On the Development of Computational Models of the English Lexicon

Kevin Stanley O'Mara

A Thesis
in
The Department
of
Computer Science

Presented in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy at
Concordia University
Montreal, Quebec, Canada.

1991

ISBN 0-315-80988-4

ABSTRACT

On The Development of Computational Models of the English Lexicon

Kevin Stanley O'Mara, Ph. D.
Concordia University, 1991

The study of the form, acquisition, use and meaning of language has been a seemingly perpetual focal point of scholarly activity. Not surprisingly, language design and use have been viewed as pivotal issues to both practical and theoretical computer science.

The research results reported in this thesis address the issues of syntactic form and lexical structure in natural language processing. In particular the lexicon defined in the Oxford Paperback Dictionary was analyzed in order to discover whatever basic principles or rules underlie the structure of the English words it lists.

The principle, *a priori*, assumption underlying this thesis is that a physical symbol system underlies the structure of the lexicon. An exhaustive analysis of the words defined in the Oxford Paperback Dictionary supports the view that the English lexicon conforms to a simple physical symbol system.

This research found in particular that there are fundamental patterns underlying the English language word structure at the orthographic level. A classification and clustering scheme referred to as Vowel Normal Form (VNF) is sufficient for approximating the basic linguistic structures found in English.

A simple prefix code underlies the relationship between the major word structures of various sizes found throughout the English language lexicon listed in the Oxford Paperback Dictionary. The prefix code structure of English language word structure assures band-filtering effects which may be exploited by simple pattern recognition routines.

A single two parameter model is sufficient to predict the size of the major VNF word group structures found in the Oxford Paperback Dictionary. The prefix code structure model when coupled with the two parameter set-size model predicts both the structure and size of the major VNF frames found in the lexicon.

A form of directed graph, referred to as a WORD-WEB, is sufficient to represent all words of a given VNF set.

The frequency of occurrence of words may be computed as the product of word-length and position-dependent letter-frequencies for the most frequently occurring smaller words. Context-sensitive rule base schema are sufficient to reduce longer words, such as 10-letter-long words to their root words or base component.

ACKNOWLEDGEMENTS

Having just finished the chore of formatting the last pages of Chapter eleven I am left with the happy task of thanking all of those people and institutions who have helped me make this thesis a reality.

I was very luckily introduced to the area of computational linguistics by Professor C. Y. Suen, who succeeded in instilling in me a consuming interest in this area of artificial intelligence. It was through this early interest in computational linguistics that I was invited to attend the NATO Advanced Studies Institute on Pattern Recognition at Oxford in 1981. It was at Oxford that I met the late Professor K. S. Fu, who was later kind enough to review, criticize and improve the material presented in Chapter 9 of this thesis. It was while staying at Saint Anne's College that I first met Dr. Robert Burchfield who is the Editor-in-chief of the Oxford Dictionaries. After a series of meetings, Dr. Burchfield graciously granted me permission to use, for academic purposes, a computer tape of the Oxford Paperback Dictionary. This dictionary was invaluable to my work in that it was the largest and most comprehensive computer readable dictionary in existence. I doubt that I would have continued my work on the lexical structure of English for my doctoral dissertation without the positive comments on my first publications in this area by Dr. Noam Chomsky. For his early encouragement of this work I am very grateful. My doctoral research was supported from 1981 to 1985 by a Concordia University Fellowship, an NSERC Fellowship and various research grants. I am very thankful of the financial and academic support that my university and my country has afforded me. Last year I was very fortunate in being awarded a paid leave to work on my dissertation by my employer, Winona State University. I am very thankful of the support of the administration and faculty at Winona. In this regard I am particularly indebted to Dr. Douglas Sweetland, the Vice-President for Academic Affairs and Dr. Denis Nielsen, the Dean of Science & Engineering at Winona State University for allowing me the time and well-being needed to complete this dissertation.

The joy of discovery is a truly exhilarating reward for the hard work and many false starts that typically lie on a scientist's path to

new knowledge. I have been blessed with a milieux at Concordia that does not sacrifice quality in either the form or the content of its research. In this regard I will always be indebted to Professors Terrill Fancott, Wojciech Jaworski, Rajjan Shinghal, T. Radhakrishnan and the late Kin Vinh Leung for an excellent academic apprenticeship.

In work that has spanned the better part of a decade it is difficult to properly acknowledge all of the helpful discussions, criticisms and help that have lead to what is today a thesis. I would be remiss however if I did not take this opportunity to thank Dr. Carol Anderson, Deganit Armon, Laszlo Becskei, Glen Begrow, Brian Brownlow, Mike Ducharme, Kent Farrell, Libero Ficocelli, William Gillespie, Sadegh Ghaderpannah, Dr. Roger Guy, Don Harris, Dr. Syed Hyder, Drs. Ted & Cathy Knous, Richard & Charles LePage, Dr. Alister McLeod, Dr. Franz Oppacher, Beth Peak, Henry Polley, Dr. Don Scheid, Dr. Don Steward, Chris Struck and Mark Weisser for their support and help.

Once the research work is done, the task of writing, formatting and editing a dissertation looms as an arduous, demanding and painstaking burden. The rapid technological advances in desktop publishing means that one comes to expect very professional graphs and images in a thesis. Of course these wonderful images also require a huge expenditure of time to prepare. I am indebted to Laszlo Becskei, Glen Begrow for their advice, support and help in typing and editing the text of this dissertation and to Mike Ducharme and Richard LePage for their assistance in formatting the many figures and tables embedded in this thesis.

My supervisor, Dr. Terrill Fancott, has always extended to me an immense amount of advice, support and good will. I have profited immensely from his guidance and can only hope to be as good a supervisor to my own students. I also hope that I have learned from Professor Fancott how to adopt his positive outlook towards research in particular and life in general.

Finally I would like to thank my father, Stanley O'Mara, for having the patience of Jove with a son who seems to spend forever working on a thesis that seemed to have no end.

DEDICATION

I would like to dedicate this thesis to the memory of my mother Veronica Rose O'Mara (nee Williams).

Table of Contents **Page**

CHAPTER 4 : WORD-LEVEL SYNTACTIC STRUCTURE

CHAPTER 5: A PREFIX CODE MODEL OF ENGLISH LANGUAGE WORD STRUCTURE

CHAPTER 6: PREDICTING THE SIZE OF THE DOMINANT VNF SETS IN ENGLISH

CHAPTER 7 : PREDICTING THE SIZE AND LOCATION OF DOMINANT VNF SETS IN ENGLISH

CHAPTER 8: WORD WEBS

CHAPTER 9 : MODELS OF WORD- AND LETTER - FREQUENCY USE

CHAPTER 10 : SYNTACTIC STRUCTURES AND WORD-LEVEL GRAMMARS IN ENGLISH

CHAPTER 11: CONCLUSIONS

LIST OF FIGURES

CHAPTER 5: A PREFIX CODE MODEL OF ENGLISH LANGUAGE WORD STRUCTURE

CHAPTER 6: PREDICTING THE SIZE OF THE DOMINANT VNF SETS IN ENGLISH

CHAPTER 7 : PREDICTING THE SIZE AND LOCATION OF DOMINANT VNF SETS IN ENGLISH

CHAPTER 8: WORD WEBS

CHAPTER 9 : MODELS OF WORD- AND LETTER - FREQUENCY USE

CHAPTER 10: SYNTACTIC STRUCTURES AND WORD-LEVEL GRAMMARS IN ENGLISH

LIST OF TABLES

CHAPTER 10 : SYNTACTIC STRUCTURES AND WORD-LEVEL GRAMMARS IN ENGLISH

CHAPTER ONE
THE SEARCH FOR STRUCTURE IN INTELLIGENCE AND LANGUAGE

1.1 INTRODUCTION

What are the agencies of mind and memory? Questions such as these have preoccupied philosophers since the dawn of time. It is only since the beginning of the scientific revolution that such questions have been posed by empiricists. For over five hundred years physicists, biologists and psychologists have sought to determine the physical concomitants of thought [1.1, 1.2]. Unfortunately, within the paradigms of science, no convincing answers have been found to such questions as: how do we know? what can we know? how do we learn? and how do we forget? [1.3, 1.4]. It is only within the last thirty years that such questions have been posed by a group of scientists who have at their disposal an ever-increasing ability to analyze and simulate the macroscopic behavior of complex, hierarchical concomitant physical systems [1.3].

Today many computer scientists, particularly those working in the areas of cognitive science and artificial intelligence, seek to model, simulate and evaluate models of problem solving, insight, emotion, thought, and reason [1.4]. The search for a basic computational metaphor of mind is, of course, not without opponents [1.5, 1.6]. Curiously, this opposition comes most from those who are displeased, frightened and appalled by attempts at applying naive determinism to the 'miracle of mind'. It seems likely that this century's classic mind-brain dichotomy [1.7] will be slowly resolved over the next decades in computer science laboratories around the world [1.4]. This great inquiry may lead us to a fundamental understanding of intelligence and even new embodiments of mind.

Language appears fundamental to our study of mind [1.8]. In fact, Newell & Simon [1.9], who coined the physical-symbol system hypothesis, claim that a necessary and sufficient condition for intelligence is the ability to manipulate a symbol system abstracted from reality. For example, consider the physical-symbol system

underlying competitive game-playing scenarios such as chess. Each chess piece has a well defined and specific repertoire of possible moves. These moves are context free and may be applied anywhere within the bounds established by the chessboard. Of course a chess-piece cannot legally attempt to move outside of the chessboard perimeter. The game of chess proceeds through a set of opening moves that establish the context of the contest to arrive at a typical phase referred to as the "mid-game". While the game's opening moves are critical to the context of the game, they have little effect on the strategies used for playing the next phase of the game. Once the opening moves have established the context of the game, a classic min-max algorithm may be employed to compute consecutive moves [1.16]. The min-max algorithm uses an objective function to always choose the move that most disadvantages your opponent while best advantaging you. The min-max algorithm may be used recursively to any depth needed to assure a 'win'. A win in such scenarios is often the result of the success of a process of delayed gratification. Your move and the subsequent move your opponent is referred to as a 'ply', The depth of recursion used in the evaluation of a move is referred to as the number of plies used in the simulation. The optimal number of plies required to win the game is, of course, not easy to determine; in fact, it may vary as the game proceeds. If the human excels at competitive game- playing, using something like a min-max algorithm, this is most likely because of our ability to adaptively choose the depth of recursion used in our game-playing. One scenario for chess playing has both players using a min-max algorithm until relatively few pieces remain on the board and an overwhelming advantage is obvious. This state signals the start of the final phase of the game. The "endgame", or final phase of the game, typically uses look-up tables to invoke previously compiled, context-sensitive moves to quickly finish the game [1.16]. A typical physical-symbol system for competitive game-playing incorporates an objective function, a min-max algorithm and a fixed set of symbolic pieces each of which have archetypical behavior.

The concept that language, which is a classic physical-symbol system, both underlies and limits human thought is often traced to a

seminal work by Benjamin Whorf entitled Language, Thought, and Reality [1.10]. Early researchers in computer science came quickly to appreciate the validity and practical importance of Whorf's work. They found that the native constructs established in the design of a computer language often limited both its utility and its domain of applicability [1.11].

Classicists such as Polya [1.12] have demonstrated that the way in which a problem is posed, or rather abstracted, can radically affect its solution. Piaget [1.13] is the first to carefully observe the maturational stages of such skills. Research in these fields established the complex and fascinating inter-relation of language and computation [1.14]. The differential growth of vocabulary in response to our experience and ability to solve problems in a given domain is an example of an object-oriented approach to problem solving. The development of formal abstract symbol manipulators, such as mathematics by humans, has often been hailed as one of mankind's greatest intellectual achievements [1.15]. Mathematics is ,of course, a language whose power and applicability may paradoxically rest with its unambiguous and relatively simple, context-free grammar. This realization and the *idee fix* of the AI community of the early 1980's led to the development of systems suitable for the elegant solution of problems drawn from the domain of calculus. While immensely practical, these systems, which were composed of less than a few hundred rules from the domain of number theory, shed very little light on intelligence; although that was the prime, *a priori*, assumption underlying researchers' preoccupation with the study of mathematical reasoning.

A previous generation of research in AI has resulted in few, hard won, conclusions. Perhaps foremost among these is the observation that any significant computational intelligence has, to date, required both copious data which is often referred to as a knowledge base and a set of algorithms suitable for inference, deduction and learning [1.16]. The process of coding has played two principle roles in the development of these systems. The first role is typically that of encryption [1.17]. Encryption has two distinct roles: to increase or to decrease the thought needed to grasp a message. In its first role,

theorems such as the Shannon-Fano codes are used to enhance the privacy of a message [1.18]. Their encryption maximizes the computation needed to extract the message from the code.. Encryption's second role is exactly opposed to its first use. Encryption may be used as Northrop Frye [1.19] points out 'in the spirit of Shakespeare', to ensure the maximum 'modality' of a message. In this light authors such as Shakespeare and Freud intuitively seek the explanation of complex ideas in simple terms. Others such as Burgess [1.20] illustrate the opposite of this idea.

One of the distinct goals of twentieth century science has been to formalize operational models of mind, language, computation and coding[1.21] . In fact, it was not until this century that the concept and power of operational models was exploited [1.22].

The turn of this century marked a rebirth of simplistic gestalt paradigms in mathematics and science. It was this period that lead to the ambitious pursuits by such giants as Russell & Whitehead [1.23] and Sir James Jeans [1.24]. It was during this period that S. Freud [1.25], C. J. Jung [1.26], W. Penfield [1.27] and K. Lashley [1.28] were establishing the first gestalt operational models of mind. Determinist schools of mind, such as those established by McCulloch & Pitts [1.29], Sherrington [1.30], and D. O. Hebb [1.31], provided the first operational models of neural activity. It was these operational models of cell assemblies and neural nets that investigators such as Papert & Minsky [1.32] exploited in their early work on artificial intelligence. It was not until much later that comprehensive hybrid models which attempted to encompass both the gestalt and local agencies of the mind were posited by Minsky's school [1.4].

In the first half of this century, the young British mathematician, Allan Turing, turned his academic interest to the study of computation, encryption and language [1.33]. Turing is cited as providing the first operational definition of artificial intelligence [1.16, 1.34], and his model of computation remains a benchmark in computer science. It was during the Second World War that Turing devoted his attention to the practical concerns of computational linguistics.

Atwell [1.34] has recently pointed out that the origins of computational linguistics date back to the development of first-generational machines and what can be referred to as modern computer science. Turing led much of the original research in this area, which focused the British war effort on coding and encryption. It is this early work, and that of Shannon and Chomsky, that eventually lead to new statistical theories of languages.

In was not until after the Second World War that Noam Chomsky radically revolutionized linguistics through a series of publications that established the normal forms and hierarchies of language that today bear his name [1.35, 1.36, 1.37]. Chomsky's models of language had great impact on early formalizations of context-free languages in computer science [1.34]. The impact of Chomsky's transformational grammars was immediately appreciated by the small community of computer scientists working in the then new field of artificial intelligence. In fact, it can be argued that the formalism underlying conjunctive normal form within computer science has its basis both in Chomsky's transformational grammars and the early work of Lewis Carroll [1.36], who developed disjunctive normal form in 1896. Transformational grammar revolutionized linguistics by providing the field with a plausible, formal framework for an understanding of the concept of semantic meaning or deep structure of a thought. The various effects which can interfere with our comprehension of deep structure are referred to by computer scientists as "representational distortions" [1.38].

The realization that there existed a fundamental relationship between the Chomsky hierarchy of formal grammars and the functional features of machines that could recognize them was one of the most important fundamental discoveries of computer science [1.34].

The early work in automata theory arose from work on formalisms of neural nets. While automata theory led to functional abstractions of computation machinery, it took the insight of a decade of computer science research to realize that the properties of these models of computational mechanisms could be described by the languages that they could accept [1.39].

It is important to understand the functional equivalence of a language and the automata that can recognize constructs conforming to the language. It is more important to understand the significance of a linguistic statement and to know what criteria must be met to successfully transmit the statement. Such issues remained open until the pioneering work on information theory by Claude Shannon [1.40] in 1948. It is from this basic research on information theory that models of prefix and separable codes were developed.

While this century has seen the development of fundamental operational models of mind, language, computation and communication, researchers are still attempting to integrate our knowledge of these areas into a cohesive whole. Unification of these models has proven to be particularly difficult and not very satisfying. It might be that we are still missing fundamental concepts such as an understanding of memory and its role within these models. Memory management may turn out to be a critical factor in discovery processes which typically involve us in cycles of inference and deduction. While perhaps critical to the future developments in the field of artificial intelligence, this question falls outside the domain of this thesis.

How much of our intellectual ability is genetically endowed? Are important high-order agencies such as the human language center among those genetic endowments, as Chomsky suggests? If our language center is genetically endowed, how can we best determine and simulate its functions?

These questions belong to a class of difficult problems at which human ingenuity excels [1.41]. Such problems often confronted the early pioneers in pattern recognition such as K.S. Fu [1.42]. Fu established, to a large extent, the domain of structural or syntactic pattern recognition and realized the intrinsic limitations of this approach in analyzing an unknown pattern. Statistical pattern recognition could, in theory, be applied to the task of assessing the similarity of patterns. Unfortunately, in order to apply such statistical techniques, it is necessary to specify the important features on which statistical measures could separate and cluster similar patterns. Sokal & Sneath [1.43] developed a field of study known as numerical

taxonomy. A major preoccupation of this field is the determination of just how many features are needed to achieve efficient classification. A second concern is which of the features, of a usually large set of mutually correlated features, should be used in this task. Pattern recognition techniques typically use a min-max procedure to empirically optimize clustering schemes. The interesting question of how we identify 'causal factors' in what is often an immensely long sequence of antecedent events remains open. Of course, there is a school of thought [1.44] that claims that we are in fact actually incapable of this process and can only really abstract, model, and manipulate that which we know through our genetic endowment.

A decade ago in an elegant, empirical study, Adams [1.45] developed a carefully controlled set of experiments intending to disprove a tenet proposed at the turn of the century by Pillsbury [1.46]. Pillsbury had hypothesized that humans read or recognize words, even words they had never seen before, much more easily than they can read nonsense strings. Adams carefully fabricated nonsense strings of specific word-sized units which were composed of letters having the same position dependent probabilities as her set of test words. Surprisingly, her experiments overwhelmingly confirmed Pillsbury's hypothesis. How could it be that humans could quickly and intuitively recognize patterns in English words? English is a difficult language whose object-based vocabulary is drawn as loanwords from many languages. It is partially because of its hybridity that there is an apparent lack of rules to specify its spelling, declension and grammar. English would appear to be a language where anything can, and in fact does, occur. How then can we explain Adams's results? The results presented in this thesis help to explain Adam's results and confirm Pillsbury's hypothesis.

1.2 PHILOSOPHICAL APPROACH & ASSUMPTIONS

This thesis proceeds in the tradition of Immanuel Kant [1.47] with a set of hypotheses which are assumed, *als ob*. First among these, is the implicit assumption that there are rules which determine whether words, such as the German loanword zeitgeist,

are accepted into English usage. A second assumption which is posited on parsimony, is that such rules might be both context-free and operate at a semantic level of understanding that is below that needed to grasp a well-formed phrase or sentence. The third assumption is that the observations of patterns at the lexical level of a language will provide a sufficient basis from which to infer general syntactic rules that apply to the language as a whole. There is additionally the assumption that a computational analysis of accepted English lexicon will shed light on the patterns, rules and mathematics that underlie English [1.48]. The fourth assumption is that observations based on computations of the relative use of word structures will enhance our understanding of the syntactic rules which were derived from a static lexical analysis. A fifth assumption is that the computational method and procedures used to isolate the patterns and rules that apply to English may prove to be valuable to the study of its spoken form. Finally there is hope that this work may extend to the analysis of other languages and perhaps deeper structures.

1.3 THESIS GOAL

The principal goal of this thesis is to establish computational models of the English language lexicon.

1.4 THESIS OUTLINE

Chapter One is an introduction to the intellectual basis of computational linguistics and the thesis topic.

Chapter Two provides an outline of the historical background underlying to the thesis topic.

Chapter Three describes the source materials used in this research.

Chapter Four describes the development of a classification scheme referred to as "Vowel Normal Form". This chapter also presents basic empirical results obtained by using vowel normal form

to cluster words of a given length into common structural templates or syntactic frames.

Chapter Five describes the use of a prefix code model of English language word structure. This model allows one to simulate the dominant structural components of an entire lexicon from a small kernel of archetypes.

Chapter Six outlines the development of a second simple model. This model allows one to predict the set size of the principle syntactic word forms found in the lexicon.

Chapter Seven describes the complementary use of the two models presented in Chapters Five and Six in predicting the size and location of the basic components of the English lexicon.

Chapter Eight describes the use of an augmented transition graph, referred to as a "Word Web", to describe all words conforming to a given vowel normal form.

Chapter Nine describes the use of context-sensitive statistics to predict the frequency of occurrence of words. Word-length and position-dependent letter-frequencies are used to predict the orthographic form of words. These statistics can also be used to infer the prefix and suffix components of longer words.

Chapter Ten describes a context-sensitive, rule-base suitable for reducing large words to their stems or roots by the application of sets of suffix rules that are tuned to the word-length and vowel normal form. Vowel normal form and word length are used in this work as the two principle features underlying the application of a specific context-sensitive rule-base.

Chapter Eleven summarizes the results of this study and outlines the further work that would be needed to generalize its conclusions to the formalized function of a lexicon in the language center of humans and perhaps other highly evolved mammals.

The results presented in this thesis offer fundamental insight into the English language and suggest new approaches to artificial learning and understanding which may release natural language processing systems from the use of tables and primitive data structures. They should allow one to mimic, in artificial intelligence systems, the behavior observed in Adams's experiments on humans.

Practical applications include the use of such systems in intelligent spelling checkers which could access the likelihood and acceptability of unknown words. Another practical application would involve embedding such systems in smart optical scanners. Immense improvements in the performance of these scanners would result from their ability to correct errors. These errors result not only from transliterations which involve letter substitutions but also from the deletion or insertion of letters in a word. This work suggests the potential utility of developing a system for detecting the insertion or deletion of letters within the prefix code structures of English.

The theory underlying this thesis has its origins in pattern recognition [1.52], artificial intelligence [1.53], computational linguistics [1.54, 1.55, 1.56], automata theory [1.57, 1.58], the theory of formal [1.59, 1.60] and natural languages [1.61, 1.62. 1.63, 1.64], including work on information theory, coding and encryption [1.65], fractal geometry [1.66, 1.67], and statistical sampling theory [1.68, 1.69, 1.70, 1.71].

1.5 REFERENCES

[1.1]. K. Clark, Civilization: A Personal View, John Murray, London, England, 1976.

[1.2]. R. Feynman, The Character of Physical Law, MIT Press, Cambridge, Massachusetts, 1965.

[1.3]. S. Grossberg, Neural Networks and Natural Intelligence, The MIT Press, Cambridge, Massachusetts, 1988.

[1.4]. M. Minsky, The Society of Mind, Simon and Schuster, New York, 1986.

[1.5]. J. Searle, Minds, Brains, and Science, Harvard University Press, 1984.

[1.6]. H. Dreyfus, What Computer's Can't Do: A Critique of Artificial Reason, Harper and Row, New York, N. Y., 1972.

[1.7]. J. Eccles and D. N. Robinson, The Wonder of Being Human: Our Brain and Our Mind, New Science Library, Shambhala, Boston, 1985.

[1.8]. N. Chomsky, Language and Mind, [Enlarged Ed.], Harcourt Bracee Jovanovich, Inc., New York, N. Y., 1972

[1.9]. A. Newell, H A. Simon, "Computer Science as Empirical Inquiry: Symbols and Search," CACM, Vol. 19, 3, pp. 113-136; 1976.

[1.10]. B. L. Whorf, Language, Thought, and Reality, ed. by J. B. Carroll, The MIT Press, Cambridge, Massachusetts, 1956.

[1.11]. M. Marcotty and H. Ledgard, Programming Language Landscape: Syntax, Semantics, and Implementation, [2 Ed.], Science Research Associates, INC., Chicago, 1986.

[1.12]. G. Polya, How to Solve It, Princeton University Press, Princeton, New Jersey, 1973.

[1.13]. H. Guber and J. Voneche, eds., The Essential Piaget, Basic Books, New York, N. Y., 1977.

[1.14]. A. M. Turing, "Computing Machinery and Intelligence," Mind, Vol. LIX, 236; 1950. Reprinted in: D. R. Hofstadter and D. C. Dennett, The Mind's I, pp. 53-68, Bantam Books, New York, N.Y., 1981.

[1.15]. J. Bronowski, The Ascent of Man, British Broadcasting Corporation, Sir Joseph Causton & Sons Ltd, London, England, 1976.

[1.16]. E. Rich and J. Knight, Artificial Intelligence, McGraw-Hill Series in Artificial Intelligence, New York, N. Y., 1991.

[1.17]. S. Goldman, Information Theory, Dover Press, New York, N. Y., 1953.

[1.18]. C. E. Shannon and J. McCarthy, eds., Automata Studies, Princeton University Press, Princeton, New Jersey, 1956.

[1.19]. N. Frye, On Culture and Literature, The University of Chicago Press, 1978.

[1.20]. A. Burgess, Earthly Powers, Hutchinson, London,1980

[1.21]. N. Wiener, Cybernetics, John Wiley & Sons, New York, N. Y., 1948.

[1.22]. K. R. Popper, The Logic of Scientific Discovery, Harper & Row, New York, N. Y., 1968.

[1.23]. A. N. Whitehead and B. Russell, Principia Mathematica, Cambridge, Massachusetts, 1927.

[1.24]. J. Jeans, The Universe Around Us, MacMillian Press, New York, N. Y., 1929.

[1.25]. S. Freud, The Basic Writing of Sigmund Freud, The Modern Library Press, New York, N. Y., 1938.

[1.26]. C. G. Jung, The Collected Works of C. G. Jung, Princeton University Press, Princeton, New York, N. Y., 1971.

[1.27]. W. Penfield, The Mystery of the Mind: A Critical Study of the Consciousness and the Human Brain, Princeton University Press, Princeton, New Jersey, 1975.

[1.28]. K. S. Lashley, "The Problem of Cerebral Organization in Vision", Biological Symposia, Vol, 7, pp. 301-22; 1942.

[1.29]. W. McCulloch and W. Pitts, " A Logical Calculus of the Ideas Immanent in Nervous Activity," Bull. Math. Biophysics, Vol. 5, p. 115-137; 1943.

[1.30]. C. S. Sherrington, The Integrative Action of the Nervous System, Cambridge University Press, London, England, 1947.

[1.31]. D. O. Hebb, The Organization of Behavior, John Wiley & Sons, New York, N. Y., 1949.

[1.32]. see 1.11

[1.33]. A. Hodges, Alan Turing: The Enigma, Simon and Schuster, New York, N. Y., 1983.

[1.34]. E. S. Atwell, "Grammatical Analysis of English by Statistical Pattern Recognition," Pattern Recognition: Lecture Notes in Computer Science, J. Kittler, ed., Vol. 301, Springer-Verlag, Berlin, pp. 626-635; 1988.

[1.35]. N. Chomsky, Syntactic Structures, The Hague: Mouton, 1957.

[1.36]. M. Salmon, Introduction to Logic and Critical Thinking, Harcourt Brace Jovanovich, San Diego, California, 1984.

[1.37]. N. Chomsky, Aspects of the Theory of Syntax, The MIT Press, Cambridge, Massachusetts, 1988.

[1.38]. W. M. Jaworski, L. Ficocelli, K. S. O'Mara, "The ABL/W4 Methodology for Systems Modeling," Systems Research, Vol. 4, 1, pp 23-37; 1987.

[1.39]. J. E. Hopcroft and J. D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley , Reading, Massachusetts, 1979.

[1.40]. C. Shannon and W. Weaver, A Mathematical Theory of Communication, University of Illinois Press, Chicago, Illinois, 1949.

[1.41]. R. Arnheim, Visual Thinking, University of California Press, Berkeley, California, 1969.

[1.42]. K. S. Fu, Syntactic Pattern Recognition and Applications, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.

[1.43]. R. Sokal, P Sneath, Numerical Taxonomy: The Principles and Practice of Numerical Classification. W. H. Freeman, San Francisco, 1973.

[1.44]. J. Eccles, ed., Brain and Conscious Experience, Springer-Verlag, New York, N. Y., 1966.

[1.45]. M. A. Adams, " Models of Word Recognition," Cognition Psychology, Vol. 11, pp. 133-176; 1979.

[1.46]. W. B. Pillsbury, " A Study in Apperception," American Journal of Psychology, Vol. 8, pp. 315-393; 1897.

[1.47]. H. Vaihinger, Die Philosophie des Als Ob, trans. by C.K. Ogden,(The Philosophy of 'As If"), Routledge & Paul, London, 1935..

[1.48]. G. Herdan, The Advanced Theory of Language as Choice and Chance, Springer-Verlag, New York, N. Y., 1966.

[1.49]. K. O'Mara, " A Model for Determining the Frequency of Occurrence of English Language Words," Pattern Recognition Theory and Applications, ed. by J. Kittler, K. S. Fu, and L. F. Pau, D. Reidel Inc., Hingham, Massachusetts, 1982.

[1.50]. K. O'Mara, W. Jaworski, and S. Klasa, " On the Development of a Recursive Model of Word Structure in English," Applied Systems and Cybernetics, ed. by G. Lasker, Pergamon Press, New York, N. Y., 1980.

[1.51]. J. Hawkins, comp., The Oxford Paperback Dictionary, Oxford University Press, Oxford, England, 1979.

[1.52]. P. A. Devijver and J. Kittler, Pattern Recognition: A Statistical Approach, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.

[1.53]. M. F. Firebaugh, Artificial Intelligence: A Knowledge-Based Approach, Boyd and Fraser, Boston,1988.

[1.54]. R. F. Simmons, Computations from the English, Prentice-Hall, Englewood Cliffs, New Jersey, 1984.

[1.55]. R. C. Schank and R. P. Abelson, Scripts, Plans, Goals, and Understanding, Lawrence Erlbaum, Hillsdale, New Jersey, 1977.

[1.56]. T. Winograd, Understanding Natural Language, Academic Press, New York, N. Y., 1972.

[1.57]. Z. Kohavi, Switching and Finite Automata Theory, [2 Ed.], McGraw-Hill, Inc., New York, N. Y., 1978.

[1.58]. M. Minsky, Computation: Finite and Infinite Machines, Prentice-Hall, Englewood Cliffs, New Jersey, 1967.

[1.59]. A. Salomaa, Jewels of Formal Language Theory, Computer Science Press, Inc., Rockville, Maryland, 1981.

[1.60]. A. Radford, Transformational Syntax: A Student's Guide to Chomsky's Extended Standard Theory, Cambridge University Press, Cambridge, Massachusetts, 1981.

[1.61]. G. K. Zipf, The Psycho-Biology of Language: An Introduction to Dynamic Philology, The MIT Press, Cambridge Massachusetts, 1968.

[1.62]. G. Herdan, Type-Token Mathematics, Mouton and Company, S-Gravenhage, The Hague, Netherland, 1960.

[1.63]. R. C. Schank, The Cognitive Computer: On Language, Learning, and Artificial Intelligence, Addison-Wesley, Reading, Massachusetts, 1984.

[1.64]. N. Chomsky, Cartesian Linguistics: A Chapter in the History of Rationalist Thought, ed. by N. Chomsky and M. Halle, Harper and Row, New York, 1966.

[1.65]. C. Cherry, Principles of the Statistical Theory of Communication, McGraw-Hill, New York, N. Y., 1963.

[1.66]. B. Mandelbrot, The Fractal Geometry of Nature, W. H. Freeman, San Francisco, California, 1982.

[1.67]. M. Barnsley, Fractals Everywhere, Academic Press, Boston, Massachusetts, 1988.

[1.68]. R. A. Fisher, Statistical Methods and Scientific Inference, Oliver and Boyd, London, England, 1956.

[1.69]. R. Fisher, Design of Experiments, Oliver and Boyd, London, England, 1935.

[1.70]. H. A. David, Order Statistics, [2 Ed.], John Wiley and Sons, Inc., New York, N. Y., 1981.

[1.71]. C. W. Therrien, Decision Estimation and Classification: An Introduction to Pattern Recognition and Related Topics, John Wiley and Sons, Inc., New York, N. Y., 1989.

[1.72]. K. O'Mara, "Syntactic Structures and Word Level Grammars in English," In proceedings of the International Institute for Advanced Studies in Systems Research and Cybernetics, Baden-Baden, Germany, 1989.

[1.73]. K. O'Mara, T. Fancott, "The Morphological Analysis and Computer Modelling of the Fundamental Word Forms Found in English," Manuscript in preparation.

[1.74]. K. O'Mara, T. Fancott, "A Syntactic Model of English Language Word Structure at the Lexical Level," Manuscript in preparation.

CHAPTER TWO
HISTORICAL PERSPECTIVE

2.1 INTRODUCTION TO THEORY AND METHODOLOGY

The development of computational models of language is one of the central areas in artificial intelligence. This chapter develops the background of my thesis work with a broad perspective of relevant concepts and paradigms in artificial intelligence.

The theoretical foundations of this thesis are based on a broad range of related, but distinct, domains which have played central roles in the evolution of computer science during this century.

The history of science demonstrates that deeper understanding usually follows the careful investigation of empirical rules which are generalizations accrued from human experience and from intuitions that have demonstrated practical importance [2.1].

The evolution of scientific understanding has, at least in observable systems, lead to the development of competing operational models which can be evaluated and verified by experiments and simulations [2.2]. A fundamental cycle of inductive and deductive inference is basic to our present functional definition of science [2.1]. It is exactly this paradigm of enquiry that underlies the success and growth of modern science and our understanding of the physical universe. Of course, in order to undertake such enquiries, the scientist must account for the peculiarities of the domain of enquiry [2.3]. Thus, while the scientific process is similar across all domains, the study of medicine, physics and linguistics share so very little in common with each other that it is difficult to see a common thread in their respective scientific methodologies.

This chapter will outline some of the theory underlying the lexical and grammatical components of computer science that is essential to this thesis. In this review some very germane problems and results in the areas of pattern recognition are also discussed.

2.2 LEXICAL THEORY

In 1935, G. K. Zipf published a text [2.4] entitled, The Psycho-Biology of Language, in which he described his observations on the relative frequency of word usage in natural language texts. Zipf found that there existed an inverse relationship between the frequency of use of a word and its relative rank. Thus, if there were a thousand different words found in a text, then the most frequently occurring word among them was used one thousand times. Empirical results such as those depicted in Figure 2.1 led Zipf to his conclusion that in general the p -th most common word in a natural language text occurs with a frequency which is approximately inversely proportional to p.

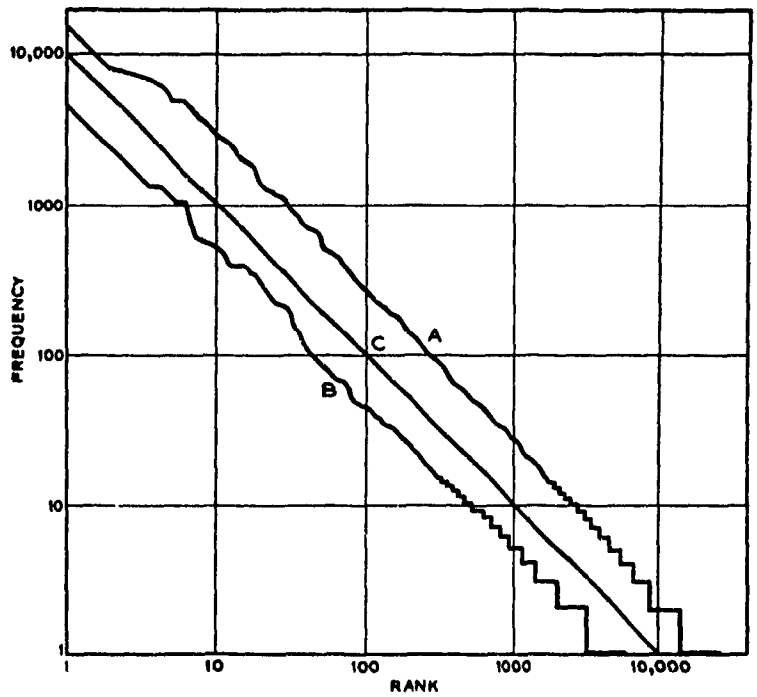

Figure 2.1 Zipf's Law. Abscissa shows frequency (number of times a word is used) plotted against rank (order of commonness) for 260,430 running words of James Joyce's Ulysses (curve A) and for 43,989 words from newspapers (curve B). The straight line C illustrates Zipf's idealized curve or 'law'. Reproduced from 2.44

There was an evolution of thought from the early work of Zipf, principally through the critiques of Yule [2.5], Van Herdan [2.6], and the thesis work of Mandelbrot [2.7] in 1952. This evolution saw an attempt to develop a sound and functionally meaningful lexical model of natural-language use from Zipf's work. Much of the work described in Chapter 6 of this thesis was undertaken within this milieu.

Zipf was obsessed with confirming that the first empirical results he obtained were in fact truly characteristic of most natural languages. It was this quest that lead to work by many other authors [2.8, 2.9] who attempted to develop and use appropriate sampling techniques for statistically meaningful computational analysis of texts. Some authors argued that Zipf's descriptive models suffered from severe theoretical limitations. Most of the objections to Zipf's equations dealt with their inappropriateness in modeling the behavior of both the most frequently occurring vocabulary elements as well as the least frequently used terms. Various alternative models were proposed to modify the behavior at its extrema of the law proposed by Zipf. Other objections to Zipf's work [2.10] focused on the inappropriate nature of his equation's convergence characteristics, which theoretically limited the size of the lexicon. Other theoreticians disliked Zipf's law because it, like Miller's "magic number 7" [2.11], were based on black box observations that appeared to describe the operating conditions of a complex system but really failed to offer any significant understanding of the system itself [2.12]. Zipf's law describes macroscopic behaviour without providing insight into any mechanism underlying these macroscopic effects.

For instance Zipf's law may be a simple concomitant of the log-normal statistical distribution observed for natural-language word usage [2.6]. On the other hand, Zipf's law may result from the internal workings of the memory-management scheme for lexical

processing in the human's language center [2.13]. In any case, Zipf's law, which as it turns out [2.14] was first noted by Estoup [2.15], remains a very interesting and useful observation which may someday fit within a comprehensive theory of language processing.

Knuth [2.16] has pointed out that Zipf's law and other probability effects [2.17] are extremely useful in the design, analysis and modeling of a wide range of text-based computer applications working at the lexical level.

A great many applications of these still-empirical observations arise in systems for both natural and computer language processing. For instance, Chapters 6 & 7 demonstrate the utility of such probability effects in predicting the macroscopic characteristics of vocabulary dispersal throughout the predicted lexical structures described in Chapter 5.

In natural language processing, one usually wants to help determine a *lexical signature* in order to help verify authorship of a disputed scholarly work [2.18]. Other interesting applications include the use of pattern recognition schemes for key-word matches [2.19] and concordance studies [2.20, 2.21] that compute partial correlation coefficients on vocabulary associations. These lexical methods have been applied to artificial intelligence applications in verbal reasoning studies [2.22, 2.23], as well as in the field of clinical psychiatry [2.24]. It is expected, in fact, that such approaches will have a major impact on our present attempts [2.25, 2.26] to unravel the molecular code of the human genome.

In the case of machine languages, lexical effects form the basis of metrics that have been proposed as a way of assessing software quality [2.27, 2.28] and the likelihood of the presence of errors in the software [2.29, 2.30].

A second important practical use of lexicon-based equations is their ability to predict the length of a source text solely from the size of its lexicon. Chapter 6 presents work which implies that such equations may be used to estimate the size of the vocabulary conforming to the dominant lexical structures found in the language.

Within the domain of computer science, lexical equations such as those proposed by Halstead [2.31] and O'Mara et al [2.32] have

been used to predict the length of source code from the details of its lexicon. This is important in that present industry standards [2.33] require that this lexicon be established in the software requirement specification of a system. Hence, these metrics may be computed long before coding is undertaken.

Shortly before his death in 1977, Halstead published a synopsis of his work in this area [2.34] in the text Software Science. Since then, Halstead's work has suffered a fate similar to that encountered by Zipf. After over twenty years of investigation, controversy still surrounds the implications and value of Halstead's observations. Perhaps the best known and most controversial of these observations is Halstead's length equation [2.35, 2.36, 2.37]. This formula (which is given in Equation 2.1) predicts the length of the source code, N, as a function of the number of distinct operators, n_1 , and operands, n_2 , needed to specify the program. Some authors [2.38, 2.39] doubt the utility of Halstead's equations altogether, while others [2.40, 2.41] note the need to establish *ad hoc* methodologies in order to apply Halstead metrics to modern programming environments. Various authors [2.42] have proposed relatively minor modifications of Halstead's formulations (such as those given in Equation 2.1) in an attempt to improve the quality and range of applicability of the metrics. However, as in the case of Zipf's law, computer scientists are once again confronted with a set of empirical generalizations that exist without a good theoretical framework. Halstead metrics were based on a paradigm of programming that reflected the fetch-execute cycle of the assembly languages of the late 1960's. As such, the application of Halstead's original equations to modern, higher-level programming languages is not clear and certainly requires the development of counting and clustering rules which are required to map the native constructs of modern, high-level languages onto Halstead's original binary classification scheme.

Both Halstead and Zipf attempted to use rationalizations based on psychological concepts, such as the "Stroud number" [2.43] and the "principle of least effort" [2.44], to support their work. Unfortunately, further psychological research showed these concepts

to be weak; and they are now considered simply outmoded within cognitive science [2.45].

$$\hat{N} = n_1 \lg n_1 + n_2 \lg n_2 \qquad (2.1$$

$$\hat{N} = n_1! \lg (n_1!) + n_2! \lg (n_2!) \qquad (2.2$$

where lg denotes the logarithm to base 2

In order to even conceptually apply Halstead's original-length-equation estimates to modern language constructs, a two step process is required: It is first necessary to translate high-level language source code into an operationally equivalent low-level form. It is the length of this translated, operationally equivalent, low-level form that is then estimatable by Halstead's metrics.

Furthermore, in higher-level languages, it is more difficult to accept Halstead's implicit view that all operators are equally important [2.32]. The functional partitioning of the linguistic elements used in all computer languages into the two disjoint categories proposed by Halstead ignores the importance and nature of higher-order native constructs.

Both high- and low-level constructs may be predefined or native to a language. However, high-level constructs, such as a sort utility, are clearly not linguistically nor computationally equivalent to low-level operators such as division. Such difficulties often complicate even simplistic lexical analysis. More comprehensive linguistic analyses which are based on context-sensitive or hierarchic features, while fundamentally worthwhile, are very difficult to design, interpret and analyze.

Some authors [2.32] have sought to refine the concept of Halstead's length equations by basing them explicitly on an hierarchical scheme of operators and operands that attempts to profile the operational features of computer languages. While initially

appealing, this approach is made difficult by the idiosyncrasies and often ill-conceived preferences of computer language designers which have contributed to the production of an immense plethora of domain-specific computer languages and their native constructs. Models such as those given in Equation 2.3 are used to quantify the impact of the powerful native constructs found in higher-level languages on source code length. Within such models [2.32], there are k types specified in the code's syntactic feature set.

$$\hat{N} = \sum_{i=1}^{k} n_i \lg (n_i) \qquad (2.3$$

When $k = 2$ with k_1 = { operands }, k_2 = { operators }, Equation 2.3 reduces to Halstead's original length equation; and the estimates provided by Equation 2.1 conform to those found by Halstead for his equations [2.34]. equation 2.3 is of course very closely related to Shannon's 'information function' which is the same as the well known 'entropy function' of statistical mechanics [1.40, 1.65]. Shannon proved that there is a unique mathematical function, which is related to Equation 2.3, that satisfies certain reasonable postulates that describe the abstract concept of 'information'. Shannon showed that there is one, and only one, way to assign a quantitative value to measure information, provided that the measure satisfies two postulates. The first postulate is that if all outcomes are equally likely then the information measure is a strictly increasing function of the number of possible outcomes. The second postulate is that the amount of information provided by an 'answer' is independent of the way in which the answer was found. Information theory measures have been extensively used in the design and analysis of coding and transmission systems. They have also been applied to a wide range of studies and it is not surprizing to see variants of 'entropy' measures used in lexical measures for approximating 'complexity' in natural language texts and source code.

Halstead's equations and their refinements differ fundamentally from Zipf's length equations in that Zipf, perhaps wisely, chose to ignore the role that a word played in linguistic use. By adopting this simple device, Zipf avoided considering the multiple roles that a word often plays in natural language semantics. Furthermore, even the number and relative usage of the parts of speech vary among languages. It is exactly the ramifications of these lexical effects on Halstead's metrics which have raised serious concerns about their practical utility. In addition, many lexical elements in high-level programming languages are splintered across the source code. For instance, does the selection construct 'If..Then..Else' count as a single operator? In practice, it appears that, in order to practically apply Halstead's metrics, a set of consistent and rather *ad hoc* lexical rules must be adopted.

The lexical equations developed by both Halstead and Zipf describe macroscopic behavior while offering little insight into whatever mechanisms that underlie this behavior. However Halstead's equations are applicable to the hierarchical analysis of context-free code. Halstead's equations are also applicable to the lexical analysis of context-sensitive text that is both expository in nature and relatively free of redundancy. Thus Halstead's equations are applicable to the analysis of scientific writing but fail in the analysis of poetry.

Zipf observed that the product of the rank, R , of a word and its frequency of occurrence, F , is a constant, c , for most natural languages. Zipf's length equation (which is given in Equation 2.6) is fundamentally different in form from those proposed for computer languages by Halstead and his followers.

$$c = R * F \qquad (2.4$$

Equation 2.4 is usually written as:

$$\log F = -\log R + W \qquad (2.5$$

where the size N of a piece of natural language text containing M distinct terms or vocabulary items is approximated by $\hat{N}$ as :

$$\hat{N} = c \sum_{i=1}^{M} i = c * M * (M + 1) / 2 \qquad (2.6$$

Zipf's law is insufficient to adequately describe vocabulary use in computer languages. A second, slightly more complex approach, which was first proposed by Halstead in his analysis of computer languages, partitions vocabulary into two fundamental functional classes. Halstead proposed a classification scheme which partitioned source code elements into one of two possible disjoint sets:{operators},{operands}. Halstead's lexical model works on restricted domains of natural language text, such as technical reports or scientific papers [2.34] but fails to describe natural language texts which contain significant redundancy.

More elaborate models attempt to establish functional classes, such as those based on the parts of speech in a natural language, or various arbitrary archetypical hierarchies of computational operators and operands [2.32].

This laudable approach is made difficult, if not fatally flawed, by context-sensitive effects which confound simple classification schemes. For instance, it is often not possible to determine the part of speech of a vocabulary item without knowing the context in which it was used. Classification schemes which attempt to ensure the specificity of their taxonometric basis are, of course, desirable in that they are the easiest to use but also, unfortunately, the hardest to derive. Chapter 4 presents a classifications method which has shown itself to be useful in describing and predicting natural language word structures found in English.

Within the domain of computer language design, the use of semantics to clarify the syntax of an ambiguous operator allows one to extend the use of a concept such as addition to many data types. Such semantic extensions within computer languages are referred to as 'overloading' [2.46]. This term reflects the belief that such generalizations of concept in computer languages are against the spirit of strongly typed languages. The use of overloaded operators, for instance, often increases the cognitive burden of really understanding

the code, let alone remembering its significance [2.47]. Overloaded operators enhance the likelihood of error in code that humans consider complex [2.48].

The impact of overloading on even such simple software metrics as those proposed by Halstead means that it is necessary to devise counting rules [2.49, 2.32] specific to source code for each programming language. The '+' operator when applied in one context may be used to specify set union, while it might also be used to specify the addition of two numeric types or the concatenation of two string types. The task of interpreting the semantic meaning of a lexical operator in a higher-level computer language often requires knowledge of its contextual application.

A second major theoretical difficulty in positing functional classes is establishing what we, as computer scientists, mean by "their equivalence" [2.50]. Much attention must be paid to the spirit in which one accepts the equivalence of constructs. Physical science, since the turn of this century [2.51], usually accepts operational equivalence while computer scientists are less pleased with this notion. Some computer scientists accept functional equivalence, while others insist on structural * rather than operational equivalence [2.47]. Semantic equivalence [2.52] is the key concept underlying transformational grammars that attempt to show a common deep structure or semantic equivalence between two natural language sentences [2.53]. To date, the concept of deep structure has not been explicitly used in computer science. Frame based, object-oriented artificial intelligence systems [2.54, 2.55, 2.56] come closest to embracing some of the basic ideas of deep structure. It is, of course, impossible to establish a classification scheme based on archetypes or seminal structures without a concomitant *a priori* agreement on the basis of equivalence. In this regard the myriad of abstract objects and classes with which computer science presently works resembles the arbitrary collections of fauna and flora that preoccupied the pre-Darwinian mind. This state of affairs, while

* The models developed in this thesis focus on simple structural equivalence. For instance all words (types) conforming to a given lexical frame are considered to be 'equivalent' within that frame.

problematic, is hardly surprising considering that computer science is still an extremely young discipline.

Unfortunately, the practical ramifications of our present situation are immense. Intent errors, introduced by the ambiguity between the semantic intent and lexical representation of the source code, are considered [2.57, 2.58] to be the most serious and expensive of coding errors. Intent errors are often associated with complex code, which is also typically heavily overloaded [2.59]. Such error-prone software seems to possess characteristic lexical and syntactic signatures or styles. The fundamental sources of these immense practical problems remain poorly understood today. It is hoped that analytical work on natural language structures will eventually offer insight into similar problems with the more restricted languages used in conventional programming. The next section of this chapter will describe current empirical approaches to style analysis in greater detail.

2.3 GRAMMATICAL AND SEMANTIC THEORY

The computational analysis of style whether it be that of Byron or Knuth can of course be undertaken at the lexical, grammatical and semantic levels.

In the previous section of this chapter we discussed the foundations and use of lexical analysis. The problems encountered in lexical analysis usually reduce to difficulties encountered in one-to-many mapping scheme where the orthographic form of a word is in itself insufficient to uniquely specify its intent [2.60]. For instance a single word in English may have a great number of meanings and be used as many different parts of speech [2.61]. The single orthographic form of a word, such as *present*, may also be pronounced differently as its use changes from noun to verb [2.62]. The results of overloading symbols in natural languages are no less ominous than those encountered in computer languages. Lexical overloading forces us to use context to determine the meaning or intent of a word.

In written work the simplest lexical context can be estimated by concordance studies which use n-gram statistics [2.63], based on a

Markov process model, to attempt to isolate the intended meaning of a word on the basis of its neighboring terms. Chapter 9 describes the use of contextual-probabilities in modeling the frequency-of-use of a given word. Such studies carry with them the modern paradigm of 'guilt by association' which is really a classic fallacy in the form of an argument *ad hominen.* Concordance studies do not require a parse of a sentence and are hence both computationally efficient and immune to the simple or even poor grammatical form that one expects from the transcripts of verbal discourse or Shakespeare. It is in this sense that lexical analysis can surmount some of the problems introduced by the immense difference between 'linguistic competence' and 'linguistic use' long touted by Chomsky [2.64]. Such differences are even extreme when one compares the written and verbal performance of an individual. In 1991 it remains an open question as to how it is that we grasp the intent of verbal utterances.

The formal study of syntactic structures in language dates primarily to the work of one, still very active, researcher. Grammatical analyses were essentially a subjective domain until Chomsky published a series of articles on the syntactic nature of language [2.65]. These works established a hierarchy [2.66] of linguistic structures which surface at a grammatical level. The form and behavior of the production rules first used by Chomsky to describe different levels of grammatical complexity precipitated a revolution of thought in their domain of discourse. The Chomsky hierarchy of language is undoubtedly one of the greatest intellectual achievements of this century. In his iconoclastic works, Chomsky at once reduced the role of vitalism in linguistics and supplanted it with a palatable and viable mathematical theory of linguistic grammars that has since fueled basic research in linguistics and computer science. A digestible synopsis of this work was published in his text, Aspects of The Theory of Syntax [2.67], in 1965. Chomsky humbly opens this text with the following statement:

> *"The idea that a language is based on a system of rules determining the nterpretation of its infinitely many sentences is by no means novel. Well over a century ago,*

it was expressed with reasonable clarity by Wilhelm von Humboldt in his famous but rarely studied introduction to general linguistics....

Nevertheless, within modern linguistics, it is chiefly within the last few years that fairly substantive attempts have been made to construct explicit generative grammars for particular languages and to explore their consequences. No great surprise should be occasioned by the extensive discussion and debate concerning the proper formulation of the theory of generative grammar and the correct description of the languages that have been most intensively studied. The tentative character of any conclusion that can now be advanced concerning linguistic theory, or, for that matter, English grammar, should certainly be obvious to anyone working in this area."

Twenty-five years after Chomsky's publication of, Syntactic Structures [2.65], the tentative character of its conclusions remains the same. If agreement on the principles of lexical analysis is difficult then it is hardly surprizing that grammatical analysis has proven to be close to intractable. This is not to say that the basic ideas of the hierarchy of syntactic form has not been generally accepted by linguists and adopted wholeheartedly by computer scientists concerned with language design. Rather it has proven very difficult to apply and implement these abstract syntactic forms. For instance there is still doubt in the literature [2.68] as to whether English is necessarily a context-sensitive language!

Unfortunately these questions appear simple in comparison to perhaps more important and principle questions that have preoccupied the structuralist school. For instance consider one that preoccupies us here: Does form predispose function?

Within a Platonic system the concept of form is, in itself, a very important component of an individual's abstracted reality [2.69, 2.70]. The relation of form to function within living systems underwent a process of profound reevaluation in science with the publication of the works of D'Arcy Thompson [2.71] at the beginning

of this century. Such concerns surfaced in twentieth century art in the form of 'assemblage'. Assemblage represented a way of creating art almost entirely from pre-existing elements or 'found objects'. The artist's contribution in this school was to formulate the links between known 'ready-made' objects rather than making them *ab initio*. This reevaluation of the role of the artist led to the rise of dadaism [2.72] in the early 1920s. It is the evolution of processes such as these that led to the advent of a new determinism that predisposed the revolutionary studies of Piaget [2.73, 2.74] and the abstract structuralists such as Chomsky [2.75] and Minsky [2.76, 2.77]. The results presented in Chapters 4 & 6 indicate that a modified form of assemblage can be applied to the analysis of English language word structure,

The formalisms of the hierarchy of grammar developed by Chomsky are critical to entire areas such as syntactic pattern recognition which was pioneered by the late K.S. Fu. In fact generative grammars have helped form the basis of a wide range of structural models such as those developed within computer science for the analysis and understanding of N-dimensional signals [2.78, 2.79]. Chapter 7 presents a comprehensive structural model of the principle components of the English lexicon.

