LEX: A Retrieval Theory of Lexical Access

by

Peter J. Kwantes

A thesis submitted to the Department of Psychology
in partial fulfilment of the requirements
for the degree of Doctor of Philosophy

Queen's University
Kingston, Ontario, Canada

August, 1999

0-612-42952-0

Abstract

In this thesis I introduce LEX, a new model of visual word identification. The model is built on three guiding assumptions. First, word identification is considered as a retrieval operation whereby identified letters are used as a probe to retrieve information from lexical memory. Second, phonological information about a word is associated with orthographic information within the same memory trace. In other words, letters are not treated as graphical representations of sounds. Finally, lexical access follows the order of the retrieval probe's letters. Specifically, I assume that the lexical access system requires a list of letters, organised from left to right, as a retrieval probe. Lexical access follows the organisation of the letters by retrieving a word starting with the first letter and terminating at the last letter. I demonstrate that LEX is capable of explaining many phenomena considered important to the validation of competing models. I also provide empirical evidence for the requirement that the lexical access system expects a list of letters to retrieve a word.

Acknowledgements

I would like to thank Dr. Kevin Munhall and Dr. Lola Cuddy for the helpful comments they made on an earlier draft of this thesis. I would also thank my external committee members, Dr. Randy Flanagan, Dr. John Kirby and Dr. Tom Carr for bringing some important issues to the fore during my defence.

I owe a great debt of gratitude to my supervisor, Dr. Doug Mewhort, who not only spent an enormous amount of time with me trying to get me to clarify my thinking, but kept me fed for the past four years. I am also very grateful to Dr. Beth Johns who, by making me part of the family, played a major role in keeping me well fed. Beth has been a valuable critic of my work and a good friend.

My family and friends have also been an enormous help during the Ph.D. process. Dr. David Smith and Dr. Irene Armstrong have been an enormous help to the whole process; both as a source of information, and encouragement. I also thank my parents, Matthew, Melanie, and Mary Ann Kwantes, Meredith Allison, Adrienne Munro, Sarah Hill, Sam Laldin, Rachel Mulloy, Jeff Stewart, and Steve Smith for not telling me to shut up when I would not stop complaining about how frustrating my work was.

Finally, thanks go to Doug Girvin and Sun Microsystems of Canada. Without Doug's help in increasing the computing power of the lab, this thesis could not have been completed.

List of Tables

Table of Figures

Table of Contents

Chapter 1: The Scope of the Problem

"Reading research" has many faces. Some study the most effective way to teach children how to read (e.g., Adams, 1990). Others are interested in the processes involved in comprehending text (e.g., Kintsch, 1998) or study the mechanisms controlling when and where a reader's eyes move across a page of text (e.g., Rayner & Pollatsek, 1989). Finally, some reading researchers study the processes used to identify individual words. The processes used to identify individual words are the focus of the present work; in particular, I will focus on how people name words and how they decide on a letter string's lexical status.

The tasks

Students of word identification have used the naming and lexical decision tasks almost exclusively in their experiments. In a naming task, subjects are shown strings of letters and asked to pronounce the letter string out loud as quickly and as accurately as they can. Each string is either a word or pronounceable nonsense, that is, a nonword like *dorch*. In a lexical decision task, subjects are shown strings of letters, some words and some nonwords, and asked to indicate whether or not the string is a word. Both tasks are designed to measure how long it takes a person make access to a word in the mental lexicon.

In the naming task, researchers are interested in both the accuracy of the response and the time it takes to initiate the pronunciation of the letter string. Early researchers took the time to say a word as a measure of the time find the word's pronunciation in the reader's mental lexicon. The time to read a nonword was thought to measure the time it takes to "sound out" a string of letters.

The time to name a word is a potentially problematic measure of lexical access time, however. Rayner and Pollatsek (1989) pointed out that the time it takes a person to read a word aloud also reflects several processes that are unrelated to finding a word in the lexicon. After a word has been found in the lexicon, a motor program for pronouncing it must be found, sent to the articulators, and finally, executed. The steps in pronouncing a word after it has been found take time, hence, response time in the naming task may be contaminated by factors unrelated to lexical access. The second problem with the naming task, according to Rayner and Pollatsek, is that familiar words are read aloud about as well as words that are unknown to readers (e.g., nonwords). Part of lexical access entails making access to its meaning. If readers can name words that they know about as well as they can name words they do not know, the task may not provide a fair measurement of the time to make lexical access.

The lexical decision task may provide a more adequate measure of the time to gain access to the lexicon because it requires subjects to decide whether a letter string is a word. Presumably, if readers know a letter string is a word, they also know its meaning. Further, the lexical decision task requires a simpler response—usually, subjects press one button on a response key if the letter string is a word, and another if it is not. The lexical decision task may, however, also be an imperfect task with which to measure lexical access time. First, there is no guarantee that a reader knows the meaning of a word even though she can identify a letter string as a word. Second, half of the stimuli used in the lexical decision task are nonwords, and the time required to decide on whether a letter string is a word depends on how word-like the nonwords are (e.g., Andrews, 1989). For example, deciding on the lexicality of a letter string is easier when the nonwords are strings of consonants (e.g., *drtw*) than when they contain letter combinations found in words (e.g., *drom*). If decision time for words is sensitive to the construction of the nonword materials, lexical decision times might not be any more accurate as measures of lexical access than latencies in the naming task.

Despite their shortcomings, the naming and lexical decision tasks are widely used in word identification experiments. In most experiments, psychologists often vary the class of word or nonword and gauge performance across the classes of stimuli. Any regularities in performance across the classes are taken to reflect storage and processing mechanisms that are common to all readers. For example, in both the naming and the lexical decision tasks, readers respond more quickly and accurately to words that occur frequently in text than words that occur relatively infrequently (e.g., Baron & Strawson, 1976). Frequency varies widely across words in text; for example, *the* occurs over 69,000 times per million words of text according to Kucera and Francis (1967) word norms, whereas *apt* occurs about once every million words of text. Any model with hopes of becoming the generally accepted theory of word identification must be able to account for such regularities in reading behaviour.

Motivation for the thesis

Prior to around 1980, research in word identification and memory were closely aligned in that both fields used the same metaphor to describe how a subject gets information out of memory or the lexicon. Access to an item in memory or a word was described in terms of a "look-up" or search operation.

Around 1980, a split occurred between reading research and memory research. The split was brought on by the invention of a new class of memory models, distributed memory models (e.g., Hintzman, 1984; Murdock, 1982; Metcalfe-Eich, 1982). Distributed models abandoned the idea of search for an item in memory in favour of the notion that an item is retrieved from memory by blending information in the memory system to create a facsimile of the probe.

While distributed models of memory have become, and still are, the dominant form of simulation model for theories of memory, most models of word identification have retained the notion that access to information in the lexicon is essentially a search operation. My thesis represents an attempt to reunify theories of word identification and theories of memory by building a model of word identification that borrows from a well-known distributed model of human memory (Hintzman's, 1984, Minerva2 memory model).

Guiding assumptions in theories of word identification.

Current theories of word identification are based on three guiding assumptions. First, identifying a word is treated as the operation by which a word's address is found. That is, the lexicon is treated as a content addressable system. Second, word identification is treated as an extension of perceptual classification. The input stimuli (letters) are passed through a series of filters that transform the letters into sounds or into units of representation that correspond to entire word entries. Finally, theories of word identification treat letters as graphical representations of sounds. As such, current theory treats word identification as the operation by which sounds, or phonemes, are derived from the letters they represent. In this thesis, I introduce a model that eschews all three assumptions and demonstrate that it is capable of reproducing several phenomena considered important in the word identification literature .

Organisation of the thesis

In this thesis, I introduce a new model of word identification to explain performance in both the naming and the lexical decision tasks. The organisation of the tnesis is as follows. I will first discuss current models of word identification. I will then introduce a new way to treat word identification. I will argue that word identification is a special case of memory retrieval that requires specialised data storage and processing. I will also empirically test one of the model's assumptions. Finally, I will demonstrate that a very simple model of memory with few processing mechanisms and a life-size lexicon can capture a large number of empirical phenomena in the word-identification literature.

Chapter 2 : The Interactive-Activation Model of Word Identification

Current models of word identification have a common ancestry in the interactive-activation model (IAM) of word identification (McClelland & Rumelhart, 1981; Rumelhart & McClelland, 1982).

Basic Structure

McClelland and Rumelhart (1981; Rumelhart & McClelland, 1982) proposed that word identification could be understood in terms of a hierarchical network of detectors. A graphical representation of the IAM is shown in Figure 2.1. The IAM uses what is called a *localist representation* in that single detectors, or nodes, represent entire entities such as letters or words.

At the lowest level of the hierarchy, detectors register the visual features of the display. The system is tuned to detect 14 features. Features are defined in terms of horizontal, vertical, and diagonal line segments.

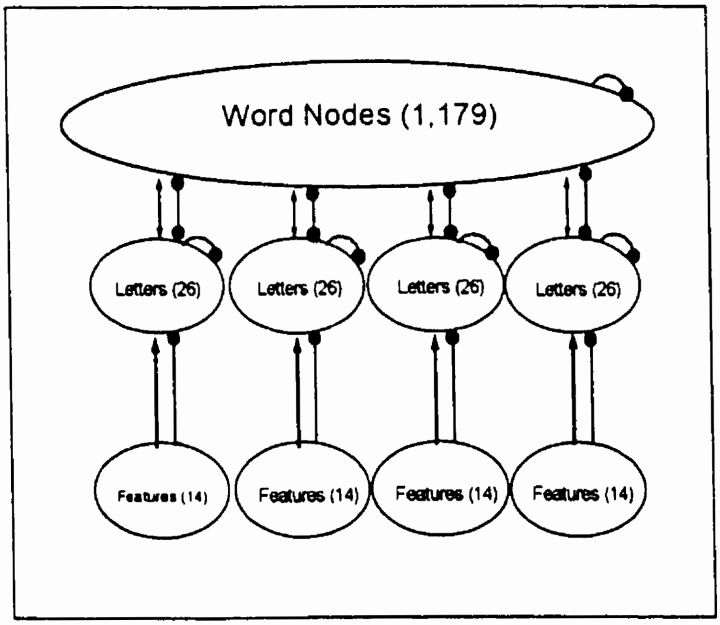

Figure 2.1 Basic architecture of the IAM (Arrows denote excitatory connections and solid circles denote inhibitory connections. The number of nodes in each part of the model is in parentheses).

Feature detection occurs separately and simultaneously in each of four letter positions, or letter channels, in the visual display. That is, there is an array of 14 feature detectors for each letter position. Once the features in each letter position are detected, their activation spreads automatically to the letter nodes. A complete set of 26 letter nodes, one for each letter of the alphabet, is connected to each letter channel. Depending on which line segments, or features, are activated by the display, a letter node is activated in each letter position. Letter nodes are mutually inhibitory—as a letter node becomes activated, the other letters in the letter channel are inhibited.

Activated letter detectors automatically activate consistent word nodes. Hence, a detector that represents a *T* in the first letter channel will activate word nodes for words with an initial *T* and will inhibit word nodes that represent words without an initial *T*. Like the letter nodes, word nodes are mutually inhibitory; activated word nodes inhibit all other word nodes.

Interconnections

Every node is connected to every other node within the same layer. As well, every node is connected to every node in its neighbouring layer. For example, every letter node is connected to every other letter node within a letter channel and connected to every word node.

Node Activation

Letter and word nodes are assumed to possess a resting level of activation. In the model's quiescent state, the resting level of activation for any node is determined by frequency of usage. For example, a word node representing a high-frequency word such as *the* would possess a higher resting activation than a low-frequency word like *apt*. During processing, however, a node's activation is determined by the impact that neighbouring nodes have upon it. The impact of neighbouring nodes on the activation of any one node (n_i) at time t, is expressed as the net input to the node from all its neighbouring nodes. The equation is given by

$$n_i(t) = \sum_j \alpha_{ij} e_j(t) - \sum_k \gamma_{ik} i_k(t)$$

Where e and i represent excitatory and inhibitory connections, respectively. α and γ represent the connection weights between neighbouring nodes. The end result of the equation is a change in the resting value of node i at time t as a function of the excitatory and inhibitory connections of neighbouring nodes.

When the net activation to a node is excitatory, the node activation is prevented from exceeding a maximum activation of 1.0 by scaling the node's activation using the formula

$$n_i'(t) = n_i(t)(1 - a_i(t))$$

When the net activation to a node is inhibitory, its activation is prevented from exceeding a minimum activation of -0.2 using the formula

$$n_i'(t) = n_i(t)(a_i(t) + 0.2)$$

Using the above formulae, the IAM updates the activation of word and letter nodes over successive processing cycles. On each cycle, letter nodes that have been activated by the features registered from the display decrease the resting activation of the other letter nodes within the letter channel. Letter nodes also push the resting level of the word nodes above or below their resting levels as the letter nodes' activation is passed to the word nodes. The connection between a letter node and a word node is excitatory if the word contains a letter in the same ordinal position as one of the letter channels. For example, a letter *T* in the fourth letter channel will push the resting activation of the word node for *cart* above its resting value. The opposite occurs for the word node for the word *cars*. Because word nodes are mutually inhibitory, as a word node's activation increases, it does so against a background of inhibition from the other word nodes that try to push its activation back down to its resting levels.

As word nodes compete with one another, they also pass their activations back down to the letter nodes. A word node has an excitatory connection back to a letter node if the two share a letter in the same ordinal position. The connection is inhibitory when they do not. The updating of the letter nodes by the word nodes at time *t* signals the end of one processing cycle.

Word identification occurs over several processing cycles. McClelland and Rumelhart (1981) calculated the new activation of any node at time t+1 as a function of the net influence of neighbouring nodes, $n'_i(t)$, and a rate of the activation's decay that occurs between cycles (θ). They express the activation a node a at time $t+1$ as:

$$a_i(t+\Delta t) = a_i - \theta_i(a_i(t) - r_i) + n'_i(t)$$

where r_i is the resting activation of the node.

If a word node exists that is consistent with the letters in the letter channels, the resting activation of the word node will gradually rise over the processing cycles. As the node's activation rises toward threshold, the activation of the other word nodes are pushed down towards the minimum activation. Word identification has occurred when only one word node's activation has reached threshold.

Problems with the IAM

The separation of identity and location information.

The IAM performs letter identification independently in each letter channel. The letter channels represent the ordinal spacing of characters in the display. Done in this way, a letter's identity is tied to its channel, that is, letter identities can not migrate across channels. Several studies, using the bar-probe task (Averbach & Coriel, 1961) however, have demonstrated that letter identity information is stored independently from letter location information; letter identities appear to migrate across the space defined by the display.

In one variation of the bar-probe task, a letter string masked immediately following a brief (a duration less than 200 ms) display. After a delay (between 20 and 200 ms), an arrow is placed under one letter position in the array. The subject is required to report the letter at the position of the probe As the delay between the mask and probe increases, subjects' accuracy for letter report decreases. Of vital importance is the types of errors subjects make. Subjects make predominantly location errors (Mewhort, Campbell, Marchetti, & Campbell, 1981). That is, they report a letter from the display, but one from a location other than the probed location. Clearly, identity and location information are stored separately because the two kind of information are not lost simultaneously. Identity and location are tied in the IAM—location errors and identity errors are not separable.

Feedback from word representations to the letter level.

Mewhort and Johns (1988) evaluated the IAM's account of the *word superiority effect* (WSE); the target phenomenon for the IAM. Made popular by Reicher (1969), the WSE refers to the finding that, using brief displays, a letter is more easily identified when it is presented in a word than when it is presented in the context of a pseudoword or alone. To account for the effect, the IAM places responsibility on the word nodes' feedback to the letter nodes. When the characters of a letter string match those of a word stored among the word nodes, the word node increases the activation level of the consistent letter nodes thus making the letter easier to identify when it is in the context of a word than other contexts. Such feedback is less strong when the letters form a nonword; hence letter activations are not as drastically increased by word activations when letter identification occurs in the context of a nonword

Mewhort and Johns (1988) challenged the notion that the WSE occurs because of feedback to the letter nodes from the activated word nodes. The IAM's word nodes are activated automatically when letter nodes are activated. Feedback to the letter nodes also occurs automatically. Hence, word node activation and feedback to the letter nodes will occur automatically yielding a WSE as long as subjects can identify letters. However, Mewhort and Johns (Experiments 3) failed to produce a WSE when the letters of a word were transformed (e.g., upside down). Even when letter identification performance was equated for upright and transformed letters in the control condition (identifying letters in a transformed pseudoword), only upright letters yielded a WSE. Letter node re-activation, if it occurs, is not automatic.

The representation of space within and between letter channels.

The IAM has an inconsistent representation of space. Within a letter channel, space is assumed explicitly—letters within each channel are defined by the spatial arrangement of lines that make up the characters. The IAM is inconsistent in its representation of space because it does not represent space *between* letter channels. As a result, although the IAM assumes that a vertical bar at the left of a letter channel is a different feature than the same bar at the right of the channel, the model has not defined space well enough to distinguish TRAP from T R A P. The point is important because letter spacing provides a boundary condition on the WSE, the target phenomenon for the model (Marchetti & Mewhort, 1986). In addition, letter spacing controls the familiarity effect in tachistoscopic whole-report (Campbell & Mewhort, 1980) and the familiarity effect predicts word identification (Mewhort & Beal, 1977).

An improvement to the IAM — BLIRNET

Mozer (1987; 1991) proposed a model, BLIRNET, as an improvement to the IAM. BLIRNET was able to identify words in arbitrary positions on an artificial retina. Because the model could identify a word in any retinal location, it overcame an important limitation of the IAM's letter-channel representation scheme. For example, word *dog* would be encoded as *_dog* or *dog_* in the four letter channels of the IAM (the _denotes a word boundary). The two words are the same, but because they are offset by one character, the IAM treats the two words as though they were different because *_dog* and *dog_* have none of the same letters in the same letter channel. BLIRNET overcomes this limitation by using a representational scheme that maintains the relative position of letters in the visual display. By passing retinal information through five additional layers of nodes that are not sensitive to retinal position, BLIRNET can identify the word, *dog* regardless of where on it is positioned on its artificial retina.

Despite BLIRNET's improvement over the IAM, and the criticisms raised above, the IAM remains an influential model of word identification. In fact, as I will discuss shortly, the IAM, and not BLIRNET, is the model that has been incorporated in the recent computational version of Coltheart's (1978; Coltheart, Curtis, Atkins, & Haller, 1993) dual-route model of word identification.

Chapter 3: Current Theories

In this chapter I introduce connectionist models of word identification and the dual-route theory of word identification. The connectionist models I will discuss in this chapter have abandoned the localist representation scheme used by the IAM in favour of what is called a *distributed representation*. Recall that the IAM uses single nodes to represent features, letters, and whole words. More recent connectionist models have opted to represent the words across several nodes.

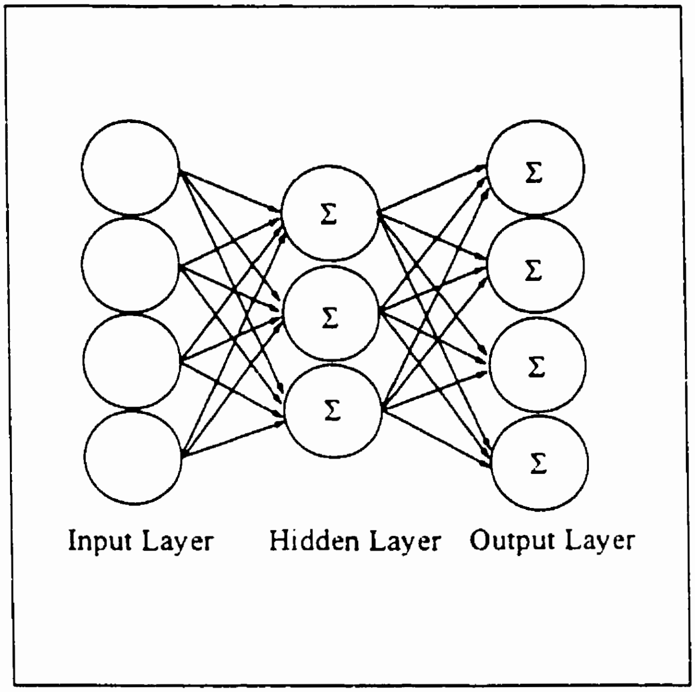

Figure 3.1 A generic three-layer artificial neural network

A Generic Neural Network

An example of a generic connectionist, or neural network, model is shown in Figure 3.1. At the left of the figure, is a layer of nodes, labelled the input layer. Information is presented to the model by "turning on" the appropriate nodes in the input layer. For example, one stimulus can be represented by the vector, [0 0 1 0 1], and another by the vector [0 1 1 0 0] where 0 represents a node in the row that is turned off, and 1 represents a node that is turned on. The middle layer of nodes, the hidden layer, integrates information from the input layer and passes it to the nodes of the next layer, the output layer. The pattern of activity in the nodes at the output layer corresponds to the response associated with the input pattern.

Notice that every node is connected to every node in the adjacent layer. Each connection is weighted to represent the strength with which one node is connected to the next. As well, each connection between nodes possesses a connection weight (a random real number between 0 and 1), so that information that is integrated from the previous level's nodes will either increase or decrease the receiving node's activation. To derive an output pattern from an input pattern, the input nodes' activations are multiplied by the weights connecting them to the hidden layer of nodes. The activations of the nodes at the next layer, the hidden layer, are calculated as a function of the net activations from all connection from the previous layer. Finally, the activations of the nodes in the output layer are calculated as a function of the net input to the output nodes from the hidden nodes.

To work, a neural network must learn to associate a stimulus (input pattern) to a response (output pattern). For example, the network could learn to associate a spelling pattern with a sound pattern. Learning is accomplished by adjusting the connection weights among the nodes so that when the network is presented with an input pattern, the appropriate output pattern emerges in the nodes of the output layer.

Learning is accomplished over several repeated pairings of the input and output patterns. One pairing of each of the input and output patterns is called an *epoch*. Prior to learning, the network's connection weights are set to random values. When the network is presented with the first stimulus, the output it generates at the output layer is compared to the correct response. The connection weights between nodes are adjusted, over several epochs, to minimise the discrepancy between the network's output and the correct response. Further, the degree to which the weights are adjusted is a function of how discrepant the output and correct patterns are—this type of learning is generally referred to as *supervised learning*. The network is said to have learned an association between an input and output pattern when the discrepancy reaches a predetermined, minimum level.

The Seidenberg and McClelland (1989) model of word identification

The model of word identification proposed by Seidenberg and McClelland (1989) is an example of a network similar to the one described above. The model has three layers of nodes: an input layer, output layer, and an intervening or hidden layer. The model's knowledge about orthography and phonology is represented in the connections between an orthographic (input) and phonological (output) layer.

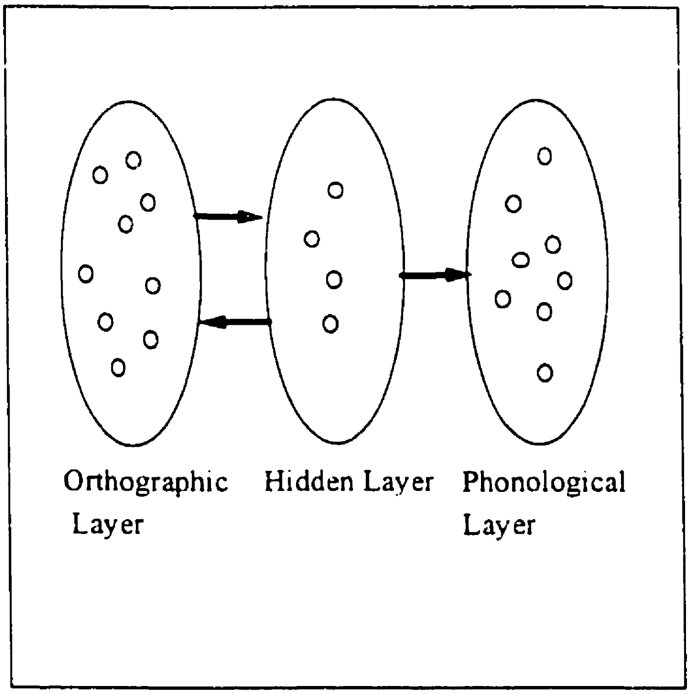

Figure 3.2 Outline of the architecture of Seidenberg and McClelland's (1989) model

Training operations.

To train the network on a set of words, Seidenberg and McClelland (1989) used a supervised learning algorithm called *back-propogation* (Rumelhart, Hinton, & Williams, 1986). At the beginning of training, a word's spelling is encoded into to the orthographic nodes. The activation of the orthographic nodes spreads through the hidden nodes to the nodes at the phonological layer. A hidden node's activation is a function of the weighted sum of all the connections that terminate upon it. Seidenberg and McClelland calculate a hidden node's activation using the formula

$$a_j = \frac{1}{1+e^{-net_j}}$$

where a_j is the activation of unit j and net_i is the summed influence of node in the orthographic layer, and is given by the formula

$$net_i = \sum_j w_{i,j} a_j + bias$$

The *bias* term treated as an "extra weight or connection to the unit from a special unit that always has an activation of 1" (Seidenberg & McClelland, 1989, p. 527). Once the hidden nodes' activations are determined, they produce a feedback pattern on the orthographic units and a feedforward pattern on the phonological units using the same formulae.

The orthographic and phonological patterns produced from the input are compared to the activation of the correct response using the formula

$$E = \sum_i (t_i - a_i)^2$$

where E is the measure of error, t is the target activation of node i, and a is the activation of the ith node in the feedback or feedforward pattern. Over several training cycles, the connection weights in the network are adjusted to minimise E, or the degree to which the output of the model mismatches the target orthography and phonology.

Representing orthography and phonology.

Representing orthographic and phonological information in the nodes presented a special challenge for Seidenberg and McClelland (1989). Intuitively, one might construct a network wherein 26 nodes, each corresponding to a letter of the alphabet, are used for the orthographic nodes. Likewise, one could dedicate one node to each phoneme in the language. Learning a word's pronunciation then, would simply be to associate letter nodes with phoneme nodes. However, without reference to the spatial arrangement of the letters, letter nodes for the word *tap* would be consistent with *tap*, *pat*, and *apt*. A similar problem exists for phonemes without reference to their organisation.

Seidenberg and McClelland solved the relative position problem by making a set of nodes collectively represent a tri-gram of letters or phonemes. Seidenberg and McClelland referred to the letter tri-grams as "wickelgrams" and the phoneme tri-grams as "wickelphones"; collectively referred to as "wickelfeatures" named in honour of Wickelgren (1969), who proposed the representation scheme. Hence, the word *tap*, was presented to their model by activating the nodes corresponding to *_ta*, the nodes corresponding to *tap*, and the nodes corresponding to *ap_* (where _ represents a word boundary).

Training regime.

Seidenberg and McClelland (1989) trained their model on every one-syllable word in the Kucera and Francis (1967) words norms that had three or more letters. After removing proper names, inflected forms of words, and abbreviations from the corpus, they were left with 2,897 words with which to train the model.

On each training epoch, every word in the corpus had a chance to be selected from the corpus. The probability a word was sampled from the corpus of words was a function of its printed frequency as tabulated by the Kucera and Francis (1967) word norms.

Simulating word recognition tasks.

To simulate word naming, wickelgrams appropriate to the orthography of a word are activated in the orthographic nodes in the input layer by clamping the appropriate nodes to an activation of 0.9. Nodes that were not relevant to the orthography were set to an activation of 0.1. Activation from the orthographic layer spreads through the network to the wickelphones represented in the phonological layer. Naming latency is estimated as a function of the difference between the models response in the phonemic nodes at the output layer and the correct response. Because response time generally increases with the probability of making an error, Seidenberg and McClelland (1989) reasoned that response latency could be estimated from the degree to which the model's output mismatched the correct output. Seidenberg and McClelland assumed that high similarity between the model's output and the correct response reflected a fast response time and that low similarity reflected a long, error-prone, response.

The lexical decision task is simulated by comparing the wickelgrams at the input level with an orthographic output. Similar to the naming task, latency in the lexical decision is calculated as a function of the discrepancy between the input pattern and the orthography that is returned to the orthographic units.

Problems with the Seidenberg and McClelland (1989) model.

Besner, McCann, Twilley, and Seergobin (1990) criticised Seidenberg and McClelland's model on the grounds that, while the model seemed to perform well when asked to read words, it did not read nonwords as well as humans. The failure of the model to mimic nonword reading in humans led Besner et al. to conclude that a separate mechanism, one which uses rules for spelling-to-sound translation, is necessary for pronouncing nonwords.

Seidenberg and McClelland (1990) defended their model by claiming first, that the model's performance is judged more harshly than human performance, and second, that its inability to read nonwords well was a consequence of the small training set of words of approximately 3000 one-syllable words.

Seidenberg and McClelland's (1990) defence of the model is not adequate. Their model reads nonwords by using word knowledge to generalise to novel stimuli. They were correct in their claim that the success with which the model can generalise depends on how much knowledge it possesses. However, regardless of how many words the model knows, because the model reads nonwords by generalising from words it knows, the model will only be able to read nonwords that contain letter combinations present in words. For example, there are no one-syllable words that contain the *nje* tri-gram in the nonword *jinje*; hence it is a problematic letter string for the Seidenberg and McClelland model. In sum, the model's difficulty with nonwords is not merely a function of a lack of word knowledge; the difficulty is caused by the constraint that the structure of the mappings between spelling and sound place on generalisation.

Naming latencies in the model are also derived inappropriately. Seidenberg and McClelland (1989) derived a naming latency from the difference between the model's response and the correct response. While there is a relationship between response time and the probability of making an error, response latency is not necessarily a function of the probability that a subject makes an error. Their method of deriving a response time ensures that error response time are always slower than correct response times—an observation that is not always made in data.

Finally, letters are encoded by Seidenberg and McClelland as a spatial arrangement across a set of orthographic units. While there is nothing inherently wrong with supposing that spatially arranged letters are used as a data structure for finding a word, the psychological reality of wickelgraphs and wickelphones is, as yet, unestablished. Further, using wickelgraphs lends Seidenberg and McClelland's model to the same criticism raised for the IAM. Space between letters is not represented, hence, the model cannot use extra-letter information to derive a response.

The Plaut, Seidenberg, McClelland, & Patterson (1996) model of word naming

The Seidenberg and McClelland (1989) model was recently improved upon by Plaut, Seidenberg, McClelland, and Patterson (1996). The new version of the model did two things that the original model could not. First, the model read words and nonwords as well as humans can, and second, the model responded to stimuli in real time.

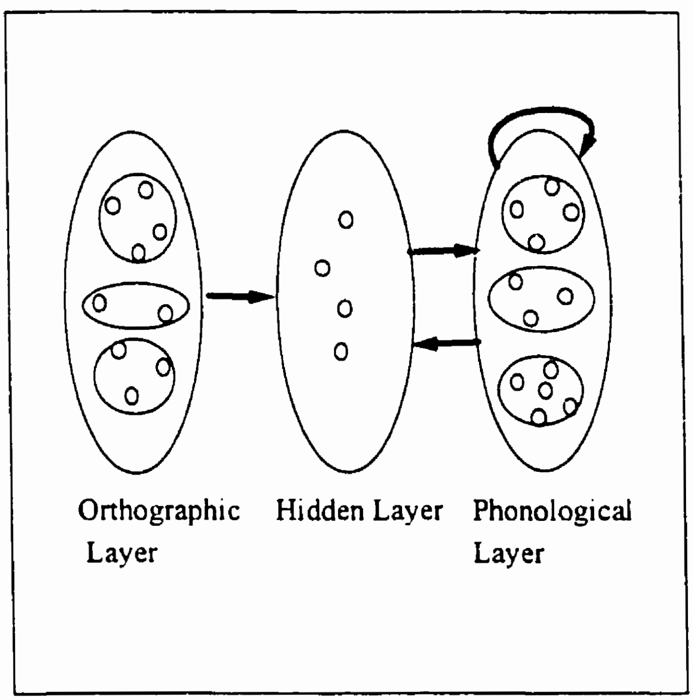

Figure 3.3 Outline of the architecture of Plaut et al.'s (1996) model. Grouped nodes represent nodes that are devoted to components of words (i.e., onset, vowel, and coda).

Plaut et al. (1996) reiterated Seidenberg and McClelland's (1990) claim that the small training corpus was, in part, responsible for the Seidenberg and McClelland (1989) model's poor nonword reading performance. Plaut et al. also blamed the use of wickelfeatures as the other source of difficulty for nonword reading. They argued that, the condensing of information using wickelfeatures comes at the cost of other useful information. For example, consider the letter *r* in the words *rag*, *grab*, and *hurt*. Each *r* must be represented in separate wickelgraphs; hence, the system loses ability to generalise among the words even though they contain the same letters. Plaut et al. referred to the loss of information as the *dispersion problem*. Their solution to the dispersion problem, was to encode graphemes and phonemes as components of a word: onset (the first grapheme or phoneme), vowel (the middle grapheme or phoneme) and coda (the final grapheme or phoneme). Hence the word *rag* as an input would be represented as the activation of nodes corresponding to the onset *r*, the vowel *a*, and the coda *g*. Using the componential representation scheme, Plaut et al. (1996) demonstrated that the Seidenberg and McClelland (1989) model could read nonwords as well as humans.

