTIC STRUCTURE 62
THE REPRESENTATION OF WORDS IN SENTENTIAL CONTEXT 69

DISCUSSION **71**

CONCLUSIONS **72**

REFERENCES **74**

APPENDIX A: NUMERICAL EXAMPLE OF THE SEQUENTIAL MEMORY RETRIEVAL COMPONENT **78**

APPENDIX B: A NUMERICAL EXAMPLE OF THE SEQUENTIAL SUBSTITUTION COMPONENT **81**

APPENDIX C: THE DYNAMIC PROGRAMMING ALGORITHM **82**

A Numerical Example using the DP Algorithm *84*

APPENDIX D: THE DYNAMIC PROGRAMMING ALGORITHM WITH GAP PROBABILITIES **87**

APPENDIX E: DERIVATION OF THE EM EQUATIONS FOR THE SP MODEL **91**

APPENDIX F: A NUMERICAL EXAMPLE OF THE RELATIONAL MEMORY RETRIEVAL COMPONENT **96**

APPENDIX G: A NUMERICAL EXAMPLE OF THE RELATIONAL SUBSTITUTION COMPONENT **97**

APPENDIX H: PROBABILITIES OF ALIGNING VERBS AND NOUNS IN SENTENTIAL CONTEXT **98**

Figures

Figure 1: The SP Model architecture. At the start of processing the lexical items are used to retrieve from lexical memory. At this stage, the John slot is filled with Bert, Steve, Dave and Michael. However, only Bert and Steve are context relevant (based on this corpus), and so after sequential retrieval Dave and Michael are eliminated. 8

Figure 2: The relational trace for the sentence "John loves Mary". Note that the order of bindings is irrelevant. 21

Figure 3: Generating RT from $\mathbf{R_k}$ 22

Figure 4: Demonstration of long term dependencies (singular) 29

Figure 5: Demonstration of long term dependencies (plural) 30

Figure 6: The impact of gap length. "was" is the verb of appropriate plurality, while "were" is inappropriate. Note the following adjustment to the default parameters was necessary Gap Factor = 7 31

Figure 7: Demonstration of structure sensitivity 33

Figure 8: Demonstration of generativity. Note the following adjustments to the default parameters were necessary $P(< x, x >| \overline{S_k \mapsto T}) = 0.025$ and $P(< x, y >| \overline{S_k \mapsto T}) = 0.2$ 35

Figure 9: Demonstration of surface form independence 36

Figure 10: Demonstration of systematicity 38

Figure 11: Demonstration of the garden path effect. 40

Figure 12: Demonstration of the non garden path control sentence 41

Figure 13: Working memory and relational memory following relational resolution in the semantic categorization task. The model correctly classifies tigers as carnivores and hamsters and elephants as herbivores. 44

Figure 14: Working memory and relational memory following relational resolution when the model is asked to determine the ferocity of tigers, hamsters and camels. 45

Figure 15: Working memory and relational memory following relational resolution when the model is asked to determine the size of tigers, hamsters and camels. 46

Figure 16: Working memory following relational retrieval in the serial recall task. 49

Figure 17: The serial position curve of the SP model. 50

Figure 18: The working memory representation at study..... 50
Figure 19: Intra list intrusions in the Serial Recall task. The graph shows the probability of producing an item from a given position in the study list as a function of output position... 51
Figure 20: Inter list intrusions in the serial recall task. The graph shows the probability of producing an item from a given position in any of the previous lists in a given output position. 52
Figure 21: Serial position curve with grouped items..... 53
Figure 22: Implicit Inference: Relational Representations of “Charlie bought the lemonade from Lucy .” and “Lucy sold the lemonade to Charlie .”..... 56
Figure 23: Explicit Inference: Working memory following sequential resolution. Note the following adjustment to the default parameters was necessary $P(< x, x >| \overline{S_k \mapsto T}) = 0.25$ 57
Figure 24: Explicit inference: Working memory following relational resolution. 58
Figure 25: Working Memory following Syntactic Resolution in the Analogical Inference Task. 60
Figure 26: Relational Memory in the Analogical Inference task. 60
Figure 27: Working Memory following Relational Resolution in the Analogical Inference Task 60
Figure 28: Hierarchical cluster analysis of the substitution probabilities of the 75 most frequent non-anchor words from the TASA corpus. N=Noun, V=Verb, A=Adjective, B=Adverb, P=Preposition. 65
Figure 29: Hierarchical cluster analysis of the LSA cosines of the 75 most frequent non-anchor words from the TASA corpus. N=Noun, V=Verb, A=Adjective, B=Adverb, P=Preposition. 66

Tables

Table 1: Possible alignments for “Bert loves Ellen” and “John loves Mary”. B=Bert, L=loves, E=Ellen, J=John, M=Mary, - is the empty word. 14
Table 2: Default values for edit type probabilities. 19
Table 3: Substitution examples. 63
Table 4: WordNet categories..... 68
Table 5: Possible alignments for “Bert loves Ellen” and “John loves Mary” and the probabilities of these alignments. B=Bert, L=loves, E=Ellen, J=John, M=Mary, - is the empty word. The alignments shaded in the bottom right hand corner are those that contain a change between “John” and “Bert”..... 79
Table 6: Matrix of partial alignments for “Bert loves Ellen” and “John loves Mary”. B=Bert, L=loves, E=Ellen, J=John, M=Mary. The dashed line represents alignment A8 (see text). . 83
Table 7: The probability of alignments that pass through the substitution <loves,loves> is the probability of the partial match of “John” and “Bert” by the probability of <loves,loves>by the probability of the partial match of “Mary” and “Ellen”. 84
Table 8: The matrix M for the example..... 84
Table 9: The matrix M’ for the example. 85
Table 10: The probabilities of all alignments traversing each partial match state. 85

Introduction

People can understand sentences of many structural forms, including many that we have never seen before, and realize equivalences between these forms (e.g. active and passive sentences). On the basis of accumulated factual knowledge we can categorize stimuli into natural kinds and make rapid and often reliable ratings on the properties of items for which we have no direct experience. We are able to retrieve information about what happened when and can make inferences remarkably easily on occasion.

What all of these phenomena have in common is that they are examples of verbal processing. Whether one is completing a categorization task, a memory retrieval task or an analogical inference task, one is interpreting and/or generating linguistic content. While it is possible that separate mechanisms underlie each of these phenomena it is more parsimonious to assume that each of these tasks provides a particular perspective on a single verbal processing mechanism. In this paper, the Syntagmatic Paradigmatic model (SP) is introduced as a candidate for such a mechanism.

In order to construct an account of verbal cognition one must address three fundamental issues - how is content represented within the system, what is the nature of the control architecture and how are content and control acquired. Existing cross domain theories tend to have complementary strengths and weaknesses. Models such as SOAR (Newell 1990) and ACT-R (Anderson & Lebiere 1998) propose that symbols form the representational medium[1] and that control is achieved via a set of if-then production rules. Typically, these models have excelled in specifying how humans perform complex cognitive tasks[2] but have struggled to provide a convincing account of the acquisition of either content or control. While both models have learning mechanisms they have not proven sufficiently powerful to operate autonomously and, in practice, when moving to a new task domain it is often necessary for the theorist to supplement with new representational items and task-specific production rules.

On the other hand, statistical models such as Latent Semantic Analysis (LSA, Landauer & Dumais 1997), Hyperspace Analog to Language (HAL, Lund & Burgess 1996) and the many connectionist approaches to verbal cognition (Elman 1990, Christiansen & Chater 1999) propose that representation is subsymbolic and have had a great deal of success in acquiring representational structure directly from exposure to corpora. For instance, LSA is capable of completing the vocabulary component of the Test of English as a Foreign Language (TOEFL) with close to student accuracy. However, when these models are applied, task specific algorithms are proposed and as yet no general account of the cognitive control architecture or how it is acquired has emerged.

The SP model is designed to bridge the gap between these alternate approaches. It is built upon two basic assumptions. Firstly, it presumes that processing an utterance/sentence is

[1] Although the latest version of ACT-R includes a subsymbolic layer, see Anderson & Lebiere (1998).

[2] including many tasks that are beyond the scope of verbal cognition.

fundamentally a memory retrieval process[3]. Completing a verbal task involves taking a representation of the current verbal context (which includes task instructions) and matching this against a large collection of linguistic traces. Under this view, the control of verbal cognition is achieved through an inner dialog derived directly from retrieval cycles and without any notion of executive direction.

Secondly, the SP model assumes that linguistic traces consist of collections of *syntagmatic* and *paradigmatic* associations between words. Syntagmatic associations are thought to exist between words that often occur together, as in "run" and "fast". By contrast, paradigmatic associations exist between words that may not appear together but can appear in the same sentence context, such as "run" and "walk" (Ervin-Tripp 1970). At the heart of the model are three kinds of long-term memory traces: lexical, sequential and relational, each of which is defined in terms of syntagmatic and paradigmatic associations. A lexical trace is defined as the paradigmatic associates of a word across sentential contexts. A sequential trace is the set of syntagmatic associations within a sentential context. And a relational trace is the set of paradigmatic associations within a sentential context. As we will see in subsequent sections, defining traces in this way allows for easy training from corpora while also dealing with a number of issues that have prevented other empirical approaches from providing adequate accounts of human linguistic processing.

In the following sections, the SP model is described and the mathematical foundations of the model including String Edit Theory (SET) and the Expectation Maximization (EM) training algorithm are outlined. Then its application is illustrated in a number of domains including sentence processing, categorization, short term memory and analogical and logical inference using a series of small examples. Finally, the model is used to extract syntactic structure from a large scale corpus. The ultimate test of a model of this kind is its ability to provide insight across domains. The discussion focuses on this issue pointing out a number of areas in which a single assumption of the model drives phenomena associated with very different tasks.

[3] The SP model was inspired by the LEX model of single word reading by Kwantes and Mewhort (1999). The LEX model demonstrates how phenomena that may seem to require elaborate control processes can emerge from large memory-based systems.

Description of the Syntagmatic Paradigmatic Model

Figure 1 depicts the SP model as it would appear when exposed to the following corpus:

1. John loves Mary
2. Bert loves Ellen
3. Steve loves Jody
4. Who does Bert love ? Ellen
5. Who does Steve love ? Jody
6. When the loud music started John left
7. When the race started Dave left
8. When the lecture started Michael left

As outlined above, the SP model consists of three long-term memory systems, lexical, sequential and relational each of which is defined in terms of syntagmatic and paradigmatic associations[4].

Lexical memory consists of a trace for each word comprised of the paradigmatic associates of that word across the corpus. In the example, the lexical trace for John is {Bert, Steve, Dave, Michael} because each of these words substitutes for "John" in a sentential context. For "Bert" and "Steve" the paradigmatic associations derive from the simple active constructions in sentences two and three, while for "Dave" and "Michael" the associations derive from the more complicated clause initial constructions in sentences seven and eight. In the lexical trace, however, these associations are accumulated regardless of their origin.

Sequential memory consists of a trace for each sentence comprised of the syntagmatic associations embodied by that sentence. In the example, the sequential trace for the sentence "John loves Mary" is the string of words, "John", "loves", and "Mary", in order.

Relational memory consists of a trace for each sentence comprised of the paradigmatic associations embodied by that sentence. In the example, the relational trace for "John loves Mary" would be {John: Bert, Steve; Mary: Ellen, Jody}. Note that although the lexical and relational traces both contain paradigmatic associations, the lexical trace is accumulated over the entire corpus for an individual word (e.g. in this corpus *John* is bound to the distributed pattern containing *Bert*, *Steve*, *Dave* and *Michael*), while the relational trace is a binding of the paradigmatic associations of each of the words in a given sentence, so that within the relational trace for "John loves Mary", *John* is bound only to *Bert* and *Steve* (not *Dave* or *Michael*).

[4] Evidence for the existence of syntagmatic and paradigmatic associations comes from studies in the early 1960s that observed a shift in underlying lexical organisation from syntagmatic to paradigmatic in the production of free associates and in word usage tests. This shift was evident both as a function of development (Brown & Berko 1960, Ervin 1961) and of training (McNeill 1966).

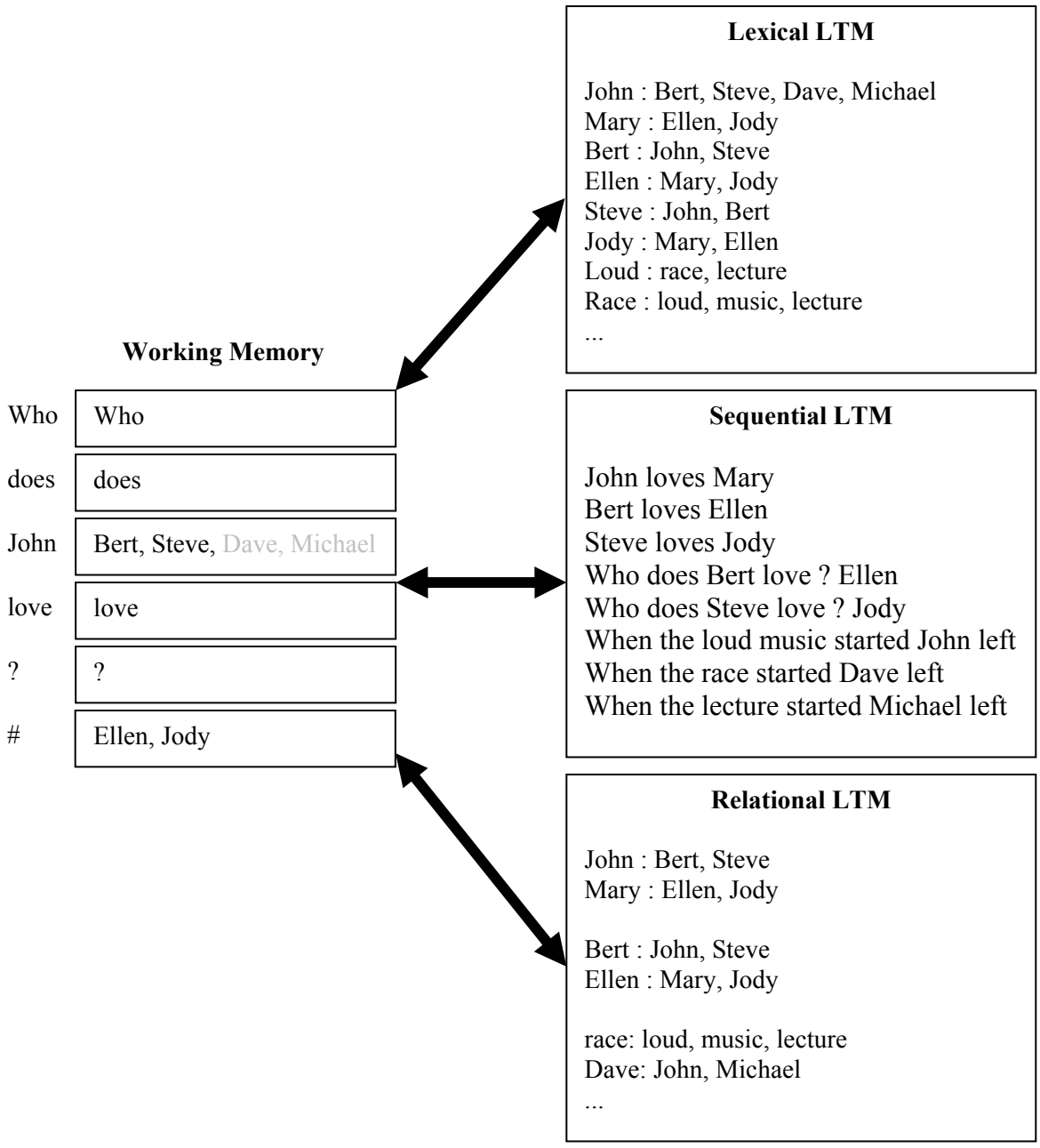

Figure 1: The SP Model architecture. At the start of processing the lexical items are used to retrieve from lexical memory. At this stage, the John slot is filled with Bert, Steve, Dave and Michael. However, only Bert and Steve are context relevant (based on this corpus), and so after sequential retrieval Dave and Michael are eliminated.

Note also that the set containing *Mary*, *Ellen*, and *Jody* can be thought of as a representation of the "lovee" role and the set containing *John*, *Bert* and *Steve* as the "lover" role, so the trace is an extraction of the relational information contained in the sentence. That is, the relational trace captures a form of deep structure (see the section Surface Structure Independence for further development of this point).

In the SP model, sentence processing is characterized as the retrieval of associative constraints from long-term memory followed by the resolution of these constraints in working memory. Creating an interpretation of a sentence/utterance involves the following steps:

Lexical Retrieval: During interpretation, as each word of an input sentence becomes available, its physical form (either phonology or orthography) is used to retrieve its representation in lexical memory, that is, its distributed pattern of paradigmatic associates accumulated across the training corpus. This pattern is inserted in the working memory slot adjacent to the input word. When presented with the question "Who does john love ? #" working memory would contain the following patterns:

who: who (0.039)
does: does (0.039)
john: john (0.024) dave (0.005) michael (0.005) bert (0.002) steve (0.002)
love: love (0.039)
?: ? (0.039)
#: # (0.080)

The "#" symbol indicates an empty slot. Ultimately, it will contain the answer to the question. The numbers in brackets are probabilities associated with the words immediately preceding them. In subsequent sections, we will discuss how these probabilities are calculated. Note that at this stage the representation has not been influenced by the sentential context, so the pattern will contain both contextually relevant information as well as information that may eventually be excluded by the context. In this example, the slot of John is initially filled with Bert, Steve, Dave and Michael.

Sequential Retrieval: Next, the current sequence of input words is used to probe sequential memory for traces containing similar sequences of words. In the example, traces four and five; "Who does Bert love ? Ellen" and "Who does Steve love ? Jody"; are the closest matches and are assigned high probabilities:

0.499 who does bert love ? ellen
0.499 who does steve love ? jody
0.001 john loves mary
0.000 bert loves ellen
0.000 steve loves jody
0.000 when the loud music started john left
0.000 when the race started dave left
0.000 when the lecture started michael left

In this simple example, the retrieved traces contain many of the same words in the same order and consequently are the best retrieval candidates. In general, however, lexical traces are used to establish structural similarity even in the absence of lexical overlap.

Sequential Resolution: The retrieved sequences are then aligned with the target sentence to determine the appropriate set of paradigmatic associates for each word. At this stage, sentential context will affect the trace words that are aligned with each of the input words:

who: who (0.997)
does: does (0.997)
john: steve (0.478) bert (0.478)
love: love (0.998)
?: ? (0.998)
#: jody (0.460) ellen (0.460)

Note that in the "John" slot the context relevant words "Bert" and "Steve" remain while "Dave" and "Michael" are suppressed as they do not appear in sentences that are similar to the target sentence. That is, the model has disambiguated the role that John plays in this sentence. Note also that the slot adjacent to the "#" symbol contains the pattern {Jody, Ellen}. This pattern represents the role that the answer to the question must fill (i.e. the answer is the lovee).

Relational Retrieval: The bindings of input words to the corresponding sets of paradigmatic associates (the relational representation of the target sentence) are then used to probe relational long-term memory. In this case, trace one is favoured as it involves the same role filler bindings as the target sentence. That is, it contains a binding of John onto the {Steve, Bert} pattern and it also contains the {Jody, Ellen} pattern.

0.687 john: bert (0.298) steve (0.298)
mary: ellen (0.307) jody (0.307)
0.089 bert: steve (0.319) john (0.226)
ellen: jody (0.320) mary (0.235)
0.089 steve: bert (0.319) john (0.226)
jody: ellen (0.320) mary (0.235)
0.060 bert: steve (0.319) john (0.226)
ellen: ellen (0.439) jody (0.320) mary (0.235)
0.060 steve: bert (0.319) john (0.226)
jody: ellen (0.320) mary (0.235)
0.005 race: lecture (0.330) loud (0.163) music (0.162)
dave: michael (0.329) john (0.328)
0.005 lecture: lecture (0.331) race (0.330) loud (0.163) music (0.162)
michael: 0.005 dave (0.329) john (0.328)
0.005 loud: race (0.163) lecture (0.163)
music: race (0.162) lecture (0.162)
john: dave (0.328) michael (0.328)

Despite the fact that "John loves Mary" has a different surface form than "Who does John love ? #" it contains similar relational information and consequently has a high retrieval probability.

Relational Resolution: Finally, the paradigmatic associations in the retrieved relational traces are used to update working memory:

who: who (0.997)
does: does (0.997)
john: john (0.500) steve (0.488) bert (0.488)[5]
love: love (0.998) loves (0.153)
?: ? (0.998)
#: mary (0.558) ellen (0.523) jody (0.523)

In the relational trace for "John loves Mary", "Mary" is bound to the {Ellen, Jody} pattern. Consequently, there is a strong probability that "Mary" should align with the "#" symbol which as a consequence of sequential retrieval is also aligned with the {Ellen, Jody} pattern. Note that the model has now answered the question - it was Mary who was loved by John.

To summarize, the model hypothesizes five basic steps. Firstly, word representations from the perceptual systems are used to retrieve lexical information. Next the series of words is used to retrieve traces that are similar from sequential long term memory. Then, the retrieved sequential traces are aligned with the input sentence to create a relational interpretation of the sentence based on the word order. This interpretation is then used to retrieve similar traces from relational long term memory. Finally, working memory is updated to reflect the paradigmatic constraints retrieved in the previous step. This process allows known facts to influence sentence processing and underpins how the model incorporates pragmatic and factual knowledge.

In a number of circumstances, it is necessary for the model to be able to distinguish between traces that were stored in the current context from those that are part of the background memory of the system. Rather than propose a separate memory system to store the recent traces, the SP model assumes that these traces are more available because they contain a representation of the current context. During retrieval they are favoured while the same context is in effect[6]. The content and control of context is poorly understood (Dennis & Humphreys 2001). Rather than try to provide explicit context processing mechanisms, the model simply uses a symbol (CC, C1, C2, …) to represent the appropriate context and otherwise treats these symbols as if they were words. When a given retrieval probe shares context with traces in memory the same context symbol is

[5] Note that at this stage the substitution strengths are generated by taking the logical OR of the information sources. More sophisticated mechanisms are possible that would result in a proper probability distribution (that sums to one), but this approach is sufficient for the demonstrations presented in this paper.

[6] Note this style of mechanism bears some similarity to the notion of long term working memory proposed by Ericcson and Kintsch (1995). However, the emphasis in the LTWM proposal is more on the retrieval structures that underpin expert performance and text comprehension, whereas the current context in the SP model refers exclusively to the pointer mechanism that isolates elements of the current episode.

used in each. Circumstances where this occurs include when two sentences occur following each other in a given experimental trial, when the model attempts to recall items from the previous list in a serial recall experiment and when the bindings made in the first half of a sentence are required in the second half of the sentence. This treatment of context is somewhat arbitrary, but is used here in lieu of a more comprehensive mechanism.

The model as outlined above makes extensive use of the distinction between syntagmatic and paradigmatic associations. A question arises, however, as to what we mean by these associations, in particular how do we determine when two words fill the same slot in different sentences. The next section outlines the mathematical foundations of the model including how syntagmatic and paradigmatic associates are extracted and used.

Mathematical Foundations

Sequential Memory

The SP model relies on the extraction of syntagmatic and paradigmatic associations from a corpus representing the relevant verbal experience of the system. Sequential retrieval is the process of determining which traces from the corpus are sequentially similar (i.e. contain similar syntagmatic associations) and sequential resolution is the process of determining how each of these traces should align with the current target sentence.

When similar sentences are of the same length they can be aligned in a one to one fashion. If we are comparing “John loves Mary” and “Bert loves Ellen”, then “John” aligns with “Bert”, by virtue of the fact that “John” and “Bert” both fill the first slot of their respective sentences, and “Mary” aligns with “Ellen” because they both fill slot three.

However, in general this will not be the case. It is typical in natural languages for structure to be embedded. If we add a single adjective to our example, so that we are now comparing “Little John loves Mary” and “Bert loves Ellen” it becomes unclear how the sentences should align. What we require is a model of the alignment process that defines what we mean by a paradigmatic slot.

One candidate for such a model is string edit theory. String edit theory was popularized in a book by Sankoff and Kruskall (1983) entitled, “Time warps, string edits and macromolecules” and has been developed in both the fields of computer science and molecular biology (Levenshtein 1965; Needleman & Wunsch 1970, Sellers 1974, Allison, Wallace & Yee 1992). As the name suggests, the purpose of string edit theory is to describe how one string, which could be composed of words, letters, amino acids etc., can be edited to form a second string. That is, what components must be inserted, deleted or changed to turn one string into another.

As an example, suppose we are trying to align the sentences “John loves Mary” and “Bert loves Ellen”. The most obvious alignment is that which maps the two sentences to each other in a one to one fashion as suggested above:

```
John   loves   Mary                                    A1
|      |       |
Bert   loves   Ellen
```

In this alignment, we have three edit operations. There is a **change** of “John” for “Bert”, a **match** of “loves” and a **change** of “Mary” for “Ellen”. In fact, this alignment can also be expressed as a sequence of edit operations:

<John, Bert>
<loves, loves>
<Mary, Ellen>

Now if we add “Little” to the first sentence, we can use a **delete** to describe one way in which the sentences could be aligned:

```
Little  John   loves   Mary                             A2
|       |      |       |
-       Bert   loves   Ellen
```

The “-“ symbol is used to fill the slot left by a deletion (or an insertion) and can be thought of as the empty word. The corresponding edit operation is denoted by <John, ->. While these alignments may be the most obvious ones, there are many other options.