One dimensional analyses are typically associated with string parsing and have been used in computer science [2.80, 2.81], molecular biology [2.82], and linguistics [2.83]. Two dimensional pattern recognition applications of generative grammars have been used in many imaging [2.84] and medical applications [2.85], including the development of grammars suitable for parsing electrocardiograms [2.86]. Three dimensional applications include the work by Waltz [2.87] on a lexicon of the 18 vertices needed to characterize 3-D trihedral surfaces as well as in Marr's primal sketch theory of vision [2.88]. Complex N-dimensional grammars show promise for use in signal processing domains such as those encountered in speech recognition [2.89, 2.90].

In some applications generative grammars are used to accept a string as parsable and therefore conforming to the rules of some arbitrary language [2.91]. Such applications only establish the

syntactic acceptability of a string of code and do not provide any assurances that the code has in fact any semantic meaning. While such applications only establish the syntactic acceptability of a string in a grammar they work over all possible finite strings. This feature provides immense power to pattern recognition problems which must be able to accept for analysis new, or at least previously unencountered, signals. Chapters 4 & 5 demonstrate the use of such models in describing and generating lexical frames in English.

In other applications generative grammars are used to specify or differentiate forms. For instance syntactic algorithms can be applied to the task of classifying airplanes from pictures of even partially obstructed views of their structure. In such cases the parse of the form of the obstructed structure can be used to specify which secondary, statistical, pattern recognition routines should be used to either verify the object's classification or enhance its image. Of course procedures such as these are only applicable to the recognition of objects which are know to exist and have already been classified. The classification of a polygon using Waltz's algorithm usually results in a unique labelling of its vertex set. As it turns out [2.92], Waltz's work may have fundamental importance to our understanding of visual processing. Various visual illusions, such as those which are interpreted by our eye as the image of an object taken in one of two possible perspectives, are found to exhibit two different mutually consistent vertex labellings under the Waltz algorithm.

Still other applications of generative grammar use the parse of a signal to specify, or partially interpret its meaning. Some diagnostic systems for the syntactic interpretation of electrocardiograms are good examples of such applications. The structure of the electrocardiogram waveform, as given by its parse, is considered to be indicative of the patient's cardiac status and in some cases is considered to be diagnostic. Few applications however have the necessary one-to-one functional relationship which underlies the success of such mappings of form to function.

The components of syntactic structure are used extensively as domain knowledge in artificial intelligence applications where they can heuristically help clarify the semantic meaning of a signal.

However when we view structure, in and of itself, we find that it is usually devoid of meaning or at best ambiguous in its interpretation [2.93].

How important is syntax as a vehicle to understanding? If Whorf's postulate on the importance of language to the process of thought is correct, then what role does syntax play in linguistic function and understanding?

Some research being undertaken by Minsky's school is attempting to demonstrate that such things as music are in fact empty syntax [2.94]. It is Minsky's view that music conforms to grammars which can be used to simulate the works of specific composers or periods. In fact, he suggests that the pleasure derived from a wide range of musical forms could well be described by a simple transformational grammar.

Transformational grammars, such as those developed by Chomsky [2.95], have been used to convert metacategories such as active sentences into their semantically equivalent passive forms.

For instance transformational grammars provide the correct syntactic form of the translation into the active voice of any syntactically correct passive sentence within a grammar. It may be that transformational grammars which show promise in establishing semantically equivalent, syntactic forms can be applied to the task of reducing the endless variation of expression that actually describes the vibrancy and value of natural language into a canonical form which is best suited to the analysis of its meaning by some agency.

Transformational grammars operate on syntactically different, yet semantically equivalent forms. Within the domain of mathematics such grammars could be used to describe various mathematically equivalent forms of an expression. Transformational grammars guarantee the semantic equivalence of a set of expressions or sentences without regard to their underlying semantic meaning. Such grammars, which are ubiquitous in traditional mathematics, are much more difficult to construct in context sensitive domains such as linguistics [2.96]. Alas transformational grammars, which promise the existence of a formal scheme for the reduction of well formed natural language sentences into some canonical form, stop short of

providing insight into whatever semantic meaning there is in such canonical forms [2.97, 2.60].

If the richness of syntax is really a matter of transformational form then how is it that we derive meaning and semantic understanding from structure? What aspects of language really limit or determine thought?

These questions remain central to the entire area of artificial intelligence. They are also reflected in research work on the number of native constructs and the size of the lexicon in computer languages.

Unfortunately today's academic, whether a linguist or computer scientist, is as preoccupied by such quests as was an entire generation of previous researchers who witnessed the publication of Chomsky's hierarchy of form and Minsky's text The Society of Mind.

It may be that Minsky and his colleagues will soon demonstrate the poverty of empty syntax in such domains as music and poetry. Such results, while very important and interesting, would *per se*, offer little insight into the nature of thought and semantic interpretation.

There has been considerable practical interest in grammatical and semantic analysis in both the artificial intelligence and software engineering communities over the last decade. Such work has led to the development of utility packages which are capable of detecting awkward grammatical and semantic constructs in texts [2.98]. Further work is needed to determine whether the methods and models developed in this thesis are applicable to such practical applications.

The software engineering community have sought to develop systems suitable for assessing the lexical, syntactic [2.99], and semantic style of computer code. These systems can be used to detect awkward or error prone constructs such as nested 'If..Then' clauses or GOTO's in computer code as well as the use of semantic forms such as recursion in source code. Present systems attempt to isolate not only chunks of error-prone code [2.100, 2.101, 2.102] but also computationally inefficient constructs [2.103]. Work in this field is based on the belief that source code is read and understood in psychologically meaningful modules referred to as chunks [2.104, 2.105]. It is the premise of this work that it is possible to build both a

comprehensive database of code chunks and a thesaurus relating statistically defined 'semantically equivalent' chunks. Smart optimizing compilers would then proceed by first recognizing an abstract chunk such as that used to implement bubble-sort and then substitute it with that specifying the equivalent, but computationally preferable, quick-sort routine [2.47].

The artificial intelligence community has sought to develop natural language systems suitable for enhanced word processors which are capable of detecting awkward constructs [2.60], tense incompatibilities [2.98], ill-formed or run-on sentences, and grossly ambiguous pronoun, or clause referencing [2.60, 2.106, 2.107]. The artificial intelligence community has also focused an immense amount of effort on the development of efficient parsing systems suitable for natural language processing [2.108, 2.109, 2.110, 2.111, 2.112, 2.113]. It is also expected that such systems will be of great value in speech recognition research [2.114]. While it can be argued that a successful parse of a sentence will not revel its semantic intent. It is also true that a successful parse can not help but improve attempts at deriving the semantic meaning of a sentence and appears to be certainly needed to determine its deep structure.

The last three decades have witnessed an immense growth in our present understanding of language whether it be at the lexical, syntactic, transformational or semantic level. This growth has been the result of great effort.

Linguistics offers a domain of enquiry wherein it is likely that its most prolific and well respected living authors and native speakers have precious little awareness of just how it is that they create masterworks. It is as if high level linguistic processes are to remain by their very nature inaccessible to analytical analysis by those who best use them.

2.4 PATTERN RECOGNITION

In a restrictive sense of the word, pattern recognition may be viewed as the process of classifying and recognizing unknown patterns from a set of *a priori* known patterns. A wider view of pattern recognition expands the domain of this field to include the processes involved in recognizing or discovering the basic *a priori* patterns needed to achieve domain specific recognition tasks. This second view of pattern recognition makes its study fundamental to the pursuit of artificial intelligence. Comprehensive linguistic models must address both of these issues. For our purposes we shall limit the review given here to a discussion of three core components of pattern recognition research: classification reasoning, classification schemes and classification/recognition problems.

2.4.1 CLASSIFICATION REASONING

Patterns appear to be critical to the means by which we interpret the world [2.115, 2.116, 2.117]. Humans readily distinguish many very complex patterns such as faces, handwritten text, diseases, music, cars, flowers, etc. Within the framework of pattern recognition, decades of research have been spent attempting to define the processes by which we are able to recognize signals that stimulate our senses and thoughts. The fundamental processes which enable us to discover and recognize complex patterns are still relatively unknown. This is not to say that domain specific recognition research has not yielded both good theoretical results and many practical pattern recognition systems [2.118, 2.119, 2.120]. If Minsky's thesis of a 'society of mind' is correct then it is not surprising that we have not found a generic pattern recognition system in humans. The 'society of mind' thesis implies that each mental agency has its own, domain specific, physical-symbol system. If this is the case evolution [2.121] would assure that pattern recognition processes would be optimized to meet the special needs of each of these physical symbol systems. Much of the early research

in pattern recognition and artificial intelligence focused attention on the ' mechanization of perception and thought processes'. After four decades, it has become clear that the vast majority of problems which have been "successfully" solved contain a common element, classification. These successful solutions have ranged from simple classification problems to multiple-classification problems whose solution mandated that sub-classification reasoning be used [2.29, 2.122, 2.123, 2.124]. However the basic fundamental processes underlying these success stories, including the process of classification, remain poorly understood.

Pattern classification is often considered to be the basis of pattern recognition. The ability to classify patterns into groups gives us a foundation necessary for recognition. Chapter 4 presents a single feature that appears to be of great use in clustering and classifying the lexical structures found in English. Given this information we can attempt to recognize new patterns by processing them to achieve the best match of a pattern to an element in some set of previously learned patterns.

2.4.2 RECOGNITION & CLASSIFICATION

Pattern classification appears to be an important part of the recognition problem. In fact both the recognition and classification of patterns are considered to be among the most fundamental of human activities.

The general guidelines for the recognition process focus on three issues. First the pattern must be perceived by the senses, Second, patterns of the same class must have been perceived and catalogued beforehand. Third, an equivalence or correlation must be estab'' ned between the perceived pattern and a past perception.

It is the study of this third consideration that has fueled much work on the theoretical applicability and the practical use of decision rules [2.125], statistical methods [2.126, 2.127, 2.128] and traditional as well as fuzzy logics [2.129, 2.130] in classification.

Classification itself requires some feature selection process. This process, as we shall see shortly, turns out to be both very important and very difficult.

First we need to consider the role of decision rules on the classification process. Developing a set of decision rules and correctly applying them are two main aspects of a pattern recognition process. Chapter 10 illustrates various uses of the sets of decision-rules. These sets of rules were derived from clusters of lexical structures which are based on the feature classification method proposed in Chapter 4 of this thesis. The various uses of these rule-based systems include deriving roots and hence a bit of the semantics of a word. However in order for a pattern recognition system to adapt or learn it must be possible to change existing rules or establish new ones. Hence the decision rule database must be modifiable as well as context-sensitive. Two early examples of recognition strategies are depicted in Figures 2.2 and 2.3.

Figure 2.2 outlines a process where initialization occurs before recognition. The decision rule base is fixed in this scenario and adaptation cannot occur. Figure 2.3 refines this process by allowing decision rules to be modified on the basis of the system's detectable error. Such adaptive systems allow for the learning and recognition phases of the pattern recognition system to occur in tandem.

To use decision rules effectively, the process must incorporate some method for relevant feature selection. The objective is to define and extract features for classification groups which will allow for the correct and efficient recognition of new patterns via decision rules. These features also dictate the formulation of the decision rules in that eventually such rules must recognize a pattern by its features.

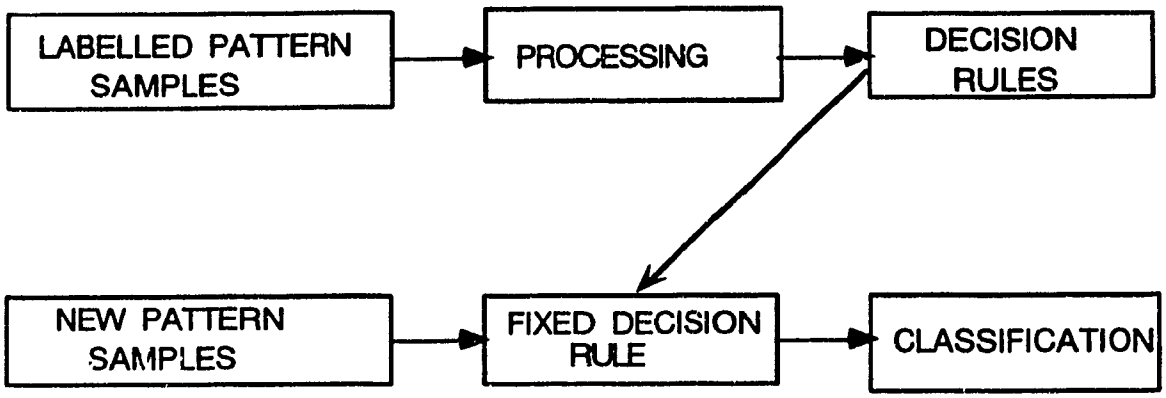

Figure 2.2 Classic fixed pattern recognition system. After [2.86]

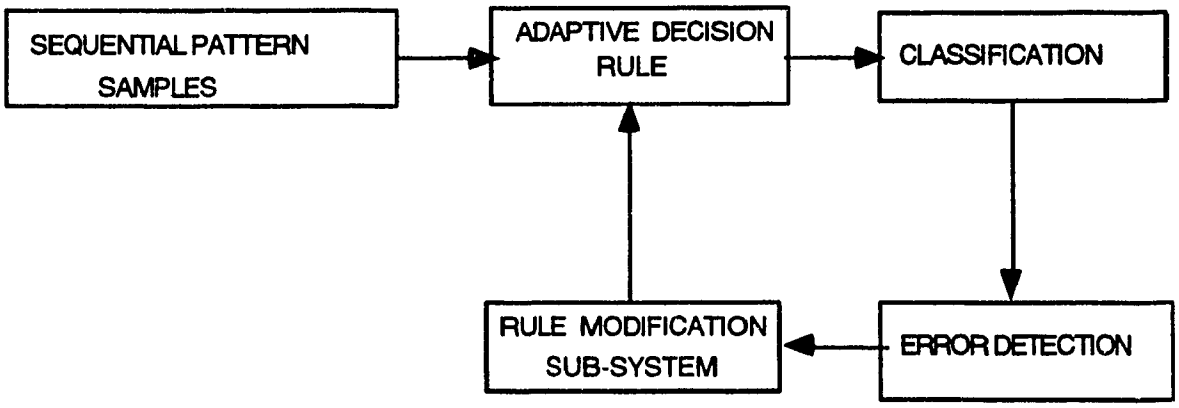

Figure 2.3 Adaptive pattern recognition system. After [2.86]

Should the feature set be poorly chosen it may not be possible for the pattern recognition system to adapt to patterns. Some recent work has focused on analyzing just what changed the rules in an adaptive rule base system. This focus is based on the observation that modified rules can complicate the pattern recognition system. New patterns may require a more comprehensive feature than the one initially chosen! Similarly more than a single feature may be needed to correctly classify an ambiguous pattern. Chapter 10 demonstrates the application of a divide-and-conquer approach, based on the feature-classification-method developed in Chapter 4, to the selection of an appropriate set of grammar rules for words of a specific lexical structure.

For example the adaptive addition of a new feature for recognition can corrupt the entire classification structure. Additional features must be added conservatively and only with the realization that a retrospective study is needed to confirm that their inclusion will not degrade the system's performance. For instance whole groups of classes which were clearly defined by the previous decision rule may be now incorrectly classified by a modified rule-base which adapted itself to correctly classify an exceptional case. As such the rule modification sub-system depicted in Figure 2.3 is critical to the success of adaptive systems. In all cases the careful selection of features in both fixed and adaptive environments is critical to the overall success of the pattern recognition system.

A proper detailed study of the functional requirements for a given application which exhibits well-defined, stationary behavior can lead to the choice of a very good feature set for a fixed system. Unfortunately systems with fixed rule-bases are context-free and have very limited scope. Furthermore the performance of fixed rule-base systems must be monitored to detect potentially serious error introduced because of their inability to handle 'new' cases.

Adaptive system such as those depicted in Figure 2.3 must implement their rule-modification subsystem very carefully if they are to avoid degrading the integrity and overall performance of the pattern recognition process. Prudence requires that a form of cost/benefit analysis must be used to access the real value of any modification to

the system's rule-base. Realistic guidelines for such an analysis are very complex.

The concept of 'adapting data' in pattern recognition processes makes these applications ideally suited to object-orientated programming paradigms [2.131]. In such a paradigm a great deal of time is spent analyzing the properties of the system's data in an attempt to formulate meaningful operations on that data. Object-oriented paradigms exploit the distinctive attributes of the data set. Such features are isolated only after extensive investigation in that they are usually neither initially known nor are they obvious. This investigative procedure can often yield simple and effective decision rules. Chapters 9 & 10 illustrate the applicability of simple morphological and probabilistic features to the extraction of context-sensitive natural language grammar rules.

2.4.3 CLASSIFICATION/RECOGNITION PROBLEMS

A great many authors have expressed their opinions on the fundamental problems of pattern classification and recognition. The problems expressed usually are related to the origins of recognition and classification. Central to these discussions is the weak fundamental basis of the field. For instance such concepts as 'shape', 'cognizance' and 'identification' are not clearly defined. While it is possible to operationally define such concepts, such definitions are only suited to their particular branch of recognition research. It may be that shape and identification are intrinsically dependent on the objective to be achieved. This observation is consistent with the 'society of mind' paradigm. A second concern is the degree of complexity needed in a pattern recognition system to assure its accuracy. In the course of designing a pattern classification scheme more than one type of process will most likely need to be implemented. An implication here is that more complex implementations provide more complete solutions. Systems which only implement one basic recognition system usually drastically reduce the domain of recognition problems which they can resolve.

Perhaps the most fundamental classification problem facing research in pattern recognition arises in the difficulty encountered in correctly defining 'natural' versus man-made or 'architected' patterns. Artificial recognition processes which deal with both natural and architected patterns encounter many 'exceptions' which are a direct result of the constraints and restrictions in the physical-symbol system adopted by their designers. As discussed previously adaptive systems can use data to modify or fine-tune a particular physical-symbol system. The adaptation process is an order of magnitude simpler than using data to determine which of the many possible physical-symbol system to invoke. Man-made or architected patterns such as an alphabet or the Dewey decimal are relatively easy to define and classify. Natural patterns such as those observed in plants, clouds, rocks and non-linear processes have been traditionally very difficult to analyze, define and classify. This difficulty can be traced to our relatively poor understanding of the processes underlying the development of natural patterns. A lack of understanding of the basic processes underlying a phenomenon can make it much more difficult to analyze and classify patterns produced by the process. This thesis tackles the problems of analyzing a system which apparently contains both natural processes and architected patterns. For unlike man-made languages such as ADA, natural languages lexicons have undergone extensive natural selection. In Chapter 7 one observes both fractal-like and architected components of the lexicon's frames. Fortunately our understanding of the patterns produced by natural processes has improved markedly over the last decade [2.132].

In fact generic mathematical models have been recently developed which incorporate fractal geometry and the principles of chaotic processes. The existence of these models allows one to classify patterns in terms of meaningful features such as the fractal dimension of an object. Wether these models are sufficient or necessary to correctly analyze natural processes and the patterns they produce is at the moment an open question. To what extent is such analysis whimsy? To what extent is the fractal nature of natural artefacts arbitrary? Progress in this field has been rapid in that these models are testable. The results of such work are needed to form the

basis of a new generation of physical-symbol systems suitable for the classification and recognition of many presently intractable natural patterns. The application of this approach to the study of the evolution of context-free language is the subject of further research.

2.4.4 FEATURE SELECTION

Features may be thought of as attributes of data which promote recognition. The number and choice of features often limits the accuracy and resolution of pattern recognition systems. Practical questions arise whenever one needs to determine how many features are needed to recognize or classify patterns [2.86, 2.78].

There is also concern over which of the many mutually correlated features are best suited for use in a model. Central to the problem of feature selection is the degree of our understanding of the process which generates the patterns. Features which are selected to correlate to parameters of a naive linear model of a non-linear natural process cannot be expected to work outside of a very limited range of application.

The process of feature selection for recognition systems that focus on architected systems is a far simpler task than those that seek to recognize and classify pattern formations that are the result of poorly understood, non-linear 'natural' processes. It remains clear that the mathematics and models of the physical universe that are assumed by the researcher in fact form a *de facto* basis for rational feature selection. For example a classic reductionistic approach to these problems is to choose features which represent reduced primary components of the data. This approach uses the principle of 'assemblage' [2.72] with its implicit assumption of basic archetypes from which all patterns can be constructed. Unfortunately the issue of the primary components refers us back to the models and mathematics that are implicit to our systems [2.1, 2.12].

Fundamental problems occur when the features selected do not reduce patterns to their primary components. For instance features which only partially decompose patterns can incorrectly categorize

them because too general a view was adopted in the decision rule analysis.

Feature selection implicitly focuses our attention on matching the complexity of the processes involved in producing a pattern with the simplicity of the model used to describe, recognize and classify these patterns. We attempt to define pattern classes which are important to us in terms of the simplest models needed to successfully handle the task at hand. Practical considerations can be used to choose the most accurate and economically measured features of our model. These considerations are not based on a desire to achieve elegance but rather the desire to produce an understandable, maintainable, accurate and non-arbitrary system. Once a feature is chosen one must deal with statistics including the degree of resolution used in its measurement and the effects of measurement error on the pattern recognition system. Chapter 5 demonstrates that the prefix model developed in this thesis needs to be complemented by a model like that developed in Chapter 6, which establishes the relative magnitude of the structures produced by the prefix models. A pattern recognition system is considered to be robust if it can recognize ambiguous patterns in noisy environments. Of course, robust systems usually achieve their results through redundant computation and are thus computationally expensive.

2.4.5 MAPPINGS & DATA TRANSFORMATIONS

The data transformation section of a pattern recognition system can be described as a mapping scheme which adopts procedures to define an equivalence between a particular pattern class and a newly perceived pattern. Such procedures use sets of rules which are based on the data's features and a well defined sequence of computations based on the rules which define equivalence classes.

The problem of defining the equivalence relations in such systems is very difficult. Careful feature selection helps reduce this task, however equivalence is usually defined quantitatively by computing and comparing a sufficient number of data properties. Such operationally defined approaches are statistically practical but

error-prone. Chapter 3 discusses the issues involved in selecting the words used in this work.

Key to the problem of successful pattern recognition is not only the relevance of the selected features but the amount of raw data needed to reliably measure a feature.

The simplest solution to this problem has been to extract as much feature information as possible, via sound mathematical transformations, from a relatively small but accurately measured and reproducible sample set that spans critical areas of the sample space.

A very important problem in this mapping process is the ability of the system to assure the integrity of the pattern classes under transformation. The pattern classes, or archetypes, were chosen, *a priori*, on the basis of distinct characteristics. Sometimes the transformations adopted by the pattern recognition system force some of the original pattern classes to be radically reclassified upon reevaluation. Such reclassification must be analyzed very carefully because if the original classification was correct then this forced reclassification under transformation implies that this process has introduced significant error into the system. If however forced reclassification under transformation is the result of the detection of an error or ambiguity in the original data set then the pattern recognition system has improved the accuracy of its own classification scheme.

2.4.6 COMPLEXITY & AMBIGUITY

Empirical observations have shown that comprehensive solutions from pattern recognition systems are associated with the degree of complexity found in the system itself. Complexity is, of course, a poorly defined term and can refer to the choices of underlying models, feature selection processes, data measurement and transformations [2.133, 2.134]. The choices made in each of these areas drastically effects the overall utility of a pattern recognition system. While every effort has been made to use an authoritative comprehensive and academic lexicon as the basis for this analysis, there is a need for further research to confirm the results of the

models developed here on the complete Oxford English Dictionary. It is expected that our data set which consists of all valid English words listed in the Oxford Paperback Dictionary has yielded results which are indicative of those obtainable from an analysis of the entire English lexicon.

2.5 REFERENCES

[2.1]. W. I. Beveridge, The Art of Scientific Investigation: An Entirely Fresh Approach to the Intelligence Adventure of Scientific Research, [3rd ed.], Vintage Books, Random House, New York, N. Y., 1957.

[2.2]. D. B. Lenat, "The Ubiquity of Discovery," Artificial Intelligence, Vol. 9, No. 3; 1977.

[2.3]. P. Langley, "Data-driven Discovery of Physical Laws," Cognitive Science, Vol. 5, pp. 31-54; 1981.

[2.4]. see 1.61

[2.5]. G. U. Yule, A Statistical Study of Vocabulary, Cambridge University Press, Massachusetts, 1947.

[2.6]. G. Herdan, The Calculus of Linguistic Observations, Mounton and Company, S-Gravenhage, Netherlands, 1962.

[2.7]. B. Mandelbrot, "Structure Formelle des Textes et Communications: Deux Etudes," Word, Vol. 10, pp. 1-27; 1954.

[2.8]. H. Kucera, N. Francis, Computational Analysis of Present Day American Emglish, Brown University Press, Providence, Rhode Island, 1967.

[2.9]. J. Carroll, P. Davis, B. Richman, Word Frequency Book, Houghton-Mifflin, Boston, Massachusetts, 1971.

[2.10]. see 1.62

[2.11]. G. Miller, "The Magic Number Seven, Plus or Minus Two: Some Limits on our Capacity for Processing Information," Psychological Review, Vol. 63, pp. 81-97; 1956.

[2.12]. R. Feynman, The Character of Physical Law, MIT Press, Cambridge, Massachusetts, 1965.

[2.13]. G. Miller, P. Gildea, "How Children Learn Words," Scientific American, Vol. 257, 3, pp. 94-99; Sept. 1987.

[2.14]. B. Mandelbrot, In Informational Theory and Psycholinguistics, ed by B. Wolman, E. Nagel, Basic Books, Inc., 1965.

[2.15]. J. B. Estoup, Les Gammes Stenographiques, privately printed for the Institute Stenographiques, Paris, France, 1916.

[2.16]. D. Knuth, The Art of Programming: Sorting and Searching, Vol. 3, Addison-Wesley, Reading, Massachusetts, 1973.

[2.17]. W. P. Heising, IBM Systems Journal, Vol. 2, pp. 114-115; 1963.

[2.18]. G. Herdan, The Calculus of Linguistic Observations, Mouton and Company, S-Gravenhage, The Hague, Netherland, 1962.

[2.19]. A. V. Hall, G. Dowling, "Approximate String Matching," Computing Surveys, pp. 381-402; Dec.,1980.

[2.20]. G. Herdan, Language as Choice and Chance. P. Noordhoff, Gnoningen, 1956.

[2.21]. C. B. Williams, "Yule's 'Characteristics' and the 'Index of Diversity'," Nature, Vol. 157, p. 482; 1946.

[2.22]. see 1.45

[2.23]. M. J. Adams, A. Collins, "A Schema-Theoretic View of Reading," R. O. Freedle, ed., New Directions in Discourse Processing, Vol. 2, Norwood, New Jersey, Ablex, 1979.

[2.24]. A. Ellis, Rational Emotive Theraphy, R. Corsini, Current Psycho-Therapies, ed by F.E. Peacock, Itasca, Illinois, 1984.

[2.25]. P. H. Sellers, "The Theory and Computation of Evolutionary Distances: Pattern Recognition," Journal of Algorithms, Vol. 1, pp. 359-373; 1980.

[2.26]. D. Sankoff, J. Kruskal, Time Warps, String Edits and Macromolecules, Addison-Wesley, Reading, Massachusetts, 1983.

[2.27]. B. Curtis, S. Sheppard, P. Millman, M. Borst, T. Love, "Measuring the Psychological Complexity of Software Maintenance Tasks with the Halstead and McCabe Metrics," IEEE Transactions on Software Engineering, SE-5, No. 2, pp. 96-104; 1979.

[2.28]. S. D. Conte, H. E. Dunsmore, V. Y. Shen, Software Engineering Metrics and Models, Benjamin & Cummings, Menlo Park, California, 1986.

[2.29]. R. Troy, R. Moawad, "Assessment of Software Reliability Models," IEEE Transactions on Software Engineering, Vol. SE-11, No. 9, pp. 839-849; September, 1985.

[2.30]. W. Harrison, K. Magel, R. Kluczny, A. DeKock, "Applying Software Complexity Metrics to Program Maintenance," Computer, pp. 65-79; September, 1982.

[2.31]. M. Halstead, "Natural Laws Controlling Algorithm Structure," ACM Sigplan Notices, Vol. 7, No. 2, pp. 19-26; 1972.

[2.32]. K. S. O'Mara, C. Collins, T. Radhakrishnan, W. M. Jaworski "Halstead's Length Metric Applied to Hierarchical Components of Tabularly Structured Programs," Manuscript submitted to Software Practice and Experience, 1991.

[2.33]. Software Engineering Standards, ANSI/IEEE Standard: IEEE Guide for Software Requirements Specifications, The Computer Society of the IEEE, Piscataway, New York, pp. 830-1984; 1987.

[2.34]. M. H. Halstead, Elements of Software Science, Elsevier, North Holland, 1977.

[2.35]. D. B. Johnston, A. M. Lister, "A Note on the Software Science Length Equation," Software: Practice and Experience, Vol. 11, pp. 875-877; 1981.

[2.36]. J. Elshoff, "An Analysis of Some Commerical Pl/1 Programs," IEEE Transactions on Software Engineering, Vol. SE-2, pp. 113-120; June 1976.

[2.37]. S. Zweben, "A Study of the Physical Structure of Algorithms," IEEE Transactions Software Engineering, Vol. SE-3, No. 3, pp. 250-258; May 1977.

[2.38]. J. Lassez, D. Van Der Knijff, J. Shepherd, "A Critical Examination of Software Science," Journal of System Software, Vol. 2, pp. 105-112; December 1981.

[2.39]. V. Shen, S. Conte, H. Dunsmore, "Software Science Revisited: A Critical Analysis of the Theory and Its Empirical Support," IEEE Transactions on Software Engineering, Vol. SE-9, No. 2, pp. 155-165; March 1983.

[2.40]. A. Fitzsimmons, T. Love, "A Review and Evaluation of Software Science," ACM Computing Surveys, Vol. 10, 1, pp. 3-18; 1978.

[2.41]. N. F. Salt, "Defining Software Science Counting Strategies," ACM Sigplan Notices, Vol. 17, 3, pp. 58-67; 1982.

[2.42]. H. Jensen, V. Vairavan, "An Experimental Study of Software Methodologies for Real-time Software," IEEE Transactions on Software Engineering, Vol. SE-11, No. 2, pp. 231-234; February 1985.

[2.43]. J. M. Stroud, "The Fine Structures of Psychological Time," H. Quastler, ed., Information Theory In Psychology, Free Press, Glencoe, Illinois, 1955.

[2.44]. G. K. Zipf, Human Behavior and the Principle of Least Effort, Addison-Wesley, Cambridge, Massachusetts, 1949.

[2.45]. N. Coulter, "Software Science and Cognitive Psychology," IEEE Transactions on Software Engineering, Vol. SE-9, No. 2, March 1983.

[2.46]. B. MacLennan, Principles of Programming Languages: Design Evaluation and Implementation, Holt, Rinehart, and Winston, New York, N. Y., 1987.

[2.47]. see 1.11

[2.48]. B. Chaudhary, H Sahasrabuddhe, "A Study in Dimensions of Psychological Complexity of Programs," Journal of Man-Machine Studies, Vol. 23, pp. 113-133; 1985.

[2.49]. K. Christensen, G. Fitsos, C. Smith, "A Perspective on Software Science," IBM Systems Journal, Vol. 20, No. 4, pp. 372-387; 1981.

[2.50]. J. Welsh, M. Sneeringer, C. Hoare, "Ambiguities and Insecurities in Pascal," Software Practice and Experience, Vol. 7, No. 6, November 1977.

[2.51]. P. W. Bridgman, The Logic of Modern Physics, MacMillan, New York, N. Y., 1927.

[2.52]. J. Arsac, "Syntactic Source to Source Transforms and Program Manipulation," CACM, Vol. 22, No. 1, pp. 43-54; 1979.

[2.53]. J. Lyons, Noam Chomsky, Viking Press, New York, N. Y., 1970.

[2.54]. R. J. Brachman, H. J. Levesque, eds., Readings in Knowledge Representation, Morgan Kaufmann, Los Altos, California, 1985.

[2.55]. J. Y. Halpern, Theoretical Aspects of Reasoning About Knowledge, Morgan Kaufmann, Los Altos, California, 1986.

[2.56]. R. Englemore, T. Morgan, eds., Blackboard Systems, Addison-Wesley, Reading Massachusetts, 1989.

[2.57]. see 1.38

[2.58]. A. Fisher, CASE Using Software Development Tools, J. Wiley and Sons, New York, N. Y., 1988.

[2.59]. V. Y. Shen, T.J. Yu, S. Thebaut, L. Paulsen, "Identifying Error Prone Software: An Empirical Study," IEEE Transactions on Software Engineering, Vol. SE-11, No. 4, pp. 317-323, April 1985.

[2.60]. G. Carlson, M. Tanenhaus, Linguistics Structure in Language Processing, Klumer Academic Press, Dordrecht, Netherlands, 1989.

[2.61]. I. J. Good, "Statistics of Language", In Encyclopedia of Information and Control, ed. by A.R. Meetham, Pergamon Press, Oxford, pp. 567-581; 1969.

[2.62]. J. Allen, Natural Language Understanding, Benjamin & Cummings, Menlo Park, California, 1987.

[2.63]. P. Becker, Recognition of Pattern Using the Frequencies of Occurrence of Binary Words, [3rd ed.], Springer-Verlag Wien, 1978.

[2.64]. N. Chomsky, Knowledge of Language: Its Nature, Origin, and Use, Praeger Press, New York, N. Y., 1986.

[2.65]. see 1.35

[2.66]. N. Chomsky, "Deep Structure, Surface Structure, and Semantic Interpretation," Semantics: An Interdisciplinary Reader in Philosophy, Linguistics, and Psychology, ed. by D. Steinberg, L. A. Jakobovits, Cambridge University Press. Massachusetts, 1971.

[2.67]. N. Chomsky, Aspects of the Theory of Syntax, MIT press, Cambridge, Massachusetts, 1965.

[2.68]. G. Pullum, "On Two Recent Attempts to Show that English is Not a CFL," Journal of Computational Linguistics, Vol. 10, No. 3-4, pp. 182-186; July 1984.

[2.69]. see 1.3

[2.70]. G. Spencer-Brown, Laws of Form, E. P. Dutton, New York, N. Y., 1969.

[2.71]. D. Thompson, On Growth and Form, J. T. Bonner, ed., Cambridge University Press, Cambridge, England,1961.

[2.72]. E. Lucie-Smith, Late Modern: The Visual Arts Since 1945, Oxford U. Press, New York, 1969.

[2.73]. J. L. Phillips, The Origins of Intellect: Piaget's Theory, W. H. Freeman and Company, San Francisco, 1969.

[2.74]. J. Bruner, J. Goodnow, G. Austin, A Study of Thinking, J. Wiley and Sons, 1956.

[2.75]. N. Chomsky, Essays on Form and Interpretation, Amsterdam, North Holland, 1977.

[2.76]. M. Minsky, "A Framework for Representing Knowledge," The Psychology of Computer Vision, P. Winston, ed., McGraw-Hill, New York, N. Y., 1975.

[2.77]. M. Minsky, S Papert, Perceptons, [Expanded ed.], MIT Press, Cambridge, Massachusetts, 1988.

[2.78]. R.M. Haralick, Pictorial Data Analysis, Springer-Verlag, Berlin, 1983.

[2.79]. M. Pavel, Fundamentals of Pattern Recognition, Marcel-Dekker, Inc., New York, N. Y., 1989.

[2.80]. D. Morrison, "PATRACIA-Practical Algorithm to Retrieve Information Code in Alphanumeric," Journal of ACM, Vol. 15, No. 4, pp.514-534; October 1968.

[2.81]. M. King, Parsing Natural Language, Academic Press, New York, N. Y., 1983.

[2.82]. see 2.25

[2.83]. W. Savitch, et al, eds., The Formal Complexity of Natural Languages, D. Reidel, Dordrecht, Netherland, 1987.

[2.84]. R. Gonzalez, P. Wintz, Digital Image Processing, Addison-Wesley, Reading Massachusetts, 1977.

[2.85]. J. P. de Valk, On the Evaluation of Medical Images, PhD. Thesis, University of Nijmegen, Eindhoven, Netherland, 1983.

[2.86]. J. Kittler, K. S. Fu, L. F. Pau, eds., Pattern Recognition Theory and Applications, D. Reidel, Dordrecht, Netherland, 1982.

[2.87]. D. L. Waltz, "Understanding Line Drawings of Scenes with Shadows," The Psychology of Computer Vision, ed. by P. Winston, McGraw-Hill, New York, N. Y., 1975

[2.88]. D. Marr, Vision: A Computational Investigation into the Human Representation and Processing of Visual Information, W. H. Freeman and Company, San Francisco, California, 1982.

[2.89]. R. D. Fennell, V. R. Lesser, "Parallelism in Artificial Intelligence Problem Solving: A Case Study of Hearsay-II," Tutorial on Parallel Processing, IEEE Computer Society, New York, N. Y., pp. 185-193; 1981.

[2.90]. A Waibel, H. Sawai, K. Shikano, "Consonant and Phoneme Recognition by Modular Construction of Large Phonemic Time-delay Neural Networks," IEEE International Conference on Acoustics, Speech, and Signal Precessing, 1989.

[2.91]. J. Hopcroft, J. Ullman, Introduction to Automata Theory Languages and Computation, Addison-Wesley, 1979.

[2.92]. see 1.16

[2.93]. see 1.41

[2.94]. M. Minsky, Personal Communication, 1990.

[2.95]. N. Chomsky, Reflections on Language, Pantheon Books, New York, N. Y., 1975.

[2.96]. D. Langendoen, P. M. Terence, P. M. Postal, The Vastness of Natural Languages, Blackwell, Oxford, England, 1984.

[2.97]. D. Langendoen, P. M. Terence, P. M. Postal, "English and the Class of Context-Free Languages," Computational Linguistics, Vol. 10, pp. 177-181; 1985.

[2.98]. B. Grosz, K. Jones, B. Webber, eds., Readings in Natural Language Processing, Morgan Kaufmann, Los Altos, California, 1986.

[2.99]. D. Card, W. Agresti, "Resolving the Software Science Anomaly," Journal of Systems and Software, Vol. 7, pp. 29-35; 1987.

[2.100]. D. Davcev, "Some New Observations about Software Science Indicators for Estimating Software Quality," Information Processing and Management, Vol. 20, No. 1, pp. 245-247; 1984.

[2.101]. A. Perlis, F. Sayward, M. Shaw, Software Metrics: An Analysis Evaluation, MIT Press, Cambridge, Massachusetts, 1981.

[2.102]. I. Vessey, R. Weber, "Some Factors Affecting Program Repair Maintenance: An Empirical Study," CACM, Vol. 26, No. 2, pp. 128-134; February 1983.

[2.103]. I. Vessey, R. Weber, "Structured Tools and Conditional Logic: An Empirical Investigation," CACM, Vol. 29, pp. 48-57; 1986.

[2.104]. B. Curtis, I. Forman, R. Brooks, E. Soloway, K. Ehrlich, "Psychological Perspectives for Software Science," Information Processing and Management, Vol. 20, No. 1, pp. 81-96; 1984.

[2.105]. M. Weiser, "Programmers Use Slices When Debugging," CACM, Vol. 25, No. 7, pp. 446-452; 1982.

[2.106]. see 2.62

[2.107]. R. C. Schank, J. P. Abelson, Scripts, Plans, Goals, and Understanding, Erlbaum, Hillsdale, New Jersey, 1977.

[2.108]. R. F. Simmons, "Semantic Networks: Their Computation and Use for Understanding English Sentences," Computer Models of Thought and Language, ed. by R. C. Schank, K. M. Colby, Freeman Press, San Francisco, California, 1973.

[2.109]. J. Slocum, Machine Translation Systems, Cambridge University Press, Cambridge, England, 1988.

[2.110]. T. Winograd, Language as a Cognitive Process: Syntax, Addison-Wesley, Reading, Massachusetts, 1983.

[2.111]. Y. A. Wilks, Grammar: Meaning and the Machine Analysis of Language, Routledge and Kegan Paul, London, England, 1972.

[2.112]. D. D. Donald, L. Bolc, Natural Language Generation Systems, Springer-Verlag, New York, N. Y., 1988.

[2.113]. M. P. Marcus, A Theory of Syntactic Recognition for Natural Language, MIT Press, Cambridge, Massachusetts, 1980.

[2.114]. R. P. Lippmann, "Review of Research on Neural Nets for Speech," Neural Computation, Vol. 1, 1, 1989.

[2.115]. A. Newell, H. A. Simon, Human Problem Solving, Prentice-Hall, Englewood Cliffs, New Jersey, 1972.

[2.116]. I. Prigogine, From Being to Becoming: Time and Complexity in the Physical Sciences, W. H. Freeman Press, San Francisco, California, 1980.

[2.117]. J. Pfeiffer, The Creative Explosion: An Inquiry into the Origins of Art and Religion, Cornell University Press, Ithaca, New York, N. Y., 1982.

[2.118]. T. Pavlidis, Computer Orientated Approaches to Pattern Recognition, Springer-Verlag, Berlin, Germany, 1977.

[2.119]. K. S. Fu, Digital Pattern Recognition, Springer-Verlag, Berlin, Germany, 1976.

[2.120]. J. Tou, R. Gonzalez, Pattern Recognition Principles, Addison-Wesley, Reading, Massachusetts, 1974.

[2.121]. E. Wilson, Sociobiology: The New Synthesis, The Belknap Press of Harvard, Cambridge, Massachusetts, 1974.

[2.122]. see 2.79

[2.123]. W. Meisel, Structural Pattern Recognition, Academic Press, New York, N. Y., 1972.

[2.124]. P. Kolers, M. Eden, eds., Recognizing Patterns, MIT Press, Cambridge, Massachusetts, 1968.

[2.125]. see 1.38

[2.126]. P. Devilver, J. Kittler, Pattern Recognition: A Statistical Approach, Englewood Cliffs, New Jersey, 1982.

[2.127]. J. Kittler, M. Duff, Image Processing Systems Architectures, Research Studies Press Limited, Letchworth, Hertforshire, England, 1985.

[2.128]. R. Sokal, P. Sneath, Numerical Taxonomy: The Principles and Practice of Numerical Classification, W. H. Freeman, San Francisco, 1973.

[2.129]. J. Kandel, Fuzzy Techniques in Pattern Recognition, J. Wiley and Sons, New York, N. Y., 1982.

[2.130]. G. Lasker, ed.,Fuzzy Sets and Fuzzy Systems Possibility Theory and Special Topics in Systems Research, Applied Systems and Cybernetics, Vol. 7, Pergamon Press, New York, N. Y., 1981.

[2. 131]. A. L. Winblad, S. D. Edwards, D. R. King, Object-Oriented Software, Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1990.

[2.132]. see 1.60

[2.133]. Z. A. Melzak, Bypasses: A Simple Approach to Complexity, J. Wiley and Sons, New York, N. Y., 1983.

[2.134] see 2.119

CHAPTER THREE
MATERIALS & METHODS

3.1 INTRODUCTION

In most dictionaries the set of words found in the language is ordered alphabetically. For the work described in this thesis a dictionary in which the words or types were ordered by length or size was required. The task of re-ordering the set of words contained in an alphabetically sorted dictionary to form a dictionary ordered by word length is formidable. However, with the exception of: 1-, 2-, and 3-letter words, it is simpler to re-order the entire dictionary than to confirm the existence, in a college level dictionary, of every possible word of a given size. Mathematically there are 26^4 or 456,976 possible 4-letter words in the English language while a typical college dictionary such as Funk & Wagnalls Standard College Dictionary, FW [3.1] contains only about 152,000 entries. It is thus a simpler task to sort the entire FW dictionary than to look up all possible 4-letter words contained within it.

The principle database used in this study was the set of words listed as parts of speech in the Oxford Paperback Dictionary, OPD, [3.2], although other sources such as Funk & Wagnalls Standard College Dictionary, [3.1] and the Oxford English Dictionary, OED, [3.3] were used in Chapter 8. These alphabetically ordered lists were subsequently sorted by word-length. These alphabetically sorted lists of words of various lengths were then further analyzed in order to isolate their common structural similarities and characteristics. For our purpose, in this thesis, we consider a dictionary to be a set of types or unique words found in the language.

The format or structure of the dictionary's entries is a relatively simple matter to establish. Once a consensus is reached by the editors of a dictionary about its intended content, scope, and audience. Standardization of format can make a dictionary easier to use. Standardization can also improve a dictionary's content and reliability. In most cases the format of an entry in a dictionary usually conforms to a well-defined structure which can be used to build pattern

recognition procedures needed to automatically isolate word entries from the text of the dictionary.

3.2 DICTIONARIES, LEXICONS & WORD-LISTS

A dictionary is more than a personal word-list, while it is true that the earliest dictionaries were little more than idiosyncratic note-lists [3.4], a modern dictionary is a very scholarly work compiled under academic review. Such standards are needed in order to assure a dictionary's uniformity, quality, as well as the accuracy with which its entries reflect received English. Conservative academic review may be used to assure that transient terms are not embedded in the dictionary. The correct spelling and pronunciation of the word entries in a modern dictionary have also undergone very careful peer review. Unfortunately peer review also typically introduces a degree of explicit as well as implicit censorship.

Many of the words which have been censored from most dictionaries are very well known and popular terms. In fact, most of us would not need to consult dictionaries like the OPD, for the correct spelling or meaning of these censored words. Actually many of our greatest authors use offensive words to great effect. It is exactly such authors, who usually coin the words which are eventually accepted into the OED.

Any censorship of a lexicon does however reduce its comprehensiveness as well as its scholarly merit. It can be argued that a censored version of the OPD is suited for use in primary and secondary schools, however a comprehensive uncensored version is also needed for scholarly work.

In addition to a consensus on the principles of censorship the editors of a dictionary must determine just what information should be contained in their dictionary. Some dictionaries contain capitalized abbreviations such as CDN, WI, CPU, AFL/CIO and MUXES as well as abbreviations such as Ms. and log. Other dictionaries contain entries for the names of mythical, historical or famous figures such as Zeus, Christ and Liberace. Some dictionaries even have entries for dates such as 1066, 1776, 79, and July 1st. Clearly things can get quickly

out of hand once one begins to include more than 'words' in a dictionary. Such 'enhanced' dictionaries often become 'poor' encyclopaedias both in the literal and practical sense of the term.

If one accepts the view that a 'proper' dictionary should only contain 'the parts of speech of a language' then all of its entries are by definition valid English words. (assuming of course that one does not admit abbreviations as a part of speech!) Thus each entry in such 'proper' dictionaries lists at least the correct spelling and part of speech of its entries as well as a definition of the word in question. Indeed, if the word is a noun then perhaps a picture or a line drawing, of the object may be included. Some dictionaries include additional information such as the word's received phonetic transcription, variant spellings and contextual examples, of the word's use, are often found in 'larger' dictionaries. More comprehensive dictionaries provide historical information such as when and where the word was first coined as well as listing the word's roots, synonyms and antonyms. Of course a great many words with identical spellings have more than one meaning. For instance many words may be used both as verbs or nouns or some other part of speech. In some cases a word may assume more than one part of speech and may be pronounced differently in accordance with its different meanings or uses.

3.3 SORT KEYS: HOW TO ACCESS A DICTIONARY'S ENTRIES

Almost all modern dictionaries are sorted alphabetically. The decision to collate the word entries in a dictionary on the basis of their spelling reflects the primary use of the modern dictionary. One usually looks up a dictionary entry to verify its spelling, meaning or pronunciation.

Early dictionaries were not necessarily sorted alphabetically [3.5]. For instance dictionaries were often constructed so that their entries were collated on the basis of the frequency with which a word was used [3.6]. Other dictionaries were constructed with collating sequences which were based on the frequency with which its entries were typically misspelled [3.7]. Still other dictionaries are collated on the basis of their entries' phonetic form. Special purpose

dictionaries, such as crossword-puzzle dictionaries, first sort their entries on the basis of word-length. A secondary key is then used to further sort these lists on the basis of some other feature such as their spelling. T' 're are, of course, dictionaries of verbs, nouns and other parts of speech.

Obviously the primary use of a dictionary effects the decision as to how to best collate its entries. One could sort and cluster the entries of a dictionary by a common semantic feature or purpose such as color, type of tool, birds of prey, or geological processes. In fact by overlaying a classification and clustering scheme onto a dictionary's entries we would effectively turn the dictionary into a crude encyclopedia. In some sense the inclusion of a word entry's history in a dictionary, such as the OED, extends its use. The OED specifies the origins of ancient words as well as their role as a dead, archaic, or a presently active part of the living lexicon. In the case of traceable or more recently coined words some dictionaries actually cite the author and passage in which the word was first coined.

3.4 FUNK & WAGNALLS STANDARD COLLEGE DICTIONARY

The principle reason for choosing the FW dictionary is that this source has a companion lexicon [3.8], CFW, in which entries are listed first by length and then alphabetically. The CFW was produced for the benefit of cross-word puzzle aficionados. It lists all 2-, 3-, 4-, 5-, and 6-letter entries contained in the FW dictionary. These listings have been censored, by the dictionary's editors, to remove all phrases and words which were deemed obscene, scatological, or racially offensive. However, the CFW contains: foreign words as well as American English; standard, informal, and slang words in every part of speech and every tense, number, and gender; variant spellings, abbreviations, acronyms, hyphenated words, and prefixes and suffixes; geographical names and the first and last names of people.

The CFW differs significantly from other dictionaries. It contains only the name or index of the entry described in the FW dictionary. Also for the benefit of the crossword puzzle enthusiast, it lists many permutations for each entry in the FW dictionary. In fact a given word

of length N will be found N!/ ((2 * (N-2)!)) times within the N-letter-long word list. An example of a page listing from CFW is reproduced in Figure 3.1. For the purposes of this thesis it was necessary to delete all redundant entries from the CFW listings. The FW dictionary must then be consulted for each remaining entry in the CFW listings to determine if the cited letter-string or sequence is in fact listed in the FW dictionary as a part of speech in the English language. Each dictionary entry so established is a valid English word, VEW. The lists of VEW contained in the CFW dictionary form dictionaries of 2-, 3-, 4-, 5-, and 6-letter VEW. It is from these derived dictionaries that some of the results, presented in Chapter 8, are obtained.

3.5 OXFORD ENGLISH DICTIONARY

The ultimate lexicon for the study of the English language is the Oxford English Dictionary, OED. While this dictionary is undoubtedly the most comprehensive single source listing of the English language it is neither complete nor unbiased to the British usage of the English language [3.5]. Some of the results, presented in Chapter 8, are derived from the OED. Access to the OED required the manual look-up of each word. An example of a word entry in the OED is given in Figure 3.2.

3.6 FREQUENCY DATA : LETTER AND WORD STATISTICS

While most of the work, presented in this thesis, is derived from the word-lists found in the OPD and the Oxford Spelling Dictionary, OSD, some auxiliary statistical data sources [3.9] were used to compute the expected, position-dependent letter-frequencies for words of a given length occurring in English text. This statistical data has been used in Chapter 10 primarily to help estimate the relative likelihood of an arbitrary suffix occurring in larger words such as 10-letter-long-words.

ARECAS
AVOCET

A•••C•
ABBACY
ABDUCT
ABJECT
ACKACK
ADDICT
ADDUCE
ADDUCT
ADVICE
AFFECT
AFRICA
AGENCY
ALMUCE
ALPACA
ALSACE
AMERCE
ANLACE
ANTICS
APERCU
APIECE
ARNICA
ARRACK
ASPECT
ASPICS
ATTACH
ATTACK
ATTICA
ATTICS
AVOUCH
AZTECS

A••••C
ACETIC
ACIDIC
ADIPIC
ADONIC
AEOLIC
AGAMIC
AGARIC
AGONIC
ALARIC
ALCAIC
ALTAIC
AMEBIC
AMIDIC
AMYLIC
ANEMIC
ANGLIC
ANODIC
ANOMIC
ANOXIC
AORTIC
APNEIC
ARABIC
ARCTIC
ATAVIC
ATAXIC
ATOMIC
ATONIC
AZONIC
AZOTIC

•AC•••
AACHEN
BACHED
BACHES
BACKED
BACKER
CACAOS
CACHED
CACHES
CACHET
CACHOU
CACKLE
CACTUS
DACHAS
DACHAU
DACOIT
DACRON
DACTYL
FACADE
FACERS
FACETS
FACIAL
FACIES
FACILE
FACING
FACTOR
FACULA
HACKED
HACKEE
HACKER
HACKIE
HACKLE
JACANA
JACKAL
JACKED
JACKET
JACKIE
JACKYS
JACOBS
LACHES
LACIER
LACILY
LACING
LACKED
LACKEY
LACTAM
LACTIC
LACUNA
MACACO
MACAWS
MACERS
MACING
MACKLE
MACLES
MACRON
MACULA
MACULE
PACERS
PACIFY
PACING
PACKED
PACKER
PACKET
RACEME

RACERS
RACHEL
RACHIS
RACIAL
RACIER
RACILY
RACINE
RACING
RACISM
RACIST
RACKED
RACKER
RACKET
RACOON
SACHEM
SACHET
SACKED
SACKER
SACRAL
SACRED
SACRUM
TACKED
TACKER
TACKEY
TACKLE
TACOMA
TACTIC
VACANT
VACATE
VACUUM
YACHTS

•A•C••
BAUCIS
CAECUM
CALCAR
CALCES
CALCIC
CANCAN
CANCEL
CANCER
CARCEL
CATCHY
CAUCUS
DANCED
DANCER
DANCES
FALCON
FARCED
FARCER
FARCES
FASCES
FASCIA
FAUCAL
FAUCES
FAUCET
GARCON
GASCON
GAUCHE
GAUCHO
HANCES
LANCED
LANCER
LANCES

LANCET
LASCAR
MADCAP
MANCHU
MARCEL
MARCIA
MARCOS
MARCUS
MASCON
MASCOT
NANCYS
PARCAE
PARCEL
PASCAL
PATCHY
RANCHO
RANCID
RANCOR
RASCAL
SAUCED
SAUCER
SAUCES
TALCED
TALCUM

•A••C•
BARUCH
BASICS
CALICO
CANUCK
CARACK
DARICS
GALACT
HAUNCH
JANICE
LAUNCE
LAUNCH
MACACO
MALACO
MALICE
MARACA
NAUTCH
PALACE
PANICE
PANICS
PAPACY
PAUNCH
VARICO

•A•••C
BALTIC
BALZAC
BARDIC
CALCIC
CALPAC
CAPRIC
CARPIC
FABRIC
GAELIC
GALLIC
GARLIC
IAMBIC
IATRIC
LACTIC

MANIAC
MANIOC
MANTIC
MASTIC
NASTIC
PARSEC
SAITIC
TACTIC
TAMBAC
TANNIC
TANREC
TARMAC

••AC••
ABACAS
ABACUS
ACACIA
AEACUS
APACHE
BEACHY
BEACON
BLACKS
BRACED
BRACER
BRACES
BRACHI
BRACHY
BRACTS
CHACMA
CLACKS
CRACKS
CRACKY
CRACOW
DEACON
DRACHM
ENACTS
EPACTS
EXACTA
EXACTS
FIACRE
FLACKS
FLACON
FRACAS
GLACES
GLACIS
GRACED
GRACES
GUACOS
KNACKS
LEACHY
NIACIN
ORACHS
ORACLE
PEACHY
PLACED
PLACER
PLACES
PLACET
PLACID
PLACKS
POACHY
QUACKS
REACTS
SHACKO

SHACKS
SLACKS
SMACKS
SNACKS
SPACED
SPACER
SPACES
STACIE
STACKS
STACTE
STACYS
TEACUP
TRACED
TRACER
TRACES
TRACHE
TRACHY
TRACKS
TRACTS
WHACKS
WRACKS

••A•C•
BIANCA
BLANCH
BRANCH
CHALCO
CHANCE
CHANCY
CRATCH
EPARCH
EXARCH
FIANCE
FIASCO
FRANCE
FRANCK
FRANCO
FRANCS
GLANCE
GLAUCO
GRAECO
INARCH
ISAACS
NUANCE
PLAICE
PLANCH
PLANCK
PRANCE
SCARCE
SEANCE
SEARCH
SNATCH
STANCE
STANCH
STARCH
SWATCH
THATCH
TRANCE
USANCE

••A••C
AGAMIC
AGARIC
ALARIC

6

Figure 3.1 Example of a page entry from Funk & Wagnalls Crossword Puzzle Dictionary [3.1]. An asterisk is used to represent a wildcard character. For example a block of words in column 2 on this page lists all 6-letter-long dictionary entries which end in "AC".