The other problem was the fashion in which a response latency is generated by the model. Recall that Seidenberg and McClelland (1989) compared the discrepancy between the pattern of activation at the phonological nodes and the correct response to simulate naming time. Such an estimation of response time is necessary in feed-forward networks such as Seidenberg and McClelland's because responses are generated by the model in a single sweep through the system. Plaut et al.'s new model of word naming using an architecture that allows responses to be generated in real time.

Response latency in the Plaut et al. model.

Unlike Seidenberg and McClelland's (1989) network, in which the nodes of each layer were connected to every node of the next layer, the phonological nodes in Plaut et al's model were connected to every other phonological node *within* the layer. As well, phonological nodes were connected back to the nodes at the hidden layer (see Figure 3.3).

Response latency was calculated as the number of processing cycles the model required to settle on a pronunciation for an input pattern. Because the nodes within the phonological layer were connected to one another, when a phonological node is activated by the presence of an orthographic pattern on the orthographic nodes, the nodes' activations are squashed by the activations of neighbouring nodes. Activation from the phonological nodes is passed back to the nodes of the hidden layer from where it is returned to the phonological layer. Response time was estimated as the number of times the activations of the phonological nodes were updated before the change in their activation reached a minimum. When the network finishes updating its connection weights it is said to have *settled on an attractor* (hence the name, Attractor network). Plaut et al.'s attractor network version of Seidenberg and McClelland's model performed as well as the original model when it read words and read nonwords about as well as human subjects. As well, the settling times of the network mapped closely onto the latency estimates derived from the feed-forward version of the model.

Problems with the Plaut et al. (1996) model

Plaut et al.'s (1996) model is clearly a better account of word naming than its 1989 ancestor. However, it is still not without its problems. Each of the model's problems I will discuss stem from the representational scheme used to represent the orthography and phonology of the printed word.

Unlike feed-forward networks, like Seidenberg and McClelland's (1989), attractor networks do not generalise well to new stimuli. They depend on both a familiar input and output. The componential representation (onset, vowel, and coda) used by Plaut et al. solved the familiarity problem. While a letter string may be unfamiliar, the components are likely to be familiar. The componential representation prevents the model from making *lexicalisation errors* (e.g., responding with the phonological pattern of the word *porch* when presented with the nonword *dorch*). Componential representation allows the onsets, vowels, and codas of separate words to combine to form a response.

A word's orthography and phonology, are assumed to be parsed into onset, vowel, and coda components. The parsing stage that divides a letter string into these components is unspecified in the model. Plaut et al. (1996) finessed this criticism by claiming that, with experience, readers gain knowledge about words that allows for a division among word components.

By using the onset, vowel, coda representational scheme, Plaut et al. cannot claim, as their 1989 counterpart could, that the model is developmental. In fact, it is unclear what kind of reader the model is supposed simulate. Componential representations are assumed to develop in skilled readers. But, if componential representations for words are responsible for the success of the model, how are words represented in the reading systems of beginning readers who do not have enough knowledge of the language to parse a word into its components? As a consequence of the componential representation, when model learns a corpus of words, its learning progress cannot can not be gauged as one would gauge a beginning reader. This is unfortunate; connectionist models learn over time and are, therefore, good candidates as developmental models.

Finally, the model is incapable of reading words that have more than the three components. In fact, it is unclear how additional components would be represented. Onset and coda components would remain unchanged, but several additional sets of nodes would need to be created in order to represent the intermediate letters or graphemes. Ultimately, a set of nodes for each letter or grapheme would need to be used— a representational scheme similar to the use of letter channels in the IAM.

The Dual-Route Cascade (DRC) Model

The DRC model (Coltheart, Curtis, Atkins, & Haller, 1993) postulates the existence of two independent routes to pronunciation in word recognition (Figure 3.4). The first route, sometimes called the lexical route, looks up a word directly from a mental lexicon. The speed with which information can be accessed by the lexical route is a function of the word's familiarity; words that are highly frequent in text are looked up more quickly than words that rare by comparison. The second route, sometimes called the nonlexical route, assembles a phonological code on the basis of rules governing spelling-to-sound correspondence. The speed with which the nonlexical route can translate a letter string into sounds depends on the length of the letter string. Letter strings with more words take longer to translate.

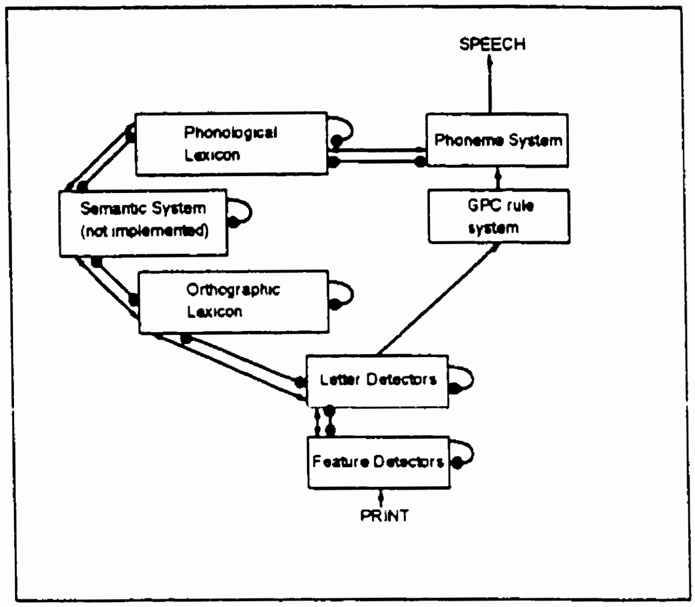

Figure 3.4. The dual-route cascade model of word identification (Arrows denote excitatory connections and dots denote inhibitory connections).

The most cited evidence for two routes to pronunciation (besides the documented cases of dyslexia that appear to affect one route or the other) is found in the interaction in between a word's printed frequency and its compliance to the rules governing spelling to sound translation (Seidenberg, Waters, Barnes, & Tannenhaus, 1984). The interaction is often referred to as the *regularity by frequency interaction*. Words that violate the spelling-to-sound rules of English (irregular words like *wad*), are named more slowly than word that obey them, so called, regular words like *bad*. The latency disadvantage for naming irregular words is attributed to conflicting evidence for two potential pronunciations derived independently from the two routes. The disadvantage, however, interacts with a word's printed frequency such that, it tends to occur mainly for words that are relatively unfamiliar to the reader. Words that are highly familiar to the reader, high-frequency words, are quickly processed by the lexical route without the need for nonlexical involvement.

Further experimental evidence that is offered for a dual-route reading system is found in experiments that examine the extent to which a reader can attend to one route while ignoring the other (e.g., Baluch & Besner, 1991; Monsell et al, 1992; Lupker et al., 1997). Monsell et al.'s subjects named exception words and nonwords (e.g., *dorch*) embedded in various list structures. When irregular words were presented in a list of only irregular words, naming latency decreased compared to a list also containing nonwords. Baluch and Besner reported similar results using Persian words. The pattern of results was interpreted by both Monsell et al. and Baluch and Besner as evidence that subjects can strategically ignore the nonlexical route when reading a pure list of irregular words. Mixing the irregular words and nonwords together preclude subjects from relying on one strategy for pronunciation. As a result, naming latencies for irregular words are increased in mixed lists.

Finally, Coltheart and Rastle (1994) demonstrated that the naming advantage for regular words over irregular words depends on the graphemic position at which the letter string becomes irregular. They found a greater influence of irregularity on irregular words whose irregularity was at the beginning of the word (e.g., *chef*) than words whose irregularity was at the end of the word (e.g., *glow*). Coltheart and Rastle claimed that when the irregularity is positioned at the end of a word, the lexical route accumulates most of the evidence for the correct pronunciation of the word before the nonlexical route translates the final letters. The lexical and nonlexical routes work in tandem to derive a response; hence, when the nonlexical route begins to translate the deviant phoneme, the correct pronunciation has been largely retrieved. The opposite is true when the deviant phoneme is positioned at the beginning of the word. When the irregularity is at the beginning of a word, the nonlexical route provides early evidence for a pronunciation that will conflict with the pronunciation derived by the lexical route.

Architecture and operations of the DRC

The lexical route. Coltheart et al. (1993) chose the IAM (McClelland & Rumelhart, 1981; Rumelhart & McClelland, 1982) to serve as the lexical route and input system for the DRC model. Like the IAM, input to the system is represented as a set of features that feeds into letter channels. Coltheart and Rastle (1994) use eight channels instead of the four used in the IAM. Like the IAM, the letter nodes activate word nodes. The nodes containing the spelling of the stored words is collectively called the *orthographic lexicon*. Each word node's resting activation level is a function of the word's printed frequency—high-frequency words are more highly activated than low-frequency words. Each word in the orthographic lexicon is connected to a node in a phonological output lexicon that represents the sound pattern for a word. Because the orthographic and phonological representations, or nodes, are linked, the phonology of a visually presented word is activated automatically when the word is looked up. When the phonological pattern of the word has been activated, phoneme nodes at an output level are activated where they await articulation.

The nonlexical route. The nonlexical route is a collection of rules used to translate graphemes (letters or letter combinations) into sounds. Rather than building grapheme-to-phoneme conversion (GPC) rules into the nonlexical route, Coltheart et al. (1993) allowed the model to discover the rules on its own by exposing it to the spelling and sound patterns of about 3000 words. Each rule derives a single sound from a grapheme.

Each time a grapheme is coupled with a single sound, the model includes the relationship in its rule base and updates its tabulated frequency. The rules are divided into three general categories: beginning (B), end (E), and medial (M) rules: the first grapheme of a word is translated to sound by a B rule, the last phoneme is derived by the E rule, and the phonemes in between are generated by M rules.

Some of the GPC rules are context sensitive. For example, consider the words *ham* and *harm*. The phoneme associated with *a* is different for each word, and the model must decide which sound to associate with it. One strategy would be to choose the most frequent grapheme-phoneme pair (in this case, *a*'s sound in *ham*). Doing so, however, would cause every word with the *ar* letter combination to be pronounced incorrectly. Coltheart et al. (1993) pointed out that there are 60 instances of *ar* words in the training corpus alone. The alternative strategy, was to allow the pronunciation of *a* to change when it is followed by an *r*. Hence, some GPC rules are context sensitive in that sometimes a letter's pronunciation depends on the letters also contained in the string.

Finally, Coltheart et al. (1993) allow the GPC route to consolidate rules. Recall that GPC rules are categorised into beginning, end, and medial rules. The GPC route allows any rule belonging to two categories to be included as an instance of the third. For example, the grapheme *oo* can occur as an argument to a medial or end rule in English words, for example, as in the words *pool* and *igloo*. The letter combination, *oo* does not occur as an initial grapheme of any word in their training corpus. Consequently, a nonword such as *oop* cannot be translated by the nonlexical route unless *oo* is also allowed to be a beginning rule. Consolidating rules, then, allows the GPC route to translate letter strings containing a beginning letter combinations that did not occur during training.

How does the nonlexical route use the GPC rules to derive a response? Translation proceeds in a left-to-right direction starting with the largest multi-letter rule that maps onto one phoneme (the largest grapheme is four letters in length). If no applicable rule can be found, the last letter is dropped from consideration and the search starts over again. The process continues until an applicable rule can be found. Once found and executed, the process begins again for the untranslated portion of the letter string.

Output of the DRC. At the output level of the model, the DRC has six phoneme slots. Each slot represents one phoneme of the word being pronounced. Each slot contains 44 phoneme nodes. Phonemes are activated one at time as the GPC rules translate a letter string from left to right. Phonemes are activated in parallel as the lexical route cycles to look up the correct pronunciation. To pronounce a word, one phoneme in each phoneme slot must exceed a criterion level of activation. The nodes within each slot are mutually inhibitory, hence, two nodes competing for activation will slow the increase in activation for the more active phoneme.

The tasks

The naming task. To name a word, the lexical and nonlexical route operate in parallel to derive a response. Because the two route operate together, the output of both routes simultaneously affect the activation of nodes in the phoneme system.

When the DRC names a regular word, the lexical and nonlexical route provide the phoneme system with the same phonological information. When a word is irregular, however, the lexical and nonlexical routes provide divergent evidence for a phoneme's pronunciation. When the two routes yield different output, the phoneme corresponding to the irregular phoneme in the phoneme system is prevented from reaching criterion. The phonemic activation created by the lexical route is, of course, the correct response When a phoneme's activation is held below its criterion value because of competition, the DRC allows the lexical route to continue cycling to help all the phonemes reach criterion.

Hence, the naming disadvantage for irregular words reflects the additional work that the lexical route must do to bring the correct phonemes to their criterial activation.

Sometimes the nonlexical route will bring the phonemes in each slot to their critical activation before the lexical route has had much chance to influence phoneme activation. When the nonlexical route brings all the phonemes to criterion prematurely, the model makes a *regularisation error*; that is, it pronounces a word like *wad* to rhyme with *bad*.

Nonwords are read easily by the DRC. When the model is presented with a list of nonword, the GPC rules of the nonlexical route translate the letter string's spelling into sound.

The lexical decision task. The DRC simulates the lexical decision task (deciding whether a letter string is a word) by searching for a letter string in the orthographic lexicon that matches the input letter string. The search continues until the word's node activation reaches threshold, or until a deadline has been met. If a word node reaches threshold, the model responds, "yes". The time limit for the search, expressed in the IAM component's cycles, is imposed on the model such that if no word node reaches the critical activation level after the deadline, the model responds "no". The deadline for search is adjusted from trial to trial to ensure variability on the finishing times for lexical decisions.

Problems with the DRC

Dual-route theories of word identification hold a privileged spot in word recognition research. In fact, competing models have done little more than show that they are at least as good as the dual-route model. However, the DRC inherits a flaw from the IAM. By building the DRC around the IAM, the DRC inherits the IAM's flawed representation of space.

To the further detriment of the dual-route model, the interaction between word frequency and spelling-to-sound regularity in the naming task can also be simulated by connectionist models (e.g., Seidenberg & McClelland, 1989; Plaut, et al., 1996). Connectionist models account for the interaction between a word's frequency and phonological regularity using a single mechanism that translates spelling to sound.

The positional sensitivity of regularity effects documented by Coltheart and

Rastle (1994) would appear problematic for connectionist models of word recognition—how would a model that operates in parallel mimic sequential effects? Plaut et al. (1996) claimed that the sensitivity to positional irregularity is likely due to the number of words that share the irregularity at each of the positions. They claimed that more words share the same letters with a regular pronunciation at the beginning of the word than at the end of the word. As a demonstration, Plaut et al. showed that their connectionist model yielded the same pattern of naming times as the DRC simulation.

The necessity of a GPC route is questionable. I concede that the nonlexical route is likely essential for the beginning reader learning how to attach sounds to the letters of novel words. However, the skilled reader has tens of thousands of words in her lexicon from which to generate a viable pronunciation of a novel word or nonword. A new word's pronunciation can be derived by analogy to words that the reader in already knows (Glushko, 1979). The ability of generalising to new stimuli is a strength of the connectionist models because they read both known and novel stimuli by analogy.

The final criticism considers the relative time course required for direct retrieval from the lexicon and the algorithmic translation of a stimulus to a response. Reading is a rapid, highly-learned activity. Theories of automaticity suggest that the use of algorithms to derive a response is generally required to process novel information. Once the input and output are familiar, a response can retrieved from memory automatically (e.g., Logan, 1988). Considering the skill that people have generalising to stimuli that are similar to learned material, it seems unlikely that an algorithmic route to pronunciation is an efficient strategy for a skilled reader. The criticism is especially salient given that the DRC applies the GPC rules on a trial-and-error basis. Applying the GPC rules on a trial and error basis should take a long time.

Chapter 4 : A New Approach to Word Identification

In this section I will make three arguments. First, I will argue that word identification or lexical access is more reasonably considered an example of retrieval from memory rather than an operation of looking up a word in the mental lexicon. I will also outline a computational model of memory based heavily on Minerva2 (Hintzman, 1984; 1988) that can be adapted to serve as a lexical memory system. Second, I will argue that the research in tachistoscopic letter research provides evidence that a *non-spatial* organisation of letters is required for lexical access. Finally, I will argue that retrieval of a word from memory is constrained by the organisation of the data that is used as a retrieval cue.

Word identification is a form of memory retrieval

The two principle tasks used by reading researchers, the naming and lexical decision tasks, can be viewed as modifications of cued recall and recognition memory tasks. In cued recall, a subject might learn a list of associated items, for example, *dog - a*, *cat - g*, etc. After learning, a subject is shown one member of the pair as a retrieval cue like *dog*, and asked to report its associate, *a*. Similarly, in the naming task, a printed string of letters can be viewed as a cue for the recall of the word's associated phonology. In the recognition memory task, a subject learns a list of items (usually words). After learning, the subject is shown items one at a time and asked to respond "yes" if that item was in the study set, and "no" if the item was not. In the same way for the lexical decision task, the subject must decide whether a letter string is among the letter strings stored in the reader's memory / lexicon. The only distinction between reading experiments and typical memory experiments is the type of memory that the tasks use—memory experiments test subjects' knowledge for what they *remember*; reading experiments test subjects' knowledge for what they *know*.

A memory retrieval approach to reading runs counter to one widely held assumption regarding lexical access. In the IAM (Rumelhart & McClelland, 1981), and DRC model of word identification (Coltheart et al., 1993), words are stored as independent nodes that are activated when the appropriate feature and letter detectors have been excited. Similarly in the connectionist models, nodes devoted to word components are activated when the right input is presented to the models. Each class of model treats the lexicon as a content addressable system. That is, lexical access involves finding a stored word's address(es). By contrast, when reading is considered a problem of memory retrieval, lexical access involves asking the memory system to return evidence that a particular word is in memory (c.f., Ratcliff, 1978). The distinction between the two approaches is that by accumulating evidence for a word's presence in the lexicon, the system is not obliged to actually find the word.

A strength of connectionist models is their ability to use stored information to create a response to novel inputs. Responses to learned and novel stimuli are based on a weighted combination of the neural connections representing the items that the system has learned. Put other way, responses are made by averaging the data contained in the system. Data averaging is also a common method of retrieving information from computational memory model such as Minerva2 (Hintzman, 1984). While I agree that extracting information from memory is done by averaging the data in the system, a memory retrieval account of lexical access differs from a connectionist account in one important way. In a connectionist model, the data to be averaged are the mappings between the input and output patterns it has learned. By contrast, a model like Minerva2 does not store mappings—it stores the input and output patterns in memory. Hence, a retrieval account of word identification, while not entirely inconsistent with connectionist notions of data averaging, assumes that the averaging is done on materials the model knows, not on mapping relationships that have been learned.

Most importantly, treating reading as memory retrieval challenges current ideas about what skilled reading actually is. Current models of word identification generally assume that reading a word aloud requires deriving sounds from letters. That is, letters are treated as graphic representations of sound patterns, and the reader must decode letters into the sounds they represent. I argued earlier that the naming task is a special example of the cued recall task; the letters of a word serve as a retrieval cue for the phonology of the word. What does this mean to the notion that a word's pronunciation is derived from print? Recall the example I used above where, given the retrieval cue, *dog*, the subject is required to report the associate, *a*. The response, *a*, cannot be derived from the cue. The *a* is retrieved from memory because, when the subject retrieves *dog* from memory, its associated information is also retrieved. Both objects are part of the same memory trace. Extending the idea to word identification, the phonology of a word is not derived from the letters of the word. A word's orthography and phonology are simply associated within one lexical entry. Reading a word aloud involves retrieving a match for the cue (i.e., the letters) from memory. When the cue has been retrieved, so has its associated information, (i.e., the word's phonology). In sum, we do not translate letters into the sounds they represent (indeed, letters are not graphical representations of sounds), phonological information falls out of memory when we retrieve a match to the cue.

Starting with the notion that reading is a memory problem, I used a global memory model to serve as an architecture in constructing a working model of word recognition. I chose to adopt a basic architecture similar to Hintzman's (1984; 1988) Minerva2 global memory model.

A Simple Memory Model

I can represent an experience or memory trace in a model as a vector of features. In the example here, I will represent the features as random integers between -1, and +1. A +1 may represent the presence of a feature, and -1 the absence of a feature. Traces are stored separately to form a matrix. To retrieve information, a probe, also represented as a vector of features, "resonates" with each memory trace. The similarity between the probe and memory trace is calculated by the formula:

$$S_i = \frac{\sum P_j \times T_{i,j}}{N}$$

Where S_i is the similarity of trace *i* to the probe. N is the number of pairs of features being compared. P_j represents the *j*th feature of the probe, and $T_{i,j}$ represents the *j*th feature of the *i*th memory trace. Each trace is activated by or resonates with the probe as non-linear function of its similarity. In Minerva2, for example, activation (A) of a trace is measured as:

$$A_i = S^3$$

To retrieve an item from memory, the features of each trace are multiplied by its activation and summed across traces using the formula:

$$E_j = \sum A_i \times T_{ij}$$

The final formula yields a vector that is a noisy composite of the probe vector. Hintzman (1984; 1988) refers to this composite vector as the *echo content* from memory (I will also adopt the term echo content later to describe the output from the lexicon).

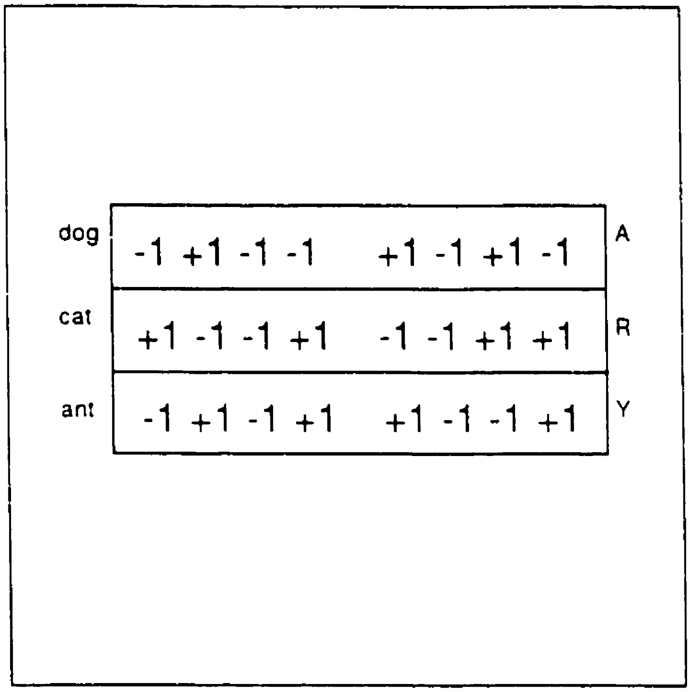

Figure 4.1 Three paired associates stored as a matrix of binary features. The first four features of a trace represent a word. The last four features represent its associated letter.

A convenient property of the model is that a trace can be subdivided to represent several dimensions of an item. For example, half of the features of a trace might represent a word, and the other half may correspond to an associated letter. Subdividing a vector allows the model to simulate cued recall. Consider the example where the model learns three associate pairs: *dog - a*, *cat - r*, *ant - y*. Each pair would be represented as a representation of the items in matrix form. When the model is probed with a vector containing the features of the word *dog*, it retrieves *dog* and its associate, *a*, by calculating trace activation from the similarity of the probe to the corresponding features of each trace. Each feature of the trace is multiplied by the trace's activation (Figure 4.2). When the activated features of the traces are summed, a composite vector representing DOG and A are contained in the echo content (Figure 4.3).

The model I describe in the chapter 5 uses an adaptation of the above model as a lexical memory system. As in the paired associate learning example above, each memory trace represents a word in memory. Half of the features of each trace represent the spelling or orthography of the word. The remaining features represent the word's phonology. Presumably, we could further subdivide a word's vector representation to include features relevant to the motor commands required to pronounce a word, or even a

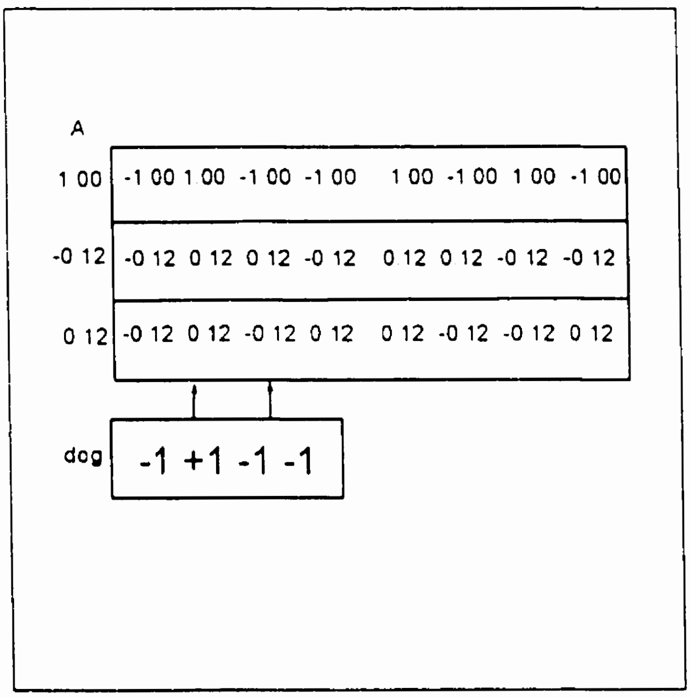

Figure 4.2 The same items as in Figure 4.1 after they have been activated by the features of the word *dog* as a probe. The numbers down the left side of the figure are the activations of each trace.

featural representation of meaning. In this thesis, however, we limit the lexicon's information to orthography and phonology because the scope of the thesis does not include an account of phenomena related to research in speech production or semantics.

Letters require encoding prior to lexical access

Current models of word recognition are mute with respect mechanisms involved in letter encoding. In fact, letter encoding is a stage in the reading process that is rarely considered. Much of the early work on letter encoding was done using tachistoscopic letter identification experiments. Out of this research, Mewhort and his associates (e.g., Mewhort, 1974; Mewhort & Campbell, 1981; Mewhort & Beal, 1977, Feldman-Stewart, 1992) proposed a theory for the initial stages of word recognition.

In their experiments (e.g., Mewhort & Cornett, 1972) subjects were briefly shown letter strings and asked to report as many letters as possible. Subjects typically showed a strong familiarity effect such that they could report more letters from a display that closely resembled a word (e.g., POLICKET) than one that did not (e.g., PRGEIDE). Further, and most importantly, report typically followed a left to right order. Mewhort (1974) postulated that the tendency towards a left-right report was the result of a mechanism, the *scan*, that loaded a short-term memory buffer with the letters of a string in a beginning-to-end order.

To explore the notion of scanning, Mewhort (1974) used a sequential presentation technique to gauge the familiarity effect and tendency for left-to-right report. Mewhort presented eight-letter pseudowords, one letter at a time, in either a left-to-right or right-to-left direction across the display screen and varied the intra-letter interval (ILI). He also presented the pseudowords in forward (e.g., POLICKET) or reversed spelling (e.g., TEKCILOP). When pseudowords were presented from left-to-right, subjects reported the letters in a left to right order regardless of the ILI. When the pseudowords were presented from right-to-left, subjects reported the letters from left-to-right at extremely short ILIs, and at long ILI's, the order of report matched the order of arrival on the display. The familiarity effect also depended upon the direction of presentation and ILI. Subjects showed a strong familiarity effect for forward-printed pseudowords at all ILI's when presented from left to right. When these pseudowords were presented from right to left, however, there was a familiarity effect at only the shortest ILI's. Reversed pseudowords exhibited a familiarity effect only when presented from right to left at the longest ILI's. Taken together, the pattern suggested that the familiarity effect depended on a beginning-to-end encoding, or ordering, of the letters.

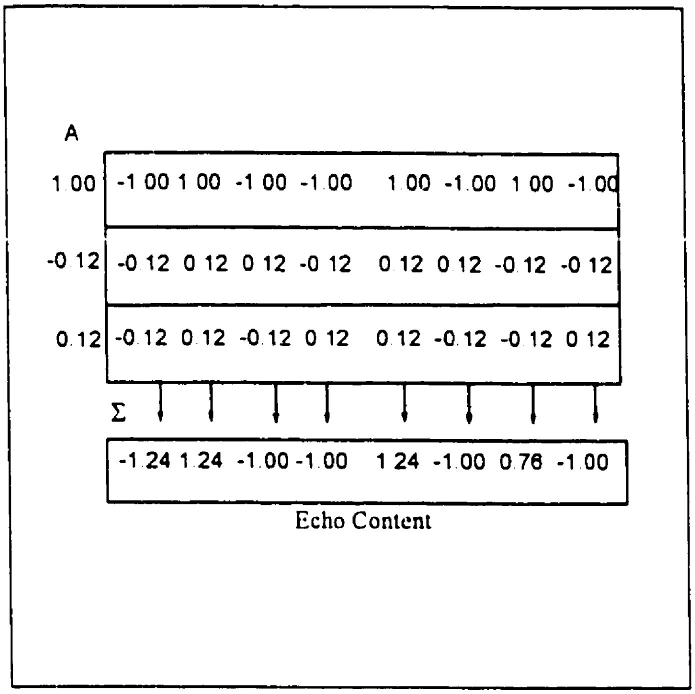

Figure 4.3 When features are summed across the activated traces, an *echo content* is created.

When a sequential presentation is rapid (e.g., a 10 ms ILI), the scan can proceed without disruption regardless of the direction of presentation; that is, all the letters are scanned into the short-term memory buffer. When the transfer is disrupted by a slow sequential presentation, the order of report is forced to reflect the order of arrival on the display.

To corroborate the notion that scanning is an obligatory part of identifying a word, Mewhort and Beal (1977) repeated Mewhort's (1974) sequential paradigm using eight-letter words. They found that the probability of identifying a word in the task mapped closely onto the size of the familiarity effect using pseudowords (n.b. Mewhort, 1974). In sum, word identification requires the ordering of the letters from beginning to end.

From these data, Mewhort and Campbell (1981) postulated a model for the initial stages of word identification called the Dual-buffer Model. According to Mewhort & Campbell (1981; see Feldman-Stewart, 1992 for a formal account) the raw features of letters are stored in a *feature buffer* upon presentation of a string of letters. Letters are identified in parallel from the features and stored as spatially arranged abstract forms in a labile storage mechanism called the *character buffer*. From the character buffer, the scan loads the letters into a temporal buffer where they can be rehearsed and/or chunked. The order of encoding is determined by the direction in which a language is written. Hence, encoding proceeds in a left to right direction for words made up of Roman characters, and the opposite for the letters of a language, such as Hebrew, that is read from right to left (Butler, Tramer, & Mewhort, 1985).

Ordering letters prior to lexical access solves a problem common to several formal models of word recognition. As I discussed above, several models have difficulty representing the spatial arrangement of the letters in a word (e.g., Coltheart et al., 1993, McClelland & Rumelhart, 1981, Seidenberg & McClelland, 1989; Johnson & Pugh, 1994; Mozer, 1991). For example, consider a model wherein the word *pat* is represented by the activation of nodes corresponding to the letters *p*, *a*, and *t*. Without reference to their spatial organisation, the letters are consistent with the spelling of the words *pat*, *tap*, and *apt*. In the previous chapters, I outlined some solutions to this problem (i.e., wickelfeatures, componential representation, and letter channels). I also mentioned earlier that there is no evidence for the psychological reality of any of these representation schemes. It is therefore unclear whether they offer a reasonable solution to the problems associated with representing spatial organisation within a letter string—especially when successful word recognition depends on them. On the other hand, if letters are converted into a list by the scan, space is not an issue.

A version of Mewhort's dual-buffer model, the Letter Processing System (LEPS), was recently formalised by Feldman-Stewart (1992). LEPS, illustrated in Figure 4.4, begins at an artificial retina and terminates at the level where identified characters are stored in the character buffer. My model of lexical access uses LEPS as a front end. That is, the output of LEPS is used as the input to my model. Using LEPS as a front end to my model is desirable for two reasons. First, I assume that research in psychology is cumulative; models of higher processes can be built on top of existing models of lower processes. Second, because LEPS begins its processing at an artificial retina, adding a retrieval system for lexical information provides a fairly complete account of lexical access.

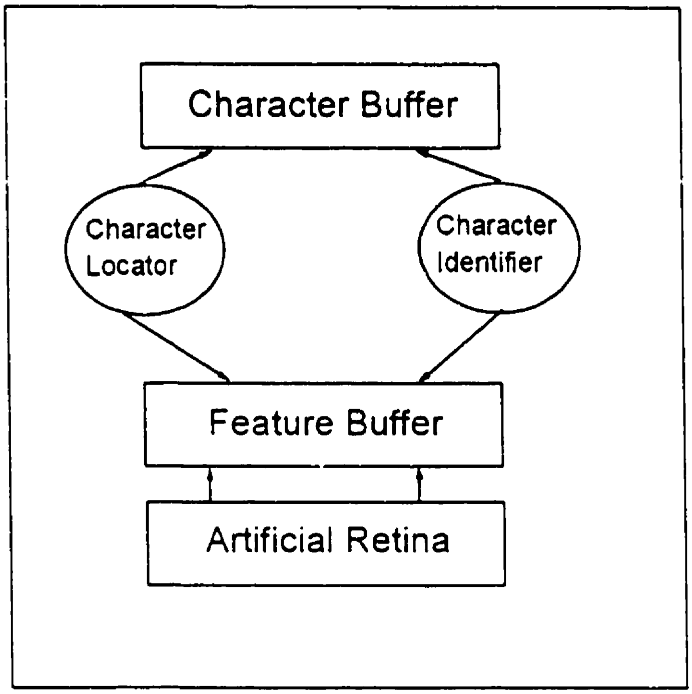

Figure 4.4 Sketch of Feldman-Stewart's (1992) model of letter identification (LEPS)

Processes in letter encoding determine the mode of lexical access

I use abstract letter identities as a retrieval cue or probe for the words stored in memory. The notion that encoded letters are ordered raises an interesting issue for retrieval. Current models of word identification are built on the premise that lexical access is an example of parallel processing. In one sense of the term *parallel processing*, the models assume that the letters of a word are processed simultaneously during lexical access. A model that embodies the assumption of parallel processing must use an input representation that is amenable to parallel processing. Hence, choosing wickelfeatures (e.g., Seidenberg & McClelland, 1989), letter channels (e.g., McClelland & Rumelhart, 1981) or word components (Plaut et al., 1996) as input representations is a consequence of the theoretical framework in which the model was constructed.