For instance, in aligning “John loves Mary” and “Bert loves Ellen”, we could start by deleting “John”:

```
John   loves   Mary   -                                A3
|      |       |      |
-      Bert    loves  Ellen
```

Note that “Ellen” is now inserted at the end of the alignment (denoted <-, Ellen>). Alternatively, we could have deleted “John”, and then inserted “Bert” to give:

```
John   -       loves   Mary                             A4
|      |       |       |
-      Bert    loves   Ellen
```

Table 1 lists the 63 possible ways in which "John loves Mary" can be aligned with "Bert loves Ellen".

Intuitively, alignment A4 seems better than A3 because the word "loves" is matched. However, this alignment still seems worse than A1 because it requires "John" to be deleted and "Bert" to be inserted. A mechanism that produces alignments of sentences should favor those that have many matches and should penalize those that require many insertions and deletions.

Table 1: Possible alignments for "Bert loves Ellen" and "John loves Mary". B=Bert, L=loves, E=Ellen, J=John, M=Mary, - is the empty word.

---JLM BLE---	-J-LM B-L-E	-JL-M BL-E-	J-L-M- -B-L-E	JL-M- --BLE	J-LM- BL--E
--J-LM BL-E--	-J-LM B-LE-	-JLM- BL--E	J-L-M -B-LE	JLM--- ---BLE	J-LM BL-E
--JL-M BL--E-	-JL--M B--LE-	-JLM BL-E	J-LM-- -B--LE	JLM-- --BLE	J-LM BLE-
--JLM- BL---E	-JL-M- B--L-E	-JLM BLE-	J-LM- -B-LE	JL--M -BLE-	JL--M B-LE-
--JLM BL--E	-JL-M B--LE	J---LM -BLE--	J-L-M -BLE-	JL-M- -BL-E	JL-M- B-L-E
--JLM BL-E-	-JLM-- B---LE	J--L-M -BL-E-	J-LM- -BL-E	JL-M -BLE	JL-M B-LE
--JLM BLE--	-JLM- B--LE	J--LM- -BL--E	J-LM -BLE	JLM-- -B-LE	JLM-- B--LE
-J--LM B-LE--	-JL-M B-LE-	J--LM -BL-E	JL---M --BLE-	JLM- -BLE	JLM- B-LE
-J-L-M B-L-E-	-JLM- B-L-E	J--LM -BLE-	JL--M- --BL-E	J--LM BLE--	JL-M BLE-
-J-LM- B-L--E	-JLM B-LE	J-L--M -B-LE-	JL--M --BLE	J-L-M BL-E-	JLM- BL-E
	-J-LM BLE--		JL-M-- --B-LE		JLM BLE

The purpose of applying string edit theory is to be able to calculate the probability of paradigmatic association, that is, how likely is it that "Bert" changes for "John" in the sentence "John loves Mary", for instance. In the following section, a Bayesian model of

the sequence retrieval and alignment processes is presented that allows the derivation of substitution probabilities.

The Bayesian Formulation

In order to apply probability theory to the alignment and retrieval process, one must posit a model of how the sentences that are presented as input have arisen (c.f. Allison, Wallace, Yee 1992)[7]. The SP model assumes that any target sentence that is to be interpreted, T, was generated by choosing a sequential trace from memory, S_k and applying a set of edit operations to this trace. What we are interested in calculating is $E_k\left[P(< W_m, T_i >| T)\right]$ the expected value of the probability that word T_i in slot i in the target sentence substitutes for the word W_m in the context of sentence T. Using the terminology $S_k \mapsto T$, to indicate that sequential trace S_k generated the target sentence T, we can write:

$$E_k\left[P(< W_m, T_i >| T)\right] = \sum_{k=1}^{N} \frac{P(S_k \mapsto T \mid T)}{\sum_{i=1}^{N} P(S_i \mapsto T \mid T)} P(< W_m, T_i >| S_k \mapsto T, T) \quad (1)$$

where N is the number of sequential traces in memory. Now we have divided the task into determining the probability that sequential trace k generated the target (the memory retrieval component) and determining the probability that T_i and S_{kj} align given that trace k did generate the target (the substitution component).

Calculating the Memory Retrieval Component

To calculate the memory retrieval component, we take a similar approach to that adopted by the Bayesian models of recognition memory (Shiffrin & Steyvers 1997, Dennis & Humphreys 2002, McClelland & Chappell 1998), which have proven very successful at capturing a variety of memory effects[8].

[7] Allison, Wallace & Yee (1992) have developed string edit theory using a minimum description length (MDL) paradigm rather than the purely Bayesian approach that has been adopted in this paper. The MDL approach has the advantage that it explicitly costs model complexity. For the most part, however, the approaches are similar in structure and the MDL component has been omitted to aid the clarity of the exposition.

[8] In particular, the Bayesian memory models provide a rationale for the ubiquitious mirror effects that are observed in recognition memory. That is, whenever a manipulation improves recognition performance the improvement occurs as a consequence of both an increase in the hit rate and a decrease in the false alarm rate.

We start with the odds ratio for $S_k \mapsto T$ given T and use Bayes theorem to convert to a likelihood ratio:

$$\frac{P(S_k \mapsto T \mid T)}{P(\overline{S_k \mapsto T} \mid T)} = \frac{P(T \mid S_k \mapsto T)P(S_k \mapsto T)}{P(T \mid \overline{S_k \mapsto T})P(\overline{S_k \mapsto T})} = \frac{P(T \mid S_k \mapsto T)}{P(T \mid \overline{S_k \mapsto T})}\gamma \tag{2}$$

where γ is the ratio of the priors which we will assume is one unless otherwise noted (i.e. assuming equal priors).

To calculate $P(T \mid S_k \mapsto T)$, we use the possible edit sequences that may have been employed to transform the trace into the target sentence as our data. Each edit sequence represents an exclusive hypothesis about how S_k generated T, so the probabilities of these hypotheses can be added to determine $P(T \mid S_k \mapsto T)$:

$$P(T \mid S_k \mapsto T) = \sum_{a_p \in A_k} P(a_p \mid S_k \mapsto T) \tag{3}$$

where a_p is one of the edit sequences relating T and S_k, and A_k is the set of these sequences.

Assuming that the edit operations (match, change, insert or delete) are sampled independently to create alignments[9]:

$$P(T \mid S_k \mapsto T) = \sum_{a_p \in A_k} \prod_{e_r \in a_p} P(e_r \mid S_k \mapsto T) \tag{4}$$

where e_r is the r-th edit operation in alignment a_p.

Similarly,

$$P(T \mid \overline{S_k \mapsto T}) = \sum_{a_p \in A_k} \prod_{e_r \in a_p} P(e_r \mid \overline{S_k \mapsto T}) \tag{5}$$

[9] This is certainly incorrect, but is used for simplicity. Allison, Powell and Dix (1999) provide an alternative.

So,

$$P(S_k \mapsto T \mid T) = \frac{\sum_{a_p \in A_k} \prod_{e_r \in a_p} P(e_r \mid S_k \mapsto T)}{\sum_{a_p \in A_k} \prod_{e_r \in a_p} P(e_r \mid S_k \mapsto T) + \sum_{a_p \in A_k} \prod_{e_r \in a_p} P(e_r \mid \overline{S_k \mapsto T})} \quad (6)$$

Appendix A provides a numeric worked example of the retrieval component.

Calculating the Substitution Component

Deriving the substitution component is straightforward, now that we have defined how alignments and edit operations are related. Since a given edit operation is either in an alignment or not we can just add the probabilities of the alignments in which this change occurs and normalize by the probability of all of the alignments:

$$\begin{aligned} P(< W_m, T_i > \mid S_k \mapsto T, T) &= \sum_{W_m = S_{kj}} P(< S_{kj}, T_i > \mid S_k \mapsto T, T) \\ &= \sum_{W_m = S_{kj}} \frac{\sum_{a_p \in A_k} P(a_p \mid S_k \mapsto T)}{\sum_{a_p \in A_k} P(a_p \mid S_k \mapsto T)} \end{aligned} \quad (7)$$

Appendix B provides a numeric worked example of the substitution component. We now have an algorithm with which we can calculate the probabilities of paradigmatic associations within sentential context. However, as currently stated it is computationally expensive. The number of alignments grows exponentially with the size of the target and trace sentences. Given that it may be necessary to make these calculations for many traces the amount of computation required could become prohibitive. Fortunately, there is a more efficient algorithm that can be employed to reduce the complexity to $O(|T||S_k|)$ time and space (see Appendix C).

Gap Probabilities

In the molecular biology literature, it is common to assign a lower probability to an initial deletion or insertion and then higher probabilities to subsequent indels in the same block. As a consequence, alignments that involve long sequences of indels are favored over alignments that have many short sequences (Waterman, Smith & Beyer 1976, Gotoh 1982, Waterman 1984). In the context of macromolecule alignment, increasing the probability of block indels is desirable because a single mutation event can often lead to a block insertion or deletion. An analogous argument is also applicable in the language

case, as it is common for entire phrases or clauses to differentiate otherwise structurally similar sentences.

To illustrate the point, consider aligning the sentences "John loves Mary" and "John who loves Ellen loves Mary". Two possible alignments are:

```
John   -     -     -     loves  Mary                    A5
|      |     |     |     |      |
John   who   loves Ellen  loves  Mary
```

and

```
John   -     loves  -     -      Mary                    A6
|      |     |      |     |      |
John   who   loves  Ellen  loves  Mary
```

Note that these alignments have the same matches and deletions and so will have the same probabilities as calculated above. However, A9 should be preferred over A10 as it involves the block deletion of a clause. To capture this property, it is assumed that the probability of an initial indel is lower than the probability of a continuing indel (the Gap Factor is a parameter representing the ratio of the continuing indel probability to the initial indel probability). Now A9 will be favoured as it involves a single start indel and two subsequent indels, whereas A10 has two start indels and one subsequent indel[10]. Again, there exists an algorithm that retains the $O(|T||S_k|)$ time and space complexity. Appendix D outlines a version of this algorithm modified for the current purposes. That completes the specification of the sequential memory component of the model.

Lexical Memory

As outlined above, the process of calculating substitution probabilities (i.e. paradigmatic associations) given a sentence context relies on substitution and indel probabilities. These comprise the lexical memory of the system. In this section, we outline a generative model that specifies how target sentences are derived. From this model we then determine how a priori estimates of the lexical probabilities can be calculated and derive a version of the Expectation Maximization algorithm that can be used to train the lexical model on a corpus.

[10] Allison, Wallace & Yee (1992) describe this process in terms of a three state finite state machine, and also generalize beyond the three-state case. In this paper, however, only the three-state case will be considered.

The Generative Model

The generative model for the sequential process consists of a combination of a mixture model and a string edit model. To account for a target sentence T we assume that a sequential trace from memory was selected and that a series of edit operations were then applied (the algorithm is presented in Appendix E). α_k denotes the probability of choosing a certain trace, while τ_y is the probability of each of the edit operation types (match, change, insert and delete). ϕ_{xx} will be used to denote the probability of word x being the subject of a match given that a match has occurred. ϕ_{xy} denotes the probability of word x substituting for word y given that a change has occurred. ϕ_{x-} denotes the probability that word x was deleted given that a deletion occurred and ϕ_{-x} denotes the probability that word x was inserted given that an insertion occurred.

Note that unlike the commonly employed string edit model, the string edit component of the SP model is asymmetric. While the trace to be used as a generator is selected from a finite set, the set of sentences that could be generated is infinite Also, note that the generative model when gap probabilities are considered is very similar. The primary difference is that the edit operation probabilities (τ_y) must also be conditioned on the preceding edit operation as outlined in the previous section.

The A Priori Lexical Model

In order to bootstrap the estimation of edit operation probabilities it is necessary to assign initial parameters. The generative model provides insight into how these parameters may be set. Firstly, the α_k can be thought of as priors on the probabilities of choosing each of the sequential traces from which to generate. The non-informative choice then is to assign $\alpha_k = 1/M$ where M is the number of traces in memory. In strings that are related, the probability of matches and changes should be high while insertions and deletions should be relatively uncommon. Initial values for τ_y should reflect these expectations and Table 2 outlines the default values that are used in the simulations reported in this paper.

Table 2: Default values for edit type probabilities.

	$P(e_r \mid S_k \mapsto T)$	$P(e_r \mid \overline{S_k \mapsto T})$
Match	0.9	0.35
Change	0.05	0.45
Insert	0.025	0.1
Delete	0.025	0.1

In the molecular biology string edit literature, type frequencies are sometimes used to calculate a priori models (Allison, Lee & Wallace 1992). However, in the context of

language strings this may be undesirable. In particular, function words which are often of very high frequency require high match probabilities but low change, insertion and deletion probabilities. Rather than use type frequencies we assume equal probabilities for each of the words.

So,

$$
\begin{aligned}
P(<x,->|\, S_k \mapsto T) &= \tau_{delete}\phi_{x-} &= \tau_{delete} / L \\
P(<-,x>|\, S_k \mapsto T) &= \tau_{insert}\phi_{-x} &= \tau_{insert} / W \\
P(<x,x>|\, S_k \mapsto T) &= \tau_{match}\phi_{xx} &= \tau_{match} / L \\
P(<x,y>|\, S_k \mapsto T) &= \tau_{change}\phi_{xy} &= \tau_{change} / L / W
\end{aligned}
$$

where L is the number of words in S_k and W is the total number of words in the lexicon. For computational convenience, we typically take a mean value of L rather than recalculating edit probabilities for each trace employed (by default this is set to 8.6).

Updating Lexical Memory with Corpus Statistics

The method for updating the edit probabilities is a version of the Expectation Maximization (EM) algorithm (Dempster, Laird & Rubin 1977). EM algorithms involve two steps. In the first step, the expected value of the log likelihood of the data given the current parameters is calculated. That is, we define Q:

$$Q(\theta, \theta^t) = \int_{y \in Y} \log(P(T, y \mid \theta) P(y \mid T, \theta^t) d\theta \tag{8}$$

where T is the set of sentences in the training corpus and Y is the set of all possible hidden variables (i.e. trace selections and edit sequences) that could have given rise to the set of traces. θ^t is the set of parameters (α, τ and ϕ) at the current step.

In the second step, we find the parameters θ which maximize Q. These will be used as the parameters for the next iteration of the algorithm:

$$\theta^{t+1} = \arg\max_{\theta} Q(\theta, \theta^t) \tag{9}$$

Repeated iterations of the EM algorithm are guaranteed to find a local minimum in the log likelihood (Dempster et. al. 1977)[11]. In the case of the SP model, the training algorithm reduces to adding the probabilities of the alignments in which each edit

[11] The algorithm presented in this paper looks for a maximum liklihood (ML) solution. There is reason to believe, however, that a maximum a posteriori (MAP) solution may be preferable. In particular, estimates of match probabilities for nonfunction words tend to be near zero with the ML approach, which is probably undesirable when generalizing to unseen sentences.

operation occurs and normalizing appropriately (see Appendix E for the complete derivation).

Relational Memory

For a given corpus each sequential trace can be used to generate a corresponding relational trace. The relational trace consists of the set of bindings of words in the target sentence to the corresponding vector of change probabilities for each word given the sentence.

For instance, in the previous example the relational trace corresponding to the sentence "John loves Mary" is shown in Figure 2.

Mary: Ellen (0.307) Jody (0.307)
John: Bert (0.298) Steve (0.298)

Figure 2: The relational trace for the sentence "John loves Mary". Note that the order of bindings is irrelevant.

Each relational trace can be thought of as a set of role filler bindings, where the role is given by the vector of change probabilities and the filler is the head word for each binding. However, the role vectors are generated purely by alignment of sentences in the corpus. There is no need to specify roles a priori. Also, because the roles are distributed representations it is possible for a single role vector to correspond to what would normally be considered roles at different functional levels. In the example above, the role vector {Bert (0.298), Steve (0.298)} is simultaneously a representation of a noun, an agent, a person, a lover and a male lover.

As in the sequential case, when interpreting a new target sentence we will assume that its relational trace (RT) has been generated via a set of edit operations on one of the relational traces in memory. Specifically, we assume that each binding in RT, which we will denote RT_i, was generated by either an insert or by taking one of the bindings in the relational trace (R_{kj}) and editing the head word and binding vector. For instance, suppose RT and R_{kj} are relational traces one and two from the previous example as shown in Figure 3.

RT

Mary: Ellen (0.307) Jody (0.307)
John: Bert (0.298) Steve (0.298)

R_k

Ellen: Jody (0.320) Mary (0.235)
Bert: Steve (0.319) John (0.226)

Edit Set 1

$$R_{k0} \mapsto RT_0 : Mary \Leftrightarrow Ellen; Ellen(0.307), Jody(0.307) \Leftrightarrow Jody(0.320), Mary(0.235)$$
$$R_{k1} \mapsto RT_1 : John \Leftrightarrow Bert; Bert(0.298), Steve(0.298) \Leftrightarrow Steve(0.319), John(0.226)$$

Edit Set 2

$$R_{k1} \mapsto RT_0 : Mary \Leftrightarrow Bert; Ellen(0.307), Jody(0.307) \Leftrightarrow Steve(0.319), John(0.226)$$
$$R_{k1} \mapsto RT_1 : John \Leftrightarrow Bert; Bert(0.298), Steve(0.298) \Leftrightarrow Steve(0.319), John(0.226)$$

Edit Set 3

$$R_{k0} \mapsto RT_0 : Mary \Leftrightarrow Ellen; Ellen(0.307), Jody(0.307) \Leftrightarrow Jody(0.320), Mary(0.235)$$
$$insert(RT_1) : insert(John); insert(Bert, 0.298), insert(Steve, 0.298)$$

Figure 3: Generating RT from $\mathbf{R_k}$

In the first edit set, RT has been generated from R_k by taking the first binding from R_k, substituting the head word Mary for Ellen and substituting {Ellen (0.307), Jody (0.307)} for {Jody (0.320), Mary (0.235)}, then taking the second binding from R_k, substituting the head word John for Bert, and substituting {Bert (0.298), Steve (0.298)} for {Steve (0.319), John (0.226)}.

While this may be the most obvious edit set, there are many other possibilities. In edit set 2, both RT_0 and RT_1 were generated from R_{k1}. This edit sequence seems less likely, however, as it involves the head word substitution of Mary for Bert which has a lower substitution probability than Mary and Ellen. Furthermore, the role vectors do not match as well.

Finally, edit set 3 demonstrates that a binding from the target sentence, in this case, RT_1 may not have appeared in the trace at all, in which case it is assumed to have been inserted spontaneously.

It might be argued that in order for a relational match to occur the head words of the relevant role-filler bindings should be required to be the same. So, in the example above we would say that there is no similarity between the relational representations of "John loves Mary" and "Bert loves Ellen" because neither the lovee or lover roles are bound to the same fillers. However, including the possibility of filler substitution improves the generalization ability of the model considerably. For instance, if the model has been presented with the sentence "John adores Mary" and is subsequently asked "Who does John love?" it will answer "Mary" even though the verbs are not identical. In this way, it avoids being overly literal.

While this form of generalization is likely to be desirable on the majority of occasions, there are circumstances under which it may lead to errors. One such situation, which is also difficult for humans, is the Moses illusion. Erickson and Mattson (1981) presented subjects with questions such as "How many animals of each kind did Moses take on the ark?" and asked them to detect any errors. It was, of course, Noah who took animals on the ark and so subjects should identify this as erroneous. On about 40% of occasions subjects will incorrectly endorse these distorted questions (and in the Moses question itself the figure is as high as 81%). Furthermore, even when participants' errors are explained and they are tested to ensure their knowledge of the correct facts they have significant difficulty overcoming the illusion (Reder & Kusbit 1991).

In the terms of the SP model, the illusion occurs because the probability of substitution of Moses and Noah is likely to be high given that they are both familiar biblical patriarchs and because this role is consistent with the sentential frame (see van Oostendorp & de Mul 1990, van Oostendorp & Kok 1990 for behavioural evidence that these are critical factors and Budiu 2001 for a similar model of the effect). Note that had the substitution occurred at the end of the sentence: "How many animals of each kind did Noah take on the boat?" then it would have been appropriate to respond "two". The cognitive system is likely to be optimized to respond appropriately in this situation rather than in the rare deceptive circumstances of the Moses illusion.

As in the sequential case, we wish to calculate the probability that a given word substitutes for T_i given the relational representation of the target (RT).

$$E_k\left[P(<W_m, T_i>|\, RT)\right] = \sum_{k=1}^{N} \frac{P(R_k \mapsto RT \mid RT)}{\sum_{i=1}^{N} P(R_i \mapsto RT \mid RT)} P(<W_m, T_i>|\, R_k \mapsto RT, RT) \tag{10}$$

So, again the problem is divided into a memory retrieval component and a substitution component.

Calculating the Memory Retrieval Component

Applying Bayes rule as we did in the sequential case we get:

$$\frac{P(R_k \mapsto RT \mid RT)}{P(\overline{R_k \mapsto RT} \mid RT)} = \frac{P(RT \mid R_k \mapsto T)P(R_k \mapsto RT)}{P(RT \mid \overline{R_k \mapsto T})P(\overline{R_k \mapsto RT})} = \frac{P(RT \mid R_k \mapsto RT)}{P(RT \mid \overline{R_k \mapsto RT})}\gamma \tag{11}$$

where γ is assumed to be one. So, we must now calculate $P(RT \mid R_k \mapsto RT)$. To do this recall that RT contains M bindings each of which are generated by independent operations:

$$P(RT \mid R_k \mapsto RT) = \prod_i^M P(RT_i \mid R_k \mapsto RT) \tag{12}$$

Furthermore, each binding in RT was generated from one of the bindings in R_k or by an insert. So,

$$P(RT_i \mid R_k \mapsto RT) = \sum_j^N P(RT_i, R_{kj} \mapsto RT_i \mid R_k \mapsto RT) + P(RT_i, insert(RT_i) \mid R_k \mapsto RT) \tag{13}$$

and

$$\begin{aligned} P(RT \mid R_k \mapsto RT) &= \prod_i^M P(RT_i \mid R_k \mapsto RT) \\ &= \prod_i^M \left[\sum_j^N P(RT_i, R_{kj} \mapsto RT_i \mid R_k \mapsto RT) + P(RT_i, insert(RT_i) \mid R_k \mapsto RT) \right] \end{aligned} \tag{14}$$

Note that if $R_{kj} \mapsto RT_i$ then
$R_k \mapsto RT$ so $P(RT_i \mid R_{kj} \mapsto RT_i, R_k \mapsto RT) = P(RT_i \mid R_{kj} \mapsto RT_i)$.
Also, $P(RT_i, insert(RT_i) \mid R_k \mapsto RT) = P(insert(RT_i) \mid R_k \mapsto RT)$.

Now $P(RT_i \mid R_{kj} \mapsto RT_i)$ is the probability that the head word of RT_i (T_i) substitutes for the head word of R_{kj} (S_{kj}) and that the vector of change probabilities of RT_i is an edited version of the R_{kj} vector. To determine the probability of head word substitution the prior substitution probability can be used $P(< S_{kj}, T_i > \mid S_k \mapsto T)$ [12]. To determine the probability

[12] Another possibility would be to learn relational substitution probabilities independently of sequential substitution probabilitites. So, two words would be considered interchangable if they tend to align with the

of vector substitution recall that each of the vectors is comprised of change probabilities. In each case only one of the words could have substituted for their respective head words, so we can multiply the probability of the trace word (R_{kjl}) by the probability of the target word (RT_{ip}) and the probability that the trace word would substitute for the target word ($P(< W_p, W_l >| S_k \mapsto T)$ to obtain an estimate of the probability that the role vector of R_{kj} was edited to produce the role vector of RT_i .

So,

$$P(RT_i \mid R_{kj} \mapsto RT_i) \quad = P(< S_{kj}, T_i >| S_k \mapsto T) \sum_p \sum_l RT_{ip} P(< W_p, W_l >| S_k \mapsto T) R_{kjl} \qquad (15)$$

where RT_{ip} is the p th component of the role vector of RT_i and R_{kjl} is the l th component of the role vector of R_{kj} . The $P(RT_i \mid \overline{R_{kj} \mapsto RT_i})$ is calculated in an analogous way using the $P(< S_{kj}, T_i >| \overline{S_k \mapsto T})$ and $P(< W_p, W_l >| \overline{S_k \mapsto T})$.

A similar logic is used to calculate the insertion probability:

$$P(insert(RT_i) \mid R_k \mapsto RT) = P(< -, T_i >| S_k \mapsto T) \sum_p P(< -, T_i >| S_k \mapsto T) RT_{ip} \qquad (16)$$

And finally the memory retrieval component is:

$$P(R_k \mapsto RT \mid RT) \quad = \frac{P(RT \mid R_k \mapsto T)}{P(RT \mid R_k \mapsto T) + P(RT \mid \overline{R_k \mapsto T})} \qquad (17)$$

The above algorithm has constant space complexity and time complexity $O(|T||S_k||W|^2)$ where $|W|$ is the size of the vocabulary. While in principle this is expensive, in practice the vocabulary component grows very slowly with $|W|$. Appendix F provides a numeric worked example.

---

same substitution vectors regardless of whether they are direct paradignmatic associates.