Reeling (rī·liŋ), *vbl. sb.*[1] [f. REEL *v.*[1] + -ING[1].]
The action of staggering, etc.

1375 BARBOUR *Bruce* XIII. 265 The king Robert be thair relyng Saw thai war neir discomfyting. 1495 *Trevisa's Barth. De P. R.* (W. de W.) v. xx. 126 The passyons of the teeth ben dyuers..brekynge, and brusynge.., relynge and wag[ging] and fallynge. *a* 1500 *Peebles to Play* ii, For reiling thair micht na man rest, For garray and for glew. *a* 1591 H. SMITH *Six Serm.* (1594) 89 As if he should say, neither the winds blowing..nor the ships reeling..should..waken him from his sleepe. 1607–12 BACON *Ess., Counsel* (Arb.) 312 They will..be full of inconstancye,..like the reeling of a drunken Man. 1664 H. MORE *Myst. Iniq.* 329 Singing and dancing and drinking and reeling were usual concomitants of all the Pagan Holy-days. 1736 E. ERSKINE *Serm.* Wks. 1871 II. 406 The Avenger of thy blood will take care of thee in public reelings. 1781 COWPER *Conversat.* 862 Though such continual zigzags in a book, Such drunken reelings, have an awkward look. 1899 *Allbutt's Syst. Med.* VII. 69 [A gait] in which there is unsteadiness, titubation, and reeling like a drunken man.

Comb. 1610 SHAKS. *Temp.* v. i. 279 Trinculo is reeling ripe: where should they Finde this grand Liquor that hath gilded 'em? 1706 E. WARD *Wooden World Diss.* (1708) 100 When he's reeling drunk ashore, he takes it for granted to be a Storm abroad.

Reeling (rī·liŋ), *vbl. sb.*[2] [f. REEL *v.*[2] + -ING[1].]
1. The action of winding on a reel.

1589 RIDER *Bibl. Schol.*, A Reeling, *alabratio.* 1603 DEKKER *Grissil* v. i, Janiculo, leave your fish-catching, and you your reeling. 1653 *Public Gen. Acts* 179 Abuses ..in the Reeling of the Yarns. 1727–41 CHAMBERS *Cycl.* s.v. *Reel*, The reel used..in the reeling or winding of silks. 1789 *Trans. Soc. Arts* VII. 143 It was..afterwards reeled off from those bobbins, and in the reeling passed through warm water 1803 W. TAYLOR in *Ann. Rev.* I. 432 The purchases [of silk] are made about the end of August when the reelings terminate. 1884 MCLAREN *Spinning* (ed. 2) 235 The processes of twisting, reeling, and scouring.

Figure 3.2 Example of an entry taken from the Oxford English Dictionary [3.3]. This dictionary defines the semantics of each word entry and cites, where possible, an example of the the word's use in each of its possible contexts. This information is presented in historical context by listing the citation date and source for each new semantic use that a word can adopt.

Most of the statistical data used in the auxiliary studies, described in Chapters 9 and 10, was extracted from two sources. The first is a book of word frequencies [3.10] that contains an extensive rank-ordered word frequency list. This source text documents a list of 86-741 different English words of types that composed a text file of 5,088,721 running words or tokens. The 86,741 word vocabulary described in this analysis [3.10] was sorted by its frequency of occurrence. This set of words is also presented [3.10] as an alphabetically-ordered dictionary that provides statistics on the absolute and relative frequency of occurrence of a word as well as measures of its dispersion over the various categories of the text file used in the analysis. The second major source of statistical data, used in Chapters 9 and 10, was a series of tables of probabilities of occurrence of characters, character-pairs, and character-triplets in English text [3.11].

3.7 OXFORD PAPERBACK DICTIONARY

The principal lexicon used for the work, presented in this thesis, is the Oxford Paperback Dictionary, OPD, and its companion text the Oxford Spelling Dictionary [3.12] This lexicon was the largest and most scholarly version of a magnetically stored dictionary available to us[1] . Its listings are almost always entries for valid English words conforming to popular British use.

A string processing routine which capitalized on the relatively rigid stylistic structure [3.13, 3.14] of the entries in the OPD was developed to parse and extract all entries which are listed as parts of speech in this dictionary. For example, Figure 3.2 depicts a word entry found in the original version of the Oxford English Dictionary , OED, while Figure 3.3 depicts an entry taken from the 1st edition of the Oxford Paperback Dictionary, OPD. Figure 3.5 provides the Backus-Naur Form, BNF, [3.15] of the frame used to describe the structure of all entries found in the OPD. The BNF given in Figure 3 5

[1] This data was made available to us for academic use by the Oxford University Press through the kind support of Dr. Robert Burchfield, CBE, the editor-in-chief of the Oxford University Dictionaries.

was used to build heuristic pattern recognition routines that isolated word entries from text in a computer readable magnetic tape file of the OPD. Figure 3.4 depicts a page taken from the Oxford Spelling Dictionary.

This lexicographically ordered word-list generated by this process was then sorted by word-size to yield, for example, a lexicographically sorted word-list of 10-letter-long words. In order to group together all words of a similar structure, each word-list was subsequently classified by a scheme presented in Chapter 4. Word-lists of a given length and classification were then submitted to further syntactic analysis using word-root data [3.16, 3.17]. These studies produced the rule-bases given in Chapter 10.

B

Baal (*pl* **Ba'alim**)
baa-lamb
Ba'al|ism
baas|skap
baba
Bab|bage
bab|bitt (alloy)
Bab|bitt (complacent business man)
Bab|bittry
babble (**bab|bling**)
bab|bler
ba'|bel (scene of confusion)
Ba'|bel (tower)
Babi
ba'|bi|roussa
Bab|ism
Bab|ist
ba'|boon
ba|bushka
baby (as *n*, *pl* **ba'|bies**, as *v*, **ba'|bies**, **ba'|bied**, **ba'|by|ing**)
ba'|by|hood
ba'|by|ish
Ba'|by|lon
Ba'|by|lon|ian
baby-sit (**baby-sat**, **baby-sitting**)
baby-sitter
bac|ca|laur|eate
bac|carat
bac|cate
Bac|chanal
Bac|chan|a|lia
Bac|chan|a|lian
Bac|chant (*pl* **Bac|chants** *or* **Bac|chan|tes**)
Bac|chante
Bac|chantic
Bac|chic
baccy
Bach
bach|elor
bach|el|or|hood
ba|cil|lary
ba|cil|li|form
ba|cil|lus (*pl* **ba|cilli**)
back (*superl* **back|most**)
back|ache
back-bench
back-bencher
back|bite (**back|bit**, **back|bit|ing**)
back|biter
back|board
back-boiler
back|bone
back|chat
back|cloth
back|comb
back-cross
back|date (**back|dat|ing**)
back|drop
backer
back-fill
back|fire (**back|fir|ing**)
back-formation
back|gam|mon
back|ground
back|hand
back|han|ded
back|hander
back|ing
back|lash
back|less
back|list
back|log
back|marker
back|most
back|pack
back-pedal (**back-pedalled**, **back-pedalling**)
back|rest
Backs (at Cambridge)
back-scattering
back|scratcher
back-seat *a*
backsheesh *use* **baksheesh**
back|side
back|sight
back|slap|ping
back|slide (**back|slid|ing**)
back|slider
back|space (**back|spa|cing**)
back|stage
back|stay
back|stitch
back|stroke
back|track
back-up *n* & *a*
back|ward *a*
back|warda|tion
back|wards *a* & *adv*
back|wash
back|water
back|woods
back woods|man (*pl* **back woods|men**)
back|yard
ba|con
Ba|con (philosopher)
Ba|co|nian
bac|teria (*pl* of **bacterium**)
bac|terial
bac|teri|cide
bac|terio|lo|gical
bac|teri|olo|gist
bac|teri|ology
bac|teri|olysis (*pl* **bac|teri|olyses**)
bac|teri|olytic
bac|terio|phage
bac|terio|stasis (*pl* **bac|terio|stases**)
bac|terio|static
bac|terium (*pl* **bac|teria**)
bad (**worse**, **worst**)
bad|dish
baddy (*pl* **bad|dies**)
bade (*past* of **bid**)
Baden
badge
badger
bad|in|age
bad|min|ton
bad-mouth *v*
Bae|deker
Baf|fin (Island)
baffle (**baf|fling**)
baffle-board
baf|fle|ment
baffle-plate
baf|fler
bag (as *v*, **bagged**, **bag|ging**)
ba|garre
ba|gasse
ba|ga|telle
ba|gel
bag|ful (*pl* **bag|fuls**)
ba|gi|gage
bag|gi|ness
baggy (**bag|gier**, **bag|gi|est**)
Bagh|dad
bag|man (*pl* **bag|men**)
bagnio (*pl* **bagnios**)
bag|pipe (as *v*, **bag|pip|ing**)
bag|piper
ba|guette
bag|wash
bag-wig
Ba|ha'i
Ba|ha'ism
Ba|ha'ist
Ba|ha'ite
Ba|ha|mas

Figure 3.3 Illustrative example taken from the Oxford Paperback Dictionary [3.2]. This example highlights the basic features of this dictionary

This dictionary features . . .

Words in large clear type	**astray** *adv* & *adj* away from the right
Phrases	path **go astray**, to be led into error or wrongdoing, (of things) to be mislaid
Countries	**Belgium** a country in Europe **Belgian** *adj* & *n*
Capital cities	**Belgrade** the capital of Yugoslavia
Meanings numbered for clarity	**coffee** *n* **1** the bean like seeds of a tropical shrub roasted and ground for making a drink **2** this drink **3** light brown colour
Compounds	**coffee bar** a place serving coffee and light refreshments from a counter
Levels of usage	**fed** see **feed** **fed up** (*informal*) discontented displeased
Counties	**Gloucestershire** a county of England
Parts of speech	**glum** *adj* (glummer glummest) sad and gloomy **glumly** *adv* **glumness** *n*
Up-to-date vocabulary	**hang-gliding** *n* the sport of being suspended in an airborne frame controlled by one's own movements **hang-glider** *n* this frame
Pronunciation (see page vii)	**haphazard** (hap**haz**erd) *adj* done or chosen at random without planning
Examples of usage	**immune** *adj* having immunity *immune from* or *against* or *to infection* etc
Comparative and superlative forms of adjectives	**lazy** *adj* (lazier laziest) **1** unwilling to work doing little work **2** showing or characterized by lack of energy *a lazy town*
Derived words	**lazily** *adv* **laziness** *n*
Other forms of verbs (see page vi)	**lend** *v* (lent lending) **1** to give or allow the use of (a thing) temporarily on the understanding that it or its equivalent will be returned
Notes on usage	**media** (**meed**ia) *pl n* see medium **the media** newspapers and broadcasting by which information is conveyed to the general public ◆ This word is the plural of *medium* and should have a plural verb e.g. *the media are* (not *is*) *influential*. It is incorrect to refer to one of these services (e.g. television) as *a media* or *the media* or to several of them as *medias*
Computer terminology	**modem** (**moh**-dem) *n* a device linking a computer system and a telephone line so that data can be transmitted at high speeds
Notes on origin	**pot**[2] *n* (*slang*) marijuana ◆ From the Mexican Spanish phrase *potacion de guava* (= drink of grief) for a drink made by soaking cannabis seed pods in wine or brandy
American States	**Texas** a State of the USA **Texan** *adj* & *n*
Abbreviations	**VDU** *abbrev* visual display unit (see visual)

Figure 3.4 Example of an entry taken from the Oxford Spelling Dictionary [3.12]. This dictionary defines derived word-root information and specifies accepted hyphenation points for words defined in the Oxford Paperback Dictionary [3.2].

```
<DICTIONARY>      ::=   <ENTRY> NEW-PARAGRAPH [<ENTRY>]*
<ENTRY>           ::=   <VEW>.
                        |<OTHER>.
<OTHER>           ::=    <COUNTRY-NAME ENTRY>.
                        |<CAPITAL-CITY-NAME  ENTRY>.
                        |<BRITISH-COUNTY-NAME  ENTRY>.
                        |<AMERICAN-STATE-NAME  ENTRY>.
                        |<ABBREVIATION  ENTRY>.
                        |<COUNTRY-NAME  ENTRY>.
<VEW>             ::=   <HEADWORD> [<PRONUNCIATION>]
                              [<CASE>]<PART-OF-SPEECH>
                              [ & [<CASE>]<PART-OF-SPEECH>]*
                              [<RELATED-WORDS>]*<DEFINITION-STUB>
                              [<EXAMPLE-OF-USE>]*
                              [ ¶ <HISTORICAL-NOTES-ON-ORIGIN>]*
                              [ ¶ <GRAMMATICAL-NOTES-ON-USAGE>]*
<DEFINITION-STUB>::=  [<DEFINITION-NUMBER> . <DEFINITION> . ]*
                              [ ≠ <PHRASE-ENTRY>]
                              [<DERIVED-WORD-ENTRY>]*.
                        |<DEFINITION> [[ ≠ ] <PHRASE-ENTRY>]
                              [<DERIVED-WORD-ENTRY>]*.
<DEFINITION>      ::=   <SENTENCE>*
                        |<PHRASE>*
<PHRASE-ENTRY>::=       <PHRASE> , [<LEVEL-OF-USE>]
                              [<PHRASE-IN-CONTEXT>, [<LEVEL-OF-USE>]]
<DERIVED-WORD-ENTRY>::=     <VEW>
.
.
.
<LEVEL-OF-USE>::=       INFORMAL
                        | SLANG
<CASE>            ::=   SINGULAR
                        | PLEURAL
.
.
```

Figure 3.5 A simplified sketch of the extended Backus Naur Form specifying the syntactic structure of the Oxford Paperback Dictionary. This structure was used to derive a heuristic text processing routine which isolated valid English word entries from the dictionary. Notation conforms to that found in [3.15].

3.8 REFERENCES

[3.1]. Funk & Wagnalls Standard College Dictionary, Canadian Ed., Fitzhenry and Whiteside Ltd., Toronto, Canada, 1978.

[3.2]. The Oxford Paperback Dictionary, complied by Hawkins, J., Oxford University Press, Oxford, England, 1979.

[3.3]. The Oxford English Dictionary (and it's supplementary volumes; A-G, 1972; H-N, 1976; O-S, 1982; S-Z, 1984), Oxford University Press, Oxford, England, 1933.

[3.4]. K. E. Murray, Caught in the Web of Words, Oxford University Press, Oxford, England, 1979.

[3.5]. S. Potter, Changing English, Andre Deutsch Ltd., London, England, 1975.

[3.6]. R. W. Burchfield, The English Language, Oxford University Press, Oxford, England, 1985.

[3.7]. C. Berlitz, Nature Tongues, Grosset and Dunlap, New York, N. Y., 1982.

[3.8]. E. I. Schwartz, L. F. Landovitz, Funk & Wagnalls Crossword Puzzle Word Finder, The Stonesong Press, Grosset and Dunlap, Inc., New York, N. Y., 1978.

[3.9]. C. Y. Suen, "N-gram Statistics for Natural Language Understanding and Text Processing," IEEE Transactions on Pattern Recognition and Machine Intelligence, PAM-1, Vol. 2, pp. 164-172; 1979.

[3.10]. see 2.9

[3.11]. G. Toussaint, R. Shingal, "Tables of Probabilities of Occurrence of Characters, Character-Pairs, and Character-Triplets in English Text," McGill University, School of Computer Science, Technical Report # SOCS 78.6; 1978.

[3.12]. The Oxford Spelling Dictionary, complied by Allen, R.E., Clarendon, Oxford, England, 1986.

[3.13]. E. Weiner, J. Hawkins, The Oxford Guide to the English Language, Oxford University Press, Oxford, England, 1984.

[3.14]. D. R. Raymond, F. Tompa, "Hypertext and the Oxford English Dictionary," Commun. ACM 31, Vol. 7, pp. 871-879; July 1988.

[3.15]. see 1.11

[3.16]. R. Clairborne, The Roots of English, Times Books, New York, N. Y., 1989.

[3.17]. D. J. Borror, Dictionary of Word Roots and Combining Forms, Mayfield, Palo Alto, California, 1971.

CHAPTER FOUR
WORD-LEVEL SYNTACTIC STRUCTURE

4.1 INTRODUCTION

Markovian production rules expressed in BNF, have been used extensively as generic representations of syntactic structure. This formalism has been applied to many practical problems through developments in syntactic pattern recognition [4.1, 4.2]. While such techniques are very powerful, syntactic pattern recognition techniques suffer from a major methodological drawback. They require that a correct *a priori* structural model of the abstraction exists. Furthermore the implementation of successful syntactic parsing routines requires not only the existence of an *a priori* structural model but also the availability of pattern recognition routines which can be used to isolate and correctly classify the model's features from raw data. Much of the work presented in this chapter has either been published [4.3] or has been submitted for publication [4.4].

4.2 VOWEL NORMAL FORM: WORD LEVEL SYNTACTIC STRUCTURE

Vowel Normal Form, VNF, is a heuristic structural feature which has been developed [4.3, 4.5] to cluster and classify words on the basis of a single hybrid feature which has orthographic, phonetic and probabilistic components.

A basic premise underlying the work presented in this paper is that a word's form may be usefully characterized by syntactic or structural features. The VNF of a word is derived by simply substituting the symbol **V** for each vowel and the symbol **C** for each consonant in a word's written form. For the purpose of the VNF classification, the set of English vowels is **V** = { a, e, i, o, u, y } and the set of consonants is **C**. For our purposes y $\in$ **V** and hence the sets **V**

and **C** are disjoint. Valid VNF strings can contain only **V** and **C** symbols and are composed by concatenating the sequence of VNF symbols that corresponds to the literal translation of the word from correctly spelled English into VNF. This classification is thus based on the inverse letter-by-letter mapping of a word from English into its corresponding VNF form. The VNF classification scheme establishes sets of homomorphic forms which are based solely on the number and relative positions of the vowels used in the word's written form. The VNF form of a word is a character string representation which maps the base twenty-six English alphabet into the base two character set { **V** }, { **C** }. There are thus 2^N possible VNF frames or structures for an N-letter-long-word.

The binary representation of word structure provided by VNF may be viewed linguistically as a syntactic frame. Alternatively VNF may be viewed as specifying an ordinal number system which may be derived by considering VNF form as specifying a binary number where **C** denotes 0 and **V** denotes 1.

A simple example of the use of Vowel Normal Form is perhaps the shortest way of describing its utility. For instance, this classification scheme would group together all 3-letter-long words composed of three vowels into the group **VVV**, while words such as { *LAT, MAT, SAT, SAW* } would be classified as elements of the set **CVC.** The word SAW is listed as a single entry in the set **CVC** in spite of its use as many different parts of speech. For instance, SAW might be used to denote the past tense of the verb, SEE. Alternatively SAW might denote the process of cutting or the object that is used to do the cutting.

As a second example consider the set **ZETA,** Z , of all 3-letter-long English words found in the OPD. These words when clustered into classes, on the basis of the number and sequence of vowels within them, specify, on the basis of VNF, eight possible disjoint sets: Z = {{**CCC**}, {**CCV**}, {**CVC**}, {**CVV**}, {**VCC**}, {**VCV**}, {**VVC**}, {**VVV**}}. Each of these sets defines a VNF word structure or syntactic frame.

The set-size or type-sum [4.7, 4.4, 4.8], of each of the syntactic subsets for words of a specified length, is defined as the number of words found listed as parts of speech in the OPD. For example the set

size or type-sum, $S|\{w\}|$, where $\{w\} \in \{Z\}$, of each of the eight syntactic classes or subsets of Z is found to be respectively {3, 41, 599, 166, 63, 34, 38, 5}. Not surprisingly there are very few triple vowel, **VVV**, combinations (5 words or 0.6%) that are considered as words in the OPD. Similarly the great majority (599 words or 67%) of English 3-letter-long words belong to the vowel normal form **CVC**. In the next section of this chapter, we will observe that for each word-length there are very few VNF classes or subsets that constitute the majority of words of a given length.

4.3 EMPIRICAL RESULTS: VNF OF THE ENGLISH LEXICON

Tables 4.1, 4.2, 4.3, 4.4 and 4.5 specify the VNF word structures or frames for all 2-, 3-, 4-, 5-, and 6-letter-long words found in the OPD. Tables 4.6, 4.7, 4.8, 4.9 and 4.10 specify the number of words (ie 'types') which are listed in the OPD that fall into each VNF word group given in Tables 4.1, 4.2, 4.3, 4.4 and 4.5. The number of words conforming to a specific structural frame or VNF form is referred to as the VNF set size, Γ. Figure 4.1, depicts the set sizes of all VNF forms for words of all lengths which were found listed in the OPD.

Figures 4.2, 4.3, 4.4, 4.5 and 4.6, depict histograms of the VNF set sizes for each of the VNF syntactic forms underlying 2-, 3-, 4-, 5-, and 6- letter-long words found in the OPD. An analysis of this data shows that for each word-length there are very few VNF classes or subsets that constitute the majority of words of a given length.

In Figure 4.4, we note that there are 1044 different 4-letter-long words with the VNF form **CVCC** listed in the OPD. The VNF form **CVCC** may be represented as the binary number 0100_2 or 4_{10}. In general we observe that not only do relatively few VNF structures account for the majority of words of a given length but that a great many VNF frames are either sparsely populated or not populated at all.

Figures 4.7, 4.8, 4.9, 4.10, 4.11, 4.12 and 4.13 depict the VNF set sizes for each of the VNF syntactic forms underlying 7-, 8-, 9-, 10-, 11-, 12- and 13-letter-long words.

The fundamental patterns underlying the VNF structures found in the English lexicon are not readily observable in Figure 4.1. In fact, while a similarity is detectable in the set size distributions found for 6-, 7-, 8-, 9-, 10-, 11-, 12- and 13-letter-long words, the nature of this similarity is not readily apparent. This similarity is more recognizable when one observes superimposed normalized VNF set size distributions. For instance, composite images may be formed from the normalized distributions that are observed in the OPD. VNF density plot distributions are normalized by scaling both their X- and Y- axes to the same linear dimensions. This procedure assures that when superimposed the spans of the distributions are equal.

The normalization of the X- axis in the superimposed images requires that the distance between adjacent points on the N-letter-long-word VNF line segment is half that used for depicting (N + 1)-letter-long-words. This scaling or doubling effect is fundamental to the superposition of the observed VNF distributions for N-letter-long-words and (N + 1)-letter-long-words.

The empirical effects described in this chapter may be observed more clearly once the effects of low density VNF sets have been deleted from superimposed images. This process can be expanded over a wider range by simply superimposing the distributions of all 6-, 7-, 8-, 9-, 10-, 11-, 12-, and 13-letter-long-words found in the OPD. This normalized image indicates that the fundamental VNF types found for words of a given length propagate throughout most of the VNF line segment.

Figure 4.14 depicts the composite image produced by superimposing the normalized VNF distributions 6-, 7-, 8-, 9-, 10-, 11-, 12-, and 13-letter-long-words found in the OPD. This Figure illustrates that the patterns observed between 6- and 7-letter-long-words are common to the vast majority of word-lengths found in the OPD. The principle peaks found are, in effect, fundamental word structures. These peaks are common to the structure found. In fact Figure 4.15, which is the filtered superimposed image of the filtered normalized density plot given in Figure 4.14, demonstrates that very few fundamental forms are basic to all 6-, 7-, 8-, 9-, 10-, 11-, 12-, and 13-letter-long-words found in the OPD.

A simple prefix code model of English language word structure is developed in Chapter 5 to account for these effects.

4.4 NOTATION: SUB-CLASS OF VNF

The numerical representation and labeling of VNF structures enhances the ease of scale transformations. Such numerical representations simplify the development of schemes suitable for clustering word structures based on distance measures in VNF space [4.9, 4.10]. All VNF classes are represented in this thesis are either base 2 structures such as **VCV**: (**VCV** = 101_2) or as base 10 number such as 5_{10} ($101_2 = 5_{10}$)

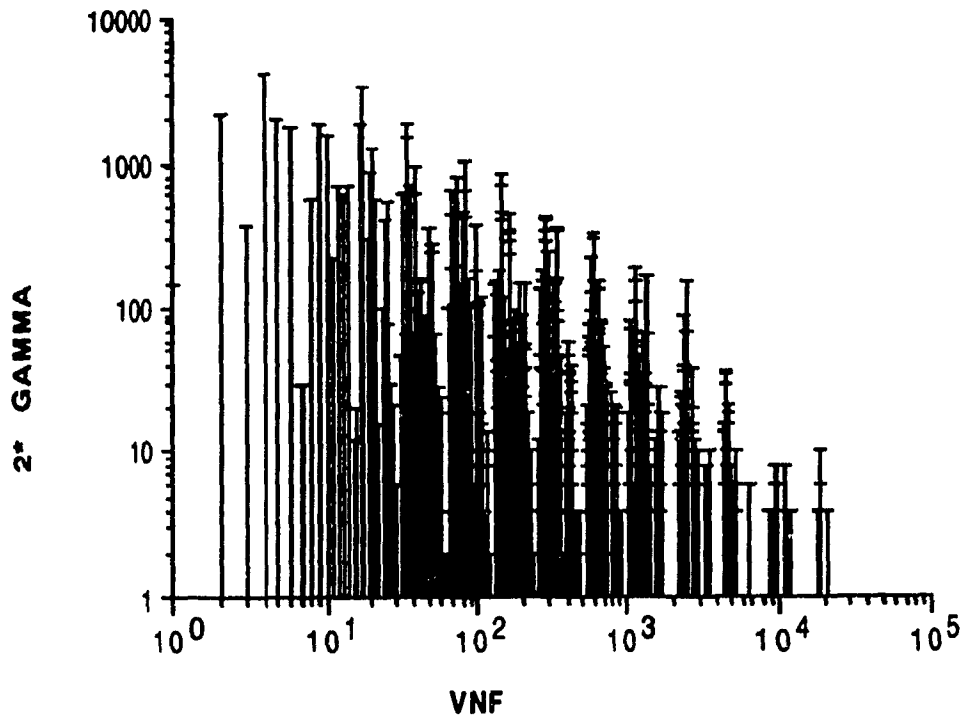

Figure 4.1 VNF density plot of all valid English words defined in the OPD [4.11] which are shorter than 12-letters-long. Abscissa VNF class or structure specified as a base 10 number. Ordinate set size in words.

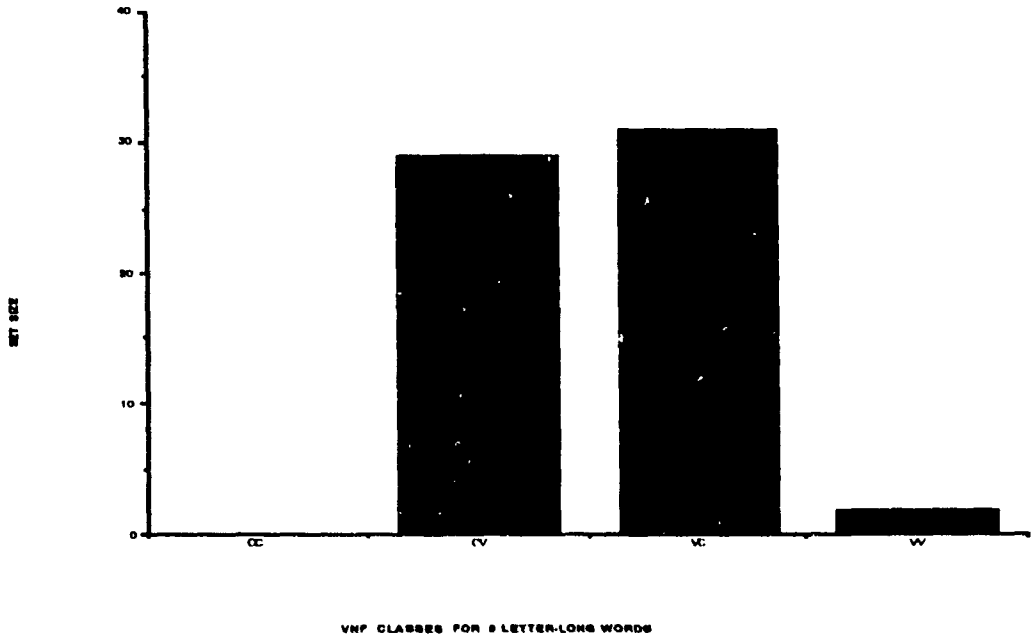

Figure 4.2 VNF density plot of all 2-letter-long valid English words defined in the OPD [4.11]. Abscissa VNF class or structure specified as a base 10 number. Ordinate set size in words

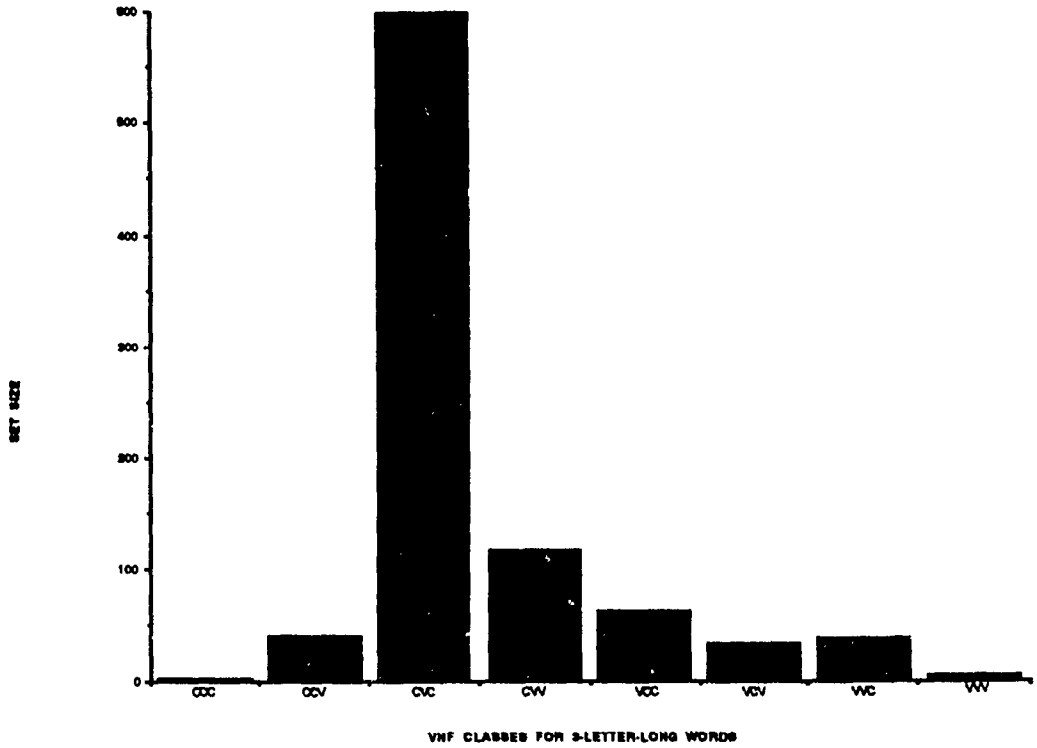

Figure 4.3 VNF density plot of all 3-letter-long valid English words defined in the OPD [4.11]. Abscissa VNF class or structure specified as a base 10 number. Ordinate set size in words

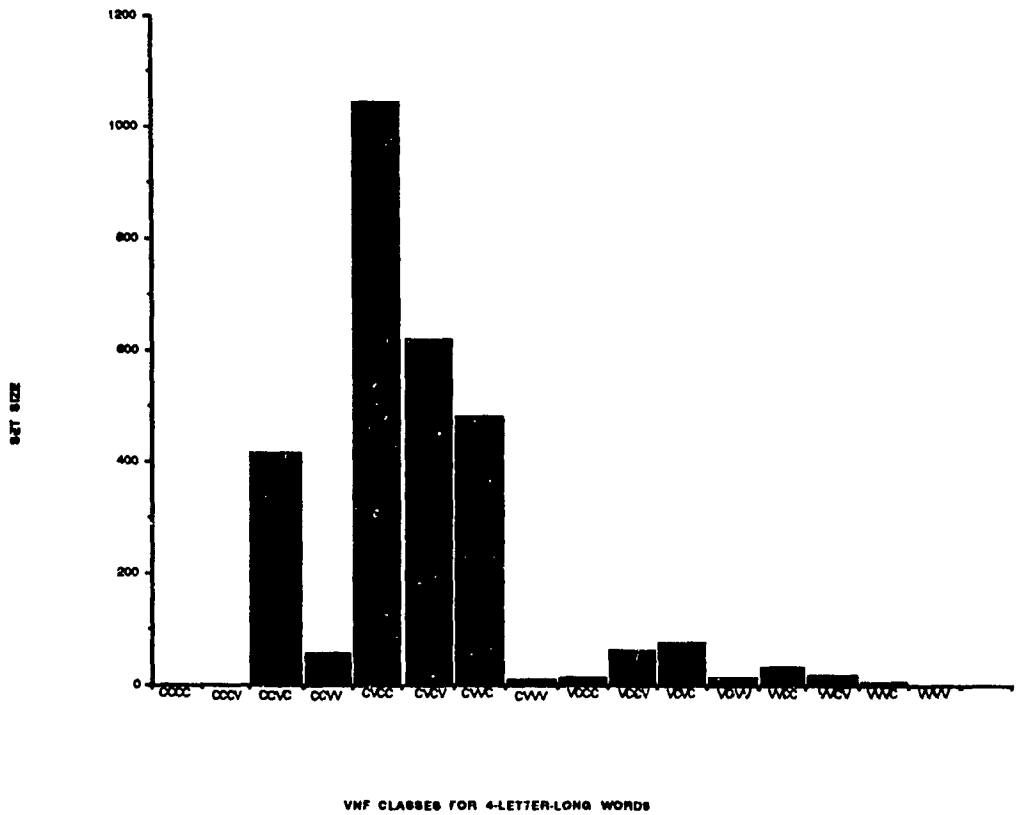

Figure 4.4 VNF density plot of all 4-letter-long valid English words defined in the OPD [4.11]. Abscissa VNF class or structure specified as a base 10 number. Ordinate set size in words

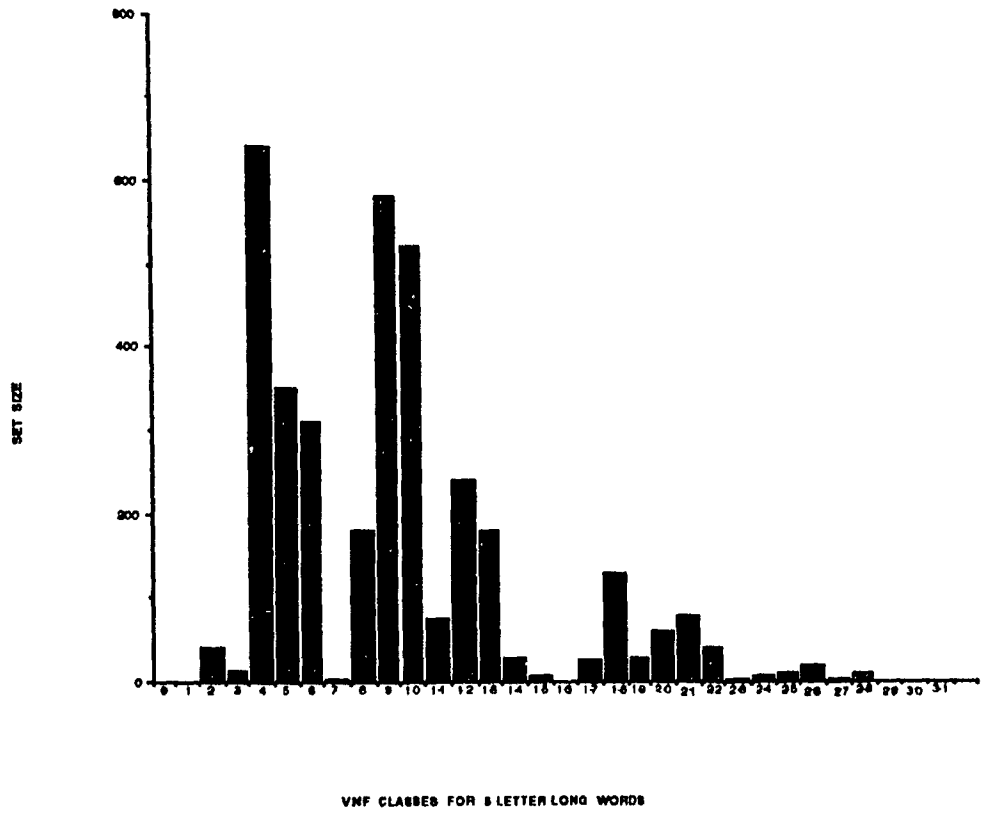

Figure 4.5 VNF density plot of all 5-letter-long valid English words defined in the OPD [4.11]. Abscissa VNF class or structure specified as a base 10 number. Ordinate set size in words

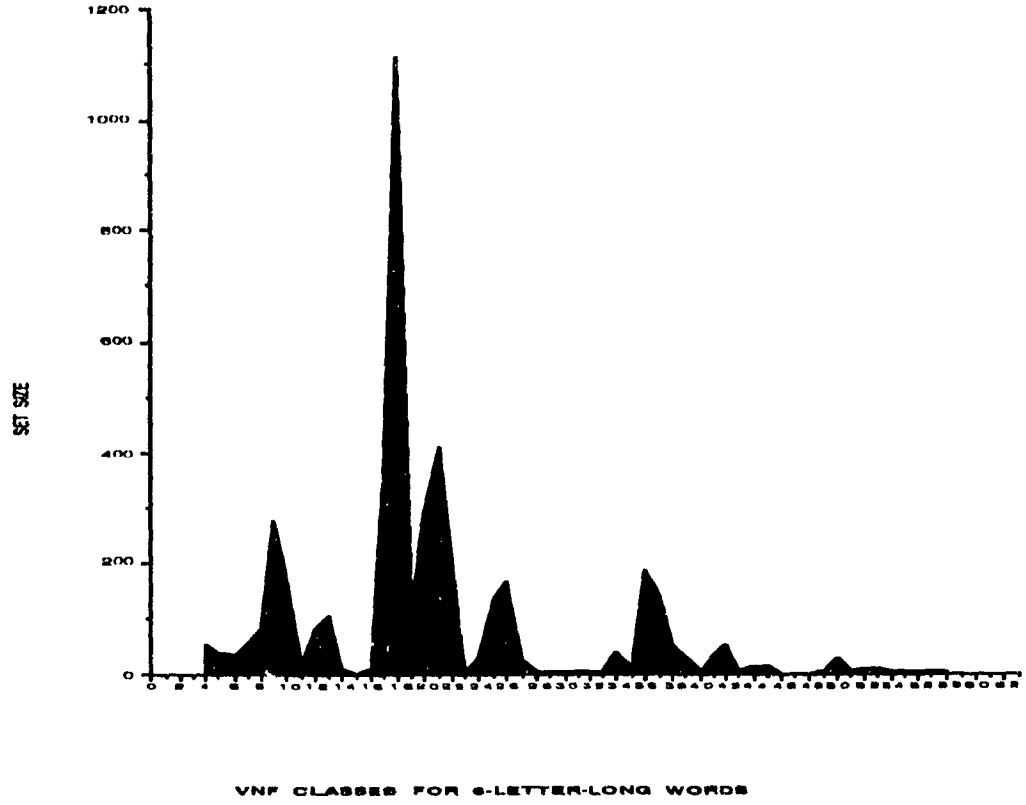

Figure 4.6 VNF density plot of all 6-letter-long valid English words defined in the OPD [4.11]. Abscissa VNF class or structure specified as a base 10 number. Ordinate set size in words

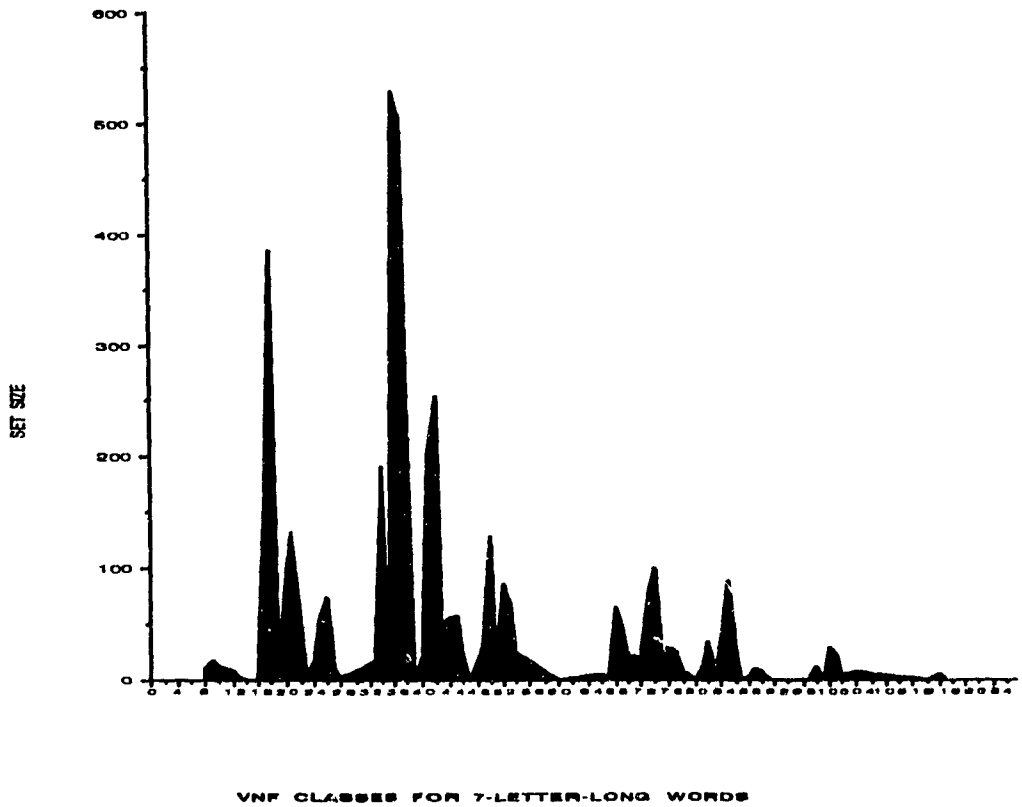

Figure 4.7 VNF density plot of all 7-letter-long valid English words defined in the OPD [4.11]. Abscissa VNF class or structure specified as a base 10 number. Ordinate set size in words

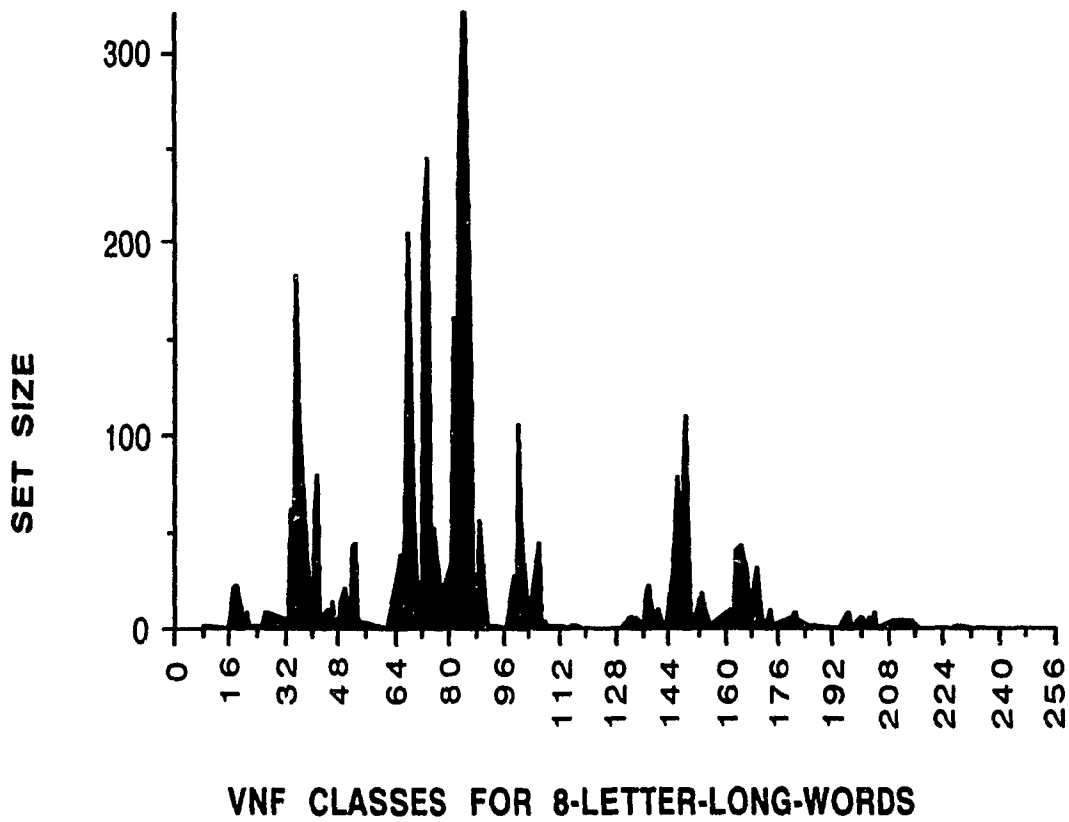

Figure 4.8 VNF density plot of all 8-letter-long valid English words defined in the OPD [4.11]. Abscissa VNF class or structure specified as a base 10 number. Ordinate set size in words

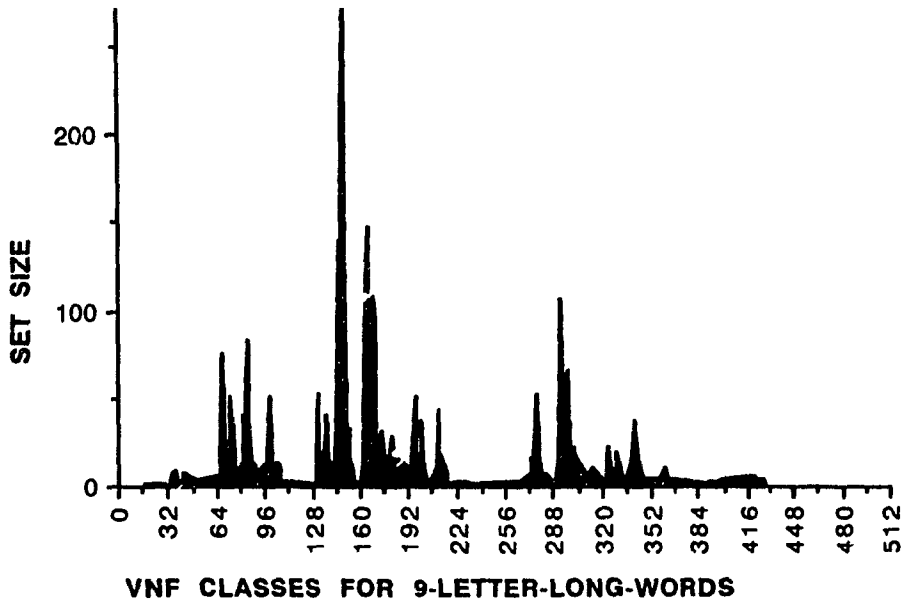

FIGURE 4.9 VNF density plot of all 9-letter-long valid English words defined in the OPD [4.11]. Abscissa VNF class or structure specified as a base 10 number. Ordinate set size in words

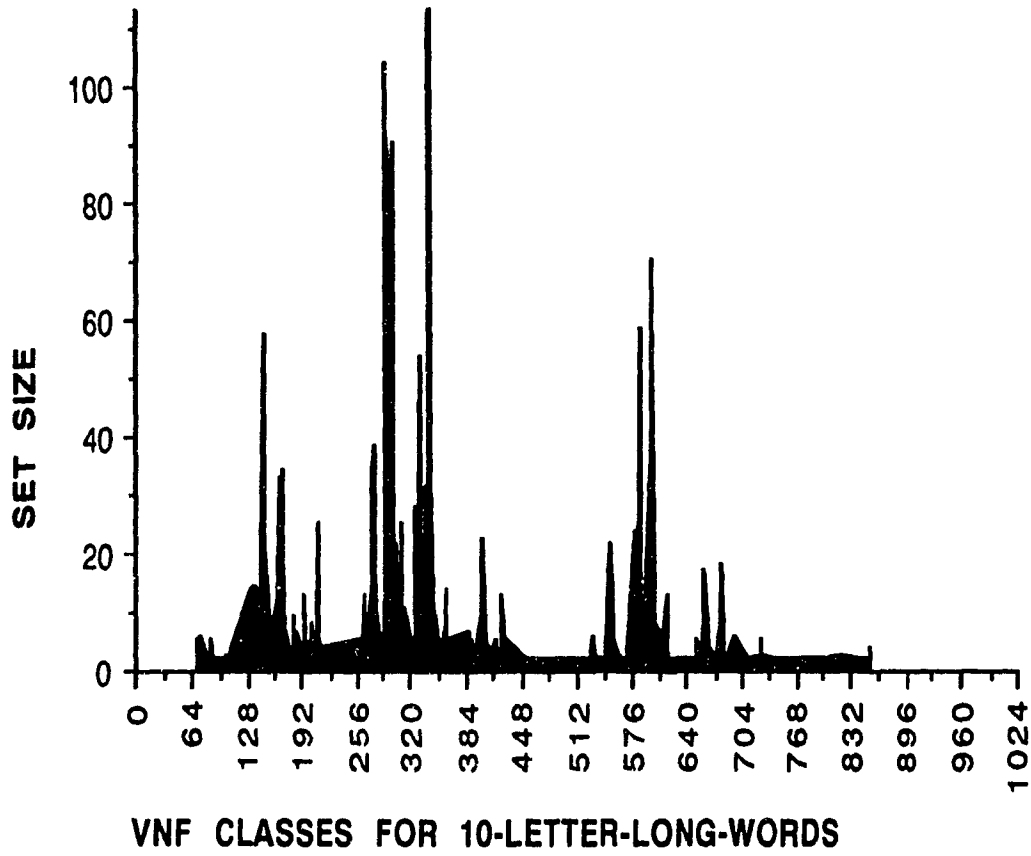

Figure 4.10 VNF density plot of all 10-letter-long valid English words defined in the OPD [4.11]. Abscissa VNF class or structure specified as a base 10 number. Ordinate set size in words

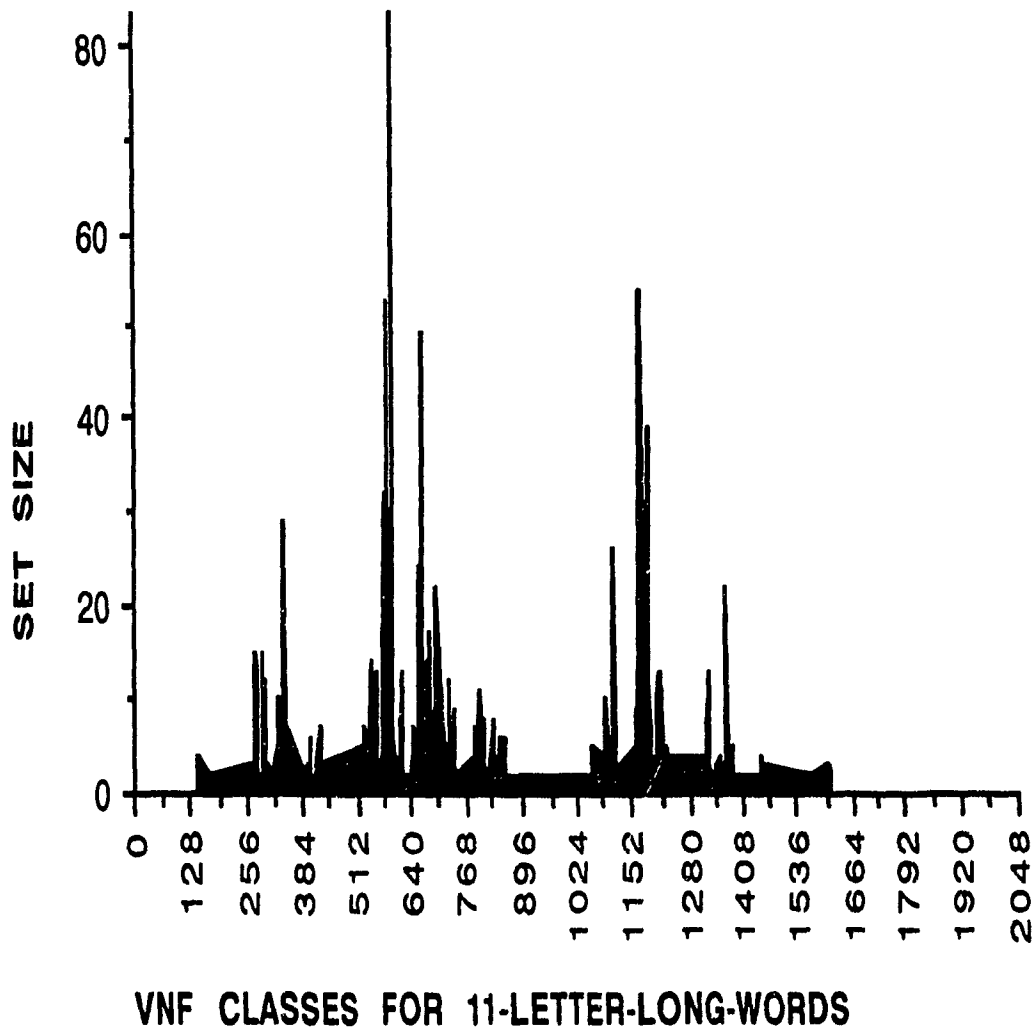

Figure 4.11 VNF density plot of all 11-letter-long valid English words defined in the OPD [4.11]. Abscissa VNF class or structure specified as a base 10 number. Ordinate set size in words

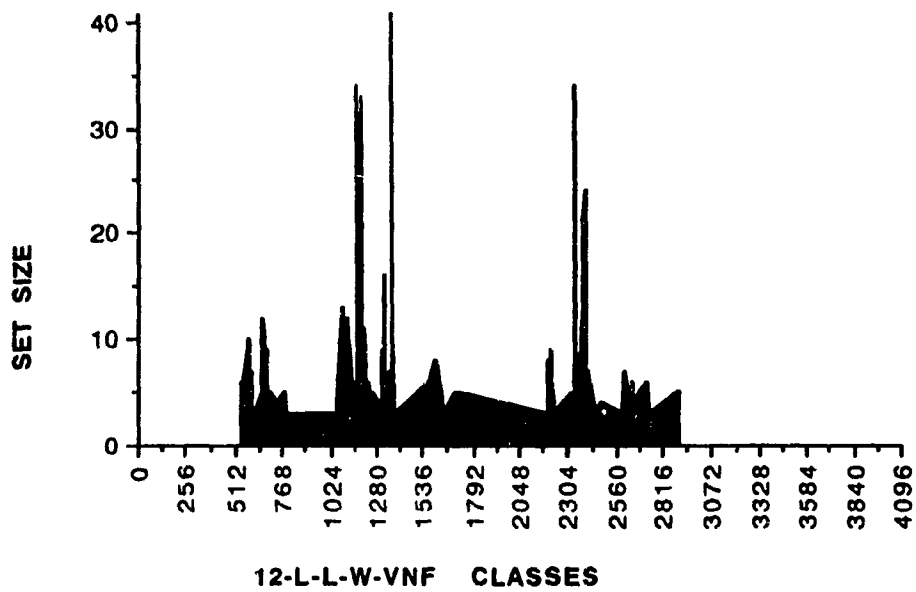

Figure 4.12 VNF density plot of all 12-letter-long valid English words defined in the OPD [4.11]. Abscissa VNF class or structure specified as a base 10 number. Ordinate set size in words.