I take a different view of model building as it relates to the choice of input representation. Instead of choosing an input representation that is amenable to assumptions about how lexical access occurs, I argue that one must choose a strategy for lexical access that is amenable to the organisation of the letters that are used as an input. When we have an idea of how identified letters are organised, we can begin to consider how, given the organisation, information is extracted from the lexicon. Because we have evidence that a list of letters is used for lexical access, I chose an retrieval method that capitalises on the order of the letters within the list. Specifically, I assume that lexical access starts with the leftmost letter of the list and proceeds down the list until the last letter has been retrieved from the lexicon.

Chapter 5 : The Theory "LEX"

In what follows I describe the details of the model. It is illustrated in Figure 5.1. I treat word identification as a three-stage process. At the first stage, the characters of the display are identified. At the second stage, identified characters serve as a cue for the retrieval of lexical information from lexical memory. The final stage constitutes the generation of a response based on the information that has been retrieved. When LEX names a word, I assume that the quality of retrieved phonological information dictates when articulation will begin. When LEX decides on the lexical status of a letter string in the lexical decision task, the quality of the orthographic information retrieved from the lexicon determines the speed of the response, and whether LEX will be biased to accept a letter string as a word, or reject it as a nonword.

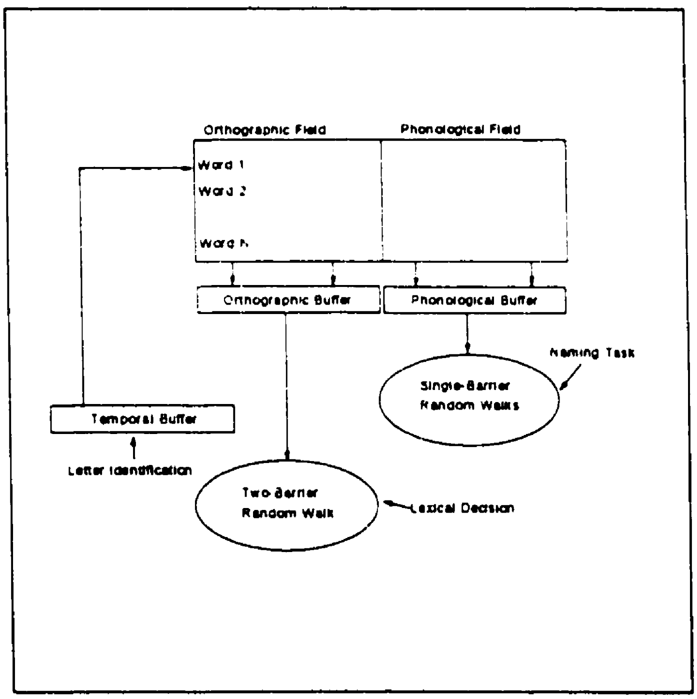

Figure 5.1. Basic architecture of LEX.

LEX embodies the second and third stages of the word identification process. That is, letter identification has already occurred prior to the point at which LEX starts processing. For a detailed description of the model I have chosen as a letter identification system, see Feldman-Stewart (1992).

An informal description of LEX

Before I give a detailed account of how LEX works, I will give a verbal description of how LEX names words and makes lexical decisions. When a printed word is presented to a reader, its letters are identified and stored as a spatial array in the character buffer (Feldman-Stewart, 1992; Mewhort & Campbell, 1981). From the character buffer, a scanning mechanism copies the contents of the character buffer into a temporal buffer. The scan copies the letters one at a time beginning with the first letter. The letters in the temporal buffer are stored as a temporal array, or a list, where they are used as a cue for retrieving a word from the lexicon.

To retrieve a word, LEX uses the list structure to guide retrieval. That is, retrieval begins with the first letter in the list. The first letter in the temporal buffer resonates with, or activates, the orthography and phonology of all the words in the lexicon to activate them. Because a word's orthography and phonology are part of the same memory trace in LEX, both dimensions of the word resonate with the probe. If the probe letter matches the first letter of a word in the lexicon, its activation is higher than if the letters mismatch. The magnitude of the word's activation also depends on the printed frequency of the word such that high-frequency words are more highly activated by the probe than low-frequency words. Once all the words in the lexicon are activated, the model can retrieve the information from the lexicon. One way to retrieve the information would be to collapse across all the words in the lexicon to yield a composite, or facsimile, of the target letter—the same retrieval operation I described in the section *A Simple Memory Model* in Chapter 4 wherein an echo content is created. Unfortunately, a retrieval method identical to the one described in Chapter 4 does not work. There are so many words in the lexicon that if LEX collapses across all the traces to find the first letter, all it retrieves is noise. The alternative is to implement the strategy embodied by other models of memory retrieval (e.g., Ratcliff, 1978). Specifically, LEX accumulates evidence for the target letter's presence in the lexicon over a series of time steps.

At each time step, LEX samples a handful of words from the lexicon at random. Each time a sample is taken, the echo content for the sample is copied into two buffers. The part of the echo content corresponding to the orthographic information within the sample is copied into an orthographic buffer, and the part corresponding to phonology is copied into a phonological buffer. For each sample, LEX measures how much the contents of the orthographic buffer have changed from the previous sample. When the change between samples reaches a criterion minimum, the search for the target letter is terminated.

If LEX finds the correct letter, it proceeds to retrieve the next one. After LEX has retrieved the first letter, it knows two things: what letter the target word begins with, and also what words the target cannot be. In other words, LEX uses both positive and negative evidence to find a word. For example, if LEX retrieved the letter *c* in the first position, it knows that the target word cannot be the word *apple*, but could be any word that starts with *c*. LEX uses negative evidence by adjusting its search space to include only words whose spelling is consistent with the identified letters of the probe. In other words, it adjusts a *cohort* of candidate matches to the probe. For example, after LEX retrieves the *c* of the word *cat*, it adjusts the cohort of candidate words to include only words that begin with the letter *c*.

LEX continues sampling and adjusting the cohort until it has retrieved the final character in the TB, the space character. However, sometimes retrieval fails and LEX settles on the wrong letter. Retrieval failure happens for two reasons: a letter may be retrieved incorrectly, or a letter of the probe might belong to a word that LEX does not know. In either case, LEX readjusts its search space and tries to retrieve the letter a second time. To readjust the search space, LEX drops the first letter of the probe so that it no longer resonates with the words in the lexicon. For example, if LEX failed to correctly retrieve the letter *t* of *cat*, it did so in a cohort containing only words that begin with *ca*. Following the failure, LEX would release the letter *c* from the probe leaving only the letter *a* to resonate with the words in the lexicon. Now, LEX would search for the letter *t* in a cohort of words that have the letter *a* as a second character. LEX usually requires only one cohort readjustment after a retrieval failure. However, if it does experience another failure, the next letter is dropped from the probe to further aid retrieval.

After LEX finishes retrieving the last character (the space character), the orthographic and phonological buffers contain features that correspond to the orthography and phonology of the letter string, respectively. It is important to note that retrieval is guided entirely by orthography—phonological information simply falls out of the lexicon as orthographic information is retrieved. Phonology falls out because it is part of the same memory trace as the orthography.

Once LEX has finished retrieving the letters, it can either name the word aloud or decide on its lexical status. When LEX names a word, the time required to begin pronunciation depends on how clear the phonemes in the phonological buffer are. If, during sampling, the phonemes of many words with irrelevant phonemes are included in the phonological buffer, the clarity of the phonology in the buffer is compromised. The less clear the phonemes are, the longer it takes to initiate pronunciation. When LEX makes a lexical decision, it evaluates how closely the contents of the orthographic buffer match the probe letters in the temporal buffer. If the contents of the two buffers are similar enough, LEX is biased to respond that the retrieved orthography is that of a word. If the similarity between the two buffers is low (as is the case after retrieval failures) LEX is biased to respond that the retrieved orthography is a nonword.

LEX's response time is calculated as the sum of two values. In the naming task, I sum the number of time steps, or samples, required to retrieve the orthography and the time required to initiate pronunciation. The time required to initiate pronunciation is taken as the time required to build a motor program to pronounce the blurriest phoneme. In the lexical decision task, I sum the number of required samples and the time it takes to decide on the letter string's lexical status. The time required to make the decision is taken as the finishing time of a standard two-choice evidence accumulation mechanism (Ratcliff, 1978).

Formal description of LEX

Knowledge

Creating a model of skilled reading based on the principles outlined in the previous chapter, required that I represent far more lexical knowledge than the few thousand words found in many current models. LEX knows the orthography and phonology of about 103,000 types. I used the Carnegie-Melon Pronunciation Dictionary (Weide, 1995) to serve as a lexicon. At first glance, 103,000 words might seem like too much knowledge (many compact dictionaries have less than half that number of entries, and, I imagine, few know the definitions of all the words contained in them). However, like humans, LEX knows several first names, surnames, street names, profanities, and expletives not found in a standard dictionary. Second, the model treats the affixed forms of a word as separate entries. For example, the various forms of the word *understand*, (i.e., *misunderstand*, *understanding*, *understandable*) are represented separately. Hence, the large number of lexical entries in the model is not unrealistic.

Representation

The basic units of representation in LEX are letters and phonemes. Each of the 27 letters (the space character delineating the end of a word is treated as a letter) and 40 phonemes (the fortieth phoneme is a null phoneme which also serves as a word delimiter. See appendix A for a listing of phonemes) are represented as a vector of 50 features. Features are random integers of -1's and +1's sampled from a rectangular distribution. LEX represents letters and phonemes as abstract identities in the sense that the features do not correspond to physical characteristics of the characters or phonemes.

A word's spelling patterns is represented in LEX by concatenating the letter vectors that spell the word. As well, the phonology of a word is created by concatenating the appropriate phoneme vectors. Each word is stored separately and contained within a 1800-feature vector. The vector can handle a word that has up to 18 letters or phonemes. The first 900 features of the word vector store the word's orthography. I refer to this half as the *orthographic field* of the word. The *phonological field* of the word, the final 900 features contains phonology of the word.

Each word is left-justified in its field. That is, the first letter and the first phoneme are placed in the first position of their appropriate fields. The final character of every word is the space character, and the final phoneme of every word is the null phoneme. To maintain equal dimensionality across all lexical entries, letter positions in the orthographic field not containing a character are assigned vectors of zeros, and phoneme positions in the phonological field not containing a phoneme are assigned a null phoneme vector.

LEX is a multiple-trace model—the number of instances of a particular word in its lexicon is a function of the frequency with which the word occurs in text. For example, LEX has more instances of the word *the* in the lexicon than instances of the word *apt*. I truncate the natural logarithm of the sum of one and a word's frequency (as tabulated by Kucera and Francis, 1967) as the number of instances for any word.

Storage Mechanisms

In addition to the lexicon, LEX has three short-term memory buffers, each of which is represented as a 900-feature vector. The *temporal buffer* (TB) receives the characters that have been scanned from the character buffer of the letter identification module (Feldman-Stewart, 1992). Whereas the characters in the character buffer are stored spatially, the characters in the temporal buffer are stored in a first-to-last order and used as a retrieval cue, or probe, for retrieving a word from the lexicon. The *orthographic* (OB) and *phonological buffers* (PB) store the orthographic and phonological information that is retrieved from the lexicon.

Retrieving information from the lexicon

Activation. LEX's retrieval operations begin where Mewhort and Campbell's (1981) dual-buffer model and LEPS (Feldman-Stewart, 1992) terminates. The output of LEPS is a spatial array of letters stored in a character buffer. From the character buffer, LEX scans the characters to the TB. The scan orders the characters from beginning-to-end.

LEX uses the order of the letters to guide lexical search. That is, lexical access begins with a search of the lexicon for a match to the first letter of the probe, and terminates with the retrieval of the last letter (always a space character). Every lexical entry is activated by the probe letter. The degree to which a lexical entry, L, is activated, A, by the probe letter, P, is a function of the similarity between the lexical entry's corresponding letter to the probe letter and its frequency. To get an activation value for a lexical entry, the measure of similarity is raised to the third power and multiplied by a number that is a function of the word's printed frequency (*F*) according to the Kucera and Francis (1967) word norms. Raising the similarity to the third power serves to accentuate the degree to which the activation of similar and non-similar letters resonate with the probe letter.

$$A_{L_j} = \left(\frac{\sum_{j=1}^{N} P_j L_j}{N} \right)^3 \times \ln(1 + F)$$

Where *j* indicates the *j*th feature of the letter's vector, and *N* indicates the number of features in a letter.

Words containing the matching letter will be activated more highly than any other lexical entry. Feature values of vectors for words not containing the matching letter are pushed below a minimum value of 1. It is important to note, that a probe letter activates whole words, not just the characters at the letter position at which it is searching. For example, whole instances of the words *cat* and *core* will be activated by the letter *C* when it is used to search for the first letter of the word *cat*.

Retrieval. After the entries in the lexicon have been activated, LEX can begin retrieving information. Retrieving information from the lexicon takes time. Following random walk theory (e.g., Ratcliff, 1978), retrieval in LEX involves the gradual accumulation of evidence for an item's presence in memory. Random walk theories measures the accumulation of evidence over time. LEX uses a sampling method for retrieval that is an instantiation of what random walk theories simulate. Retrieval in LEX is the gradual accumulation of evidence that the probe letter is present in memory. To accumulate evidence, LEX repeatedly samples available lexical information against a background of noise from words in the lexicon.

The probability of including any one word in a sample is a function of the word's frequency. Specifically, the probability of sampling any word is given by the formula:

$$P(sampling) = \frac{\ln(1+F)_j}{\sum^{a} \ln(1+F)_a}$$

Where $ln(1+F)_j$ is one plus the natural logarithm of word j's printed frequency as calculated by Kucera and Francis (1967). The denominator of the equation is the sum of $ln(1+F)$ across all words in the lexicon that are available for sampling. On each sample, or processing cycle, a composite vector, or *echo content*, is created from the words in the sample. The echo content is created in the same way as I described in Chapter 4. First, the features of each word in the sample are weighted by their activations. The echo content is created by summing corresponding features across the traces in the sample. The first 900 features of the echo content are copied into the OB. The final 900 features are copied into the PB. The formula for creating the echo content is

$$E_j = \sum_{j=1}^{n} A_i \times L_{i,j}$$

Where E_j is the jth feature of the echo content, A_i is the activation of the *i*th word, L, in the sample. In the simulations to follow, I used a sample size of 100 items. Smaller sample sizes tended to result in a speed-accuracy trade-off, and larger sample sizes introduced too much noise to the echo content.

LEX stops sampling the lexicon for a match to the probe when it settles on a letter. On each cycle during sampling, LEX calculates how much the contents of the OB have changed from the previous cycle. The change is measured only for the features of the OB that correspond to the position of the probe letter. For example, if LEX is probing the lexicon with the first letter in the TB, the change in the features of the first letter position in the OB is measured. LEX is considered to have settled on a letter when the difference in the correlation, as measured by Pearson's r, between the OB's features on cycle N and cycle N-1 reaches a minimum (a parameter of the model I set to 0.0008).

It is worth restating that the decision to stop sampling is made on the basis of how a letter vector in the OB changes on successive cycles, *not* on how similar the cue letter in the TB is to the echo content in the OB on successive cycles. LEX always settles on a letter; whether the correct letter has been retrieved is determined after the system settles.

After LEX stops sampling, the features in the OB corresponding to the retrieved letter are compared to the probe letter. LEX has settled on the correct letter if the correlation between the features of the OB and TB is higher than the correlation between the features of the OB and any other letter.

If LEX settles on the correct letter, the search space for the next probe letter is redefined by excluding the words that failed to match the retrieved letter. For example, finding an *c* for the first letter precludes any words not starting with *c* from the search space when LEX searches for the next letter. The search space is consistently readjusted until the final character, always the space character, is retrieved. In other words, at each letter, LEX defines a new *cohort* of candidate words

I borrowed the idea of reducing a cohort of candidate words from Marslen-Wilson's (1984) model of auditory word identification. He proposed on-line processing while listening to a spoken word. The processing narrows the range of possible words in real time as phonemes are delivered to the listener. Hence, the listener may know the meaning of the word before the utterance is complete. LEX performs a similar operation on the letters of a visually presented word.

My retrieval method can be justified on two grounds. First, I use sampling during retrieval to acknowledge that the adult's lexicon is large; so large that readers cannot think about all the words they know simultaneously. Second, cohort reduction acknowledges that the system can use both positive and negative evidence to identify a word. Retrieval starts with a small amount of evidence about the word's identity, but the same information provides strong evidence for what the word cannot be. If the first letter retrieved from the system is *c*, for example, there are a large number of words consistent with that fact. But, it is also clear that the target word cannot be the word *apple* or any other word that does not start with *c*.

The use of negative evidence in recognition has recently been studied by Mewhort and Johns (in press). They report several examples in which subjects use negative evidence to drive a response in a recognition memory task. For example, test items that contain a feature novel to the study set are easy to identify as new items in a recognition memory task.

When Retrieval Fails. When LEX retrieves the wrong letter it adjusts the cohort to make re-identifying the missed letter easier. LEX adjusts the cohort by excluding the first letter from the probe. When the first letter has been eliminated from the probe, the letter no longer resonates with the words in the lexicon. Lexical entries that were previously excluded from the cohort of candidate words become reinstated because they share letters with the remaining letters in the OB. If retrieval fails again, the next letter in the probe is eliminated. For example, suppose LEX mis-identified the *t* of the word, *cart* as an *s*, that is, LEX settled on *cars*. In trying to find *t*, LEX used a search space containing words with *c*, *a*, and *r* as the first letters. To try again on *t*, *c* is dropped from the letters of the probe. Now, any word with *ar* as the second and third characters are contained within the search space. If LEX fails again, the *a* is dropped leaving only words with an *r* in the third position.

When a letter is dropped from the probe, the letter in the corresponding position in the OB is ignored when the echo content is copied into the OB. The OB is a short-term memory buffer. Because the features of the first letter in the OB are no longer being reinforced by retrieval, I assume that its features decay mildly over successive samples. To simulate decay, I subtract a random value, taken from a rectangular distribution ranging from 0 to 0.1, from each feature of the ignored letter in the OB. However, the decay is mild, hence, there is no danger that the identity of the first letter will be lost by the time LEX retrieves the final letter. Introducing decay to the unprobed letters was also motivated by the finding that the word frequency effect in the lexical decision tends to be larger than in the naming task. In LEX, retrieval from the lexicon is a common stage to both the naming and lexical decision tasks. Hence, for LEX, the greater frequency effect in lexical decision reflects a difference in how much the retrieved orthography of a high- or low-frequency word resembles the probe letters in the temporal buffer. Whereas the retrieved orthography of a high-frequency word already tends to resemble the probe letters more closely the retrieved orthography of a low-frequency word does, the mild decay on the unprobed letters serves to accentuate the difference.

Retrieval failure also forces LEX to adjust the how phonemes are copied into the PB. If after adjusting the cohort, LEX continued to copy **all** the features of the echo content into the PB, phonemes in the beginning positions in the PB would be overwritten by those from irrelevant words in the echo content. Instead, LEX treats the first phoneme in the PB as correct and ignores the features corresponding to the first phoneme in the echo content when it copies the echo content into the PB.

How does LEX know where to begin copying phonemes into the PB when retrieval fails? There are two possibilities. If the second phoneme in the PB has near perfect clarity, LEX continues copying phonemes at the second phoneme. If, on the other hand, the second phoneme is not near perfect, LEX continues copying phonemes from the echo content at the phoneme preceding the blurriest phoneme. Why the two different strategies? The clarity of the phonemes after retrieval is higher when LEX reads words than when it reads nonwords. Hence, if after retrieval failure, the second phoneme is pristine, LEX has evidence that the letter string it is reading is a word and, as such, producing a pronunciation by analogy from the second phoneme will yield an accurate pronunciation. On the other hand, if the second phoneme is not near perfect, LEX has some evidence that the letter string is a nonword. If LEX has some evidence that the letter string is a nonword, new phonemes can be copied into the PB wherever the clarity of the blurriest phoneme can be increased.

It should be clear at this point that, for LEX, lexical access is the search for a letter string's orthography in the lexicon. No currency is placed on the phonology of the word during retrieval. Building a phonological representation necessary for naming a word occurs as a corollary to finding the orthography—phonological information is retrieved automatically when an activated word has been sampled. In short, phonological information is retrieved from the lexicon because it is associated with orthographic information; a notion clearly at odds with the popular idea that reading aloud involves mapping letters or letter combinations to sounds. Ignoring phonological information during word identification is also at odds with claims that phonological information is used by the reader at an early stage in word identification (see Lukatela and Turvey, 1994a, 1994b). My position is that readers do not generally use phonological information at the beginning stage of lexical access. However, because phonological information is retrieved early, I believe a reader could exploit it.

Measuring the Time to Retrieve Information from the Lexicon

Retrieving information from the lexicon takes time. To measure lexical retrieval time, I count the number of cycles it takes to retrieve all the letters from the lexicon. After retrieval, LEX uses the retrieved information to generate a response.

Measuring the Time to Make a Response After Information Retrieval

Once lexical information has been retrieved from the lexicon, LEX can make a response. To perform the naming task, LEX uses the information contained in the PB. In a lexical decision task, LEX bases its response on the information stored in the OB.

An account of the articulatory mechanisms involved in naming a word, and the decision mechanisms involved in making a lexical decision is beyond the scope of this thesis. To acknowledge the point and to simulate response time after retrieval from the lexicon, the quality of the representation in the PB and OB determines the speed of the response. When the information in the OB and PB is blurry, or unclear, LEX requires more time to generate a response than when there is little ambiguity within the information.

Simulating the Lexical Decision Task

In a lexical-decision task, subjects are asked to decide whether or not a letter string is a word. I assume, along with others (e.g., Andrews, 1989; Forster & Shen, 1996) that the lexical decision task requires a decision stage after lexical access. LEX decides on the lexical status of a letter string by comparing the contents of the OB to the TB. The comparison yields a similarity value that measures the quality of match between the two buffers. If the match exceeds a criterion, LEX is biased to consider the retrieved orthography to be that of a word; otherwise it is rejected as a nonword. Because the lexicon contains only words, it less likely to obtain a good match to a nonword.

In the decision stage, LEX accumulates evidence for or against a string's lexical status. Evidence for either response is calculated as the difference between a criterion match and the match of a small sample of pairs of features from the two buffers. The difference is summed over several iterations until a criterion amount of evidence is accumulated. The accumulated evidence can be positive (where LEX decides that a letter string is a word), or negative (where LEX decides that a letter string is a nonword).

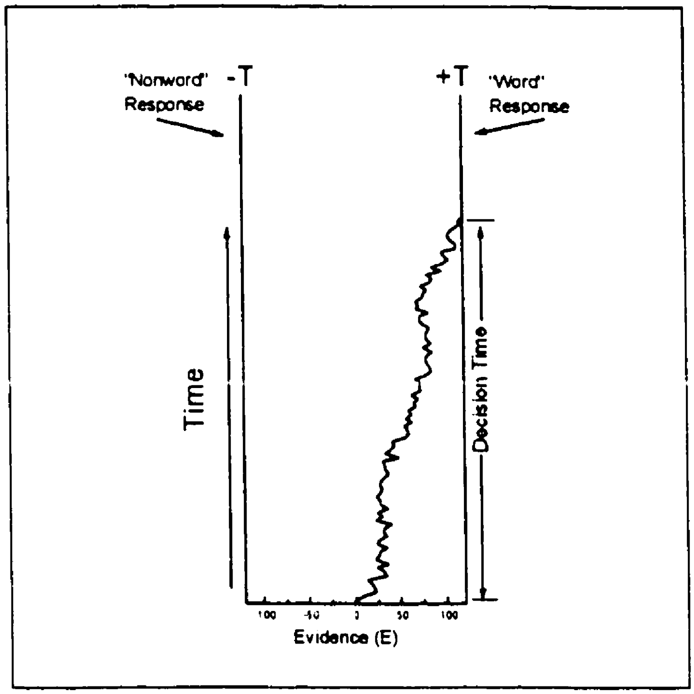

Figure 5.2 A diagram of the gradual accumulation of evidence over time in a two-barrier random walk.

Because the decision mechanism I have described is computationally expensive, LEX uses a two-barrier evidence accumulation mechanism, the random walk (Ratcliff, 1978; Link, 1975; 1991; Link & Heath, 1975) to simulate decision time (DT) for the lexical decision task. The random walk was designed to explain response time and accuracy in a two-alternative, forced-choice task. The random walk is illustrated in Figure 5.2. Random-walk theory postulates that, to make a two-alternative, forced-choice decision, evidence for one response or the other must accumulate over time. Accumulation continues until there is enough evidence to make a response. Using the random walk as a decision mechanism is functionally equivalent to the method I described above for LEX's decision stage, but it has one advantage. An analytic expression is known for the random walk so that I can calculate the expected value of the finishing time instead of implementing the full stochastic decision process.

The evidence for one response over the other is expressed as a signal, s. When s is negative, it is evidence for one response, and evidence for the other response when s is positive. To accumulate evidence, the signal value is summed across subsequent time slices, or steps. Accumulation continues until the evidence equals or surpasses a barrier or threshold, T. Because the random walk simulates a two-choice decision, there are two barriers, -T and +T, one corresponding to each decision.

Evidence accumulation is a noisy process—at each time step, it is accumulated against a background of noise. The addition of noise at each step causes the accumulated evidence to deviate randomly from a straight course to one of the two barriers.

Noise is introduced to s on each cycle by adding a gaussian deviate from a distribution with a mean of 0 and a standard deviation of σ. Hence, at each step, the amount of accumulated evidence can be calculated as:

$$E_t = E_{t-1} + s + N(0, \sigma)$$

Where E_t represents the accumulated evidence at time t, s corresponds to the signal value, and $N(0, \sigma)$ corresponds to a random deviate sampled from a gaussian distribution with a mean of 0 and a standard deviation of σ.

Once a letter string's orthography has been retrieved from the lexicon, a signal for a random walk is calculated as a function of the similarity between the letters used as a probe and the retrieved orthography. Similar to the technique used by Seidenberg and McClelland (1989), LEX considers the retrieved orthography to be a word when the correlation between the features of the OB and the TB exceeds a minimum, C. If the value fails to reach the minimum, LEX considers the letter string to be a nonword.

To simulate the decision and its latency, I calculate a signal for the random walk by subtracting the value from C, i.e.,

$$s = r(OB, TB) - C$$

(where r is Pearson's product moment correlation). If the value exceeds C, the random walk has a positive signal; that is, evidence that the letter string is a word. The signal is negative when the value fails to reach C, and is taken as evidence that the letter string is a nonword.

The C parameter is LEX's only free parameter. I allow it to vary to acknowledge that readers can adjust how carefully they decide on the lexical status of a letter string. For example, consider how readers might change how they make lexical decisions when the nonword foils in the lexical decision task are strings of random consonants, e.g., *ghtk*, versus foils that closely resemble words, e.g., *lave*. When readers are presented with nonword foils that do not resemble words, Andrew's (1989) demonstrated that decision latencies decrease compared to the case where word-like foils were used. From LEX's perspective, because a string of random consonants does not closely resemble any word in the lexicon, the correlation between the contents of the OB after retrieval and the contents of the TB will be low relative to the case wherein the nonword foils are very word-like. I assume that, when the decision about a letter string's lexical status is made easier by using un-wordlike foils, the correlation between the contents of the OB and TB does not need to be very high to correctly accept letter strings as words. On the other hand, when the nonword foils closely resemble words, the minimum similarity between the two buffers must be higher. Hence, when the correlation is high for nonword foils, LEX requires a higher more evidence to decide that the letter string is a word.

One barrier of LEX's random walk corresponds to a *nonword* response (set to -30 in LEX), and the other to a *word* response (set to +30). The gaussian distribution of noise that I used had a mean of 0 and a standard deviation of 0.5. Decision time (DT) is measured as the number of steps the random walk takes to accumulate enough evidence to cross one of the two barriers.

In the simulations that I report in Chapter 6, I calculated the expected DT and probability of an error for each trial from the signal values. Using expected values instead of actually running a random walk has two advantages. First, direct calculation is computationally cheaper than waiting for random walks to finish. Second, the expected DT is a less noisy estimate of finishing times. The expected DT for a signal value is calculated by the formula:

$$E(DT) = \frac{T}{s} \times (1 - (2 \times p_{\varepsilon})$$

Where p_{ε} is the probability of an error and is calculated as

$$p_\epsilon = \frac{1}{e^{\frac{2sI}{\sigma^2}}}$$

LEX's average response time (retrieval time + DT) across trials (*MLDT*), is calculated as the mean of the expected DTs + letter retrieval time (*LRT*) weighted by their probability of being correct,

$$E(MLDT) = \frac{\sum^{i}(E(DT)_i + LRT_i) \times (1-p_\epsilon)_i}{\sum^{i}(1-p_\epsilon)_i}$$

Simulating the Naming Task.

In the naming task, subjects are shown a letter string and asked to pronounce it as quickly as they can. LEX models the naming task by reading off the phonemes that are copied into the PB during retrieval.

The phonemes in the PB are not generally pristine following retrieval. Each phoneme in the PB contains features irrelevant to the correct pronunciation. Irrelevant features occur, of course, because irrelevant words were sampled during retrieval. I assume that articulation begins after a pronunciation program has been created by the reader. To create the program, the retrieved phonology must first be de-blurred so that a pristine copy of each phoneme can be used to create a pronunciation. I assume further, that the time it takes to begin creating the pronunciation program depends on how long the phonemes will take to de-blur.

I assume that the phonemes are deblurred in parallel. Hence, the time to it takes to pronounce a word following retrieval depends on how long it takes to clean up the blurriest (non-null) phoneme. De-blurring occurs by first having the phonemes in the PB act as a probe to activate a set of canonical phonemes. The activation (A) of a canonical phoneme (c) is a function of its similarity to the phoneme in the probe (p). Specifically,

$$A_i = \left(\frac{\sum p_i \times c_i}{\sqrt{\sum p_i^2 \times \sum c_i^2}}\right)^5$$

Following the activation of the canonical phonemes, a facsimile of the probe is created by taking the weighted (θ) sum across the features of the activated canonical phonemes using the formula:

$$E_j = \sum A_j \times c_{j,i} \times \theta$$

Each facsimile is weighted (set to 0.01) so that the clean up process occurs gradually over several cycles. The facsimile is used as a probe to the canonical phoneme set again to create another echo that is copied on top of the previous one. Each time the echo is used as a probe, the activations of the canonical phonemes change such that the phoneme most similar to the contents in the PB increases to approach 1 while the activations of the others approach zero. The de-blurring process stops when the activation of one of the phonemes in the canonical set reaches a criterion (set to 0.99). At the point where the criterion is met, the canonical phoneme most similar to the contents of the PB is the only one that is active; as well, the facsimile created from the canonical set is a near perfect copy of the canonical form. The winning phoneme is used in the creation of a motor program for pronunciation.

The de-blurring process is computationally expensive. Hence, I simulated the time required for de-blurring and articulatory program creation by using the clarity of the phonemes in the PB as a signal to drive a single-barrier random walk (SBRW) mechanism. To express the clarity of each phoneme in the PB, I measure the correlation between its features in the PB and each possible phoneme. The phoneme to which the features of the PB is most highly correlated is the phoneme that LEX will pronounce. The magnitude of the correlation reflects the degree to which the features of irrelevant phonemes are also present at that position. Hence, if a phoneme vector is most highly correlated to the */ah/* sound, we know two things: First, we know which phoneme the model is storing in that position, and second we can express the clarity of the phoneme as a function of the magnitude of that correlation.

Like its two-barrier counterpart, a SBRW accumulates evidence over successive time steps against a background of noise until it reaches a barrier. As well, noise during evidence accumulation is acknowledged by adding a deviate from a normal distribution on each time step (I used a distribution with a mean of 0 and a σ of 5 as a noise distribution). Response time is measured as the number of steps required for the evidence to reach the barrier (a parameter I set to 300). Because there is only one barrier, the SBRW does not make errors; it simply yields a latency estimate for a given signal value.

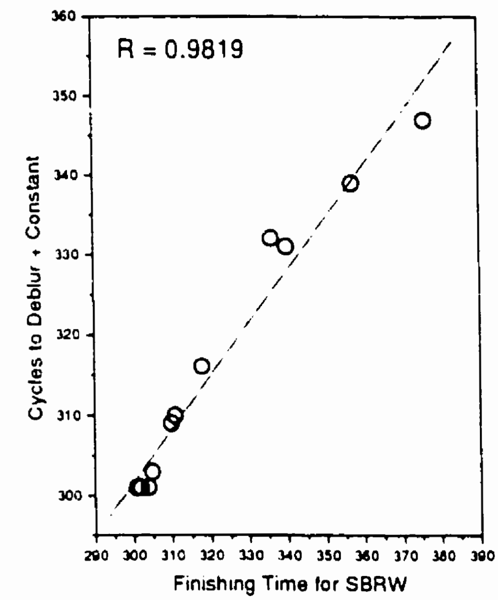

Figure 5.3. Finishing times of the SBRW plotted against the finishing times for de-blurring.