Calculating the Substitution Component

To calculate the probability of substitution we note that a substitution of T_i for S_{kj} has occurred whenever $R_{kj} \mapsto RT_i$. As a consequence the following derivation applies.

$$P(< W_m, T_i >| R_k \mapsto RT, RT) = \sum_{W_m = S_{kj}} P(< S_{kj}, T_i >| R_k \mapsto RT, RT) \quad (18)$$

$$= \sum_{W_m = S_{kj}} \frac{P(RT_i \mid R_{kj} \mapsto RT_i)}{\sum_j P(RT_i \mid R_{kj} \mapsto RT_i) + P(insert(RT_i \mid R_k \mapsto R}$$

Appendix G provides a numeric worked example.

Combining Sequential and Relational Substitution Probabilities

We now have two procedures by which we can generate estimates of the substitution probabilities of trace and target words – one based on the sequence of words in the target (sequential) and one based on retrieved role-filler bindings (relational). The final question is how the estimates based on these two different sources of information should be combined to arrive at a final set of substitution probabilities. Taking the logical OR of the information sources and assuming independence we get:

$$P(< W_m, T_i >) = P(< W_m, T_i >| T) + P(< W_m, T_i >| RT) - P(< W_m, T_i >| T)P(< W_m, T_i >| RT) \quad (19)$$

We have now completed the mathematical derivation of the model. In the following sections a number of small demonstrations are used to illustrate how the model addresses a range of verbal phenomena.

Sentence Processing

This section outlines how the SP model captures some important properties of sentence processing and how it addresses some key objections to associative approaches to language acquisition.

Long Distance Dependencies

An early critique of simple associative models of language focused on their inability to account for long distance dependencies among words and phrases (Fodor, Bever & Garrett, 1974). For instance, plurality information is often retained over many intervening words as in "The man who saw the altar and gave thanks was awed". Demonstrations using the Simple Recurrent Network (SRN, Elman 1993) have shown that associative models using a hidden layer can, in principle, capture these phenomena. However, in practice simulations have been restricted to limited vocabularies with small gaps and training rapidly becomes more difficult as size of the vocabulary or the length of the gap increases.

The SP model is capable of retaining dependency information over long gaps. Instead of relying on a hidden layer to encode the relevant information, the SP model directly represents the dependency in sequential traces and relies upon the sentential context to choose between stored traces containing the relevant associations.

To demonstrate this ability, consider a model that has been exposed to the following sentences:

1. The man was furious .
2. The men were drunk .
3. The man was grateful .
4. The men were organized .
5. The man who stole the briefcase was worried .
6. The men who shot the sheriff were scared .
7. The man who ran the race was tired .
8. The men who swam the river were fast .

Figure 4 and Figure 5 show the working memory and sequential memory representations when the model is probed with the singular form ("The man who saw the altar and gave thanks # # .") and the plural form ("The men who saw the altar and gave thanks # # .") respectively. After seven intervening items (i.e. the first of the # symbols) the model predicts that "was" is more likely to occur following "man", while "were" is more probable following "men".

It is the ability to retrieve appropriate traces from sequential memory that underpins performance. In the singular case (Figure 4), the traces "the man who stole the briefcase was worried ." and "the man who ran the race was tired ." dominate retrieval. Both of

these traces predict that “was” will occur. By contrast, in the plural case (Figure 5) it is the traces “the men who shot the sheriff were scared .” and “the men who swam the river were fast .” that dominate retrieval, both of which predict “were” will occur.

Unlike the SRN, the SP mechanism employs no hidden unit space in which words must compete to be retained. As a consequence, vocabulary size has little effect on the performance of the model. What is critical is the number of similar sequential traces in memory and whether these traces have gaps of equivalent length to that which must be subtended to retain the relevant plurality information.

Even when the corpus does not contain traces with long gap lengths the gap probability model (i.e. that initial indels have low probability while continuing indels have higher probability) allows the model to generalize to long gap lengths. To examine the ability of the model to retain plurality information, the gap length was manipulated by adding conjunctions to the embedded clause (e.g. “the man who saw the altar and gave thanks and knelt down # # .”), while keeping the background corpus constant. Figure 6 shows the substitution probabilities of “was” (correct plurality) and “were” (incorrect plurality) for the first # symbol as the gap length went from four to thirteen. Although there is clearly a decrement in performance as the gap increases, even after thirteen items there remains some ability to distinguish appropriate plurality information. In the corpus, however, the maximum gap length is four items.

Working Memory

the: the (1.00)
man: man (.96) men (.04)
who: who (.96) was (.02)
saw: stole (.45) ran (.45) shot (.02) swam (.02)
the: the (.96)
altar: briefcase (.27) race (.27) was (.02) sheriff (.01) river (.01)
and: was (.29) briefcase (.03) race (.03) were (.01)
gave: was (.08) worried (.06) tired (.06) briefcase (.03) race (.03)
thanks: was (.08) briefcase (.06) race (.06) worried (.03) tired (.03)
#: was (.29) worried (.03) tired (.03) were (.01)
#: worried (.27) tired (.27) was (.02) scared (.01) fast (.01)
.: . (1.00)

Sequential Memory

0.460 the man who stole the briefcase was worried .
0.460 the man who ran the race was tired .
0.020 the men who shot the sheriff were scared .
0.020 the men who swam the river were fast .
0.019 the man was furious .
0.019 the man was grateful .
0.001 the men were drunk .
0.001 the men were organized .

Figure 4: Demonstration of long term dependencies (singular)

Working Memory

the: the (1.00)
men: men (.96) man (.04)
who: who (.96) were (.02)
saw: shot (.45) swam (.45) stole (.02) ran (.02)
the: the (.96)
altar: sheriff (.27) river (.27) were (.02) briefcase (.01) race (.01)
and: were (.29) sheriff (.03) river (.03) was (.01)
gave: were (.08) scared (.06) fast (.06) sheriff (.03) river (.03)
thanks: were (.08) sheriff (.06) river (.06) scared (.03) fast (.03)
#: were (.29) scared (.03) fast (.03) was (.01)
#: scared (.27) fast (.27) were (.02) worried (.01) tired (.01)
.: . (1.00)

Sequential Memory

0.460 the men who shot the sheriff were scared .
0.460 the men who swam the river were fast .
0.020 the man who stole the briefcase was worried .
0.020 the man who ran the race was tired .
0.019 the men were drunk .
0.019 the men were organized .
0.001 the man was furious .
0.001 the man was grateful .

Figure 5: Demonstration of long term dependencies (plural)

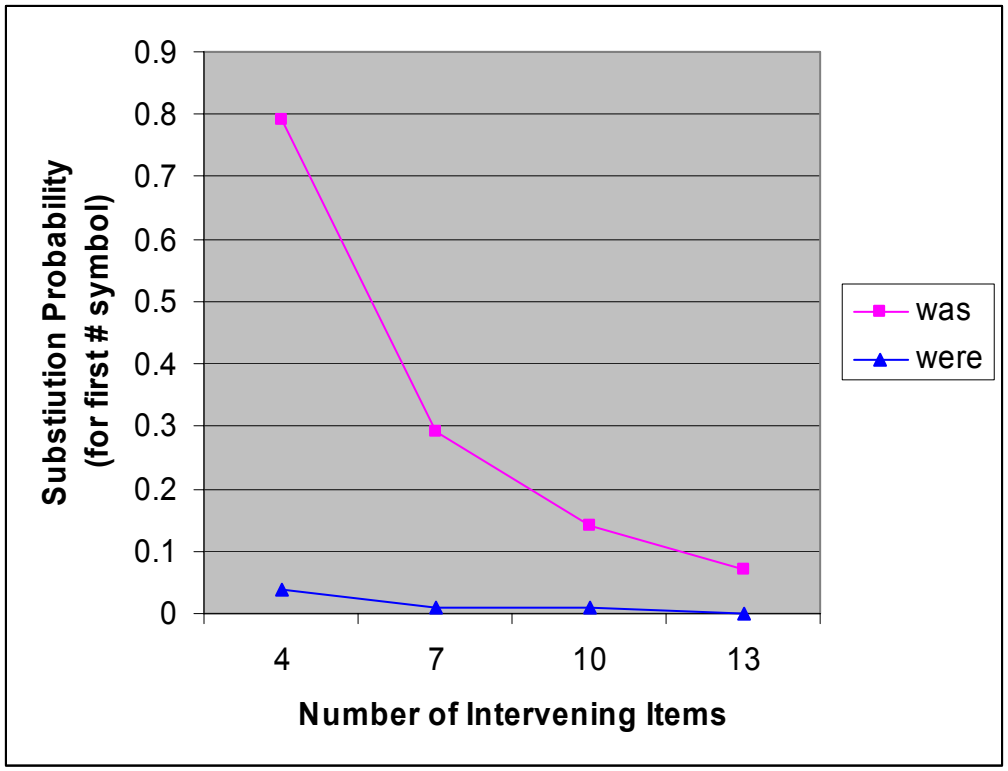

Figure 6: The impact of gap length. "was" is the verb of appropriate plurality, while "were" is inappropriate. Note the following adjustment to the default parameters was necessary Gap Factor = 7.

Structure Sensitivity

In addition to being able to retain dependency information over long gaps people show sensitivity to structure when employing dependency information. For instance, in the sentence "the cat the cats see sees" the plurality information in the embedded clause does not prevent people from correctly retaining and using the plurality information from the initial noun when analysing the final verb.

Chomsky (1957) argued on the basis of a number of recursive structures including this type of center-embedding that associative mechanisms were inadequate to explain human language. This position has been attacked on two grounds. Firstly, people find center-embedded structures difficult to process. When more than one level of embedding is present in a sentence, participants will often judge it ungrammatical (Marks 1968) and have difficulty retaining the content of the sentence (Miller & Isard 1964). Note, however, that people do remain above chance, so that it is still necessary to account for their ability to process these sentences.

Secondly, Christiansen and Chater (1999) have demonstrated that the Simple Recurrent Network (SRN, Elman 1991) is capable of correctly processing the recursive structures

discussed by Chomsky (1957). In a series of simulations they showed that a network trained on examples of counting recursive, mirror recursive (i.e. center embedded) and identity recursive structures could make appropriate class predictions given novel examples. So, while the simplest associative mechanisms may have difficulties with recursive structures the inclusion of a hidden layer overcomes this limitation.

In the SP model there is no reified notion of a clause or phase and no hidden layer that would allow the development of any type of bridging representation. All sequential representations are just sequences of words. Nonetheless the model can capture at least some phenomena which may appear to require clause or phrase knowledge, such as center embedding.

To demonstrate the point, suppose that the sequential memory of the SP model has been loaded with the following corpus:

1. the dogs see
2. the cats see
3. the dog sees
4. the cat sees
5. the cats the cats chase chase
6. the cats the cat chases chase
7. the cat the cats chase chases
8. the cat the cat chases chases

As a consequence of training, an association is formed between “dog” and “cat” and between “dogs” and “cats” because they fill the same slots in traces one to four. If the model is probed with “the dog the dogs # #”, the working memory representation in Figure 7 results.

Note that in the empty slots (represented by the # symbols) the plural verb “chase” precedes the singular verb “chases” as is required by the order of the singular and plural nouns. That is, the plurality information from the first noun is maintained over the embedded clause as if the model were representing a phrase structure tree.

Again, retrieval of appropriate traces underpins performance. In the fragment used as a probe, “the dog the dogs # #” the singular noun occurs before the plural noun and so the most probable sequential trace is “the cat the cats chase chases”. This trace contains the verbs in the appropriate order, which supports appropriate resolution.

Working Memory

the: the (1.00)
dog: cat (.71) cats (.29)
the: the (1.00)
dogs: cats (.71) cat (.29)
#: chase (.71) chases (.29)
#: chases (.71) chase (.29)

Sequential Memory

0.445 the cat the cats chase chases
0.262 the cat the cat chases chases
0.261 the cats the cats chase chase
0.032 the cats the cat chases chase
0.000 the cats see
0.000 the cat sees
0.000 the dogs see
0.000 the dog sees

Figure 7: Demonstration of structure sensitivity

In principle, then the SP model is capable of demonstrating structure sensitivity. However, performance in these tasks is dependent on the appropriate traces being available in memory. Examples of sentences containing deep embedding are rare in human discourse and so performance should be expected to degrade rapidly as is the case. However, because the SP model is a trace-based mechanism the provision of just a couple of traces, especially if their retrieval is supported by local context, may be sufficient to improve performance dramatically. This would seem to be an interesting avenue for further research.

Generativity

The importance of the recursive structures discussed in the previous section goes beyond the demonstration of structure sensitivity. In addition, the ability of people to process these sentences has been taken to show that people are generative, that is, capable of processing and generating sentence structures for which they have no direct experience (Chomsky 1957).

Christiansen and Chater (1999) have demonstrated that the SRN is capable of generalizing to greater levels of embedding than were present in its training corpus (see also Wiles & Elman 1995). However, in a model, such as the SP model, that operates by retrieving specific sentence instances, one may be concerned that only those sentence structures that appear in memory will be processed correctly. That is, that the model will

be incapable of demonstrating the generativity that is characteristic of human language users. However, because the SP model incorporates constraints from multiple sequential traces during resolution it is able to generalize beyond the structures represented in memory. To test the model's generative ability, the corpus below was constructed:

1. the cat the man hears chases
2. the cat the men hear chases
3. the cats the man hears chase
4. the cats the men hear chase
5. the man the dog sees hears
6. the man the dogs see hears
7. the men the dog sees hear
8. the men the dogs see hear

Note that within this corpus, sentences contain only one level of center-embedding. Now, if the model is probed with "the cats the men the dog # # #" the representations in Figure 8 result. Both "the cat the men hear chases" and "the men the dog sees hear" traces are retrieved. As a consequence both sets of syntagmatic constraints will apply and the model will correctly complete with "sees hear chase", despite the fact that the model was never exposed to a sentence that contained two levels of center-embedding.

Although, the model is able to generalize appropriately in this case, the probabilities of items in the empty slots are lower than in the previous examples. Furthermore, there are alternatives to the correct items within the each slot that have significant probabilities. As a consequence the model would predict that people who have never been exposed to these sorts of constructions should experience significant difficulty in processing them.

Working Memory

the: the (.82)
cat: cat (.63)
the: the (.81)
men: men (.98)
the: the (.37) hear (.27)
dog: dog (.36) hear (.14) chases (.08)
#: sees (.23) hear (.11) chases (.11)
#: hear (.20) chases (.15) sees (.12)
#: chases (.27) hear (.23)

Sequential Memory

0.623 the cat the men hear chases
0.361 the men the dog sees hear
0.007 the cat the man hears chases
0.004 the man the dog sees hears
0.003 the cats the men hear chase
0.002 the men the dogs see hear
0.000 the cats the man hears chase
0.000 the man the dogs see hears

Figure 8: Demonstration of generativity. Note the following adjustments to the default parameters were necessary $P(< x, x >| \overline{S_k \mapsto T}) = 0.025$ and $P(< x, y >| \overline{S_k \mapsto T}) = 0.2$

Surface Structure Independence

One of the ubiquitous characteristics of human languages is the way in which multiple surface forms can express the same underlying factual content. So, for instance, the sentence “John loves Mary” represents the same relational information as “Mary is loved by John”, while “John is loved by Mary” which on the surface appears more similar actually represents different relational content.

Traditional linguistic theories, especially in the generative tradition, have sought to capture this through a system of transformation rules (Chomsky 1957). Within this tradition, it has been assumed that these rules cannot be derived from inductive data-driven processes alone. Connectionist approaches, such as the SRN, have not directly addressed the issue and will have significant difficulties in doing so as their mechanisms are directly tied to the sequence of words as they appear in the input.

The SP model offers a mechanism by which relational structure can be captured in a surface form independent fashion. To illustrate the point assume that sequential memory contains the following traces:

1. Bert loves Ellen.
2. Steve loves Jody.
3. Ellen is loved by Bert.
4. Jody is loved by Steve.

Figure 9 shows the relational representation of the active sentence "John loves Mary", the passive sentence "Mary is loved by John." and the relationally dissimilar sentence "John is loved by Mary." Despite the fact that the sequential traces for the active and passive forms are different, the relational traces are similar, while the relational trace for the dissimilar form binds fillers to incorrect roles.

Relational Representation (John loves Mary)

john: bert (.49) steve (.49)
mary: ellen (.49) jody (.49)

Relational Representation (Mary is loved by John)

john: bert (.49) steve (.49)
mary: ellen (.49) jody (.49)

Relational Representation (John is loved by Mary)

john: ellen (.49) jody (.49)
mary: bert (.49) steve (.49)

Figure 9: Demonstration of surface form independence

The surface form independence demonstrated by the model relies on the existence of traces of different surface forms that contain the same tokens (i.e. "Bert loves Ellen" and "Ellen is loved by Bert."). While such restatements are likely to be rare in written text, in spoken discourse it is quite common for an interlocutor to repeat a fact using a different surface form either as a point of clarification or emphasis. Furthermore, the model relies only on there being a subset of such occurrences. It is not necessary that every instance of an active construction have a corresponding passive exemplar.

In this demonstration we have used the active to passive equivalence as an example. Another example of an alternative surface structure is the interrogative. In the next section, this form is used to illustrate the systematic properties of the SP model.

Systematicity

The strongest criticism of connectionist models of language processing and learning is that they are not able to capture the systematic nature of language (Fodor & Pylyshyn, 1988; Marcus, 1998; but see Elman 1998, for counter arguments). In particular, Fodor and Pylyshyn (1988) argued that anyone capable of understanding the utterance "John loves Mary" will also be capable of understanding the utterance "Mary loves John." – that is, you do not find people who can bind Mary to the object role but not the subject role.

As an illustration of people's ability to form arbitrary bindings consider the famous skit by Abbott and Costello in which Abbott is attempting to familiarize Costello with the baseball players on his new team:

Abbott: we have Who's on first, What's on second, I Don't Know's on third.
Costello: That's what I wanna find out.
Abbott: I say Who's on first, What's on second, I Don't Know's on third -
Costello: You know the fellows' names?
Abbott: Certainly!
Costello: Well then who's on first?
Abbott: Yes!
Costello: I mean the fellow's name!
Abbott: Who!
Costello: The guy on first!
Abbott: Who!
Costello: The first baseman!
Abbott: Who!
Costello: The guy playing first!
Abbott: Who is on first!
Costello: Now whaddya askin' me for?
…

In order to understand the humour behind the skit one must realize that "Who" is the subject of the sentence and is not marking a question. Although this ambiguity creates confusion, ultimately most English speakers are able to bind "Who" to this unfamiliar role.

Connectionist models, however, do not as a consequence of their basic mechanisms make such generalizations. While it is possible to train a network to exhibit this kind of systematicity, it has been argued that the amount of data required grows in proportion to the (combinatorial) number of possible utterances and is unrealistic (Phillips 2000, Hadley 1994).

By contrast, the SP model does exhibit such systematicity because it is able to bind arbitrary role vectors to items when creating relational representations. To demonstrate the point, consider the following corpus:

1. joan is loved by bert
2. sue is loved by mark
3. jenny is loved by joe
4. mary is loved by john
5. Who does bert love ? joan
6. Who does mark love ? sue
7. Who does joe love ? jenny

Figure 10 shows the working memory representation after relational resolution when the model has been probed with "Who does John love ? #". The model has correctly inferred that it is "Mary" that should fill the answer slot despite the fact that "Mary is loved by John" appears in the passive form but not in the interrogative form and "Mary" does not appear as the answer to any question. Furthermore, no lexical training was employed so the ability to assign Mary to the answer slot is attributable to relational retrieval and resolution exclusively.

Working Memory

who: who (1.00)
does: does (1.00)
john: john (.82) bert (.34) mark (.34) joe (.34)
love: love (1.00)
?: ? (1.00)
#: mary (.77) joan (.36) sue (.36) jenny (.36) loved (.01) is (.01) by (.01) john (.01)

Figure 10: Demonstration of systematicity

One possible concern is that the model is too systematic. In the discussion of the Abbott and Costello example it was noted that people do experience some initial confusion which must be overcome in order to understand the skit. While it may be that they are reluctant to make a binding between "Who" and the subject role, another possibility is that they retrieve inappropriate sequential traces. An initial "Who" is indicative of a question and so initially one would expect that sequential traces representing questions would be retrieved. As evidence amounts that there is a problem with the interpretation the initial retrieval may be revised.

To this point, the model has been used as if the entire probe were available simultaneously. Generally, however, words become available sequentially and there is a large literature on the time course of sentence processing. While it is beyond the scope of the current paper to address these phenomena, in the next section we illustrate how the model might be applied to this domain using garden path sentences as an example.

Online Processing: Garden Path Sentences

A garden path sentence is one in which an interpretation suggested by the initial words in the sentence must be overridden when subsequent words are found to be contradictory. That is, the start of the sentence leads people "down the garden path". The classic example is "The horse raced past the barn fell." (Bever 1970). Initially, having read or heard "The horse raced past the barn" most people will assume they are processing a simple active sentence. The word "fell", however, is inconsistent and they must revise to a reduced relative interpretation (i.e. "The horse that raced past the barn fell"). This revision is associated with long reading and fixation times for these inconsistent words.

Current models of garden path effects often rely upon the idea that as sentence processing proceeds people maintain multiple possible interpretations of the sentence. For instance, in the model by Narayanan and Jurafsky (1998), a Bayesian belief network containing Probabilistic Context Free Grammar (PCFG) probabilities as well as other lexical and thematic information is employed. If the probability of an interpretation becomes very small it is pruned – even in the case where it will turn out to be the correct interpretation. As a consequence, when the inconsistent word is encountered the person must return to the start of the sentence and conduct another parse thus inducing a long reading time.

In the SP model a similar mechanism is employed, however, it is sentence instances in sequential memory rather than abstract internal representations that are activated. Furthermore, no pruning or reparsing is required. To illustrate consider the garden path sentence "The waiter served calzone complained." This is a garden path sentence because having read "The waiter served calzone" people are inclined to presume that it is a simple active sentence and that it was the waiter who was doing the serving. When the word "complained" becomes available, however, it becomes apparent that the waiter was in fact a customer and was the one being served. This garden path effect is reinforced by the fact that waiters typically do serve and can be eliminated by substituting "customer" for "waiter".

If the SP model contains the following corpus:

1. the actress served duck complained .
2. the customer served meat complained .
3. the waiter served pizza .
4. the waitress served oysters .

and is presented with the probe "the waiter served calzone complained ." one word at a time the sequence of working memory and sequential memory representations seen in Figure 11 results.

Working Memory	Sequential Memory
the: the (.99) #: . (.24) served (.24) pizza (.11) oysters (.11) waiter (.11) waitress (.11) complained (.01)	0.462 the waiter served pizza . 0.462 the waitress served oysters . 0.038 the actress served duck complained . 0.038 the customer served meat complained .
the: the (1.00) waiter: waiter (.96) waitress (.02) served (.02) #: . (.33) served (.32) pizza (.32) oysters (.01)	0.961 the waiter served pizza . 0.033 the waitress served oysters . 0.003 the actress served duck complained . 0.003 the customer served meat complained .
the: the (1.00) waiter: waiter (.95) waitress (.05) served: served (1.00) #: . (.49) pizza (.46) oysters (.02)	0.946 the waiter served pizza . 0.047 the waitress served oysters . 0.004 the actress served duck complained . 0.004 the customer served meat complained .
the: the (1.00) waiter: waiter (.94) waitress (.05) served: served (1.00) calzone: pizza (.90) oysters (.05) . (.01) #: . (.95) pizza (.01)	0.940 the waiter served pizza . 0.049 the waitress served oysters . 0.006 the actress served duck complained . 0.006 the customer served meat complained .
the: the (1.00) waiter: actress (.34) customer (.34) waiter (.29) served: served (1.00) calzone: duck (.34) meat (.34) pizza (.19) complained: complained (.70) . (.10) pizza (.09) #: . (.88)	0.350 the actress served duck complained . 0.350 the customer served meat complained . 0.291 the waiter served pizza . 0.009 the waitress served oysters .
the: the (1.00) waiter: actress (.35) customer (.35) waiter (.28) served: served (1.00) calzone: duck (.35) meat (.35) pizza (.14) complained: complained (.71) pizza (.14) .: . (.99) #:	0.356 the actress served duck complained . 0.356 the customer served meat complained . 0.280 the waiter served pizza . 0.009 the waitress served oysters .

Figure 11: Demonstration of the garden path effect.

At the start of processing (steps two to four) there is little change in the emerging representation of the sentence. "waiter" is aligned with the pattern {waiter, waitress} which one could interpret as a "server" role vector. As soon as "complained" is presented, however, the representation changes dramatically. Now the reduced relative traces in sequential memory become active and "waiter" is aligned with the {actress, customer} pattern, which one might interpret as the "servee" role. The longer reading times for the garden path word occur not because a parse must be restarted from the beginning of the sentence (although this may happen on occasions), but because transitioning from the working memory and sequential memory representations in step four to those in step five takes additional time. By contrast, Figure 12 shows the sequence of working memory and sequential representations that results when the nongarden path probe "the customer served calzone complained ." is presented. In this case, there is never the dramatic change that occurs in the garden path example, and consequently no long reading time is induced.