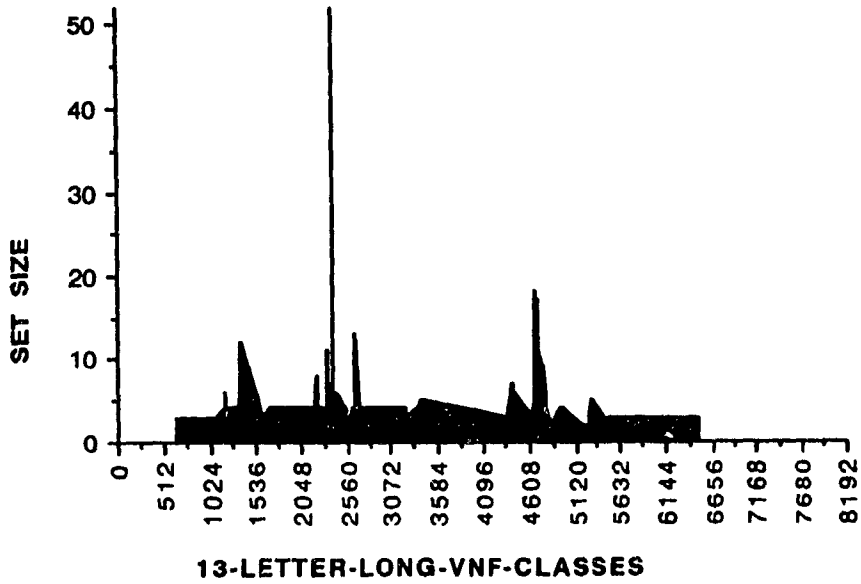

Figure 4.13 VNF density plot of all 13-letter-long valid English words defined in the OPD [4.11]. Abscissa VNF class or structure specified as a base 10 number. Ordinate set size in words

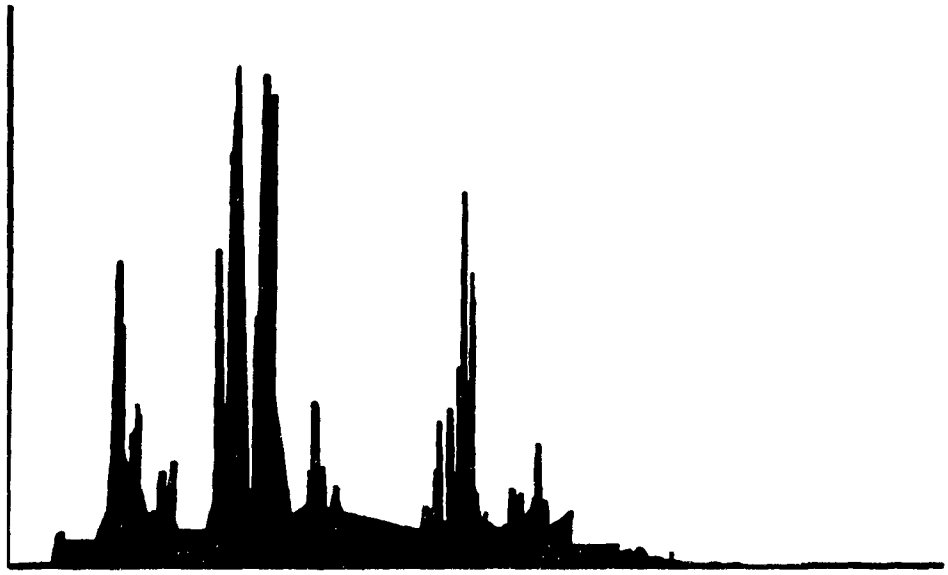

Figure 4.14 Superimposed image of the Normalized 6-, 7-, 8-, 9-, 10-, 11-, 12- and 13-letter-long VNF density plots depicted in Figures 4.15, 4.17, 4.19 and 4.21.

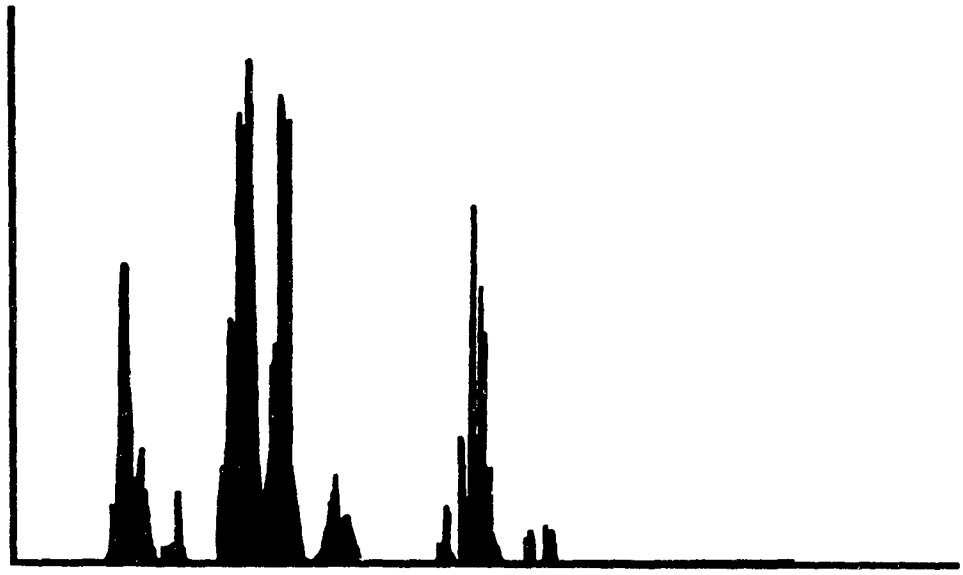

Figure 4.15 Superimposed image of the Filtered Normalized VNF Density Plots for 6-, 7-, 8-,9-,10-,11-, 12-,and 13-letter-long words depicted in Figures 4.23, 4.25, 4.27 and 4.29.

DERIVED	VNF FORM	SUFFIX
V + SUFFIX	C + SUFFIX	VNF BASE
VV	CV	V
VC	CC	C

Table 4.1 Derived 2-letter-long Vowel Normal Form frames constructed from a 1-letter-long suffix base

DERIVED	VNF FORM	SUFFIX
V + SUFFIX	C + SUFFIX	VNF BASE
VVV	CVV	VV
VVC	CVC	VC
VCV	CCV	CV
VCC	CCC	CC

Table 4.2 Derived 3-letter-long Vowel Normal Form frames constructed from a 2-letter-long suffix base

DERIVED	VNF FORM	SUFFIX
V + SUFFIX	C + SUFFIX	VNF BASE
VVVV	CVVV	VVV
VVVC	CVVC	VVC
VVCV	CVCV	VCV
VVCC	CVCC	VCC
VCVV	CCVV	CVV
VCVC	CCVC	CVC
VCCV	CCCV	CCV
VCCC	CCCC	CCC

Table 4.3 Derived 4-letter-long Vowel Normal Form frames constructed from a 3-letter-long suffix base

DERIVED VNF FORM		SUFFIX
V + SUFFIX	C + SUFFIX	VNF BASE
VVVVV	CVVVV	**VVVV**
VVVVC	CVVVC	**VVVC**
VVVCV	CVVCV	**VVCV**
VVVCC	CVVCC	**VVCC**
VVCVV	CVCVV	**VCVV**
VVCVC	CVCVC	**VCVC**
VVCCV	CVCCV	**VCCV**
VVCCC	CVCCC	**VCCC**
VCVVV	CCVVV	**CVVV**
VCVVC	CCVVC	**CVVC**
VCVCV	CCVCV	**CVCV**
VCVCC	CCVCC	**CVCC**
VCCVV	CCCVV	**CCVV**
VCCVC	CCCVC	**CCVC**
VCCCV	CCCCV	**CCCV**
VCCCC	CCCCC	**CCCC**

Table 4.4 Derived 5-letter-long Vowel Normal Form frames constructed from their 4-letter-long suffix base.

DERIVED VNF FORM		SUFFIX
V + SUFFIX	C + SUFFIX	VNF BASE
VVVVVV	CVVVVV	**VVVVV**
VVVVVC	CVVVVC	**VVVVC**
VVVVCV	CVVVCV	**VVVCV**
VVVVCC	CVVVCC	**VVVCC**
VVVCVV	CVVCVV	**VVCVV**
VVVCVC	CVVCVC	**VVCVC**
VVVCCV	CVVCCV	**VVCCV**
VVVCCC	CVVCCC	**VVCCC**
VVCVVV	CVCVVV	**VCVVV**
VVCVVC	CVCVVC	**VCVVC**
VVCVCV	CVCVCV	**VCVCV**
VVCVCC	CVCVCC	**VCVCC**
VVCCVV	CVCCVV	**VCCVV**
VVCCVC	CVCCVC	**VCCVC**
VVCCCV	CVCCCV	**VCCCV**
VVCCCC	CVCCCC	**VCCCC**
VCVVVV	CCVVVV	**CVVVV**
VCVVVC	CCVVVC	**CVVVC**
VCVVCV	CCVVCV	**CVVCV**
VCVVCC	CCVVCC	**CVVCC**
VCVCVV	CCVCVV	**CVCVV**
VCVCVC	CCVCVC	**CVCVC**
VCVCCV	CCVCCV	**CVCCV**
VCVCCC	CCVCCC	**CVCCC**
VCCVVV	CCCVVV	**CCVVV**
VCCVVC	CCCVVC	**CCVVC**
VCCVCV	CCCVCV	**CCVCV**
VCCVCC	CCCVCC	**CCVCC**
VCCCVV	CCCCVV	**CCCVV**
VCCCVC	CCCCVC	**CCCVC**
VCCCCV	CCCCCV	**CCCCV**
VCCCCC	CCCCCC	**CCCCC**

Table 4.5 Derived 6-letter-long Vowel Normal Form frames constructed from their 5-letter-long suffix base.

DERIVED		VNF FORM	
V + SUFFIX	SET SIZE	C + SUFFIX	SET SIZE
VV	2	CV	29
VC	31	CC	0

Table 4.6 Set size of the derived 2-letter-long Vowel Normal Form frames

DERIVED		VNF FORM	
V + SUFFIX	SET SIZE	C + SUFFIX	SET SIZE
VVV	5	CVV	116
VVC	38	CVC	599
VCV	34	CCV	41
VCC	63	CCC	3

Table 4.7 Set size of the derived 3-letter-long Vowel Normal Form frames

DERIVED		VNF FORM	
V + SUFFIX	SET SIZE	C + SUFFIX	SET SIZE
VVVV	0	CVVV	8
VVVC	5	CVVC	480
VVCV	17	CVCV	619
VVCC	30	CVCC	1044
VCVV	13	CCVV	58
VCVC	77	CCVC	412
VCCV	60	CCCV	1
VCCC	12	CCCC	1

Table 4.8 Set size of the derived 4-letter-long Vowel Normal Form frames

DERIVED		VNF FORM	
V + SUFFIX	SET SIZE	C + SUFFIX	SET SIZE
VVVVV	0	CVVVV	5
VVVVC	0	CVVVC	29
VVVCV	0	CVVCV	181
VVVCC	8	CVVCC	239
VVCVV	3	CVCVV	76
VVCVC	19	CVCVC	521
VVCCV	8	CVCCV	581
VVCCC	5	CVCCC	178
VCVVV	1	CCVVV	2
VCVVC	39	CCVVC	309
VCVCV	79	CCVCV	351
VCVCC	60	CCVCC	642
VCCVV	28	CCCVV	11
VCCVC	128	CCCVC	40
VCCCV	26	CCCCV	0
VCCCC	0	CCCCC	0

Table 4.9 Set size of the derived 5-letter-long Vowel Normal Form frames

DERIVED		VNF FORM	
V + SUFFIX	SET SIZE	C + SUFFIX	SET SIZE
VVVVVV	0	CVVVVV	0
VVVVVC	0	CVVVVC	3
VVVVCV	0	CVVVCV	7
VVVVCC	0	CVVVCC	5
VVVCVV	0	CVVCVV	25
VVVCVC	4	CVVCVC	163
VVVCCV	3	CVVCCV	133
VVVCCC	0	CVVCCC	26
VVCVVV	0	CVCVVV	3
VVCVVC	2	CVCVVC	199
VVCVCV	11	CVCVCV	410
VVCVCC	9	CVCVCC	290
VVCCVV	3	CVCCVV	112
VVCCVC	28	CVCCVC	1112
VVCCCV	3	CVCCCV	341
VVCCCC	1	CVCCCC	8
VCVVVV	0	CCVVVV	1
VCVVVC	1	CCVVVC	9
VCVVCV	15	CCVVCV	105
VCVVCC	13	CCVVCC	79
VCVCVV	7	CCVCVV	19
VCVCVC	51	CCVCVC	174
VCVCCV	31	CCVCCV	275
VCVCCC	6	CCVCCC	78
VCCVVV	0	CCCVVV	0
VCCVVC	52	CCCVVC	30
VCCVCV	139	CCCVCV	35
VCCVCC	185	CCCVCC	58
VCCCVV	9	CCCCVV	0
VCCCVC	35	CCCCVC	0
VCCCCV	2	CCCCCV	0
VCCCCC	0	CCCCCC	0

Table 4.10 Set size of the derived 6-letter-long Vowel Normal Form frames

4.5 REFERENCES

[4.1]. see 2.79

[4.2]. see 1.42

[4.3]. see 1.5

[4.4]. see 1.72

[4.5]. see 1.73

[4.6]. see 1.74

[4.7]. W. N. Francis, H. Kucera, Frequency Analysis of English Usage: Lexicon and Grammar, Houghton Mifflin, Boston, 1982.

[4.8]. S. Johansson, K., Hofland, Frequency Analysis of English Vocabulary and Grammar, Clarendon Press, Oxford, 1989.

[4.9]. R. Duda, P. Hart, Pattern Classification and Scene Analysis, J. Wiley and sons, New York, N. Y., 1973.

[4.10]. see 1.7

[4.11]. see 3.2

CHAPTER FIVE
A PREFIX CODE MODEL
OF ENGLISH LANGUAGE WORD STRUCTURE

5.1 INTRODUCTION

In Chapter 4 we observed that the binary representation of word structure provided by its VNF may be viewed linguistically as a syntactic frame [5.1]. Alternatively VNF may be viewed as specifying an ordinal number system which may be derived by considering VNF form as specifying a binary number where **C** denotes 0 and **V** denotes 1. The numerical representation and labeling of VNF structure enhances the ease of scale transformations and the use of numbering schemes suitable for clustering word structures based on distance measures in VNF space [5.2]. In Chapter 4 we observed common patterns in VNF word structures used by the vast majority of words listed in the OPD. These common patterns can be simply described in terms of a prefix code model of English language word structure. The simplest restricted form of the prefix code is sufficient to describe our results [5.1, 5.2, 5.3]. We inferred a propagation effect from our empirical analysis, wherein each n-digit VNF structure forms two (n + 1)-digit VNF structures. If x is the location of the n-digit VNF structure on the VNF line segment, then $2x$ and $2x + 1$ are the locations of the two (n + 1)-digit structures derived from x . Much of the work presented in this chapter has been submitted for publication [5.1, 5.2].

5.2 METHODS

To see if a model is more than simply descriptive it is often informative to use it in a predictive manner. In this instance it would be useful to see the degree to which the Prefix Code Model of English language word structure is predictive.

Such predictive behavior can be assessed at both the macroscopic and the microscopic level. For instance at a microscopic level the model could be used to predict the set size of an arbitrary VNF group. Macroscopic predictions would focus on predicting the overall presence and size of band-filtered effects in English language word structure.

The robustness of a model is determined by the degree to which macroscopic effects are not seriously effected by perturbations in the model's parameters. It is possible to apply such perturbation tests to the assessment of the validity of the proposed Prefix Model of English language word structure at a macroscopic level [5.2].

In this chapter we will see that perturbations can be introduced by initializing the Prefix Model with different VNF kernel sets.

5.3 THEORY

Let $*(A)$ be a function which operates on the values of a set, A. The application of $*$ to A is used to compute the values $(2 * x)$ and $(2 * x) + 1$ for each x in the set A. The elements of the set A form a kernel for this simulation. The function $*$ may be applied recursively to the set A. Using this notation the depth of the recursive application of $*$ to A is specified by N in the expression $*^N(A)$. More formally we have that $\forall N \geq 2$:

$$*^N(A)$$

$$= \left\{ \forall x : x \in \{*^{N-1}(A)\} \; \exists y \in \{*^N(A)\} \right.$$

$$\left. \mid y = 2 * x, \; y = 2 * x + 1 \right\} \qquad (5.1$$

where the set $\{*^1(A)\} = \{A\}$ (5.2

For example if the set $A = \{a, b, c\}$

then $\{*^2(A)\} = \{\forall x : x \in \{*^1(A)\}$

$$| \; y = 2 * x, \; y = 2 * x + 1 \} \qquad (5.3$$

thus $*^2(A) = \{ 2a, 2a+1, 2b, 2b+1, 2c, 2c+1 \}$ (5.4

while $*^3(A) = \{\forall x : x \in \{*^2(A)\}$

$$| \; y = 2 * x, \; y = 2 * x + 1 \} \qquad (5.5$$

$= \{ 4a, 4a+2, 4b, 4b+2, 4c, 4c+2, 4a+1,$
$4a+3, 4b+1, 4b+3, 4c+1, 4c+3 \}$ (5.6

and $*^4(A) = \{\forall x : x \in \{*^3(A)\}$

$$| \; y = 2 * x, \; y = 2 * x + 1 \} \qquad (5.7$$

$= \{ 8a, 8a+4, 8b, 8b+4, 8c, 8c+4, 8a+2, 8a+6, 8b+2,$
$8b+6, 8c+2, 8c+6, 8a+1, 8a+5, 8b+1, 8b+5,$
$8c+1, 8c+5, 8a+3, 8a+7, 8b+3, 8b+7,$
$8c+3, 8c+7 \}$ (5.8

Clearly the number of terms in the set, $*^N$ is a simple function of the size of its kernel set A. Following convention let the size of A be denoted as: $S \mid A \mid$.

$$S \mid *^N(A) \mid = 2^{(N-1)} * S \mid A \mid \qquad (5.9$$

for instance in the example given above where $S \mid A \mid = 3$ then

$$S \mid *^2(A) \mid = 2 * 3 = 6$$

$$S \mid *^{3}(A) \mid = 2^{2} * 3 = 12$$

$$S \mid *^{4}(A) \mid = 2^{3} * 3 = 24$$

The superset of terms computed in a simulation of the VNF line segment that includes $*^{1}(A)$, $*^{2}(A)$, $*^{3}(A)$,...,$*^{N}(A)$ terms may be computed by the function $\Phi^{N}(A)$. The function $\Phi^{N}(A)$ may be defined as:

$$\Phi^{N}(A) = \{\{*^{N}(A)\} \cup \{*^{N-1}(A)\} \cup \{*^{N-2}(A)\} \ldots \{*^{1}(A)\}\} \qquad (5.10$$

or as:

$$\Phi^{N}(A) = *^{N}(A) \cup \Phi^{N-1}(A) \qquad (5.11$$

where $*^{N}(A)$ may be defined recursively as:

$$*^{N}(A) = \{\forall x : x \in \{*^{N-1}(A)\} \exists y \in \{*^{N}(A)\} \mid y = 2 * x,\ y = 2 * x + 1\}$$

and $\{*^{1}(A)\} = \{A\}$

The total number of terms in a simulation of the VNF line segment which includes the frames computed in the sets $*^{1}(A)$, $*^{2}(A)$, $*^{3}(A)$,...,$*^{N}(A)$ is given by the function $\Psi(N, A)$:

$$\Psi(N, A) = \sum_{i=1}^{N} S \mid *^{i}(A) \mid = \sum_{i=1}^{N} 2^{(i-1)} * S \mid A \mid \qquad (5.12$$

N

$$= \quad S|A| * \sum_{i=1} 2^{(i-1)} \qquad (5.13$$

$$= \quad S|A| * 2^{(N)} - 1 \qquad (5.14$$

The term, $\Psi(N, A)$, may be used to compute the total number of terms in the set, $\Phi^N(A)$, for any given kernel A and any recursive depth of computation N.

All VNF line segments are bounded by 1 and κ. The numerical value of κ represents the largest VNF form computable from the application of $*^N(A)$ on a given kernel set A. κ can be computed simply as $2^{(N+M)} - 1$ where N is the depth of the recursive computation used and M is the word-length of the kernel used in A. How well packed is the integer line segment bounded by 1 and κ?

In general the term, Tau, τ, may be used to describe the gross macroscopic density of the VNF line segment. The term τ specifies a simple global measure of the degree of packing of the VNF line segment. Tau, τ, is a measure of the ratio of the number of VNF groups that are occupied to those which could be populated.

$$\tau = \frac{S|A| * 2^{(N)} - 1}{2^{(N+M)} - 1} \qquad (5.15$$

For large N Equation 5.15 may be approximated as:

$$\tau \approx S|A| \frac{2^{(N)}}{2^{(N+M)}} \qquad (5.16$$

Let the size of the kernel, S | A |, be denoted as δ then Equation 5.16 may be rewritten as:

$$\tau \approx \frac{\delta}{2^M} \qquad (5.17$$

For example if $\delta = 10$ and $M = 6$ we would expect that only approximately one sixth of the possible VNF word groups could possibly be densely populated.

5.4 RESULTS

In the last chapter we observed complex structures throughout the wide range of VNF line segments used to map 2-, ..., 12-letter-long English language word frames. The complexity of the VNF line segments increase as we observe composite structures depicting VNF structures found in words of many different lengths. Perhaps the most complex of these is the histogram depicting VNF set size as a function of set structure for all words listed in the OPD [5.6]. This histogram was presented in Figure 4.1 while a scatter plot of the same data is presented in Figure 5.1. A histogram depicting a filtered image of Figure 4.1 that shows the location of the more sparsely populated VNF frames is illustrated in Figure 5.2. The horizontal band structures found between the more-sparsely and less-sparsely populated VNF structures in this figure are consistent with those observed in other filtered images of Figure 4.1. Complex band filtered effects are observed across the wide range of VNF set sizes shown in Figures 5.3 and 5.4. A comprehensive model of the lexicon must account both for its most populated VNF frames and the band structures found for sets of any given size within a VNF line segment.

Estimates of τ derived from Figure 5.2 allow one to compute the number of terms in the kernel set A needed to simulate the dominant features of the English lexicon. Given a $\tau = 0.1$ then Equation 5.17 predicts that 6 6-letter-long VNF frames or 10 5-letter-long frames

are needed to accurately simulate the major features on the VNF line segments computed to a κ of 4095. While Equation 5.17 allows us to determine the minimum number of terms or VNF frames to include in our kernel, it does not help us to determine which structures to include. Next we will consider just how to derive which VNF structures to include in a kernel which we have determined must contain at least a specified number of elements.

Table 5.1 gives the VNF structure, as VNF_{10}, of the ten-largest VNF frames found to occur for 6-, 7-, 8-, 9-,10-, 11- and 12-letter-long words in the OPD [5.6]. The data presented in Table 5.1 is rank-ordered. For instance Row 1, the top row, of Tables 5.1 and 5.2 gives the VNF_{10} structure of the most populated word-group or frame found to occur for all words listed in the OPD. The first column in Table 5.2, which is referred to as Column 6, lists the VNF_{10} structures for the top-ten most-populated VNF_{10} frames found to occur in all 6-letter-long words listed in the OPD. Thus Column 12, Row 10, lists the tenth most densely populated VNF_{10} found to occur for all 12-letter-long words listed in the OPD.

A simplified version of Table 5.1 is depicted in Table 5.3. In this simplified table each VNF structure is symbolized by simple dots which may be connected by three different types of arrows. Case 1: A closed arrow is used to depict an n-letter-long VNF structure which was exactly computed by a prefix code model applied to a (n - 1)-letter-long frame. Case 2: An open arrow is used to depict an n-letter-long VNF structure which was approximately computed by a prefix code model of an (n - 1)-letter-long frame. The approximation used was that the VNF_{10} address had to be within a distance of 4_{10} of its predicted value. Case 3: A thin arrow connecting a dot to the table's outer edge is used to depict those n-letter-long VNF structures than could not be either exactly nor approximately computed from the top-ten VNF frames found in the (n - 1)-letter-long word frames. For our analysis of this effect we use a restricted form of prefix coding in

which an n-digit binary number x forms an $(n + 1)$-digit binary number whose numerical value is either $2x$ or $(2x + 1)$.

Analysis of the data given in Tables 5.4 for the top-ten largest VNF frames found for all 6-, 7-, 8-, 9-, 10-, 11- and 12-letter-long words in the OPD demonstrates that this simple approach was sufficient to correctly predict the exact VNF form in 32 cases (53%) and was able to approximately predict the location of a further 20 VNF frames (33%) (with an absolute error of $< 4_{10}$). Simple prefix coding, was insufficient to accurately predict the location of 8 VNF frames (13%). However only 1 of these 8 exceptional VNF frames, **VCVCCVCVC**, had not been computed previously. A composite image of the data is found in Table 5.5. Table 5.5 illustrates that a simple prefix code is capable of exactly predicting many of the dominant features of the English language lexicon from a single small kernel of VNF frames. This figure also demonstrates that the prefix code model is insufficient to exactly determine all of the dominant word frames found in English.

In the next section of this chapter we will explore some simulations of the prefix code model's performance. For these purposes we will use the VNF frame structures found to be most densely populated in 4-, 5-, 6- and 7-letter-long words. In addition a few simulations will use kernels constructed from information other than that obtained by a statistical analysis of rank-ordered set size. For example the 5-letter-long VNF kernel or basis of Table 5.1 is { 4, 5, 8, 9, 10, 11, 13, 18 }, while the empirically observed top-ten[4] 5-letter-long word groups are { 4, 9, 10, 5, 6, 12, 13, 8, 18, 21}. Since these sets are very similar only the results of a few simulations are presented in this chapter.

---

[4] The elements of this set are listed in descending rank-order; thus 4 is the base 10 VNF address of the most populated 5-letter-long VNF class and 21 is the address of the tenth-largest 5-letter-long VNF frame

5.5 SIMULATIONS

The simulations described in this chapter use N-th order VNF delta functions [5.1, 5.2] to predict the VNF forms found in the lexicon on the basis of the forms used in the kernel set A.

If the set A is composed of the most frequently occurring 5-letter-long VNF word forms then, $*^2(A)$, specifies the dominant 6-letter-long VNF word forms. Similarly the set, $*^3(A)$, specifies the dominant 7-letter-long VNF word forms on the basis of the kernel set A. Figures 5.6 and 5.7 depict the 5-letter-long VNF sets; $*^2(A)$ and $*^3(A)$ respectively derived from the set A of 5-letter-long word frames given in Figure 5.5.. In this example the kernel A is composed of the eight most populated 5-letter-long VNF word groups. Given a kernel of the top eight 5-letter-long VNF word groups the computation of the set $\{ \{*^2(A)\}, \{*^3(A)\}, \{*^4(A)\}, \{*^5(A)\}, \{*^6(A)\}, \{*^7(A)\}, \{*^8(A)\} \}$ would predict the major word forms for 6-, 7-, 8-, 9-, 10-, 11- , and 12-letter-long English words on the basis of their 5-letter-long kernel.

The total size of the set of VNF groups produced by these computations, $\Psi(N, A)$, is given by Equation 5.14. In this example where $S \mid A \mid = 10$ and $N = 8$ the value of $\Psi(N, A)$, is 2550. Each of these 2550 terms can be represented as a spike on the VNF number line. In this example $M = 5$ because 5-letter-long words were used as the basis for the kernel A. The largest VNF group described by this process is thus $2^{12} - 1 = 4095$. Thus in this example, over half (2550/4095), of all possible VNF groups were depicted as being well populated. Using the top ten 6-letter-long VNF word forms ($M = 6$) as a kernel and limiting our simulation to an upper limit of $*^7(A)$ will maintain the value of $M + N$ at 12 and thus the value of κ will remain at 4095. Figures 5.9, 5.10, 5.11, 5.12, 5.13 and 5.14 depict the 7-, 8-, 9-, 10-, 11- and 12-letter-long word sets $*^2(A)$, $*^3(A)$, $*^4(A)$, $*^5(A)$, $*^6(A)$, $*^7(A)$ derived from the 6-letter-long word frames given in Figure 5.8. In this example the value of $S \mid A \mid = 10$ and $N = 7$ and the value of Ψ becomes 1270. Therefore, in this case, approximately a

third (1270/4095), of all possible VNF word groups were depicted as being heavily populated. Similarly if A used a 6-letter-long VNF kernel and computations were carried to $*^{10}(A)$ then $10 * (2^{10} - 1)$ of the $2^{16} - 1$, possible VNF forms are specified as being well populated. In this case approximately one sixth (10,230/65,535) of the VNF word groups are predicted to be densely occupied.

Figure 5.15 depicts the macroscopic band-filtered characteristics of the VNF word line constructed on the basis of the eight most frequently used 5-letter-long VNF word groups, $\Phi^7(A)$, A = { 4, 9, 10, 5, 6, 12, 13, 8 }.

Figure 5.16 depicts the similar macroscopic effects computed on the basis of the eight most frequently used 6-letter-long VNF word groups, $\Phi^7(A)$, A = { 18, 21, 17, 20, 9, 22, 36, 10 }.

In all cases these simulations show that the macroscopic effects of a band-filtered response is independent of the kernel used to generate a given VNF line segment. The results of these simulations indicate that the presence and location of band-filtering in English language word-form groups is relatively independent of the word-size and base form used to initialize the simulation of the English lexicon.

Simultation results such as those depicted in Figures 5.15 and 5.16 do however indicate that the fine-structure detail of these simulations is dependent upon the kernel or base set of the VNF form used to compute the structure of the English language lexicon. The accuracy of such fine-structure detail is of course dependent upon the terms used in these simulations.

5.6 CONCLUSIONS

Detailed analysis of the data presented in this chapter demonstrates that in most instances it is possible to predict the structure of the top-ten VNF frames for n-letter-long-words directly from the structure of the top-ten VNF frames for (n - 1)-letter-long-words. The analysis of the most densely populated VNF structures found in 6-, 7-, ..., 12-letter-long word frames demonstrate that these

dominant VNF frames are related in English across word-size by a simple mathematical relation which makes use of prefix coding.

The simulation results described in this chapter indicate that the choice of the specific kernel of VNF word forms used to initialize the prefix code model has little effect on the macroscopic behavior of the model. In each simulation band-filtering occurs. Word groups are separated from adjacent clusters of word-groups by forbidden bands where English language word groups do not occur. Such simulations do not predict the size of the VNF word group but rather that it can or cannot exist. As such these models produce band-limited [5.2], comb-filtered VNF line segments. These segments resemble the Dirac Delta Functions [5.4, 5.5] first formulated in atomic physics.

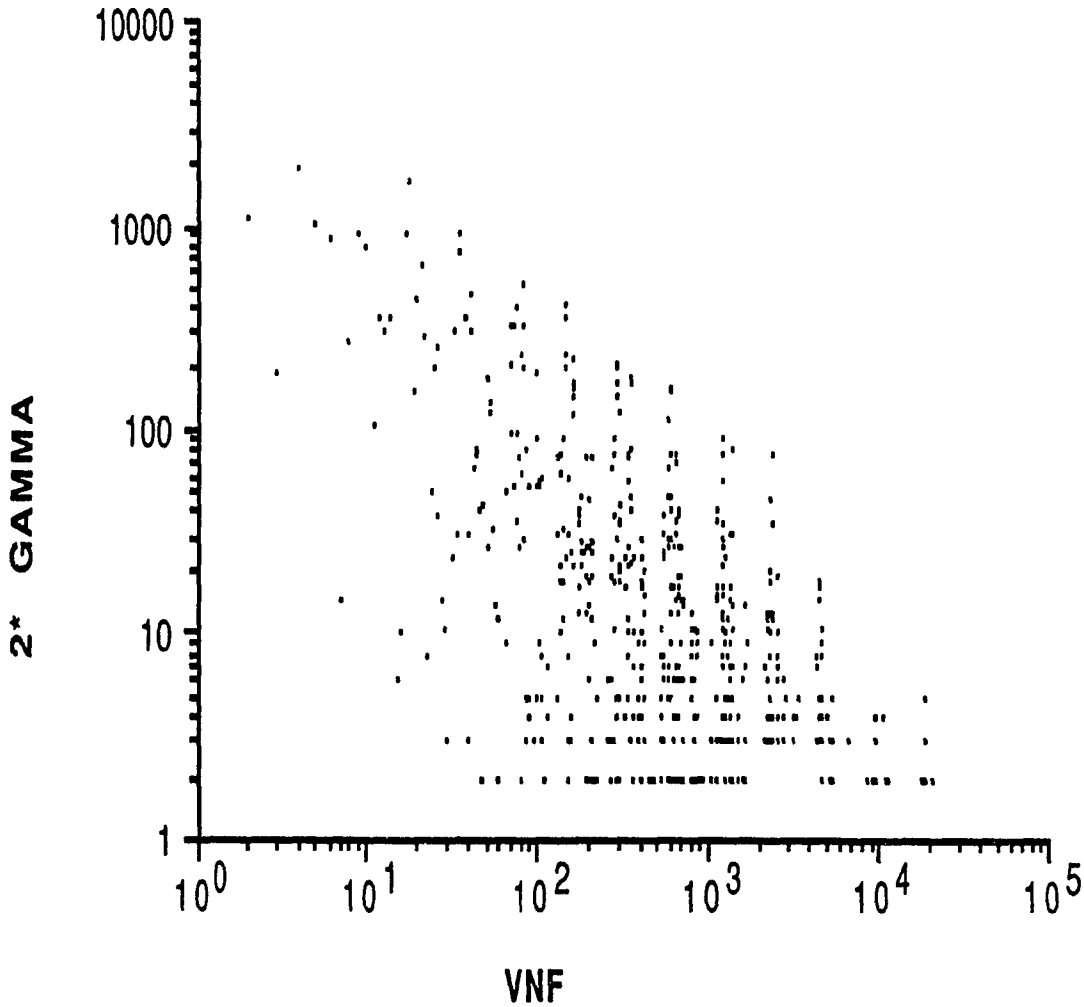

Figure 5.1 Scatter plot of VNF set size as a function of VNF structure. Abscissa: VNF class or structure specified as a base 10 number. Ordinate: two times VNF set size. See Figure 4.1 for a histogram of this data.

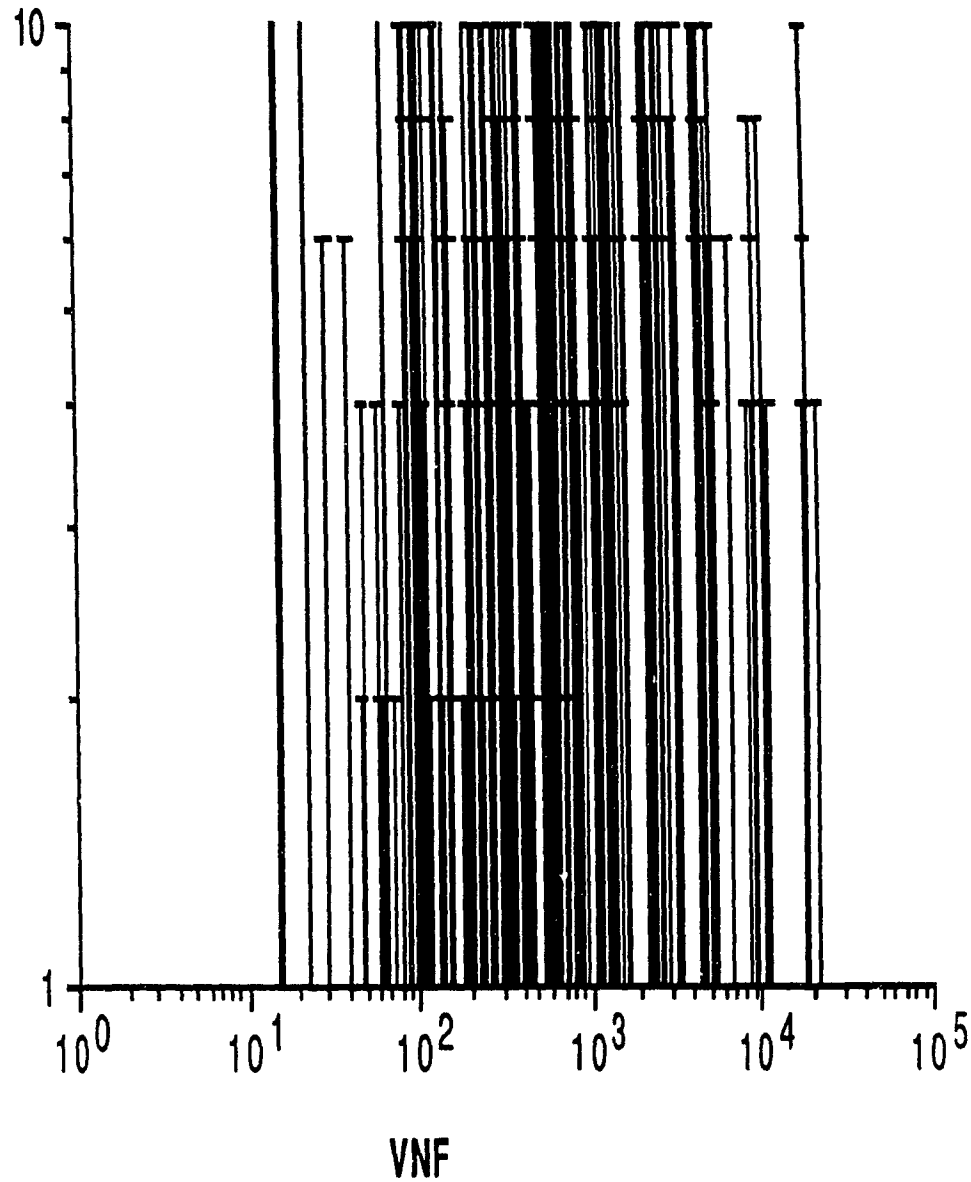

Figure 5.2 Sparsely populated VNF structures. Histogram of VNF set size as a function of VNF structure for VNF frames containing between 1 and 10 words. Abscissa: VNF class or structure specified as a base 10 number. Ordinate: log of VNF set size. Horizontal bands in this figure illustrate the effect of VNF set size on the band filtering observed in sparsely populated VNF structures. See Figure 4.1 for a non-filtered histogram of this data.

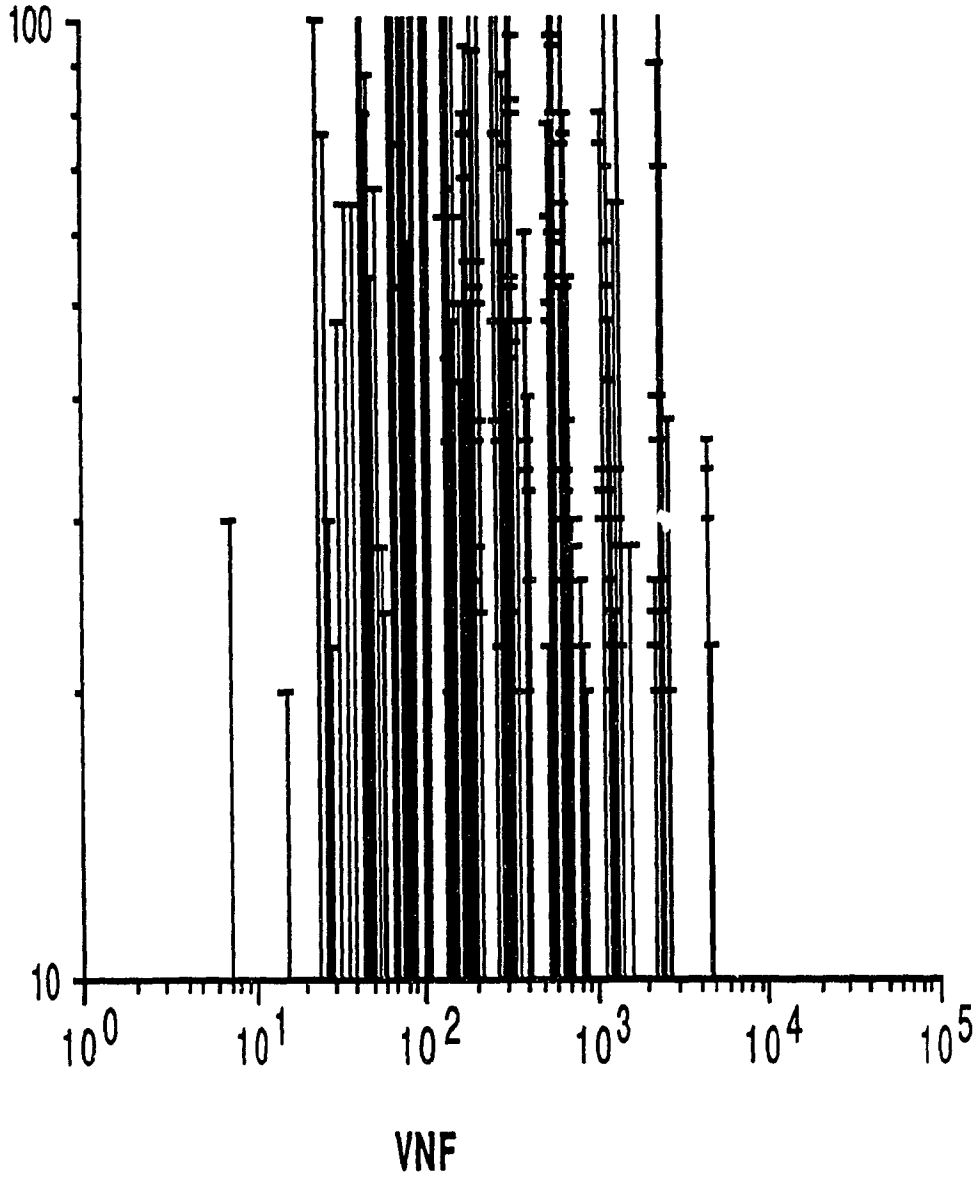

Figure 5.3 Sparsely populated VNF structures. Histogram of VNF set size as a function of VNF structure for VNF frames containing between 10 and 100 words. Abscissa: VNF class or structure specified as a base 10 number. Ordinate: log of VNF set size. Horizontal bands in this figure illustrate the effect of VNF set size on the band filtering observed in sparsely populated VNF structures. See Figure 4.1 for a non-filtered histogram of this data.

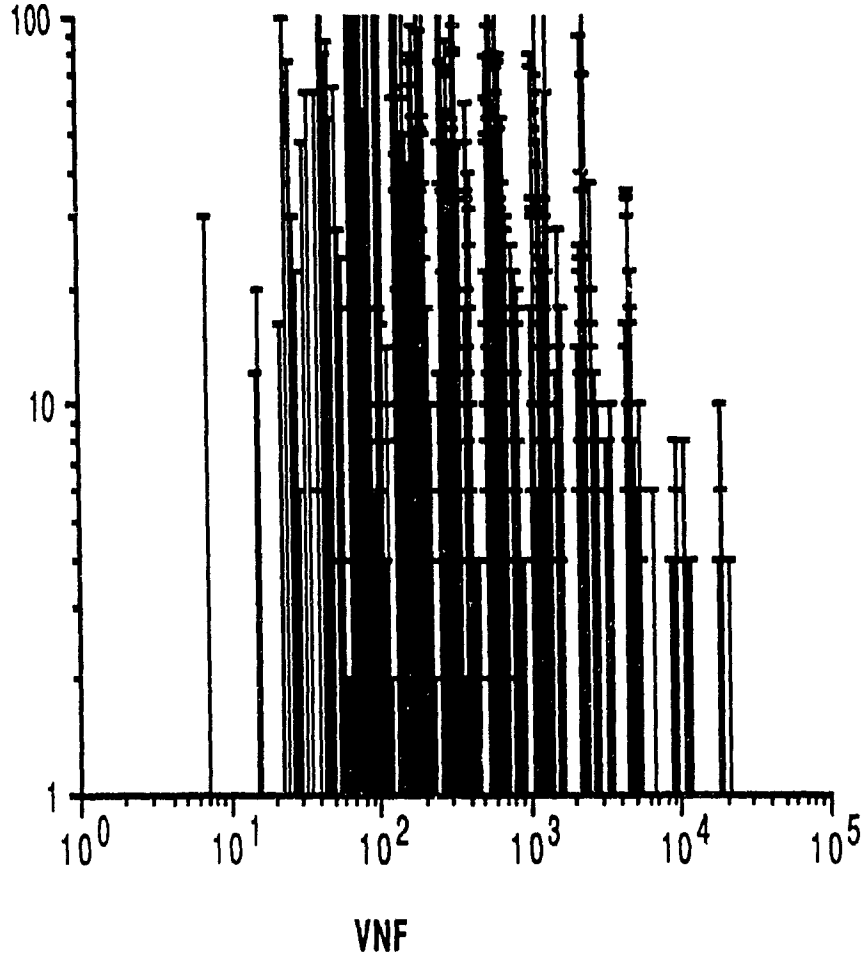

Figure 5.4 Composite image of the more sparsely populated VNF structures. Histogram of VNF set size as a function of VNF structure for VNF frames containing between 1 and 100 words. See Figures 5.2, 5.3 and 5.4 for fine-level detail of this image. The horizontal bands illustrating band filtering as a function of set size. Abscissa: VNF class or structure specified as a base 10 number. Ordinate: log of VNF set size. See Figure 4.1 for a non-filtered histogram of this data.

RANK-ORDER	6	7	8	9	10	11	12
1	1·8	3·6	8·5	149	342	598	11·73
2	2·1	3·7	7·4	148	341	597	23·45
3	1·7	1·8	8·4	165	292	11·73	1193
4	2·0	3·8	6·8	146	293	585	11·74
5	9	4·2	7·3	150	297	661	2390
6	2·2	4·1	3·6	169	294	662	2389
7	3·6	3·4	8·2	293	597	11·93	1365
8	1·0	1·7	8·6	164	298	580	1321
9	2·6	2·1	6·9	170	585	596	11·94
10	3·7	5·0	3·7	8·5	148	11·74	13·18

WORDLENGTH

Table 5.1 The top-ten rank-ordered VNF_{10} word frames found in all 6-, 7-, 8-, 9-, 10-, 11- and 12-letter-long words listed in the OPD [5.6]. Columns specify word size. Rows specify the 1st, 2nd, ..., 10th most-densely populated VNF word groups. VNF structures are given as base 10 numbers.

RANK-ORDER	6	7	8	9	10	11	12
1	1·8	3·6	8·5	149	342	598	1173
2	2·1	3·7	7·4	148	341	597	2345
3	1·7	1·8	8·4	165	292	1173	1193
4	2·0	3·8	6·8	146	293	585	1174
5	9	4·2	7·3	150	29·	661	2390
6	2·2	4·1	3·6	169	294	662	2389
7	3·6	3·4	8·2	293	597	1193	1365
8	1·0	1·7	8·6	164	298	580	1321
9	2·6	2·1	6·9	170	585	596	1194
10	3·7	5·0	3·7	8·5	148	1174	1318

WORDLENGTH

Table 5.2 The top-ten rank-ordered VNF_{10} word frames found in all 6-, 7-, 8-, 9-, 10-, 11- and 12-letter-long words listed in the OPD [5.6]. Columns specify word size. Rows specify the 1st, 2nd, ..., 10th most-densely populated VNF word groups. VNF structures are given as base 10 numbers. Two types of directed arrows are used in this table to show that, in all cases, the most-populated VNF structure found for a given word-length can be either exactly or approximately computed from a well-populated VNF structure found in the next largest word-size. Closed arrows are used to specify VNF structures which can be exactly computed from the VNF frames found in the next largest word-size by the application of a simple prefix code. Open arrows are used to specify VNF structures which can be approximately computed (with an error of less than 4_{10}) from VNF structures found in the next largest word-size by the application of a simple prefix code.

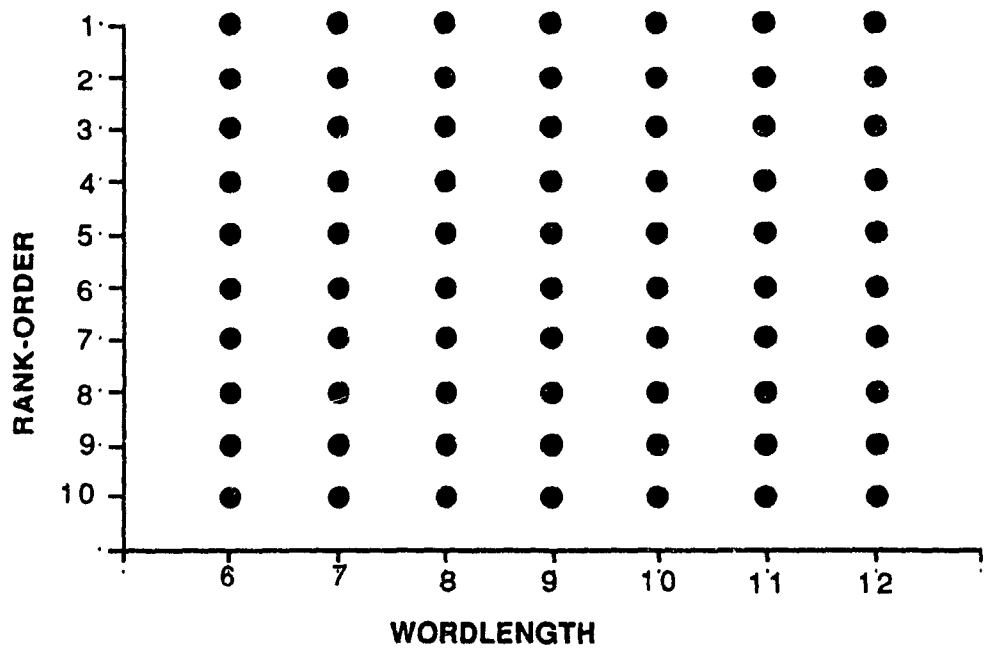

Table 5.3 Simplified version of Table 5.1 in which the specific value of all VNF word frames are substituted by dots. The relations observed between the top-ten rank-ordered word frames found in Table 5.1 are depicted by three types of directed arrows. Closed arrows such as those found in Table 5.2 are used to specify VNF structures which can be exactly computed from the VNF frames found in the next largest word-size by the application of a simple prefix code. Open arrows such as those used in Table 5.2 are used to specify VNF structures which can be approximately computed (with an error of less than 4_{10}) from VNF structures found in the next largest word-size by the application of a simple prefix code. . Thin arrows, such as those first seen in Table 5.8, are used to depict those VNF structures which cannot be either exactly or approximately computed in such a simple manner.Columns specify word size. Rows specify the 1st, 2nd, ..., 10th most-populated structures

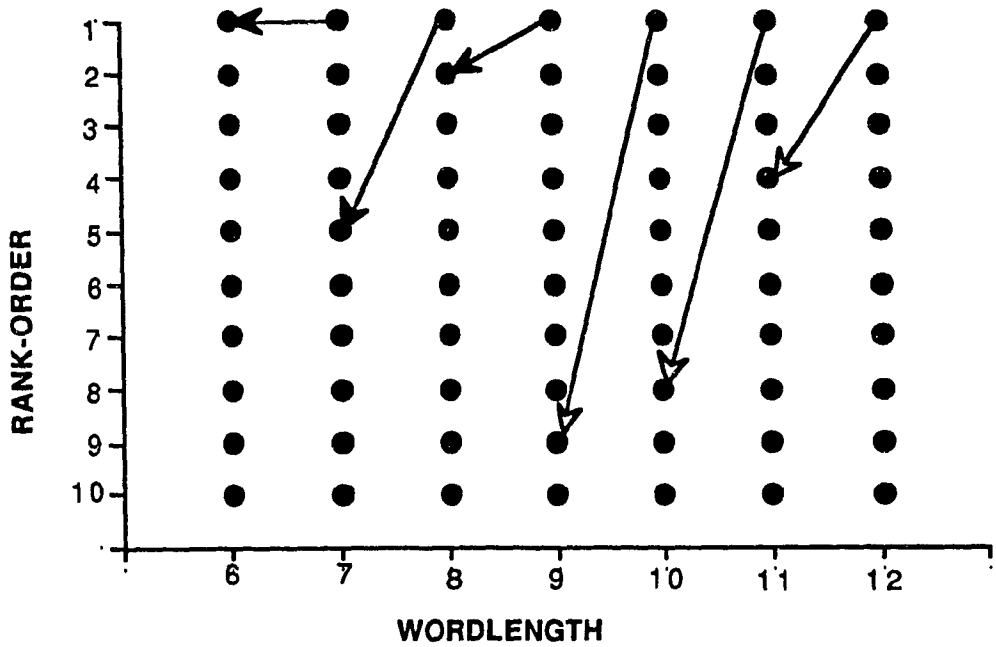

Table 5.4 Predicting the most-populated VNF frames for words of lengths 6,..., 12 as a function of the structures used in the next-largest word-length. Simplified version of Table 5.1 in which the specific value of all VNF word frames are substituted by dots. The prefix relations observed between the top-ten rank-ordered word frames found in Table 5.1 are depicted by three types of directed arrows described in Tables 5.2 and 5.3.