LEX has one SBRW attached to each phoneme in the PB. Each SBRW uses the clarity of its phoneme to derive a signal to the walk. To calculate the signals, each phoneme's clarity is raised to the 7th power. That is,

$$S_i = C_i^7$$

Where S is signal value for the *i*th SBRW, derived from the clarity ,*C* of the *i*th phoneme.

The clarity of the phonemes for words tends to range between 0.900 and 0.999, a range generally too small for latencies from the evidence accumulator to exhibit strong sensitivity to differences in phonemic clarity. The large exponent accentuates differences in clarity among the phonemes, making the SBRW, and of course, naming latency, more sensitive to changes in clarity.

A SBRW for each phoneme is started simultaneously. Naming begins when the PB's blurriest phoneme has been included in a program for articulation. Hence, naming latency, following lexical retrieval, is taken as the finishing time of the slowest SBRW.

Unlike the calculations for DT for the lexical decision task, I do not calculate expected values for naming latency. Instead, I run one SBRW using the signal derived from the blurriest phoneme and take its finishing time as an estimate for naming latency after lexical retrieval.

Mapping De-blurring time to the SBRW. Of course, using a SBRW to estimate response preparation time after retrieval requires that I demonstrate that its finishing times map closely on to the finishing times for the de-blurring mechanism. To demonstrate that the SBRW is an useful tool in estimating response preparation time, LEX named 11 words for which the phonemes in the PB varied in clarity. Figure 5.3 plots the finishing times of the de-blurring mechanism for the words (measured in number of samples of the canonical phoneme set) against the corresponding finishing times for the SBRW. A constant (300) was added to the de-blurring times to reflect the time it takes to generate the articulation program after de-blurring. As is clear in the figure, there is a very close relationship between the two.

Response Latency for a Trial

For LEX, there are two stages in word identification: letter retrieval, and response generation/selection. Both stages take time. The retrieval stage is common to both the naming task and the lexical decision. To estimate the response latency for a trial in which LEX names a word, I sum the number of samples required to retrieve orthographic information from the lexicon and the number of cycles it takes the SBRW to reach threshold. To generate a response latency for a trial in which LEX makes a lexical decision, I sum the number of samples required to retrieve the word and the expected finishing time of the two-barrier random walk.

Summary

In this chapter, I provided a detailed account of LEX, a model of visual word identification. LEX retrieves words from the lexicon beginning at the first letter of a probe stimulus. As it retrieves each letter, LEX reduces its search space by creating a cohort of candidate words. The process continues until the final letter (always a space character) has been retrieved. Because the orthography and phonology of lexical items are stored in single memory traces, as letters are retrieved out of the lexicon, so are phonemes.

Response latency in LEX is estimated by summing two values: The number of samples required to retrieve lexical information, and the time to initiate a response. In the lexical decision task, a two-barrier random walk simulates decision time. One barrier of the random walk corresponds to a *word* response, and the other to a *nonword* response. In the naming task, the finishing time of the slowest in a group of single-barrier random walks is taken as an estimate of how the time required to build and begin the execution of an articulation program.

Chapter 6: Relating the Theory to Data

This chapter presents some tests of LEX using archival data. Before applying LEX to individual experiments, there are several procedural consideration that deserve comment. The first concerns the parameterisation of the model; A second concerns the scope of the model.

The simulations will, except as noted, use the same parameters throughout. The archival data are based on experiments that manipulate classes of words without retraining subjects. To be consistent with that strategy, it would be inappropriate to adjust the parameter of the model to fit the data. Phenomena that emerge empirically by changing classes of items should fall naturally out of the model.

The size of LEX's vocabulary, the familiarity of the words within it, and processes that LEX uses to identify words are invariant across runs of the model. Presumably, all three factors differ across readers. Because the three factors are fixed in LEX, successive runs of the model are, in effect, data from the same subject. Because retrieval is a stochastic process, there is variability across runs. That variability represents trial-to-trial variability for a single subject. The simulation data described here were obtained by averaging across 16 independent runs of the model.

In the chapter, separate sections are devoted to each of several phenomena along with a brief description of it. The description includes a simulation and explanation derived from LEX. Where possible, the description also includes an account of how the dual-route and connectionist models would explain the phenomenon.

One final point about the simulation data is worth making. The model has two mechanisms contributing to response latency. To get a response time, I sum the finishing times for both mechanisms. Because the mechanisms are differentially affected by the characteristics of words, the size of the main effects in graphs showing an interaction can differ. Clearly, what must be done to the output of the model is a relative weighting of the contributions of each mechanism to response latency.

Overall Performance of LEX

In the sections to follow, I will demonstrate that LEX does a good job reproducing the phenomena considered important by the reading literature. As a test of LEX's ability to capture readers' response latencies for words in the naming and lexical decision tasks, I directly compared LEX's response times to subjects' response times.

All the simulation data for the naming task were obtained from a frozen model. That is, none of the parameters were changed across runs. I allowed one parameter, the word criterion parameter, to vary across runs for the lexical decision task to reflect the difficulty of the decision as a result of the structure of the nonword foils. Allowing the one parameter to vary also brings the predicted error rates to a reasonable level. Figures 6.1 and 6.2 plot LEX's mean response latency against subjects' mean response latency for the words every cell across all the experiments in the simulations to follow. Figure 6.1 plots the relationship for naming latency, and Figure 6.2 plots it for lexical decision latency. As is clear in both figures, LEX's response times can be mapped onto human response times.

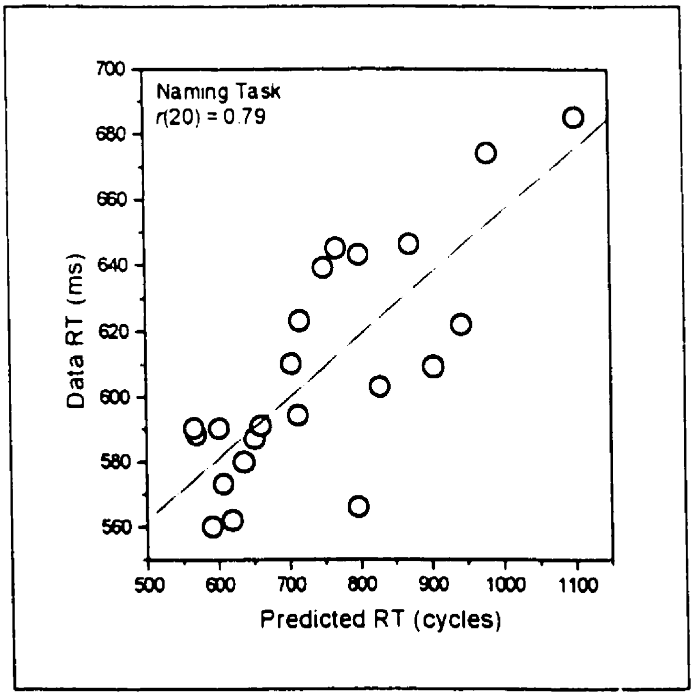

Figure 6.1 Mean naming latencies from each cell of each experiment plotted against LEX's mean naming latency for the same cells

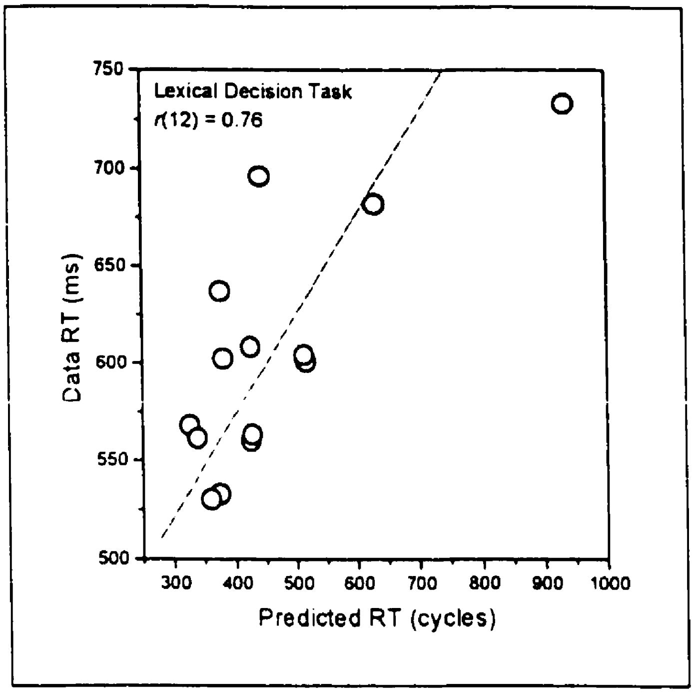

Figure 6.2 Mean lexical decision latencies from each cell of each experiment plotted against LEX's mean naming latency for the same cells

The Word-Frequency Effect

Words that occur frequently in text are identified more quickly than words that occur less frequently. The *frequency effect*, as it is called, is one of the most widely cited and replicated phenomenon in word-recognition research.

The DRC's account

The DRC model (and by association, the IAM) explains the frequency effect in terms of the activation of word nodes in the orthographic lexicon. The resting activation of each word node in the orthographic lexicon is a function of the word's frequency in print. Nodes representing high-frequency words have a larger resting activation than nodes for low-frequency words. To identify a word, the activation of a word node is updated over successive processing cycles. On each cycle, the activation is increased for words that are consistent with the target letter string. Word nodes are mutually inhibitory; that is, as a node's activation increases, it inhibits other word nodes. Given extended processing, the inhibition implies that only one word node will remain above threshold. Because high-frequency words have a high resting level of activation, they require fewer processing cycles to reach threshold than nodes for low-frequency words.

The Connectionist's account

Plaut et al. (1996) and Seidenberg and McClelland (1989) do not use word nodes to represent word. Instead, word knowledge is confined to mappings between an input (orthographic) and output (phonological) layer of nodes. In both models, the frequency effect is explained in terms of the strength of the connections between the input and output nodes.

During training, letter patterns are paired with sound patterns. Each time a particular letter pattern is paired with a sound pattern, the strength of the connections between the nodes increases. High-frequency words are presented during training more often so that a word's frequency in print is represented by greater training. Hence, the connections that pair the spelling and sound of a high-frequency word are stronger than those for a low-frequency word.

A production account

Balota and Chumbley (1985) provide a different kind of account. They argued that lexical access is only partially responsible for the frequency effect in the naming task. They claimed that the motor program required to pronounce a low-frequency word takes longer to compile and execute than one for a high-frequency word. In support of their argument, Balota and Chumbley showed subjects high- and low-frequency words and required them name then aloud, but to wait until cued, a so-called delayed-naming task. They varied the delay between the words' onset and the cue randomly between 1000 and 2400 milliseconds (ms). Even long after lexical access would have taken place, there was an advantage for high-frequency. Their results point to a pronunciation factor that presumably contributes to the frequency effect. Neither the DRC nor the connectionist models can accommodate Balota and Chumbley's data.

LEX's account

LEX uses word-frequency data at two points during retrieval. First, the probability that a word is sampled from the lexicon depends on its frequency. Secondly, when the similarity of the probe to the word in a sample is calculated, it is weighted by the word's frequency. For both reasons, retrieval for high-frequency words is faster than for low frequency words. An example that illustrates LEX's frequency effect is illustrated is Figure 6.3. For the simulation data shown in Figure 6.3, LEX retrieved the orthography of the high-frequency words in a mean of 53 cycles. Low-frequency words, on the other hand, took a mean of 124 cycles.

The naming task. Pronunciation begins when a motor program has been compiled for the phonemes in the PB. Recall that motor program initiation time is based on the clarity of the phonemes. Figure 6.6 shows the average blurriness of the blurriest phoneme for high- and low frequency words. As is clear in the figure, the blurriest phonemes in the PB for high-frequency words have higher clarity than those for low-frequency words. A motor program to pronounce a low-frequency word will take longer to build and execute for than for a high-frequency word. LEX's two loci for the word frequency effect is an important and unique characteristic of the model. Because the phonemic clarity serves as a locus for a naming advantage for high-frequency, LEX can explain the advantage in a delayed-naming task.

The lexical decision task. To decide on the lexical status of a letter string, LEX must decide if the retrieved orthography is similar enough to the probe letters to be called a word. After retrieval, the orthographic representation of low-frequency words in the OB is less similar to the probe (r = .965) than the OB's representation of high-frequency words (r = .995). The difference in clarity occurs for three reasons. First, because the system requires more samples to retrieve the orthography of low-frequency words, more irrelevant letters are included in the featural representation stored in the OB. Second, letter retrieval is more likely to fail when the system is retrieving the letters of a low-frequency word. When letter retrieval fails, the cohort of candidate words is adjusted. The adjustment *also* increases the number of irrelevant words from which the system samples. Finally, I assume that when retrieval fails, the letters contained in the echo that correspond to the positions of the already retrieved letters are not copied into the OB. After a retrieval failure, I introduce a mild decay to the letters in the OB that have already been retrieved. In sum, the frequency effect in the lexical decision task not only reflects the time it takes to retrieve information from the lexicon, it also reflects how well the retrieved information matches the probe.

The Interaction Between Word Frequency and Spelling-to-Sound Regularity

In the naming task, words that violate spelling-to-sound correspondence rules (i.e., irregular words such as WAD) take longer to name aloud than words that do not (i.e., regular words such as BAD). This is only true, however, for low-frequency words. Regular and irregular words are named with about the same latency when they are high-frequency (Seidenberg, Waters, Barnes, & Tannenhaus, 1984; Taraban & McClelland, 1987).

Spelling-to-sound regularity affects response time only when subjects are asked to read words aloud. In the lexical decision task, subjects exhibit a reliable frequency effect, but the words' regularity has no effect on performance. The right panels of Figure 6.1 and 6.3 show the typical pattern of data for the naming task and LDT respectively. The data are taken from mean latencies reported by Seidenberg *et al.* (1984).

The DRC's explanation

The Naming Task. The DRC interprets the interaction between word frequency and regularity in terms of competition between the lexical and nonlexical routes. Recall that the speed with which the nonlexical route can derive a pronunciation using its grapheme-to-phoneme pronunciation rules is a function of the number of letters in the string; the more letters there are to translate, the longer it will take to create a pronunciation. The speed with which the lexical route activates a word node and its pronunciation depends on it resting level of activation. When the model reads a word aloud, the lexical and nonlexical routes operate simultaneously to derive a response.

The lexical and nonlexical activate phonemes in the phoneme slots at the output level of the model to await pronunciation. When the DRC reads a low-frequency word, the two routes derive a pronunciation at about the same speed. If the two routes derive divergent pronunciations, i.e., when the system is reading an irregular word, the DRC must decide which of the two pronunciations is the correct one. The extra time required to resolve the conflict caused by the divergent phonological codes causes low-frequency irregular words to be named more slowly than low-frequency regular words. When the DRC is reading a high-frequency word, the lexical route activates the phonological code for a word much faster than the nonlexical route can derive it. Hence, at the level of the phoneme slots, there is no conflict between phonological codes to resolve and regularity does no affect naming latency.

The Lexical Decision Task. In the lexical decision task, the DRC searches the orthographic lexicon to see if the stimulus letter string is present. If it is present, the model responds that the string is a word. If, after a predetermined deadline, the letter string has not been found in the lexicon, the model considers the letter string to be nonword. Because, deriving a phonological code is not required for the lexical decision task, regularity has no influence on the response.

The Connectionist's Explanation

The Naming Task. Connectionist models learn words by associating letter clusters with phoneme clusters over several training sessions. The speed with which the models identify a word is a function of how many times the clusters were paired during training. High frequency words are named more quickly than low frequency words because the clusters in high-frequency words are associated more strongly than those in low-frequency words.

A word's regularity affects naming latency in the same fashion. There is a quasi-regular mapping between letter clusters and phoneme clusters in English. Irregular words exemplify situations in which a frequent mapping does not apply. For example, the *ave* letter combination is most frequently associated with the phonemes in words such as *gave*, *save*, *rave*, and *cave*. The combination *ave* is also associated with the phonemes in *have*. Because the letter cluster is more frequently associated with the phoneme clusters in regular words, naming an irregular word is more difficult to read than a regular word.

The interaction between word frequency and regularity in the naming task occurs because of the relative frequency with which the letter clusters are associated with phoneme clusters. The *ave* letter cluster is associated with more regular words than irregular words, however, *have* is a high-frequency word that will be encountered by the model many times during training. Hence, despite *have*'s divergent mapping to phonemes, the mapping is frequent, making it, and other high-frequency words, insensitive to a word's regularity in the naming task.

The Lexical Decision Task. To make a lexical decision, a Seidenberg and McClelland's (1989) model uses a set of letter clusters as an input. The output is also set of letter cluster nodes. If the input stimulus is a word, the two sets of letter clusters will match closely. If the two sets of letter clusters are not similar enough for the model to consider it a word, it is labelled a nonword.

Because the mapping between letter and phoneme clusters is irrelevant in the lexical decision, irregular spelling to sound relationships do not affect response time. Hence, while connectionist models show a string word frequency effect in the lexical decision, the advantage for low-frequency regular words over low-frequency irregular words is absent in the lexical decision task.

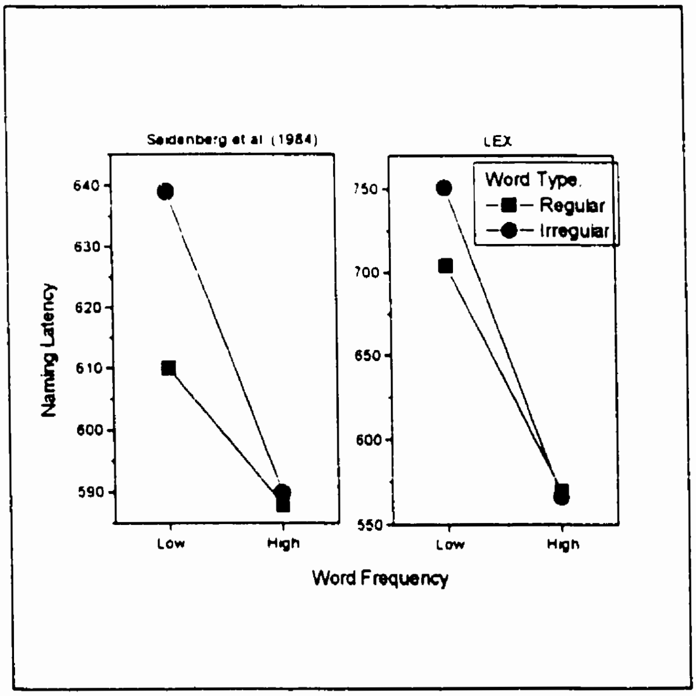

Figure 6.3. Naming latencies for subjects (left panel) and LEX (right panel) for the items used by Seidenberg et al. (1984)

A Simulation

In the simulations to follow, LEX named and made lexical decisions on the 52 words used by Seidenberg et al. (1984). LEX also named the 96 words used by Taraban and McClelland (1987). There were four types of word in their lists: low-frequency regular, low-frequency irregular, high-frequency regular, and high-frequency irregular. The proportion of error for each simulation is shown in Table 6.1.

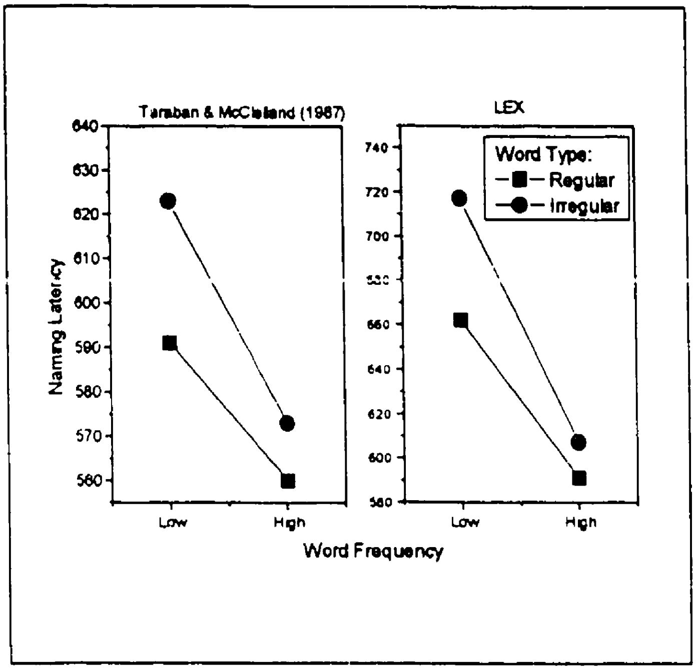

Figure 6.4. Naming latencies for subjects (left panel) and LEX (right panel) for the items used by Taraban and McClelland (1987)

The right panels of Figures 6.3 and 6.4 show LEX's mean naming times, in cycles, for each type of word in Seidenberg et al.'s (1984) and Taraban and McClelland's (1987) experiments. The left panels of Figures 6.3 and 6.4 show the accompanying latency data from subjects. As is clear in the figures, LEX replicates the naming advantage for regular words over irregular words when they are low frequency. High-frequency words exhibit no naming advantage for regular words over irregular words.

The right panel of Figure 6.5 shows LEX's mean response times for lexical decisions to the words used by Seidenberg et al. (1984). The left panel contains subjects' data for the same words. In both panels, there is a clear advantage for high-frequency words over low-frequency words, but no advantage for regular words over irregular words.

Table 6.1

Proportion of errors for regular and irregular words in each simulation (Name = naming task, LDT = lexical decision task)

	Word Frequency	
Experiment	Low	High
Seidenberg et al. (1984), Name		
Regular	0	0
Irregular	0.1	0
Seidenberg et al. (1984), LDT		
Regular	0.1	0
Irregular	0.03	0
Taraban & McClelland (1987)		
Regular	0	0
Irregular	0.01	0.01

Why does it work?

I will consider the naming and lexical decision tasks separately

The Naming Task Recall that, after LEX retrieves the letters of a word, the clarity of the phonemes retrieved from the lexicon dictates how quickly pronunciation will begin. In the previous section, I explained why LEX is able to capture the advantage for high-frequency words over low-frequency words—word frequency affects both the number of samples required to retrieve a word from the lexicon, and the clarity of the phonemes that are retrieved.

The naming advantage for low-frequency regular words over low-frequency irregular words occurs because of the difference in clarity between the phonemes of the two types of words. The clarity of the phoneme responsible for the spelling-to-sound irregularity tends to be much lower than its corresponding phoneme in a yoked regular word, and the other phonemes of the word (see Figure 6.6). Figure 6.7 shows the clarity values for each phoneme in the words *pint* and *mint*. Notice that the clarity of the second phoneme of *pint* is much lower than that for its corresponding phoneme in *mint*. The decrease in clarity occurs for two reasons. First, an *i* following a *p* is most frequently pronounced as /i/ as in *pin*, *picture*, *pit*, *pill*, etc.

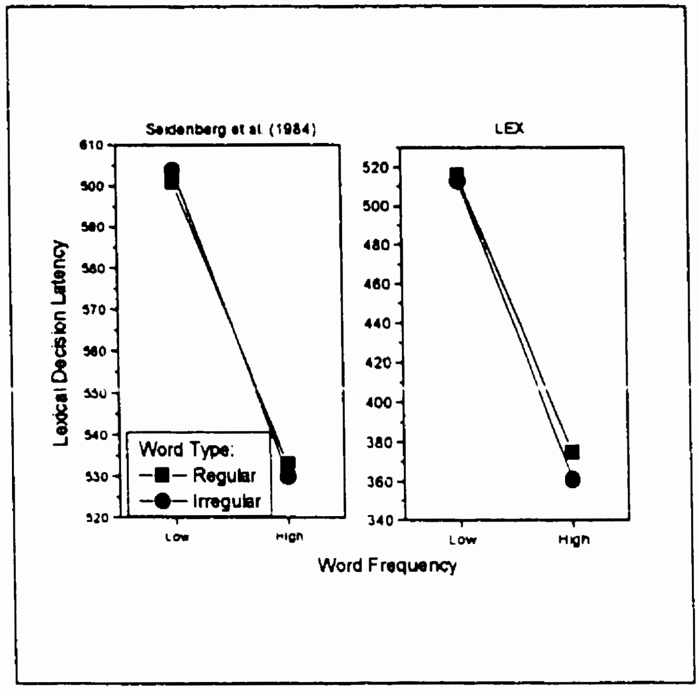

Figure 6.5 Lexical decision latencies for subjects (left panel) and LEX (right panel) for the items used by Seidenberg et al. (1984).

Hence, when the second phoneme is switched to the /I/ of *pint*, the features of /i/, already in position from previous samples, interfere with those of the /I/ phoneme. The second reason for the decreased clarity of the second phoneme occurs when LEX must restart its search for a letter after access has failed. Recall that, when LEX is forced to restart its search, a pronunciation is built as an analogy to other words whose spelling are consistent with the target starting at the second letter. *Pint* is the only four-letter word with the *int* letter string that does not rhyme with *mint*. Hence, a pronunciation derived by analogy will cause the clarity of the second phoneme to suffer leading to increased naming time, and possible mispronunciations of the word.

When LEX retrieves a high-frequency word, however, its phonemes tend to be quite clear. The increased clarity associated with high-frequency words is the result of the high resting level of excitation that high-frequency words possess. When LEX retrieves the phonemes of a high-frequency word, inconsistent phonemes from low-frequency words will have little impact on the clarity of the output relative to the

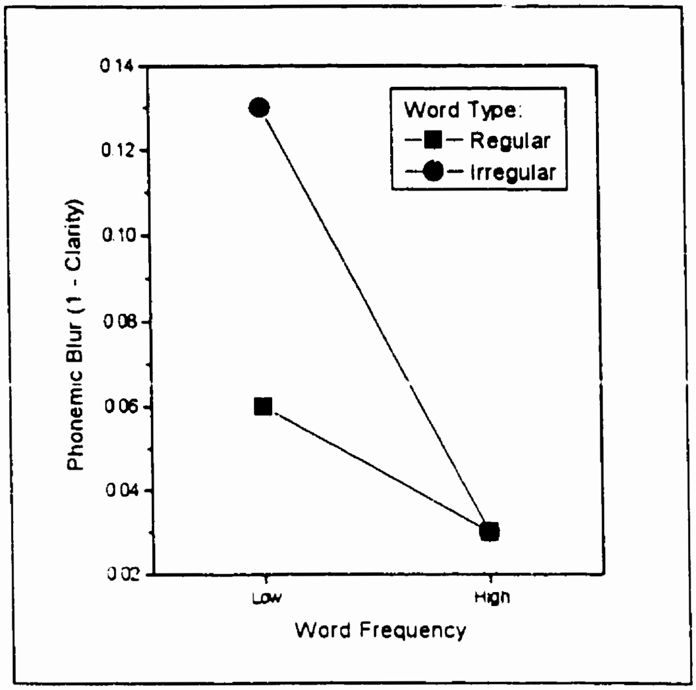

Figure 6.6. Average clarity of the blurriest phonemes for each type of word in Seidenberg et al.'s materials.

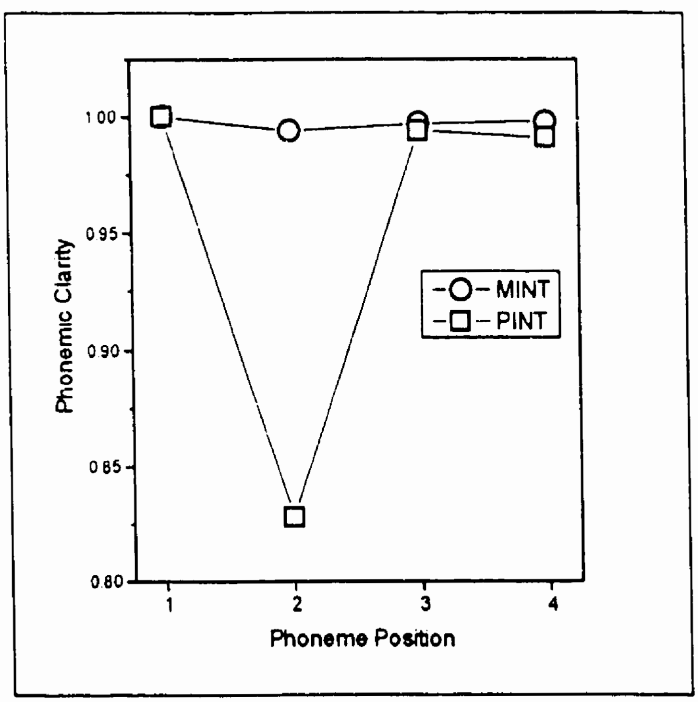

Figure 6.7. Clarity of the phonemes for the words *pint* and *mint*

large impact that the phonemes from high-frequency words will have on the clarity of the phonological representation—the influence of phonemes from high-frequency words far outweighs any impact that inconsistent phonemes may have on the output.

My account of the interaction between word frequency and regularity is in a strange position with respect to where it differs from the DRC and connectionist accounts. The DRC postulates two strategies for pronunciation; my account, like the connectionist account requires only one. The connectionist account considers word frequency and regularity effects in the naming task as the same effect, i.e., regularity effects are frequency effects. My account, like the DRC, places responsibility for frequency and regularity effects on separate mechanisms

The Lexical Decision Task. In the lexical decision task, the clarity of phonological information is irrelevant to the response that LEX makes. Instead , LEX retrieves the letters from the lexicon using the target letters as a retrieval probe. Once the letters have been retrieved, LEX decides if the retrieved letters are a good match to the target. The extent to which the retrieved letters are a good match to the probe letters is a match of the orthographic clarity of the retrieved letter string. Phonological irregularity does not affect the clarity of the retrieved orthography, hence, it has no effect on response time in he lexical decision task.

My account of the lexical decision differs greatly from the DRC account. The DRC searches the lexicon for a word's address until a deadline has been met. If a word has not been found by the deadline, the DRC considers the input to be a nonword. By contrast, LEX does not look for words; it retrieves information from the lexicon and decides if that information is good enough to be considered as belonging to a word.

The connectionist account of Plaut et al. (1996) did not make lexical decisions. Hence, my discussion of the connectionist account of the lexical decision task will be limited to the model by Seidenberg and McClelland (1989). My account is similar to Seidenberg and McClelland's account in that the decision as to whether a letter string is a word is based on how well the system can reproduce a facsimile of the original letter string. The decision has nothing to do with finding the target word in the lexicon.

The connectionist account and my account differ in that LEX treats the similarity of the probe letters to the retrieved orthography as part of the processing required for the lexical decision task. Absent from Seidenberg and McClelland's model is a process by which creating the facsimile takes time. The lexical decision is a two-stage process in LEX. First, A facsimile of the original letter string is created over successive samples of lexical information. After retrieval, LEX decides whether the facsimile is word-like enough to be considered a word.

Positionally Sensitive Regularity Effects in Word Naming

Coltheart and Rastle (1994) demonstrated an important constraint on the naming advantage for low-frequency regular words over low-frequency irregular words: The influence of irregularity on naming latency decreases as the position of the irregular phoneme, counting from left to right, nears the end of a word.

The DRC's Explanation

Coltheart and Rastle (1994) reported the positional dependence of the naming advantage for low-frequency regular words over low-frequency irregular words as evidence for the necessity for the DRC's grapheme-to-phoneme conversion route during reading aloud. They proposed that the decreasing naming advantage for regular words reflects a decrease in the conflict between the outputs of the lexical and nonlexical routes as the irregular phoneme position nears the end of a letter string.

Recall that the lexical and nonlexical routes operate simultaneously to derive a pronunciation for a letter string. Translating graphemes to phonemes via the nonlexical route takes time. If the irregular phoneme is positioned near the end of a word such as *memoir*, the lexical route will sometimes have enough time to activate the correct pronunciation before the irregular phoneme is translated—effectively avoiding a conflict between the outputs of the two routes. On the other hand, if the irregular phoneme is positioned at the beginning of a word such as *chic*, conflict between the two routes is unavoidable; the nonlexical route will decide on a different pronunciation for the first grapheme before the lexical route is able to look up the correct pronunciation.

The Connectionist's Explanation

A massively parallel account of regularity effects in word naming does not predict that the position of an irregular phoneme will affect on the magnitude of the naming advantage for regular words over irregular words. Plaut et al. (1996) offered other interpretations of Coltheart and Rastle's (1994) result.

First, Plaut et al. (1996) argued that the position of a word's regularity may be confounded with the degree to which words had a consistent pronunciation. For example, the *ch* in the word *chic* is highly inconsistent because the /*sh*/ phoneme is found in only five out of 63 monosyllabic words that start with *ch*. A word like *tomb*, on the other hand, is only moderately inconsistent: *tomb* has one orthographically similar word that shares the same body, *womb*, and only two that do not (*comb*, and *bomb*).