Working Memory	Sequential Memory
the: the (.99) #: . (.24) served (.24) pizza (.11) oysters (.11) waiter (.11) waitress (.11) complained (.01)	0.462 the waiter served pizza . 0.462 the waitress served oysters . 0.038 the actress served duck complained . 0.038 the customer served meat complained .
the: the (1.00) customer: customer (.53) served (.15) waiter (.11) waitress (.11) pizza (.04) oysters (.04) #: . (.36) served (.21) complained (.14) meat (.13) pizza (.07) oysters (.07)	0.534 the customer served meat complained . 0.223 the waiter served pizza . 0.223 the waitress served oysters . 0.020 the actress served duck complained .
the: the (1.00) customer: customer (.58) waiter (.20) waitress (.20) actress (.02) served: served (1.00) #: . (.40) complained (.20) meat (.19) pizza (.10) oysters (.10)	0.582 the customer served meat complained . 0.201 the waiter served pizza . 0.201 the waitress served oysters . 0.016 the actress served duck complained .
the: the (1.00) customer: customer (.64) waiter (.17) waitress (.17) actress (.02) served: served (1.00) calzone: meat (.41) complained (.22) pizza (.16) oysters (.16) duck(.01) #: . (.75) complained (.21)	0.642 the customer served meat complained . 0.169 the waiter served pizza . 0.169 the waitress served oysters . 0.019 the actress served duck complained .
the: the (1.00) customer: customer (.87) actress (.12) served: served (1.00) calzone: meat (.85) duck (.12) complained: complained (.99) #: . (.97)	0.873 the customer served meat complained . 0.121 the actress served duck complained . 0.003 the waiter served pizza . 0.003 the waitress served oysters .
the: the (1.00) customer: customer (.78) actress (.20) served: served (1.00) calzone: meat (.76) duck (.20) complained: complained (.99) .: . (1.00) #:	0.782 the customer served meat complained . 0.207 the actress served duck complained . 0.005 the waiter served pizza . 0.005 the waitress served oysters .

Figure 12: Demonstration of the non garden path control sentence

In the preceding section, a number of important properties of the SP model have been illustrated. Its ability to deal with long distance dependencies, structure sensitivity, generativity, surface form independence and systematicity suggest that it is capable of meeting some critical preconditions for an account of sentence interpretation. In addition, the model's explanation of garden path effects is suggestive that it may also prove to be a reasonable account of online sentence processing. In the next section, we move from sentence processing into the semantic memory domain.

Semantic Memory: Categorization and Rating

Early work on semantic memory focused on how conceptual knowledge is structured. Our understanding has evolved from the hierarchical model (Collins & Quillian 1969), through spreading activation models (Collins & Loftus 1975) to feature based models (Smith, Shoben & Rips 1974). Subsequently, models such as ACT (Anderson 1983, 1993, Anderson & Lebiere 1998) and Construction Integration theory (Kintsch 1998) have emphasized the need for propositional representations to capture the relationships between concepts.

More recently interest in semantic memory has shifted to mechanisms for acquiring word meaning from text corpora. Models such Latent Semantic Analysis (LSA, Landauer & Dumais 1997) and Hyperspace Analog to Language (HAL, Lund & Burgess) have provided important insights into vocabulary development, text comprehension and neurological impairments of semantic memory.

The SP model is an attempt to extend this understanding in two important ways. Firstly, the model provides an automated method for extracting relational (or propositional) information from a corpus. One of the remarkable successes of LSA has been its ability to account for the comprehension of text by adding the vectors representing the words in that text (Landauer & Dumias 1997). However, as many theorists have noted (Anderson 1983, Kintsch 1998), it is clear that there are occasions on which a level of representation beyond that captured by the constituent words alone is necessary. To use our example from before, “Mary is loved by John” is not identical in meaning to “John is loved by Mary” and the SP model not only explains this difference, but also provides a data-driven mechanism for the extraction of this knowledge.

Secondly, the SP model provides a partial explanation of the control processes of semantic memory. LSA and HAL are both representational theories. They provide mechanisms for acquiring semantic knowledge and a list of procedures for how that knowledge might be used in different tasks. But they do not describe how the cognitive system acquired these procedures or how it decides when each should be employed. In this section, we will examine two tasks, categorization and rating, and demonstrate how the SP model exploits stored sequential fragments to distinguish the demands of different tasks.

A key point in addressing control issues is that the same information must be exploitable by different control procedures. Rather than create an independent corpus for the categorization and rating tasks, then, we will use a single corpus and use the retrieval probes to distinguish tasks.

The following corpus, loosely based around the Rumelhart and Abrahamson (1973) animal space, was created to illustrate the main points:

1. the lion searched for prey .
2. the lion walked across the savannah .
3. the mouse ate the leaves .
4. the mouse scurried through the hole .
5. the elephant ambled across the savannah .
6. the elephant ate the leaves .
7. the tiger looked for prey .
8. the tiger ran across the savannah .
9. the camel ambled across the dessert .
10. the camel ate the leaves .
11. the hamster nibbled the leaves .
12. the hamster ran through the hole .
13. a elephant is a herbivore .
14. a lion is a carnivore .
15. is a lion fierce ? yes
16. is a mouse fierce ? no
17. is a elephant fierce ? no
18. is a lion big ? yes
19. is a mouse big ? no
20. is a elephant big ? yes

Traces one to six establish some properties of a set of reference animals, namely lions, mice and elephants. Lions and elephants are sorts of animals that walk across savannahs, while mice are sorts of animals the scurry through holes. Likewise mice and elephants are sorts of animals that eat leaves, while lions search for prey.

Traces seven to twelve establish the properties of a set of test animals. Tigers share properties with lions, hamsters share properties with mice and camels share properties with elephants.

Traces thirteen and fourteen categorize elephants as herbivores and lions as carnivores respectively, while traces fifteen to twenty rate (on a two point scale) the ferocity and size of the reference animals (lions, mice and elephants).

Categorization

The first task that will be considered is semantic categorization. How do people on the basis of known facts assign an item to a category? Note that while tigers share properties with lions there is no explicit information classifying them as carnivores. Similarly, while hamsters and camels are never explicitly classified as herbivores they do share properties with mice and elephants. The model correctly completes the empty slot with “carnivore” when presented with the probe “a tiger is a # .” (see Figure 13). Similarly, hamsters and camels are identified as herbivores.

The completion is possible because during lexical training "tiger" is associated with "lion" because the slots it fills in traces seven and eight are the same as those filled by "lion" in traces one and two. As a consequence, during relational resolution the role filler binding {tiger: lion, elephant} is similar to {lion: lion, elephant} and so trace fourteen ("a lion is a carnivore") is retrieved (see Figure 13). This trace has "carnivore" bound to the category role (i.e. the {carnivore, herbivore} vector) and so "carnivore" is completed in the empty slot.

Working Memory	Relational Memory
a: a (1.00) tiger: lion (.83) elephant (.66) is: is (1.00) a: a (1.00) #: carnivore (.83) herbivore (.66) .: . (1.00)	0.668 a lion is a carnivore . 0.331 a elephant is a herbivore .
a: a (1.00) hamster: elephant (.97) lion (.37) is: is (1.00) a: a (1.00) #: herbivore (.97) carnivore (.37) .: . (1.00)	0.915 a elephant is a herbivore . 0.078 a lion is a carnivore . 0.002 the hamster nibbled the leaves . 0.001 the hamster ran through the hole . 0.001 is a mouse fierce ? no 0.001 is a mouse big ? no 0.001 the mouse ate the leaves .
a: a (1.00) camel: elephant (.98) lion (.49) is: is (1.00) a: a (1.00) #: herbivore (.98) carnivore (.49) .: . (1.00)	0.964 a elephant is a herbivore . 0.035 a lion is a carnivore .

Figure 13: Working memory and relational memory following relational resolution in the semantic categorization task. The model correctly classifies tigers as carnivores and hamsters and elephants as herbivores.

Note that the model has not been altered in anyway in light of the fact that this is a categorization task. The linguistic context is sufficient to establish a categorization virtual machine which the architecture then implements (c.f. Wiles & Bloesch 1992). As will be demonstrated in the next section, given a different linguistic context a different virtual machine is established. It is in this sense that the claim is made that the SP model provides a partial solution to the control problem.

Rating

We will now consider what would be required for the SP model to complete a semantic rating task. The model's task will be to determine the size and ferocity of tigers, hamsters and camels on the basis of shared properties with lions, mice and elephants. Figure 14 shows working memory following relational resolution in each case. It has determined that tigers are fierce, while hamsters and camels are not fierce.

Working Memory	Relational Memory
a: a (1.00) tiger: lion (.80) elephant (.55) mouse (.30) fierce: fierce (.98) big (.51) ?: ? (1.00) #: yes (.86) no (.63)	0.652 is a lion fierce ? yes 0.294 is a elephant fierce ? no 0.035 is a lion big ? yes 0.014 is a elephant big ? yes 0.005 is a mouse fierce ? no
is: is (1.00) a: a (1.00) hamster: mouse (.91) elephant (.44) lion (.29) fierce: fierce (.97) big (.51) ?: ? (1.00) #: no (.99) yes (.47)	0.819 is a mouse fierce ? no 0.124 is a elephant fierce ? no 0.044 is a mouse big ? no 0.008 is a elephant big ? yes 0.005 is a lion fierce ? yes
is: is (1.00) a: a (1.00) camel: elephant (.71) mouse (.61) lion (.34) fierce: fierce (.97) big (.52) ?: ? (1.00) #: no (.97) yes (.52)	0.530 is a elephant fierce ? no 0.397 is a mouse fierce ? no 0.029 is a elephant big ? yes 0.021 is a mouse big ? no 0.021 is a lion fierce ? yes 0.001 is a lion big ? yes

Figure 14: Working memory and relational memory following relational resolution when the model is asked to determine the ferocity of tigers, hamsters and camels.

As before the task is completed on the basis of the lexical representations formed during training. Because tigers share slots with lions, hamsters with mice and camels with elephants, the lexical system establishes the corresponding substitution probabilities. When presented with "Is a tiger fierce?", for example, trace fifteen ("Is a lion fierce?") is favoured during relational retrieval (see Figure 14). In this trace the rating slot is filled with "yes" and hence the model responds as required. Similarly, hamsters and camels are considered not to be fierce.

Note that the sentence context of the rating question retrieves a different set of sequential traces from the categorization question. As a consequence, the role vector that aligns with the answer slot is different, reflecting the change in task and the change in the virtual machine required to complete the task.

Also, when presented with a different rating question, "Is a X big?", the pattern of results changes substantially. Now tigers and camels are identified as big, while hamsters are not (see Figure 15). Again this difference occurs because different sets of traces are retrieved.

Working Memory	Relational Memory
is: is (1.00) a: a (1.00) tiger: lion (.82) elephant (.53) mouse (.30) big: big (.97) fierce (.51) ?: ? (1.00) #: yes (.99) no (.48)	0.678 is a lion big ? yes 0.264 is a elephant big ? yes 0.037 is a lion fierce ? yes 0.017 is a elephant fierce ? no 0.005 is a mouse big ? no
is: is (1.00) a: a (1.00) hamster: mouse (.90) elephant (.46) lion (.28) big: big (.98) fierce (.50) ?: ? (1.00) #: no (.93) yes (.54)	0.799 is a mouse big ? no 0.146 is a elephant big ? yes 0.043 is a mouse fierce ? no 0.007 is a elephant fierce ? no 0.005 is a lion big ? yes
is: is (1.00) a: a (1.00) camel: elephant (.71) mouse (.61) lion (.34) big: big (.97) fierce (.52) ?: ? (1.00) #: yes (.78) no (.72)	0.537 is a elephant big ? yes 0.392 is a mouse big ? no 0.028 is a elephant fierce ? no 0.021 is a mouse fierce ? no 0.020 is a lion big ? yes 0.001 is a lion fierce ? yes

Figure 15: Working memory and relational memory following relational resolution when the model is asked to determine the size of tigers, hamsters and camels.

As a consequence of its basic mechanisms, the SP model implements what is effectively a common features model of similarity. However, it is well established that when making similarity judgments, subjects also use distinctive features to establish the dissimilarity of two stimuli (Tversky 1977). One possibility is that distinctive features are used only when subjects are employing explicit reasoning processes. In support of this claim, Navarro (2002) found that subjects find it easy to choose the two most similar stimuli from a group of four countries, but have great difficulty choosing the two most dissimilar stimuli. Perhaps this occurs because distinctive features are not a fundamental property of the underlying similarity representation, but are rather a property of the similarity rating process under certain conditions. Discovering appropriate distinctive features when there are four alternatives from which only two must be chosen might involve having to consider pair wise combinations of stimuli, explicitly.

In any case, it is clear that on occasions, particularly in more abstract domains when linguistic experience with given concepts is limited, a more explicit level of reasoning might occur. Tigers might be identified as carnivores, not because they share properties in general with lions, but because the category carnivore has associated with it a necessary and sufficient condition – namely that the animal must eat meat – and that it is by the retrieval of this fact in relation to a given animal that a decision is made. In a latter section on inference, the distinction between implicit and explicit reasoning will be addressed and the use of implicit reasoning in the current simulations is not meant to imply that explicit processes are never employed in these sorts of circumstances.

To reiterate the point from the last section, it is the linguistic context that establishes that the tasks addressed here are rating tasks. Both the categorization task and the rating task access semantic memory, but the slots that require completion in each case are different. In the categorization task, the empty slot aligns with a category filler vector and so a category is required. In the rating tasks, the empty slot aligns with a rating filler vector and so a rating is required. The control architecture is established directly from the linguistic fragments without the intervention of an abstract level of control (e.g. a production system architecture).

Short Term Memory: Serial Recall

The area of short term memory and, in particular, serial recall has engendered a great deal of debate and led to the creation of a number of simple, yet powerful computational models. Current accounts can be divided into chaining models (e.g. Lewandowsky & Murdock 1989; Murdock 1995) in which list items are assumed to be associated each to the next, ordinal models (Page & Norris 1998) in which list items are assumed to be activated in proportion to their position in the list and positional models (Anderson & Matessa 1997; Brown Preece & Hulme 2000, Henson & Burgess 1997, Hitch, Burgess, Towse & Culpin 1996) in which list items are each associated with a unique positional cue. Of these three types of models it is the positional models that have been most influential in recent years. In particular, there are now a number of models that propose that temporal cues are generated through oscillatory mechanisms and that it is time rather than position per se that determines performance (Burgess & Hitch 1992; Brown, et. al. 2000; Henson & Burgess 1997).

While these models have been successful over a wide range of data they have been largely silent on the issue of how short term memory interacts with other cognitive processes, most notably with the language system. Since Miller and Selfridge (1950) it has been known that serial lists that mimic the sequential structure of language can induce dramatically increased short term memory spans. This suggests that rather than considering short term memory as a resource employed by the language system, it might be more appropriate to think of serial recall and other short term memory tasks as epiphenomena of the language processing apparatus. If this is the case, then an alignment based approach such as the SP model, which relies on retrieval from previous linguistic experience, may provide a viable account of serial recall.

The Serial Position Curve

In creating an SP account of any given phenomena we must first determine what previous experience is likely to be driving performance. In the serial recall task, it is presumably our experience with lists such as phone numbers, shopping lists etc that provide the traces upon which the control of the task depends. Suppose for instance that the sequential memory system contains the following traces:

1. C1 study the following list , bread milk shampoo fruit meat toothpaste .
2. C2 study the following list , Bill Mary Peter Harry Sue Bert .
3. C3 study the following list , oak gum willow birch pine aspen .
4. C1 recall the items bread milk shampoo fruit meat toothpaste .
5. C2 recall the items Bill Mary Peter Harry Sue Bert .
6. C3 recall the items oak gum willow birch pine aspen .
7. C4 study the following list , 1 2 3 4 5 6 7 .

C1-C4 represent context markers designed to isolate the list that must be recalled. As the current context is always used as a retrieval cue, traces from the corresponding study list are more available at recall (i.e. are more likely to be retrieved) than other traces.

Each of the traces is either an instance of studying a list or recalling a list. For the purposes of the example, quite stylized instructions have been used (i.e. "study the following list" or "recall the items"). It is assumed that a much broader set of possible utterances would be available and that the lexical system would allow the model to identify alternative ways of invoking the same process.

Note that there is no recall instance in the C4 context. This is the list that the model will be required to recall and so we probe the model with the following string "C4 recall the items # # # # # # # ." The items in the list have been labelled 1 through 7. Note, however, that this labelling is purely to facilitate interpretation of the results. The model only has access to the traces listed above and therefore has no background knowledge that would allow it to identify the numerical ordering of these labels.

Figure 16 shows the working memory representation following relational retrieval in the serial recall task and Figure 17 shows the probability of retrieval (i.e. the substitution probability of the correct item) as a function of serial position following relational retrieval.

c4: c4 (.99) c1 (.30) c2 (.30) c3 (.30)
recall: recall (1.00)
the: the (1.00)
items: items (1.00)
#: 1 (.59) bread (.22) bill (.22) oak (.22) 2 (.16) 3 (.05) milk (.03) mary (.03) gum (.03) list (.02) 4 (.02) , (.02) following (.02) the (.01) . (.01) study (.01) c4 (.01) 5 (.01) 7 (.01)
#: 2 (.38) 3 (.19) milk (.17) mary (.17) gum (.17) 1 (.17) 4 (.07) shampoo (.05) peter (.05) willow (.05) bread (.03) bill (.03) oak (.03) 5 (.03) , (.02) list (.02) following (.02) the (.02) . (.02)
#: 3 (.31) 4 (.19) 2 (.19) shampoo (.14) peter (.14) willow (.14) 5 (.08) 1 (.05) cheese (.05) sue (.05) pine (.05) milk (.05) mary (.05) gum (.05) 6 (.03) , (.02) list (.02) following (.02) the (.02)
#: 4 (.29) 3 (.19) 5 (.19) cheese (.14) sue (.14) pine (.14) 2 (.07) 6 (.07) shampoo (.05) peter (.05) willow (.05) tomatoes (.05) tom (.05) fir (.05) 1 (.02) 7 (.02) , (.02) list (.02) the (.02)
#: 5 (.31) 4 (.19) 6 (.19) tomatoes (.14) tom (.14) fir (.14) 3 (.08) cheese (.05) sue (.05) pine (.05) 7 (.05) meat (.05) jack (.05) beech (.05) 2 (.03) . (.02) , (.02) the (.02) list (.02)
#: 6 (.38) 5 (.19) meat (.17) jack (.17) beech (.17) 7 (.17) 4 (.07) tomatoes (.05) tom (.05) fir (.05) carrots (.03) ruby (.03) aspen (.03) 3 (.03) . (.02) the (.02) list (.02) following (.02) study (.02)
#: 7 (.60) carrots (.22) ruby (.22) aspen (.22) 6 (.16) 5 (.05) meat (.03) jack (.03) beech (.03) 4 (.02) . (.02) the (.01) following (.01) list (.01) study (.01) , (.01) c4 (.01) 3 (.01) 1 (.01)
.: . (1.00)

Figure 16: Working memory following relational retrieval in the serial recall task.

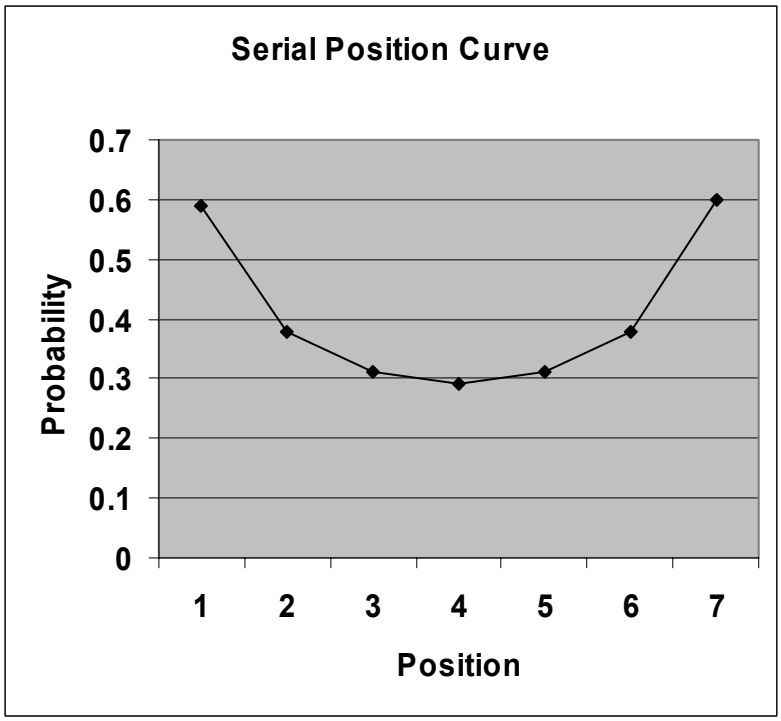

Figure 17: The serial position curve of the SP model.

Note that the serial position graph shows both the primary and recency components which are the hallmark of the serial recall task, although it tends to overestimate the size of the recency effect when compared to human data (Hitch, Burgess, Towse & Culpin 1996). To understand why the SP model produces the serial position curve consider the sequential representation formed during processing of the study list (Figure 18).

c4: c1 (.30) c2 (.30) c3 (.30)
study: study (1.00)
the: the (1.00)
following: following (1.00)
list: list (1.00)
,: , (1.00)
1: bread (.22) bill (.22) oak (.22) milk (.03) mary (.03) gum (.03)
2: milk (.17) mary (.17) gum (.17) shampoo (.05) peter (.05) willow (.05) bread (.03) bill (.03) oak (.03)
3: shampoo (.14) peter (.14) willow (.14) cheese (.05) sue (.05) pine (.05) milk (.05) mary (.05) gum (.05)
4: cheese (.14) sue (.14) pine (.14) shampoo (.05) peter (.05) willow (.05) tomatoes (.05) tom (.05) fir (.05)
5: tomatoes (.14) tom (.14) fir (.14) cheese (.05) sue (.05) pine (.05) meat (.05) jack (.05) beech (.05)
6: meat (.17) jack (.17) beech (.17) tomatoes (.05) tom (.05) fir (.05) carrots (.03) ruby (.03) aspen (.03)
7: carrots (.22) ruby (.22) aspen (.22) meat (.03) jack (.03) beech (.03)
.: . (1.00)

Figure 18: The working memory representation at study.

Note that the vectors associated with the start and end of the list have stronger representations than those closer to the middle of the list and therefore are stronger cues at test. The reason is that the instruction words and the end of the sentence form anchors

in the study list because they match in each of the retrieved sequential traces. Because alignments containing long contiguous gaps are preferred over alignments containing many short gaps, positional uncertainty increases as you move away from the anchor points. This compromises performance in the middle locations generating the serial position curve. The exaggerated recency effect shown by the model may occur because in the simulation the end of list marker (i.e. the period) is provided as part of the cue. In reality, however, subjects would have to project forward the location of the end-of-list marker making its location uncertain. This uncertainty would tend to decrease performance at final positions.

Intra-list Intrusions

In addition to producing the serial position curve, the uncertainty that accumulates as one moves away from the start and end anchors also means that intra list intrusions (i.e. producing an item that did appear on the list in an incorrect position) are more likely in adjacent positions and in the middle of the list (see Figure 19) as is the case in human data.

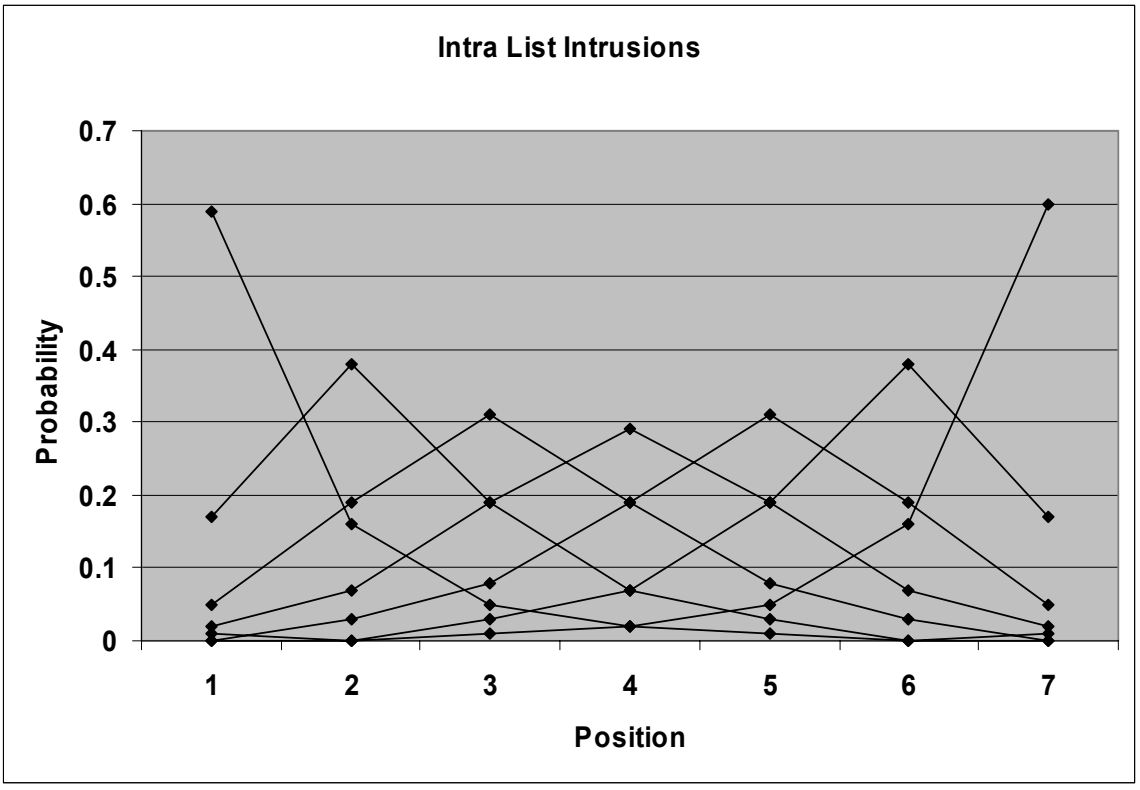

Figure 19: Intra list intrusions in the Serial Recall task. The graph shows the probability of producing an item from a given position in the study list as a function of output position.