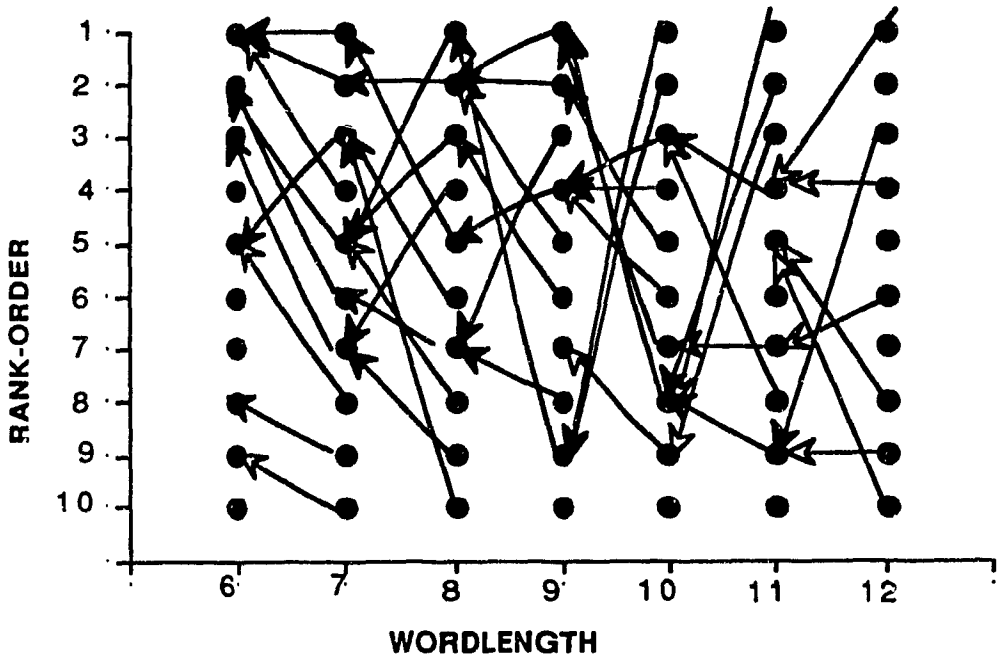

Table 5.5 Predicting the kernels, or the top-ten most-densely populated VNF frames, for words of lengths 6,..., 12 as a function of the structures used in the kernels of the next-largest word-size. This composite table demonstrates that (for 6-, ..., 11-letter-long-words), in most cases, the kernels of the each set can be computed from the dominant VNF structures found in the larger words. Simplified version of Table 5.1 in which the specific value of all VNF word frames are substituted by dots. The prefix relations observed between the top-ten rank-ordered word frames found in Table 5.1 are depicted by three types of directed arrows described in Tables 5.2 and 5.3. For the sake of clarity only two types of arrows are depicted in this table.

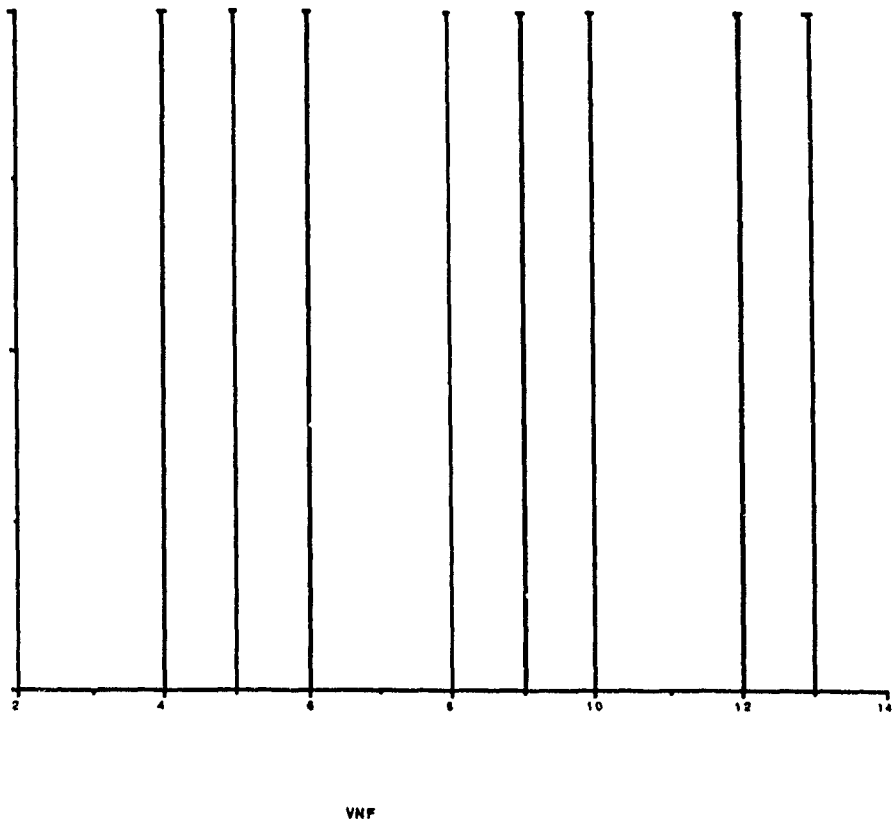

Figure 5.5 Base 10 VNF number line segment (2..14) depicting the set A which, in this case, is composed of the top-eight most populated 5-letter-long frames { 4, 9, 10, 5, 6, 12, 13, 8 } found in the OPD [5.6]. Vertical spikes are used to denote the presence of a dominant VNF set.

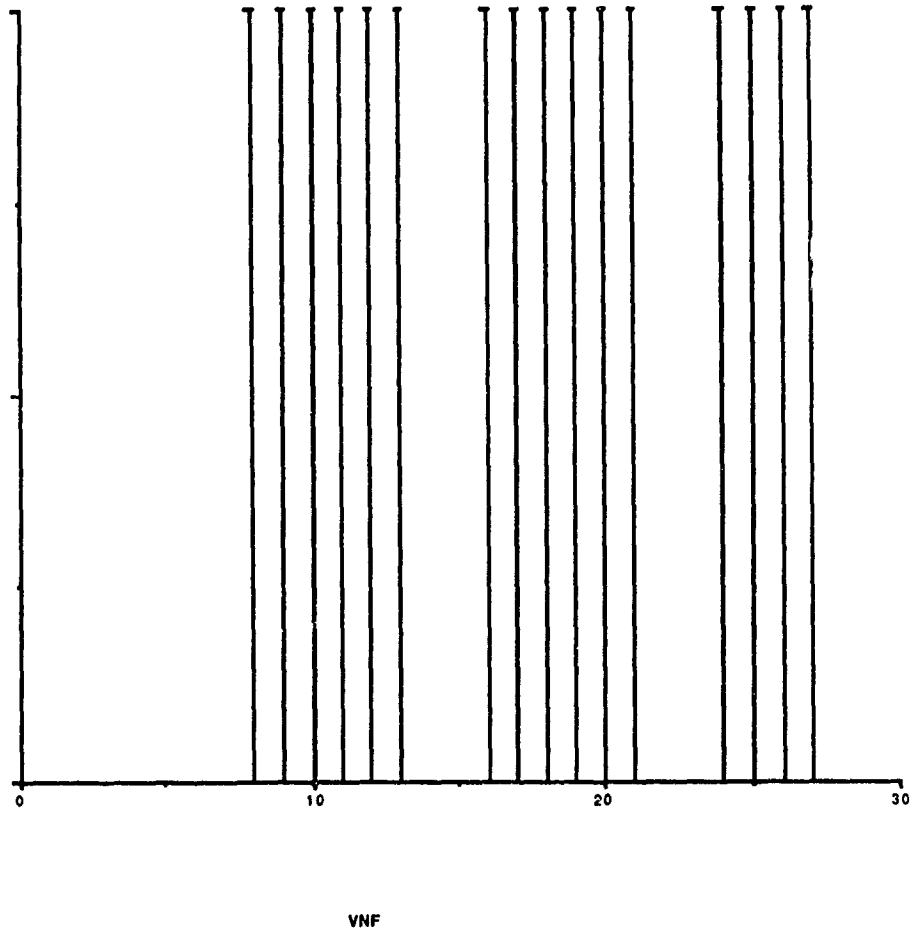

Figure 5.6 Base 10 VNF number line segment (0..30) depicting $*^1(A)$. $*^1(A)$ is, in this case, the predicted dominant 6-letter-long VNF word groups computed on the basis of the kernel set A of 5-letter-long VNF word groups given in Figure 5.5. Vertical spikes are used to denote the predicted presence of a dominant VNF set.

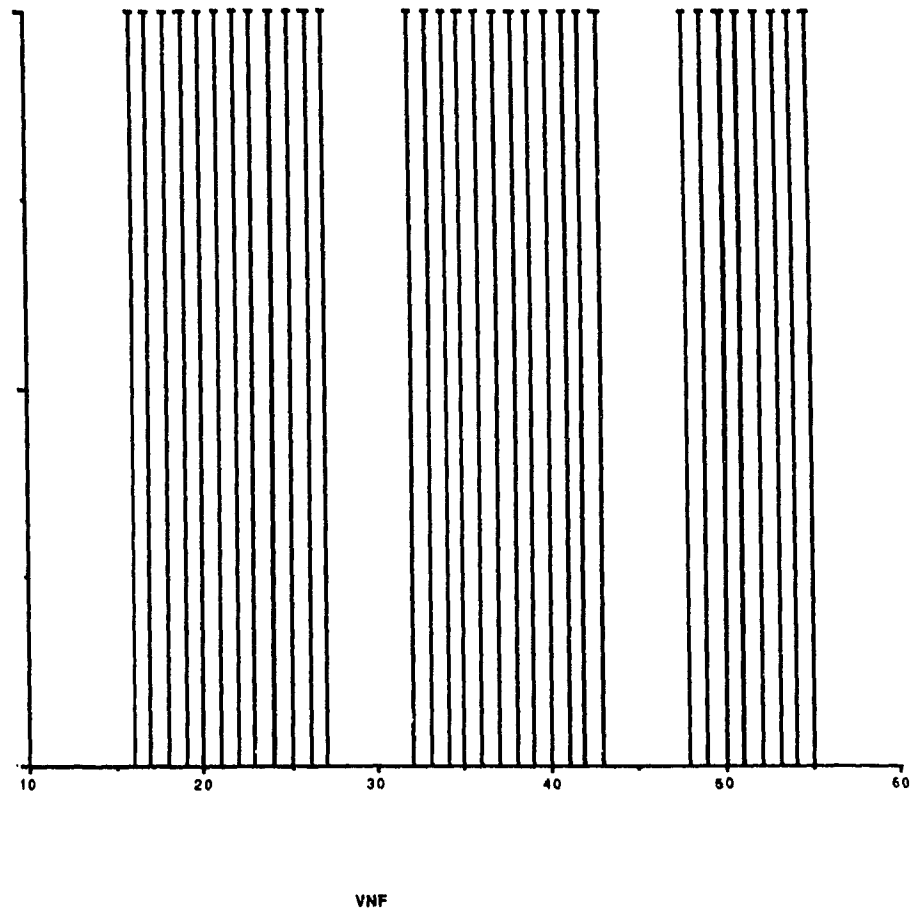

Figure 5.7 Base 10 VNF number line segment (10..60) depicting $*^2$ (A). $*^2$ (A) is, in this case, the predicted dominant 7-letter-long VNF word groups computed on the basis of the kernel set A of 5-letter-long VNF word groups given in Figure 5.5. Vertical spikes are used to denote the predicted presence of a dominant VNF set.

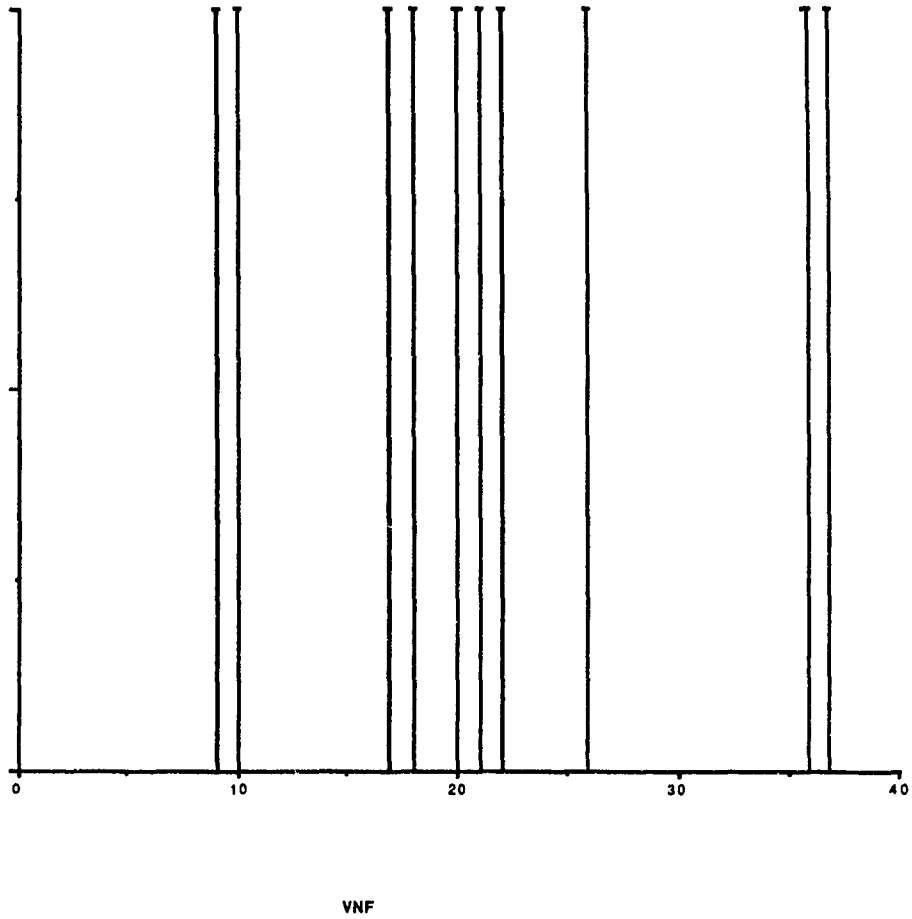

Figure 5.8 Base 10 VNF number line segment (10..40) depicting the set A which, in this case, is composed of the top-ten most densely populated 6-letter-long VNF frames { 18, 21, 17, 20, 9, 22, 36, 10, 26, 37 } found in the OPD [5.6]. Vertical spikes are used to denote the presence of a dominant VNF set.

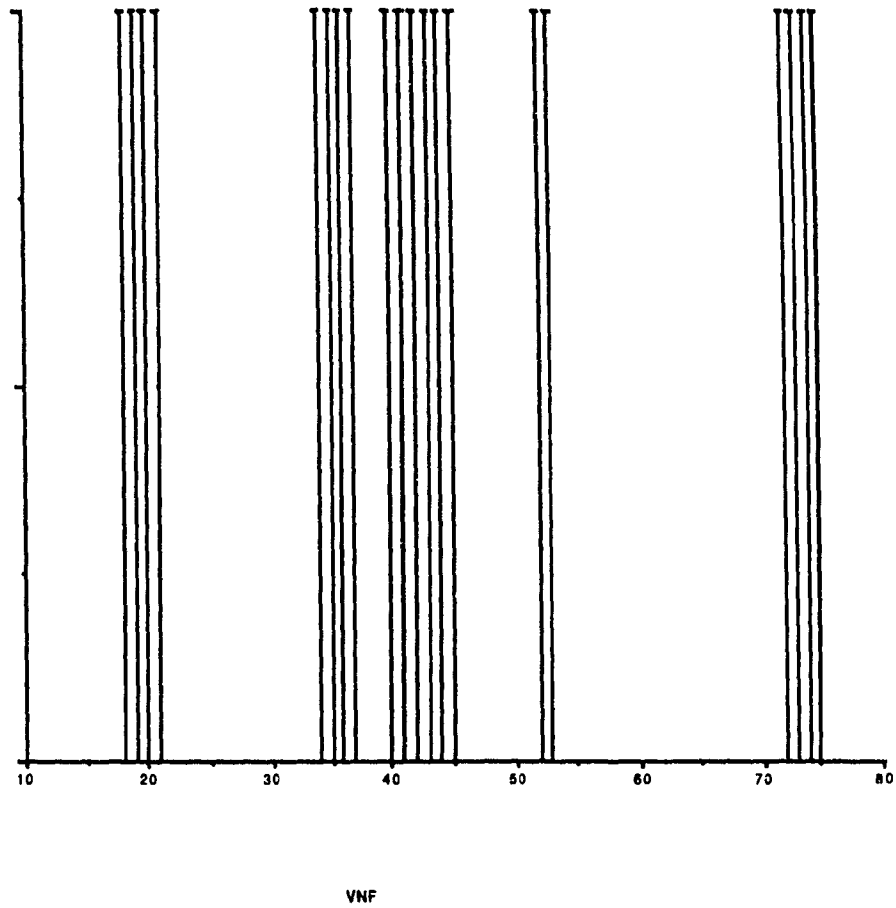

Figure 5.9 Base 10 VNF number line segment (10..80) depicting $*^2$ (A). $*^2$ (A) is, in this case, the predicted dominant 7-letter-long VNF word groups computed on the basis of the kernel set A of 6-letter-long VNF word groups given in Figure 5.8. Vertical spikes are used to denote the predicted presence of a dominant VNF set.

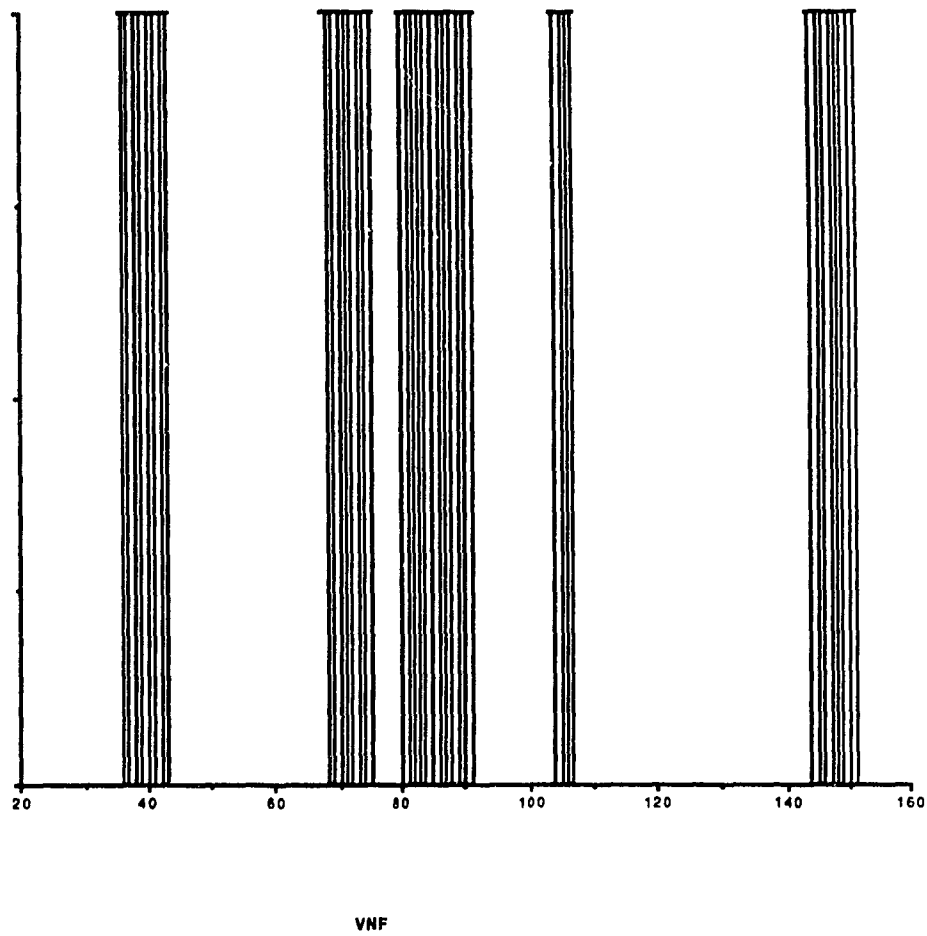

Figure 5.10 Base 10 VNF number line segment (20..160) depicting $*^3$ (A). $*^3$ (A) is, in this case, the predicted dominant 8-letter-long VNF word groups computed on the basis of the kernel set A of 6-letter-long VNF word groups given in Figure 5.8. Vertical spikes are used to denote the predicted presence of a dominant VNF set.

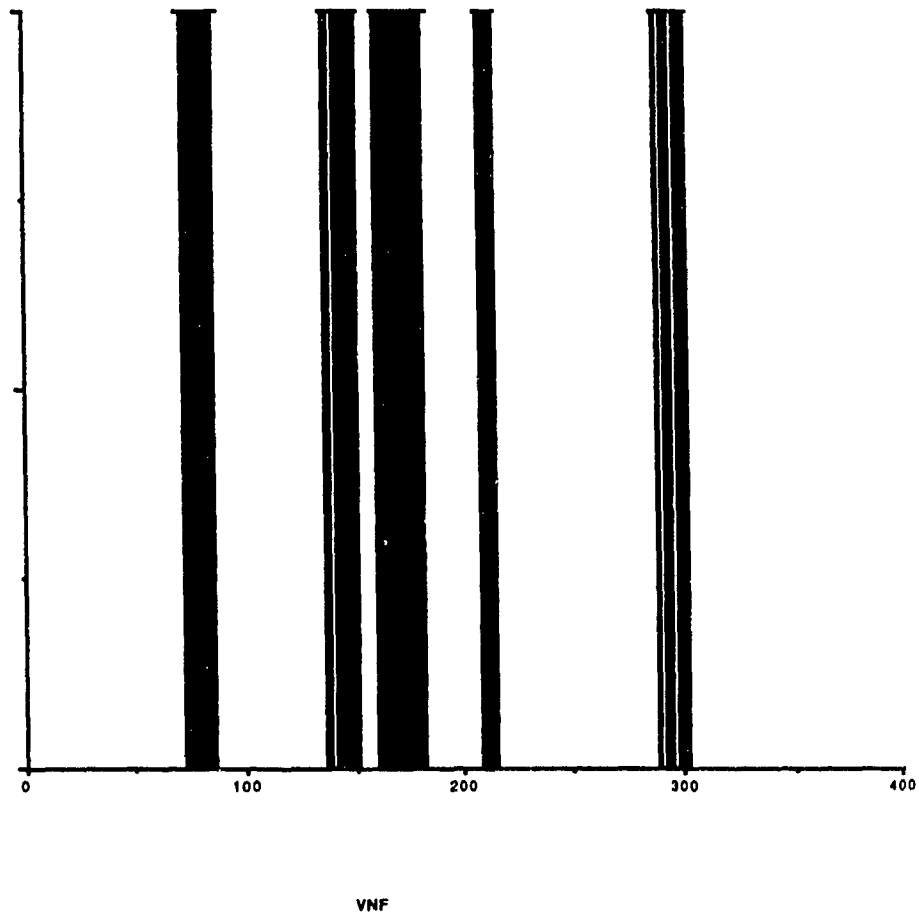

Figure 5.11 Base 10 VNF number line segment (0..400) depicting $*^4$ (A). $*^4$ (A) is, in this case, the predicted dominant 9-letter-long VNF word groups computed on the basis of the kernel set A of 6-letter-long VNF word groups given in Figure 5.8. Vertical spikes are used to denote the presence of a densely populated VNF set.

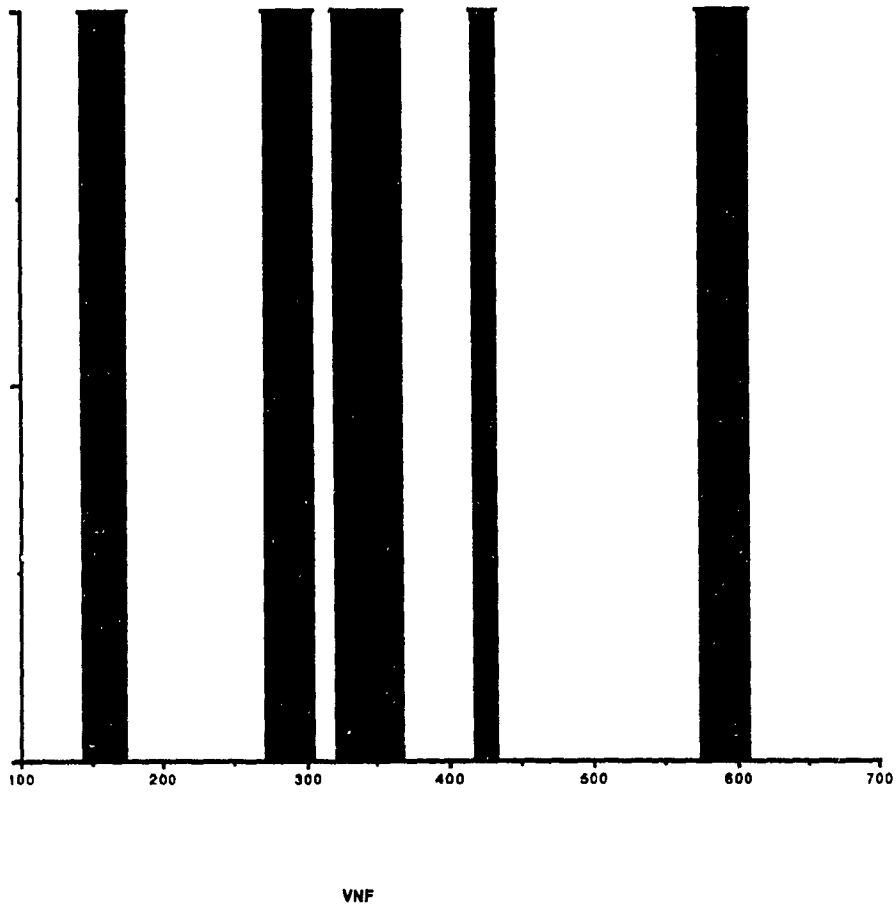

Figure 5.12 Base 10 VNF number line segment (100..700) depicting $*^5$ (A). $*^5$ (A) is, in this case, the predicted dominant 10-letter-long VNF word groups computed on the basis of the kernel set A of 6-letter-long VNF word groups given in Figure 5.8. Vertical spikes are used to denote the presence of a densely populated VNF set.

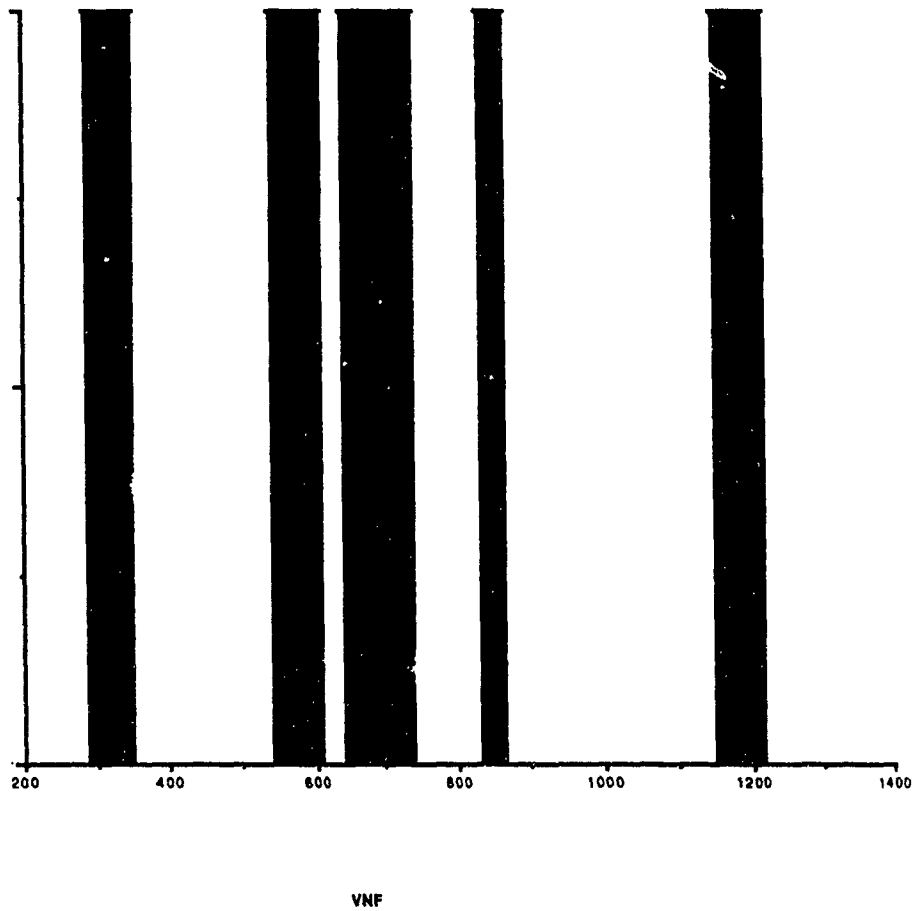

Figure 5.13 Base 10 VNF number line segment (200..1400) depicting $*^6$ (A). $*^6$ (A) is, in this case, the predicted dominant 11-letter-long VNF word groups computed on the basis of the kernel set A of 6-letter-long VNF word groups given in Figure 5.8. Vertical spikes are used to denote the presence of a densely populated VNF set.

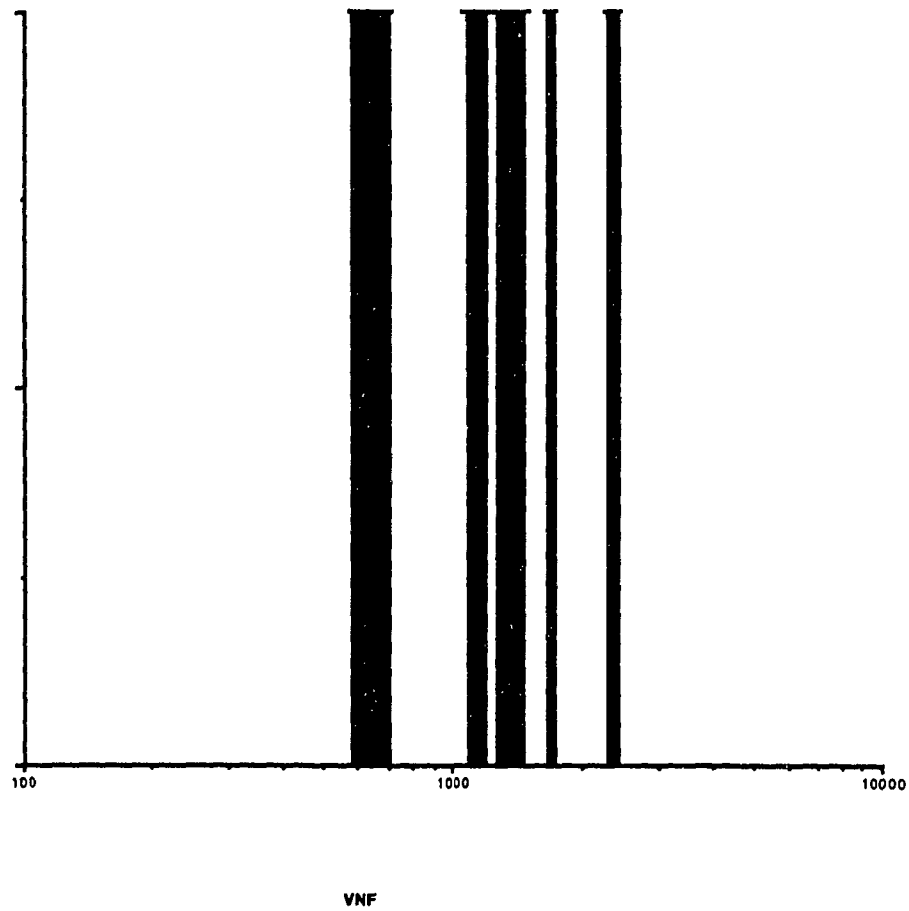

Figure 5.14 Base 10 VNF number line segment

(100..10000) depicting $*^7$ (A). $*^7$ (A) is, in this case, the predicted dominant 12-letter-long VNF word groups computed on the basis of the kernel set A of 6-letter-long VNF word groups given in Figure 5.8. Vertical spikes are used to denote the predicted presence of densely populated VNF sets.

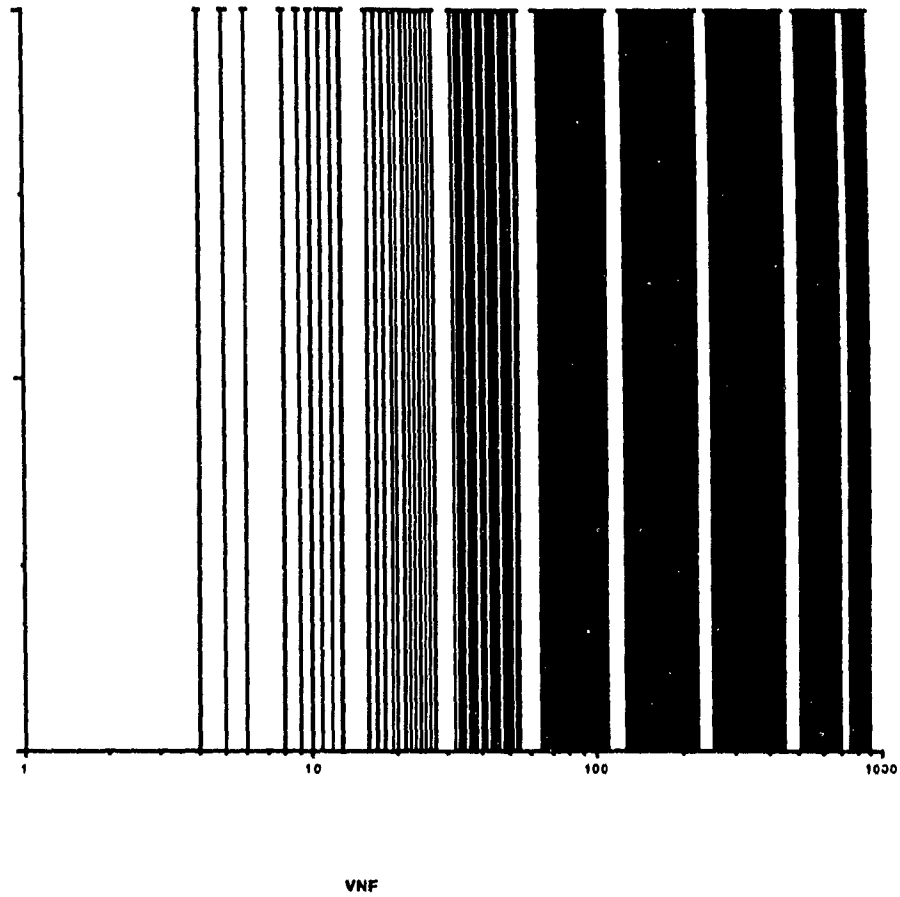

Figure 5.15 Base 10 VNF number line segment (1..1000) Φ^7(A) depicted here is constructed on the basis of the top-eight most densely populated 5-letter-long word groups found in the OPD [5.6]. Φ^7(A) produces a composite image formed as the union of elements contained in seven sets: A, $*^2$ (A),.. .,$*^7$ (A) respectively.

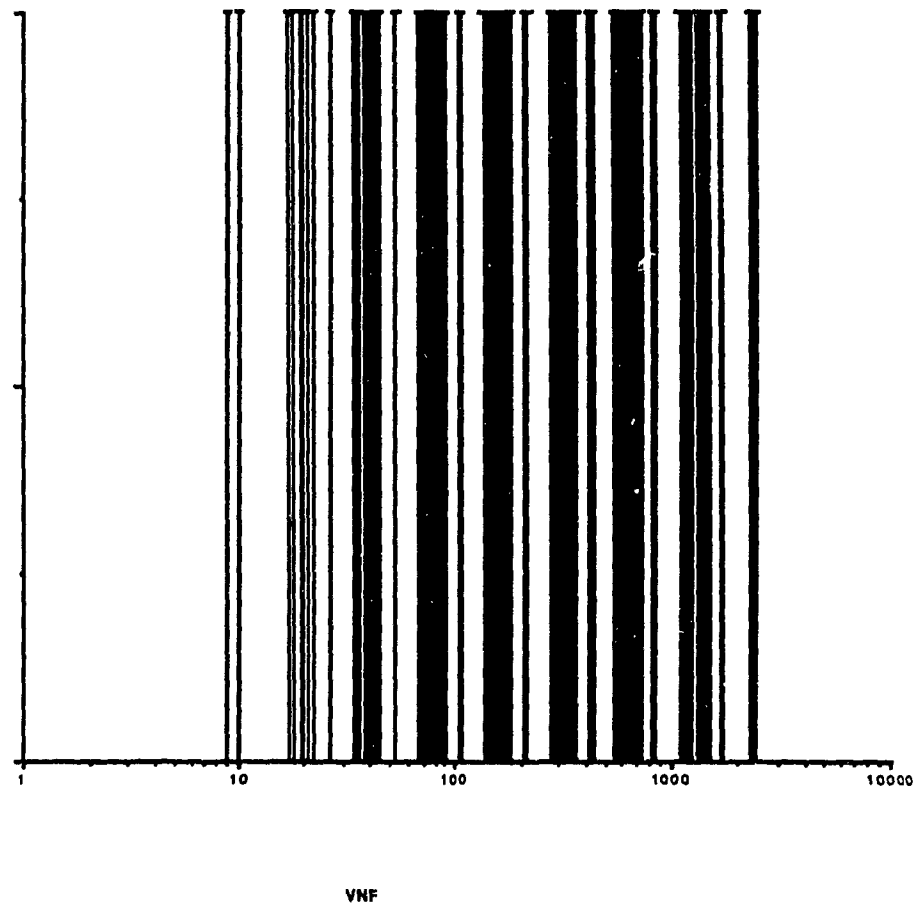

Figure 5.16 Base 10 VNF number line segment (1..10000) Φ^7(A) depicted here is constructed on the basis of the top-ten most densely populated 6-letter-long word groups found in the OPD [5.6]. Φ^7(A) produces a composite image formed as the union of elements contained in seven sets: A, $*^2$ (A),...,$*^7$ (A) given in Figures 5.8,...,5.14 respectively.

5.7 REFERENCES

[5.1] see 1.74

[5.2] see 1.73

[5.3] N. Abramson, Information Theory and Coding, McGraw-Hill, N.Y., 1963.

[5.4] R. Feynman, R. Leighton, M. Sands, The Feynman Lectures on Physics, Addison-Wesley, Reading, Massachusetts, 1963 - 65.

[5.5] P. Dirac, The Principles of Quantum Mechanics, [4th ed.], Clarendon, Oxford, England, 1958.

[5.6] see 3.2

CHAPTER SIX
PREDICTING THE SIZE OF THE DOMINANT VNF SETS IN ENGLISH

6.1 INTRODUCTION

The previous chapter presented work on a Prefix Code Model of word structure which was developed to determine which VNF word structures or frames are dominant in the lexicon defined in the OPD [6.1]. The models presented in Chapters 4 and 5 provided insight into the existence of VNF band-filtered frames found in the English lexicon. However, such prefix models, *per se*, are insufficient to determine the actual set size of the dominant VNF frames.

Equations, such as those presented in this chapter, can be used to compute the predicted set sizes of the most popular VNF structures found in English words of a given length. Much of the work presented in this chapter has been submitted for publication [6.2, 6.3].

6.2 THEORY & RESULTS

Figure 6.1, which is from the work of Kucera, [6.4], depicts the number of distinct words or "types" of a given length found in a lexicon. Figure 6.1 also depicts the number of "tokens" or the total number of words of a given length found in the same lexicon.

Log-normal distributions, such as that shown in Figure 6.1, typically describe both the "type-counts" and the "token-counts" for natural language texts [6.2]. Statistical data, which has a log-normal distribution [6.4, 6.5], also conforms to Zipf's law [6.5, 6.6]. Zipf noted that such data, when sorted into a rank ordered sequence, exhibits a relationship where the logarithm of the rank of an item is inversely proportional to the logarithm of some measure such as its frequency of use or size. More recently various researchers have demonstrated that classic, inverse power-law relationships such as those depicted in Figure 6.2, occur in many natural settings [6.7, 6.8, 6.9], including computational linguistics [6.6, 6.7]. These Zipf-like

effects are different from those found in the study of English language VNF set size and form.

The empirically observed exponential relationship found in the analysis presented in this chapter resembles that observed by Halstead and his colleagues for Chomsky's type 2 languages [6.8].

The results of an empirical analysis of the actual VNF set size, Γ, for the ten most-populated VNF word forms used in 5-, 6-, 7-, 8-, 9-, 10-, 11-, and 12-letter-long words, in the OPD, are depicted in Figures 6.3 and the histogram given in Figure 6.4. Figure 6.4 illustrates that a simple exponential relationship accurately represents set size as a function of rank for the most-populated, second-most-populated, third-largest, fourth-largest, ..., tenth-largest VNF word forms found in 5-, 6-, 7-, 8-, 9-, 10-, 11-, and 12-letter-long words defined in the OPD.

In general, we observed in Chapter 4, that relatively few VNF structures account for the majority of words of a given length and that a great many VNF frames are either sparsely populated or not populated at all. The set of histograms found in Figure 6.5 depicts the VNF set sizes of the rank-ordered top-ten VNF structures found for 4-, .., 12-letter-long-words listed in the OPD. From Figure 6.5 we can observe the characteristic shape and self-similarity of the histograms for the largest, second-largest, .., tenth-largest VNF frame. From these results we observe that, for words of any given length, rank-ordered VNF set size follows a simple exponential decay. This observation does not hold for the relatively few very long words found in the OPD. Similarly 2-, 3- and 4-letter-long words, which have very few possible VNF frames or structures, do not conform to the regularity observed across the great mass of words listed in the dictionary. Figure 6.4 demonstrates the remarkable degree to which the observed regularity holds for words listed in the OPD of lengths 5 through 12. Furthermore, the collinearity of the decay curves found in this study indicates that a single function may be used to accurately describe set size as a function of rank.

The predicted VNF set size, Γ, for word structures of a given rank order, ρ, and length, l, may be given in terms of a simple, two parameter equation of the form:

$$\Gamma(\rho, l) = \alpha * l + e^{\beta\rho} + \kappa \qquad (6.1$$

for $4 < l < 14$ and $\rho = 1..10$
where $\alpha = -101.67$, $\beta = 0.16698$ and $\kappa = 1246.7$

Figures 6.6, 6.7, and 6.8 depict the disparity observed between the VNF set size data found in Figure 6.4 and that predicted by Equation 6.1. The correlation coefficient, r^2, for this model when computed over the range, $4 < l < 14$ and $\rho = 1..10$, is 0.844. If one deletes the single outlier ($\rho = 1$, $l = 6$) the model's correlation coefficient raises to a value of 0.946.

Figure 6.5 shows that the 4-letter-long words, in the OPD, do not follow the decay curve given by Equation 6.1. The 4-letter-long word results are perhaps accounted for by the exclusion from the OPD of words that are deemed socially unacceptable or offensive by the editors of the Oxford dictionaries. Typically words that are considered to be racist, obscene, sexist, or scatological are deleted from the OPD. Among the words that are censored in this process are very popular 4-letter-long words belonging to VNF forms such as **CVCC**, and **CCVC**. Other censored words are 3-letter-long words, belonging to VNF forms such as **CVC**. As mentioned in Chapter 3, censorship makes it difficult to use a dictionary database to perform a proper analysis of 3- and 4-letter-long word groups.

Figure 6.9 illustrates the relationship observed by Kucera [6.4] between the number of 'types' and the number of 'tokens' observed in text samples as well as that predicted on the basis of a log normal sample distribution. Figure 6.10 illustrates the relationship observed in this study between the population density of a VNF frame and its rank-order. The relationship between type-count and token-count is only crudely described by Zipf's law in these two cases.Figures 6.9 and 6.10 appear to show remarkably similar complex dynamics that will be the subject of further research.

6.4 CONCLUSION

The ten most used VNF word groups of a given word length, account for the vast majority of words of a given size. A single function suffices to predict the size of the most populated VNF word groups as a simple function of word length and rank order, for $l > 4$.

The empirically observed exponential relationship shown in Figure 6.4, that was found to hold between word rank and VNF set size may be used to accurately estimate VNF set size with the exception of a single VNF group. This word group, which has the form **CVCCVC**, is predicted, by Equation 6.1, to have only 551 elements while over 1112 words are found in the OPD to have this structure. There is no obvious reason for the aberrant behavior of this outlying VNF set. Further in-depth analysis, of the words conforming to this structure, may provide some insight into the aberrant behavior of this word frame.

Simple equations such as Equation 6.1 can be used to accurately predict rank order VNF set size shown in Figures 6.4 and 6.5.

Chapter 7 will discuss the role of functions such as those presented here in developing comprehensive models of basic English language word frames. A comprehensive model of the lexicon must not only predict the structure of basic linguistic frames but also their relative size.

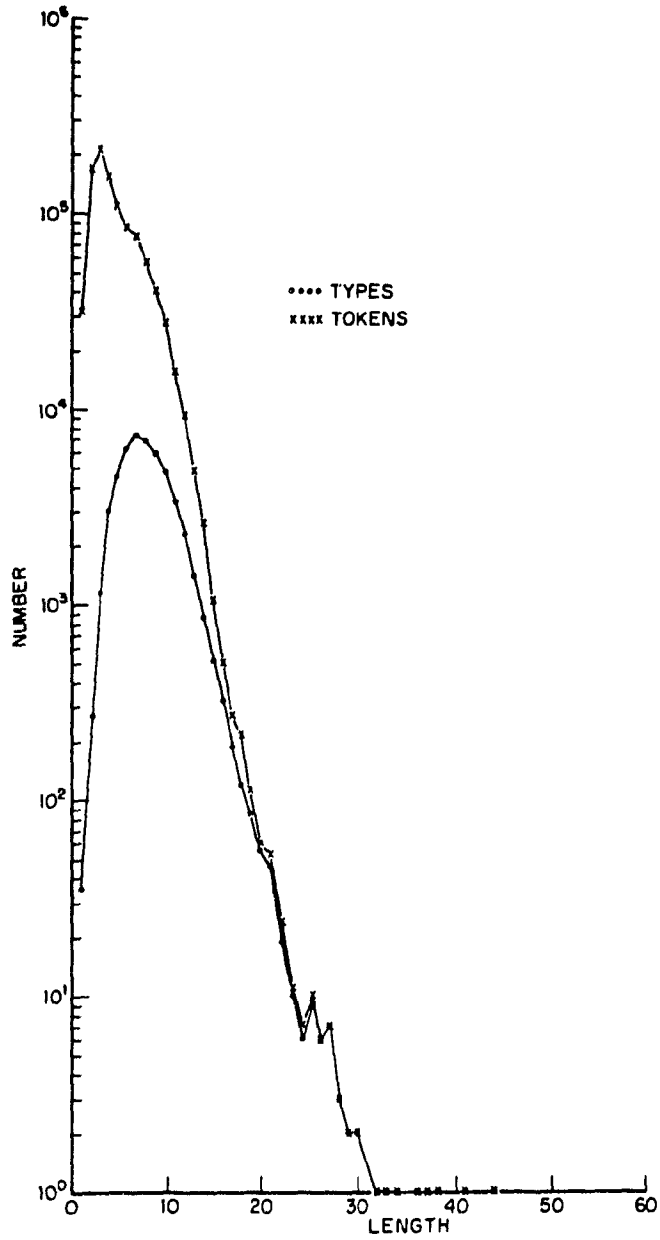

Figure 6.1 Abscissa: Length in alphabetic characters. Ordinate: Number of 'types' and 'tokens'. A 'type' is a distinct word such as the article 'THE' which if used 10,000 times in a textbook would have a 'token-value' of 10,000. For example the 'type-count' of 2-letter-long words found in this figure caption is "5" while the caption's 'token-count' is "11". This figure was taken from Kucera [6.4].

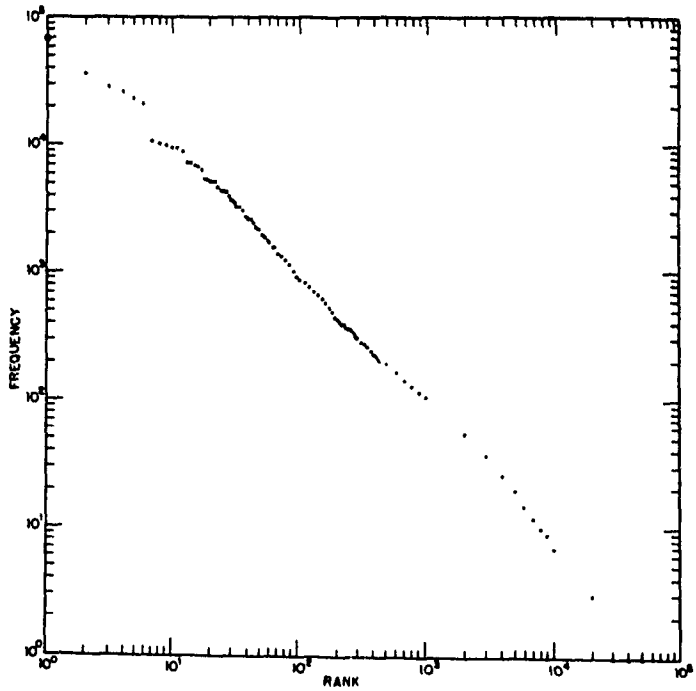

Figure 6.2 Zipf's Law. Inverse power 'law' or relationship observed between the actual frequency of occurrence of a word in a text and its rank order. For this purpose words are ranked in descending order so that the most frequently used word is given order one, while the least frequently used word of N distinct vocabulary words is given order N. Abscissa: Base 10 logarithm of the word's relative frequency of occurrence or rank in a text. Ordinate: Base 10 logarithm of the actual frequency of occurrence of the word in a text. This figure was taken from Kucera [6.4]

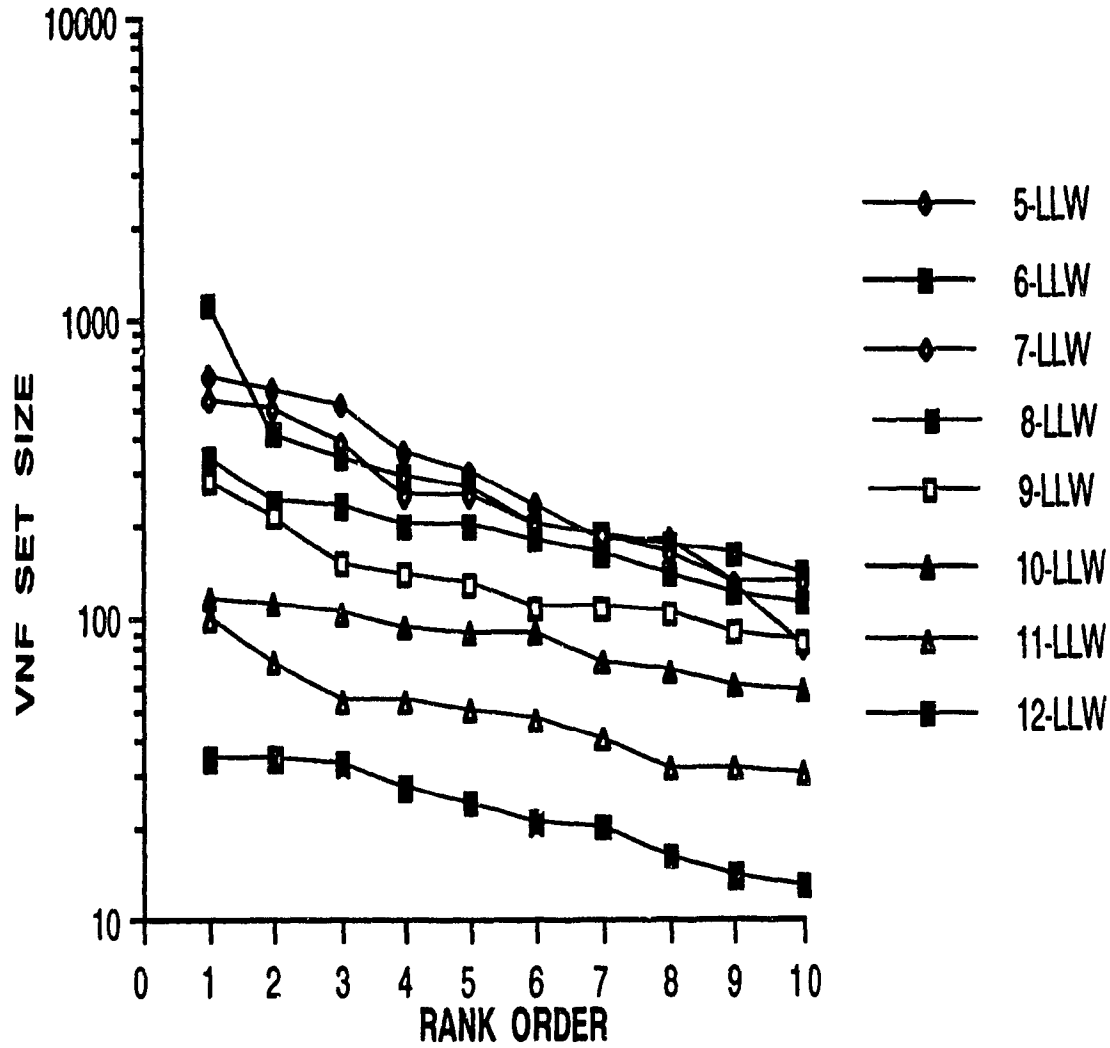

Figure 6.3 VNF set size as a function of rank order. The set sizes of the ten most populated VNF frames for 5-, ..., 12-letter-long words defined in the OPD are given as a function of their rank order. These plots illustrate a simple exponential decay in set size as a function of rank order. Abscissa: Rank order, 1 being assigned to the largest set size and 10 being assigned to the smallest set size. Ordinate: Base 10 logarithm of the set size in words.