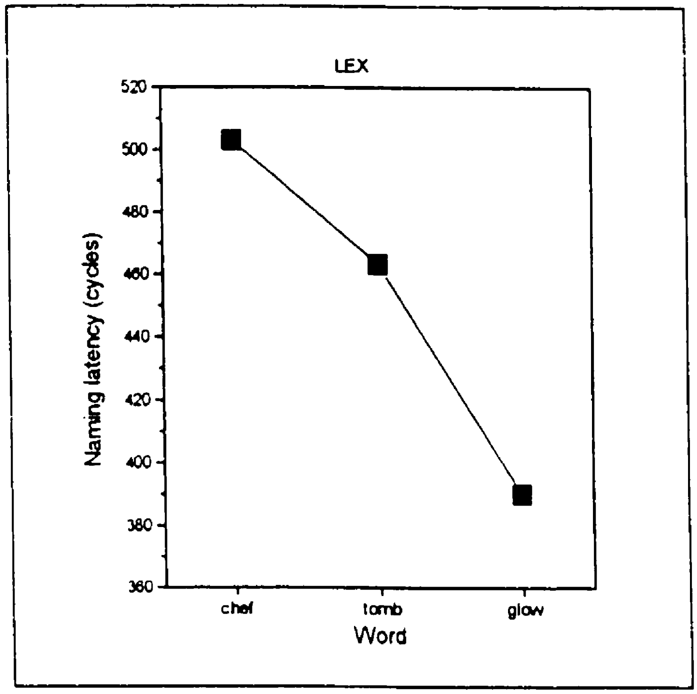

Figure 6.8 LEX's naming latencies for the same three words used in Coltheart et al.'s (1993) and Plaut et al.'s (1996) simulation.

Plaut et al. (1996) also objected to defining irregularity in terms of GPC rules. The DRC considers the letter combination, *age*, of *bandage*, and the *ive* of *festive* as irregular. Plaut et al. pointed out that most two-syllable words with *age* or *ive* endings are not pronounced with a long vowel. It is questionable, therefore, whether such words can be considered irregular.

Finally, Plaut et al. (1996) addressed the possibility that Coltheart and Rastle's effect was genuine. Plaut et al. speculated that the phenomenon might not be inconsistent with a parallel account of word naming if the time required to initiate articulation depends on properties of the initial phonemes after parallel generation of the phonological code.

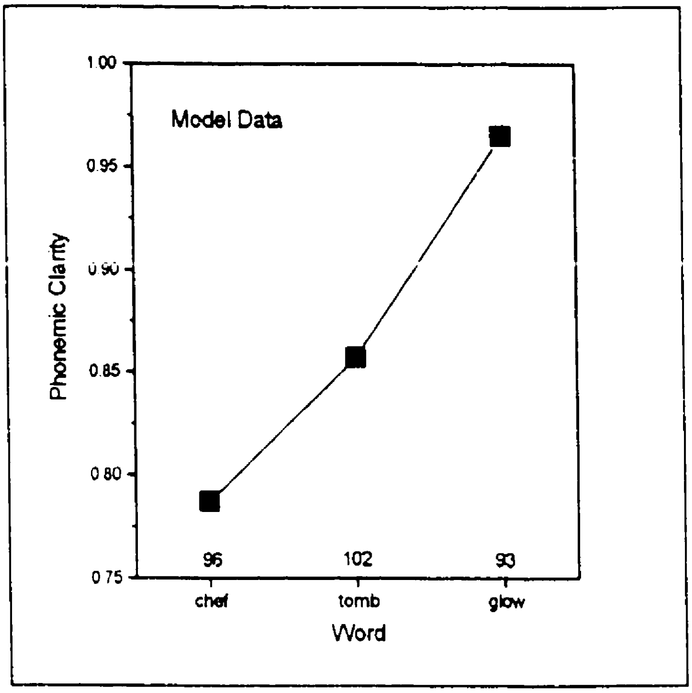

Figure 6.9. Average clarity of the blurriest phoneme for each word (the average number of samples from the lexicon required to retrieve the letters are above each word)

A Simulation

Considering the criticisms that Plaut et al. (1996) raise regarding Coltheart and Rastle's (1994) items, I was hesitant to use the same words in a simulation. Basically, if the position of irregularity is confounded with the number of words that share a similar spelling, simulating Coltheart and Rastle's results using their words does nothing to provide evidence for a sequential operation during reading.

Instead, LEX read the same words as the DRC and Plaut et al.'s (1996) models read in their simulations. LEX read the words *chef, tomb, and glow* 16 times. The word *chef* is irregular at the first phoneme, *tomb* at the second, and *glow* at the third. LEX made four pronunciation errors by regularising the pronunciation of *chef*. As is clear in Figure 6.8, LEX shows the same pattern of naming times as the DRC and Plaut et al.'s connectionist account; as the position of the irregular phoneme nears the end of a word, there is a decrease in the degree to which the irregularity affects naming latency.

Why does it work?

Recall that, after LEX retrieves a letter, it reduces the cohort of candidate words to include only those words that contain matches to the retrieved letters. Hence, at each cohort, LEX is more likely to sample a word from the lexicon that contains the target pronunciation for the irregular phoneme. In addition, because LEX only samples words from a cohort of candidate matches to the target, phonemes at the beginning of the retrieved word are reinforced on each sample.

Consider the case wherein LEX reads a word like *glow* whose irregularity is positioned at the final phoneme. By the time LEX reaches the letters roughly corresponding to the irregular phoneme, LEX has almost narrowed in on the pronunciation of the word. That is, there are few words left in the cohort after retrieving *glo* from the lexicon. Because there is little competition from other words in the lexicon containing the same letters and divergent pronunciations, there is little effect of irregularity on words whose regularity is positioned at the end of the word.

Now consider what happens when LEX reads a word such as *chef.* Until LEX reaches the final letter, the dominant pronunciation for the first phoneme comes from words such as chief, chore, choir, and chord. It is the retrieval of the final letter that changes the first phoneme to the */sh/* sound. Even though the first phoneme changes to the */sh/* sound, it does so against a background of evidence for the alternative pronunciations for *ch*.

To illustrate the point, Figure 6.9 shows the clarity of the irregular phoneme of each word LEX read (for each word, the irregular phoneme was the blurriest one). The number of cycles required to retrieve the orthography from memory is listed above each word in the figure. As is clear in the figure, the time it took to retrieve the word from memory was fairly constant across the words. Consistent with decreased phonological competition as a consequence for cohort reduction, there is monotonic increase in clarity as the irregular phoneme approaches the end of the word.

LEX's account shares some similarity to both Plaut et al.'s (1996) explanation and the DRC account for the phenomenon. Consistent with Plaut et al., LEX is sensitive to the position of the irregular phoneme because fewer words contain competing phonemes as the phoneme's position nears the end of a word. Plaut et al. postulated that the reduced competition is a property of the way English words are constructed. By contrast, LEX reduces the number of words with competing phonemes through cohort reduction.

The DRC account treats the effect as a reflection of the relative time-course for the lexical and nonlexical routes. If the irregular phonemes is positioned near the end of a word, the lexical route will likely have had enough time to look up the correct pronunciation before the nonlexical route reaches the critical graphemes. In a similar fashion, LEX is more likely to have already narrowed-in on the target word when the irregular phoneme is positioned near the end than when its position is at the beginning. Sequential processing is responsible for the effect in both LEX and the DRC. The obvious point of departure from Coltheart and Rastle's (1994) DRC account and LEX's is the number of routes required to produce it. In LEX, the effect occurs because of how information from the lexicon is retrieved.

Neighbourhood Density Effects in Word Identification Tasks

A word's neighbours (or neighbourhood) are the words that can be created by changing one letter in any letter position (Landauer & Streeter, 1973). For example, the neighbours of *cart* are *dart*, *part*, *mart*, *tart*, *curt*, *cast*, *cars*, *card*, *care*, and *carp*. Word identification is generally faster in the lexical decision and naming tasks when a word has many neighbours (a high-N word, or a word with a high neighbourhood density) than when it has few neighbours (a low-N word, or low neighbourhood density). There are two potential loci for a latency advantage for high-N words.

To some extent, the advantage for high-N words in the lexical decision task is intuitive—a word with many neighbours resembles many words and is therefore more *word-like* than a word with few neighbours. The notion that word-likeness is a critical factor affecting identification is corroborated by the finding that nonwords with many neighbours, i.e., word-like nonwords, are more difficult to classify as nonwords in the lexical decision task than nonwords with few or no neighbours (Coltheart, Davelaar, Besner, & Johnason, 1977; Forster & Shen, 1996).

A latency advantage for high-N words in the naming task is less intuitive. If a word is similar to many words that the reader knows, word identification should be more difficult. Specifically, the reader should be more inclined to misidentify the target word as one of its neighbours when the word has many neighbours than when it has few. If, however, a large neighbourhood size speeds up lexical access, there should be a both the naming and lexical decision tasks because the lexical access stage is common to both.

Andrews (1989, 1992) examined the effect of neighbourhood density on word identification to determine the stage at which neighbourhood size exerts its influence. She reasoned that, if a decision stage after lexical access is responsible for neighbourhood density effects in the lexical decision, making the decision component easy should attenuate or eliminate the effect. To test the notion, Andrews (1989, Experiment 2) changed the nonword foils in the lexical decision task to illegal letter strings, e.g., *tfrk*. Illegal letter strings should make the decision stage of the lexical decision task easy because non-wordlike foils would be easily distinguishable from words. She found that, while illegal foils decreased lexical decision latency and increased accuracy, the latency advantage for high-N words over low-N words persisted (in fact, it increased slightly). A pattern suggesting that neighbourhood density effects reflect differences in lexical access times for high- and low-N words.

Andrews (1989; see also Sears, Hino, & Lupker, 1995, Experiment 3a for a replication) also noted that neighbourhood density affects performance only for low-frequency words. Her interpretation for the difference between high- and low-frequency words was couched in term of the operations of the IAM. Because the IAM was adopted as the lexical route in the DRC, I will defer discussion of the explanation to the next section.

The DRC's Explanation

Coltheart and Rastle (1994) simulated the effect of neighbourhood density in the lexical decision task using the DRC. Unfortunately, they did not include simulations that examined the neighbourhood density's effect on performance in the naming task. Hence, we will limit our discussion of the DRC account of neighbourhood density effects to data collected using the lexical decision task.

Andrews (1989; see also Coltheart & Rastle, 1995; Coltheart, et al., 1993) placed responsibility for the effect of neighbourhood density in the lexical decision task and its interaction with a word's frequency on the feedback between the word and letter nodes of the IAM. When the IAM is presented with a low frequency word, activation from the letter nodes is passed up to the word nodes. Word nodes that are consistent with the activated letter nodes are excited. Low-frequency word nodes have a low resting activation level, hence, lexical access is unlikely to occur the first time the word nodes are activated (Remember, lexical access occurs over several processing cycles). Because the IAM postulates mutual inhibition between word nodes, one would expect that many neighbours would inhibit identification of a target word. However, Andrew's postulated that, when the model is presented with a low frequency word, activated word nodes increase the activation of consistent letter nodes. Because the model is interactive, letter node activation is sent back to the word nodes strengthening the activation of the target word's node. The more word nodes feeding back to the letter nodes, the more activation is passed back to the word nodes from the word nodes causing the facilitatory effect of neighbourhood density on word identification.

High-frequency word nodes have high resting activation levels relative to low-frequency words. When the IAM is presented with a high-frequency word, lexical access occurs with little aid from reciprocal feedback between the letter and word nodes. That is, the target word's node reaches its threshold before it can gain much benefit from the interactive activation between the letter and word nodes. In sum, neighbourhood density effects reflect the extent to which similar words in the lexicon can increase the target word's activation.

For the same reason that words with many neighbours are easier to detect than ones with few neighbours, nonwords with many neighbours are harder to detect than nonwords with few neighbours. A nonword that is similar to many words is very "word-like" relative to a nonword that shares little similarity with words the reader knows. A nonword's letters activate words that are consistent with the letters of the target word. As word entries are activated over several cycles, they feed their activation back to the letters, which in turn, further activate the consistent words. Hence, the more words in the lexicon that share letters with the nonword, the more the nonword appears like a word for the DRC, and the harder it is to label a nonword.

The Connectionist's Explanation

The recent model by Plaut et al. (1996) does not include a discussion of neighbourhood effect in word identification; hence, I will forego speculating how their model might simulate them. Seidenberg and McClelland (1989) gave neighbourhood effects some treatment in their model so I will focus instead on their account instead.

Seidenberg and McClelland (1989) limited their discussion of neighbourhood density effects to a simulation of the advantage for high-N words over low-N words in the naming task. Recall that, during training, Seidenberg and McClelland's model was presented with components of words in the form of wickelfeatures. Words with many neighbours necessarily share wickelfeatures. For example, the wickelgraphs, ART, and RT_ are contained in many of the neighbours of *cart* (*dart*, *part*, *mart*, *tart*). Connectionist models are able simulate neighbourhood effects because of the frequency with which letter and phoneme combinations are paired during training. The components of words with many neighbours are more frequently presented to the model during training than the components of words with few neighbours. Because the components of high-N words are more highly learned than those of low-N words, words with a large neighbourhood density are identified more quickly than words with a small neighbourhood density.

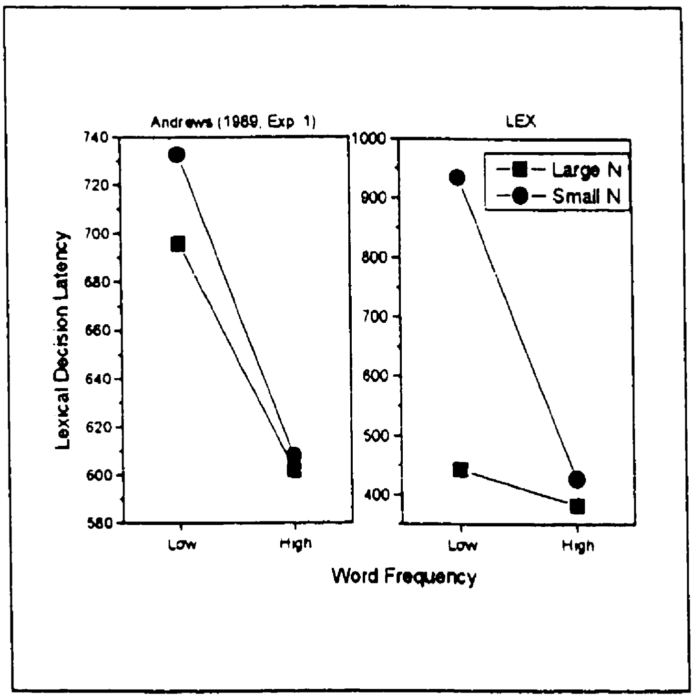

Figure 6.10 Subject and simulation data for Andrew's (1989) Experiment 1

Simulations

The DRC and connectionist explanations for neighbourhood density effects in word identification are incomplete because neither camp simulated both the lexical decision and naming tasks together. In the simulations to follow, LEX named and made lexical decisions to high- and low-frequency words that had either large or small neighbourhoods. In the first three simulations, I used Andrews' (1989) words. The words comprising the factorial combination of high or low word frequency and large or small neighbourhood density. In the last simulation, LEX made lexical decisions on both words and nonwords that varied in neighbourhood density. The items for the final simulation were taken from Coltheart, Davelaar, Jonnasson and Besner (1977). LEX's error rates for Andrews' materials are shown in Table 6.2. LEX's Error rates for the materials used by Coltheart et al. are shown in Table 6.3

Table 6.2

Proportion of Error Trial in LEX's Simulation of Andrews' (1989) Experiments

		Word Frequency	
Experiment		Low	High
Exp 1			
	Large N	0	0
	Small N	0.05	0
Exp 2			
	Large N	0	0
	Small N	0	0
Exp 3			
	Large N	0	0
	Small N	0.04	0

Simulations 1 and 2 To simulate the increased ease with which a lexical decision can be made when illegal nonwords are used as foils, I adjusted LEX's word criterion parameter. Recall that, after retrieval, LEX uses the similarity (as measured by Pearson's correlation coefficient) between the probe letters and the retrieved orthography to derive a signal value for a two-barrier random walk decision mechanism. The signal is calculated as the difference between the correlation and the word criterion. The word criterion denotes the minimum correlation required for LEX to consider a letter string a word. For example, with the word criterion set to 0.85, a correlation of 0.80 would yield a signal value of -0.05, corresponding to evidence that the retrieved orthography is not a word. A correlation of 0.90 is yields a positive signal (0.05) and is taken as evidence that the retrieved orthography has lexical status.

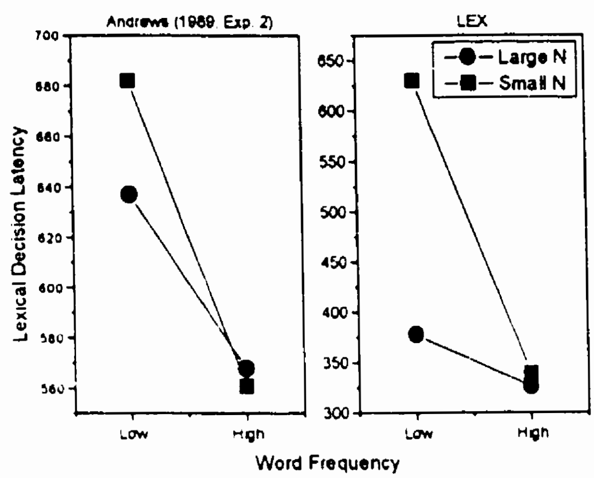

Figure 6.11 Subject and simulation data for Andrew's (1989) Experiment 2

I assume that LEX, and readers, relax the word criterion when all the foils are illegal nonwords. Hence, to simulate Andrews' (1989, Experiment 1) experiment wherein legal nonwords were used as foils, the word criterion was set to 0.90. I dropped the criterion to 0.88 in the second simulation to reflect the increased ease with which the system can make the decision when the foils are illegal letter strings (Andrews, Experiment 2).

The left panels of Figures 6.10, 6.11, and 6.12 summarise the data reported by Andrews (1989). Figures 6.8 and 6.9 shows summary data from the lexical decision task when legal and illegal nonwords are used as foils, respectively. Figure 6.12 contains Andrews' data from the naming task. The right panels of Figures 6.10, 6.11 and 6.12

summarise data from LEX's performance in the same tasks. As is clear in the figures, LEX replicates the patterns reported by Andrews (1989) for both the naming a lexical decision tasks. Low-frequency words exhibit a strong advantage for words with high-density neighbourhoods over low-density neighbourhoods. There is no effect of neighbourhood density for high-frequency words.

Simulation 3 As mentioned above, nonwords exhibit a pattern of performance that is the opposite to words. Nonwords with many neighbours take longer to classify as nonwords than words with few neighbours (Andrews, 1989; Coltheart, Davelaar, Besner, & Johnasson, 1977; Forster & Shen, 1996; Sears, et al, 1995). Put simply, nonwords that are similar to many words are more "word-like" than nonwords that are similar to few words. Coltheart et al. (1977) included the word and nonword stimuli in their article introducing the effect, so LEX made lexical decisions on their items.

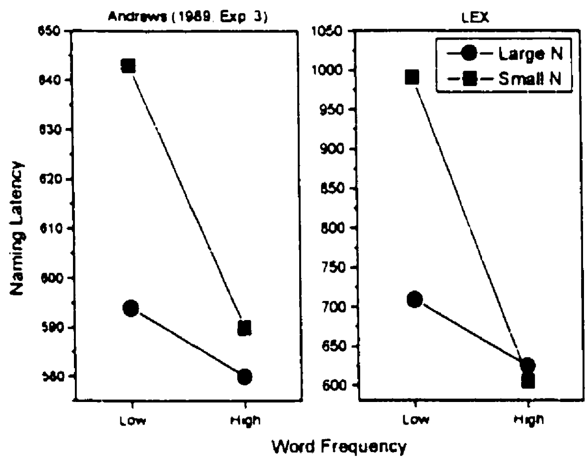

Figure 6.12 Subject and simulation data for Andrew's (1989) Experiment 3

Table 6.3

LEX's error rates in the lexical decision for the Coltheart et al.'s (1977) materials

Materials		Neighbourhood Size	
		Small	Large
	Words	0.12	0.06
	Nonwords	0.25	0.17

There is one peculiarity about the data reported by Coltheart et al. (1977). They reported an inhibitory effect of neighbourhood density on lexical decision latencies for nonwords, and a null effect of neighbourhood density on word identification latencies. The left panel of Figure 6.13 mean the latency data reported by Coltheart et al. (1977). The right panel contains mean latencies from LEX. As is clear in the figure, LEX reproduces the inhibitory effect of neighbourhood density on decision times for nonwords, and the null effect of neighbourhood density on decision times for words.

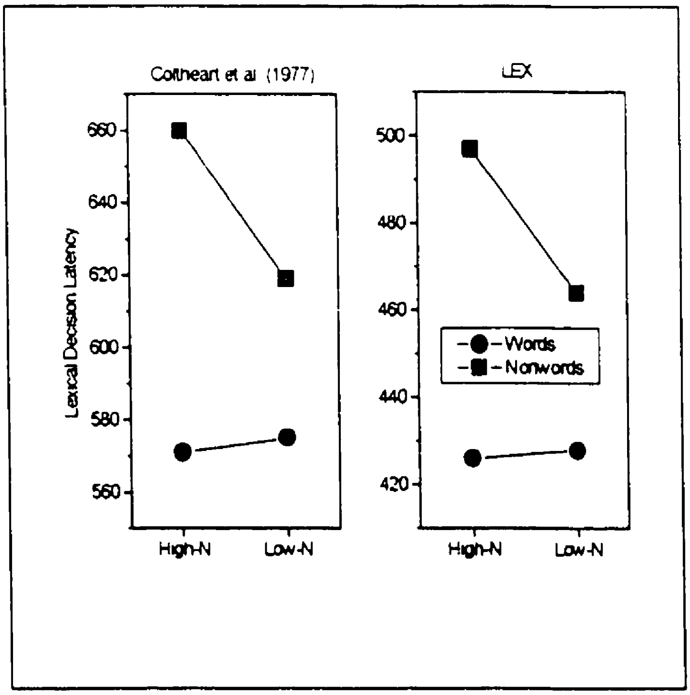

Figure 6.13 Subject and simulation data for Coltheart et al. (1977), Experiment 2

Why does it work?

Consistent with Andrews (1989) and Balota and Chumbley (1984; 1985), LEX treats lexical decision as a two-stage processes. During the first stage, information is retrieved from lexical memory. Once information has been retrieved, the lexical decision is made by a decision mechanism that decides whether the retrieved orthography is a word. LEX also treats the naming task as a two-stage process—the task is performed by a set of mechanisms that are sensitive to the clarity of the phonemes that have fallen out of lexical memory during lexical access.

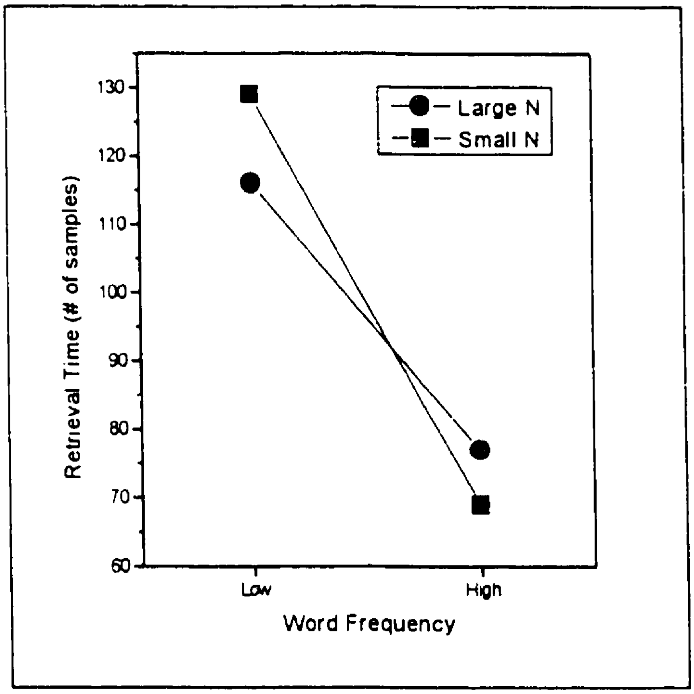

Figure 6.14 Mean number of samples required to retrieve the orthography of the words used by Andrews (1989)

Andrews (1989), Experiments 1 & 2 : Lexical Decision. When Andrews (1989) used illegal letter strings as nonword foils in the lexical decision task (Experiment 2), the effect of neighbourhood density did not change. Andrews (1989) argued, therefore, that neighbourhood effects occur during lexical access. Presumably, if the effect occurred at the decision stage, the neighbourhood effect should disappear because subjects would base the decision on the legality of the letter string. Placing the locus of neighbourhood

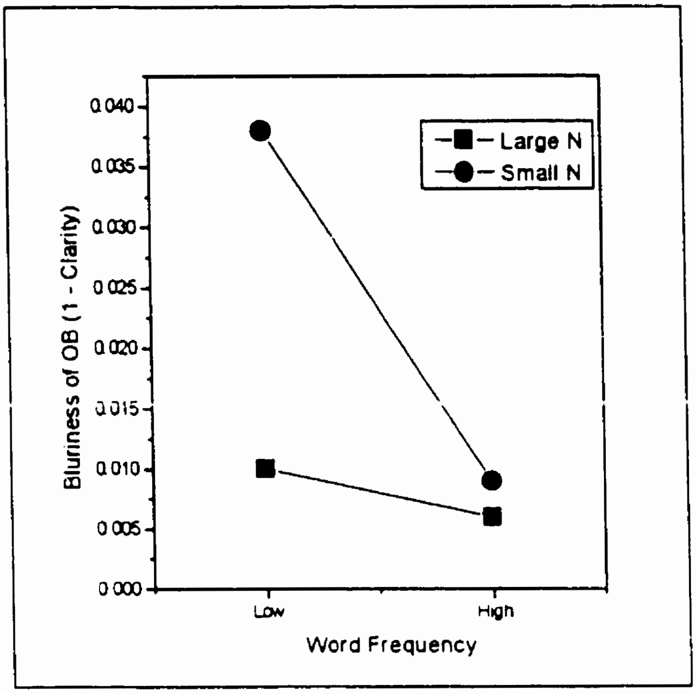

Figure 6.15 Average blurriness of the orthography in the OB for Andrew's (1989)

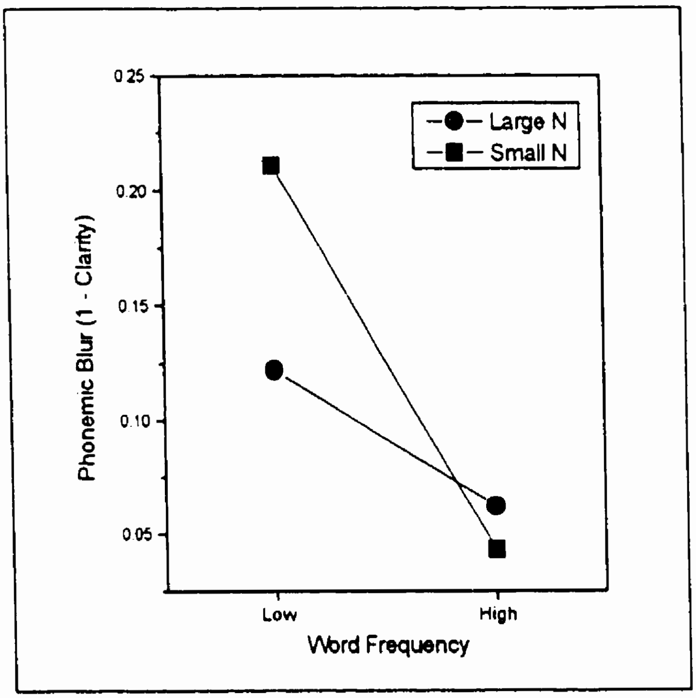

Figure 6.16 Average blurriness of the blurriest phoneme in the PB for Andrews' (1989) words

effects during lexical access is also consistent with the similar pattern of performance in the naming task—both tasks involve the lexical access stage. Forster and Shen (1996) countered Andrews' argument by demonstrating an absence of neighbourhood density words effects in a semantic decision task. Presumably, a semantic decision also requires lexical access. Hence, if Andrews' analysis is correct, neighbourhood density should affect semantic decisions as well. Forster and Shen argued therefore, that neighbourhood density affects performance at the decision stage of processing.

LEX is sensitive to neighbourhood density for the reasons offered by Forster and Shen; that is, the effect occurs during the decision stage. Recall that sometimes LEX fails to retrieve the correct letter from the lexicon (especially when the word is low-frequency). When the failure occurs, LEX readjusts its search space and continues retrieving the remaining letters. If the target word has many neighbours, there are more words with letter information that is consistent with the probe in the cohort than when the target has few neighbours. As shown in Figure 6.14, however, neighbourhood density, has little impact on the number of samples required to retrieve the lexical information from memory. Rather, because low-N words share fewer letters in the same position with other words in the lexicon, the clarity of the representation the OB is compromised. As is clear in figure 6.15, there is a clear difference between the average OB's clarity for the large- and small-N words when the words are low-frequency, but little corresponding difference for high-frequency words. Because high-frequency words are generally retrieved without failure, the clarity of the orthography in the OB is not as affected by repeated retrieval of irrelevant words. Consequently, high-frequency words do not exhibit sensitivity to neighbourhood density.

It is easy to understand why Andrews (1989) placed responsibility for neighbourhood density effect at the lexical access stage. By using illegal nonword foils, she correctly assumed that the decision stage was made easier. However, making the decision stage easier does not eliminate decision—the task requires it. I made the decision stage easier by decreasing LEX's word criterion. The decrease reflects a reader's strategic adjustment of how word-like a letter string must be to be considered a word. If the nonwords look much like words, the reader must be cautious not to make an error, and the criterion must be high. If, on the other hand, all the nonword foils are consonant strings, the reader can afford to be more lax. Decreasing the word criterion to a magnitude that still allows high accuracy increases the signal value to the decision mechanism yielding two results: faster response latencies and decreased errors. Because the decrease does not eliminate the contribution of the decision stage to the response, the latency advantage for high-N words persists when the word criterion is lowered.

Andrews (1989) Experiment 3: The Naming Task LEX's sensitivity to neighbourhood density in the naming task also reflects the clarity of the information that is retrieved from lexical memory. Specifically, the clarity of the phonemes for low-N words is lower than that of high-N words causing longer naming latencies for low-N words (see Figure 6.16). The explanation for the latency advantage for high-N words over low-N words in the naming task is similar to LEX's explanation for the lexical decision task.

As mentioned earlier, following retrieval, the phonemes of high-frequency words tend to have almost perfect clarity for two reasons: (1) letter retrieval rarely fails and (2) the phonemes of words are weighted by their frequency during retrieval. The weighting causing the phonemes of high-frequency words to have a large impact on the phonemes in the PB. Because the phonemes are almost at ceiling, there is little effect of neighbourhood density on naming latency when the words are high-frequency.

When the words are low-frequency, however, LEX is more likely to experience failures during retrieval. Because LEX must adjust the cohort to aid identifying the missed letter, phonemes from several irrelevant words are included in the new cohort, and of course, each sample of words from lexical memory. If the word has many neighbours, many of the neighbours will also be included in the new cohort. A word's neighbours tend to have many phonemes in common as well as letters; hence, as is shown in Figure 6.16, the increased clarity of the features in the OB is paralleled by an increase in the clarity of the phonemes in the PB.

Coltheart et al. (1977). Coltheart et al. demonstrated a latency advantage for low-N nonwords over high-N nonwords in the lexical decision task. They found no effect of neighbourhood density for lexical decisions made to words. Andrews (1989) pointed out that the words they used were generally high-frequency. Because the latency advantage for high-N words is limited to low-frequency words, it is not surprising that they failed to find an effect. For the same reason, LEX also failed to find an effect of neighbourhood density using their words.

Because LEX is retrieving the letters of a nonword, retrieval failure is inevitable—at some point, LEX will reach the letter that defines the letter string as a nonword. When the cohort is adjusted, many of a nonword's neighbours will be included in the cohort of potential matches. As with words, if the nonword has many neighbours the clarity of the letters in the OB will be higher than if the nonword has few neighbours.

The higher the clarity of the OB, the more wordlike the features of the OB are considered to be. Hence, as the clarity increases, the signal to the random walk decreases. In this simulation, the word criterion was set to 0.89. The average clarity of the OB for low-N and high-N nonwords was .76 and .78 respectively. Using the clarities to derive signals for the random walk (clarity - word criterion), low-N nonwords had a mean signal of -0.13, and high-N nonwords had a signal of -0.11.

In sum, the lexical status of a high-N nonwords takes longer to decide than the status of a low-N nonword because the former nonword is more wordlike than the latter.

Basic Orthographic Syllable Structure (BOSS) Effects in Lexical Decision

Taft (1979; 1986; Taft & Forster, 1976) theorised that reading a word requires matching a word's first orthographically and morphologically defined syllable with a sensory match in the lexicon. Taft referred to the first syllable as the BOSS, an acronym for Basic Orthographic Syllable Structure. To read the word *blemish*, for example, the BOSS, *blem* is matched to its sensory representation in the lexicon, and subsequently, it is paired with *ish* to form *blemish*. To demonstrate the importance of the BOSS as an anchor for accessing a word, Taft and Forster (1976; see also Taft, 1986) examined lexical decision performance for three types of nonwords: nonwords that were BOSSes of words, for example, *blem* of the word, *blemish*, nonwords that formed the beginning of words, but were not BOSSes, for example, *roun* of the word *round*, and nonwords that did not form the beginning of any word, for example, *vuth*.