Inter-list Intrusions

The most relevant traces in memory for the study episode are typically the immediately preceding lists. They are often of equivalent length, are constructed of similar materials and have identical sets of instructions. As a consequence, they are likely to be retrieved during study list processing and will become part of the relational representation of the current list. At test then, there will be a relatively high probability that items from the previous lists will intrude. Figure 20 shows the probability of substitution of items from a given position in each of the previous lists as a function of output position. Note that items are most likely to be output in the same or a similar position to that in which they appeared as is true of human subjects.

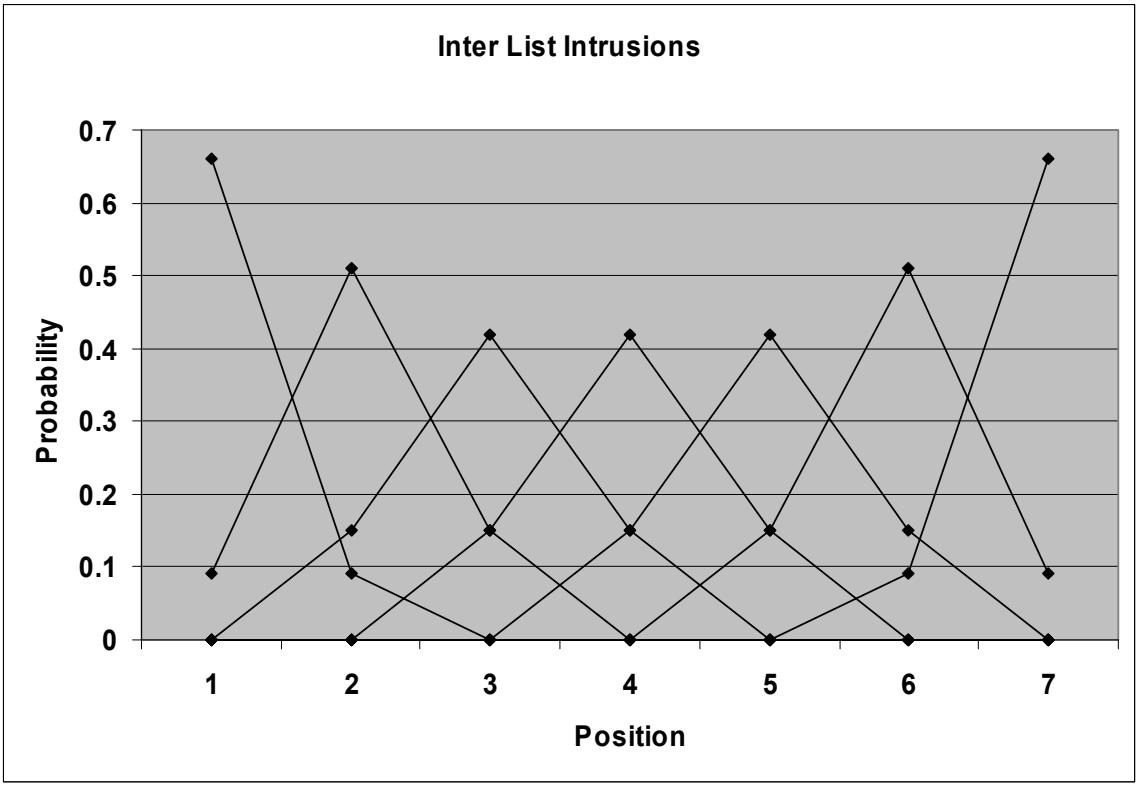

Figure 20: Inter list intrusions in the serial recall task. The graph shows the probability of producing an item from a given position in any of the previous lists in a given output position.

Serial Position with Grouped Lists

Finally, we consider how the SP model would deal with lists that are grouped using short pauses between groups. Grouped lists have been of recent interest as they allow the effects of timing versus position to be separated (c.f. Ng & Mayberry 2002). In one such experiment (Hitch et. al. 1996) subjects were presented with lists of nine items, each of which was divided into three groups of three with short pauses. As with other similar

designs an overall U shaped serial position curve is observed as well as mini-serial position curves within each group. To simulate this data the SP model was exposed to the following corpus:

1. C1 study , oak gum pine .
2. C1 recall . oak gum pine .
3. C2 study , bread milk muffins . shampoo tomatoes beans . meat carrots jam .
4. C2 recall . bread milk muffins . shampoo tomatoes beans . meat carrots jam .
5. C3 study , 1 2 3 . 4 5 6 . 7 8 9 .

The periods embedded in the list represent the pauses. When cued with "C3 recall . # # # . # # # . # # # ." the serial position curve seen in Figure 21 resulted. Again the model reproduces the shape of the human data reasonably well without any parameter fitting, except that the model displays a more pronounced recency effect than is typically the case.

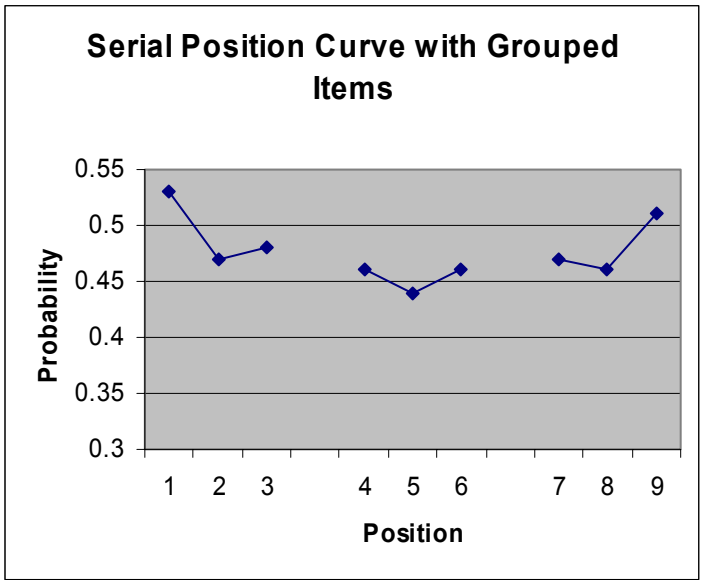

Figure 21: Serial position curve with grouped items

In qualitative terms, then, the SP model seems capable of addressing some key phenomena from the serial recall literature. There are clearly many more phenomena in this domain to which the model must be applied before it can be considered a serious contender as a model of short term memory. In one sense, however, the SP account of short term memory can be seen as an implementation of existing models such as the Start End model (Henson & Burgess 1997). While the SP model emphasizes position over the temporal factors that underpin many current models it seems that some form of oscillatory mechanism would be a good candidate for explaining how the sequential mechanism of the SP model could be implemented in a real time dynamical system (c.f.

Hummel & Holyoak 1997, Baird, Troyer & Eeckman 1992). If this is the case, then the SP account would be quite compatible with many existing models.

In another sense, though, the SP model departs from the normal way of viewing short term memory. Rather than assume that short term memory is a limited resource which the language system employs to carry out sentence processing, the SP model conceptualizes short term memory as an epiphenomena of the language system. If measures of short term memory are reliable indicators of linguistic performance it is because they are measuring linguistic capability directly, rather than because they are quantifying the resources available for linguistic processing. One need not resort to chunking hypotheses to reconcile how different amounts of information of different kinds are retained in a fixed capacity store, but can appeal to the varying levels of support these structures receive from sequential long term memory to account for span variability. While much remains to be done to substantiate such a view the SP model offers a blueprint for how the areas of short term memory and sentence processing might be more fundamentally integrated than is currently the case.

Any account of verbal processing must endeavour to describe how people are able to make the rapid and robust inferences that they do in the course of linguistic processing. The next section addresses this issue.

Inference

Kintsch (1998, see also Kintsch 1993) provides a two-by-two classification of the types of inference that typically occur during the process of text comprehension. On one dimension, inferences are distinguished by whether they involve the retrieval of existing information or the generation of new information. When making retrieval-based inferences, information enters working memory directly from long term memory in response to cues in short term memory. By contrast, generation-based inference involves the production of new knowledge that would not have existed in long term memory such as new facts generated by logical inference processes. On a second dimension, Kintsch (1998) distinguishes controlled from automatic inferences. Controlled inferences are strategic, resource demanding and subject to disruption under dual task conditions, whereas automatic inferences occur without strategic initiation and may be difficult for people to distinguish from facts that were presented explicitly in the text in subsequent memory tests (Bransford, Barclay & Franks 1972). To this classification I would add a distinction within the controlled inferences between logical inferences and analogical inferences. Logical inferences are those that implement a logical rule such as "if X sold the Y to Z then Z bought the Y from X". Analogical inferences involve mapping some base domain (e.g. a man is to a house) into a target domain (e.g. a dog is to a kennel) where no necessary logical relationship exists.

Kintsch (1998) argues that retrieval-based inferences should not be considered inferences as they simply involve the retrieval of existing background information. In this section, then, we will focus on the generation-based inferences, providing an SP account of automatic, controlled/logical and controlled/analogical inferences.

Automatic Inference

In production system models of inferencing, common sense knowledge is encoded in the form of if-then rules. For instance, most people know that "if X sold the Y to Z then it is also the case that Z bought the Y from X". Furthermore, it is the case that "Z now has the Y" and "X does not have the Y".

In production system parlance this might be expressed as:

```
if SOLD(X, Z, Y) then
        BOUGHT(Z, X, Y)
        POSSESS(Z, Y)
        NOT POSSESS(X, Y)
```

When attempting to establish whether "John owns a camera" the inferencing system might be asked to verify that POSSESS(John, camera), in which case it would activate the above rule and search memory to determine if John had ever been sold a camera (i.e. SOLD(X, John, camera)). This process, known as backward chaining, has two major disadvantages. Firstly, successful inference relies on all relevant common sense being

appropriately coded in production system rules. Secondly, even inferences that people would find trivial can be computationally expensive because there may be many such rules that must be consulted in order to establish the truth value of the predicate. In this example, John may possess a camera because he was given it, or because he found it, or because he stole it etc. All of these possibilities must be considered as new predicates which in turn may rely upon additional rules and so forth.

The SP model provides a mechanism that exploits corpus statistics to avoid the necessity of explicit inferencing of this kind. To illustrate, suppose the model had been exposed to the following corpus:

1. IBM sold the widget to Microsoft .
2. Microsoft bought the widget from IBM .
3. SPSS sold the software to SAS .
4. SAS bought the software from SPSS .
5. Liverpool sold the player to Manchester .
6. Manchester bought the player from Liverpool .

Figure 22 shows the relational representations for the sentences "Charlie bought the lemonade from Lucy ." and "Lucy sold the lemonade to Charlie ." As the order of bindings in relational traces is irrelevant these sentences have identical representations. That is, if we know that "Charlie bought the lemonade from Lucy" we automatically also know that "Lucy sold the lemonade to Charlie ." without ever explicitly extracting or applying a rule to effect the transformation. Furthermore, if we consider the pattern {Microsoft, SAS, Manchester} as an owner role then the model also automatically "knows" that Charlie owns the lemonade.

Charlie bought the lemonade from Lucy .

charlie: microsoft (.32) sas (.32) manchester (.32)
lemonade: widget (.33) software (.33) player (.33)
lucy: ibm (.32) spss (.32) liverpool (.32)

Lucy sold the lemonade to Charlie .

lucy: ibm (.32) spss (.32) liverpool (.32)
lemonade: widget (.33) software (.33) player (.33)
charlie: microsoft (.32) sas (.32) manchester (.32)

Figure 22: Implicit Inference: Relational Representations of "Charlie bought the lemonade from Lucy ." and "Lucy sold the lemonade to Charlie ."

By this account, the process that allows us to implicitly form simple inferences and the process that allows us to generalize over different surface forms of a sentence are identical. As in the case of surface form independence, it is the existence of a critical

subset of sequential traces that allows the inference to occur. One would not expect such inferences to be affected by dual task conditions, as beyond the processing of the sentence itself there are no additional processes that must be invoked. The process could be thought of as inference by coincidence. Note that this contrasts with the type of controlled/logical inference outlined in the next section.

Controlled/Logical Inference

The ability to follow an abstract rule is crucial to the success of logical reasoning processes. People are able to follow rules such as "if X sold the Y to Z then it is also the case that Z bought the Y from X" even in the absence of examples illustrating this principle. To show how this type of inference can be addressed suppose that the SP model had been exposed to the following corpus:

1. C1 if X sold the Y to Z
2. C1 then it is the case that Z bought the Y from X
3. C2 if Lucy sold the lemonade to Charlie

Note that trace one and two provide the linguistic fragments necessary to define the abstract rule, while trace three is a specific example, which the model will be required to use to infer that "Charlie bought the lemonade from Lucy". No examples, however, make the connection between buying and selling in the way that they did in the previous section, so the model cannot rely on automatic inference. Figure 23 shows working memory following sequential resolution when the model is probed with "C2 then it is the case that # # # # # #". The appropriate fragment of the abstract rule has been retrieved and used to complete the empty slots. At this point, however, it is the abstract variables that are instantiated rather than their values in the current context.

c2: c1 (.98)
then: then (1.00)
it: it (1.00)
is: is (1.00)
the: the (1.00)
case: case (1.00)
that: that (1.00)
#: z (.88) bought (.05)
#: bought (.81) the (.07) z (.05)
#: the (.78) y (.08) bought (.07)
#: y (.78) the (.08) from (.07)
#: from (.81) y (.07) x (.05)
#: x (.88) from (.05)

Figure 23: Explicit Inference: Working memory following sequential resolution.
Note the following adjustment to the default parameters was necessary
$P(< x, x >| \overline{S_k \mapsto T}) = 0.25$

Upon relational resolution the abstract variables (X, Y and Z) are replaced with their appropriate values within the current context (see Figure 24).

c2: c2 (1.00) c1 (.98)
then: then (1.00)
it: it (1.00)
is: is (1.00)
the: the (1.00)
case: case (1.00)
that: that (1.00)
#: charlie (.96) z (.88) bought (.05)
#: bought (.81) the (.52) charlie (.32) z (.05) lemonade (.04) sold (.03) if (.03) to (.03) c2 (.03) lucy (.03)
#: the (.97) lemonade (.09) y (.08) bought (.07)
#: lemonade (.87) y (.78) the (.16) from (.07)
#: from (.81) lemonade (.48) lucy (.32) y (.07) x (.05) the (.04) sold (.03) if (.03) to (.03) charlie (.03) c2 (.03)
#: lucy (.96) x (.88) from (.05)

Figure 24: Explicit inference: Working memory following relational resolution.

In this way, the inference has been made. Note, however, that the use of an abstract rule in this way does not lead to the rich set of bindings that were generated in the previous section. For instance, "Lucy" will be bound to {X} using the explicit rule whereas in the previous example "Lucy" was bound to {IBM, SPSS, Liverpool}, all symbols that have a grounded meaning and hence are liable to be used systematically within the systems experience. The symbol "X", on the other hand, may be used to mean many different things in different circumstances and consequently is not a useful symbol when making additional inferences such as inferring possession. So, while explicit reasoning is useful in domains where direct experience is limited, matching explicit rules may interfere with the process of acquiring richly interwoven knowledge structures when direct experience can be made available (a fact which may underpin differences in first and second language acquisition).

Also, note that implicit inference made the relevant information immediately available in relational memory, whereas explicit inference relies on the application of a rule which must be retrieved from sequential memory, resolved, retrieved from relational memory and resolved before the relational trace is entered into relational memory. This difference may explain why explicit inference takes more time, is more prone to dual task interference and why the process of explicit inference is more likely to be available for report.

In this example, the inference was made by the application of a logical rule. As suggested above, however, some controlled inference is made not by the application of a rule, but rather by analogy to a separate domain. The next section addresses this kind of controlled/analogical inference.

Controlled/Analogical Inference

There is now a well developed literature on analogical reasoning with a long history of well specified computational models (Kokinov & French 2002). A prerequisite for all of these models is a propositional representation of the relevant facts in the base and target domains. In some models these propositional representations are symbolic (ANALOGY, Evens 1964; SMT, Gentner 1983; Falkenhainer, Forbus & Gentner 1989; ACME, Holyoak & Thagard 1989), while other models employ distributed representations (STAR Halford, Wilson, Guo, Gayler, Wiles, Stewart 1994, Wilson, Halford, Gray & Phillips 2001, LISA, Hummel & Holyoak 1997). In either case, however, the practice has been to hand code the relevant facts in the appropriate representational format. While there has been considerable work on representation creation mechanisms (COPYCAT, Mitchell, 1993; Hofstadter, 1995; French, 1995), the focus in this work has been on allowing different aspects of an analogical mapping problem to be highlighted by dynamically changing the representational scheme rather than on extracting the relevant information from naturalistic corpora (e.g. text corpora). While the problem of extraction is in some sense ancillary to the problem of analogical reasoning, the inability to solve it has been an important limitation as a key issue in assessing the viability of an analogical reasoning model is its ability to scale to realistic size knowledge bases. The SP model has the potential to solve this dilemma as it provides a mechanism by which a propositional knowledge base can be extracted automatically from a large corpus.

In addition, however, the basic mechanisms of the SP model implement a form of analogical inference. To illustrate we will use the model to solve a simple proportional analogy such as MAN:HOUSE::DOG:?.

Suppose the model has been exposed to the following corpus:

1. a house is where a man lives
2. a kennel is where a dog lives
3. a mother is to a daughter
4. a father is to a son
5. CC a man is to a house

In a similar fashion to previous examples, we will use the symbol “CC” as an indication of the current context and probe the model with “CC a dog is to a #”. Figure 25 shows the working memory representations after syntactic resolution. The input word “dog” has been aligned with “man”, “mother” and “father” as a consequence of traces three, four and five and the empty slot at the end of the sentence is aligned with “house”, “daughter” and “son”.

cc: cc (.74)
a: a (1.00)
dog: man (.73) mother (.12) father (.12)
is: is (1.00)
to: to (1.00)
a: a (1.00)
#: house (.73) daughter (.12) son (.12)

Figure 25: Working Memory following Syntactic Resolution in the Analogical Inference Task

Now, the relational trace for "a kennel is where a dog lives" also has a binding of "man" onto "dog" and hence is selected in relational retrieval (see Figure 26).

0.934 a kennel is where a dog lives
0.025 a mother is to a daughter
0.025 a father is to a son
0.014 cc a man is to a house
0.001 a house is where a man lives

Figure 26: Relational Memory in the Analogical Inference task.

In the selected trace "kennel" is bound to "house" so during relational retrieval the empty slot is now filled with the correct answer (see Figure 27).

cc: cc (.74) a (.06) is (.03) where (.02) lives (.02) kennel (.02) dog (.02)
a: a (1.00)
dog: dog (.93) man (.73) mother (.13) father (.13)
is: is (1.00)
to: to (1.00) where (.04) a (.01)
a: a (1.00)
#: kennel (.89) house (.73) daughter (.14) son (.14) a (.01) lives (.01)

Figure 27: Working Memory following Relational Resolution in the Analogical Inference Task

While the simple mechanisms proposed in the SP model are clearly insufficient to model the broad spectrum of analogical reasoning results, they may prove useful in describing the sort of analogical inference that happens routinely and automatically during the normal course of text comprehension.

In this section, three types of inference that can be implemented by the SP model have been demonstrated. Automatic, controlled-logical and controlled-analogical inference all occur as a consequence of the basic processing assumptions of the model with minimal requirements on the content of the input corpus. In particular, the distinction between automatic and controlled inference is explained by the fact that automatic inference is not

a process at all, but rather a coincidence of relational representation that embodies an inference.

This is the last of the small demonstrations of the model. In the next section, we investigate how the model scales when exposed to a large corpus of text.

Applying the Model to Natural Corpora

In the previous section, each of the demonstrations was designed to be as small as possible in order to clearly portray the properties of the model. However, one of the key advantages of the SP model is that it can be applied to naturalistic text. Experience with models such as LSA (Landauer & Dumais 1997) and HAL (Lund & Burgess 1996) suggests that many interesting properties of distributional models only become apparent when they are applied to large corpora. In this section, the SP model is used to induce syntactic structure and to assign syntactic classes to multi-class words within sentential context using a large corpus of naturally occurring text.

Inducing Syntactic Structure

In order to populate the substitution matrix of lexical memory one needs to collect statistics over a large corpus. Ideally, each sentence fragment would be used as a retrieval cue against all other fragments from the corpus and substitution probabilities would be calculated using the EM algorithm. Unfortunately, such a procedure is computationally expensive for large corpora where there may be tens of millions of fragments to be compared against each other.

By making a few assumptions, however, it possible to construct a fast approximation to the generic procedure. The key to the algorithm is to divide the fragments into equivalence classes such that each fragment need only be compared against those from the same equivalence class rather than the entire corpus. To do this we note that alignment will be dominated by the very high frequency words which will typically have high match probabilities. We define a fragment as a sequence of words bounded by very high frequency words (and the end of sentences boundaries) and assign fragments with the same word patterns to the same equivalence class. For instance, the sentence "THE book showed A picture OF THE author carrying A copy OF THE manuscript." would be divided into the following fragments:

1. [S] THE book showed A
2. A picture OF THE
3. OF THE author carrying A
4. A copy OF THE
5. OF THE manuscript [E]

where the very high frequency words (and end of sentence markers) are in capital letters. Note that the second and fourth fragments would be assigned to the same equivalence class as they contain the same pattern of words. As a consequence, it would be deduced that "picture" and "copy" may substitute for one another.

A second assumption is that change probabilities are much higher than indel probabilities, so that alignments that involve indels are unlikely to make a significant

contribution to the substitution probabilities. As a consequence, equivalence classes are restricted to contain fragments of the same length. So, "A picture OF THE" and "A small picture OF THE" would belong to different equivalence classes. While making this assumption will alter the results somewhat, it has the computational advantage that only words within the same slot within fragments can be substituted and the probability of retrieving a trace given the target fragment is determined purely by the number of matches.

The algorithm was run over the TASA corpus[13] using the 200 most frequent words as fragment boundaries. The corpus contains 1.2 million words, in 38000 documents and 750000 sentences. Substitution probabilities were collected for the 4000 most frequent words (note, however, that the 200 most frequent can never enter into substitutions and so in fact the substitution matrix is restricted to the 3800 subsequent words).

Table 3 presents a number of examples drawn from the substitution matrix to demonstrate different sorts of information that is extracted by the algorithm. Each row shows a word, the ten words most likely to substitute for the word in order of substitution probability and a label for the type of information captured in each list. The first ten examples show the sensitivity of the model to syntactic categories, while the remaining examples capture semantic categories.

Table 3: Substitution examples.

Word	Ten Words most likely to Substitute	Type of Information
band	group, kind, piece, amount, lot, set, variety, series, type, line	singular nouns
bands	amounts, groups, cells, pieces, patterns, natural, kinds, waves, hundreds, society	plural nouns
agree	want, believe, deal, play, try, talk, begin, feel, learn, live	present tense verbs
agreed	wanted, tried, decided, believed, learned, continued, seemed, started, refused, turned	past tense verbs
below	above, behind, across, among, against, near, along, inside, toward, within	positional adjectives
bottom	end, top, edge, beginning, center, surface, side, foot, middle, base	positional nouns
didn't	don't, couldn't, doesn't, am, wouldn't, wasn't, can't, felt, hadn't, shall	negation contractions
don't	didn't, can't, couldn't, am, won't, suppose, cannot, probably, shall, wouldn't	negation contractions
i'd	i've, he'd, i'll, it's, she'd, you'll, don't, you've, i'm, scientists	other contractions

[13] We thank the late Stephen Ivens and Touchstone Applied Science Associates (TASA) of Brewster, New York for providing this valuable resource. The corpus consists of representative random samples of text of all kinds read by students in each grade through first year of college in the United States.

Table 3 : Substitution examples (cont)

myself	himself, yourself, themselves, herself, possible, meeting, going, someone, memory, want	self-pronouns
kind	type, kinds, group, sort, form, amount, set, method, piece	quantities
mr	mrs, dr, miss, president, aunt, uncle, although, john, since, king	forms of address
australia	china, india, europe, power, canada, california, england, america, africa, mexico	countries
afternoon	morning, night, year, evening, summer, room, winter, week, early, late	times
april	november, june, july, march, october, january, pages, august, september, december	months
billy	mr, dad, everyone, bill, himself, waves, john, congress, danny, special	gender
blue	red, white, green, big, black, young, north, american, hard, dark	colors
brother	father, mother, eyes, wife, name, family, head, son, friends, friend	family relationships
chapter	section, book, unit, lesson, figure, case, area, country, fig, investigation	manuscript parts
diagram	picture, map, drawing, chart, book, pictures, bank, section, page, maps	manuscript parts
dinner	lunch, breakfast, night, last, supper, year, eyes, least, times, war	meals
hand	head, eyes, side, hands, mind, face, arms, father, arm, mouth	body parts
neck	head, face, eyes, father, shoulders, legs, feet, hands, shoulder, mother	body parts
hot	cold, warm, big, fast, hard, late, strong, fresh, deep, early	temperatures
nine	six, four, several, five, seven, eight, ten, least, twelve, twenty	numbers
hundred	thousand, million, dozen, billion, side, thing, short, days, hand, big	numerical terms
inches	feet, miles, percent, meters, weeks, degrees, hours, minutes, words, days	measurement terms
insects	animals, plants, fish, birds, organisms, mammals, cells, scientists, materials, products	organism types
neutrons	protons, electrons, waves, others, atoms, animals, services, plants, metal, rays	subatomic physical terms
river	sea, ocean, mountains, road, door, city, surface, floor, room, ground	physical landmark terms
west	north, south, east, ground, door, next, river, western, sun, morning	directions

Figure 28 shows a hierarchical cluster analysis performed on the first 75 words of the resultant matrix. The model seems to capture some important linguistic intuitions, with the major syntactic classes separated at the highest levels and further subdivision as one moves deeper into the tree.