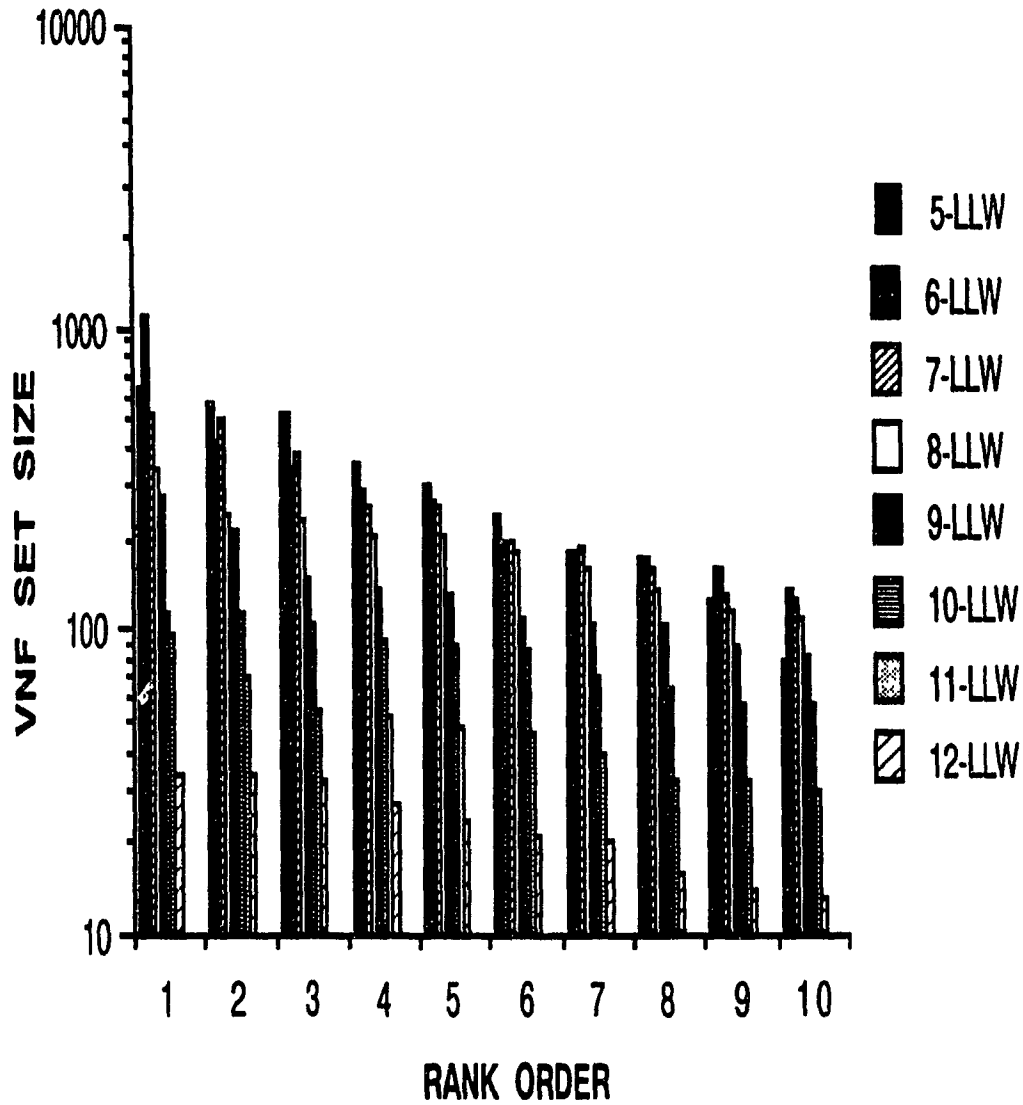

Figure 6.4 Composite images of ten histograms. Each of these histograms depict the relative set sizes of 5-, , 12-letter-long-word frames defined in the OPD. A separate histogram is used to illustrate the effect of word-length on set size for the most-populated, second-most-populated, .., tenth-most-populated VNF classes. These histograms illustrate the simple regularity of the observed exponential decay process throughout the top-ten major VNF classes found in 5- to 12-letter-long words.

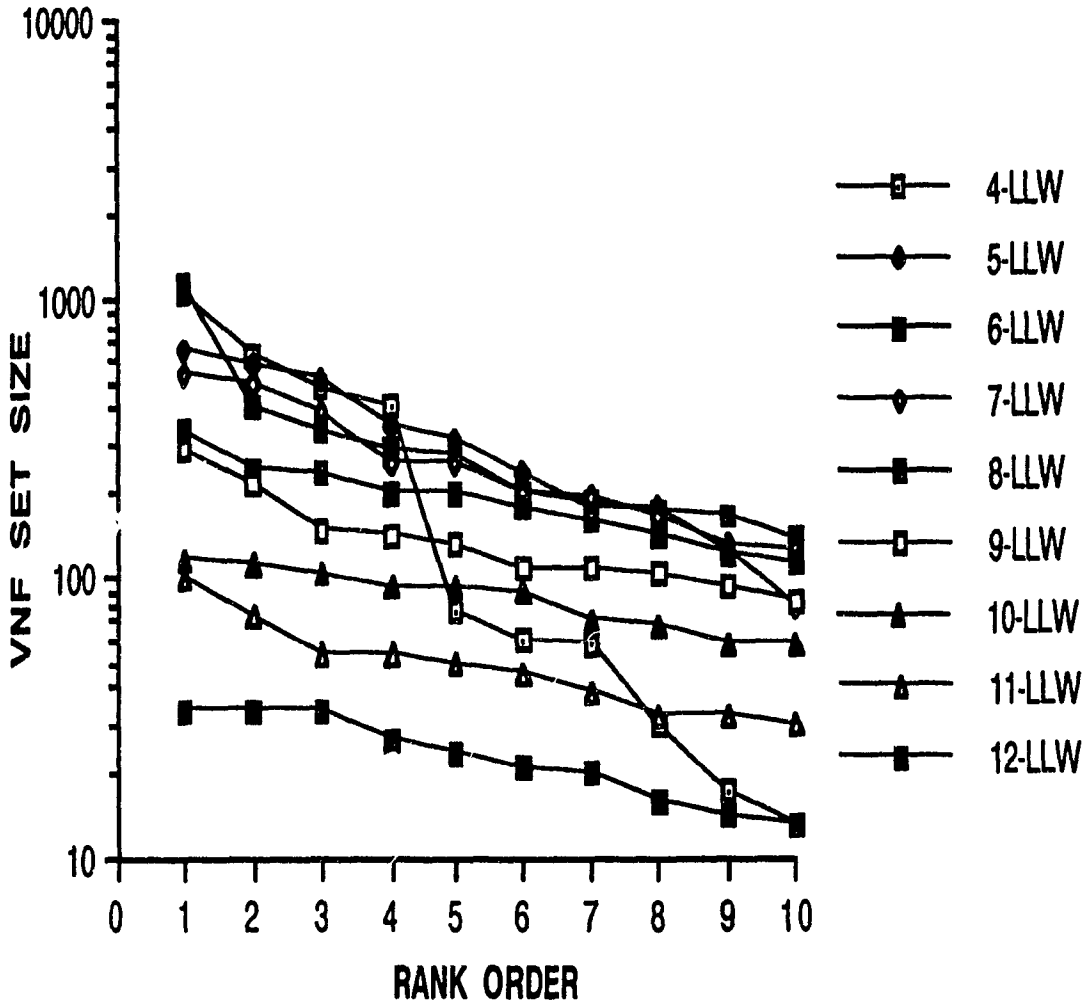

Figure 6.5 VNF set size as a function of rank order. The set sizes of the ten most populated VNF frames for 4-, ..., 12-letter-long words defined in the OPD are given as a function of their rank order. These plots illustrate that the simple exponential decay in set size as a function of rank order that was observed in Figure 6.3 does not hold for 4-letter-long words. Abscissa: Rank order, 1 being assigned to the largest set size and 10 being assigned to the smallest set size. Ordinate: Base 10 logarithm of the set size in words.

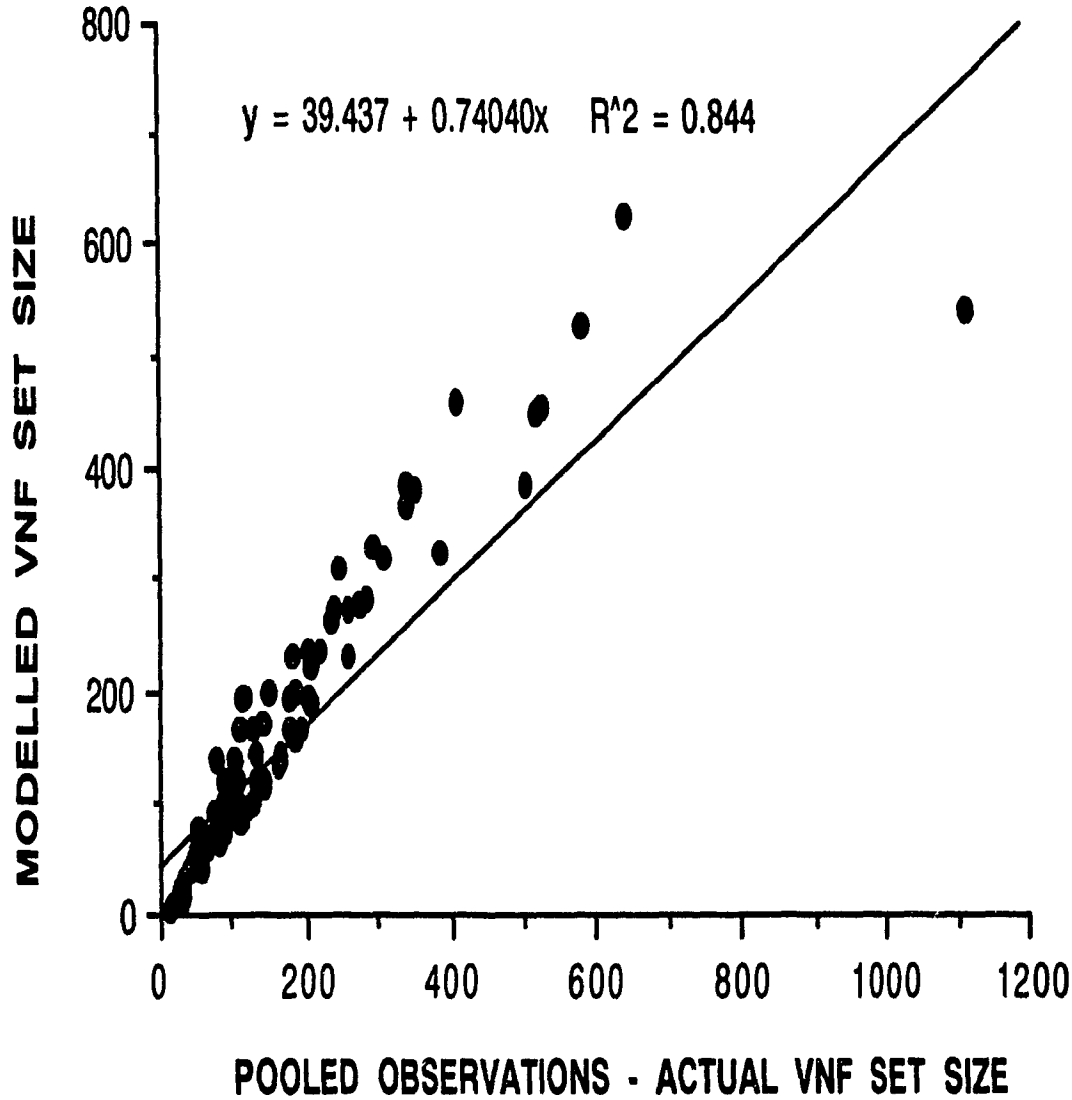

Figure 6.6 Scatter plot of the observed set sizes for the top-ten most populated VNF frames found in 5-, , 12-letter-long words defined in the OPD and that predicted by Equation 6.1. A single outlier reduces the model's correlation coefficient to a value of 0.844. Ordinate & Abscissa : VNF set size in words.

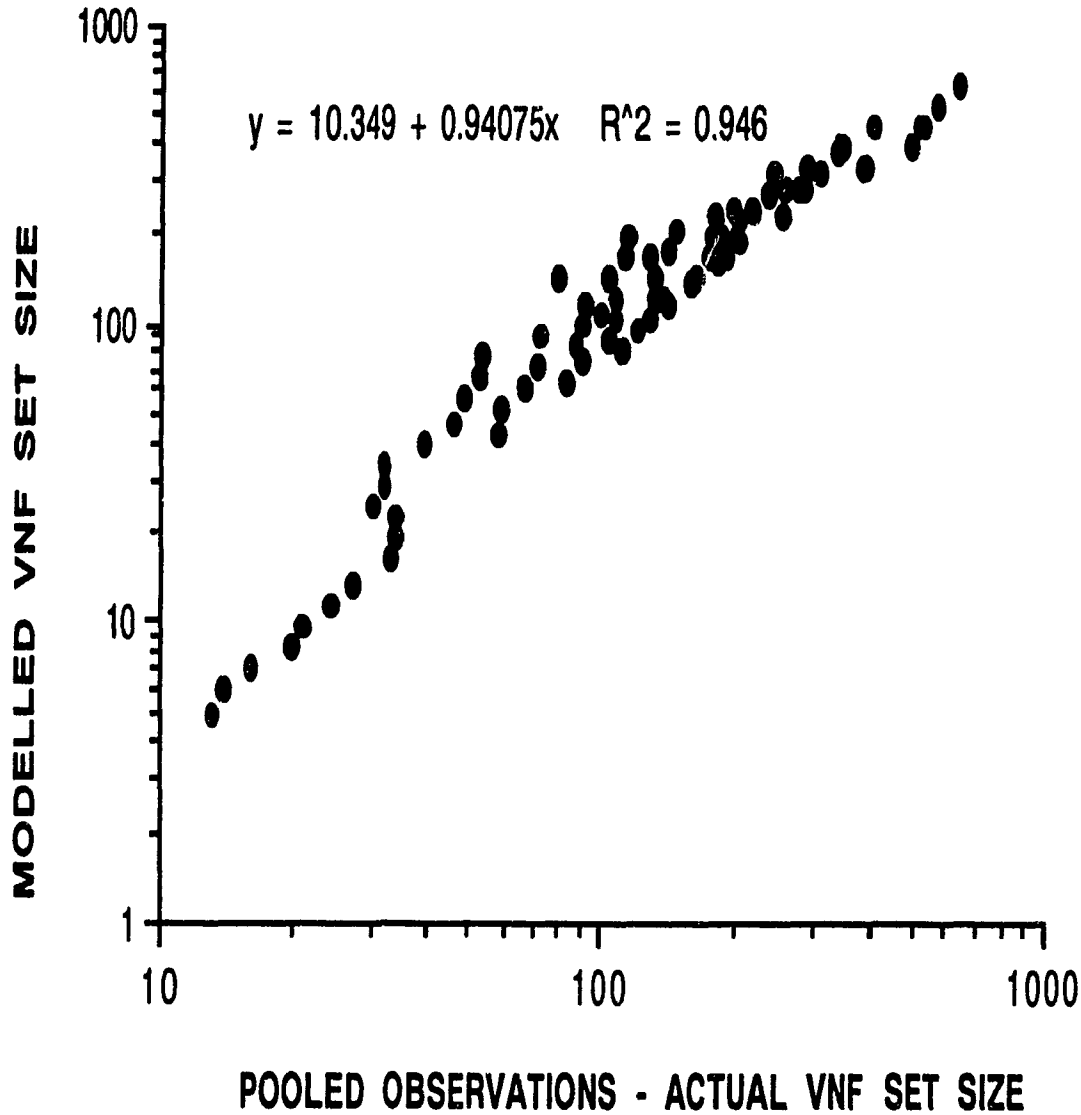

Figure 6.7 Scatter plot of the observed set sizes for the top-ten most populated VNF frames found in 5-, , 12-letter-long words defined in the OPD and that predicted by Equation 6.1. The single outlier has been removed from the data set in this analysis. Filtering this outlying VNF set raises the model's correlation coefficient to a value of 0.946. Ordinate & Abscissa: VNF set size in words.

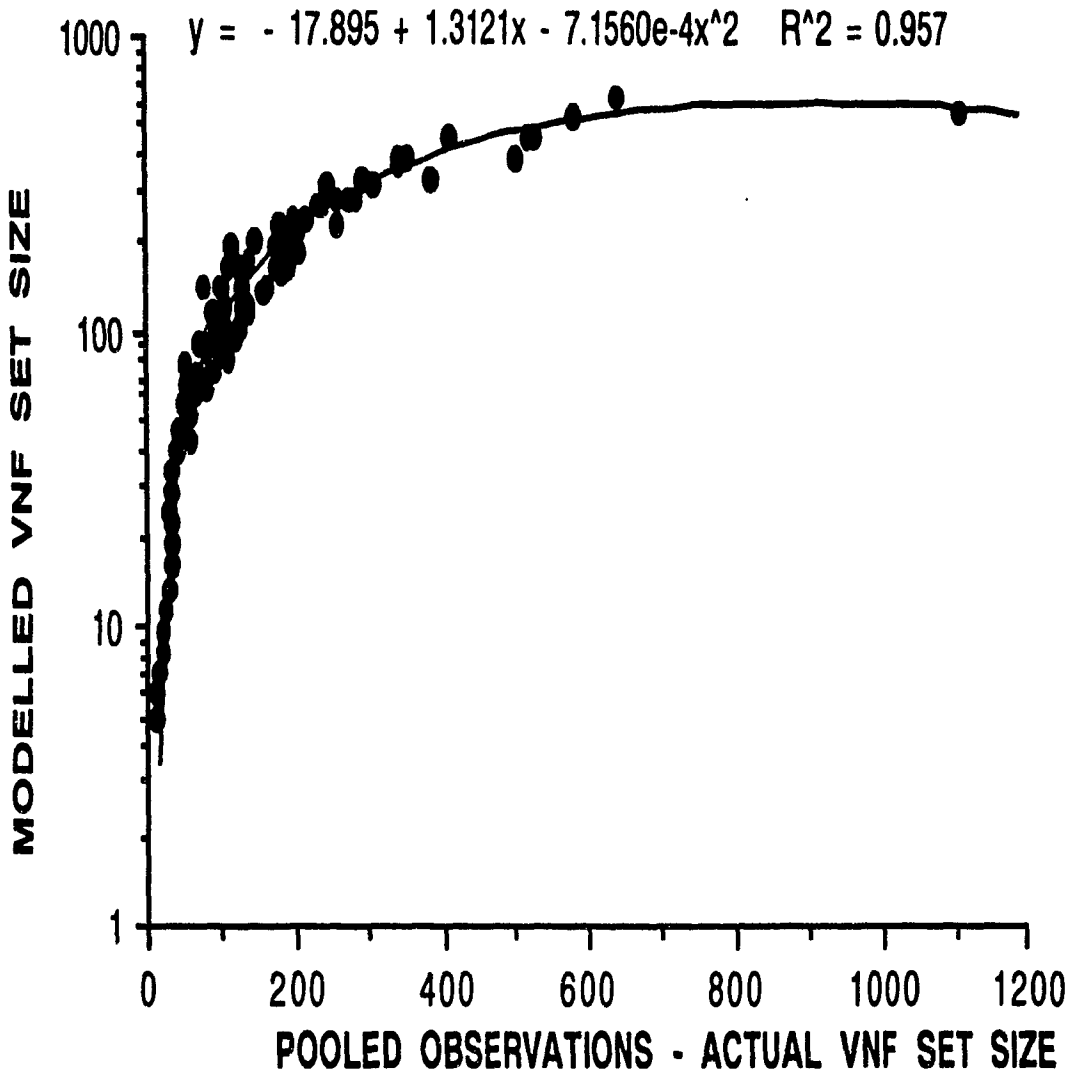

Figure 6.8 Scatter plot of the observed set sizes for the top-ten most populated VNF frames found in 5-, .., 12-letter-long words defined in the OPD and that predicted by Equation 6.1. The model has been reevaluated to demonstrate that the single outlier may indicate a more complex effect than that given by Equation 6.1. The case illustrates the possible presence of a power law relationship between set size and rank-order. The simpler model is preferred until studies on other languages clarify this point.. Ordinate & Abscissa : VNF set size in words. Ordinate: Base 10 logarithm of VNF set size in words.

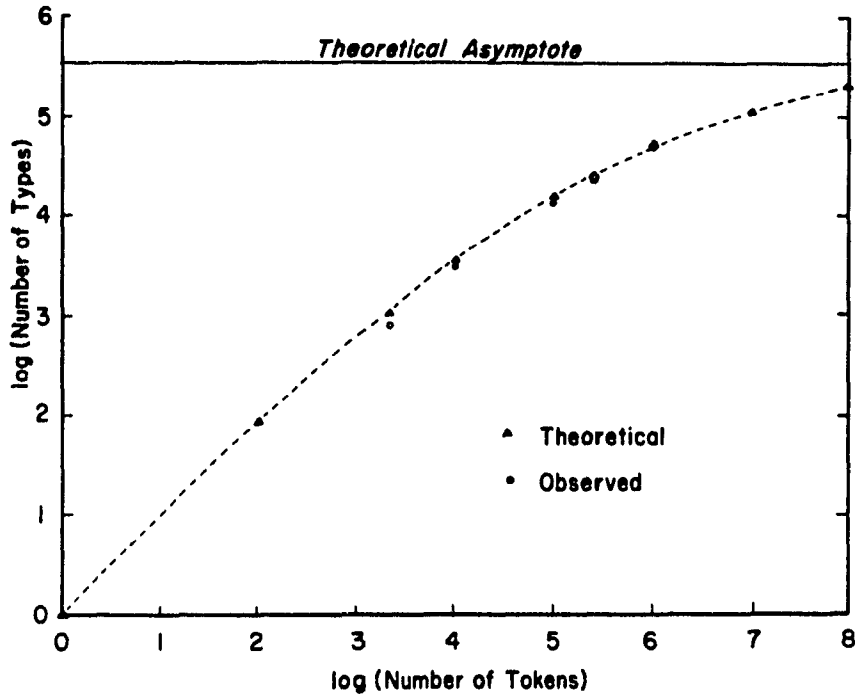

Figure 6.9 The relationship of 'type-counts' to 'token-counts' formulated by Kucera [6.4] for natural languages which exhibit log-normal distributions (such as that depicted in Figure 6.1) in their type and token counts. Abscissa: Base 10 logarithm of the number of tokens in words. Ordinate: Base 10 logarithm of the number of types in words. Taken from Kucera [6.4].

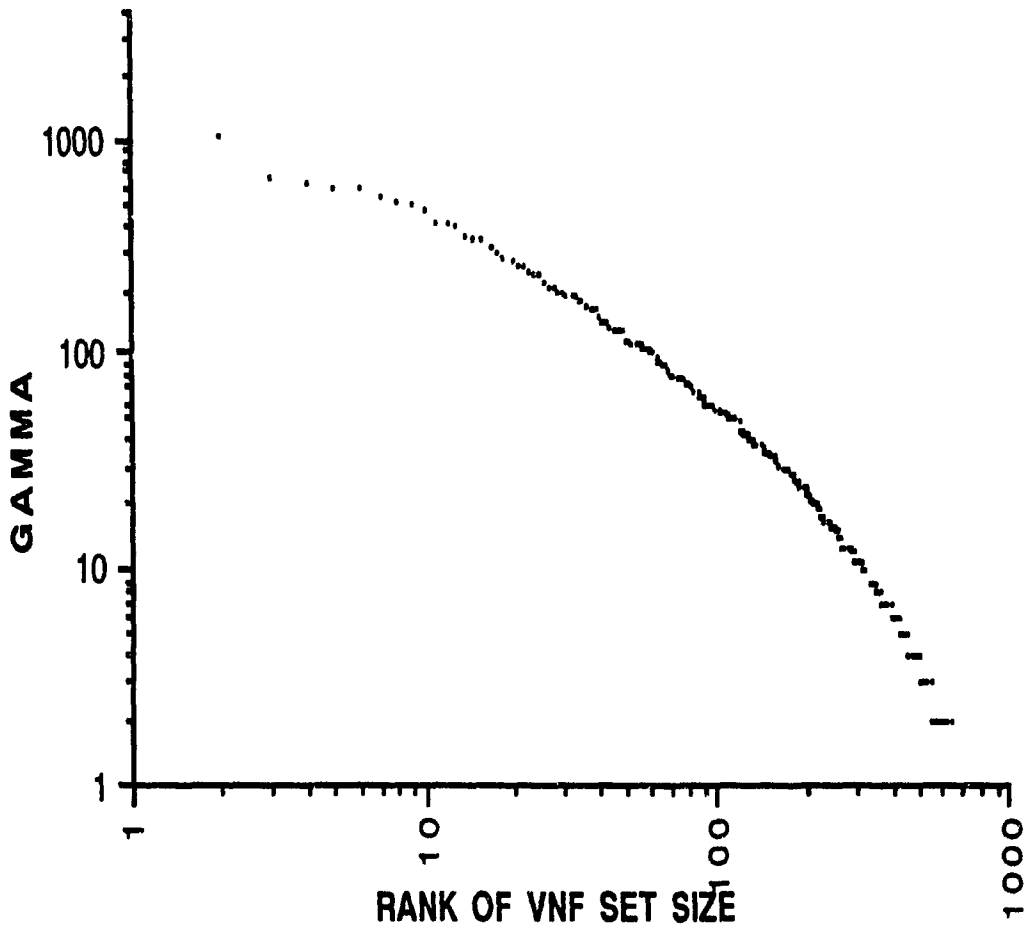

Figure 6.10 The empirically observed relationship between actual VNF set size and the rank of the VNF set. VNF set rank was assigned by sorting VNF set sizes to produce a rank-ordered listing based on descending VNF set size. This figure demonstrates that the effects observed here using VNF set features resemble those obtained by a raw analysis of the lexicon depicted in Figure 6.9. Such curves have been found [6.7, 6.8, 6.9, 6.10, 6.11, 6.12, 6.13] to be typical of natural 'fractal' behavior. Abscissa: Base 10 logarithm of rank-ordered VNF set-size. Ordinate: Base 10 logarithm of the number of words per VNF frame.

6.5 REFERENCES

[6.1] see 3.2

[6.2] see 1.73

[6.3] see 1.74

[6.4] see 2.8

[6.5] see 3.10

[6.5a] see 4.8

[6.6] see 1.48

[6.7] B. West, A. Goldberger, "Physiology in Fractal Dimensions", American Scientist, Vol 75, No. 4, August 1987, 354-65.

[6.8] A. Goldberger, B. West, V. Bhargava, " Nonlinear mechanisms in physiology and pathophysiology: Towards a dynamical theory of health and disease" in Proceedings of the 11th International Association for Mathematics and Computers in Simulation. B. Wahlstrom, R Henriksen, M. Sundby, [Eds.], North-Holland, Oslo, 1985.

[6.9] A. Goldberger, V. Bhargava, B. West, A. Mandell, "On a mechanism of cardiac electrical stability: The fractal hypothesis", Biophysics Journal, 48, 1985, 525-28.

[6.10] P. Peebles, The Large Scale Structure of the Universe, Princeton University Press, Princeton, Ma.,1980.

[6.11] P. Jourdain, in Contributions to Transfinite Numbers, Dover, New York, N.Y., 1955.

[6.12] K. Wilson, "Problems in physics with many scales of length", Scientific American, 241, 1979, 158-79.

[6.13] B. West, V. Bhargava, A. Goldberger, "Beyond the principle of similitude: Renormalization in the bronchial tree", Journal Applied Physiology, 60, 1986, 189-97.

[6.14] S. Zweben, M. Halstead, "The Frequency Distribution of Operators in PL/1 Programs," IEEE Transactions on Software Engineering, Vol. SE-5, No. 2, pp. 91-95; March 1979.

CHAPTER SEVEN
PREDICTING THE SIZE AND LOCATION OF DOMINANT VNF SETS IN ENGLISH

7.1 INTRODUCTION

The set size equation developed in Chapter 6 can be used in conjunction with the Prefix Model developed in Chapter 5. For instance, the Prefix Model allows one to compute the structural forms of the most dominant VNF found in the English language. Similarly, Equation 6.1 can be used to predict the actual set size of a computed VNF form, given its rank, ρ, and word length, l. Much of the work presented in this chapter has been submitted for publication.

7.2 RESULTS

The predicted set sizes for the top-ten most-popular VNF word forms, found in 5-, 6-, 7-, and 8-letter-long words, are given in column 3 of Table 7.1. This information is also depicted in the form of a histogram given in Figure 7.1. This histogram plots the observed VNF set size for each of the top-ten VNF frames found in 5- to 12-letter-long words defined in the Oxford Paperback Dictionary.

The observed VNF set sizes for the lexicon's dominant frames are given in column 5 of Table 7.1. The histogram given in Figure 7.2 plots the predicted VNF set size for each of the top-ten VNF frames found in 6 to 9-letter-long words defined in the Oxford Paperback Dictionary.

The difference between the actual VNF set sizes and those predicted by Equation 6.1 for the top-ten VNF classes found in 5 to 12-letter-long words is depicted in Figure 7.4. The results depicted in Figure 7.4 demonstrate that this model is not perfect. However these results depict a good fit, over the entire range, of observed set-sizes. Furthermore there is also no trace of a systematic error term in these results. The source of the model's residual error is a topic for further research.

Let the set A = { 4, 9, 10, 5, 6, 12, 13, 8, 18, 21 }, be the 5-letter-long VNF kernel which will be used as the basis for the simulations presented in this chapter. This kernel which corresponds to VNF_2 frames: { **CCVCC**, **CVCCV**, **CVCVC**, **CCVCV**, **CCVVC**, **CVVCC**, **CVVCV**, **CVCCC**, **VCCVC** } is composed of the top-ten rank-ordered VNF word structures.

The dominant VNF structures, predicted on the basis of this kernel for a lexicon containing 6- to 9-letter-long words, are given in column 3 of Table 7.2. In contrast, the observed top-ten VNF frames, for words of lengths 6..9, are given in column 5 of Table 7.2.

The predicted values listed in column 3 of Table 7.2 are generated by successively doubling the VNF address of the previously generated top-ten word frame. If the previous base 10 representation of this VNF structure was an odd number the predicted value of the new VNF address is simply double that of its base. However, if the previous VNF structure was an even number the new VNF structure was doubled and then incremented by one. This simple procedure allows one to keep the same relative-ratio of odd-to-even numbers initially observed in the kernel A, throughout the entire simulation. This procedure also allows one to simply compute the top-ten N-letter-long successors from their top-ten (N-1)-letter-long predecessors. For example; the VNF element 4 found in the kernel A would predict 9 (to be a top-ten VNF structure for 6-letter-long-words), which, in turn, would predict 18 (to be a top-ten VNF structure for 7-letter-long-words), which, in turn, would predict 37...

Figure 7.3 is a histogram which plots the location of the VNF frames tabulated in column 3 of Table 7.2. Figure 7.3 also depicts the predicted set-sizes of these predicted VNF structures. Thus both axes of this Figure 7.3 depict predicted values, while both axes of Figure 7.1 illustrate observed data.

The difference between the location of the actual top-ten VNF frames found in 5- to 12-letter-long words and those predicted by a simple prefix code is shown in Figure 7.6.

In fact, careful observation traces most of the error introduced in Figure 7.3 to the failure of the single seven-letter-long frame **CVVCCVC** to propagate.

While the results illustrated in Figures 7.3 and 7.5 are very good, the ability to trace most of the error observed in the model to the presence of a single non-propagating VNF structure is truly remarkable.

The final simulation presented in this chapter attempts to access the effects of an unrealistic, degenerate case on our results. The kernel A, given above, was used for this simulation. However, the predicted VNF frames were computed by simply doubling the address of each previously predicted structure. As such, only even addresses can be generated by this procedure. The results of this simulation illustrates similar qualitatively effects similar to those observed in Figure 7.3. Of course, these predictions contain substantially more error.

7.3 CONCLUSION

The agreement, both in location of the principle VNF sets and their set sizes, demonstrates that the computation of both the structure and size of the dominant VNF forms, for English, is a simple computational feat.

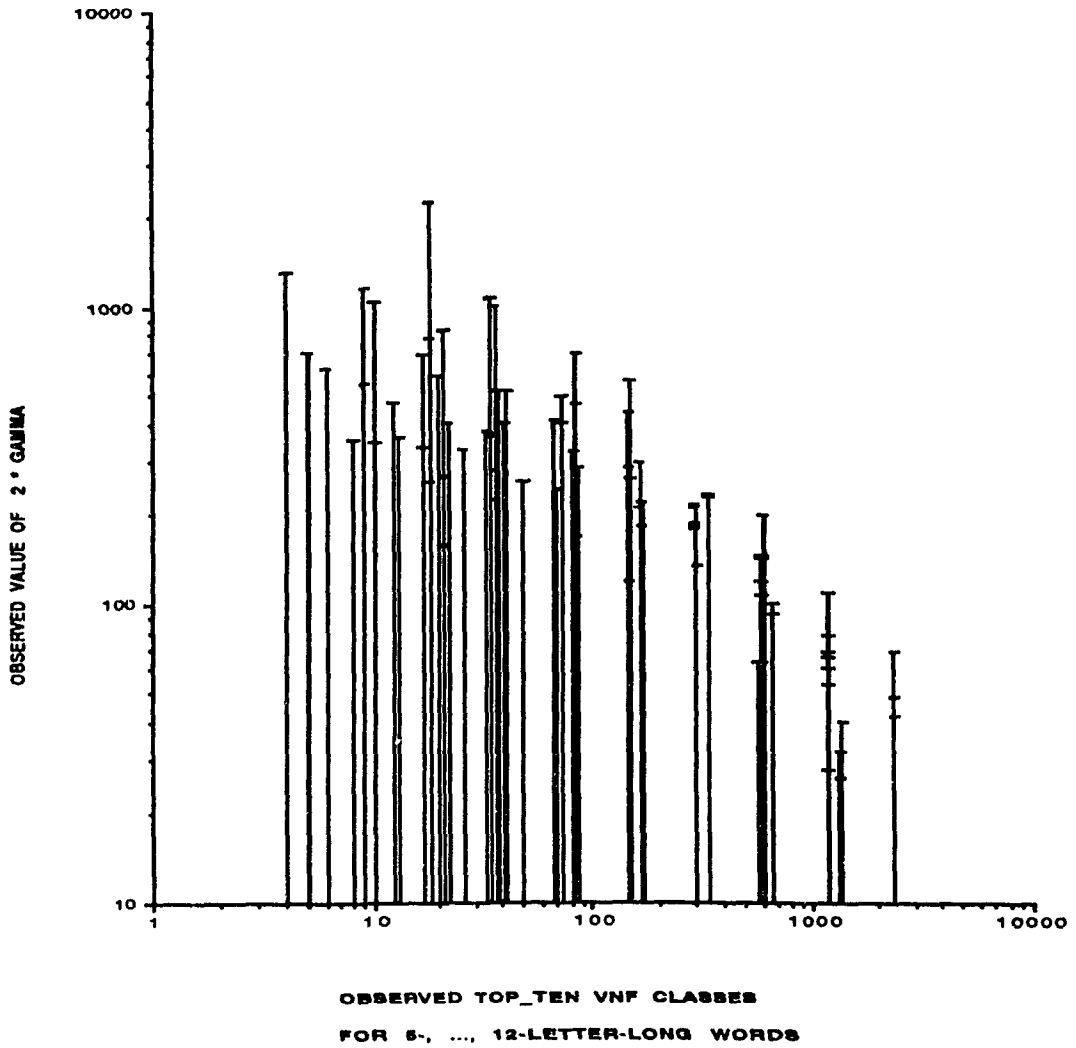

Figure 7.1 Observed value of VNF set size for top-ten 5-, .., 12-LLW. OBSERVED SET SIZE PLOTTED AGAINST OBSERVED VNF CLASS.

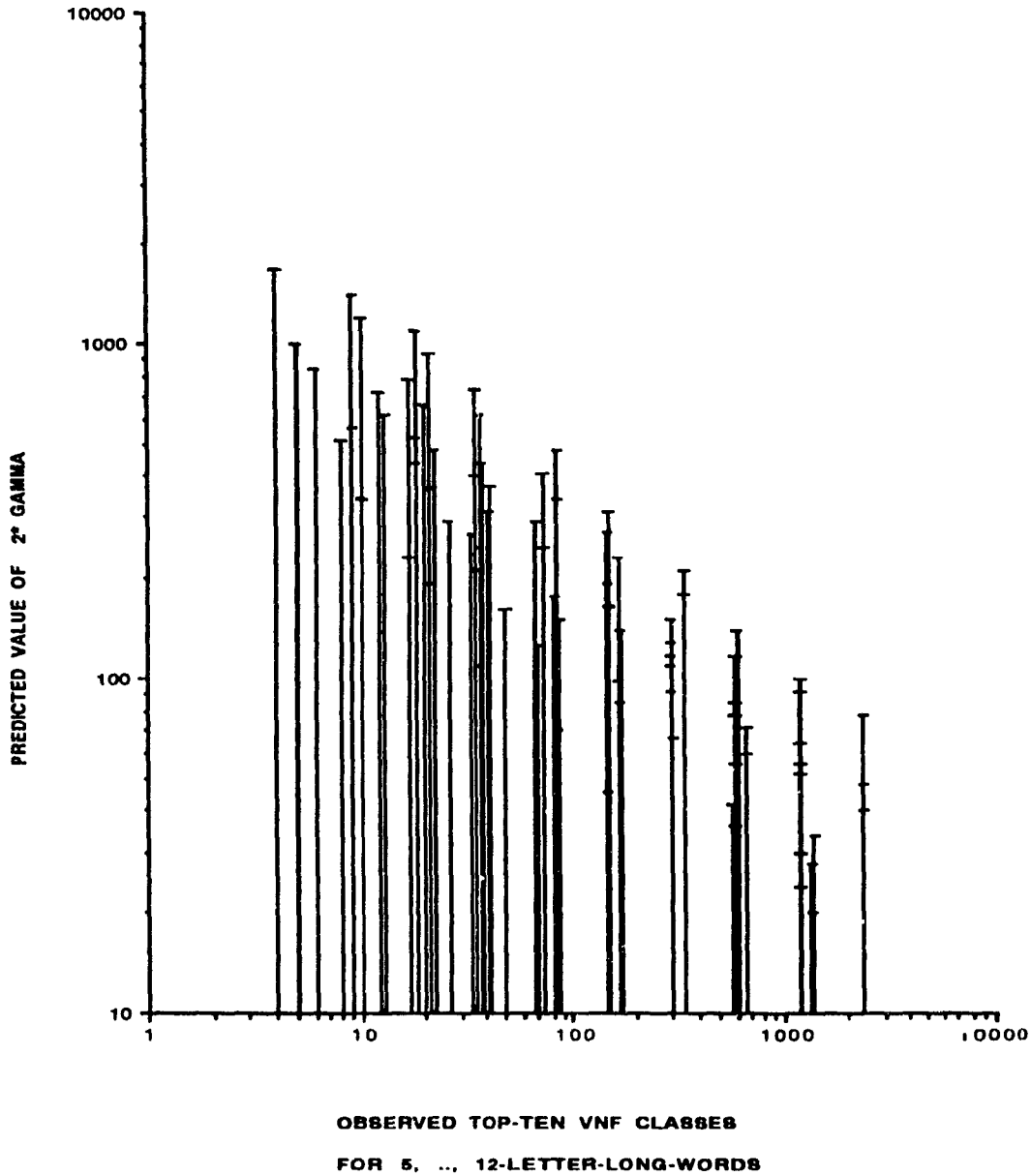

Figure 7.2 Actual VNF locations for top-ten 5-, .. ,12-LLW. ACTUAL LOCATIONS ARE PLOTTED AGAINST SET SIZE PREDICTED BY EQUATION 6.1

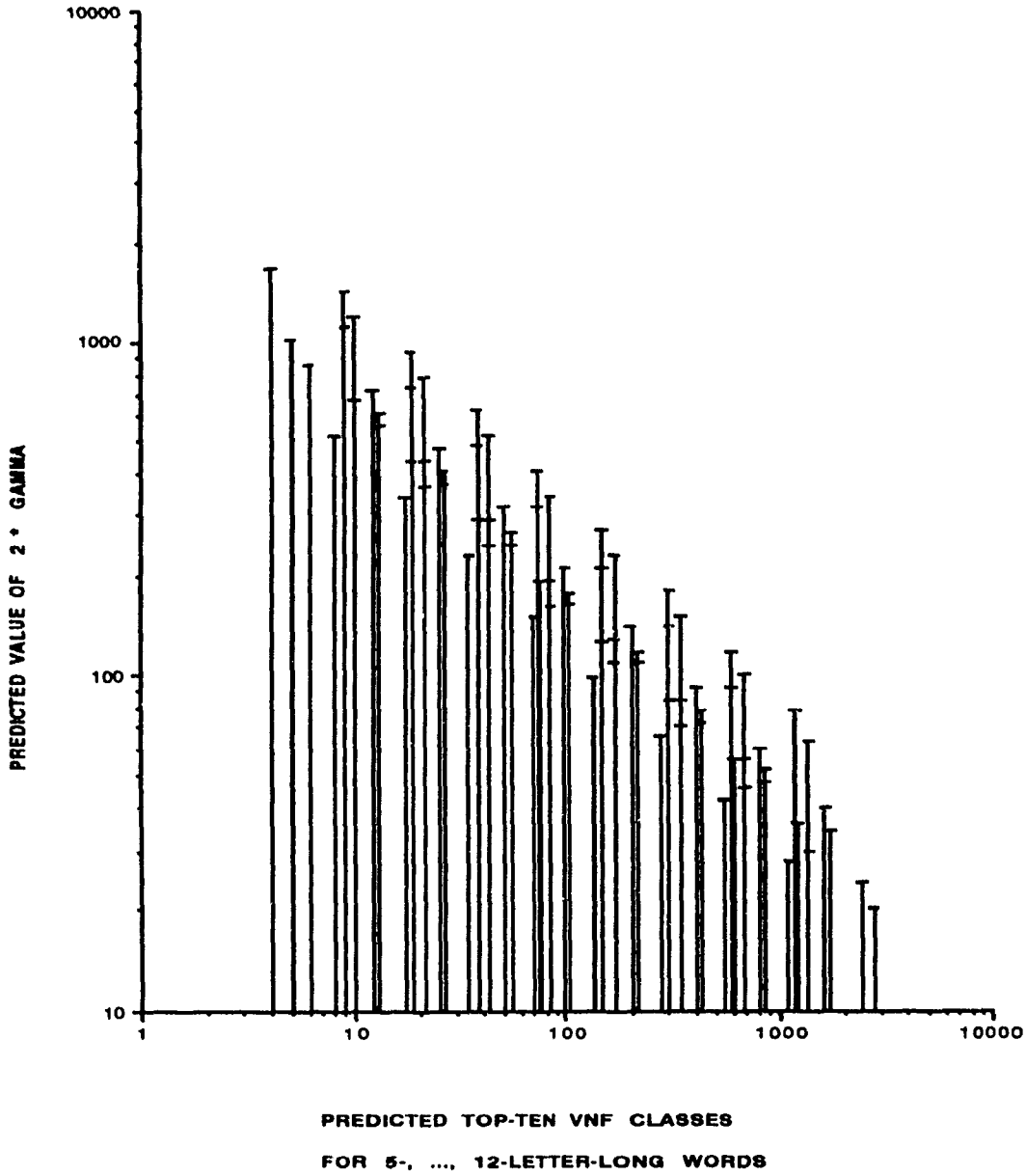

Figure 7.3 Predicted VNF Classes for 5-, .., 12-LLW.
PREDICTED SET SIZES FOR PREDICTED CLASSES

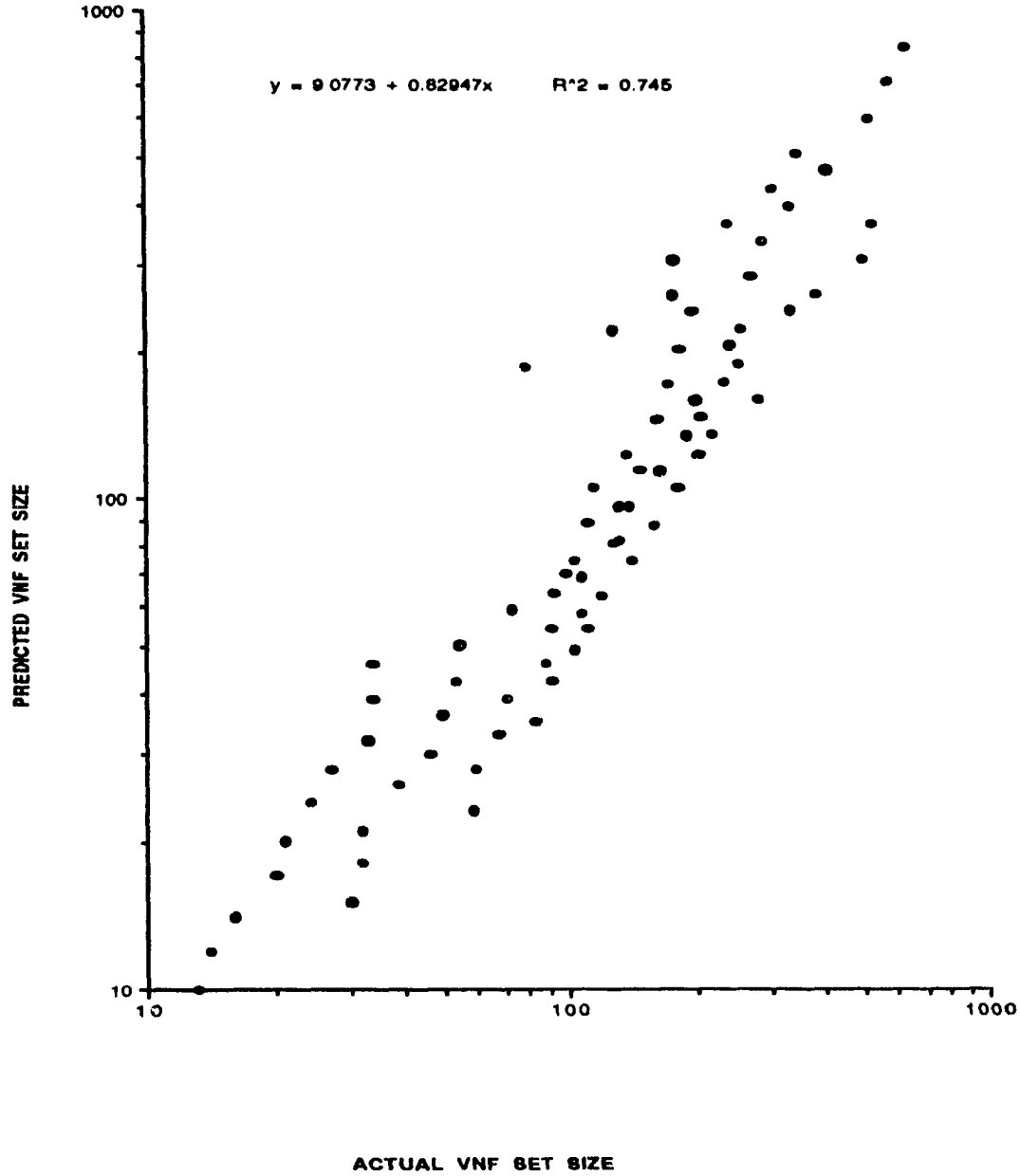

Figure 7.4 Predicted VNF Classes for 5-, .., 12-LLW.
PREDICTED SET SIZES FOR PREDICTED CLASSES

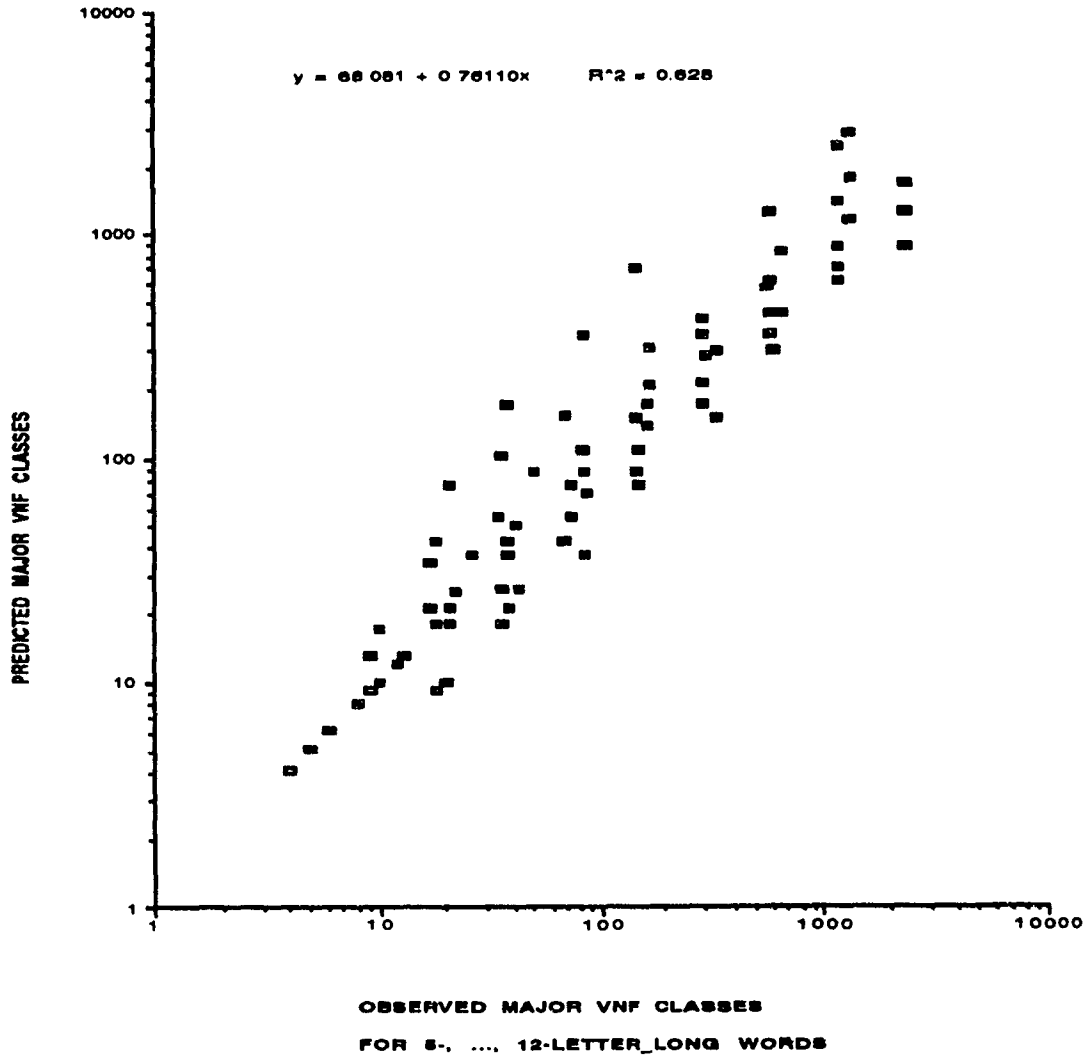

Figure 7.5 Actual vs Predicted VNF set locations.
Data for each of the top-ten VNF word groups found in 5-, .., 12-LLW.

Table 7.1

Word Length	Rank order	Predicted Set size	VNF_{10}	VNF Set size
5 Letters	1	833	CCVCC	642
	2	705	CVCCV	581
	3	597	CVCVC	521
	4	505	CCVCV	351
	5	427	CCVVC	309
	6	362	CVVCC	239
	7	306	CVVCV	181
	8	259	CCVCC	178
	9	219	VCCVC	128
	10	185	VCVCV	79

Word Length	Rank order	Predicted Set size	VNF_{10}	VNF Set size
6 Letters	1	551	18	1112
	2	466	21	410
	3	394	17	341
	4	334	20	290
	5	282	9	275
	6	239	22	199
	7	202	36	185
	8	171	10	174
	9	145	26	163
	10	123	37	139

Word Length	Rank order	Predicted Set size	VNF_{10}	VNF Set size
7 Letters	1	364	36	530
	2	308	37	503
	3	261	18	387
	4	221	38	257
	5	187	42	256
	6	158	41	200
	7	134	34	191
	8	113	17	165
	9	96	21	132
	10	81	50	128

Word Length	Rank order	Predicted Set size	VNF_{10}	VNF Set size
8 Letters	1	241	85	340
	2	204	74	244
	3	172	84	234
	4	146	68	205
	5	123	73	203
	6	105	36	183
	7	88	82	160
	8	75	86	141
	9	63	69	120
	10	54	37	111
			* 149	111

* There were two frames tied for the rank of tenth place. Both structures and their sizes are thus listed.