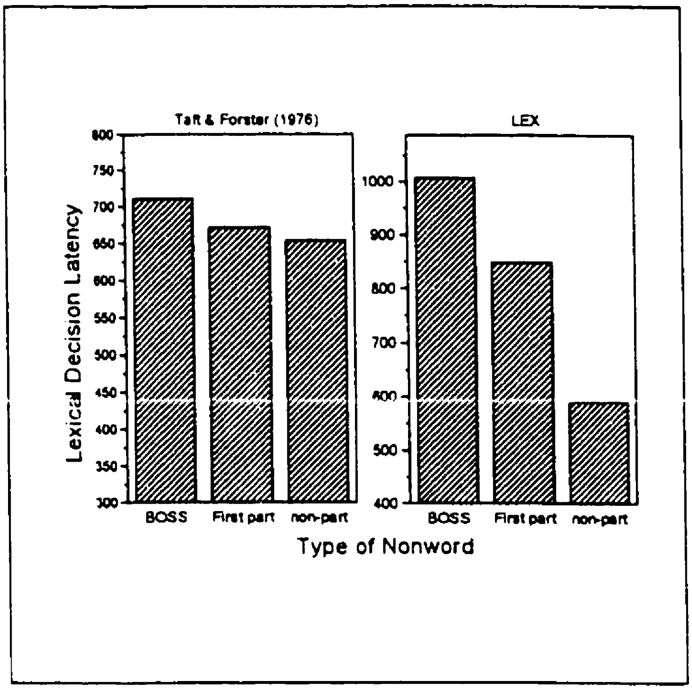

Figure 6.17 Subject and simulation data for lexical decision latencies for Taft and Forster's (1976) nonwords

Taft and Forster (1976; and also Taft, 1986) showed that BOSS nonwords took longer to reject in the lexical decision task than either nonwords that formed the beginning of a word or nonwords that did not form the beginning of a word. They interpreted the pattern of decision latencies as reflecting the special status of the BOSS's in the lexicon. Their analysis makes sense; if a nonword is a BOSS of a word it is represented in the lexicon. If it is represented in the lexicon, it is very wordlike and should be difficult to reject in a lexical decision task. The left panel of Figure 6.17 contains the mean lexical decision latencies reported by Taft and Forster (1976).

A simulation

The right panel of Figure 6.17 shows LEX's lexical decision latencies for Taft and Forster's nonwords. LEX replicates the basic pattern of data reported by Taft and Forster; BOSS nonwords take longer to classify than non-BOSS nonwords. LEX's error rates for BOSS nonwords, non-BOSS nonwords, and non-part nonwords were 0.26, 0.19, and 0.10 respectively.

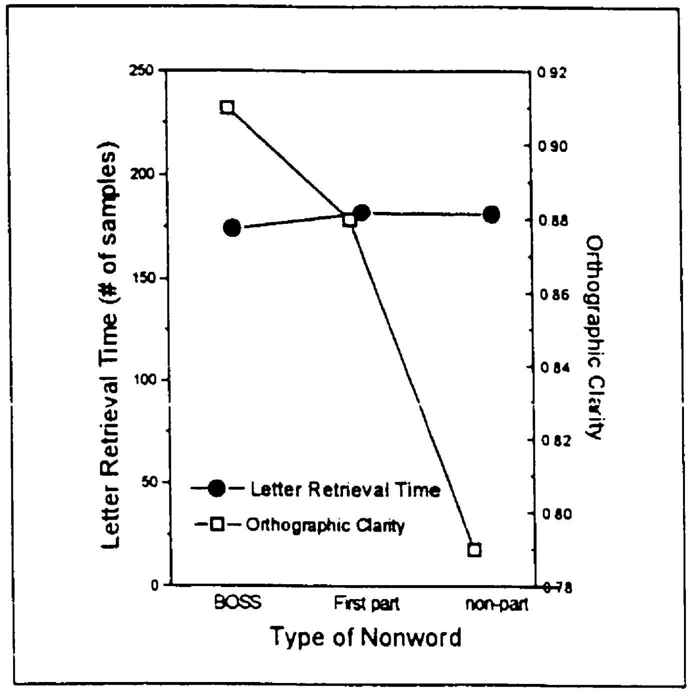

Figure 6.18 Letter retrieval time and orthographic clarity of the OB for each class of nonword using Taft and Forster's (1976) nonwords

Why does it work?

In contrast to Taft's (1979; 1986; Taft & Forster, 1976) theory of lexical access, LEX does not explicitly represent *any* orthographic structures within words. LEX's sensitivity to the BOSS reflects the scale of its lexicon. When a lexicon is made to scale, what looks like sensitivity to structure may simply reflect sensitivity to redundancy in the language. To illustrate the point, I tabulated how many words in LEX's lexicon contained Taft and Forster's nonwords as beginnings. Across nonwords in each category, there are, on average, 13 words in LEX's lexicon that start with Taft and Forster's BOSSes, 7 words that start with their non-BOSS nonwords, and 3 words that start with their non-part nonwords (some of Taft and Forster's non-part nonwords were misclassified. For example, *pren* forms the beginning of the word *prenatal*). Clearly, BOSS nonwords are more wordlike that non-BOSS nonwords, which are in turn more wordlike than non-part nonwords.

Figure 6.18 plots the average clarity of the OB for each type of nonword. As the Figure shows, as a nonword becomes less word-like, the degree to which the retrieved letters match the probe letters decreases. The less a letter string resembles a word, the faster, and more accurately, LEX can correctly respond to it. In sum, LEX's performance is sensitive to redundancy in the language that is correlated to the BOSS, not the BOSS itself.

Reading Multi-Syllabic Words

Almost every simulation model of word identification limits its lexical knowledge to one-syllable words (for an exception, see Anns, et al., 1999). Why are multi-syllabic words so often excluded? There are two possible reasons. Perhaps, theorists are unsure how to implement a parsing mechanism that can divide a word into its constituent syllables. Indeed, syllables appear to be important structures for lexical access. Some models of letter encoding (e.g., Mewhort and Campbell, 1981; Spoehr and Smith, 1973) have even postulated that the syllable is the functional unit for lexical access. The other reason may be that the representation scheme used by many models for encoding words would have to change drastically to accommodate polysyllabic words. For example, a back-propogation network model that uses Wickelfeatures to represent orthography and phonology at its input and output layers could not uniquely represent words like *banana*, *mississippi*, and *chihuahua*; their representations as wickelfeatures are indistinguishable from *bana*, *missippi*, and *chihua*. In a like fashion, the vowel, onset, and coda component scheme chosen Plaut et al. (1996) would require change to accommodate several more components of words. As the number of components grows, eventually Plaut et al's model would be forced into using the letter channel representation scheme used by the DRC and IAM. To its credit, because the DRC uses letter channels to represent the spatial arrangement of letters in a display, it can represent the letters of long multi-syllabic words. Coltheart and Rastle (1994) reported that the DRC can represent words up to nine letters in length. However, because the DRC uses the IAM's letter channel representation, it inherits the earlier mentioned problems associated with an inconsistent representation of space between letters.

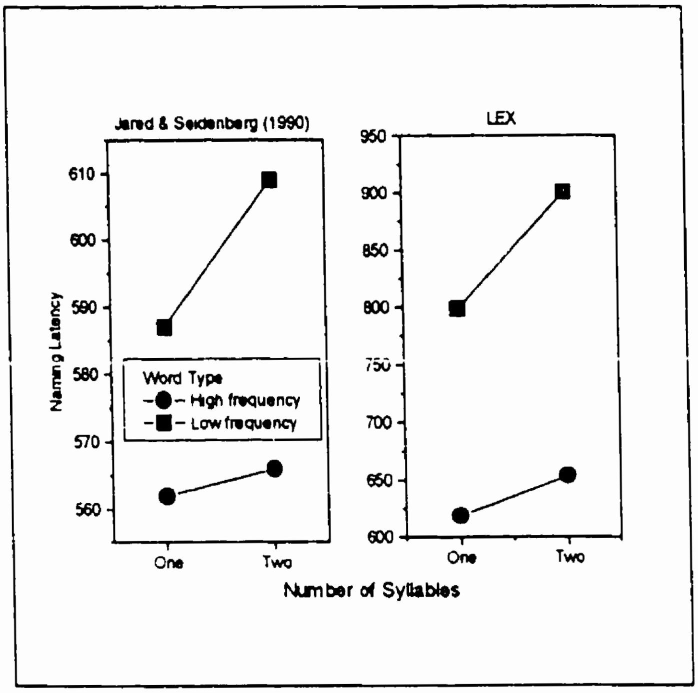

Figure 6.19 Subject and simulation data for Jared and Seidenberg's (1990) six-letter words in the naming task

Jared and Seidenberg (1990, Experiment 3) demonstrated readers' sensitivity to syllabic structure in word naming; multi-syllabic words take longer to pronounce than monosyllabic words. They also noted that the effect is more pronounced when the words are low-frequency than high-frequency. Clearly, the attenuation of a naming advantage for one-syllable high-frequency words over two-syllable high-frequency words is problematic for Spoehr and Smith's (1973) and Mewhort and Campbell's (1981) hypothesis that reading involves an obligatory parsing stage prior to lexical access.

Jared and Seidenberg (1990) suggested that readers' sensitivity to syllabic structure could reflect a correlation between orthographic and phonological information contained in the words a reader knows. In essence, they proposed that readers are not sensitive to syllables *per se*; they are sensitive to lexical knowledge that is correlated to syllables. They further reasoned that a model that stores little more than information correlating orthography and phonology (for example, Seidenberg and McClelland's, 1989, connectionist account) should be able to reproduce the pattern of naming data.

In the simulation that follows, we offer a confirmation of Jared and Seidenberg's (1990) speculation that a "model lacking an explicit level of syllabic representation of syllabification rules" (p. 103) can demonstrate sensitivity to the number of syllables in a naming task.

A Simulation

LEX named the 60 six-letter words used by Jared and Seidenberg (1990). Half of the words were monosyllabic, and the other half were bisyllabic. Half of the words were low-frequency and half were high-frequency words.

The mean latency data for the six-letter words reported by Jared and Seidenberg (1990) are shown in the left panel of Figure 6.19. LEX made no pronunciation errors. There is a clear naming latency advantage for monosyllabic words over bisyllabic words when the words are low-frequency, but not when the words are high-frequency. LEX's mean naming latencies for the same stimuli are shown in the right panel of Figure 6.19; a clear replication of the basic interaction reported by Jared and Seidenberg.

Why does it work?

The clarity of the information in LEX's phonological buffer dictates how long it will take to begin pronunciation after retrieval. Pronunciation begins when an articulatory code for the blurriest phoneme has been prepared. As shown in Figure 6.20 the average phonemic blurriness of bisyllabic words tends to be greater than that of monosyllabic words, but only when the words are low-frequency.

The inconsistency with which vowels are often pronounced in English is responsible for LEX's ability to reproduce the phenomenon. When LEX reads a word, the vowels are often the blurriest of the retrieved phonemes; hence, the clarity of the vowels often determines naming latency. Figure 6.21 plots the average clarity of each phoneme for low-frequency, one and two-syllable words. The phonemes of two-syllable words are more blurry than those of one-syllable words. The noise associated with two-syllable words occurs because a two-syllable word generally has a vowel near the end of the word; a one-syllable word's vowel tends to be near the beginning of the word.

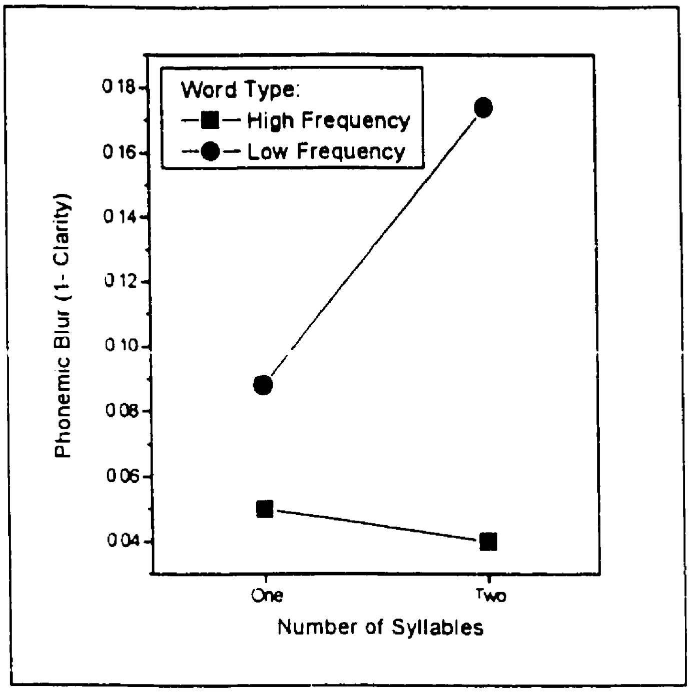

Figure 6.20 Average phonemic blur for each class of word used by Jared and Seidenberg (1990)

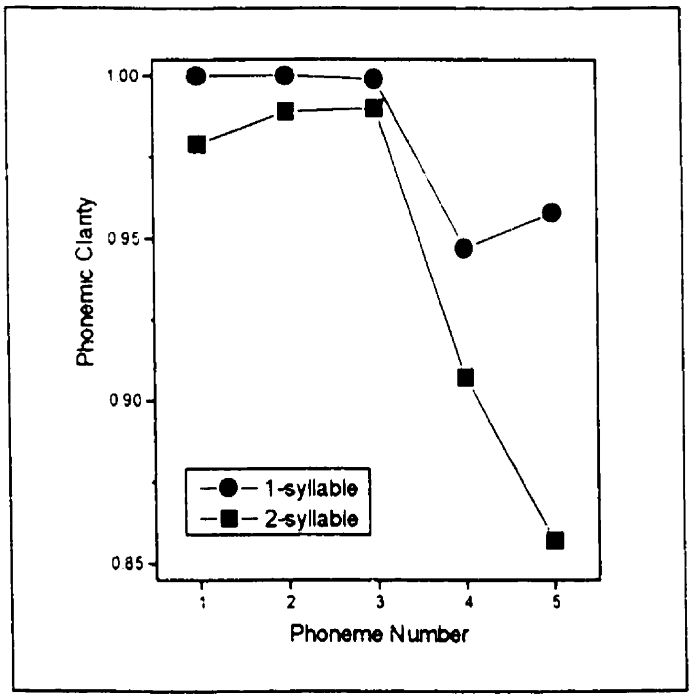

Figure 6.21 Average clarity of each phoneme in one- and two-syllable low-frequency words

As LEX builds the information in the orthographic and phonological buffers by sampling lexical information, words that are inconsistent with the probe are eliminated through cohort reduction. Only lexical entries that are consistent with the probe so far are available for sampling. Consequently, the letters and phonemes that have been retrieved are reinforced. Because initial letters and phonemes are reinforced more often than the final ones, a vowel positioned near the end of a word tends to be more blurry than one near the beginning.

The Orthographic Uniqueness Point Effect

The uniqueness point (UP) is a characteristic of a word made popular by Marslen-Wilson's (1984) tests of his cohort model of auditory word identification. A word's UP is the phoneme, in the stream of phonemes that make up a word, that differentiates a word from all other words that the listener knows. When subjects are asked to decide on the lexical status of a word that is presented auditorially, it is recognised as a word more quickly when it has an early UP than when the UP comes late in the stream of phonemes.

Radeau, Morais, Mousty, Searens, and Bertelson (1992) reasoned that, if reading a word required left-to-right processing through the letters of a word, readers should be sensitive to the uniqueness point (UP) of the printed word as well. Radeau et al. had subjects read words known to yield the UP in the auditory lexical decision task (Radeau, Mousty, & Serens, 1989). They found no evidence that a word's UP affected identification latency in either a naming task or a gender classification task (French nouns were used in their experiments). On the basis of their null result, they concluded that reading words was an example of parallel processing.

Kwantes and Mewhort (1999) suggested that Radeau et al.'s (1992) use of the UP did not provide a fair test of sequential processing in a reading task because the UP is a phonological characteristic of a word. Kwantes and Mewhort selected 100 seven-letter words and classified them on the basis of their orthographic uniqueness point (OUP). The OUP was defined as the letter position, reading from left to right, that differentiates a word from all other words the reader may know.

In two experiments, Kwantes and Mewhort (1999) found that naming latency was shorter for words that had their OUPs after the third letter than words that had OUPs after the sixth or seventh letter position. A pattern pointing to a sequential operation at some point during reading. The naming advantage for early-OUP words disappeared when they were named in a delayed-naming task; suggesting that the mechanisms responsible for production are not solely responsible for the effect.

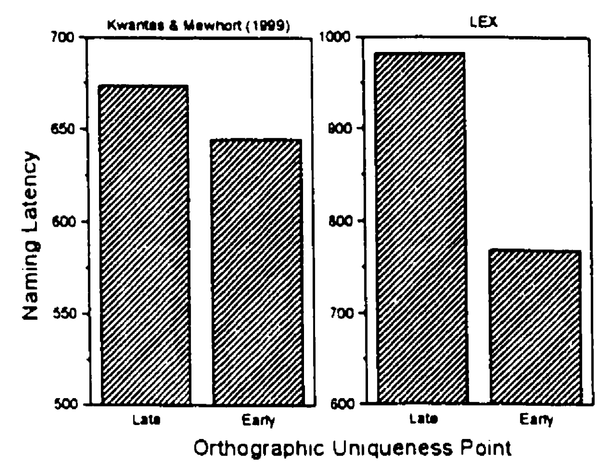

Figure 6.22 Subject and simulation data for the words used by Kwantes and Mewhort (1999)

A Simulation

For this simulation, LEX named the words used by Kwantes and Mewhort (1999). LEX erred on four trials when it read words with early-OUP words and 16 trials when it read late-OUP words. The left panel of Figure 6.22 shows mean naming latency for subjects. The right panel contains mean naming latencies from LEX. As is clear in Figure 6.22, LEX captures the naming advantage for early-OUP words.

Why does it work?

The latency advantage for words with an early OUP is a novel prediction from LEX. As LEX retrieves each letter from the lexicon, a cohort of candidate matches to the target is adjusted to include only the word that are consistent with the target so far. At some point during retrieval, there will be only few words in the cohort that are consistent with the target. In the case of an early-OUP word, such as *biplane*, the retrieval of the letter *l* at the fourth position differentiates *biplane* from all other words in the lexicon. In the case of a word such as *curtail*, a late-OUP word, LEX is uncertain about the identity if the word until it retrieves the final letter. Until LEX retrieves the *l*, the word could be *curtain*. The naming advantage for early-OUP words reflects the speed with which LEX can retrieve the remaining letters after the OUP of a word has been reached.

LEX required 158 and 272 samples from the lexicon to retrieve the letters of early- and late-OUP words, respectively. In addition to a greater number of samples required to retrieve the letters of late-OUP words, the phonemes of late-OUP words in the PB were also more blurry (.78) than the phonemes for early-OUP words (.84). Because of cohort reduction, by the time LEX reaches the letter corresponding to the OUP, only one word is available for sampling. With only one word available for sampling, the remaining letters are retrieved quickly from the lexicon. Cohort reduction is also responsible for the greater clarity associated with the phonemes of early-OUP words. When the OUP has been reached early in retrieval, the phonemes of only one word are being copied into the PB. When a word has an early OUP, its phonemes in the PB of the word are reinforced by many more sample than the phonemes retrieved for a word that is uniquely defined near the end of the letter string.

Reading Nonwords

Thus far, I have demonstrated that LEX and readers are sensitive to the same characteristics of words in two reading tasks. An important additional test of whether LEX's mechanisms are also present in readers is how closely its performance on novel stimuli parallels readers' performance. In order to match theory and data, three conditions must be met: first, LEX must generate plausible pronunciations for nonwords about as often as readers do (or at least as good as competing models). Second, LEX's performance must be sensitive to the same characteristics of nonwords that readers are. Third, there is often more than one way to pronounce a nonword which makes unclear what the correct pronunciation of a novel word should be. If, however, readers are biased to produce one pronunciation for a nonword over other pronunciations, the model should also be biased to make the same pronunciations. Finally, LEX should reflect any variability in readers' pronunciation of nonwords.

There are two ways in which a reader might read nonwords: by analogy to words she knows or by implementing an algorithm that translates spelling to sound. The two strategies for deriving a pronunciation are at the heart of the debate between proponents of connectionist and dual-route models of reading aloud. Recall that the dual-route model's nonlexical route uses grapheme-to-phoneme conversion rules to derive a pronunciation for nonwords. Connectionist models such as those by Seidenberg and McClelland (1989) and Plaut et al. (1996) read nonwords (and words, for that matter) by analogy.

Andrews and Scarratt (1998) examined how nonwords that could be read by analogy would be pronounced by readers. They divided the nonwords into four categories of body. The body refers to the letter cluster following the first letter or grapheme in a monosyllabic word. For example, *inch* is the body of the word *pinch*. First, Regular Consistent Body (RCB) nonwords contain word bodies whose pronunciation does not vary across words containing the body. For example, the body, *inch*, possesses a pronunciation that is invariant across words containing it. Second, their subjects read nonwords with Inconsistent Bodies (IB), that is, nonwords with bodies that had more than one common pronunciation. For example, *ead*, has two common pronunciations such as in the words *head* and *bead*. The final type of nonwords had bodies for which there was no regular analogy (NRA). That is, the body of the nonword is always pronounced irregularly when it appears in words. They used two such types of nonword: NRA-M nonwords had many neighbouring words sharing the same body and irregular pronunciation, for example, the *ight* of *yight* is found in several words: *fight*, *flight*, *sight*, *might*, *right*, *light*, *night*, and *tight*. In every instance of a word containing the *ight* letter combination, the *i* is a long vowel, and the *g* is unpronounced. NRA-U nonwords were nonwords for which the word body was irregular, but unique to only one word, for example, *sign* is the only word containing the body *ign*.

Because all of Andrews and Scarratt's (1998) nonwords could be read by analogy to words, they provided a fair test to determine which class of model, dual-route or connectionist, predicts pronunciations that are common to readers. Because connectionist should also be biased to make the same pronunciations. Finally, LEX should reflect any variability in readers' pronunciation of nonwords.

There are two ways in which a reader might read nonwords: by analogy to words she knows or by implementing an algorithm that translates spelling to sound. The two strategies for deriving a pronunciation are at the heart of the debate between proponents of connectionist and dual-route models of reading aloud. Recall that the dual-route model's nonlexical route uses grapheme-to-phoneme conversion rules to derive a pronunciation for nonwords. Connectionist models such as those by Seidenberg and McClelland (1989) and Plaut et al. (1996) read nonwords (and words, for that matter) by analogy.

Andrews and Scarratt (1998) examined how nonwords that could be read by analogy would be pronounced by readers. They divided the nonwords into four categories of body. The body refers to the letter cluster following the first letter or grapheme in a monosyllabic word. For example, *inch* is the body of the word *pinch*. First, Regular Consistent Body (RCB) nonwords contain word bodies whose pronunciation does not vary across words containing the body. For example, the body, *inch*, possesses a pronunciation that is invariant across words containing it. Second, their subjects read nonwords with Inconsistent Bodies (IB), that is, nonwords with bodies that had more than one common pronunciation. For example, *ead*, has two common pronunciations such as in the words *head* and *bead*. The final type of nonwords had bodies for which there was no regular analogy (NRA). That is, the body of the nonword is always pronounced irregularly when it appears in words. They used two such types of nonword: NRA-M nonwords had many neighbouring words sharing the same body and irregular pronunciation, for example, the *ight* of *yight* is found in several words: *fight*, *flight*, *sight*, *might*, *right*, *light*, *night*, and *tight*. In every instance of a word containing the *ight* letter combination, the *i* is a long vowel, and the *g* is unpronounced. NRA-U nonwords were nonwords for which the word body was irregular, but unique to only one word, for example, *sign* is the only word containing the body *ign*.

Because all of Andrews and Scarratt's (1998) nonwords could be read by analogy to words, they provided a fair test to determine which class of model, dual-route or connectionist, predicts pronunciations that are common to readers. Because connectionist models read nonwords as analogies to words that they know, they are biased to yield pronunciations corresponding to the most frequent pronunciation of the word bodies stored in the model. For example, the body of the nonword *linth* would be pronounced to rhyme with ninth by connectionist models because it is the only pronunciation associated with the body, *inth*. On the other hand, the DRC translates letter strings into phonemes starting from the leftmost letter to the ending letter. Because the letter *i* is not followed by an *e* at the end of the nonword, the DRC's appropriate GPC rule will give it a short vowel pronunciation.

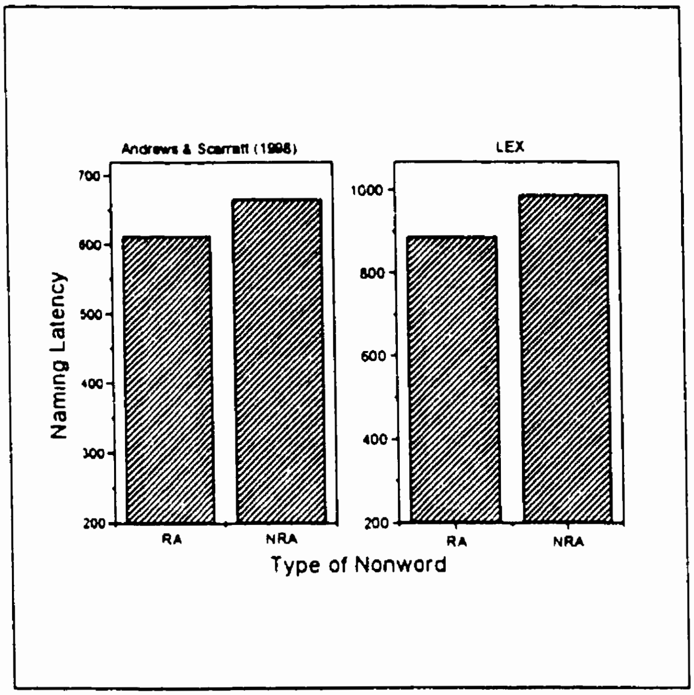

Figure 6.23 Subject and simulation data for Andrews and Scarratt's (1998) nonwords

The DRC predicts that nonwords will be generally be regularised regardless of the frequency of the pronunciations of the nonwords' bodies. On the other hand, connectionist models will regularise the pronunciations of nonwords whose bodies are most commonly found in regular word. Nonwords with bodies most commonly associated with exceptional pronunciations will be pronounced consistent with irregular words.

Instead of discussing Andrews and Scarratt's (1998) results first, and follow the discussion with a simulation, I will describe their results concurrently with the results of a simulation. Andrews and Scarratt had subjects name two lists of 64 nonwords. LEX named only the nonwords from one of the lists (list b) because the two lists were composed largely of nonwords with the same bodies. LEX named each nonword 16 times and produced plausible pronunciations for 96% of the trials. Andrews and Scarratt had three measures of reading performance: naming latency, the likelihood that a nonword's pronunciation would be regularised, and uncertainty about the nonwords pronunciation. I will discuss each measure of reading performance separately.

Naming Latency

Andrews And Scarratt (1998) noted that nonwords without regular analogies (NRA nonwords) took longer to name that nonwords that had regular analogy bodies (RB). The left panel of Figure 6.23 plots subjects' mean naming latencies for the two types of nonwords. The right panel corresponds to LEX's naming latencies for the same nonwords. As is clear in the right panel, LEX replicates the basic pattern reported by Andrews and Scarratt.

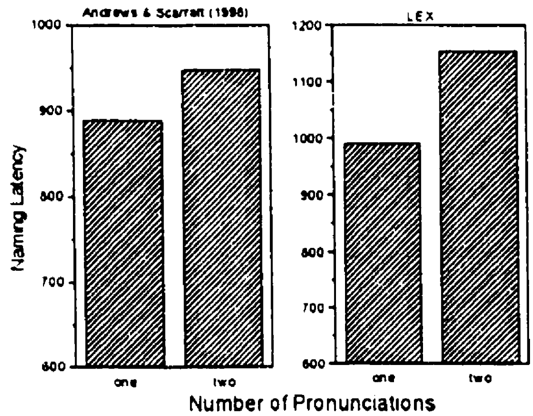

Figure 6.24 Subject and simulation naming latencies for nonwords that yielded one and two pronunciations

Andrews and Scarratt (1998; see also Seidenberg, Plaut, Patterson, McClelland, & McRae, 1995) also noted that nonwords with one pronunciation across subjects were named faster than nonwords that were given two pronunciations. Their data are summarised in the left panel of Figure 6.24. The right panel of the figure contains LEX's mean naming latencies for nonwords to which it gave one or two pronunciations across runs of the model. As is clearly shown in the figure, it also took LEX longer to name nonwords for which it settled on one of two pronunciations than nonwords that were given a single pronunciation.

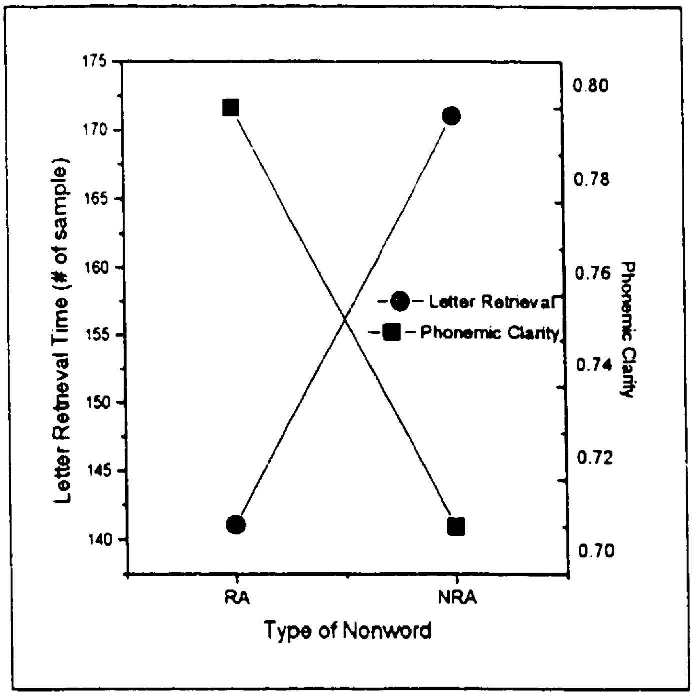

Figure 6.25 Average number of samples required to retrieve each class of nonword and the average phonemic clarity of each class of nonword

Why does it work?

Mean letter retrieval times are illustrated in Figure 6.25. Nonwords that contained regular or inconsistent bodies, contained letter sequences that were common in many words. Half of the NRA nonwords, however, contained bodies that appear in few very few words. Hence, when LEX read NRA nonwords it took longer to retrieve their letters than the letters of nonwords containing bodies of several words.

The mean clarity of the blurriest phonemes for each class of nonwords are also shown in Figure 6.25. Because half of the NRA nonwords had only one word-body neighbour, the letters of NRA nonword were difficult to find in the lexicon. When nonwords are harder to find in the lexicon, LEX requires to take more samples to retrieve the letters than when the letters are easy to find. The more samples LEX is forced to take from the lexicon, the more irrelevant phonological information is copied into the PB. Irrelevant phonological information compromises the clarity of the PB, which in turn, increases naming latency.

Regularisation of Nonword Pronunciation

Andrews and Scarratt (1998) noted that nonword pronunciations tended to be consistent with that of a regular word than an irregular pronunciation; even when the only other words containing the same body were irregular. For example, the letter *i* of the nonword *linth* was pronounced as a short vowel rather than rhyming with the word *ninth*, the only word containing the *inth* body The tendency to regularise nonwords is problematic for strict analogy based models of word and nonword naming. The nonword *linth* would consistently be pronounced to rhyme with *ninth* by a connectionist model because it is the only pronunciation associated with the body *inth* pronunciation during training. On the other hand, the DRC reads nonwords using its GPC rules. According to the rules, the *i* of the letter combination, *inth* should be pronounced with a short vowel because there is no *e* at the end of the string to modify the *i*.

Andrews and Scarratt (1998) compared readers' and the DRC's tendency to regularise nonword pronunciations. Andrews and Scarratt divided their nonwords into seven categories corresponding to how often they were regularised by subjects: 100% of the time, 90-99%, 60-89%, 40-59%, 20-39%, 10-19%, and 0-9%. The open squares of Figure 6.26 shows the average regularisations produced by the DRC for the same nonwords. As is clear in the figure, the DRC generally overestimates the how many nonwords will be regularised. By comparison, the solid circles of the figure correspond to the average number of LEX's regularisations for the same nonwords. Note the clear correspondence between subjects' and LEX's tendency to regularise nonwords. In sum, LEX regularises nonword pronunciations about as often as readers do, and predicts which nonwords will be regularised better than the DRC.