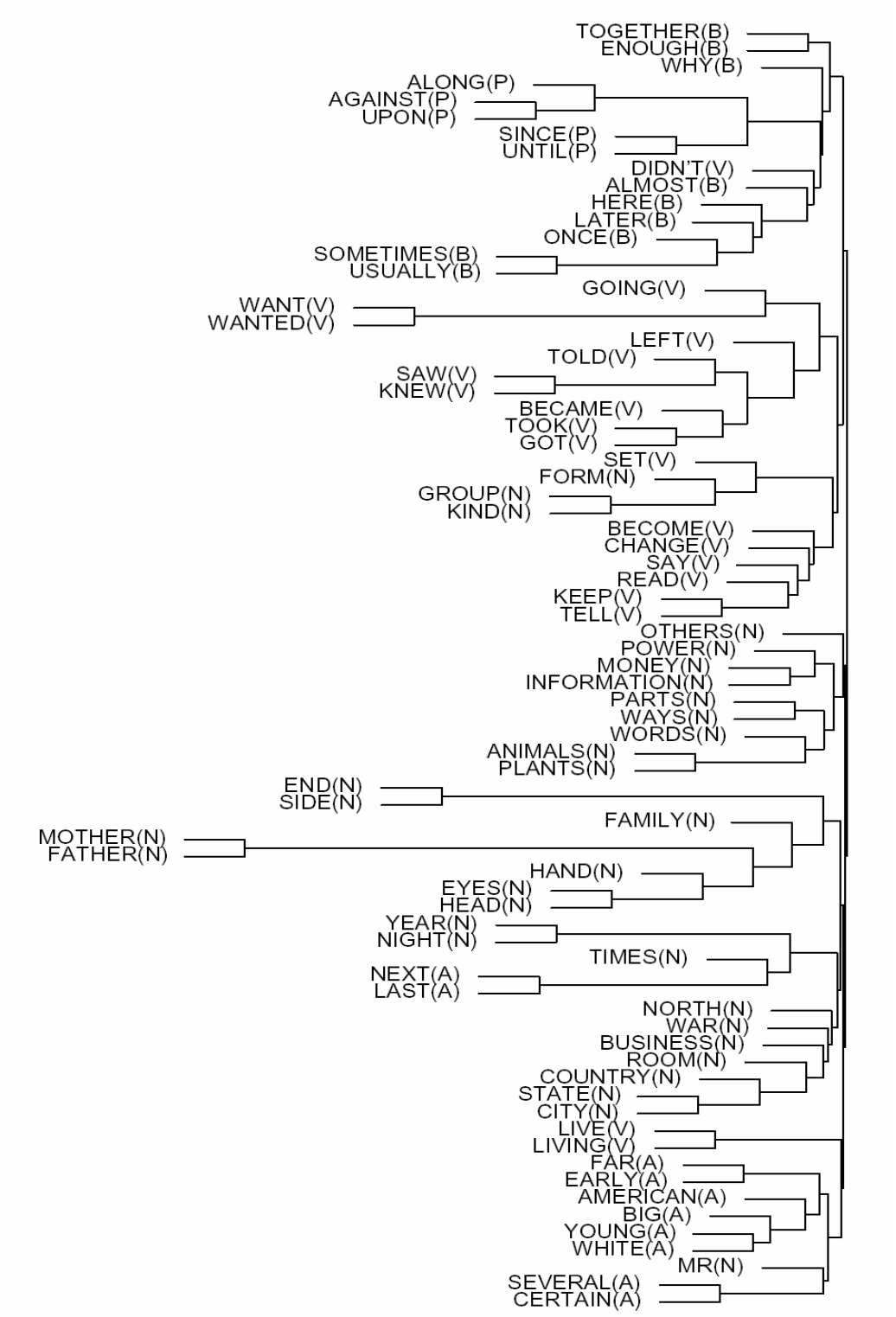

Figure 28: Hierarchical cluster analysis of the substitution probabilities of the 75 most frequent non-anchor words from the TASA corpus. N=Noun, V=Verb, A=Adjective, B=Adverb, P=Preposition.

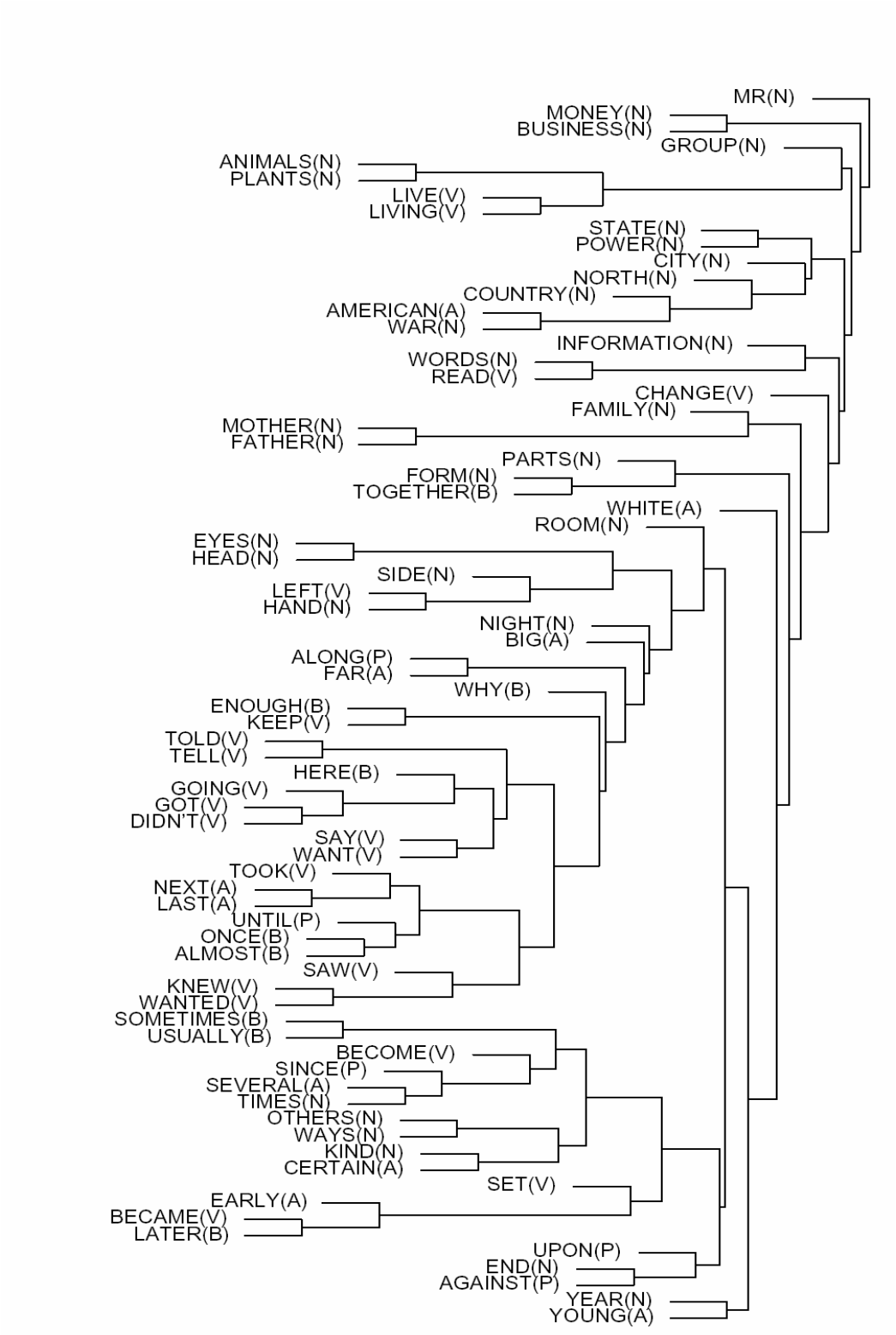

Figure 29: Hierarchical cluster analysis of the LSA cosines of the 75 most frequent non-anchor words from the TASA corpus. N=Noun, V=Verb, A=Adjective, B=Adverb, P=Preposition.

By contrast, Latent Semantic Analysis (LSA, Landauer & Dumais 1997) extracts some additional semantic relationships (e.g. INFORMATION, READ and WORDS) but does not capture the syntactic structure as well (see Figure 29).

Note that while Hierarchical Cluster Analysis (HCA) is a useful mechanism for visualizing the structure of the space, as a similarity model the tree architecture is too restrictive for this domain. As a consequence, some distortions are introduced. For instance, the word "SET" which can be either a verb or a noun, but which appears dominantly as a verb, has high substitution probabilities with "FORM", "GROUP" and "KIND" all of which are nouns. Consequently, HCA places these words within the verb branch of the tree rather than the noun branch. A less restrictive model, however, would allow "SET" to appear in both the verb and noun branches thus avoiding this problem.

Finally, as a quantitative test of the ability of the model to align words appropriately, syntactic categories from the WordNet database (Miller 1990) were used to determine how often substitution pairs shared a syntactic category. WordNet classifies each word as either a noun, a verb, an adjective, a satellite adjective or an adverb. Words from other classes are not included and some words are assigned to more than one category. For each word that appeared in the WordNet database, the syntactic category of the word with the largest substitution probability was extracted. On 92.6% of occasions these words shared a syntactic category with the word for which they could substitute. Furthermore, if the words with the five largest substitution probabilities are used this percentage drops only marginally to 91.5% and when the top ten words are used the percentage is 90.7%.

In addition to syntactic categories, WordNet also reports a more fine grained classification particularly for nouns and verbs. Table 4 shows the 45 categories in the WordNet classification (Miller 1990). A more stringent test of the model is the extent to which the aligned words share the WordNet categories. When this criterion is applied 68.2% of the highest probability substitutions, 64.2% of the five words with the highest substitution probabilities and 61.9% of the top ten words share a category.

Care must be taken in estimating chance performance to incorporate the degree of polysemony and the frequency of substitution candidates. To ensure an appropriate baseline the target words of the substitution matrix were permuted and the analysis was repeated. Using the permuted matrix the percentages of overlapping syntactic classes were 63.0%, 64.3% and 63.9% for the one, five and ten word cases, respectively. The percentages of overlapping WordNet classes were 29.0%, 28.5% and 28.5%. So in each case, the model is performing well above chance.

Table 4: WordNet categories

all adjective clusters	quantities and units of measure
relational adjectives (pertainyms)	relations between people or things or ideas
all adverbs	two and three dimensional shapes
unique beginners for nouns	stable states of affairs
acts or actions	substances
animals	time and temporal relations
man-made objects	grooming, dressing and bodily care
attributes of people and objects	size, temperature change, intensifying, etc.
body parts	thinking, judging, analyzing, doubting
cognitive processes and contents	telling, asking, ordering, singing
communicative processes and contents	fighting, athletic activities
natural events	eating and drinking
feelings and emotions	touching, hitting, tying, digging
foods and drinks	sewing, baking, painting, performing
groupings of people or objects	feeling
spatial position	walking, flying, swimming
goals	seeing, hearing, feeling
natural objects (not man-made)	buying, selling, owning
people	political and social activities and events
natural phenomena	being, having, spatial relations
plants	raining, snowing, thawing, thundering
possession and transfer of possession	participial adjectives
natural processes	

In general, it appears that this simple mechanism is capturing a great deal of structure inherent in the corpus. Note that these results are consistent with those found using other statistical models. For instance, Redington, Chater and Finch (1997) accumulated frequency counts of the words that appeared in the two positions immediately before and immediately after target words. They concatenated these vectors, used Spearman's rank correlation to create a similarity matrix and then applied hierarchical cluster analysis to define categories. Their results were similar, clearly demonstrating the ability to extract both syntactic and semantic information from corpora.

However, there is an important difference. The SP model assumes that similarities (substitution probabilities) are determined by the retrieval of entire fragments, so that the joint information carried by the trace is utilized. If accumulation across linguistic contexts occurs prior to the similarity calculation, as in the Redington et. al. (1997) model, the joint information does not play a role. While it is not clear that this characteristic of the SP model confers any advantage in the acquisition of structure, as we will see in the next section, it does allow the disambiguation of polysemous words within sentential context.

The Representation of Words in Sentential Context

A surprisingly large number of words belong to multiple syntactic categories. In the WordNet database about 7% of the words recorded belong to more than one syntactic class. Furthermore, high frequency words tend to be better candidates for polysemy, so by token the percentage is much higher. As a consequence, while it is useful to be able to determine the most likely category to which a word will belong, it is also necessary to explain how it is that people are able to assign the correct category to polysemous words when processing an utterance.

To test the SP model's ability to disambiguate syntactic class in context, 20 words that could be both nouns and verbs were chosen from the WordNet database. The database provides counts of the number of times words have appeared as different parts of speech in the WordNet corpora. For the majority of words one use dominates. The words used in this study were chosen so that the non-dominate meaning was recorded at least 3 times in WordNet. For each of the twenty words, six fragments were sampled from the TASA corpus, three in which the word was being used as a verb and three in which it was being used as a noun. Sampling consisted of searching sequentially from the beginning of the corpus for the keyword until three of each use were found. The fragments were then submitted to the full SP model with the match probabilities set to 100 times indel probabilities and change probabilities set to twice the indel probabilities. The alignment probabilities for verbs and for nouns were summed, with the highest probability recorded as the disambiguation for the word in that fragment.

As an example, consider the word "state" which can be either a verb or a noun. In the fragment "TO state BOTH THE" it is being used as a verb. When this fragment is submitted to the model the following fragments with their corresponding retrieval probabilities are returned:

0.049 TO state clearly THE
0.049 TO state clearly THE
0.049 TO state unequivocally THE
0.049 TO state ALL THE
0.049 TO satisfy BOTH THE
0.049 TO organize BOTH THE
0.049 TO learn BOTH THE
0.049 TO overcome BOTH THE
0.049 TO view BOTH THE
0.049 TO avoid BOTH THE
0.049 TO save BOTH THE
0.049 TO observe BOTH THE
0.049 TO consider BOTH THE
0.049 TO win BOTH THE
…

The alignment probabilities that result are:

TO: TO (0.999)
STATE: STATE (0.324) ADJUST (0.049) ORGANIZE (0.049) OVERCOME (0.049) SATISFY (0.049) OBSERVE (0.049) WIN (0.049) AVOID (0.049) SAVE (0.049) ADD (0.049) …
BOTH: BOTH (0.594) CLEARLY (0.099) UNEQUIVOCALLY (0.049) ALL (0.049) THE (0.003) STATE (0.003) MEASURE (0.002) …
THE: THE (0.997)

Considering the words that align with "state" we find that "add", "adjust", "avoid", "observe", "organize", "overcome", "satisfy", "save" and "win" are verbs according to the WordNet database. Adding their probabilities we have a total of 0.441. Now the words "save" and "win" are also recorded as nouns in the WordNet database, so the total probability for nouns is 0.098. In this case, the model has classified the word "state" as a verb which is consistent with the fragment in which it appears. Alternatively, we also present the word "state" in a noun context "OF ANY state IN THE".

Now the following fragments are retrieved:

0.070 OF ANY changes IN THE
0.070 OF ANY matter IN THE
0.070 OF ANY city IN THE
0.070 OF ANY country IN THE
0.070 OF ANY substance IN THE
0.070 OF ANY organization IN THE
0.070 OF ANY kind IN THE
0.070 OF ANY officer IN THE
0.070 OF ANY defects IN THE
0.070 OF ANY defects IN THE
0.070 OF THAT state IN THE
0.070 OF THE state IN THE
0.036 OF ANY state THE
…

and the alignment probabilities are:

OF: OF (0.999)
ANY: ANY (0.736) THAT (0.070) THE (0.070) THINE (0.001) …
STATE: STATE (0.176) DEFECTS (0.140) CHANGES (0.071) OFFICER (0.070) SUBSTANCE (0.070) ORGANIZATION (0.070) MATTER (0.070) KIND (0.070) COUNTRY (0.070) CITY (0.070)
IN: IN (0.962)
THE: THE (0.999)

Now only the word "officer" is recorded as a verb in the WordNet database, so the total verb probability is 0.07. However, the words "city", "country", "kind", "matter", "officer", "organization" and "substance" are classified as nouns so the total noun probability is 0.49. In this case, the model has classified the word as a noun.

Appendix H shows each of the fragments and the sum of the probabilities for the verb and noun categorizations. On 79% of occasions the model chose words in a way consistent with its context. On 8% of occasions the verb and noun probabilities were equal and on 13% of occasions the model incorrectly aligns more words from the incorrect category.

In general, the model seems to be doing a reasonable job of disambiguating the syntactic class of a word in sentential context. Furthermore, it may be that these statistics are conservative, as many words although they can appear in more than one category are rarely seen in one of the categories. While the target words were chosen to avoid this, the classification of the aligned words was not influenced by frequency counts. Consequently, words like "officer" which are predominately nouns but can be thought of as verbs, contribute to the total probabilities of both verb and noun categories equally.

In the assessment of the model's ability to perform appropriately both in creating lexical memory (in the last section) and in interpreting words in context (in this section) we have used the notion of syntactic categories. Note, however, that the notion of category has only been used for assessment purposes. Syntactic categories are never reified in the model, not even in the sense of a region in a vector space as is the case in connectionist models such as the Simple Recurrent Network (SRN, Elman 1990). It would seem to be an open question, whether the reification of syntactic and other linguistic categories is necessary to explain human performance.

Discussion

The primary purpose of a model such as the SP model is to highlight across domain similarities that might not otherwise have been apparent when working on separate models from different domains. In the discussion, I outline three specific examples.

Firstly, importing the gap probabilities from the string edit literature, where the first indel in a sequence is given a lower probability than subsequent indels, has two important consequences. In sentence processing it ensures that simple sentences will tend to match the start and end of more complex sentences containing embedded clauses. This property is critical to the models ability to correctly align longer sentences. However, it is also this property that leads to the U shaped serial position curve in serial recall. Gap probabilities result in less certainty in alignment towards the middle of serial lists. This uncertainty leads to role vectors in the relational representation that are more blurred and hence to the serial position curve and the patterns of inter-list and intra-lists intrusions characteristic of human performance.

Secondly, the SP model assumes that during sentence processing paradigmatic associations form between the input items and the items from previous sequential fragments that align with them. These paradigmatic associations are both the mechanism by which thematic roles are bound to their constituents in the course of sentence processing and the mechanism by which position cues are bound to items during serial recall.

Finally, the SP model argues that people's ability to recognize and make use of the relational equivalence of different surface forms of a given proposition (e.g. active, passive, interrogative) is a special case of a much broader ability to perform implicit inference. So the ability to extract a common relational representation for "John loves Mary" and "Mary is loved by John" is similar to extracting a common relational form for "Microsoft bought the widget from IBM" and "IBM sold the widget to Microsoft", or even "Microsoft owns the widget". The capacity to extract this kind of information is a primary contribution of the model.

Conclusions

Much remains to be done to complete the SP model. A theory of sublexical effects (c.f. LEX, Kwantes & Mewhort 1999), a mechanism for encoding and utilizing the macrostructure of a text or discourse (Kintsch 1998) and a more comprehensive and principled treatment of context are just some of the components that are required. Most critically it must be determined whether the input necessary to support each of the tasks presented in the paper actually does appear in naturally occurring speech and prose.

However, the model also accomplishes much. At the outset, I stated three fundamental issues that a unified theory of verbal cognition must address. Firstly, it must specify how content within the system is represented. In the SP model, there are three types of representation. Lexical memory contains paradigmatic associations between words across experience and captures syntactic/semantic class structure. Sequential memory contains traces consisting of the syntagmatic associations within sentences and captures syntactic structural regularities. Relational memory contains traces consisting of the paradigmatic associations within sentences which capture propositional information independent of surface structure. The demonstrations within the paper show how these simple representational schemes are sufficient to support a number of cognitive tasks, including sentence processing, categorization, recall and inference.

Secondly, a unified theory must outline a control architecture. For the SP model, control is embedded in linguistic fragments rather than in an abstract production system architecture. What should occur next is determined by analogy to what has occurred next in the fragments that comprise sequential memory – cognitive control is literally speech in the head. As was illustrated in the demonstrations of categorization and rating, linguistic context controls the type of information that will be returned in response to any given task – that is, the model creates on the fly a task virtual machine.

Finally, a unified theory must explain how content and control are acquired. The SP model learns in two different ways. To compile lexical memory the iterative EM algorithm is employed to extract corpus statistics. To compile sequential and relational memory, traces are simply added as they become available. It is in this final area that the SP model makes its most important contribution. Existing symbolic models (SOAR Newell 1990; ACT-R Anderson & Lebiere 1998) give a comprehensive account of the control architecture of cognition, but have struggled to provide a convincing explanation of the acquisition of either content or control, while distributional models (LSA, Landauer & Dumais 1997; HAL, Lund & Burgess 1996) have provided useful content learning mechanisms, but have not be able to capture propositional information and have not addressed the acquisition of control. The SP model addresses precisely these issues.

Ultimately, the test of a unified theory (Newell 1990, Anderson & Lebiere 1998) such as the SP model is its ability to provide across domain insights. In this paper, it has been demonstrated how phenomena such as sentence processing, semantic categorization and rating, short term serial recall and automatic and controlled inference can be unified in a concise, mathematically-principled way that scales to naturally occurring human-sized knowledge bases.

References

Allison, L., Powell, D. & Dix, T. I. (1999) Compression and approximate matching. The Computer Journal, **42**:1, 1-10.

Allison, L., Wallace, C. S. & Yee, C. N. (1992) Finite-state models in the alignment of macromolecules. J. Molec. Evol., **35**, 77-89.

Allison, L. & Yee, C. N. (1990).

Anderson, J. (1983). The Architecture of Cognition. Cambridge, MA: Harvard University Press.

Anderson, J. R. (1993). *Rules of the mind*. Hillsdale, NJ: Erlbaum.

Anderson, J. R. & Lebiere, C. (1998). *The atomic components of thought.* Mahwah, NJ: Erlbaum.

Baird, B., Troyer, T., Eeckman, F, H. (1992). Synchronization and Grammatical Inference in an Oscillating Elman Net. Neural Information Processing Systems, 236-243.

Bever, T. G. (1970). The cognitive basis for linguistic structures. In Hayes, J. R. (Ed.) Cognition and the development of language, pp 279-352. Wiley, New York.

Bransford, J. D., Barclay, J. R., Franks, J. J. (1972). Sentnece memory: A constructive versus interpretive approach. Cognitive Psychology, 3, 193-209.

Brown, R. & Berko, J. (1960). Word association and the acquisition of grammar. *Child Development, 31*, 1-14.

Budiu, R. (2001). The Role of Background Knowledge in Sentence Processing. Doctoral Dissertation, School of Computer Science, Carnegie Mellon University. (Available as Technical Report No. CMU-CS-O1-148).

Chomsky, N. (1957). Syntactic Structures. The Hague: Mouton.

Christiansen, M. H., & Chater, N. (1999). Toward a connectionist model of recursion in human linguistic performance. *Cognitive Science, 23*, 157–206.

Collins, A. & Loftus, E. (1975). A spreading activation theory of semantic memory. Psychological Review, 82, 407-428.

Collins, A. & Quillian M. R., (1969). Retrieval time from semantic memory, Journal of verbal learning and verbal memory. 8, 240-247.

Dempster, ,A. P., Laird, N. M. & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm (with discussion). J. of the Royal Statistical Society series B, 39, 1-38.

Dennis, S. (2001a). Generating distributed relational representations of sentences for information retrieval: Beyond the "Bag of Words". Twelfth Annual Winter Conference on Discourse, Text & Cognition, Jackson Hole, Wyoming.

Dennis, S. (2001b). The Syntagmatic Paradigmatic Model of Sentence Processing. Twenty-Sixth Annual Interdisciplinary Conference, Jackson Hole, Wyoming.

Elman, J. L. (1993). Learning and development in neural networks: The importance of starting small. *Cognition, 48*, 71-99.

Elman, J. L. (1998). Connectionism: Whence, Whither, and Why? Proceedings of the Twentieth Annual Meeting of the Cognitive Science Society. Lawrence Erlbaum Associates.

Erickson, T., & Matteson, M. (1981). From words to meaning: A semantic illusion. Journal of Verbal Learning and Verbal Behavior, 20, 540-552.

Ericsson, K. A., & Kintsch, W. (1995). Long-term working memory. Psychological Review, 102, 211-245.

Ervin, S. M. (1961). Changes with age in the verbal determinants of word association. *American Journal of Psychology*, 74, 361-372.

Ervin-Tripp, S. M. (1970). Substitution, context, and association. In L. Postman and G. Keppel (Ed.), Norms of word association. New York: Academic Press, pp. 383-467.

Evans, T. (1964). A Heuristic Program to Solve Geometric-Analogy Problems. In: *Spring Joint Computer Conference*, vol. 25. Reprinted in: M. Fischler & O. Firschein (eds.) *Readings in Computer Vision.* Morgan Kaufman Publ., 1987

Falkenhainer, B., Forbus, K. D., & Gentner, D. (1989). The structure-mapping engine: Algorithm and examples. *Artificial Intelligence, 41,* 1-63.

Fodor, J. A., Bever, T. G. & Garrett, M. F. (1974). *The psychology of language*. New York: McGraw Hill.