Table 7.2

Word Length 6 Letters	Rank Order	Actual Structure VNF	VNF_{10}	Predicted Structure VNF	VNF_{10}
	1	CVCCVC	18	CCVCCV	9
	2	CVCVCV	21	CVCCVC	18
	3	CVCCCV	17	CVCVCV	21
	4	CVCVCC	20	CCVCVC	10
	5	CCVCCV	9	CCVVCV	13
	6	CVCVVC	22	CVVCCV	25
	7	VCCVCC	36	CVVCVC	26
	8	CCVCVC	10	CVCCCV	17
	9	CVVCVC	26	VCCVCV	37
	10	VCCVCV	37	VCVCVC	42

Word Length 7 Letters	Rank Order	Actual Structure VNF	VNF_{10}	Predicted Structure VNF	VNF_{10}
	1	CVCCVCC	36	CCVCCVC	18
	2	CVCCVCV	37	CVCCVCV	37
	3	CCVCCVC	18	CVCVCVC	42
	4	CVCCVVC	38	CCVCVCV	21
	5	CVCVCVC	42	CCVVCVC	26
	6	CCVVCCV	41	CVVCCVC	50
	7	CVCCCVC	34	CVVCVCV	53
	8	CCVCCCV	17	CVCCCVC	34
	9	CCVCVCV	21	VCCVCVV	75
	10	CVVCCVC	50	VCVCVCV	85

Word Length 8 Letters	Rank Order	Actual Structure VNF	VNF_{10}	Predicted Structure VNF	VNF_{10}
	1	CVCVCVCV	85	CCVCCVCV	37
	2	CVCCVCVC	74	CVCCVCVC	74
	3	CCVCVVCC	84	CVCVCVCV	85
	4	CCVCCVCC	68	CCVCVCVC	42
	5	CVCCVCCV	73	CCVVCVCV	53
	6	CCVCCVCC	36	CVVCCVCV	101
	7	CVCVCCVC	82	CVVCVCVC	106
	8	CVCVCVVC	86	CVCCCVCV	69
	9	CVCCCVCV	69	VCCVCVVC	150
	10	CCVCCVCV	37	VCVCVCVC	170
		VCCVCVCV	* 149		

Word Length 9 Letters	Rank Order	Actual Structure VNF	VNF_{10}	Predicted Structure VNF	VNF_{10}
	1	CVCCVCVCV	149	CCVCCVCVC	74
	2	CVCCVCVCC	148	CVCCVCVCV	149
	3	CVCVCCVCV	165	CVCVCVCVC	170
	4	CVCCVCCVC	146	CCVCVCVCV	85
	5	CVCCVCVCV	150	CCVVCVCVC	106
	6	CVCVCVCCV	169	CVVCCVCVC	202
	7	VCCVCCCVC	293	CVVCVCVCV	213
	8	CVCVCCVCC	164	CVCCCVCVC	138
	9	CVCVCVCVC	170	VCCVCVVCV	301
	10	CCVCVCVCV	85	VCVCVCVCV	341

* There were two frames tied for the rank of tenth place. Both structures and their sizes are thus listed.

CHAPTER EIGHT
WORD WEBS

8.1 FOREWORD

The previous three chapters of this thesis focused on the use of two complementary models which have been coupled to allow one to predict both the dominant lexical frames of English language words and their set-size. Chapters 4 and 5 demonstrated the use of a simple prefix code model in predicting the dominant lexical structure of English Language words. Chapter 7 demonstrated that these simple models when coupled account for the principle macroscopic characteristics of English Language word structure.

This chapter focuses on the use of a microscopic model to describe in detail the words that conform to a given VNF. A structural model referred to as a 'word web', was developed for this task and has been described elsewhere [8.1].

A word web condenses word structure by exploiting the occurrence of common natural language suffixes. A word web allows more than one prefix form to share common suffixes. In fact, each natural language derivative may be explicitly constructed only once in such structures.

A word web may be represented as a special form of a directed graph. The vertices in a word web are of two types: terminal nodes and starting nodes. Starting nodes are depicted in a word web by a capital letter with a single circle surrounding it. In contrast, terminal nodes are depicted by a capital letter surrounded by two concentric circles. In the text of this chapter we shall let an outlined capital letter, such as **A**, denote a starting node in a word web, while an underlined outlined capital letter, such as **A**, will be used to denote a terminal node in a word web. A word-web may be traversed, starting from any starting node. Traversal may terminate at any terminal node that is reachable from the traversal's starting node.

8.2 INTRODUCTION

Natural languages have a highly context-sensitive form. Chomsky in 1956, [8.2] has described this form as a type 1 language. The exact manner in which a human is able to interpret language has been of interest to psychologists such as Pillsbury [8.3] since at least the the end of the last century. Early studies sought to identify the perceptual orthographic clues that may underlie our ability to read Woodworth [8.4]. Modern psychologists have sought to determine the degree to which the recognition process involves whole words, letter clusters, or single letters [8.5, 8.6]. Conventional models of word recognition assume multiple levels of stimulus processing. These models, using the constructs of information theory, attempt to account for perceptual ability in terms of the redundancy of natural languages [8.7, 8.8, 8.9]. These paradigms of perception assume that reading involves learning a sophisticated guessing procedure [8.10] or criterion bias [8.11], which allows us to perceive what sensory information alone is insufficient to determine [8.12, 8.13, 8.14, 8.15]. It would appear that it is possible to concisely specify microscopic rules for the synthesis and recognition of English words. These rules complement the macroscopic approach to lexical analysis that was developed in Chapters 4, 5, 6 and 7 of this thesis. Much of the work presented in this chapter has been published elsewhere [8.1, 8.16, 8.17].

8.3 METHODS

The fundamental idea underlying a word web is that any sequence of suffix production rules which is common to more than one prefix is shared or made accessible to all appropriate prefixes [8.1, 8.16, 8.17].

An adjacency matrix, A (i , j), such as that depicted in Figure 8.1, may be used to describe words of a specific structural form. Figures 8.1, 8.2 and 8.3 show the adjacency matrixes for all 2-letter-long words listed in Funk & Wagnalls American Dictionary, FW [8.18,

8.19]. Figures 8.4, 8.5 and 8.6 depict the word webs for the VNF classes, **VV**, **CV**, and **VC** given in Figures 8.1, 8.2 and 8.3.

In the adjacency matrix A (*i* , *j*), given in Figure 8.1, the value of the initial letter of all 2-letter-long valid English word, VEW, is specified by the row index *i*, while the second letter of each word is specified by the column index, *j*. Thus A (*3*,*4*) specifies the 2-letter VEW, " IF " in Figure 8.1.

	B	C	D	F	G	H	J	K	L	M	N	P	Q	R	S	T	V	W	X	Z
A			3			3				3	3			3	3	3			3	
E						3			3	3	3	3							3	
I			3	3							3				3	3				
O			3	3		3		3		3	3			3	3				3	
U												3			3	3				
Y																				

Figure 8.1 Adjacency matrix for all 2-letter-long words defined in Funk & Wagnalls Dictionary [8.19] having the form **VC**. Row Index specifies the first letter of the word while the Column Index specifies the last letter of the word. All **VC** frames defined in [8.19] are demarcated by an ' 3 '.

	A	E	I	O	U	Y
A			3			3
E						3
I			3	3		
O			3	3		3
U						
Y						

Figure 8.2 Adjacency matrix for all 2-letter-long words defined in Funk & Wagnalls Dictionary [8.19] having the form **VV**. Row & Column Indices are the same as used in Figure 8.1.

	A	E	I	O	U	Y
B	3	3				3
C						
D		3		3		
F	3					
G				3		
H	3	3	3	3		
J	3			3		
K	3					
L	3		3	3		
M	3	3	3		3	3
N	3			3	3	
P	3	3	3			
Q						
R		3				
S			3	3		
T			3	3		
V						
W		3				
X			3			
Z						

Figure 8.3 Adjacency matrix for all 2-letter-long words defined in Funk & Wagnalls Dictionary [8.19] having the form **CV**. Row Index specifies the first letter of the word while the Column Index specifies the last letter of the word. All **CV** frames defined in [8.19] are demarcated by an ' 3 '.

The adjacency matrices given in Figures 8.1, 8.2 and 8.3 may be used to construct the word webs given in Figures 8.4, 8.5 and 8.6.

A valid path in the word web given in Figure 8.4 starts at an initial state depicted as a starting node, such as **A**, and terminates at any terminal node reachable from that initial state. For example, all traversals from the initial state **A** form { AT, AS, AR } by a single transition; { AD, AH } by two transitions; { AX, AN } by three transitions, and { AN } by four transitions. Similarly all traversals from the initial state **I** form { IT, IS, IF, ID, IN }. While traversal from the initial state **E** form { EL, EN, EX, EM, EH }.

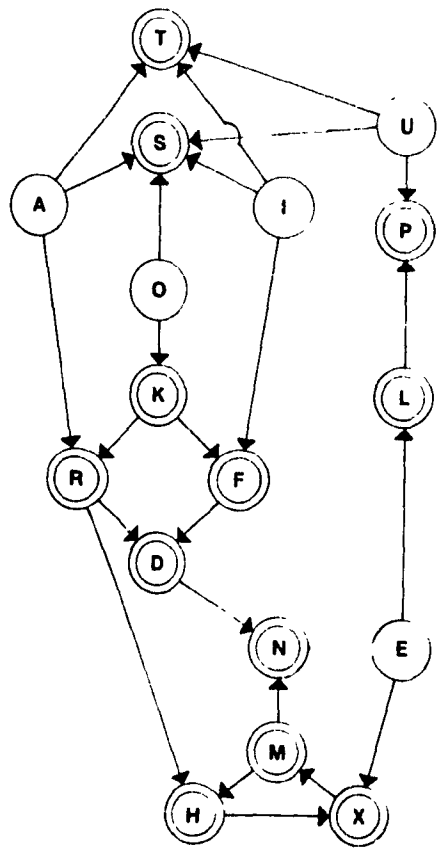

Figure 8.4 Word Web of all 2-letter-long words defined in Funk & Wagnalls Dictionary [8.19] having the form **VC**. All starting nodes are denoted by an encapsulated letter. All terminal nodes are denoted by a letter surrounded by two concentric circles.

All 2-letter-long-words starting with a vowel and ending in a consonant, **VC**, which are defined in FW [8.19], can be represented as valid paths in the word web depicted in Figure 8.4. Thus the same set of words is depicted in the adjacency matrix given in Figures 8.1 and the word web depicted in Figure 8.4.

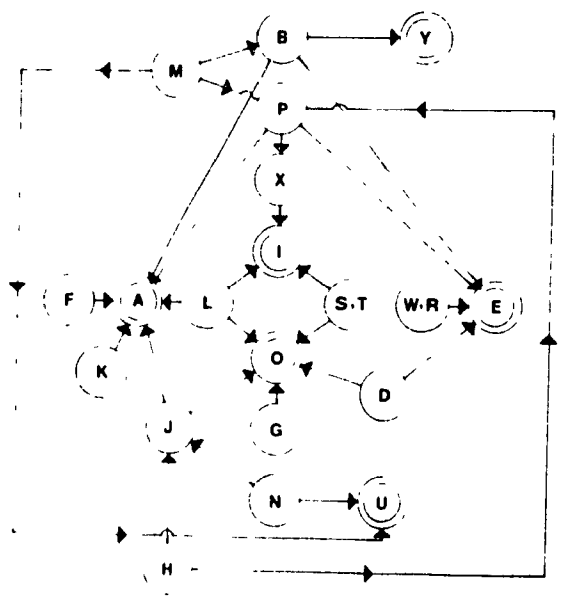

Figure 8.5 Word Web of all 2-letter-long words defined in Funk & Wagnalls Dictionary [8.19] having the form **CV**. All starting nodes are denoted by an encapsulated letter. All terminal nodes are denoted by a letter surrounded by two concentric circles.

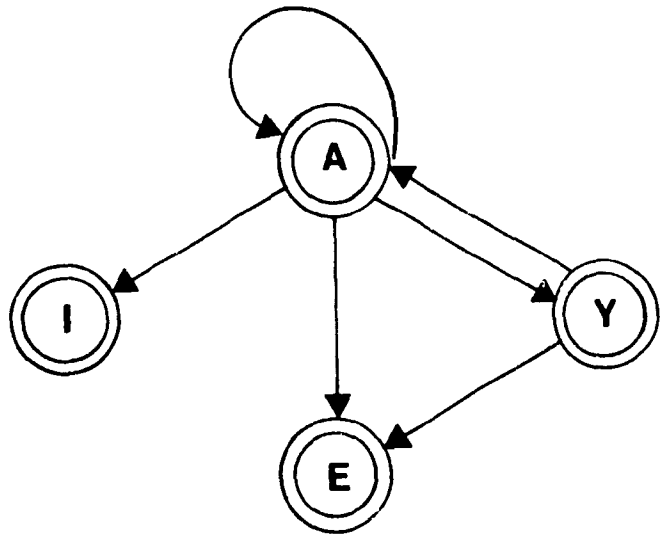

Figure 8.6 Word Web of all 2-letter-long words defined in Funk & Wagnalls Dictionary [8.19] having the form **VV**. All starting nodes are denoted by an encapsulated letter. All terminal nodes are denoted by a letter surrounded by two concentric circles.

8.4 FORMING WORD WEBS

There are many ways of constructing a word web from an adjacency matrix. The following section briefly outlines one of these and gives a simple example of its use on a fictitious case.

Given any adjacency matrix, such as that depicted in Figure 8.7, we can compute its occupancy, as measured by its row and column sums. It is then a straightforward matter to apply elementary row and column operators to the task of permuting the matrix into an upper triangular form, such as that illustrated in Figure 8.8. In this form both the row and column-sums are ranked in order of their descending occupancy. (The adjacency matrix may be condensed or filtered at this point by deleting all rows and columns whose occupancy equals zero or falls below some threshold.) The columns in

Figure 8.8 represent the terminal nodes in a word-web, while the rows of this matrix represent the starting nodes in a word web.

PREFIX	B	C	D	F	G	H	I	J	K	L	M	N	P	Q	R	S	T	V	X	Z	
BA			*		*												*				3
CA	*		*		*							*	*		*		*				7
TA	*		*		*						*	*	*		*		*		*		9
BO	*				*								*						*		4
BU			*		*						*	*					*				5
	3		4		5						2	3	3		2		4		2		

Figure 8.7 Hypothetical adjacency matrix. Column sums are given in the last row of this matrix while row sums are given in the last column of this matrix. Column headers represent terminal nodes in a word web. Row headers represent starting nodes in a word web

	G	D	T	B	N	P	M	R	X	
TA	*	*	*	*	*	*	*	*	*	9
CA	*	*	*	*	*	*		*		7
BU	*	*	*		*		*			5
BO	*			*		*			*	4
BA	*	*	*							3
	5	4	4	3	3	3	2	2	2	

Figure 8.8 Condensed version of the adjacency matrix given in Figure 8.7. The rows and columns of this matrix have been permuted so as to place them in descending order of occupancy. Column sums are given in the last row of this matrix while row sums are given in the last column of this matrix.

By treating each row in Figure 8.8 as a set of elements we may compute their degree of similarity by computing their set

intersections. The intersection of two sets can be used to isolate a common sub-set if one exists. Let R1, R2,.., R5 denote the names of the sets enumerated in the first, second, .., fifth row of the adjacency matrix given in Figure 8.8. We need to compute ten intersections: R1 ∩ R5, R2 ∩ R5, R3 ∩ R5, R4 ∩ R5, R1 ∩ R4, R2 ∩ R4, R3 ∩ R4, R1 ∩ R3, R2 ∩ R3, R1 ∩ R2 to determine, by brute force, that BA is a subset of BU, CA and TA; BO is a subset of TA; BU is a subset of CA and TA; and CA is a subset of TA. We can make use of this information to construct the Venn diagram shown in Figure 8.9 in which BA and BO are classified as a level-1 set; BU a level-2 set and; CA a level-3 set.

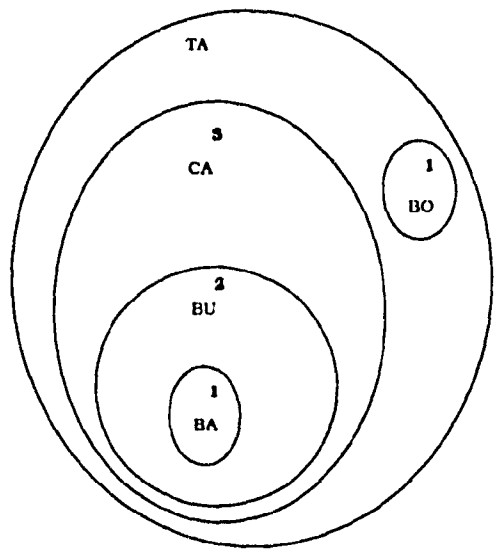

Figure 8.9 Venn diagram illustrating the hierarchy of sub-sets observed in the data given in the adjacency matrix depicted in Figure 8.8.

A word web may be constructed from an adjacency matrix, such as that given in Figure 8.8, by first transposing the horizontal header of the matrix. Each column header in a condensed adjacency matrix is a terminal node in the word web. Once transposed each column header is surrounded by two concentric circles and placed so as to form the vertical backbone of the word web+. This backbone is usually positioned so as to be centered in the viewer's visual field.

Next we introduce all level-1 sets. These nodes are placed within a vertical band on either side of the word web's backbone+ . The paths between the level-1 starting nodes and their terminals are then drawn as directed arcs on the word web. Figure 8.10 depicts the results of applying these steps to the data given in the adjacency matrix shown in Figure 8.8. The names of the word web's starting nodes are specified in the row headers of its adjacency matrix. Starting nodes in a word web are always encapsulated by a single circle.

Level-2 nodes are then added to the word web. Level-2 starting nodes are placed within a more distal vertical band on either side of the word web+. The paths between the level-2 starting nodes and their level-1 subsets are then drawn as directed arcs. Finally, the remaining paths between the level-2 starting nodes and their terminals are drawn as directed arcs. Figure 8.11 illustrates results of applying these steps to the word web outlined in Figure 8.10.

This procedure is repeated until the starting nodes of all levels are entered on the word web. In the example given in Figures 8.8 and 8.9 there are only three levels of nested sub-sets and the word web is completed upon the addition of its level-3 nodes. Figure 8.12 depicts the final word web drawn from the data contained in the adjacency matrix found in Figure 8.8.

+ The vertical component of the node's position may be computed using a geometric mean or some other aesthetic measure.

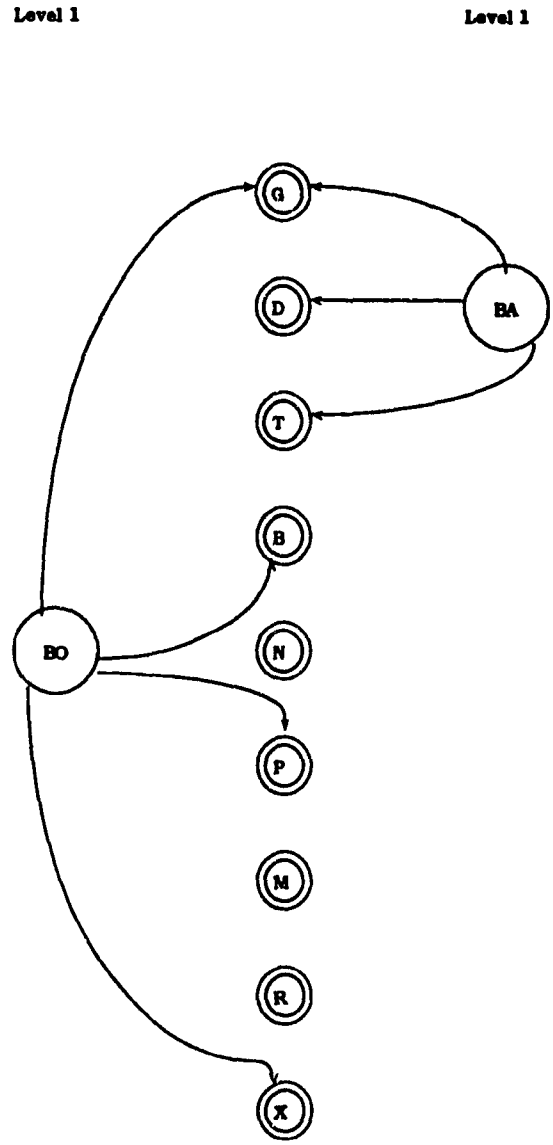

Figure 8.10 Word web drawn from the data contained in the adjacency matrix given in Figure 8.8. This web depicts level-1 starting nodes and their terminals.

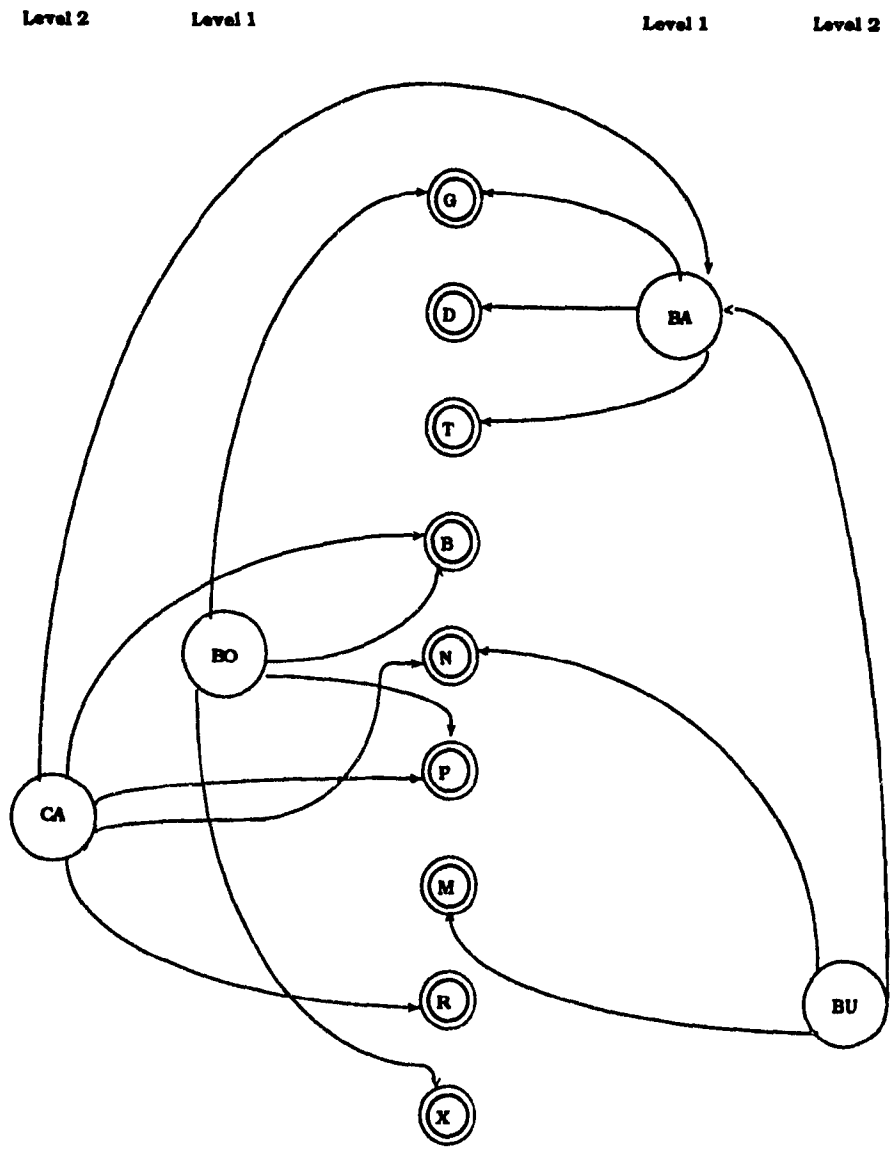

Figure 8.11 Word web drawn from the data contained in the adjacency matrix given in Figure 8.8. This web depicts level-2 starting nodes, their level-1 sub-sets and their terminals.

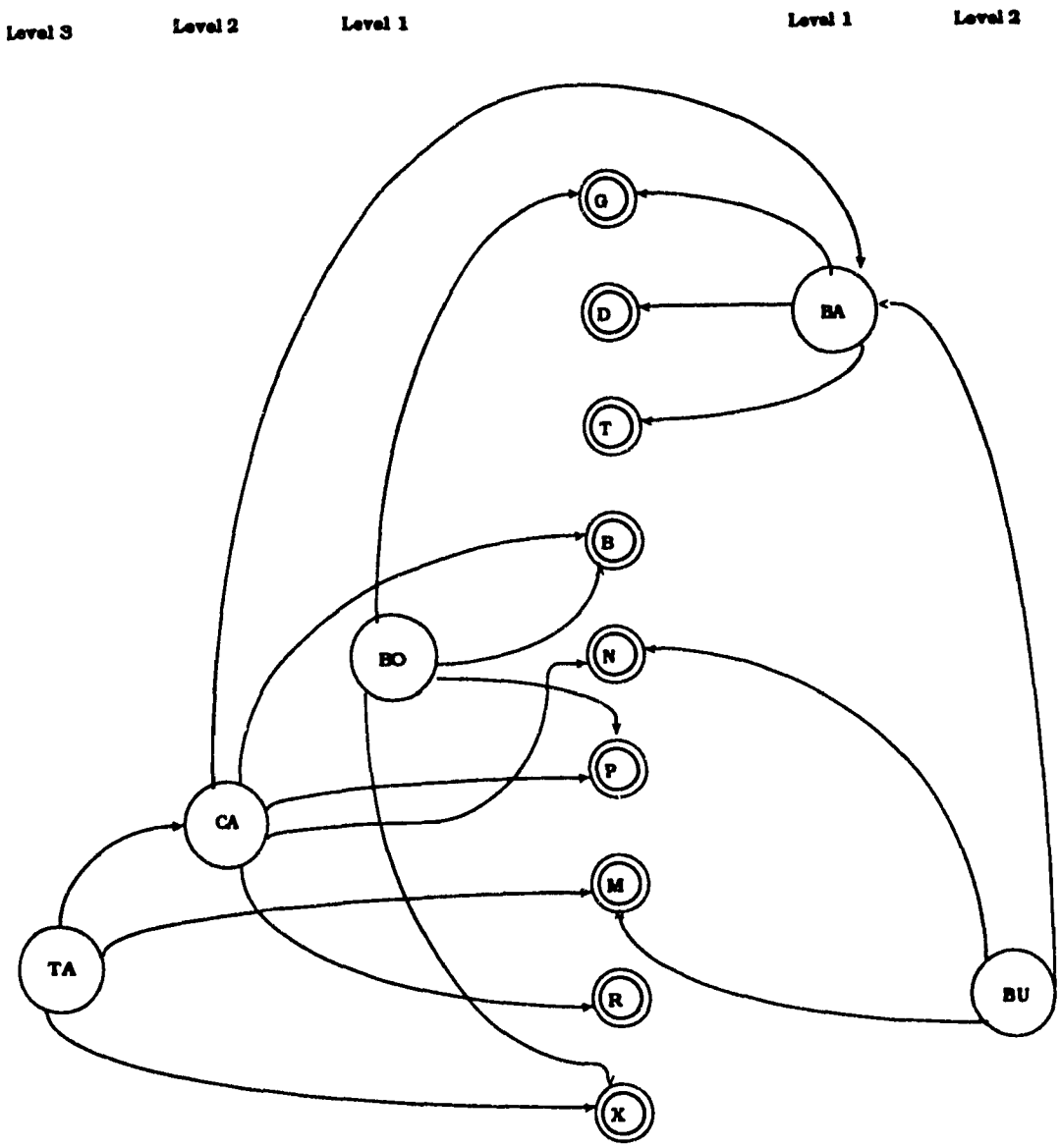

Figure 8.12 Complete word web drawn from the data contained in the adjacency matrix given in Figure 8.8.

The condensed adjacency matrix given in Figure 8.8 is a classic example of a representation that is well suited to human information processing. However in general the brute force method outlined in this section requires $O(n^2)$ set intersections to compute all possible sub-sets of an adjacency matrix $A(n, m)$. One simple way to improve on the computational cost of this procedure is to cull from the set of all possible set intersections those that are not likely to contain significant overlaps. One may use statistical information and a thresholding procedure to isolate those sets which share an arbitrary, *ad hoc*, percentage of elements in common. By treating each row in Figure 8.8 as a binary vector we may compute their degree of similarity by using a dot product calculation to produce the similarity matrix given in Figure 8.13. From this matrix we can readily observe that CA and TA share 7 out of a total of 9 nodes in common. By inspecting Column 5 of Figure 8.9 we observe that the smallest set BA is a subset of BU, CA and TA. We choose to make BA a subset of BU in that it is the smallest of these three sets.

	TA	CA	BU	BO	BA
TA	9	7	5	4	3
CA		7	4	3	3
BU			5	1	3
BO				4	1
BA					3

Figure 8.13 Similarity matrix. The elements of this matrix are computed as the dot product of the rows of the condensed adjacency matrix given in Figure 8.8. For this computation the rows of the adjacency matrix are treated as representing binary vectors. The matrix is symmetric about its diagonal.

One may isolate the major similarities in Figure 8.13 by filtering it in order to retain only those overlaps which account for more than some arbitrary percentage of each sets elements.

For example Figure 8.14 is produced by filtering Figure 8.13 of all set overlaps which account for less than forty percent of the set's size. The size of each set is given as the value of the diagonal elements

in Figures 8.13 and 8.14. Significant overlap (> 40%) in the membership of two sets is denoted by an asterisk in Figure 8.14. By inspecting column 5 of Figure 8.14 we note that there is a significant overlap in the membership of the set BA and the sets BU and CA. A similar inspection of column 3 in Figure 8.14 shows that there is a significant overlap between BU and the sets CA and TA. Column 2 illustrates that the set CA and TA have a significant overlap in their membership while column 4 specifies that there is no significant overlap between the set BO and any other set in this series other than TA.

In all there are only six pairs of candidates that are selected for set intersection as the result of this thresholding procedure. Furthermore the result of the analysis of the selected computation is sufficient to determine that BA is a subset of BU and CA; BO is a subset of TA; BU is a subset of CA and TA; and CA is a subset of TA. Thus we can make use of this information to draw the Venn diagram shown in Figure 8.9 in which BA and BO are classified as a level-1 set; BU a level-2 set and; CA a level-3 set. Thus the six intersections that were deemed deemed worthy of further inspection proved to be useful in the construction of the word web given in Figure 8.12.

	TA	CA	BU	BO	BA
TA	9	*	*	*	
CA		6	*		*
BU			5		*
BO				4	
BA					3

Figure 8.14 Filtered similarity matrix. This matrix is constructed from Figure 8.13. If the ratio of an element's value to its row sum exceeds a threshold of 0.4 the element is defined to be 'significant'. Any element found to be significant in Figure 8.13 is denoted by the presence of an asterisk in this figure. The matrix is symmetric about its diagonal.

The word web construction procedure may be summarized as follows:

0: Obtain the word web's adjacency matrix.
1: Transform the adjacency matrix into its upper triangular form.
2: Condense the transformed adjacency matrix.
3: Compute a similarity matrix from 2.
4: Filter the similarity matrix produced from 3.
5: Compute the set intersection on all pairs of sets deemed significant in step 4.
6: Establish the level of each subset found in 5.
7: Construct the word web's backbone.
8: Overlay the word webs for each sub-set level found in 6.

8.5 OBSERVATION & RESULTS

1. All uni-letter valid English words, VEW, are vowels. They are: a, i, o
2. All two-letter VEW listed in CFW [8.18] contain at least one vowel.

We shall consider these words to fall into three classes. The first case are VEW which are composed of a vowel prefixed to a consonant. The second case is the set of VEW composed of a vowel suffixed to a consonant. A third case is the set of VEW composed of two vowels. We shall denote these cases as: **VC**, **CV**, **VV**; where **V** $\in$ { a, e , i, o, u, y } and **C** $\in$ { b, c, d, f, g, h, j, k, l, m, n, p, q, r, s, t, v, w, x, z }.

A complete list of the 73, 2-letter VEW, listed in the OED and FW, may be tabulated in matrix form in Figure 8.15. All VEW contained in FW are demarcated by an, ' ɜ.' All additional 2-letter VEW found in the OED are demarcated by an, ' ∋ ', while those cited in the OED as obsolete, archaic, or belonging to Middle English are denoted by an ' = ' [8.1].

	A	E	I	O	U	Y	B	C	D	F	G	H	J	K	L	M	N	P	Q	R	S	T	V	W	X	Z
A	3	3	3		∋	3	∋	=	3	∋		3		=	=	3	3			3	3	3		=	3	
E	=	∋	=		=	3				∋		3			3	3	3	3		∋	=	=		=	3	
I		=		∋					3	3					=	=	3			=	3	3		=		
O		=		=	=	∋		=	3	3		3		3		3	3	=		3	3			=	3	
U				=					∋		=	∋				∋	=	3		∋	3	3				=
Y	3	3	∋	=	=				=	=				=	=		=			=	∋	=		=		
B	3	3	=	3	=	3																				
C	3	∋		=	=							=														
D	∋	3		3	∋	=																				
F	3	=				=																				
G	=			3		=																				
H	3	3	3	3	=	∋																				
J	3			3	=																					
K	3		=	∋	=	∋																				
L	3	=	3	3	=	=																				
M	3	3	3	∋	3	3																				
N	3	∋		3	3	=																				
P	3	3	3	=	∋	=																				
Q					=																					
R	=	3		=	=																					
S	=	=	3	3	=	=						∋														
T	∋	=	3	3	=																					
V	=	3		=		=																				
W	∋	∋	=	∋	=	=												=		=						
X	=		3																							
Z		=		=																						

Figure 8.15 Composite adjacency matrix for all 2-letter-long words defined in either the OED, or FW, having the VNF forms **VV, CV, VC, CC**. Row Index specifies the first letter of the word while the Column Index specifies the last letter of the word.

From the adjacency matrix, given in Figure 8.15, it may be clearly seen that 2-letter words of the form, **CC**, have been prohibited throughout the history of the English language. The only living exception to this rule, which is defined as a word in the OED, is the 2-letter injunction, ' SH ' . In fact it would appear that all strings of the form, C^N, are prohibited from forming N-letter long VEW. It is

interesting to note that most of the other possible **CV**, **VC**, **VV** permutations have been valid English words at some point in the history of the language.

Our results have shown that it is possible to generate the 425 3-letter-long VEW contained in FW in terms of the 276 possible permutations specified by **CV**, **VC**, **VV**. Seven transition diagrams of the forms; **CVV**, **VVV**, **CVC**, **VVC**, **CCV**, **VCV**, **VCC** are required for this task. They may be formed by prefixing either a vowel or a consonant to **CV**, **VC**, **VV** to yield the following partitions:

	VV	**CV**	**VC**
V	VVV	VCV	VVC
C	CVV	CCV	CVC

Alternatively they may be formed by suffixing either C or V to **CV**, **VC**, **VV** as:

VV	**CV**	**VC**	
VVV	CVV	VCV	V
VVC	CVC	VCC	C

VEW which may be formed from either a prefix or suffix operation are called 'multiradical' words * . Thus the multiradical VEW, " PIN ", of type **CVC** may be formed as:

CVC ← [C ⇐ **VC**] (for example: PIN ← P + IN), or
[**CV** ⇒ C]→ **CVC** (for example: PI + N → PIN).

Approximately 64% of 3-letter-long VEW are multiradical while the rest can be legally formed only as:

---

* I would like to thank Professor P.F. McCullach of McGill University's Classics Department for his help in coining this term.

CCV ←[C ⇐ CV] (for example: SKY ← S + KY)

in that CC is not a valid 2-letter-long base form.

In this simple recursive manner it is possible to describe all but two of the 2875 4-letter-long VEW found in FW. For this task 23 transition diagrams are needed.

It would appear that in general all N-letter-long English words may be specified with 4 * (N - 1)! -1 transition diagrams. As we have seen in Chapters 4, 5, and 6, relatively few VNF classes are used to represent the majority of English Language Words. Hence relatively few word webs would be needed to specify the actual words found in the more densely populated VNF groups.

Furthermore, it is likely that further research will find that word webs will exhibit similarities that are a function of their related VNF frames. The prefix code model developed in Chapter 5 can be used to predict these related frames

8.6 CONCLUSIONS

This chapter presents a scheme of transition diagrams for finite automata which can synthesize English language words. To date this model has synthesized almost all words of size < 5. The three transition diagrams for all 2-letter-long words contained in FW are given.

For each VNF group there is a word web. The extension of the process described in this chapter to the synthesis of word webs for words of larger size is a simple straightforward process. These webs may be enhanced by transforming them into something akin to augmented transition networks [2.62, 2.92]. Such networks add additional information such as the part of speech or origin of the term constructed by traversing a path in a word web. Such models may be used to enhance our knowledge of the process of reading.

As we will see, in the next chapter of this thesis, our results indicate that one may compute the relative frequency of use [8.17] of

each VEW obtainable from modified transition graphs such as the word webs introduced here. A word's frequency of use will be approximated by calculations based on the product of the positional-probability of each letter within its lexical structure or frame.

On-going research** work seeks to extend the grammar rules, depicted in this chapter as word webs, to cover the problems of concatenation, hyphenation, and hopefully, syllabification.

** I would like to thank William Gillespie and Kent Farrell for their many thought provoking discussions on the work which was described in this chapter and published in [8.1]. I would also like to thank Professor. C.Y. Suen of Concordia University's Department of Computer Science for introducing me to this area of research. This work was partially supported by NSERC grant No. A9372.

8.7 REFERENCES

[8.1]. see 1.50

[8.2]. N. Chomsky, "Three Models for the Description of Language," IEEE Transactions on Information Theory, 2, (3), pp. 113-124; 1956.

[8.3]. see 1.46

[8.4]. R. S. Woodworth, Experimental Psychology, Henry Holt and Company, New York, N. Y., 1938.

[8.5]. D. J. K. Mewhart, "Accuracy and Order of Report in Tachistoscopic Identification," Canadian Journal of Psychology, Vol. 28, pp. 383-398; 1974.

[8.6]. J. Gibson, A. Pick, H. Osser, M. Hammond, "The Role of Graphemephoneme Correspondence in the Perception Words," American Journal of Psychology, Vol. 75, pp. 554-570; 1962.

[8.7]. see 1.48

[8.8]. J. R. Pierce, Symbols, Signals, and Noise, Harper and Row Bros., Inc., New York, N. Y., 1965.

[8.9]. C. Shannon, "Prediction and Entropy of English Language," The Bell System Technical Journal, Vol. 38, pp. 50-64; 1951.

[8.10]. D. Rumelhart, P. Siple, "Process of Recognizing Tachistoscopically Presented Words," Psychological Review, Vol. 81, pp. 99-118; 1974.

[8.11]. D. Broadbent, "Word Frequency Effect and Response Bias," Psychological Review, Vol. 75, pp. 1-15; 1976.

[8.12]. see 1.45

[8.13]. see 1.63

[8.14]. see 1.10

[8.15]. see 1.61

[8.16]. see 1.50

[8.17]. see 1.72

[8.18]. see 3.8

[8.19]. see 3.1

[8.20]. R. Sedgewick, Algorithms, Addison-Wesley, Reading, Massachusetts, 1983.

[8.21]. A. Aho, J. Hopcroft and J. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, Massachusetts, 1974.

[8.22]. S. Kleene, Introduction to Metamathematics, Van Nostrand, Princeton, N.J., 1950.

[8.23]. N. Wirth, Algorithms + Data Structures = Programs, Prentice-Hall, Englewood Cliffs, N.J., 1976.

[8.24]. M. Seidenberg, 'Reading Complex Words', in Linguistic Structure in Language Processing, ed. by G. Carlson & M. Tanenhaus, Kluwer Academic, Dordrecht, The Netherlands, pp 53 - 106, 1989.

[8.25]. P. Utgoff, Machine Learning of Inductive Bias, Kluwer Academic, Dordrecht, The Netherlands, 1986.

CHAPTER NINE
MODELS OF WORD- AND LETTER-FREQUENCY USE

9.1 INTRODUCTION

The study of the English language words presented in this thesis has, up to this point, dealt with structural or morphological features of the written word. The results of this research have shown that even such limited studies can produce fundamental results.

However, for over half a century dynamic philology has focused on models of word use that emphasized the frequency of use, or token-value, of the words, or types, that form a lexicon. Such statistical studies have helped researchers, such as Halstead, study word use in both natural and computer languages.

Similar context-sensitive statistical studies of letter-frequency usage within words can provide insight into the rules of English language concatenation. In particular these studies, can help derive the form of a language's prefix and suffix structures.

As we will see in Chapter 10, the results of such intra-word studies of larger, English language, word-frames, can be heuristically applied to the task of reducing a larger, or derived, word to its root or base.

This chapter proposes a scheme for estimating the frequency of occurrence of English words from the product of position-dependent letter-frequencies. A sampling method is described for computing these frequencies at a given confidence limit from a minimum number of words. Computations for words of different lengths can be normalized under the assumption of a log-normal distribution of word-size within the language. The normalized position-dependent letter-frequency plots for 2-, 3-, and 4-letter-long English words are presented in this chapter. These plots are derived from the set of types of a given length that account for 80% of the observed tokens of the same length within a large corpus [9.1]. The frequency of occurrence of English words can be approximated when modified conditional probability plots are used in conjunction with a scheme of transition diagrams for finite automata, described in the previous

chapter, that synthesize these words. The three transition diagrams for all 2-letter-long English words contained in the Oxford English Dictionary are presented along with statistics on their observed and estimated word-frequencies. Much of the work presented in this chapter is taken from work published elsewhere [9.1, 9.2].

The role of context in problems of pattern recognition and perception has been intensively studied by philosophers and biologists. Today it is also an important area of research in computer science. Since Plato's time, research on this topic has provided valuable insights into the way in which man perceives the world. Neurophysiologists [9.1, 9.2] and cognitive psychologists [9.3, 9.4, 9.5] have yet to understand the manner in which a human is able to preferentially recognize words from random-letter sequences and pseudo-words. In fact, it now appears that one is able to rapidly recognize English words from pseudo-words that are equivalent to Markovian approximations (n-grams) of English words [9.4].

In numerous attempts to understand and simulate the human's ability to read, psychologists, and computer scientists have studied word form [9.6, 9.7, 9.8] as well as spatial and temporal context [9.9, 9.10, 9.11, 9.12, 9.13, 9.14, 9.15, 9.16] at the grammatical level of the word.

For practical purposes, computer scientists have attempted to use contextual information to improve machine character recognition of type font and cursive script [9.17, 9.18, 9.19, 9.20]. Contextual information is also of great value in the error detection and correction techniques [9.21, 9.22, 9.23, 9.24] that are needed to enhance the ability of word processors to spell, format, and transmit information.

There are two traditional approaches [9.25] for the use of contextual information at the syntactic level: the dictionary look-up method and the Markov process. The accuracy of the Markovian method is limited by the availability of *a priori* knowledge about the statistical structure of the language. The dictionary look-up method requires that the word exist in a previously compiled dictionary available to the recognizer. Dictionary methods achieve low error

rates at a cost of large storage demands and high computational complexity. Consequently, hybrid methods [9.26, 9.27, 9.28] have been developed to make use of the best characteristics of both bottom-up Markovian and top-down dictionary methods. This chapter presents the basis for one such hybrid technique.

9.2 THEORY AND RESULTS

Zipf [9.34] and Estoup [9.35] were the first to describe an inverse relationship between the frequency of a word's occurrence and its rank, or order of commonness, in written usage. Since then many authors [9.36, 9.37, 9.38, 9.11] have proposed various models of the constraints that may underlie this relationship.

Under the assumption of a log-normal model of word-frequency distribution, first advanced by Herdan [9.39], it is a simple matter to compute the number of types and tokens of a specific length that are expected to occur within a text of a given number of tokens. In this particular analysis the token distribution observed by Suen [9.14] was used to estimate the percentage to text, I_l, occupied by words of a given length, l.

As shown in Figure 9.1 it is possible, from word-frequency statistics, to compute the minimum number, κ, of types of a given length, l, that can account for a percentage, ϕ, of I_l as:

$$\kappa \;\Big|\; \left(I_l - \sum_{j=1}^{\kappa} f_{(l,j)} \right) \geq \phi\, I_l \qquad (9.1$$

where each term $f_{(l,j)}$ of the sequence $\{ f_{(m,n)} \}$ is the observed relative frequency-of-occurrence of the type specified by $f_{(l,j)}$. In this notation $f_{(l,j)}$ is the j-th term of the l- th sequence $\{ f_{(m,n)} \}$ defined for all positive integers $m = 1, 2, 3, \ldots s$ where s

specifies the length, or number of characters of the largest type found to occur in the text. For each m, the set of types of length m is sorted into the descending rank-order of each type's frequency-of-occurrence within the text. Hence the rank order of a given type $f_{(m,j)}$ within any sequence, m, is specified by the value of ordinal number, j. This ordinal number is defined for all positive integers j = 1, 2, 3, ...z, where z specifies the last and least-frequently-used word of the sequence.

This process is equivalent to partitioning the published rank-ordered frequency-list [9.29], by word-length, into a series of rank-ordered sub-lists. From these sub-lists it is then possible to directly compute the principle components of the overall letter-distribution for words of a given length, l. The accuracy of these statistics in describing the lexicon's behavior for each l is proportional to ϕ and thus determined by the number of terms, κ_l, used in this computation. In a similar manner it is possible to calculate estimates of the position-dependent letter-frequency distributions for a given l from these sub-lists.

This situation is analogous to the decomposition of an absolutely convergent series into absolutely convergent sub-series [9.40]. If the probability of a letter's occurrence within the English lexicon is P_λ, then the absolutely convergent series $\sum P_\lambda$ has the value 1.

$$P_a + P_b + P_c + \ldots\ldots + P_z = \sum_{\lambda = 1}^{\Omega} P_\lambda = 1 \qquad (9.2$$

$\Omega = 26$ for English

Where P_a is the probability of the letter ' a ' occurring in the lexicon. This series may be decomposed, rearranged, and written as:

$$
\begin{aligned}
&P_{1,a} + P_{2,a} + P_{3,a} + \ldots + P_{\phi,z} + \\
&P_{1,b} + P_{2,b} + P_{3,b} + \ldots + P_{\phi,b} + \\
&P_{1,c} + \ldots\ldots \qquad\qquad\qquad\qquad + \\
&\cdot \\
&P_{1,z} + P_{2,z} + P_{3,z} + \ldots + P_{\phi,z} \qquad = 1
\end{aligned}
\qquad (9.3
$$

For example $P_{(3,b)}$ is the probability of the letter ' b ' occurring in three-letter-long words. Let P_{ω} be the probability of occurrence of ω-letter-long words in the lexicon. Then each column-sum of Equation 9.3 is equal to the probability of occurrence of words of a given length within the lexicon:

$$
\sum_{\lambda=1}^{\eta} P_{\omega,\lambda} = P_{\omega} \qquad (9.4
$$

η is the size of the alphabet
$\eta = 26$ for English

and the sum of these probabilities over the entire lexicon equals one:

$$
P_1 + P_2 + P_3 + \ldots P_{\phi} = \sum_{\omega=1}^{\phi} P_{\omega} = 1 \qquad (9.5
$$

Furthermore, each row-sum of Equation 9.3 is equal to the probability of occurrence of a specific letter within the lexicon:

$$
\sum_{\omega=1}^{\phi} P_{\lambda,\omega} = P_{\lambda} \qquad (9.6
$$

In a similar manner Equation 9.3 may be further decomposed into position-dependent probabilities. Position-dependent letter-frequency distributions can also be computed from the row- and column-sums of tables [9.30] of n-gram statistics that have been tabulated for tokens of a given length.

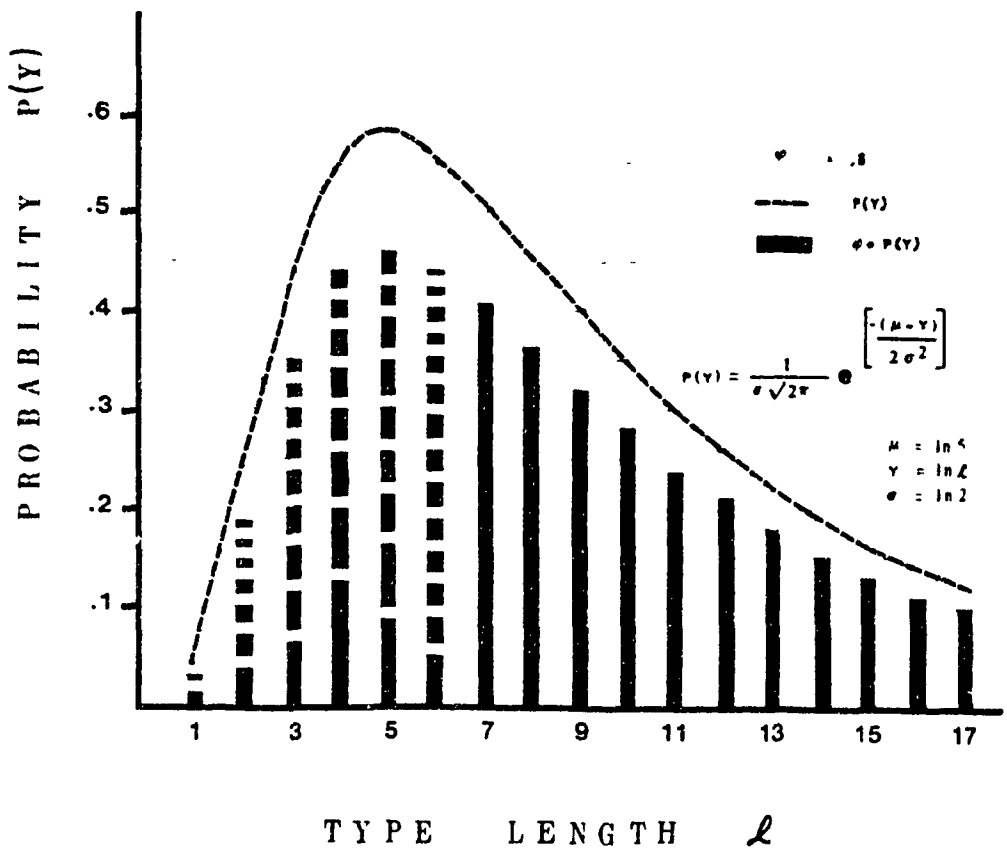

Figure 9.1 The partial decomposition of a simulated log-normal frequency distribution, P(Y). The total proportion, ϕ, of each discrete word-length interval, l , is approximated by the sum of terms in the sequence associated with each l. κ_l is the minimum number of terms in the sequence needed to account for the proportion, ϕ, of the distribution, P(Y), at each word length, l. The values of κ_l associated with l = 1, 2, 3, 4, 5 and 6 in this diagram are respectively: 2, 7, 9, 8, 9 and 14.

9.3 RESULTS

Figure 9.2 depicts the minimum number of types, κ_l, needed to account for a percentage, ϕ, of tokens of a given length, l, in a log-normal word-frequency distribution. These results show that κ_l is linearly related to l at least over the range of the various word-lengths considered in this study.

Figure 9.3, as well as Figures 9.4a, 9.4b, 9.4c, and 9.4d, are plots of the frequency-of-occurrence of the various letters of the English alphabet. When placed in descending rank order, as in Figure 9.3, the observed [9.30] overall occurrence of the letters of the alphabet assumes the form:

$$P(\lambda_I) = Ae^{\alpha I} + Be^{\beta I} \qquad (9.7$$

where $P(\lambda_I)$ is the frequency-of-occurrence of the I-th letter and I is an integer.

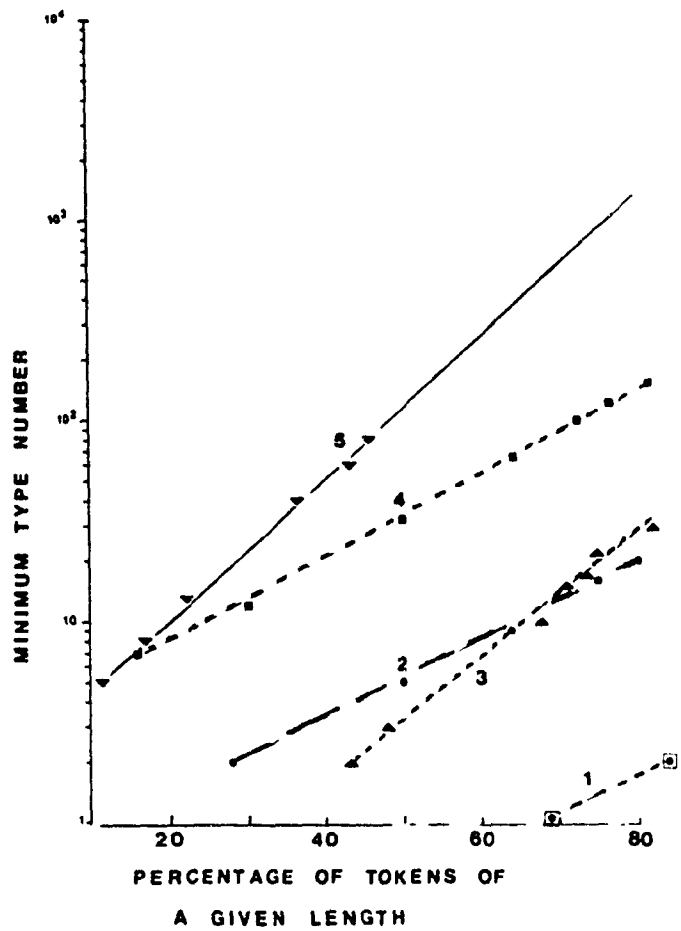

Figure 9.2 The minimum number of terms, κ_l, in a monotonically descending sequence, $\{ f_{(m, n)} \}$, associated with a given word-length, l, as a function of ϕ, for 1-, 2-, 3-, 4-, and 5-letter-long Englis' words.

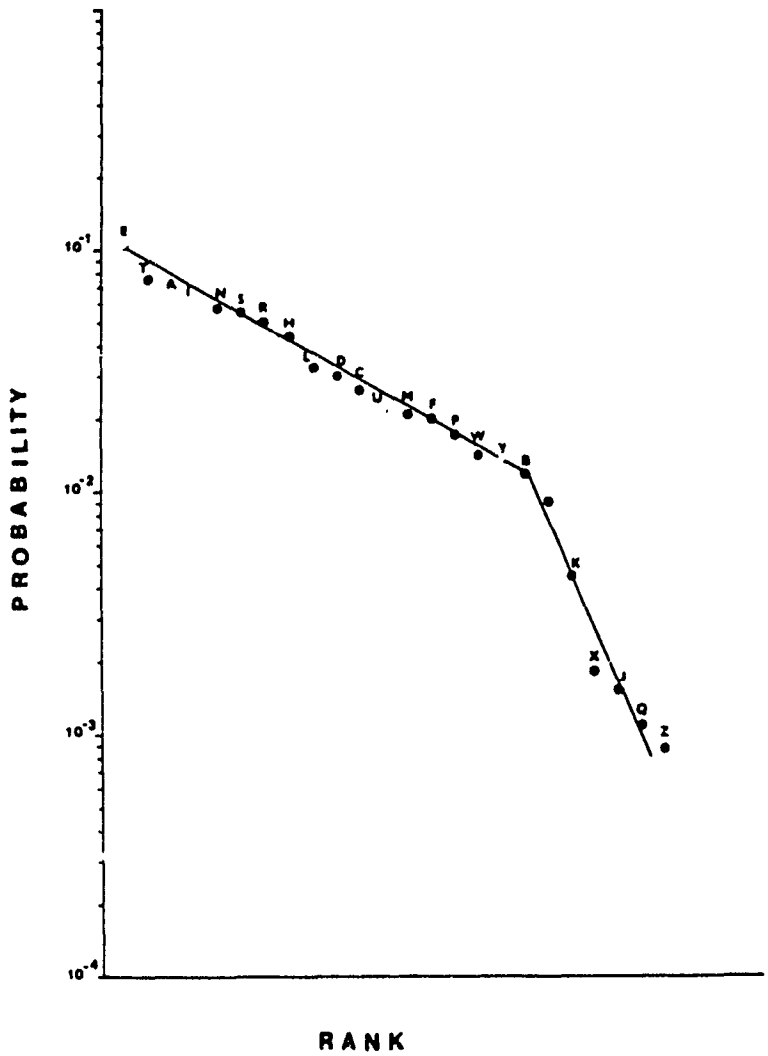

Figure 9.3 Probability of occurrence of the letters of the English Alphabet. The vertical axis is in a logarithmic scale. The horizontal axis is an ordinal scale which is linearly ranked in the order of decreasing frequency-of-occurrence.