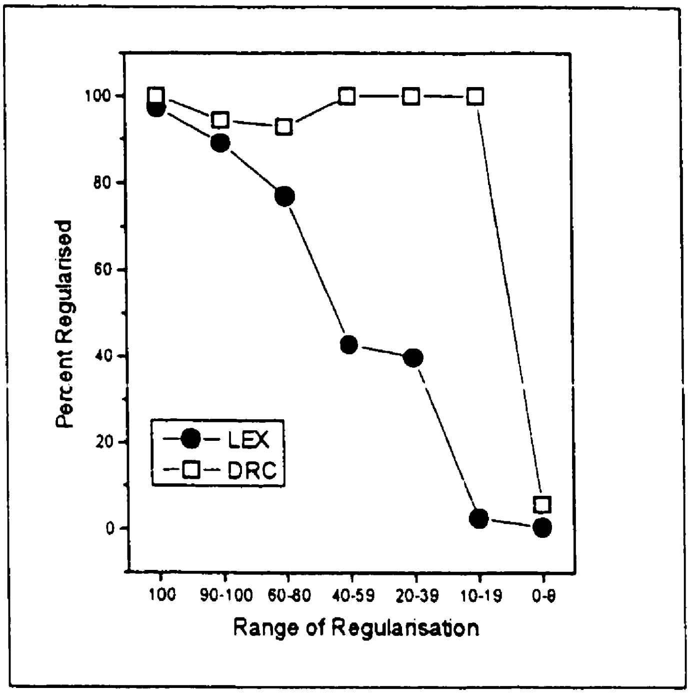

Figure 6.26 A comparison of LEX's and the DRC's tendency to regularise groups of nonwords

While Andrews and Scarratt (1998) did not include a simulation from a connectionist model, it is easy to anticipate how such a model would handle their stimuli. Connectionist models create analogous pronunciations for nonwords from the overlap among nodes at the orthographic and phonological levels. For example, in Seidenberg and McClelland's (1989) model, the pronunciation of a nonword such as *pask* would be derived from a blend of connections between orthographic and phonological nodes that come from words that contain the right wickelfeatures. Specifically, the wickelfeatures used to pronounce *pask* would come from an "average" of words such as (with the wickelfeature in parentheses), *part* (_PA), *past* (PAS), *task* (ASK) and *wisk* (SK_). Likewise, Plaut et al.'s (1996) improvement to Seidenberg and McClelland's (1989) model creates pronunciations for nonwords from the overlap among word components. Hence, the nonword *pask* is pronounced from an average of words such as (with the components in parentheses), *pick* (P onset), *past* (P onset and A vowel), *task* (A vowel and SK coda), and *wisk* (SK coda). Using the componential or wickelfeature scheme, the pronunciation of any letter is directly proportional to the frequency with which it is paired with a particular phoneme. Hence, while the number of nonwords with regular analogy bodies will be accurately regularised, connectionist model will underestimate the degree to which nonwords with NRA bodies will be regularised.

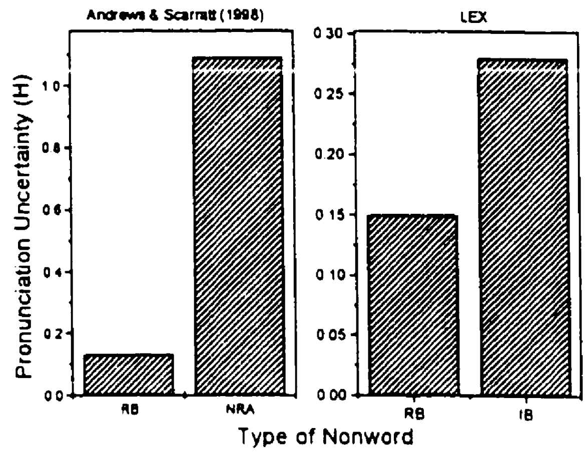

Figure 6.27 Pronunciation uncertainty for each class of nonword

Why does it work?

Like connectionist models, LEX is an analogy-based model. LEX is superior to connectionist models and the DRC in estimating the extent to which nonwords are regularised because LEX does not map letters to phonemes. Instead, the analogy is built from the leftmost to the rightmost letter. By building the pronunciation from left to right, the phonemes at the beginning of the word are determined early and are quite clear. LEX copies phonemes into the PB at a position corresponding to the blurriest phoneme. Because the clarity of an early phoneme, like the *ea* of *kead*, is high, it is unlikely to be overwritten by the analogy.

Pronunciation Uncertainty

Nonwords can often have more than one acceptable pronunciation. For example, the nonword, *kead* could be acceptably pronounced to rhyme with *head* or *bead*. As a final test of how nonword are read by subjects, Andrews and Scarratt (1998) measured the uncertainty with which subjects generated pronunciations for their nonwords. Uncertainty reflects two aspects of pronunciation: the number of pronunciations generated by a nonword, and the degree to which subjects are biased to yield some pronunciations over others. To measure uncertainty, Andrews and Scarratt adopted an uncertainty measure used by Trieman, Mullenix, Bijeljac and Richmond-Welty (1995) where uncertainty (H) is expressed as, $H = \Sigma p \log_2 p$. Where p is the probability of each pronunciation in a sample of correct pronunciations

Andrews and Scarratt (1998) noted that pronunciation uncertainty for nonwords paralleled naming latency. Pronunciation uncertainty was higher for nonwords for which their bodies had no regular analogies than nonwords that had regular analogies. The left panel Figure 6.27 plots H for each type of nonword. The panel on the right plots LEX's pronunciation uncertainty for the same classes of nonwords. LEX clearly replicates the basic pattern of pronunciation uncertainty documented by Andrews and Scarratt.

Why does it work?

Because NRA nonwords contain bodies that have no regular analogies, LEX was sometimes biased to pronounce the nonwords consistent with their irregular word neighbours. In other words, sometimes, *linth* was pronounced to rhyme with *ninth*. Whether LEX pronounces the nonword consistent with an irregular word depends entirely on which phoneme LEX begins copying phonemes after retrieval failure. If, after retrieval failure, the second phoneme in the PB is near perfectly clear, or the blurriest phoneme happens to be the one following the first phoneme of the nonword's body, LEX will pronounce the nonword consistent with an irregular pronunciation. On the other hand, when nonwords have regular analogies, there is often only one pronunciation for the nonword regardless of which phoneme in the PB is the blurriest. In sum, LEX's ability to replicate the pronunciation uncertainty reported by Andrews and Scarratt reflects the fact that nonwords with regular analogy bodies often have only one plausible pronunciation, and nonwords without regular analogy bodies often have more than one.

Chapter 7: A test of LEX's letter encoding assumption

In this chapter, I test LEX's most important assumption; that the reading system expects a list of letters as a retrieval probe for information stored in the lexicon. As I mentioned above, current models of word identification choose input representations for their models that are amenable to the assumption that the letters of a word are processed in parallel. I have built LEX on a different principle; to allow the organisation of the letters guide retrieval. To test the assumption that the reading system requires a list of letters, I used a parafoveal priming technique developed by Rayner and his associates (e.g., Rayner, McConkie, & Erlich, 1978; Rayner, McConkie, & Zola, 1980).

Parafoveal Priming

Imagine fixating a dot in the middle of a display monitor. While you are holding fixation, a letter string (the prime stimulus) is shown briefly in the parafovea at a position to the left or the right of fixation. Shortly thereafter, the dot is replaced by a word (the target). Your task is to name the target as quickly as possible. If the target and prime share the first few letters, you will be able to name the target more quickly than if they do not share the first few letters. The advantage in naming time is known as parafoveal orthographic priming.

Rayner and his associates have used the priming paradigm to study the nature of the information that readers extract from parafoveal text (e.g., Rayner, et al, 1980; see also Rayner, et al., 1978). According to Rayner and his associates, neither semantic nor morphological overlap yields priming (see also Lima, 1987; Lima & Inhoff, 1985; Rayner & Morris, 1992). Although there is evidence that limited phonological information can be obtained during a parafoveal preview (Pollatsek, Lesch, Morris, & Rayner, 1992), priming is largely controlled by the orthographic similarity of the target and prime stimuli. That is, the target stimulus is facilitated when it shares letters with the prime stimulus.

Two details of the overlap are of particular interest. Firstly, facilitation is equivalent when the two stimuli are printed in different cases; we conclude that data obtained from both the target and the prime must be represented at an abstract, non-iconic, level. Secondly, priming is asymmetrical in the sense that it occurs only when the prime shares the first letters with the target. Hence, *bend* will prime the target *bent*, but *rent* will not. The asymmetry is the point of interest for this chapter.

Rayner et al. (1980) proposed a preliminary letter-identification hypothesis to explain the asymmetry for primes that share letters with the target. The hypothesis suggests that readers identify the prime stimulus' first few letters. If its letters match the first letters of the target, identification of the prime's letters establish a context that assists the identification of the remaining letters of the target. Alternately, having identified the first letters, readers may be able to focus attention on the letters that have not yet been identified.

The asymmetry in parafoveal priming can also be interpreted in terms of assumptions required by LEX. LEX postulates that identified characters, represented as abstract letter identities, are ordered in a list. The list serves as a retrieval probe to gain access to lexical memory, and the list's structure is used to guide the retrieval process (Mewhort, Kwantes, & Feldman-Stewart, 1997; see also Mewhort, 1974; Mewhort & Beal, 1977). Specifically, retrieval begins at the first letter in the list and terminates when a match for the last letter has been found.

How does LEX handle parafoveal priming? I postulate that the letters of the prime stimulus ordered in a list and stored in LEX temporal buffer (TB). When the target is displayed, its characters are identified and copied on top of the list in the TB that stored the prime's letters. Note that letters of the target and prime are identified independently, a point consistent with the fact that mismatching the case of the prime and target does not affect the amount of priming. If the letters of the target and prime match, the composite formed by copying one onto the other will be clear; if the letters mismatch, the composite will be noisy.

The composite formed by overprinting letters from the target and prime serves as a retrieval probe to the lexicon. Retrieval starts with the first letter of the list and will be hampered if the composite is noisy. If the noise is toward the end of the list, as in the case when *bend* is used as a prime for *bent*, the retrieval system can use the context provided by the partially retrieved word to aid re-identification the mismatched letters. By contrast, if the noise is toward the beginning of the list, as is the case when *rent* or *xpjk* is used as a prime for *bent*, retrieval cannot start until the letters of the target have been re-identified.

The beginning letters of the target and prime are always on the same side of the character string, the left side in English. Hence, it is unclear whether the benefit provided by *bend* as a prime for *bent* reflects the fact that subjects are biased to identify letters from the left side or the fact that subjects process identified letters in left-to-right order. On the first view, the position of the first letters is known as a reading habit without reference to the stimulus, but on the second, it is a consequence of processing. To distinguish the two possibilities, I need to separate spatial bias from temporal ordering when defining the first letters of a word.

To separate spatial from the temporal priority, I used English-Hebrew bilingual readers. The beginning of an English word is on the left side, whereas the beginning of a Hebrew word is on the right side. If an English-Hebrew bilingual is shown words from both languages in random order, a spatial bias to identify letters from one side would fail, and priming should be attenuated or disappear. If subjects identify the letters and then order them so that the first letter is from the language-appropriate side, priming should occur as usual; that is, priming should occur in English when the target and prime match on the left, whereas priming should occur in Hebrew when they match on the right.

I report two experiments in this chapter: In the first, I replicated the basic phenomenon—the asymmetry discussed above. In the second, I eliminated a confound between spatial position and the beginning of a word by using bilingual English-Hebrew readers naming words of both languages. If naming Hebrew and English words is fastest when preceded by primes that share the first letters, priming must occur because the letters of the prime are ordered by the reader.

Experiment 1

The first experiment was conducted to ensure that I could replicate the studies of Rayner and his associates (e.g., Rayner et al., 1980). On each trial, subjects were given a parafoveal preview of a letter string followed by a word (the target) presented foveally. I varied the prime's similarity to the target by changing its letters to create nonwords that contained three, one, or none of the letters of the target. In addition, I used the target itself as a prime.

Two types of nonwords were created so that common letters with the target were at the beginning or end of the prime. Hence, a target such as *bent* could be primed with *bent*, *benk*, *barp*, *zent*, *lort*, and *xpjk*.

I used nonword primes for two reasons. Firstly, orthographic priming from a parafoveal preview occurs prelexically ; hence, a nonword functions as an adequate prime (Rayner, et al. 1978). Secondly, by using nonwords, I could set the frequency of the prime at zero.

Following the Rayner et al. (1980) study, I anticipated two results. Firstly, primes sharing all or the first three letters with the target should speed naming the target relative to primes that share no letters with the target. Secondly, primes sharing only final letters should not speed target naming.

Rayner et al. (1980) found no reliable priming when the only the first letter was common to both the prime and target. If the first letter of the target and prime match, our account of the asymmetry in parafoveal priming anticipates that the match should provide enough context to yield a modest priming effect. The discrepancy can be explained in terms of the power of Rayner et al.'s experiments. Their experiments included a maximum of five subjects and, hence, may have lacked the power to detect a naming advantage for targets preceded by a first-letter prime. By using a larger number of subjects in our experiment, I expect to find a priming effect using first-letter primes.

Method

Subjects. Twelve students enrolled in introductory psychology at Queen's University served as subjects for the experiment. All subjects had normal or corrected-to-normal vision, and all had English as a first language.

Materials and apparatus. Two lists of twelve four-letter target words were constructed. Each word had a printed frequency between 2 and 10 occurrences per million words of text (Kucera & Francis, 1967). For each word, five nonwords were constructed: one sharing the first three letters with the word, one sharing the first letter of the word, one sharing the last three letters with the word, and one sharing the last letter of the word. A control with no overlap with the target was constructed using consonants (cf., Rayner et al., 1980).

Items were presented on an IBM compatible PC equipped with an SVGA monitor. Subjects responded to stimuli by speaking into a head mounted microphone that triggered a response switch. The timing and screen control routines were taken from Heathcote (1988). Subjects sat 1.2 meters from the computer's monitor in a darkened room. Stimuli were presented in white letters on a dark background and subtended a visual angle of 1.19°.

Procedure. Each trial started with a fixation dot presented at the centre of the computer's monitor. After 1200 ms, a prime was presented for 184 ms to either the left or the right side of the dot. Seventeen ms after the prime's offset, a target word was presented foveally; it remained until the subject named it. The space between the dot and the first letter of the prime to the right of fixation subtended a visual angle of 2°. The space between the dot and the last letter of the prime to the left subtended the same visual angle.

The subjects were required to name the target presented in the middle of the screen as quickly and accurately as possible. They were informed that the targets would be preceded by a letter string and that the string would be shown so briefly that they would be unable to shift they eyes to it. Instead, they were required to hold fixation at the dot. Subjects were cautioned to avoid making sounds that would trigger the voice-operated response key.

Table 7.1

Mean Naming Latency (in ms) for Targets in Each Priming Condition in Experiment 1 (Standard Deviations are Shown in Parentheses)

	Prime letters in common with target					
Side of fixation	All	3 left	1 left	3 right	1 right	no letters
Right	468	486	498	508	516	510
	(58.3)	(62.6)	(70.0)	(60.7)	(75.9)	(80.3)
Left	470	491	489	502	500	498
	(55.0)	(63.9)	(60.8)	(72.3)	(70.8)	(76.8)

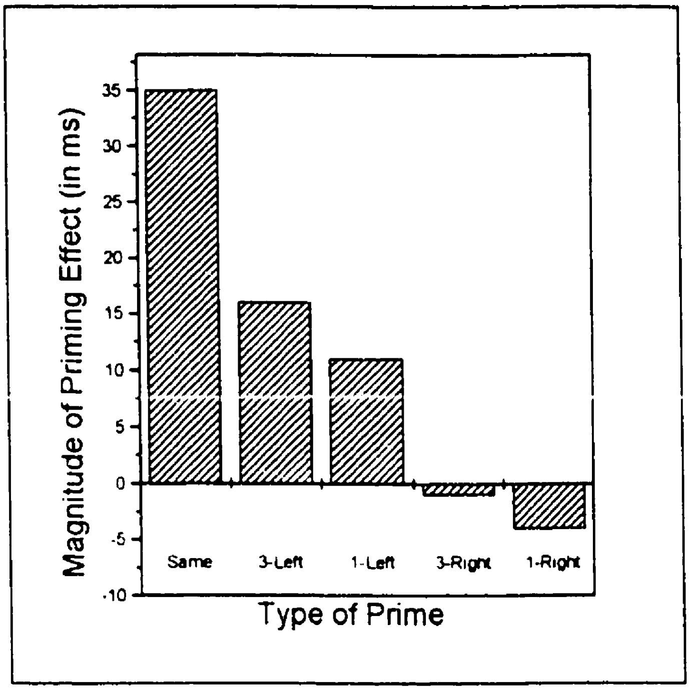

Figure 7.1 Magnitude of the priming effect for each overlap condition relative to the control condition

After each trial, the experimenter scored the trial (correct or incorrect) and typed the decision into the computer The fixation dot signalling the next trial re-appeared immediately after the trial had been scored.

Subjects performed 20 practice trials prior to reading the words of each list. After the practice trials, which subjects read each list of words twelve times.

Design. Each subject named 24 words in each of 12 priming conditions for a total of 288 trials. On separate trials, a target was preceded by itself, and by each of the five nonwords that had been constructed for the word. Primes were presented once to the left side of fixation and once to right of fixation.

The 288 experimental trials were divided into 24 blocks of twelve trials. Within each block, a target was presented once. As well, within each block, each of the twelve priming conditions was represented once. Item order was randomised for each block, and the order of the priming conditions was counterbalanced across blocks. After each block of 12 trials, subjects were given an opportunity to rest.

Results and Discussion

There were only six trials (out of the total 3456) on which subjects misnamed the target word. I focus, therefore, on naming latency.

Mean naming latency for correct trials is shown in Table 7.1 as a function of priming condition. Naming latency was independent of the position of the prime relative to the fixation point; when the prime was to the right (shown in the top row of the table), naming latency was 496 ms and when it was to the left (shown in the third row of the table), naming latency was 492, $F(1, 11) = 2.0$, $.15 < p < .20$. The position factor did not interact with prime type, $F(5, 55) = 1.5$, $.2 < p < .25$. Because the position of the prime relative to the fixation did not affect performance, we collapsed across that factor in the subsequent analyses.

Figure 7.1 shows the magnitude, in ms, of the priming effect for each condition relative to the no-overlap control. To calculate the priming effect, we subtracted each subject's mean naming latency for the zero overlap condition from the scores for the remaining five conditions.

As is clear in Figure 7.1, only prime stimuli sharing the left-most letters with the target word decreased naming latency for the target word. Using the target as a prime stimulus yielded larger priming than any other condition, $F(1, 11) = 33.31$, $p < .001$. Priming occurred when the letters on the left of the prime overlapped the corresponding letters in the target but not when the letters on the right overlapped the target, $F(1, 11) = 47.10$, $p < .001$. There was no reliable difference between the one-letter and three-letter overlap conditions, $F(1, 11) < 1$.

Recall that Rayner et al. (1980) found priming only when at least the first two letters overlapped. My results show the same trend—greater priming with three-letters overlap than with one-letter overlap—but the advantage for three-letters over one-letter overlap was not reliable. The difference between our results and those of Rayner et al. can be understood in terms of the statistical power of the two experiments.

The experiment confirms the asymmetry in parafoveal priming reported by Rayner et al. (1978, 1980): orthographic priming occurs only when the prime share the first letters of the target. The question remains, however, why it occurs. Rayner et al.'s account postulates that subjects identify the first letters of the prime, information that they use when naming the target. The account works only if the subjects can anticipate where the first letters are positioned.

Experiment 2

Experiment 2 was conducted to demonstrate orthographic parafoveal priming under circumstances that do not allow the subjects to anticipate the position of the first letters of a word. When subjects cannot anticipate the position of the first letters, under Rayner et al.'s (1980) account, priming should be attenuated, or disappear. By contrast, LEX postulates that subjects identify the letters and then order them. Priming reflects the clarity of a composite formed by overwriting the characters of the prime by the characters of the target. Because ordering is performed after the letters have been identified, however, subjects do not need to anticipate the position of the first letters. Hence, even under circumstances that do not allow the subject to anticipate the position of the first letters, LEX predicts that priming will occur provided that the prime and target share the first letters.

To deny subjects the ability to anticipate the position of the first letters of a word, we asked bilingual Hebrew-English readers to name both Hebrew and English words. English is read from left to right, whereas Hebrew is read from right to left. By mixed the languages randomly across trials, we ensured that subjects could not anticipate the position of the first letters of the prime.

Method

Subjects. Four Israeli students and three native Canadian Hebrew-English bilingual students at Queen's University (5 undergraduate, 2 graduate) served as subjects. Subjects were paid $7 for their participation. All had normal, or corrected-to-normal vision. All subjects reported reading both languages for recreation on a regular basis.

Materials. The stimuli were 48 four-letter words, 24 in English and 24 in Hebrew. Word frequency for the English words was held between 18 and 25 occurrences per million, according to the Kucera and Francis (1967) norms. The frequency of the English words was higher here than in Experiment 1 to ensure that the subjects would be familiar with the items. Frequency norms are not available for Hebrew words; a Professor of Jewish Studies at Queen's University verified that the Hebrew words would be familiar to Israeli readers.

As in the first experiment, each word served as a prime stimulus for itself as a target. As well, five nonwords were created to serve as primes. The nonwords shared zero, one or three letters with the target. Two of the nonwords shared the target's first letters, and two shared the target's last letters.

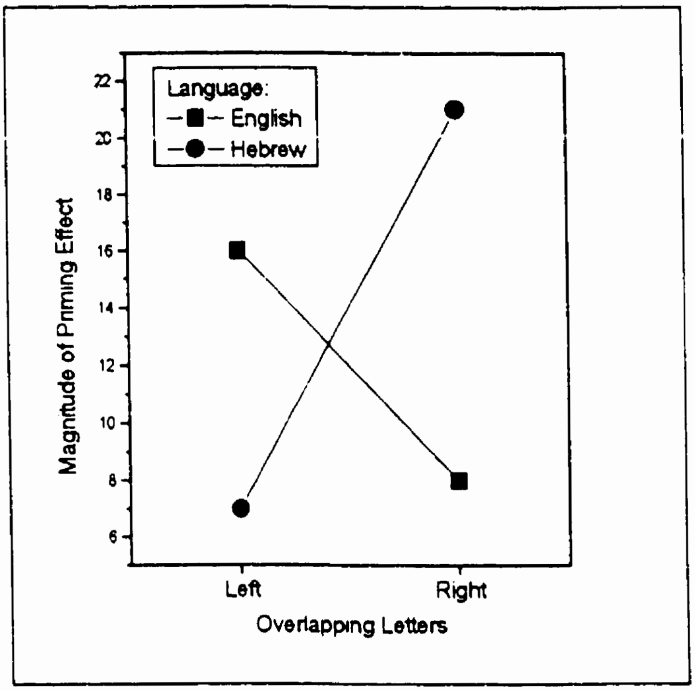

Figure 7.2 Magnitude of the priming effect in both languages for targets preceded by prime stimuli overlapping on the left or right letters

I created fonts for the alphabets of both languages so that the letters were defined using the same basic components (vertical, diagonal, and horizontal lines). Lines were two pixels in thickness. The characters were defined within a matrix of 23 by 15 pixels. Characters within each letter string were separated by 7 pixels. The stimuli subtended the same visual angles used in Experiment 1. The hardware and timing routines were the same as those used in Experiment 1.

Procedure. Prior to the experiment, subjects were shown each word and asked to name it aloud. They were also asked to indicate whether any of the words were unfamiliar. Subjects reported that all of the words were familiar. With the exception of the preview of the targets, the procedure was identical to that of Experiment 1.

Design. Each subject named 48 words in each of 12 priming conditions for a total of 576 trials. On separate trials, a target was preceded by itself, and by each of the five nonwords that had been constructed for the word. Primes were presented once to the left side of fixation and once to right of fixation.

The 576 experimental trials were divided into 24 blocks of 24 trials. Within each block, a target was presented once; half the targets were English words and half were Hebrew words. Within each block, each of the twelve priming conditions was represented twice. Item order was
randomised for each block, and the order of the priming conditions was counterbalanced across blocks. After each block of 24 trials, subjects were given an opportunity to rest.

To make it easier to read the Hebrew targets, they were pointed (i.e., optional vowel markings were included). To avoid confounding the size of the prime's characters with language, I did not include the vowel markings when Hebrew letters were used as a prime stimulus. All seven subjects reported, however, that they ignored the vowels when they named the Hebrew words.

Results and Discussion

As in Experiment 1, subjects made few errors; they mispronounced the target item on only thirteen of the 2304 trials (5 Hebrew and 7 English trials). I focus, therefore, on naming latency.

Mean naming latency for correct trials is shown in Table 7.2 as a function of priming condition and language. Naming latency was independent of the position of the prime relative to the fixation point: When the prime was to the left, naming latency was 525 ms, and when it was to the right, the corresponding latency was 523 ms, $F(1, 11) < 1$. The position variable did not interact with prime type, $F(5, 30) = 2.7$, $.15 < p < .20$, or with language, $F(1, 6) < 1$. Because the position relative to the fixation did not affect performance, subsequent analyses collapsed across that factor.

As in Experiment 1, I analysed the magnitude of the priming effect across conditions. As before, I subtracted each subject's mean naming latency for the zero overlap condition from the mean latency for the five priming conditions, but I made the calculation separately for each language.

The chief data of main interest concern the size of the priming effect as a function of the type of prime and the language of the target. As shown in Figure 7.2, naming an English target was facilitated only when the target and prime shared the left letters, i.e., the first letters. Similarly, naming a Hebrew word was facilitated only when the target and prime shared the right letters, $F(1, 6) = 18.05, p < .01$.

In addition to the interaction between the prime and language documented in Figure 7.2, naming latency for the target decreased as overlap between of the target and prime increased; that is, the advantage for targets preceded by primes containing all the letter target's letters (58 ms) was greater than the advantage for targets preceded by primes containing fewer of the target's letters (14 ms), $F(1, 6) = 43.5, p < .05$. In particular, a prime that shared three letters with the target shortened naming more (21 ms) than primes that shared only one letter (6 ms), $F(1, 6) = 16.4, p < .05$. Hence, using only seven subjects, a number close to the number of subjects tested by Rayner et al. (1978; 1980), the present results, like Rayner et al.'s, indicate that the target and prime must overlap more than one letter to obtain priming.

The results provide a clear confirmation of the prediction derived from LEX: Even when subjects cannot anticipate the position of the first letters of the prime stimulus, naming the target was facilitated only when the prime shared the first letters with the target. Because the subject could not anticipate the position of the first letters without identifying the letters of the prime, priming cannot reflect a bias to identify the first letters. If one assumes that subjects identify the characters and then order them, the priming data can be understood in terms of interference that occurs when the first letters of the target and prime mismatch.

General Discussion

In two experiments, I have confirmed that the naming latency for a target word is shortened by a parafoveal prime only if the prime overlaps the first letters of the target. Experiment 1 allowed subjects to anticipate the position of the target's first letters and replicated the asymmetry reported by Rayner et al. (1978; 1980). In Experiment 2, I denied subjects the ability to anticipate the position of the first letters and showed that the asymmetry persists.

Table 7.2

Mean Target Naming Latencies (in ms) for Each Priming Condition in Experiment 2 (Standard Deviations shown in parentheses)

		Prime letters in common with target					
Language		Same	3 left	1 left	3 right	1 right	no letters
Hebrew	Side of fixation						
	Right	502	558	562	526	554	570
		(77.2)	(72.6)	(73.8)	(74.4)	(85.5)	(81.3)
	Left	504	548	556	536	549	555
		(69.0)	(74.0)	(75.4)	(71.2)	(74.5)	(84.2)
English	Side of fixation						
	Right	465	509	514	501	522	520
		(54.7)	(44.0)	(40.5)	(34.5)	(37.3)	(41.3)
	Left	467	487	508	508	520	522
		(47.0)	(43.6)	(38.8)	(50.3)	(57.2)	(42.1)

The pattern of results is not consistent with accounts of orthographic priming that depend on the subjects' ability to anticipate the position of the target's first letters. The pattern is consistent with—indeed it was predicted by—LEX. According to LEX, a reader first identifies the letters of a word and stores them in first-to-last order (see also Mewhort, 1974; Mewhort & Beal, 1977). First-to-last structure can only be determined after the characters have been identified; left-right for English and right-left for Hebrew (see Butler, Tramer, & Mewhort, 1985). The list structure controls retrieval of the target word's pronunciation from the lexicon.

According to the theory, parafoveal orthographic priming occurs because the list derived from the target overwrites the list derived from the prime. If the two lists do not share the same first letters, the composite representation is noisy, and its effectiveness as a retrieval probe is reduced. As a result, naming the target is slow. By contrast, if the two lists share the same first letters, the composite list is clear and provides an effective retrieval probe for the target.

Rayner et al. (1978; 1980) treat orthographic priming as a faciliatory phenomena; that is, the prime facilitates identification of the target's letters. By contrast, LEX treats orthographic priming as an interference phenomenon. Overlapping the first letters of the prime and target allows faster naming of the target than the no-overlap condition because overlapping the first letters reduces the interference inherent in the control case.

The difference between the account of parafoveal priming proposed by Rayner et al. (1980) and LEX's account is analogous to the difference between early- and late-selection accounts of attention. In an early-selection account, attention operates on precategorical data to facilitate the identification process. In late-selection accounts, attention operates on postcategorical data to select objects for further processing. Rayner et al.'s account, like early-selection accounts of attention, claims that priming is facilatory in the sense that subjects use the first few letters of the prime to assist identification of the corresponding letters of the target. By contrast, LEX claims that the apparent facilitation reflects reduced inhibition at a post-identification stage.

The evidence is based on the manipulation of subjects' ability to anticipate the position of the One might object to my evidence for LEX's account of priming on the grounds that I have not ruled out all possible ways that a subject might anticipate the position of the first letters of the target.

I made the Roman and Hebrew alphabets alike in terms of the basic features out of which they were constructed. However, one could argue that subjects were biased to identify the letters on one side of the parafoveal string on the basis of the visual characteristics of the letters. To test the potential confound, I compared the visual similarity of the two alphabets. Each letter is represented as a matrix of pixels. To measure the visual similarity between any pair of letters, I assigned an inactive pixel a value of -1, and an active pixel a value of +1. I evaluated the similarity between any two letters by calculating the dot product of the vectors created by concatenating rows of the matrices representing the two letters. I scaled the dot product by dividing it by the number pixels pairs being compared to make it analogous to a Pearson's correlation coefficient. I calculated the scaled dot product for every letter with every other letter excluding itself. Roman characters had an average similarity value of .54 with each other. Hebrew characters' average similarity value to other Hebrew characters was .49. Most importantly, Hebrew characters' average similarity to English characters (.51) was almost equal to Hebrew characters' similarity to each other. It is unlikely, then, that subjects were biased to identify letters on one side of the parafoveal display on the basis of visual characteristics of the letters.

Implications for Current Models of Word Recognition

LEX is unique in its use of a list structure as the input to the lexical access system. The data structure I use solves a representational problem common to other models of word recognition. Current models have not attempted to include a psychologically plausible representation for storing identified letters.

Connectionist models of word identification, assume that all the letters of a word are processed simultaneously to derive a pronunciation. Consequently, connectionist models are forced to somehow represent each letter's position relative to the others using wickelfeatures (Seidenberg & McClelland, 1989) or componential representations (Plaut, et al., 1996). Both representational schemes are chosen as a convenience, not as psychologically plausible data structures.

I have provided a psychologically plausible data structure for identified characters. How would a connectionist model derive a pronunciation from letters stored in a list? The first challenge would be finding a way to represent a list within the nodes of a neural network. If a list can be represented in a connectionist model, perhaps one way to derive a pronunciation for a letter string would be to allow a network to operate sequentially through the list of letters. Clearly, this is an unattractive strategy for proponents of connectionist models— it is inconsistent with the assumption of parallel processing that they embody.

The IAM (McClelland & Rumelhart, 1981) and, by inheritance, the DRC model of word identification (Coltheart, Curtis, Atkins, & Haller, 1993), represent letter positions by storing characters in separate letter channels. The letter channels store ordinal position information without any commitment to a data structure for the letters. Mewhort and Johns (1988) have criticised the use of letter channels claiming that, while the IAM represents space explicitly within a letter channel, space is not represented between channels. If the organisation of the letters is not explicitly spatial, on what basis are identified letters positioned relative to each other?

Paradoxically, a list structure, such as LEX prefers, would be ideal for a version of the dual-route model. The DRC's grapheme-to-phoneme conversion mechanism, the non-lexical route, operates in a left-to-right direction through the letters of the word. A list structure provides a natural guide for the non-lexical route.

To summarise, the data from the previous experiments suggest that readers order identified characters from beginning-to-end prior to lexical access. Current models represent the relative position of letters to one another in a fashion that permits parallel access to lexical information. I take a different approach— I allow the organisation of the letters dictate how lexical access occurs. Hence, in LEX, lexical access begins at the first letter of a word, and terminates when the final letter has been retrieved from the lexicon.

Chapter 8 : General Discussion and Conclusions

Comparison and Contrast to Other Perspectives

In this section, I will outline in more detail how my approach differs from other approaches and discuss some of the commonalties among the approaches.

Serial vs. Parallel Processing.

LEX's approach to reading is clearly at odds with the traditional connectionist approach to modelling word identification. In connectionist models letters of a word are processed simultaneously. Phonemes of a word are also delivered simultaneously. That is, processing is done in parallel. I agree that reading involves parallel processing—the letter identification system I use as LEX's front end (LEPS) identifies and localises the letters of a display in parallel. However, the lexical access system expects a list of letters for a retrieval cue. If the list's structure is exploited during retrieval, lexical access is sequential. In this respect, LEX aligns itself with the dual-route approach to reading. LEX and the DRC operate from left to right through the letters of a word. However, unlike the DRC, it is lexical access, not grapheme-to-phoneme conversion that is sequential in nature.

One vs. Two Routes to Pronunciation.

Like connectionist models, LEX reads words and nonwords using a single route to pronunciation; LEX reads novel stimuli by analogy to the words that it knows. It is worth noting however, that the analogies are built differently in the two classes of model. For example, LEX is sensitive to the consistency with which sound can be derived from a word's spelling for very different reasons than those offered by connectionist models. Connectionist models reproduce the naming advantage for regular words over irregular words because of the frequency with which letter patterns are mapped to sound patterns during training. LEX captures the advantage because of the frequency with which letters and phonemes co-occur in lexical entries. LEX does not link letters, or letter clusters, to sound; hence, when it exhibits sensitivity to spelling-to-sound consistency, it is because, during sampling, it retrieves words from the lexicon that share the same letters but whose associated phonological patterns contain phonemes with more frequent pronunciations.