Fodor, J., & Pylyshyn, Z. (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition, 28*, 3-71.

Frazier, L., & Clifton, C. (1996). *Construal*. Cambridge, MA: MIT Press.

French, R. (1995). *The Subtlety of Sameness: A theory and computer model of analogy-making*. Cambridge, MA: MIT Press.

Gentner, D. (1983). Structure-Mapping: A Theoretical Framework for Analogy, *Cognitive Science*, 7(2), 155-170

Gibson, E. (1998). Linguistic complexity: Locality of sequential dependencies. *Cognition, 68*, 1-76.

Goldinger, S. D. (1998). Echoes of echoes? An episodic theory of lexical access. *Psychological Review, 105*(2), 251-279.

Gotoh, O. (1982) An improved algorithm for matching biological sequences. J. Molec. Biol., **162**, 705-708.

Hadley, R. F. (1994). *Systematicity in connectionist language learning*. Mind and Language, 9(3):247—272.

Halford, G., Wilson, W., Guo, J., Gayler, R., Wiles, J., Stewart, J. (1994). Connectionist implications for processing capacity limitations in analogies. In: K. Holyoak & J. Barnden (eds.) *Advances in connectionist and neural computation theory, vol. 2, Analogical Connections,* pp. 363--415. Norwood, NJ: Ablex.

Hintzman, D. L. (1988). Judgments of frequency and recognition memory in a multiple-trace memory model. *Psychological Review, 95 (4),* 528-551.

Hitch, G. J., Burgess, N., Towse, J. N. & Culpin, V. (1996). Temporal grouping effects in immediate recall: A working memory analysis. Quarterly Journal of Experimental Psychology, 49A, 116-139.

Hofstadter, D. and the FARG (1995). *Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought.* NY: Basic Books.

Holyoak K. & Thagard P. (1989). Analogical Mapping by Constraint Satisfaction. *Cognitive Science*, 13, 295-355.

Hummel, J. & Holyoak, K. (1997). Distributed Representations of Structure: A Theory of Analogical Access and Mapping. Psychological Review, 104, 427-466.

Kintsch, W. (1993). Information accretion and reduction in text processing: Inferences. Discourse Processes, 16, 193-202.
Kintsch, W. (1998). Comprehension: A paradigm for cognition. New York, NY: Cambridge University Press.
Kokinov, B. & French, R. M. (to appear). Computational models of analogy-making. *The Macmillan Encyclopedia of the Cogntive Sciences*.
Kwantes, P.J. & Mewhort, D. J. K. (1999). Modeling lexical decision and word naming as a retrieval process. *Canadian Journal of Experimental Psychology, 53,* 306-315.
Landauer, T. K., & Dumais, S. T. (1997). A solution to Plato's problem: The Latent Semantic Analysis theory of acquisition, induction and representation of knowledge. Psychological Review, 105, 221-240.
Levenshtein, V. I. (1965) Binary codes capable of correcting deletions, insertions and reversals. Dokl. Akad. Nauk SSSR, 163, 845-848 (Engl. Transl. (1966) Sov. Phys.-Dokl.,10,707-710).
Lund, K., & Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instrumentation, and Computers, 28,* 203-208.
McKoon, G., & Ratcliff, R. (1998). Memory-based language processing: Psycholinguistic research in the 1990s. *Annual Review of Psychology, 49*, 25-42.
McNeill, D. (1966) A study of word association. *Journal of Verbal Learning and Verbal Behavior, 2*, 250-262.
Marcus, G. F. (1998). Can connectionism save constructivism? *Cognition, 66*, 153-182.
Marks, L. E. (1968). Scaling of grammaticalness of self-embedded English sentences. Journal of Verbal Learning and Verbal Behavior, 7, 965-967.

Miller, G. (1990). The place of language in scientific psychology. Psychological Science, 1(1), 7-14.
Miller, G. A., ed. (1990). WordNet: An On-Line Lexical Database. *International Journal of Lexicography* 3(4).
Miller, G. A. & Isard, S. (1964). Free recall of self-embedded English sentences. Information and Control, 7, 292-303.
Miller, G. A., & Selfridge, J. A. (1950). Verbal context and the recall of meaningful material. American Journal of Psychology, 63, 176-185.
Mitchell, M. (1993). Analogy-making as perception: A computer model. Cambridge, MA: MIT Press.
Nakisa, R. C., & Plunkett, K. (1998). Evolution for rapidly learned representation for speech. *Language & Cognitive Processes, 13*(2/3), 105-127.
Navarro, D. (2002). Representing Stimulus Similarity. Unpublished doctoral dissertation. University of Adelaide, Australia.
Needleman, S. B. & Wunsch, C. D. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins, J. Molec. Biol., **48**, 443-453.
Newell, A. (1990). Unified Theories of Cognition. Cambridge, MA: Harvard University Press.
Phillips, S. (2000). Constituent similarity and systematicity: The limits of first-order connectionism. Connection Science, 12(1), 45-63.

Reder, L., & Kusbit, G. (1991). Locus of the Moses illusion: Imperfect encoding, retrieval, or match? Jornal of Memory and language, 30, 385-406.

Rumelhart, D. E., & Abrahamson, A. A. (1973). A model for analogical reasoning. Cognitive Psychology, 5, 1-28.

Sankoff, D. & Kruskal, J. B., eds (1983).Time warps, string edits and macromolecules: the theory and practice of sequence comparison. Addison Wesley.

Scha, R., Bod, R., & Sima'an, K. (1999). A memory-based model of sequential analysis: Data oriented parsing. *Journal of Experimental and Theoretical Artificial Intelligence*, 11, 409-440.

Sellers, P. H. (1974). An algorithm for the distance between two finite sequences. J. Combinatorial Theory,16,253-258.

Shiffrin, R. M. & Steyvers, M. (199 ?). <REM paper>

Smith, J. D., & Minda, J. P. (1998). Prototypes in the mist: The early epochs of category learning. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 24(6)*, 1411-1436.

Smith, E. E., Shoben, E. J. & Rips, L. J. (1974). Structure and process in semantic memory: a featural model for semantic decisions. Psychological Review, 81, 214-241.

Tabor, W., & Tanenhaus, M. K. (1999). Dynamical models of sentence processing. *Cognitive Science, 23*(4), 491-515.Luce, P. A., & Lyons, E. A. (1998). Specificity of memory representations for spoken words. *Memory and Cognition, 26*(4), 708-715.

Tversky, A. (1977). Features of similarity. *Psychological Review 84* (4), 327–352.

van Oostendorp, H., & de Mul, S. (1990). Moses beats Adam: A semantic relatedness effect on a semantic illusion. Acta Psychologica, 74, 35-46.

van Oostendorp, H., & Kok, I. (1990). Failing to notice errors in sentences. Language and cognitive processes, 5, 105-113.

Waterman, M. S., Smith, T. F. & Beyer, W. A. (1976). Some biological sequence metrics. Advan. Math. 20, 367-387.

Waterman, M. S. (1984). General methods of sequence comparison. Bulletin of Mathematical Biology. 46(4), 473-500.

Wiles, J., & Bloesch, A. (1992). Operators and curried functions: Training and analysis of simple recurrent networks. In J. E. Moody, S. J. Hanson, & R. P. Lippmann (Eds.), Advances in Neural Information Processing Systems 4, San Mateo, CA: Morgan Kaufmann.

Wiles, J. & Elman, J. (1995). Learning to count without a counter: A case study of dynamics and activation landscapes in recurrent networks. In Proceedings of the Seventeenth Annual Meeting of the Cognitive Science Society (pp. 482-487). Hillsdale, N.J: Lawrence Erlbaum Associates.

Wilson, W., Halford, G., Gray, B., Phillips, S. (2001). The STAR-2 Model for Mapping Hierarchically Structured Analogs. In: D. Gentner, K. Holyoak, B. Kokinov (eds.) *The Analogical Mind.* Cambridge, MA: MIT Press.

Williams, J. N. (1999). Memory, attention, and inductive learning. *Studies in Second Language Acquisition, 21*, 1-48.

Appendix A: Numerical Example of the Sequential Memory Retrieval Component

To consolidate an understanding of how string edit theory is applied to determining if sentences are related consider the case of matching the target sentence "John loves Mary" against the sequential trace "Bert loves Ellen".

To begin we must assume edit probabilities. Suppose that the following probabilities apply:

$$\begin{aligned} P(<x,x>) &= 0.1081 \\ P(<x,y>) &= 0.0032 \\ P(<x,->) &= 0.0216 \\ P(<-,y>) &= 0.005 \end{aligned} \tag{20}$$

We can now proceed with calculating the probabilities of the alignments. For instance, the probability of the following alignment:

```
John   -      loves   Mary                    A7
|      |      |       |
-      Bert   loves   Ellen
```

is

$$\begin{aligned} P(a_p) &= \prod_r P(e_r) \\ &= P(<John,->)P(<-,Bert>) \\ &\quad P(<loves,loves>)P(<Mary,Ellen>) \\ &= 0.0216*0.005*0.1081*0.0032 \\ &= 0.000000038 \end{aligned} \tag{21}$$

Table 5 shows each of the alignments of "John loves Mary" and "Bert loves Ellen" and their probabilities.

Table 5: Possible alignments for "Bert loves Ellen" and "John loves Mary" and the probabilities of these alignments. B=Bert, L=loves, E=Ellen, J=John, M=Mary, - is the empty word. The alignments shaded in the bottom right hand corner are those that contain a change between "John" and "Bert".

Align	Probability	Align	Probability	Align	Probability
---JLM BLE---	0.000000000001	-JL-M BL-E-	0.000000000038	JL-M- --BLE	0.000000000038
--J-LM BL-E--	0.000000000001	-JLM- BL--E	0.000000000038	JLM--- ---BLE	0.000000000001
--JL-M BL--E-	0.000000000001	-JLM BL-E	0.000000001137	JLM-- --BLE	0.000000000038
--JLM- BL---E	0.000000000001	-JLM BLE-	0.000000001137	JL--M -BLE-	0.000000000038
--JLM BL--E	0.000000000038	J---LM -BLE--	0.000000000001	JL-M- -BL-E	0.000000000038
--JLM BL-E-	0.000000000038	J--L-M -BL-E-	0.000000000001	JL-M -BLE	0.000000001137
--JLM BLE--	0.000000000038	J--LM- -BL--E	0.000000000001	JLM-- -B-LE	0.000000000038
-J--LM B-LE--	0.000000000001	J--LM -BL-E	0.000000000038	JLM- -BLE	0.000000001137
-J-L-M B-L-E-	0.000000000001	J--LM -BLE-	0.000000000038	J--LM BLE--	0.000000000038
-J-LM- B-L--E	0.000000000001	J-L--M -B-LE-	0.000000000001	J-L-M BL-E-	0.000000000038
-J-LM B-L-E	0.000000000038	J-L-M- -B-L-E	0.000000000001	J-LM- BL--E	0.000000000038
-J-LM B-LE-	0.000000000038	J-L-M -B-LE	0.000000000038	J-LM BL-E	0.000000001137
-JL--M B--LE-	0.000000000001	J-LM-- -B--LE	0.000000000001	J-LM BLE-	0.000000001137
-JL-M- B--L-E	0.000000000001	J-LM- -B-LE	0.000000000038	JL--M B-LE-	0.000000000038
-JL-M B--LE	0.000000000038	J-L-M -BLE-	0.000000001263	JL-M- B-L-E	0.000000000038
-JLM-- B---LE	0.000000000001	J-LM- -BL-E	0.000000001263	JL-M B-LE	0.000000001137
-JLM- B--LE	0.000000000038	J-LM -BLE	0.000000037905	JLM-- B--LE	0.000000000038
-JL-M B-LE-	0.000000001263	JL---M --BLE-	0.000000000001	JLM- B-LE	0.000000001137
-JLM- B-L-E	0.000000001263	JL--M- --BL-E	0.000000000001	JL-M BLE-	0.000000037905
-JLM B-LE	0.000000037905	JL--M --BLE	0.000000000038	JLM- BL-E	0.000000037905
-J-LM BLE--	0.000000000038	JL-M-- --B-LE	0.000000000001	JLM BLE	0.000001137149

Note that the theory captures the intuition that an alignment like:

John	loves	Mary		A8
|	|	|		
Bert	loves	Ellen	P = 0.000001137149	

is more probable than an alignment like:

John	-	loves	-	Mary	-		A9
|	|	|	|	|	|		
-	Bert	-	loves	_	Ellen	P = 0.000000000001	

To calculate the probability of A_k we add the probabilities of each of the alignments:

$$P(A_k) = \sum_{a_p \in A_k} P(a_p) \quad (22)$$
$$= 0.0000013$$

While this probability may seem low for two sentences that are obviously related to each other, this probability will be normalized against the probabilities of the other traces in memory, so what is critical is that traces that are related to the target have higher probabilities than those that are not related.

Appendix B: A Numerical Example of the Sequential Substitution Component

As an example, we calculate the probability that "John" substitutes for "Bert" in the sentence "John loves Mary". The alignments in which "John" substitutes for "Bert" are indicated by the grey cells in Table 5.

To determine the probability of the substitution we add the probabilities of these alignments and divide by the total probability of all alignments:

$$P(< S_{kj}, T_i > | S_k \mapsto T, T) = \frac{\sum_{a_p \in A, <S_{kj}, T_i> \in a_p} P(a_p)}{P(A_k)} \quad (23)$$

$$= \frac{0.000001218}{0.000001304}$$

$$= 0.93$$

So the algorithm correctly deduces that "John" is a paradigmatic associate of "Bert" within the context of the sentence "John loves Mary".

Appendix C: The Dynamic Programming Algorithm

The dynamic programming algorithm is based on the observation that many alignments have common subsequences of edit operations and that it is unnecessary to recalculate the probabilities of these subsequences (Needleman & Wunsch 1970, Allison & Yee 1990).

Consider matching “John loves Mary” and “Bert loves Ellen”. If our first edit operation is a change of “Bert” for “John” then we have matched the “John” and “Bert” components of the sentences, and what remains to be matched is “loves Mary” and “loves Ellen”. Similarly, if an alignment begins with a deletion of “John” followed by an insertion of “Bert” or with an insertion of “Bert” followed by a deletion of “John” we will also have matched the “John” and “Bert” components of the sentences and will be left with “loves Mary” and “loves Ellen” to match. Furthermore, the total probability of matching “John” against “Bert” does not depend on which sequence of edit operations occur subsequently so we need calculate this value only once.

Table 6 shows all of the partial match results that can occur. Alignments can be traced through this matrix by starting in the top left hand corner (where none of the words from either sentence have been aligned) and moving toward the bottom right hand corner (where all of the words from both sentences have been aligned). A move to the right corresponds to a deletion, downwards to an insertion and on the diagonal to either a match or a change.

For example, the alignment:

```
John   -      loves   Mary                    A10
|      |      |       |
-      Bert   loves   Ellen
```

would involve starting in the top right hand corner, moving right (delete John), then down (insert Bert) and then diagonally (match loves) and then diagonally again (change Mary for Ellen).

Table 6: Matrix of partial alignments for "Bert loves Ellen" and "John loves Mary". B=Bert, L=loves, E=Ellen, J=John, M=Mary. The dashed line represents alignment A8 (see text).

	J	L	M
:	J:	JL:	JLM:
B			
:B	J:B	JL:B	JLM:B
L			
:BL	J:BL	JL:BL	JLM:BL
E			
:BLE	J:BLE	JL:BLE	JLM:BLE

Allison & Yee (1990) introduced the following probabilistic version of the dynamic programming algorithm first proposed by Needleman & Wunsch (1970) to calculate the probability of each of the partial alignments:

$$
\begin{aligned}
M[0,0] &= 1 \\
M[i,0] &= M[i-1,0]P(<-,T_{i-1}>) \\
M[0,j] &= M[0,j-1]P(<S_{k(j-1)},->) \\
M[i,j] &= M[i-1,j-1]P(<S_{k(j-1)},T_{i-1}>) + M[i-1,j]P(<-,T_{i-1}>) + M[i,j-1]P(<S_{k(j-1)},->)
\end{aligned}
\tag{24}
$$

On completion of the algorithm $P(A_k) = M[|T|,|S_k|]$. Note that there are only $(|T|+1)(|S_k|+1)$ cells in the matrix (where $|T|$ is the length of the target sentence and $|S_k|$ is the length of the sequential trace sentence). Because the value of each cell can be calculated in constant time, $P(A_k)$ can be calculated in O($|T||S_k|$) time (and space). So we now have an efficient means of calculating the memory component.

To calculate the substitution component the probabilities of all alignments that involve the given substitution must be added. This value can be determined using the DP algorithm by multiplying the probability of the partial match preceding the substitution by the probability of the substitution and by the total probability of the partial match following the substitution (see Table 7). M contains the probabilities of the partial matches preceding each substitution and the probabilities of the substitutions are known. To calculate the probability of the partial matches following a substitution each of the sentences is reversed and the DP algorithm applied as above. The matrix created in this way is denoted M'.

Table 7: The probability of alignments that pass through the substitution <loves,loves> is the probability of the partial match of "John" and "Bert" by the probability of <loves,loves>by the probability of the partial match of "Mary" and "Ellen".

	Loves	
P(J:B)		
Loves	$P(< loves, loves >)$	
		P(M:E)

$$P(< S_{kj}, T_i >| S_k \mapsto T, T) = \frac{\sum_{a_p \in A} P(a_p)}{\frac{<S_{kj}, T_i> \in a_p}{P(A_k)}} = \frac{M[i-1, j-1]P(< S_{kj}, T_i >)M'[i, j]}{P(A_k)} \quad (25)$$

Note that this algorithm is still O($|T||S_k|$) time (and space), so we have an efficient means with which to calculate the substitution component.

A Numerical Example using the DP Algorithm

Both the memory and substitution components for the small example used previously can now be calculated using the DP algorithm. The matrix M would look as in Table 8.

Table 8: The matrix M for the example.

	J		L		M	
1.000000000		0.021621622		0.000467495		0.000010108
B						
0.005000000		0.003459459		0.000147261		0.000004751
L						
0.000025000		0.000034054		0.000375468		0.000008620
E						
0.000000125		0.000000254		0.000001993		0.000001304

Note that the value in the bottom right hand corner is $P(A_k)$ and is that same as that calculated by considering all alignments separately. Table 9 shows M' for the example.

Table 9: The matrix M' for the example.

	J		L		M	
	0.000001304		0.000001993		0.000000254	0.000000125
B						
	0.000008620		0.000375468		0.000034054	0.000025000
L						
	0.000004751		0.000147261		0.003459459	0.005000000
E						
	0.000010108		0.000467495		0.021621622	1.000000000

Now we can calculate the probability that "John" substitutes for "Bert" given that $S_k \mapsto R$:

$$P(< S_{kj}, T_i > | S_k \mapsto T, T) = \frac{\sum_{a_p \in A, <S_{kj}, T_i> \in a_p} P(a_p)}{P(A_k)} = \frac{M[i-1, j-1] P(< S_{kj}, T_i >) M'[i, j]}{P(A_k)} = \frac{1.0 * 0.00324 * 0.000375468}{0.000001304} = 0.93 \quad (26)$$

Again this is the same as the figure that we obtained by considering all alignments separately.

As an aside, it is useful in visualizing alignments to calculate $\frac{M[i, j] M'[i, j]}{P(A_k)}$ which is the total probability of all alignments that traverse each possible partial match state (Allison, Wallace & Yee 1992). Table 10 shows this matrix for the example.

Table 10: The probabilities of all alignments traversing each partial match state.

	J		L		M	
	1.000		0.033		0.000	0.000
B						
	0.033		0.996		0.004	0.000
L						
	0.000		0.004		0.996	0.033
E						
	0.000		0.000		0.033	1.000

In this case, the alignment that consists of three substitutions (i.e. “John” for “Bert”, “loves” for “loves” and “Mary” for “Ellen”) accounts for nearly all of the probability mass.

The probabilities that are calculated using these algorithms are often small. With long sentences there is a danger of numerical underflow. As a consequence, rather than deal directly with the probabilities it is common to calculate with the $-\log_2$ probabilities. Making this transformation eliminates underflow problems, allows the majority of calculations to be additions rather than multiplications (which can lead to better efficiency) and also has an interpretation in terms of information theory (Allison & Yee 1990; Allison, Wallace & Yee 1992).

Appendix D: The Dynamic Programming Algorithm with Gap Probabilities

Gotoh (1982) introduced an extension of the DP algorithm designed to find the minimum cost alignment using linear gap costs. The following modified version of Gotoh's algorithm uses probabilities rather than costs and calculates the total probability of all alignments rather than the minimum alignment cost (c.f. Allison, Wallace & Yee 1992).

In the standard DP algorithm, the probability of a partial match can be calculated directly from the cells above (deletion), above and left (match or change) and to the left (insertion). Now, however, the probability of insertions and deletions is dependent not only on the immediately preceding cells, but also those that precede these both vertically and horizontally back to the start of the deletion or insertion block.

As a consequence we must specify the probabilities of the edit operation conditional on the preceding operation. To avoid adding a great many degrees of freedom, we place a number of constraints on the new edit operation probability table:

	P(<x, x>)	P(<x, y>)	P(<x, ->)	P(<-, x>)	
After Insert	au	bu	c	ds	=1
After Delete	av	bv	cs	d	=1
After Change	a	b	c	d	=1
After Match	a	b	c	d	=1

where (c+a+b)/c>s>1, (d+a+b)/d > s > 1, u = (d - ds + a + b)/(a+b), and
v = (c-cs + a + b) / (a + b). Note these constraints ensure that the probabilities of the edit operations given each of the possible edit operations on the last iteration add to one. They also ensure that the impact of accounting for gap probabilities on a series of insertions is the same as the impact on a series of deletions.

The probability of a block of insertions is:

$$P(T[i..j]) = P(<-,T_i>_{\overline{insert}}) \prod_{l=i+1..j} P(<-,T_l>_{insert}) \tag{27}$$

where $P(<-,T_i>_{\overline{insert}})$ is the probability of an initial insertion (that is the probability of an insertion preceded by anything other than an insertion) and $P(<-,T_i>_{insert})$ is the probability of a continuing insertion. Similarly, the probability of a block of deletions is:

$$P(S_k[i..j]) = P(<S_{ki},->_{\overline{delete}}) \prod_{l=i+1..j} P(<S_{kl},->_{delete}) \tag{28}$$

Now the recursive function in the DP algorithm becomes:

$$M[i,j] = D[i,j] + V[i,j] + H[i,j] \tag{29}$$

where

$$V[i,j] = \sum_{1 \le l \le i} M[i-l, j] P(T[i-l+1..i]) \tag{30}$$

$$H[i,j] = \sum_{1 \le l \le j} M[i, j-l] P(S_k[i-l+1..j]) \tag{31}$$

and

$$\begin{aligned} D[i,j] \quad &= D[i-1, j-1] P(< S_{kj}, T_i >_{sub}) + \\ &V[i-1, j-1] P(< S_{kj}, T_i >_{insert}) + \\ &H[i-1, j-1] P(< S_{kj}, T_i >_{delete}) \end{aligned} \tag{32}$$

where $P(< S_{kj}, T_i >_{sub}) = P(< S_{kj}, T_i >_{match}) = P(< S_{kj}, T_i >_{change})$.

As it stands this algorithm, which is a modification of Waterman et. al. (1976), is $O(|T|^2 |S_k| + |T| |S_k|^2)$ time. However, Gotoh (1982) noted that V and H could also be calculated by the following recursion:

$$\begin{aligned} V[i,j] \quad &= \sum_{1 \le l \le i} M[i-l, j] P(T[i-l+1..i]) \\ &= M[i-1, j] P(T[i..i]) + \sum_{2 \le l \le i} M[i-l, j] P(T[i-l+1..i]) \\ &= M[i-1, j] P(< -, T_i >_{\overline{insert}}) + \sum_{1 \le l \le i-1} M[i-l-1, j] P(T[i-l..i]) \\ &= M[i-1, j] P(< -, T_i >_{\overline{insert}}) + \sum_{1 \le l \le i-1} M[i-l-1, j] P(T[i-l..i-1]) P(< -, T_i >_{insert}) \\ &= M[i-1, j] P(< -, T_i >_{\overline{insert}}) + V[i-1, j] P(< -, T_i >_{insert}) \end{aligned} \tag{33}$$

similarly:

$$H[i,j] = M[i, j-1] P(< S_{kj}, - >_{\overline{delete}}) + H[i, j-1] P(< S_{kj}, - >_{delete}) \tag{34}$$

so by keeping the V and H matrices the calculation of each cell of the DP algorithm can be done in constant time, and the time (and space) complexity of the algorithm remains at $O(|T| |S_k|)$.