Despite LEX's success with simulating word and nonword naming with a single route, I am not willing to claim that readers do not have a second route to pronunciation similar to the grapheme-to-phoneme route in the DRC. At issue is not whether readers have a nonlexical route, but whether readers always use it. In LEX's account, both words and nonwords are read as analogies to the words that are contained in the reader's lexicon—up until the last character, the space, has been found, LEX must often sample several candidates in the cohort of possible matches to the target word. If both words and nonwords can be read by analogy, there is no need to postulate a separate mechanism to read nonwords. It is plausible that a rule-based route that converts spelling to sound, if it exists, is necessary only when a word cannot be read by analogy.

Storing Mappings vs. Storing Data.

Connectionist models are generally offered as neurologically inspired models of intelligent behaviour. I have tried to remain neutral with respect to claims about the extent to which LEX's architecture is neurological inspired. LEX shares some similarity to connectionist models of reading: it uses vectors to represent words, and it obtains information about a word by blending information across several pieces of data stored in the system's memory.

Although LEX and a connectionist model such as Seidenberg and McClelland (1989) blend information to identify a word, it is done very differently in the two models. Connectionist models of word identification store mappings between letters and sounds in the connections between layers of nodes. Because the mappings link letters to sounds, connectionist models can use the information for that purpose only. For example, Seidenberg and McClelland's model of reading can map letters to sounds, but it cannot, within the same network, use sounds as an input to get spelling information. By contrast, LEX stores words. Because LEX stores words, it can use the information contained in the lexicon for more than one purpose—it can retrieve phonology from the lexicon when letters are used as a probe, or it can retrieve the spelling of a word when it is presented with a string of phonemes.

Other Empirical Issues

List-Structure Effects in Word Identification.

None of the simulations I reported in Chapter 6 dealt with readers' apparent ability to readjust how they read letter strings strategically when the type of letter string can be anticipated. Several experiments have demonstrated that the speed and accuracy with which subjects can read words or nonwords changes when they are mixed with other stimuli (e.g., Baluch & Besner, 1992; Marmurek & Kwantes, 1996; Monsell, Patterson, Graham, Hughes, & Milroy, 1992). For example, Monsell et al. showed that subjects could read irregular words more quickly when they were presented in pure lists of irregular words than when the words were presented in a list containing nonwords. They interpreted the difference as evidence for a dual-route reading system. Specifically, they suggested that the difference reflected a reader's ability to strategically attend to the output of one route to pronunciation over the other.

I have not given LEX the ability to change strategies while reading because doing so requires that I build in extra mechanisms that a) tell the model when it has made an error (so that it knows when it must be more careful on the next trial), and b) allows LEX to anticipate what kind of letter string it will be reading (so that it can choose the most efficient way to read them).

One possibility for a strategic component for LEX can be found in recent work by Lupker, Brown, and Columbo (1997). Lupker et al. noted that, in experiments using the list structure manipulation, increased speed often comes at the cost of accuracy. In fact, the correlation between the increase in speed and the decrease in accuracy for pure over mixed lists of stimuli across the four experiments reported by Lupker et al. was -0.615 ($p < .01$). Speed-accuracy trade-offs are more consistent with a strategic change of response deadline than a strategic de-emphasis of one strategy for pronunciation over another. LEX could incorporate such a strategic component. The only point at which LEX may know it has made an error is during letter retrieval—sometimes LEX settles on the wrong letter during retrieval from the lexicon. LEX has a parameter that controls how much the echo content is allowed to change over successive samples before retrieval begins on the next letter. Suppose that, over several words, letter retrieval during lexical access occurs without error. Every time a word is retrieved without failure, the parameter could be relaxed slightly so that each letters would be retrieved more quickly. At some point, the parameter would be so lax that the system would make an error retrieving the letter. At the point where retrieval fails, the parameter could be readjusted to make the criterion more stringent. If the criterion were adjusted dynamically in this fashion, homogeneous words in a list would be read more quickly, but more prone to errors, than the same words in a list also containing other stimuli like nonwords.

Acquired Dyslexia

It is fashionable to include a discussion of how a model accounts for reading deficits associated with head trauma. I did not include simulation data for any of the acquired dyslexias in this thesis. Instead, I will discuss briefly how LEX can account for three acquired dyslexias that are often taken to represent the evidence for different routes to pronunciation. Missing from the list of acquired dyslexias is *deep dyslexia* which is characterised by a reader's tendency to make semantic errors while reading (e.g., reading *tree* as *bush*). LEX does not have a semantic system yet, hence I must forego discussion about how the model will account for it.

Phonological dyslexia. Phonological dyslexia is characterised by a reader's difficulty reading unfamiliar or new words (Funnell, 1983; Beauvois & Derouesne, 1979; Shallice & Warrington, 1980; Patterson, 1982). Persons with this syndrome can read words quite easily, especially familiar words. In the dual-route interpretation, phonological dyslexia represents selective damage to the GPC route. Plaut et al.'s (1996) interpretation placed the responsibility on the contribution of a semantic system in reading words. If the phonological pathway of a connectionist network is damaged, the model can enlist the aid of the semantic system to name words. Because neither nonwords nor new words have representations in the semantic system, nonwords cannot be named without great difficulty.

LEX would exhibit the symptoms of phonological dyslexia if, after it fails to retrieve the correct letter from the lexicon, it could not properly readjust the cohort of candidate words to continue the search. Recall how LEX reads: As each letter is retrieved, only lexical entries that are consistent with the probe up to that point are available for sampling. Now, suppose LEX was reading the nonword *burse*. Up until the final *e*, the letter string could be a word, and when LEX tries to retrieve the final letter, it will fail. When LEX is intact, the first letter is dropped from the probe, allowing the remaining letters in the probe to resonate with the entries in the lexicon containing the letter combination, *urs* in the second, third, and fourth letter positions. Now, when LEX searches for the *e*, the search will likely be successful because it does so in a search space that includes words such as *nurse, purse,* and *curse*. On the other hand, if LEX lost the ability to readjust its search space, it could not read novel stimuli effectively.

Surface dyslexia. Readers with surface dyslexia can read novel stimuli and regular words with ease relative to irregular words. Irregular words such as *pint* are often regularised; that is, *pint* is pronounced to rhyme with *mint* (Marshall & Newcombe, 1973).

From the dual-route perspective, surface dyslexia represents selective damage to the lexical route. Reading is mediated entirely by the undamaged nonlexical route and readers appear to "sound out" everything they read. Marcel (1980) and Henderson (1982) pointed out that such a syndrome is also consistent with a readers' inability to create accurate or appropriate analogies to words they know. From a connectionist perspective, Plaut et al. (1996) suggested, and provided simulation evidence, that surface dyslexia arises from damage to the semantic system. Without aid from semantics, the phonological pathway of their model is responsible for generating a pronunciation. Plaut et al.'s notion that the semantic system is involved is corroborated by neuropsychological case studies of patients with certain types of dementia. For patients with semantic or Alzheimer-type dementias, there is an increase in regularisation errors associated with the progression of the disease.

The other possibility offered by Plaut et al. was that surface dyslexia reflects damage to the connections between levels of units within the network. They demonstrated that, when the network was lesioned between the hidden and phonological units, its performance on word reading approached that of a patient with mild surface dyslexia.

In terms of LEX's mechanisms, surface dyslexia reflects a faulty letter retrieval process. Whenever letter retrieval fails, LEX must readjusts its search space to continue retrieval. If LEX is prone to making letter retrieval errors, constant readjustment of the search space will have little effect on nonword or regular word reading. On the other hand, retrieval failure will be highly detrimental to reading irregular words; such words will often be regularised.

Word-form dyslexia. With word-form, or letter-by-letter, dyslexia, a reader appears to name each letter prior to naming the word. The syndrome has been interpreted as reflecting the reader's use of the spelling system to help name a word (Patterson & Kay, 1982; Shallice & Warrington, 1980).

LEX would have word-form dyslexia if the scanning mechanism that orders letters into a list was damaged. Recall that the scan provides LEX with a list of letters that is used for lexical access. From LEX's perspective, a person with word-form dyslexia has damaged the mechanism that orders the letters. To substitute for the scan, the reader sub-vocalises the letters to impose order on them prior to lexical access. As far as I know, LEX is the only model that can account for, or even predicts, word-form dyslexia without the need for a separate spelling system.

Priming Effects

LEX possesses two pre-lexical buffers that store letter information in different organisations. The character buffer (CB) stores identified characters in a spatial array. That is, the buffer represents letters' identity and location. The contents of the CB are sent to the temporal buffer (TB) by the scan where they are stored as a temporal array; the letters are stored in a beginning to end order and, I have argued, that it is the organisation of the contents in the TB that LEX expects in order to access the lexicon. With separate buffers capable of storing information, LEX has two putative loci for orthographically based priming effects.

I have not started simulation work on priming effects. Nevertheless, I can briefly discuss how LEX might accommodate masked orthographic priming (e.g., Forster & Davis, 1984; Bodner & Masson, 1996) and review LEX's account of parafoveal priming (e.g., Rayner, McConkie, Erlich, 1978; Rayner, McConkie, & Zola, 1980).

Masked Orthographic Priming or Form Priming. In one variation of the orthographic priming task, a reader is presented with a pattern mask (often a string of nonlinguistic characters like, &&&&) that is followed by a brief presentation of a letter string, the prime stimulus. Immediately following the prime stimulus' offset, a target word is presented which the reader must name aloud. The speed with which a reader can name the target is affected by the orthographic similarity of the prime stimulus to the target such that, naming latency shortens as the prime and target share more letters in the same letter position.

As long as the prime and target stimuli share letters in the same position, target naming is facilitated. The only other boundary condition for the effect is that the prime stimulus' presentation must be too brief to allow the subject to identify it. Typically, the degree to which an orthographically similar prime facilitates target naming is independent of the target word's frequency—a pattern suggesting strongly that orthographic priming occurs at a pre-lexical stage of word identification (see Bodner & Masson, 1998 for futher evidence that orthographic priming is nonlexical in nature).

I place responsibility for orthographic priming at the level of LEX's character buffer. This is the only pre-lexical buffer that stores letters spatially. At the level of the character buffer, when the letters of the prime and target coexist in the CB, and as long as they share letters in the same positions, target naming will be more rapid than when none of the letters of the stimuli overlap.

Parafoveal Priming. Rayner, McConkie, and Zola (1980) demonstrated that, under some circumstances, a brief exposure to a prime stimulus in the parafovea facilitates naming a foveally presented word. The facilitation is referred to as parafoveal priming. Interestingly, facilitation only occurs if the prime and target stimuli share their first letters. That is, the priming is asymmetrical; presenting *rend* in the parafovea facilitates naming *rent* when it is presented in the fovea, but there is no facilitation for *rent* when *bent* is the parafoveal stimulus. Rayner et al. (see also Rayner, McConkie, & Erlich, 1978) suggested that the asymmetry occurs because the first few letters of the prime stimulus are identified. When the target stimulus is presented, the identified characters may serve as a context for aiding word identification during the foveal presentation of the target, or may allow readers to focus their attention on the letters that have not yet been identified.

I showed in the previous chapter that the asymmetry in parafoveal priming can also be explained by LEX. The phenomenon reported by Rayner et al. (1980) might reflect interference from the letters in the TB. Recall that, the TB stores letters in a beginning-to-end order. Hence, from the perspective of the TB, *rend* is similar to *rent* because they are the similar starting from the beginning of the letter strings. On the other hand, *bent* is dissimilar to *rent* because they differ starting at the first letter. Because of their relative similarities to the target word, *rent* should be named more quickly when preceded by *rend* than when preceded by *bent*.

According to LEX's account, the first letter in the TB must be relatively clear to serve as an adequate retrieval cue. If the first letter of the probe is unclear because it overlaps with the first letter of the prime, the letters of the cue must be reidentified. If the first letter is clear, retrieval can begin. At the point where a probe letter becomes unclear, LEX could use the context provided by the retrieved orthography so far to aid the re-identification of the remaining letters.

Conclusions

In this thesis, I introduced LEX, a model of visual word identification with a full-scale lexicon, and few retrieval mechanisms. I demonstrated that LEX is able to reproduce several phenomena considered to be benchmarks for the validation of any model of word identification. LEX's success derives from two sources. First, it treats reading as retrieval from memory, and second, it uses a full-scale lexicon.

Reading as retrieval

LEX treats reading differently from other models. Most models treat reading, and learning to read, as an operation by which letters (or letter groups) are translated to, or mapped onto, sounds. LEX does not map letters onto sounds. LEX treats the naming and lexical decision tasks as special instances of cued recall and recognition memory tasks, respectively. Identified characters serve as a retrieval probe to get orthographic and phonological information from the lexicon. Hence, for LEX, any correlation between orthography and phonology that is reflected in the model's performance arises because orthographic and phonological information exist in the same memory trace. An example may help to clarify the distinction. Consider the letter string *aaa*. If LEX mapped letters to sounds, it might pronounce the string as *ah-ah-ah*. Instead, LEX pronounces *aaa* as *t-r-ih-p-uh-l-ay* because it is the phonological pattern associated with *aaa* (I have used the phoneme notation found in appendix A). If reading is treated as a problem of memory retrieval, one is forced to rethink other assumptions about word identification as well. I will discuss each of the assumptions in turn.

Most models describe word identification in terms of the activation of units which can either represent whole words or parts of words. Reading is described, in models such as the DRC and the connectionist models, as the process by which information from an input level is filtered through a series of processors until the information activates processors at an output level. As I mentioned briefly in Chapter 1, such a perspective makes reading akin to a perceptual process, and as such, it precludes the need for mechanisms that store information. When we treat reading as a problem of retrieval from memory, the story changes drastically—suddenly, we require buffers to store the retrieval cue and the information that falls out of memory.

Storing a retrieval cue in a buffer introduces a new constraint on the reading system; the information contained in the retrieval cue, in LEX's case, identified characters, must be structured. On the basis of previous work done on letter and word identification using tachistoscopic displays (Mewhort, 1974; Mewhort & Beal, 1977; Mewhort & Campbell, 1981), I assume that the reading system expects a list of letters. That is, identified characters are encoded into a list prior to lexical access. LEX uses the list structure to guide retrieval. Retrieval from the lexicon begins with the first letter in the list and terminates when the final letter has been found. The orthography that is retrieved from the lexicon acts as a control structure to guide retrieval by adjusting a cohort of candidate words on the basis of the information that has been retrieved.

I am quite clearly attacking the problem of input and output representations differently from my competitors. Typically, assumptions about how lexical access occurs guide the choice of input and output representation schemes for a model. For example, one who assumes that lexical access is performed in parallel chooses input and output representations that are amenable to parallel processing. Wickelfeatures, letter channels, and componential representations are used by theorists because they are convenient strategies for representing an arrangement of characters that can be processed simultaneously. Despite the widespread use of such representations, little effort is spent testing, or arguing for, their psychological reality. I have taken the opposite approach. First, I considered what type and form of information the lexical access system expects.

On the basis of what the system expects, I theorised about how lexical access would use those data to make access to the lexicon.

Model-size and life-size lexions

LEX possesses a life-size lexicon. Most models implement a lexicon of between 3000 and 7000 words. While the size of a lexicon does not *necessarily* make one model better than another, using a full-scale lexicon has two advantages over a small one. First, and most obviously, a life-size model represents a closer approximation to a human's knowledge of the language; and computing power is cheap enough that, with a little data abstraction, full-scale lexicons are easily implemented in a computational model of reading. Second, only a full-scale lexicon gives the theorist hints about which mechanisms are necessary and adequate for reading.

My second point deserves expansion. By definition, a model is a scaled down version of a larger system. It is not at all surprising that theorists build small-scale lexicons in their models. Indeed, if the mechanisms that a theorist postulates are basically correct, the amount of lexical knowledge that the system possesses should be independent of how closely the model represents the life-size system. On the other hand, small-scale models can lead researchers into postulating unnecessary mechanisms to account for a wider range of data. A good example of this danger is found in LEX's sensitivity to BOSSes and syllabic structure in word identification tasks. LEX does not represent BOSSes or syllables; its sensitivity to the structures comes as a consequence of having a life-size lexicon. When a model possesses a realistic amount of knowledge about a language, the model's performance reflects the structure within the language. On the other hand, with a relatively small amount of lexical knowledge, a theorist is forced to give structures special status—either by explicitly representing them in the lexicon (e.g., the BOSS) or by building mechanisms that can derive them (e.g., a parsing mechanism to derive syllables). I have learned a lesson in theory building from building LEX; models of reading should be built to scale with a minimal number of processing mechanisms. Once the model is built, the theorist can count how many phenomena the model can reproduce without additional mechanisms. First determine how many phenomena the model gets for free, and add processing mechanisms only when the model's performance has reached its limit.

References

Adams, M. J. (1990). *Beginning to read : Thinking and learning about print.* Cambridge, MA: MIT Press.

Andrews, S. (1989). Frequency and Neighbourhood effects on lexical access: Activation or search? *Journal of Experimental Psychology: Learning, Memory and Cognition, 15*, 802-815.

Andrews, S. (1992). Frequency and neighborhood effects on lexical access: Lexical similarity or orthographic redundancy? *Journal of Experimental Psychology: Learning, Memory, and Cognition, 18*, 234-254.

Andrews, S., & Scarratt, D. R. (1998). Rule and analogy mechanisms in reading nonwords: Hough dou peapel rede gnew wirds? *Journal of Experimental Psychology: Human Perception and Performance, 24*, 1052-1086.

Ans, B., Carbonnel, S., Valdois, S. (1998). A connectionist multiple trace model for reading polysyllabic words. *Psychological Review, 105*, 678-723.

Averbach, E. & Coriell, A. S. (1961). Short-term memory in vision. *Bell System Technical Journal, 40*, 309-328.

Balota, D., & Chumbley, J, (1985). The locus of the word frequency effect in the pronunciation task: Lexical access and/or production, *Journal of Memory and Language, 24*, 89-106.

Baluch, B. & Besner, D. (1991). Visual word recognition: Evidence for strategic control of lexical and nonlexical routines in oral reading. *Journal of Experimental Psychology: Learning, Memory and Cognition, 17*, 644-652.

Baron, J. & Strawson, C. (1976). Use of orthographic and word-specific knowledge in reading words aloud. *Journal of Experimental Psychology: Human Perception and Performance, 4*, 207-214.

Beauvois, M. F., & Derouesne, J. (1979). Phonological alexia: Three dissociations. *Journal of Neuology, Neurosurgery, and Psychiatry, 42*, 1115-1124.

Besner, D., Twilley, L., McCann, R., & Seergobin, K. (1990). On the connection between connectionism and data: Are a few words necessary? *Psychological Review, 97*, 432-446.

Bodner, G. E., & Masson, M. E. J. (1997). Masked repetition priming of words and nonwords: Evidence for a nonlexical basis for priming. *Journal of Memory and Language, 37*, 268-293.

Coltheart, M. (1978). Lexical access in simple reading tasks. In G. Underwood (Ed.), *Strategies of Information Processing* (pp. 151 - 216). New York: Academic Press.

Coltheart, M., Davelaar, E., Johnasson, J. T., Besner, D. (1977). Access to the internal lexicon. In S. Dornic (Ed.) *Attention and performance VI.* (pp. 535-555). Hillsdale, NJ: Erlbaum.

Coltheart, M. Curtis, B., Atkins, P., & Haller, M. (1993). Models of reading aloud: Dual-route and parallel-distributed-processing approaches. *Psychological Review, 100*, 589-608.

Coltheart, M. & Rastle, K. (1994). Serial processing in reading aloud: Evidence for dual-route models of reading. *Journal of Experimental Psychology: Human Perception and Performance, 20*, 1197-1211.

Feldman-Stewart, D. (1992). *A hybrid connectionist and symbol-processing model for letter identification.* Unpublished Doctoral Thesis, Queen's Univerisity at Kingston.

Feldman-Stewart, D., & Mewhort, D. J. K. (1994). Learning in small connectionist networks does not generalize to large networks. *Psychological Research Psychologische Forschung, 56*, 99-103.

Forster, K. I., Davis, C. (1984). Repetition priming and frequency attenuation in lexical access. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 20*, 844-865.

Forster, K. I., & Shen, D. (1996) No enemies in the neighborhood: Absence of inhibitory neighborhood effects in lexical decision and semantic categorization, *Journal of Experimental Psychology: Learning, Memory, and Cognition, 22*, 696-713.

Funnell, G. (1983). Phonological processes in reading: New evidence from acquired dylexia. *British Journal of Psychology, 74*, 159-180.

Gibson, J. J. (1966). *The senses considered as perceptual systems.* Boston: Houghton Mifflin.

Heathcote, A. (1988). Screen control and timing routines for the IBM microcomputer family using a high-level language. *Behavior Research Methods, Instruments, and Computers, 20,* 289-297.

Henderson, L. (1982). *Orthography and word recognition in reading.* New York: Academic Press.

Hintzman, D. L. (1984). MINERVA 2: A simulation model of human memory. *Behavior Research Methods, Instruments, & Computers, 16,* 96-101.

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

Jared, D. & Seidenberg, M. S., (1990). Naming multisyllabic words. *Journal of Experimental Psychology: Human Perception and Performance, 16,* 92-105.

Johnson, N. F., & Pugh, K. R. (1994). A cohort model of word identification. *Cognitive Psychology, 26,* 240-346.

Kucera, H. & Francis, W. N. (1967). *A computational analysis of present-day American English.* Brown University Press: Providence, RI.

Kintsch, W. (1998). *Comprehension : A paradigm for cognition.* New York: Cambridge University Press.

Kwantes, P. J. & Mewhort, D. J. K. (1999). Evidence for sequential processing in visual word identification. *Journal of Experimental Psychology: Human Perception and Performance, 25,* 376-381

Landauer, T. & Streeter, L. (1973). Structural differences between common and rare words: Failure of equivalence assumptions for theories of word recognition. *Journal of Verbal Learning and Verbal Behavior, 12,* 119-131.

Lima, S. D. (1987). Morphological analysis in sentence reading. *Journal of Memory and Language, 26,* 84-99

Lima, S. D. & Inhoff, A. W. (1985). Lexical access during eye fixations in reading: Effects of word-initial letter sequence. *Journal of Experimental Psychology: Human Perception and Performance, 11,* 272-285.

Lupker, S. J., Brown, P., & Columbo, L. (1997). Strategic control in a naming task: Changing routes or changing deadlines? *Journal of Experimental Psychology: Learning, Memory, and Cognition, 23,* 570-590.

Link, S. W. (1975). The relative judgment theory of two choice response time. *Journal of Mathematical Psychology, 12*, 319-333.

Link, S. W. (1991). *The wave theory of difference an similarity*. Hillsdale, NJ: Erlbaum.

Link, S. W. & Heath, R. A. (1975). A sequential theory of psychological discrimination. *Psychomtrica, 40*, 77-105.

Logan, G. D. (1988). Toward an instance theory of automaticity. *Psychological Review, 95*, 492-527.

Lukatela, G. & Turvey, M. T. (1994a).Visual access is initially phonological: 2. Evidence from associative priming by words, homophones, and pseudohomophones. *Journal of Experimental Psychology: General, 123*, 107-128.

Lukatela, G. & Turvey, M. T. (1994b).Visual access is initially phonological: 2. Evidence from associative priming by words, homophones, and pseudohomophones. *Journal of Experimental Psychology: General, 123*, 331-353.

Marcel, A. J. (1980) Surface dyslexia and beginning reading: A revised hypothesis of pronunciation from print and its impairments. In M. Coltheart, K. Patterson, and J. C. Marshall (Eds.) *Deep dyslexia*. London: Routledge & Kegan Paul.

Marchetti, F. M. & Mewhort, D. J. K. (1986). On the word-superiority effect. *Psychological Research, 48*, 23-35.

Marmurek, H. H. C., & Kwantes, P. J. (1996). Reading words and wirds: Phonology and lexical access. *The Quarterly Journal of Experimental Psychology, 49A*, 696-714.

Marshall, J. C., & Newcombe, F. (1973). Patterns of paralexia: A psycholinguistic approach. *Journal of Psycholinguistic Research, 2*, 175-200.

Marslen-Wilson, W. (1984). Function and process in spoken word identification: A tutorial review. In H. Bouma & D.G. Bouwhuis (Eds.) *Attention and performance X: Control of Language Processes* (pp. 125-150). Hillsdale, NJ: Erlbaum.

McClelland, J. L., & Rumelhart, D. R. (1981). An interactive activation model of context effects in letter perception: Part 1. An account of basic findings. *Psychological Review, 88*, 375-407.

Metcalfe Eich, J. (1982). A composite holographic associative recall model. *Psychological Review, 89*, 627-661.

Mewhort, D. J. K. (1974). Accuracy and order of report in tachistoscopic identification. *Canadian Journal of Psychology, 29*, 348-359.

Mewhort, D.J.K., & Beal, A. L. (1977). Mechanisms of word identification. *Journal of Experimental Psychology: Human Perception and Performance, 3*, 629-640.

Mewhort, D. J. K., & Campbell, A. J. (1981). Toward a model of skilled reading: An analysis of performance in tachistoscopic tasks. In G. E. McKinnon & G. T. Waller (Eds.), *Reading research: Advances in theory and practice Vol. 3* (pp. 39-118). New York : Academic Press.

Mewhort, D. J. K., Campbell, A. J., Marchetti, F. M., & Campbell, J. I. D. (1981). Identification, localization, and "iconic memory": An evaluation of the bar-probe task. *Memory and Cognition, 9*, 50-67.

Mewhort, D. J. K. & Cornett, S. (1972). Scanning and the familiarity effect in tachistoscopic recognition. *Canadian Journal of Psychology, 26*, 181-189.

Mewhort, D. J. K. & Johns, E. E. (1988). Some tests of the interactive-activation model for word identification. *Psychological Research, 50*, 135-147.

Mewhort, D. J. K., Kwantes, P. J., & Feldman-Stewart, D. (1999). A model for letter recognition and lexical access. In R. Heath & A. Heathcote (Eds.), *Proceedings of the Fourth Australasian Cognitive Science Society, Newcastle, NSW.* [CDROM: ISBN: 0725910593]

Monsell, S., Doyle, M. C., Haggard, P. N. (1989). *Effects of frequency on visual word recognition tasks: Where are they? Journal of Experimental Psychology: General, 118*, 43-71.

Monsell, S., Patterson, K. E., Graham, A., Hughes, C. H., & Milroy, R. (1992). Lexical and sublexical translation of spelling to sound: Strategic anticipation of lexical status. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 18*, 452-467.

Mozer, M. C. (1987). Early parallel processing in reading: A connectionist approach. In M. Coltheart (Ed.), *Attention & Performance XII The psychology of reading* (pp. 83-104). Hillsdale, NJ: Erlbaum.

Mozer, M. C. (1991). *The perception of multiple objects: A connectionist approach.* Cambridge, MA: MIT.

Murdock, B. B. (1982). A theory for the storage and retrieval of item and associative information. *Psychological Review, 89,* 609-626.

Patterson, K. E. (1982). The relation between reading and phonological coding: Further neuropsychological observations. In A. W. Ellis (Ed.), *Normality and pathology in cognitive functions.* London: Academic Press.

Plaut, D. C., McClelland, J. L., Seidenberg, M. S., Patterson, K. (1996). Understanding normal and impaired word reading: Computational principles in quasi-regular domains. *Psychological Review, 103,* 56-115.

Pollatsek, A., Lesch, M., Morris, R. K., & Rayner, K. (1992). Phonological codes are used in integrating information across saccades in word identification and reading. *Journal of Experimental Psychology: Human Perception and Performance, 18,* 148-162.

Radeau, M., Mousty, P., & Bertelson, P. (1989). The effect of uniqueness point on spoken-word recognition. *Psychological Research, 51,* 123-128.

Radeau, M., Morais, J., Mousty, P., Saerens, M., & Bertelson, P. (1992). A listener's investigation of printed word processing. *Journal of Experimental Psychology: Human Perception and Performance, 18,* 861-871.

Ratcliff, R. (1978). A theory of memory retrieval. *Psychological Review, 85,* 59-108.

Rayner, K., McConkie, G. W., & Zola, D. (1980). Integrating information across eye movements. *Cognitive Psychology, 12,* 206-226.

Rayner, K., McConkie, G. W., & Erlich, S. (1978). Eye movements and integrating information across fixations. *Journal of Experimental Psychology: Human Perception and Performance, 4,* 529-543.

Rayner, K., & Morris, R. K., (1992). Eye movement control in reading: Evidence against semantic preprocessing. *Journal of Experimental Psychology: Human Perception and Performance. 18,* 163-172.

Rayner, K., & Pollatsek, A. (1989). *The psychology of reading.* Erlbaum: Hillsdale, NJ.

Reicher, G. M. (1969). Perceptual recognition as a function of meaningfulness of stimulus. *Journal of Experimental Psychology, 81,* 274-280.

Rumelhart, D. E., Hinton, G., & Williams, R. (1986). Learning internal representations by error propogation. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition.* (Vol. 2, pp. 216-271). Cambridge MA: MIT Press.

Rumelhart, D. E., & McClelland, J. L. (1982). An interactive activaiton model of context effects in letter perception: Part 2. The contextual enhancement effect and some tests and extensions of the model. *Psychological Review, 96,* 523-568.

Sears, C. R., Hino, Y., & Lupker, S. J. (1995). Neighborhood size and neighborhood frequency effects in word recognition. *Jounal of Experimental Psychology: Human Perception and Performance, 21,* 876-900.

Seidenberg, M. S. & McClelland, J. L. (1989). A distributed, developmental model of word recognition and naming. *Psychological Review,* ***96,*** 523-568.

Seidenberg , M. S. & McClelland, J. L. (1990). More words but still no lexicon: Reply to Besner et al. (1990). *Psychological Review, 97,* 523-568.

Seidenberg, M. S., Plaut, D. C., Petersen, A. S., McClelland, J. L., & McCrae, K. (1994). Nonword pronunciation and models of word recognition. *Journal of Experimental Psychology: Human Perception and Performance, 20,* 1177-1196.

Seidenberg, M. S., Waters, G. S., Barnes, M. A., & Tannenhaus, M. K. (1984). When does irregular spelling or pronunciation influence word recognition. *Journal of Verbal Learning and Verbal Behavior, 23,* 383-404.

Shallice, T. & Warrington, E. K., (1980). Single and multiple component central dyslexic syndromes. In M. Coltheart, K. Patterson, and J. C. Marshall (Eds.) *Deep dyslexia.* London: Routledge & Kegan Paul.

Spoehr, K. T., & Smith, E. E. (1973). The role of syllables in perceptual processing. *Cognitive Psychology, 5,* 71-89.

Taft, M. (1979). Lexical access via an orthographic code: The Basic Orthographic Syllable Structure (BOSS). *Journal of Verbal Learning and Verbal Behavior, 18,* 289-297.

Taft, M. (1986). Lexical access codes in visual and auditory word recognition. *Language and Cognitive Processes, 1,* 297-308.

Taft, M. & Forster, K. I. (1976). Lexical storage and retrieval of polymorphemic and polysyllabic words. *Journal of Verbal Learning and Verbal Behavior, 15,* 607-620.

Taraban, R., & McClelland, J. L. (1987). Conspiracy effects in word recognition. *Journal of Memory and Language, 26,* 608-631.

Tramer, O., Butler, B. E., & Mewhort, D. J. K. (1985). Evidence for scanning with unilateral visual presentation of letters. *Brain and Language, 25,* 1-18.

Treiman, R., Mullenix, J., Bijeljac-Babic, R., & Richmond-Welty, E. D. (1995). The special role of rimes in the description, use, and acquistition of English orthography. *Jounal of Experimental Psychology: General, 124,* 107-136.

Vickers, D. (1980). Discrimination. In A.T. Welford (Ed.) *Reaction times* (pp. 25-72) New York: Academic Press.

Wickelgren, W. A. (1969). Context-sensitive coding, associative memory, and serial order in (speech) behavior. *Psychological Review, 76,* 1-15.

Weide, R. (1995). *The Carnegie-Mellon Pronunciation Dictionary.* Pittsburg: Carnegie-Mellon University.

Appendix A: Table of Phonemes Used by LEX

The phoneme symbols in the appendix are as they appear in the Carnegie-Melon Pronunciation dictionary. The list was compiled by Jerry Quinn at Bell Northern Research.

Phoneme	*Example*	*Translation*
AA	odd	AA D
AE	at	AE T
AH	hut	HH AH T
AO	ought	AO T
AW	cow	K AW
AY	hide	HH AY D
B	be	B IY
CH	cheese	CH IY Z
D	dee	D IY
DH	thee	DH IY
EH	Ed	EH D
ER	hurt	HH ER T
EY	ate	EY T
F	fee	F IY
G	green	G R IY N
HH	he	HH IY
IH	it	IH T
IY	eat	IY T
JH	gee	JH IY
K	key	K IY
L	lee	L IY
M	me	M IY
N	knee	N IY
NG	ping	P IH NG
OW	oat	OW T
OY	toy	T OY
P	pea	P IY
R	read	R IY D
S	sea	S IY
SH	she	SH IY
T	tea	T IY
TH	theta	TH EY T AH
UH	hood	HH UH D
UW	two	T UW
V	vee	V IY
W	we	W IY
Y	yield	Y IY L D
Z	zee	Z IY
ZH	seizure	S IY ZH ER