To initialize the matrices it is assumed that an indel at the start of a sentence (either the target or trace) is an initial indel:

$$
\begin{aligned}
M[0,0] &= 1 \\
V[0,0] &= 0 \\
H[0,0] &= 0 \\
M[1,0] &= M[0,0]P(<-,T_0>_{\overline{insert}}) \\
V[1,0] &= V[0,0]P(<-,T_0>_{\overline{insert}}) \\
M[0,1] &= M[0,0]P(<S_{k0},->_{\overline{delete}}) \\
H[0,1] &= H[0,0]P(<S_{k0},->_{\overline{delete}}) \\
M[i,0] &= M[i-1,0]P(<-,T_{i-1}>_{insert}) \\
V[i,0] &= V[i-1,0]P(<-,T_{i-1}>_{insert}) \\
M[0,j] &= M[0,j-1]P(<S_{k(j-1)},->_{delete}) \\
H[0,j] &= H[0,j-1]P(<S_{k(j-1)},->_{delete})
\end{aligned}
\tag{35}
$$

Finally, the algorithm for calculating the substitution probabilities must be updated to account for the fact that the order in which indels are applied affects the alignment probability. As a consequence, M' cannot be calculated by reversing the target and trace strings and using common indel probabilities. Instead, the probabilities of a terminating partial match given the direction of entry to a cell (D', H' and V') must be calculated as follows.

$$
\begin{aligned}
H'[0,0] &= 1 \\
V'[0,0] &= 1 \\
D'[0,0] &= 1 \\
H'[|T|,j] &= P(<S_{kj},->_{\overline{delete}})V'[|T|,j+1] \\
V'[|T|,j] &= P(<S_{kj},->_{delete})V'[|T|,j+1] \\
D'[|T|,j] &= P(<S_{kj},->_{\overline{delete}})V'[|T|,j+1] \\
H'[i,|S_k|] &= P(<-,T_i>_{insert})H'[i+1,|S_k|] \\
V'[i,|S_k|] &= P(<-,T_i>_{\overline{insert}})H'[i+1,|S_k|] \\
D'[i,|S_k|] &= P(<-,T_i>_{\overline{insert}})H'[i+1,|S_k|] \\
H'[i,j] &= P(<-,T_j>_{insert})H'[i+1,j] + P(<S_{kj},->_{\overline{delete}})V'[i,j+1] + P(<S_{kj},T_i>_{insert})D'[i+1,j+1] \\
V'[i,j] &= P(<-,T_j>_{\overline{insert}})H'[i+1,j] + P(<S_{kj},->_{delete})V'[i,j+1] + P(<S_{kj},T_i>_{delete})D'[i+1,j+1] \\
D'[i,j] &= P(<-,T_j>_{\overline{insert}})H'[i+1,j] + P(<S_{kj},->_{\overline{delete}})V'[i,j+1] + P(<S_{kj},T_i>_{sub})D'[i+1,j+1]
\end{aligned}
\tag{36}
$$

To calculate the conditional probabilities the following formulae apply:

$$
\begin{aligned}
P(< S_{kj}, - >| S_k \mapsto T, T) &= \frac{V[i,j]V'[i,j]}{M[|T|,|S_k|]} \\
P(< -, T_i >| S_k \mapsto T, T) &= \frac{H[i,j]H'[i,j]}{M[|T|,|S_k|]} \\
P(< S_{kj}, T_i >| S_k \mapsto T, T) &= \frac{D[i,j]D'[i,j]}{M[|T|,|S_k|]}
\end{aligned}
\tag{37}
$$

That is, the probability of a given deletion is the probability of the partial match following it given that the cell was entered vertically by the probability of continuing from that cell given that you entered vertically divided by the total probability of all alignments (and similarly for the insertions and substitutions). To verify the above formulae it is a useful exercise to choose two short strings, enumerate all alignments, calculate their probabilities and ensure that the substitution probability calculated from the DP algorithm corresponds to the addition of those alignments containing the substitution.

Appendix E: Derivation of the EM equations for the SP Model

The EM algorithm involves two steps. In the expectation step, the expected value of the log likelihood of the data given the current parameters is calculated. That is, we define Q:

$$Q(\theta,\theta^t) = \int_{y\in Y} \log(P(T,y \mid \theta)P(y \mid T,\theta^t)d\theta \tag{38}$$

where y is an assignment of sequential assignments and alignments to all target sentences. T is the set of all target sentences and y_i is the assignment of a sequential trace and an alignment for the i[th] target sentence. Presuming the hidden variables of the generation sequence are compiled in the variable u, the model (not considering gap probabilities) is as follows:

1. Choose a sequential trace from memory according to a multinomial distribution with parameters $P(s_u = k \mid \theta) = \alpha_k$ where s_u is the index of the sequential trace associated with u. Consider the chosen trace, S_k, as a queue and reference the head of this queue as $h(S_k)$. If the queue is empty $h(S_k) = empty$.
2. Choose an edit operation e_{ul}, which can be a match, change, insertion or deletion with probability $P(e_{ul} = y \mid \theta) = \tau_y$.
    a. If the head word of the queue is empty ($h(S_k) = empty$) and e_{ul} is not an insertion then stop.
    b. If $e_{ul} = match$ then output $h(S_k)$, consume it from the queue and go to step 2.
    c. If $e_{ul} = change$ then choose a word to output, output the chosen word, consume $h(S_k)$ and go to step 2.
    d. If $e_{ul} = delete$ then consume $h(S_k)$ and go to step 2.
    e. If $e_{ul} = insert$ then choose a word to output, output the chosen word and go to step 2.

ϕ_{xx} will be used to denote the probability of word x being the subject of a match given that a match has occurred. ϕ_{xy} denotes the probability of word x substituting for word y given that a change has occurred. ϕ_{x-} denotes the probability that word x was deleted given that a deletion occurred and ϕ_{-x} denotes the probability that word x was inserted given that an insertion occurred.The description of the generative process provides definitions for the variables used below.

While there are an infinite $y \in Y$ only a finite subset have $P(y \mid T,\theta^t) > 0$, so:

$$
\begin{aligned}
Q(\theta,\theta^t) &= \sum_{y\in Y}\log\left[\prod_{i=1}^{N}P(T_i,y_i\mid\theta)\right]P(y\mid T,\theta^t) \\
&= \sum_{y\in Y}\sum_{i=1}^{N}\log P(T_i,y_i\mid\theta)\prod_{j-1}^{N}P(y_j\mid T_j,\theta^t) \\
&= \sum_{y_1}\cdots\sum_{y_N}\sum_{i=1}^{N}\log P(T_i,y_i\mid\theta)\prod_{j-1}^{N}P(y_j\mid T_j,\theta^t) \\
&= \sum_{y_1}\cdots\sum_{y_N}\sum_{i=1}^{N}\sum_{u}\delta_{y_i,u}\log P(T_i,y_i\mid\theta)\prod_{j-1}^{N}P(y_j\mid T_j,\theta^t) \\
&= \sum_{u}\sum_{i=1}^{N}\log P(T_i,u\mid\theta)\sum_{y_1}\cdots\sum_{y_N}\delta_{y_i,u}\prod_{j-1}^{N}P(y_j\mid T_j,\theta^t)
\end{aligned}
\tag{39}
$$

Now,

$$
\begin{aligned}
\sum_{y_1}\cdots\sum_{y_N}\delta_{y_i,u}\prod_{j=1}^{N}P(y_j\mid T_j,\theta^t) &= \sum_{y_1}\cdots\sum_{y_{i-1}}\sum_{y_{i+1}}\cdots\sum_{y_N}\delta_{y_i,u}\prod_{\substack{j=1\\ i\neq j}}^{N}P(y_j\mid T_j,\theta^t)P(y_i\mid T_i,\theta^t) \\
&= \sum_{y_1}\cdots\sum_{y_{i-1}}\sum_{y_{i+1}}\cdots\sum_{y_N}\prod_{\substack{j=1\\ i\neq j}}^{N}P(y_j\mid T_j,\theta^t)P(u\mid T_i,\theta^t) \\
&= \sum_{y_1}\cdots\sum_{y_{i-1}}\sum_{y_{i+1}}\cdots\sum_{y_N}P(y_N\mid T_N,\theta^t)\prod_{\substack{j=1\\ i\neq j}}^{N-1}P(y_j\mid T_j,\theta^t)P(u\mid T_i,\theta^t) \\
&= \sum_{y_1}\cdots\sum_{y_{i-1}}\sum_{y_{i+1}}\cdots\sum_{y_{N-1}}\prod_{\substack{j=1\\ i\neq j}}^{N-1}P(y_j\mid T_j,\theta^t)P(u\mid T_i,\theta^t) \\
&= P(u\mid T_i,\theta^t)
\end{aligned}
\tag{40}
$$

So,

$$
\begin{aligned}
Q(\theta,\theta^t) &= \sum_{u}\sum_{i=1}^{N}\log P(T_i,u\mid\theta)P(u\mid T_i,\theta^t) \\
&= \sum_{u}\sum_{i=1}^{N}\log P(u\mid\theta)P(u\mid T_i,\theta^t)
\end{aligned}
\tag{41}
$$

Note we can make this transformation as u is restricted to those that generate T_i. Now substituting in the probabilities that arise as a consequence of the generative process we get:

$$
\begin{aligned}
Q(\theta,\theta^t) &= \sum_{u}\sum_{i=1}^{N}\log\left[\alpha_k\prod_{l}\tau_h\phi_{gf}\right]P(u\mid T_i,\theta^t) \\
&= \sum_{u}\sum_{i=1}^{N}\log\alpha_k P(u\mid T_i,\theta^t)+\sum_{u}\sum_{i=1}^{N}\left[\sum_{l}\log\tau_h+\log\phi_{gf}\right]P(u\mid T_i,\theta^t)
\end{aligned}
\tag{42}
$$

This completes the derivation of the expectation of the log likelihood function. In the second step, we find the parameters θ which maximize Q. These will be used as the parameters for the next iteration of the algorithm:

$$\theta^{t+1} = \arg\max_{\theta} Q(\theta, \theta^t) \tag{43}$$

Note that the first term of Q is independent of the second and so we can optimize with respect to α_k considering only the first term. Note also that α_k constitute a multinomial distribution and hence must add to one. To capture this constraint an extra term with a the Lagrange coefficient, λ, will be introduced.

$$\begin{aligned}
&\frac{\partial}{\partial \alpha_k} \sum_u \sum_{i=1}^{N} \log \alpha_k P(u \mid T_i, \theta^t) + \lambda \left(\sum_{p=1}^{M} \alpha_p - 1 \right) && = 0 \\
&\frac{1}{\alpha_k} \sum_{\substack{u \\ s_u = k}} \sum_{i=1}^{N} P(u \mid T_i, \theta^t) + \lambda && = 0 \\
&\sum_{\substack{u \\ s_u = k}} \sum_{i=1}^{N} P(u \mid T_i, \theta^t) && = -\lambda \alpha_k \\
&\sum_{k=1}^{M} \sum_{\substack{u \\ s_u = k}} \sum_{i=1}^{N} P(u \mid T_i, \theta^t) && = \sum_{k=1}^{M} -\lambda \alpha_k \\
&-\sum_u \sum_{i=1}^{N} P(u \mid T_i, \theta^t) && = \lambda
\end{aligned} \tag{44}$$

Substituting λ back in and solving for α_k we get:

$$\alpha_k = \frac{\sum_{\substack{u \\ s_u = k}} \sum_{i=1}^{N} P(u \mid T_i, \theta^t)}{\sum_u \sum_{i=1}^{N} P(u \mid T_i, \theta^t)} \tag{45}$$

The update equations for the other parameters are derived in a similar manner. For ϕ, we note that $\sum_g \phi_{g-} = 1, \sum_f \phi_{-f} = 1 \sum_g \phi_{gg} = 1 \sum_{gf} \phi_{gf} = 1$ so the following equations can be adapted to each case:

$$\frac{\partial Q(\theta,\theta^t)+\lambda(\sum_f \phi_{gf}-1)}{\partial \phi_{gf}} = 0 \tag{46}$$

$$\frac{1}{\phi_{gf}}\sum_u\sum_{i=1}^{N}\left[\sum_{\substack{l \\ w1_{ul}=f \\ w2_{ul}=g}} P(u\mid T_i,\theta^t)\right]+\lambda = 0$$

$$\phi_{gf} = \frac{\sum_u\sum_{i=1}^{N}\left[\sum_{\substack{l \\ w1_{ul}=f \\ w2_{ul}=g}} P(u\mid T_i,\theta^t)\right]}{\sum_u\sum_{i=1}^{N}\left[\sum_{l} P(u\mid T_i,\theta^t)\right]} \tag{47}$$

For τ , we note that $\sum_h \tau_{hf}=1$, so:

$$\frac{\partial Q(\theta,\theta^t)+\lambda(\sum_h \tau_{hf}-1)}{\partial \tau_h} = 0 \tag{48}$$

$$\frac{1}{\tau_h}\sum_u\sum_{i=1}^{N}\sum_{\substack{l \\ e_{ul}=h}} P(u\mid T_i,\theta^t)+\lambda = 0$$

$$\tau_h = \frac{\sum_u\sum_{i=1}^{N}\sum_{\substack{l \\ e_{ul}=h}} P(u\mid T_i,\theta^t)}{\sum_u\sum_{i=1}^{N}\sum_{l} P(u\mid T_i,\theta^t)} \tag{49}$$

Finally, we must determine how to calculate $P(u \mid T_i, \theta^t)$. Let a_u be the alignment component of u.

$$
\begin{aligned}
p(u \mid T_i, \theta^t) &= P(s_u = k, a_u \mid T_i, \theta^t) \\
&= P(a_u \mid s_u = k, T_i, \theta^t) P(s_u = k \mid T_i, \theta^t) \\
&= P(a_u \mid s_u = k, T_i, \theta^t) \frac{P(T_i \mid s_u = k, \theta^t) P(s_u = k \mid \theta^t)}{P(T_i \mid \theta^t)} \\
&= P(a_u \mid s_u = k, T_i, \theta^t) \frac{P(T_i \mid s_u = k, \theta^t) P(s_u = k \mid \theta^t)}{\sum_{l=1}^{M} P(T_i \mid s_u = l, \theta^t) P(s_u = l \mid \theta^t)}
\end{aligned}
\tag{50}
$$

Note $P(s_u = k \mid \theta^t) = \alpha_k$ and $P(T_i \mid s_u = k, \theta^t)$ and the necessary sums of $P(a_u \mid s_u = k, T_i, \theta^t)$ can be derived from the DP algorithm.

Appendix F: A Numerical Example of the Relational Memory Retrieval Component

To illustrate the calculation of the memory component suppose that we are comparing RT and R_k as presented in Figure 3. Further, suppose that we employ the matrix of substitution probabilities provided in Appendix A.

$$
\begin{aligned}
P(RT_0 \mid R_{k0} \mapsto RT_0) &= P(<Ellen, Mary>)\begin{bmatrix} RT_{0Ellen}P(<Ellen, Jody>)R_{k0Jody} + RT_{0Ellen}P(<Ellen, Mary>)R_{k0Mary} + \\ RT_{0Jody}P(<Jody, Jody>)R_{k0Jody} + RT_{0Jody}P(<Jody, Mary>)R_{k0Mary} \end{bmatrix} \\
&= 0.0032 * \begin{bmatrix} 0.307 * 0.0032 * 0.320 + 0.307 * 0.0032 * 0.235 + \\ 0.307 * 0.1081 * 0.320 + 0.307 * 0.0032 * 0.235 \end{bmatrix} = 0.0000365 \\
P(RT_0 \mid R_{k1} \mapsto RT_0) &= P(<Bert, Mary>)\begin{bmatrix} RT_{0Ellen}P(<Ellen, Steve>)R_{k0Steve} + RT_{0Ellen}P(<Ellen, John>)R_{k0John} + \\ RT_{0Jody}P(<Jody, Steve>)R_{k0Steve} + RT_{0Jody}P(<Jody, John>)R_{k0John} \end{bmatrix} \\
&= 0.0032 * \begin{bmatrix} 0.307 * 0.0032 * 0.319 + 0.307 * 0.0032 * 0.226 + \\ 0.307 * 0.0032 * 0.319 + 0.307 * 0.0032 * 0.226 \end{bmatrix} = 0.0000034 \\
P(RT_1 \mid R_{k0} \mapsto RT_1) &= P(<Ellen, John>)\begin{bmatrix} RT_{0Bert}P(<Bert, Jody>)R_{k0Jody} + RT_{0Bert}P(<Bert, Mary>)R_{k0Mary} + \\ RT_{0Steve}P(<Steve, Jody>)R_{k0Jody} + RT_{0Steve}P(<Steve, Mary>)R_{k0Mary} \end{bmatrix} \\
&= 0.0032 * \begin{bmatrix} 0.298 * 0.0032 * 0.320 + 0.298 * 0.0032 * 0.235 + \\ 0.298 * 0.0032 * 0.320 + 0.298 * 0.0032 * 0.235 \end{bmatrix} = 0.0000035 \\
P(RT_1 \mid R_{k1} \mapsto RT_1) &= P(<Bert, John>)\begin{bmatrix} RT_{0Bert}P(<Bert, Steve>)R_{k0Steve} + RT_{0Bert}P(<Bert, John>)R_{k0John} + \\ RT_{0Steve}P(<Steve, Steve>)R_{k0Steve} + RT_{0Jody}P(<Steve, John>)R_{k0John} \end{bmatrix} \\
&= 0.0032 * \begin{bmatrix} 0.298 * 0.0032 * 0.319 + 0.298 * 0.0032 * 0.226 + \\ 0.298 * 0.1081 * 0.319 + 0.298 * 0.0032 * 0.226 \end{bmatrix} = 0.0000352 \\
P(insert(RT_0) \mid R_k \mapsto RT) &= P(<-, Mary>)\left[P(<-, Ellen>)RT_{0Ellen} + P(<-, Jody>)RT_{0Jody}\right] \\
&= 0.005 * \left[0.005 + 0.005\right] = 0.00005 \\
P(insert(RT_1) \mid R_k \mapsto RT) &= P(<-, John>)\left[P(<-, Bert>)RT_{0Bert} + P(<-, Steve>)RT_{0Steve}\right] \\
&= 0.005 * \left[0.005 + 0.005\right] = 0.00005
\end{aligned} \tag{51}
$$

$$
\begin{aligned}
P(RT \mid R_k \mapsto RT) &= \prod_i^M \left[\sum_j^N P(RT_i \mid R_{kj} \mapsto RT_i) + P(insert(RT_i) \mid R_k \mapsto RT) \right] \\
&= \left[P(RT_0 \mid R_{k0} \mapsto RT_0) + P(RT_0 \mid R_{k1} \mapsto RT_0) + P(insert(RT_i) \mid R_k \mapsto RT)\right] \\
&= \left[P(RT_1 \mid R_{k0} \mapsto RT_1) + P(RT_1 \mid R_{k1} \mapsto RT_1) + P(insert(RT_i) \mid R_k \mapsto RT)\right] \\
&= \left[0.0000365 + 0.0000034 + 0.00005\right]\left[0.0000035 + 0.0000352 + 0.00005\right] \\
&= 0.00000000797
\end{aligned} \tag{52}
$$

Note that the fact that the head words for the {Ellen, Jody} role do not match decreases the $P(RT \mid R_k \mapsto RT)$ substantially.

Appendix G: A Numerical Example of the Relational Substitution Component

As an example of the calculation of the substitution component, consider the calculation of the probability that "Mary" substitutes for "Ellen".

$$
\begin{aligned}
P(<Mary, Ellen> \mid R_k \mapsto RT, RT) &= \frac{P(RT_0 \mid R_{k0} \mapsto RT_0)}{P(RT_0 \mid R_{k0} \mapsto RT_0) + P(RT_0 \mid R_{k1} \mapsto RT_0) + P(insert(RT_0) \mid R_k \mapsto RT)} \\
&= \frac{0.0000365}{0.0000365 + 0.0000034 + 0.00005} \\
&= 0.406
\end{aligned}
\tag{53}
$$

So, the model is somewhat confident that "Mary" substitutes for "Ellen" given that the trace generated the target sentence, although there is a slightly higher probability that the binding was inserted, again because the head words are not the same.

Appendix H: Probabilities of aligning verbs and nouns in sentential context

Note: #N refers to the total number of WordNet references as a noun and #V refers to the number of WordNet references as a verb. The numbers in bond are instances where the model correctly aligned words of the appropriate syntactic category.

Polysemous Word	Context Fragment	P(Verb)	P(Noun)
move	could move people to	**0.457**	0.016
	something move on another	0.00	0.054
	you move away [E]	**0.037**	0.014
#N=539	on the move either pursuing an	0.044	**0.131**
#V=300	made no move to	0.007	**0.759**
	but in a more ominous move they	0.011	**0.449**
report	to report her to the	**0.394**	0.358
	if that happens report directly to me [E]	0.057	0.057
	is to report happenings and	**0.12**	0.02
#N=796	the weather report [E]	0.00	**0.053**
#V=135	her report she	0.063	**0.126**
	a report to the	0.18	**0.28**
state	may state which	**0.025**	0.024
	and state that it has been	0.17	0.228
	to state both the	**0.441**	0.098
#N=1064	of the state instead of	0.189	**0.254**
#V=95	each state except rhode island which had	0.00	**0.002**
	of any state in the	0.07	**0.49**
force	and force the	**0.08**	0.06
	might force the	**0.15**	0.11
	may force the	**0.27**	0.19
#N=1039	did so by his force and	0.055	**0.159**
#V=94	a force of	0.29	**0.34**
	little rock police force have said	0.026	**0.425**
fight	they had to fight other	**0.416**	0.224
	to fight the	**0.23**	0.13
	the colonists fight bravely in the	0.044	0.071
#N=570	into a fight [E]	0.16	**0.22**
#V=159	to get back in the fight again	0.286	**0.644**
	in what looked like a fight to the	0.869	**0.907**
plan	from the past plan for the	0.12	0.16
	to plan careers [E]	0.165	0.314
	have to plan coverage before during and after the	0.025	0.042
#N=504	a plan to	0.17	**0.27**
#V=90	the plan can be	0.064	**0.14**
	that the plan was put	0.061	**0.161**

attack	to attack [E]	**0.16**	0.10
	if bacteria attack the	**0.08**	0.06
	may violently attack the	**0.135**	0.095
#N=925	from native american attack in the new	0.036	**0.045**
#V=44	had an attack of	0.254	**0.652**
	of a heart attack [E]	0.008	**0.018**
process	to think and process language or are these	**0.203**	0.195
	to process data [E]	**0.465**	0.186
	are able to process information more completely	**0.348**	0.036
	than are those under the		
#N=108	was a slow process [E]	0.174	**0.189**
#V=431	in the election process [E]	0.00	0.00
	[S] this elimination process is called	0.002	**0.007**
shot	of how he shot the	0.102	0.102
	[S] they shot out the	0.254	0.254
	was shot and	**0.16**	0.06
#N=2068	to get a shot [E]	0.448	**0.692**
#V=97	for a second shot [E]	0.367	**0.379**
	and not a shot had been	0.318	**0.424**
cost	also cost money [E]	0.01	0.045
	will cost you	**0.823**	0.757
	and canned vegetables actually cost less than	**0.042**	0.026
#N=449	to the cost of	0.10	**0.16**
#V=51	at low cost a home that	0.008	**0.011**
	of the cost of their	0.00	**0.21**
record	and record this	0.06	0.07
	[S] record this	**0.2**	0.15
	and record the	**0.08**	0.06
#N=794	put her new mozart record on the	0.373	**0.424**
#V=50	his record keeping and his	0.08	**0.12**
	is a complete record of the	0.087	**0.152**
test	to test ph [E]	0.006	0.149
	to test this before you	**0.26**	0.112
	do not test the	**0.396**	0.204
#N=798	i have to take my swimming test [E]	0.365	**0.412**
#V=50	on one test got the	0.759	**0.88**
	the leakdown test should be	0.02	**0.075**
hit	that will soon hit the	**0.127**	0.011
	through my window hit the	0.037	0.037
	was hit with a	0.00	0.00
#N=18	made a hit all right	0.13	**0.294**
#V=1602	to have a hit everybody was	0.493	**0.517**
	a hit made with a man on	0.212	**0.285**

demand	[S] jobs today demand more	0.008	0.262
	need and demand many	0.043	0.069
	and demand the	**0.08**	0.06
#N=561	the demand for	0.01	**0.15**
#V=76	to meet consumer demand [E]	0.032	0.032
	that the demand for the	0.092	**0.188**
help	that will help them get good	**0.062**	0.047
	to help their	**0.22**	0.08
	do to help poor people [E]	**0.056**	0.049
#N=456	you may need some help sheila [E]	0.138	**0.151**
#V=1035	your help [E]	0.04	**0.166**
	the help of	0.07	**0.18**
share	does not share equally in its	0.001	0.001
	so that they can share in the	**0.418**	0.019
	had to share the	**0.275**	0.183
#N=651	only for our share of what is	0.00	**0.019**
#V=63	only that we should get our share in it	0.101	0.031
	a share of the	0.09	**0.29**
practice	you have to practice very	**0.212**	0.127
	if she doesn't practice [E]	0.141	0.235
	with practice [E]	0.05	0.15
#N=35	the practice in the	0.08	**0.19**
#V=435	[S] with a little practice it is	0.009	**0.969**
	i was out of practice because for years	0.363	0.363
charge	can be used to charge batteries and to	**0.922**	0.045
	[S] solar cells charge the	0.518	0.608
	[S] some banks charge a	**0.38**	0.08
#N=1558	be in charge of	0.004	**0.006**
#V=46	was in charge of our	0.008	**0.015**
	it in charge of my	0.132	**0.176**
call	they would call back to him	**0.102**	0.044
	to call you	**0.47**	0.42
	[S] i'll call dr moravia [E]	**0.219**	0.177
#N=20	a call and	0.07	**0.11**
#V=2866	the call in the	0.08	**0.19**
	[S] it is a call for	0.202	**0.646**
date	can date and	**0.327**	0.213
	do we date the	**0.566**	0.233
	[S] movie cartoons date back to the time when	0.432	0.458
#N=893	[S] you have a date or something [E]	0.048	**0.215**
#V=3	the date of your	0.07	**0.21**
	and the date [E]	0.06	**0.21**