In Figure 9.4 we show the decomposition of the overall rank-frequency graph, shown in Figure 9.3, into letter-distributions observed [9.29] for 1-, 2-, 3-, and 4-letter-long English words. The simplest rank-frequency graphs computed from frequency data on words contained in Funk and Wagnall's Standard College Dictionary exhibit an exponential relationship between the observed frequency-of-occurrence of a letter and its rank. This relationship is shown in, Figures 9.4a, 9.4b, and 9.4c, to occur in normalized (ϕ = 80%). samples of 1-, 2-, and 3-letter-long English words. The letter-frequency data computed for 4-letter-long word samples exhibited a twin exponential form when graphed as a function of descending rank order. The more complicated form of the 4-letter-long word rank-frequency plot, shown in Figure 9.4d, (ϕ = 80%), resembles that observed in Figure 9.3 for the overall frequency distribution of the various letters in English.

The letter distribution shown in Figure 9.4 may be further decomposed into position-dependent or fundamental rank-frequency plots. Consider, for example, that the simple exponential relationship observed in Figure 9.4c, between a letter's rank and its frequency-of-occurrence in a 3-letter-long word arises as the sum of the data contained in the three fundamental rank-frequency plots shown in Figure 9.6. From Figure 9.6 we see that the fundamental rank-frequency plots describing the alphabetic distribution observed [9.30] within the first, second, and third positions of these words are all simple exponentials.

Figure 9.5 depicts the position-dependent decomposition of the rank-frequency plot given in Figure 9.4b. The twin exponential forms of the rank-frequency plot observed in Figure 9.4d for 4-letter-long words may be described in terms of the sum of the four position-dependent, rank-frequency plots presented in Figure 9.7. The twin exponential form of Figure 9.4d appears to result from the asymptotic behavior of Figure 9.7 at the lower end of the fundamental rank-frequency plot for the first and second position of 4-letter-long words.

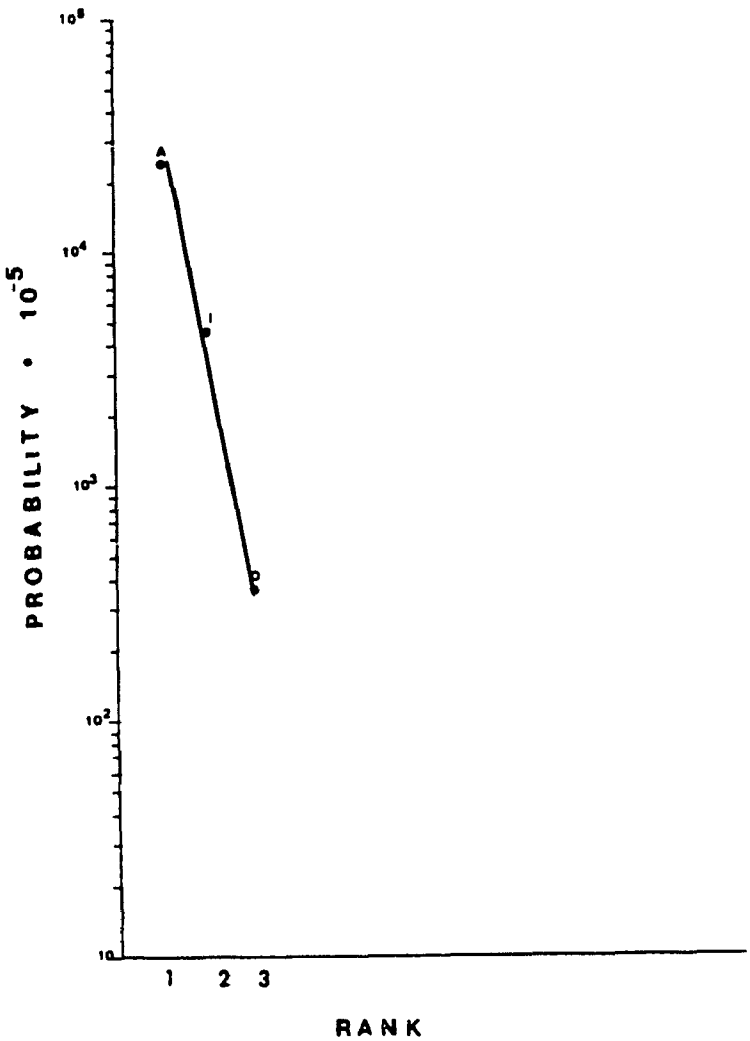

Figure 9.4a The frequency-of-occurrence of the letters of the English alphabet as a function of word-length observed in samples where $\phi = 80$. The vertical axis is in a logarithmic scale. The horizontal axis is linearly ranked in the order of decreasing frequency-of-occurrence. Figure 4a is a plot of the observed frequency-of-occurrence of the various letters of the alphabet found in the 1-letter-long word sample used in this study.

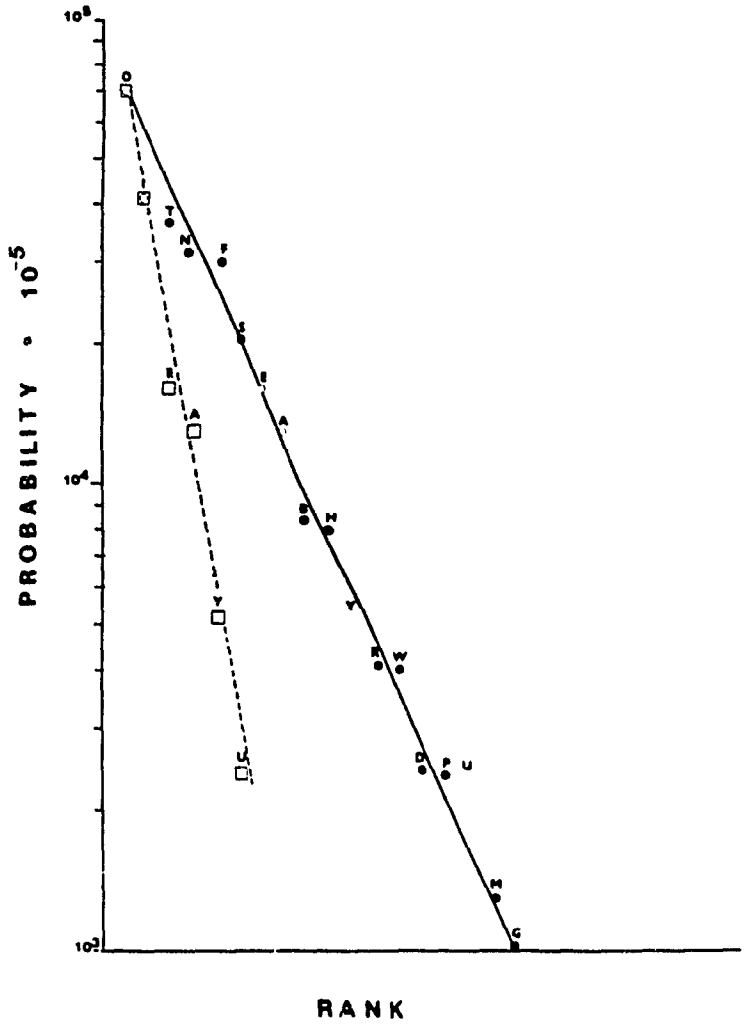

Figure 9.4b The frequency-of-occurrence of the letters of the English alphabet as a function of word-length observed in samples where ϕ = 80. The vertical axis is in a logarithmic scale. The horizontal axis is linearly ranked in the order of decreasing frequency-of-occurrence. Figure 4b is a plot of the observed frequency-of-occurrence of the various letters of the alphabet found in the 2-letter-long word sample used in this study.

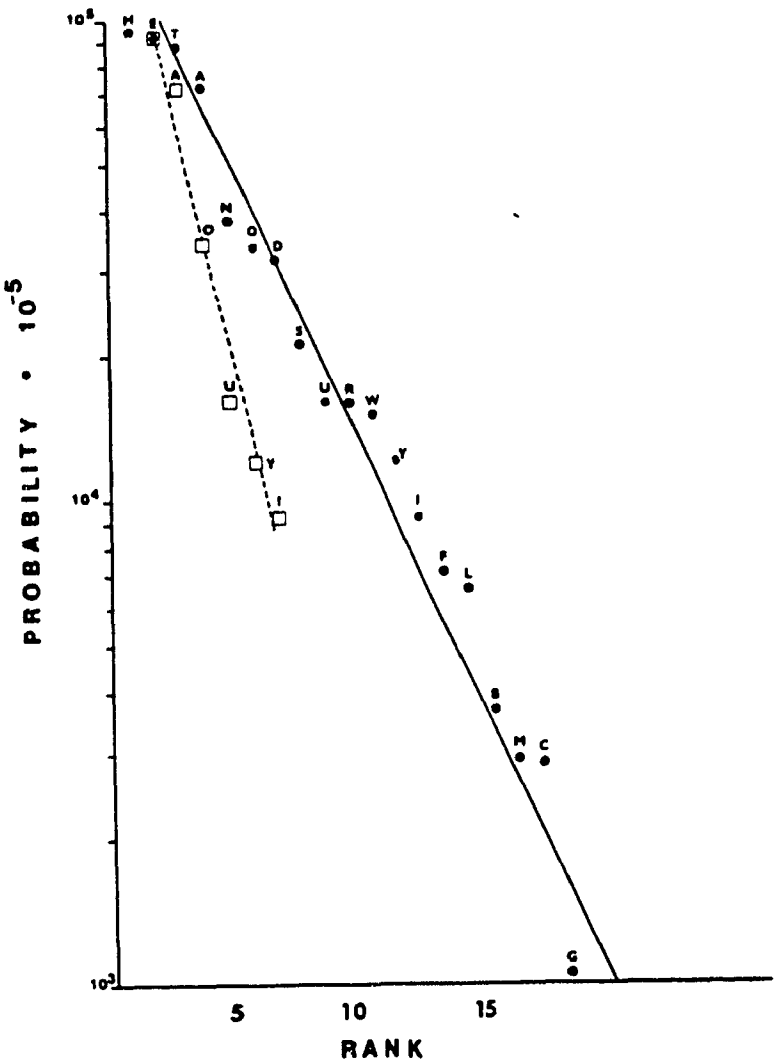

Figure 9.4c The frequency-of-occurrence of the letters of the English alphabet as a function of word-length observed in samples where $\phi = 80$. The vertical axis is in a logarithmic scale. The horizontal axis is linearly ranked in the order of decreasing frequency-of-occurrence. Figure 4c is a plot of the observed frequency-of-occurrence of the various letters of the alphabet found in the 3-letter-long word sample used in this study.

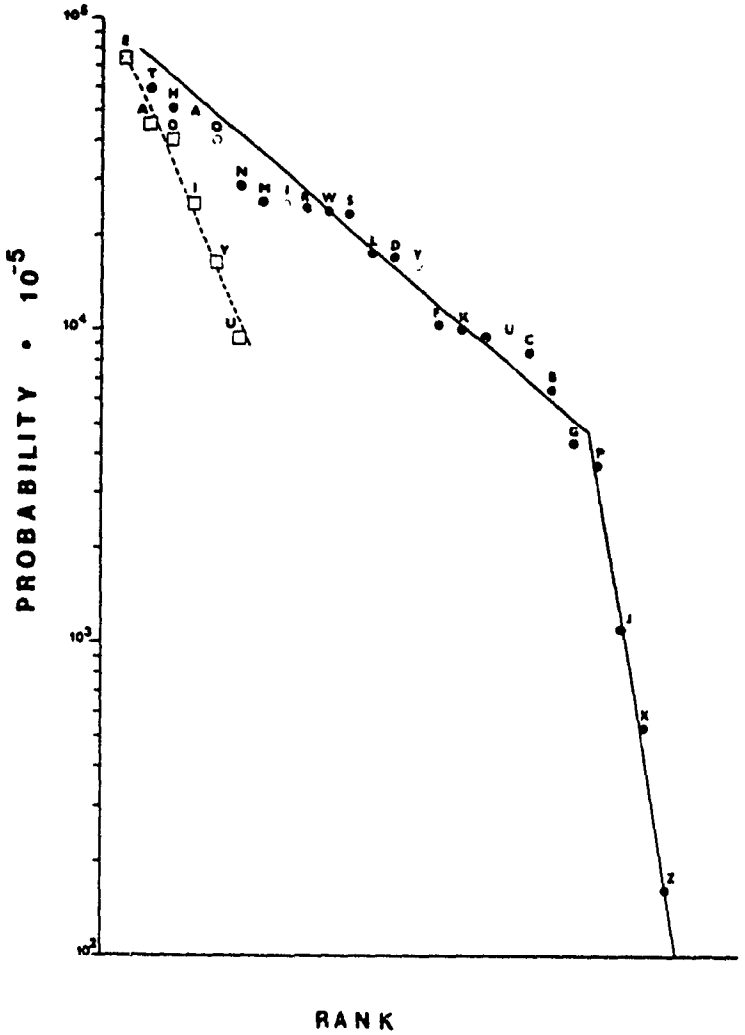

Figure 9.4d The frequency-of-occurrence of the letters of the English alphabet as a function of word-length observed in samples where $\phi = 80$. The vertical axis is in a logarithmic scale. The horizontal axis is linearly ranked in the order of decreasing frequency-of-occurrence. Figure 4d is a plot of the observed frequency-of-occurrence of the various letters of the alphabet found in the 4-letter-long word sample used in this study.

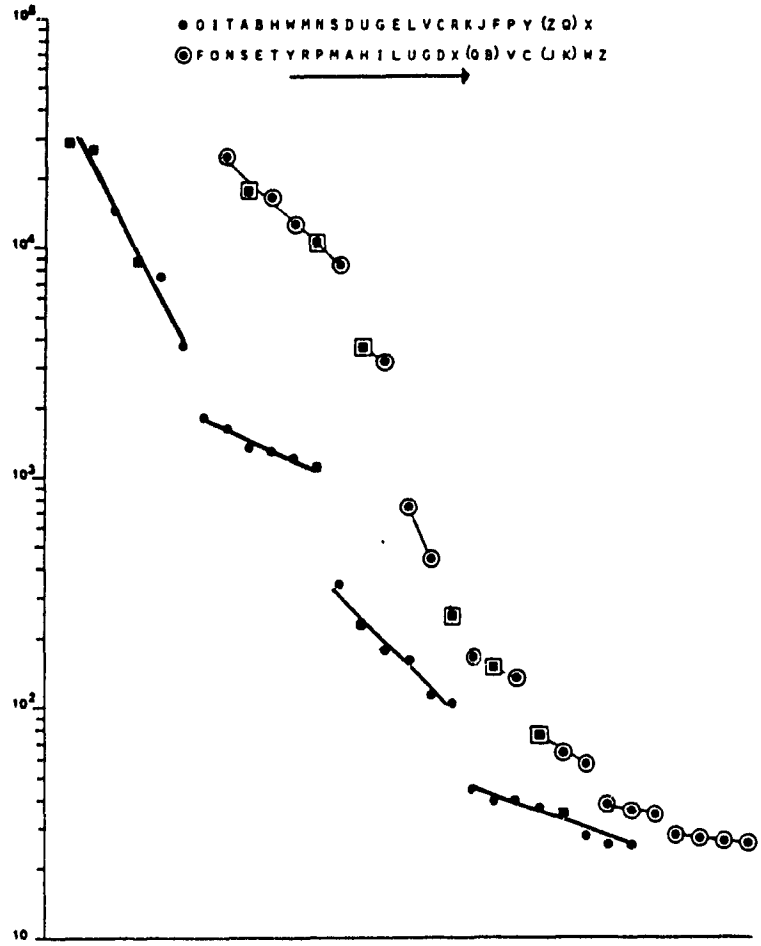

Figure 9.5 Position-dependent rank-frequency plots computed from 2-letter-long English word samples with $\phi = 80$. The vertical axis is in a logarithmic scale. The horizontal axis is linearly ranked in the order of decreasing frequency-of-occurrence for each letter position. The leftmost rank-frequency plot depicts the distribution observed for the first letter position of 2-letter-long words. The rightmost rank-frequency plot depicts the distribution observed for the last letter position of these words. This plot gives the frequency-of-occurrence of the various letters of the alphabet in the first and second positions of the 2-letter-long word sample.

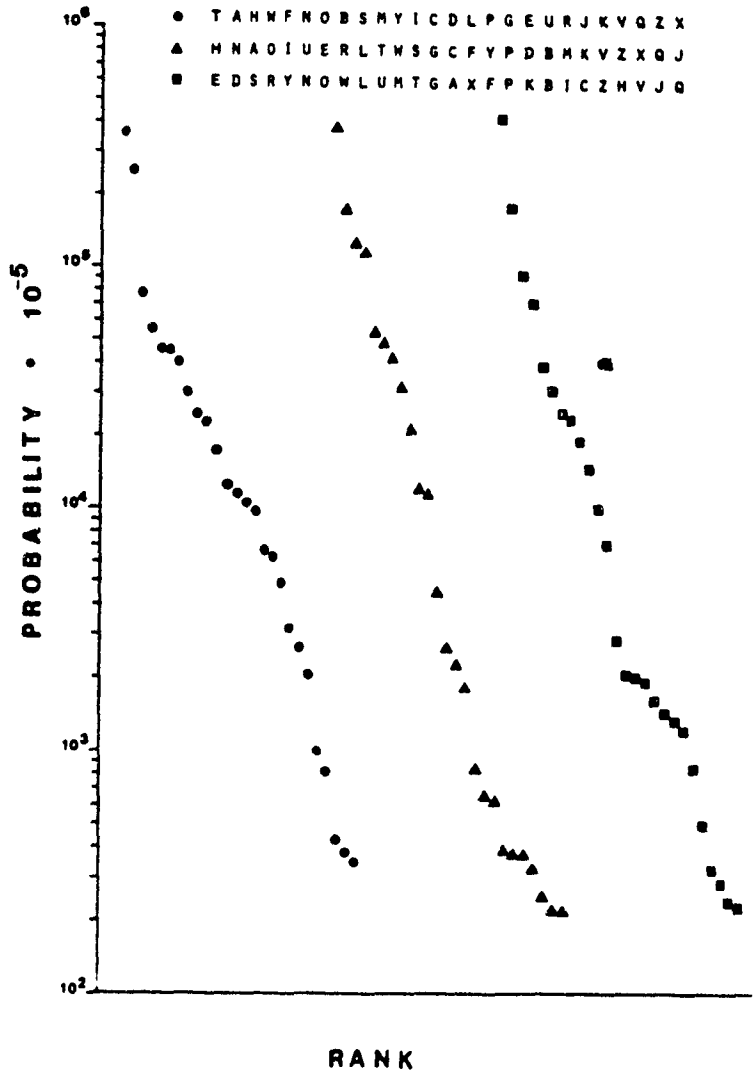

Figure 9.6 Position-dependent rank-frequency plots computed from 3-letter-long English word samples with $\phi = 80$. The vertical axis is in a logarithmic scale. The horizontal axis is linearly ranked in the order of decreasing frequency-of-occurrence for each letter position. The leftmost rank-frequency plot depicts the distribution observed for the first letter position of 3-letter-long words. The rightmost rank-frequency plot depicts the distribution observed for the last letter position of these words. This plot gives the frequency-of-occurrence of the various letters of the alphabet in the first, second and third positions of the 3-letter-long word sample.

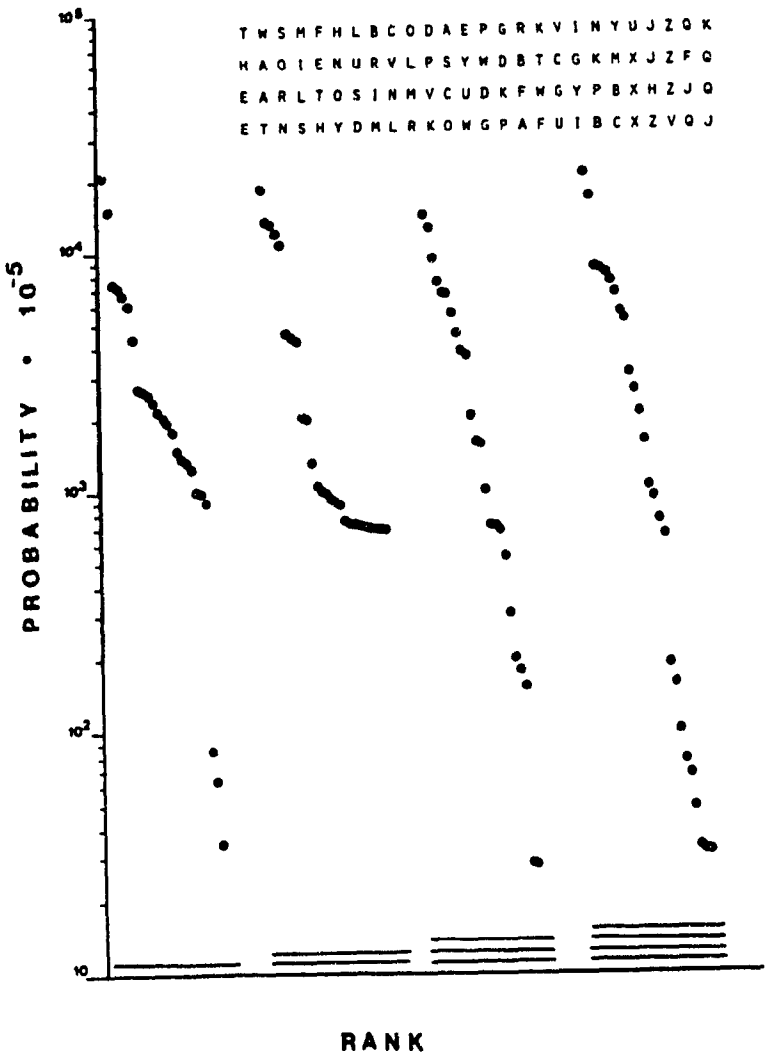

Figure 9.7 Position-dependent rank-frequency plots computed from 4-letter-long English word samples with $\phi = 80$. The vertical axis is in a logarithmic scale. The horizontal axis is linearly ranked in the order of decreasing frequency-of-occurrence for each letter position. The leftmost rank-frequency plot depicts the distribution observed for the first letter position of 4-letter-long words. The rightmost rank-frequency plot depicts the distribution observed for the last letter position of these words. This plot gives the frequency-of-occurrence of the various letters of the alphabet in the first, second, third and fourth positions of the 4-letter-long word sample.

It is possible to demonstrate that the overall frequency distribution of the letters of the alphabet, shown in Figure 9.3, can be approximated by the weighted sum of the distributions given in Figures 9.4a, 9.4b, 9.4c, and 9.4d (ϕ = 80%). An analysis of variance shows that the weighted sum of these distributions accounts for 7.7, 43.7, 78.7, and 82.8 percent, respectively of the variance observed in Figure 9.3. These approximations are computed under the assumption of a log-normal distribution of word length where the relative weight of each component distribution in the cumulative sum is that observed [9.14] in the sampled data.

One may derive a first-hand approximation of the frequency-of-occurrence of any N-letter-long word in the English language from the position-dependent, rank-frequency plots computed for words of length N. Unfortunately, given the N fundamental rank-frequency plots, the model will generate fictitious probabilities for any sequence of letters of length N which can be permutated from the English alphabet. The computed probability of a sequence of letters only has meaning for those permutations which are listed as a part of speech in the English language. As such, for practical applications, it is necessary to somehow maintain a dictionary of valid English words [9.8]. Rather than store many dictionaries listing words of specific length, it is possible [9.8] to specify all N-letter long words in terms of $2^N - 1$ automata graphs, referred to as word webs in Chapter 8.

The three transition diagrams for finite automata which can be used to exclusively generate all valid 2-letter-long English words listed in the Oxford English Dictionary are given in Figures 9.8, 9.9 and 9.10. It is possible to compute the expected frequency of occurrence of any 2-letter-long word listed in word webs, such as those given in the previous chapter, as the product of its position-dependent letter-frequencies $f_{(2,j)}$, given in Figure 9.5.

In general, the frequency of occurrence of any l -letter long type, $f(T)$, can be approximated by the product of its position-dependent letter frequencies, $f_{(l,j)}$, as:

$$f(T) = \prod_{j=1}^{l} f_{(l,j)} \qquad (9.8$$

The validity of this computation is somewhat constrained by the fact that the frequency-of-occurrence of a word cannot be wholly ascribed to the simple product of the observed, disjoint, position-dependent frequency-of-occurrence of its letters.

A rank-correlation test showed significant agreement (Kendall, Tau = 0.613, SD = 0.150, $\rho < 0.001$) between the most-frequently observed 2-letter-long words [9.29] and those predicted on the basis of Equation 9.8 to be the most-frequently used. The observed 'types' used for this study included all 2-letter words known to occur at least 500 times per million tokens of running text.

Two obvious outliers or exceptions exist in this list. The computed frequencies of the rare words " OS " and " AY " are, erroneously very large. These errors occur as the result of the very frequent use of words such as { OF, ON, IS, AS } and { AS, AN, BY, MY} which like " OS " and " AY " either start with the letters " O " or " A " or end in the letters " S " or " Y ". Hence, both " OS " and " AY " must be noted as valid exceptions to the fidelity of this method.

Using refined position-dependent frequency-data it is possible to obtain almost perfect rank-correlations between the observed and computed frequency-of-occurrence of all 2-letter-long types known to occur at least twice-per-million tokens of running text. These results are obtained at a cost of two more exceptions to the fidelity of Equation 9.8. Word-size and position-depended letter-frequency data may thus be used to compute the expected frequency-of-occurrence of most words.

The overall position-dependent letter-frequencies for 5-, 6-, 7-, 8-, 9-, and 10-letter-long-words are given in Figures 9.11, 9.12, 9.13, 9.14, 9.15, and 9.16. In all cases the observed distributions are similar to those described here for the smaller words. Word-size and position-dependent letter-frequency data may also be used to infer the most likely size of the suffix and prefix structures found in larger

'derived' words. Such information proves to be very useful in the development of context-sensitive rule bases for the reduction of a derived word from its base or word-root [9.41]. The development of such rule bases is the topic of Chapter 10.

9.4 CONCLUSION

The method and results presented in this chapter demonstrate that it is possible to easily compute the existence and approximate frequency-of-use of popular N-letter-long-words in the English language.

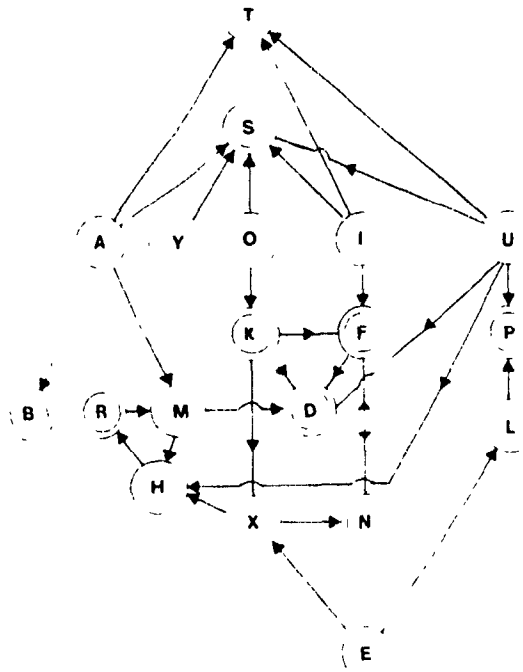

Figure 9.8 Word web, or transition diagram, for the finite automata which may be used to exclusively generate all 2-letter-long words of the form **VC** listed in the Oxford English Dictionary. All vowels are initial states in this directed graph.

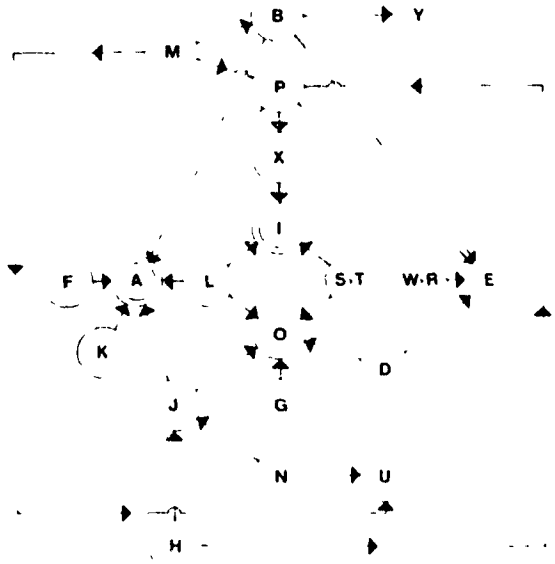

Figure 9.9 Word web, or transition diagram, for the finite automata which may be used to exclusively generate all 2-letter-long words of the form **CV** listed in the Oxford English Dictionary. All consonants are initial states in this directed graph.

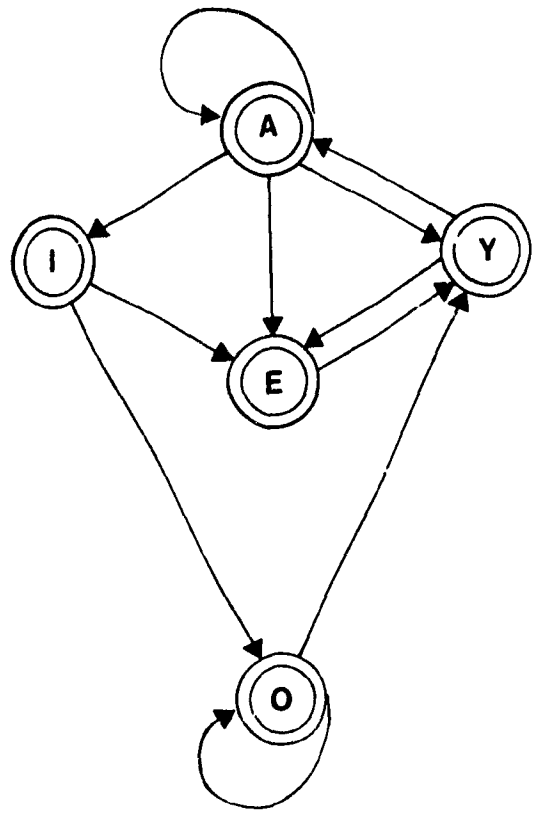

Figure 9.10 Word web, or transition diagram, for the finite automata which may be used to exclusively generate all 2-letter-long words of the form **VV** listed in the Oxford English Dictionary.

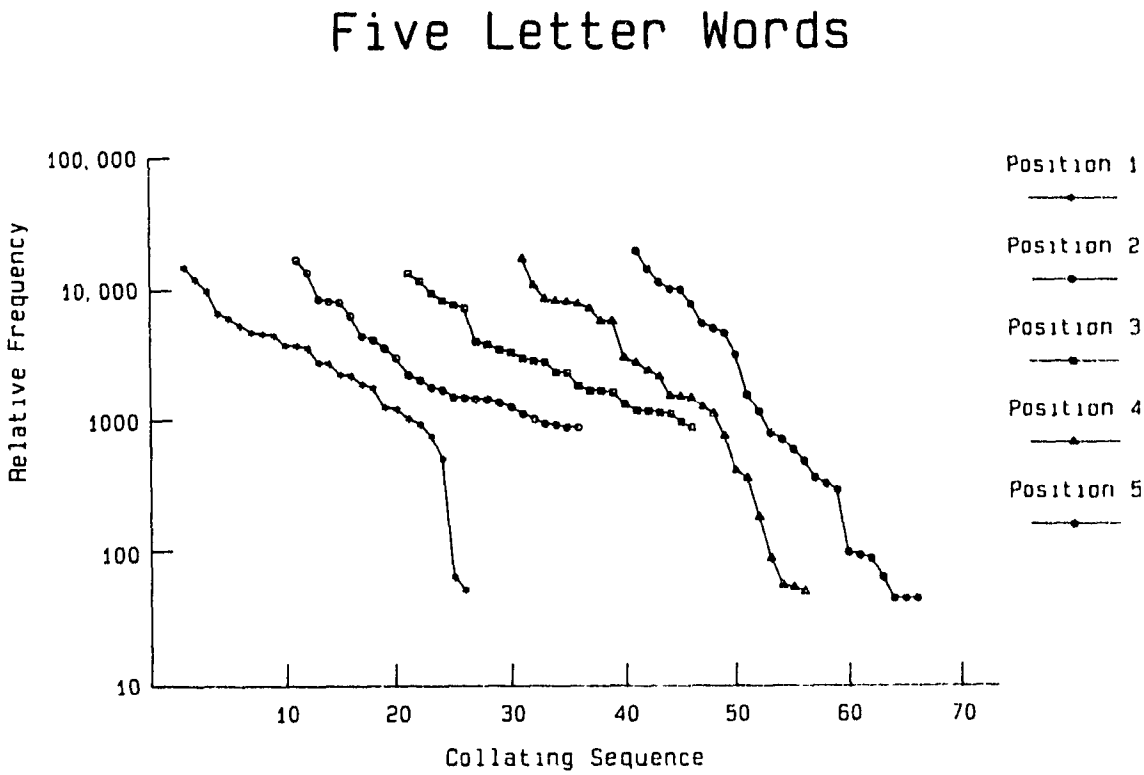

Figure 9.11 The frequency-of-occurrence of the letters of the English alphabet as a function of position within a 5-letter-long English word sample. The vertical axis is in a logarithmic scale. The horizontal axis is linearly ranked in the order of decreasing frequency-of-occurrence for each position within a 5-letter-long template.

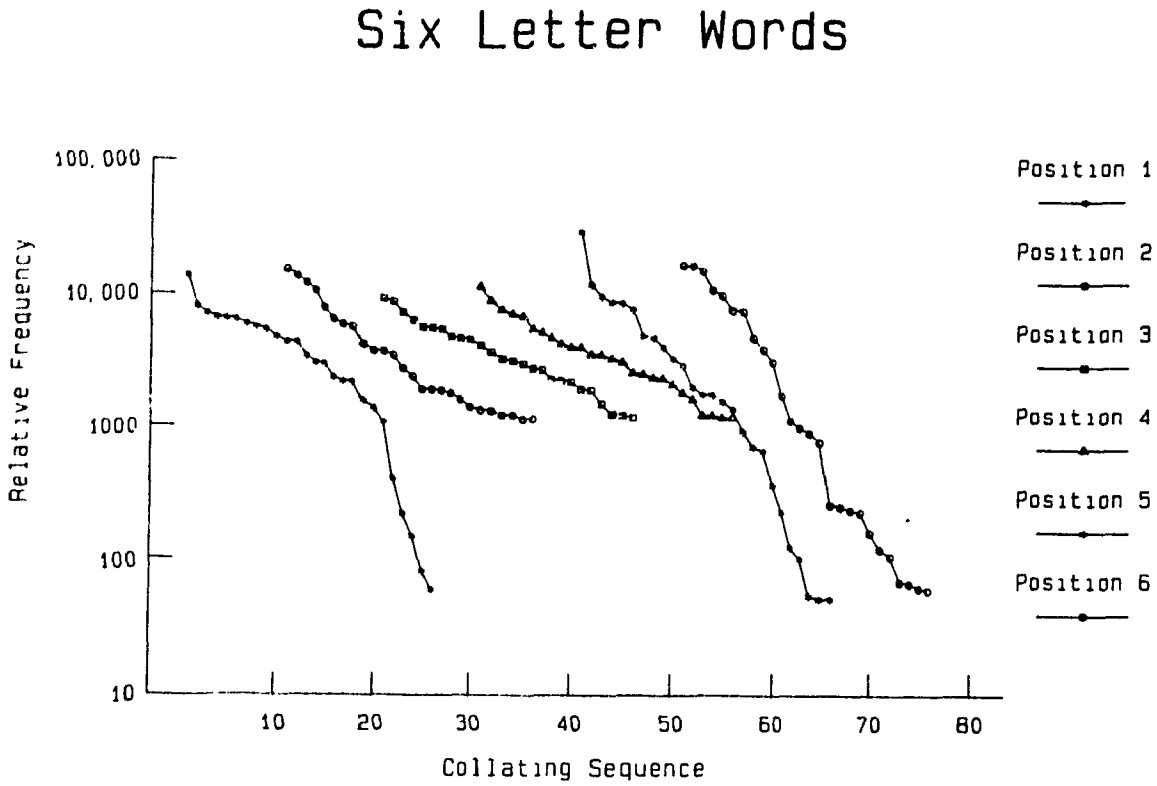

Figure 9.12 The frequency-of-occurrence of the letters of the English alphabet as a function of position within a 6-letter-long English word sample. The vertical axis is in a logarithmic scale. The horizontal axis is linearly ranked in the order of decreasing frequency-of-occurrence for each position within a 6-letter-long template.

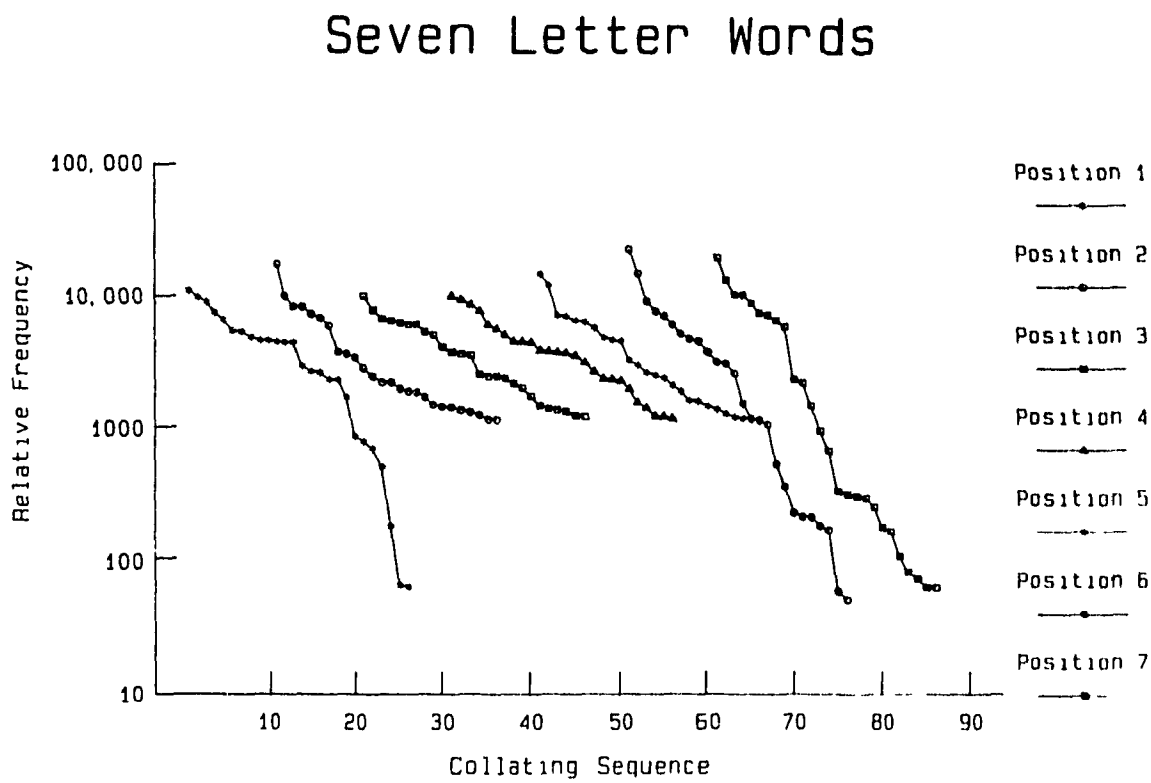

Figure 9.13 The frequency-of-occurrence of the letters of the English alphabet as a function of position within a 7-letter-long English word sample. The vertical axis is in a logarithmic scale. The horizontal axis is linearly ranked in the order of decreasing frequency-of-occurrence for each position within a 7-letter-long template.

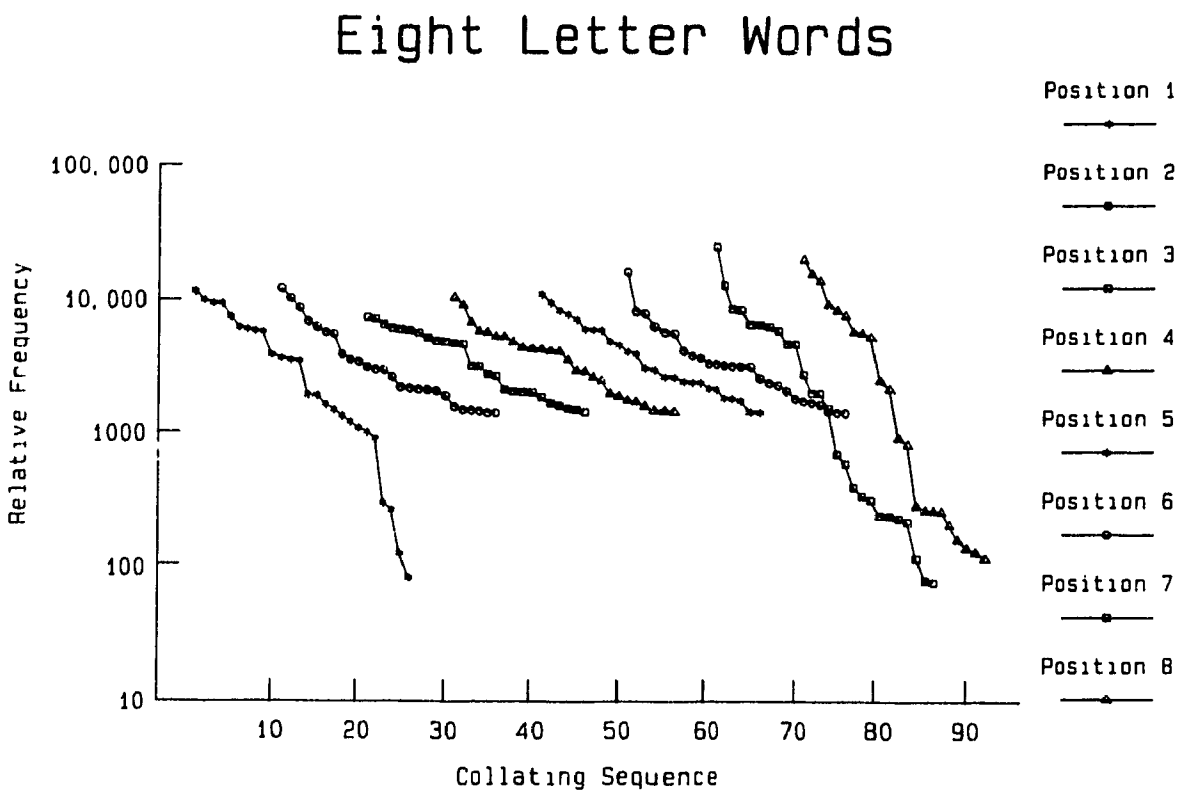

Figure 9.14 The frequency-of-occurrence of the letters of the English alphabet as a function of position within a 8-letter-long English word sample. The vertical axis is in a logarithmic scale. The horizontal axis is linearly ranked in the order of decreasing frequency-of-occurrence for each position within a 8-letter-long template.

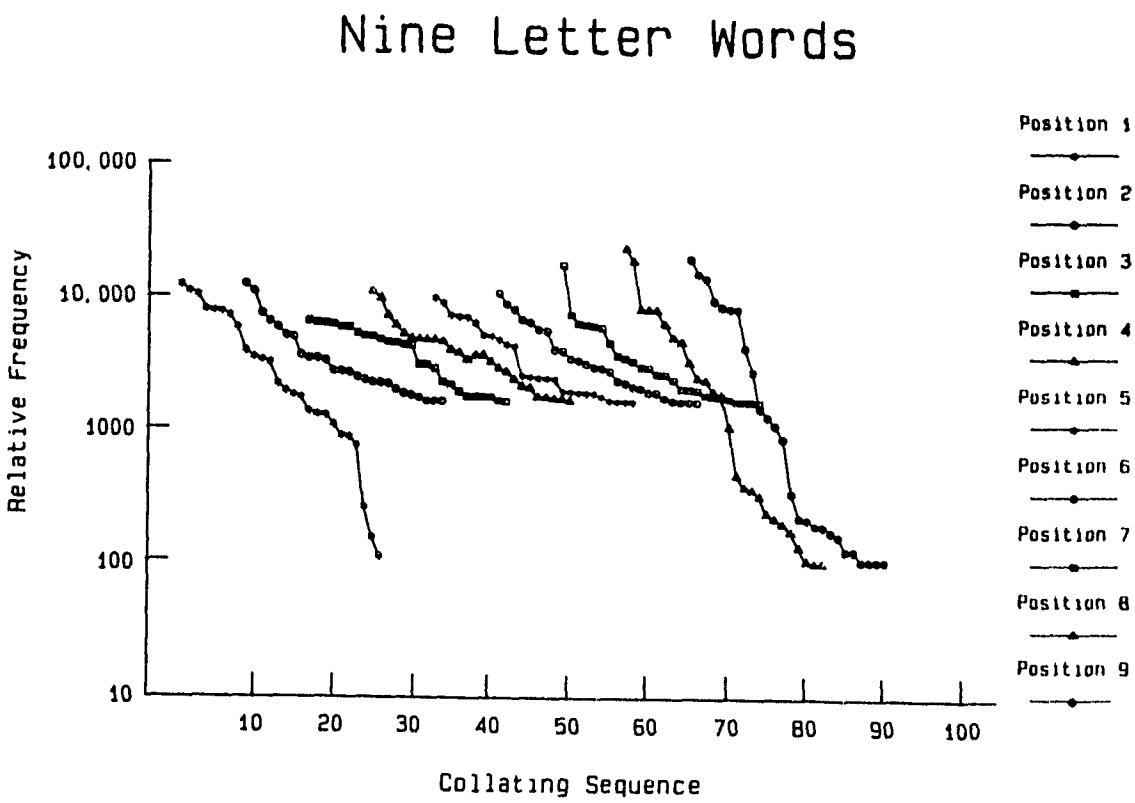

Figure 9.15 The frequency-of-occurrence of the letters of the English alphabet as a function of position within a 9-letter-long English word sample. The vertical axis is in a logarithmic scale. The horizontal axis is linearly ranked in the order of decreasing frequency-of-occurrence for each position within a 9-letter-long template.

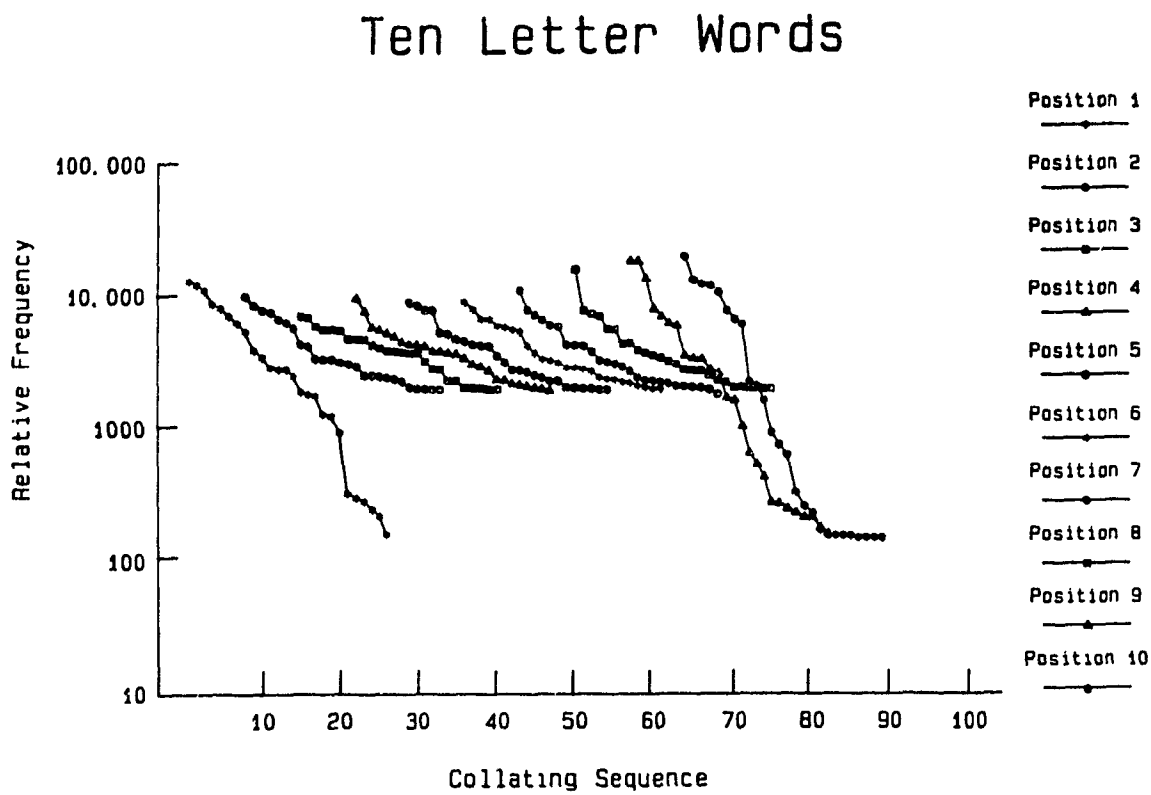

Figure 9.16 The frequency-of-occurrence of the letters of the English alphabet as a function of position within a 10-letter-long English word sample. The vertical axis is in a logarithmic scale. The horizontal axis is linearly ranked in the order of decreasing frequency-of-occurrence for each position within a 10-letter-long template.

9.5 REFERENCES

[9.1]. see 1.49

[9.2]. see 1.72

[9.3]. E. H. Lenneberg, Biological Foundations of Language, Wiley, New York, 1967.

[9.4]. see 1.45

[9.5]. see 8.11

[9.6]. see 1.46

[9.7]. see 8.4

[9.8]. see 1.5

[9.9]. see 8.5

[9.10]. see 8.6

[9.11]. see 1.48

[9.12]. see 8.9

[9.13]. see 8.8

[9.14]. see 3.9

[9.15]. G. T. Toussaint, R. Shinghal, "Cluster Analysis of English Text," In proceedings of Pattern Recognition and Image Processing Conference, pp. 164-172, Chicago, 1978.

[9.16]. G. T. Toussaint, "Recent Progress in Statistical Methods Applied to Pattern Recognition," In proceedings 2 and International; Joint Conference on Pattern Recognition, Copenhagen, 1974.

[9.17]. A. R. Hanson, E. M. Riseman, E. Fisher, "Context in Word Recognition," Pattern Recognition, Vol. 8, pp. 35-45; 1976.

[9.18]. R. Ehrich, K. Koehler, "Experiments in the Contextual Recognition of Cursive Script," IEEE Transactions on Computers, Vol. c-24, 2, pp. 182-193; 1975.

[9.19]. G. Toussaint, R. Donaldson, "Some Simple Contextual Decoding Algorithms Applied to Recognition of Hand-Printed Text," In proceedings of Annual Canadian Computer Conference, pp. 422101-422116; 1972.

[9.20]. R. O. Duda, P. E. Hart, "Experiments in the Recognition of Hand-Printing Text: Part II-Context Analysis," AFIPS Conference Proceedings, Vol. 33, pp. 1139-1149; 1968.

[9.21]. E. M. Riseman, A. R. Hanson, "A Contextual Post-processing System for Error Correction Using Binary n-grams," IEEE Transaction on Computers, Vol. c-23, 5, pp. 480-493; 1974.

[9.22]. C. M.Vossler, N. M. Branston, "The Use of Context for Correcting Garbled English Text," In proceedings of ACM 19th National Conference, pp. D2 4-1 to D2 4-3; 1964.

[9.23]. C. R. Blair, "A Program for Correcting Spelling Errors," Information and Control, Vol. 3, pp. 60-67; 1960.

[9.24]. G. Carlson, "Techniques for Replacing Characters that are Garbled on Input," In proceedings of the Spring Joint Computer Conference, pp. 189-192; 1966.

[9.25]. R. Shinhal, G. Toussaint, "A Bottom-up and Top-down Approach to Using Context in Text Recognition," International Journal of Man-Machine Studies, Vol. 11, pp. 201-212; 1979.

[9.26]. R. Shinghal, G. Toussaint, "Experiments in Text Recognition with the Modified Viterbi Algorithm." IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-1, 2, pp. 184-193; 1979.

[9.27]. R. Shinghal, D. Rosenberg, G. Toussaint, "A Simplified Heuristic Version of a Recursive Bayes Algorithm for Using Context in Text Recognition," IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-8, pp. 412-414; 1978.

[9.28]. R. Shinghal, G. Toussaint, "The Sensitivity of the Modified Viterbi Algorithm to the Source Statistics," IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. PAMI-1, 2, pp. 181-185; 1980.

[9.29]. see 2.9

[9.30]. see 3.11

[9.31]. The Compact Edition of the English Dictionary, C. T. Onions, ed., Oxford University Press, Oxford, England, 1971.

[9.32]. see 3.1

[9.33]. see 3.3

[9.34]. see 1.61

[9.35]. see 2.15

[9.36]. B. Mandelbrot, "On the Theory of Word Frequencies and on Related Markovian Models of Discourse," R. Jakobson, ed., Structure of Language and its Mathematical Aspects, American Mathematical Society, Providence, Rhonde Island, pp. 190-219; 1961.

[9.37]. see 2.5

[9.38]. see 2.8

[9.39]. see 1.62

[9.40]. K. Knopp, Infinite Sequences and Series, Dover Publications, Inc., New York, N. Y., pp. 80-90; 1956

[9.41]. see 1.74

CHAPTER TEN
SYNTACTIC STRUCTURES AND WORD-LEVEL GRAMMARS IN ENGLISH

10.1 INTRODUCTION

Research activity in the field of computational linguistics has grown at the rapid rate which parallels the recent attention given to the study of artificial intelligence and the widening application of computers to non-numerical problem domains.

In this chapter we shall discuss Natural Language Processing, (NLP), in the areas of word recognition and word understanding. Atwell [10.1] has recently pointed out that Chomsky's early work on syntactic structures [10.2] has greatly influenced the importance placed on metaknowledge, such as deep structure, in NLP systems. However, it would now appear [10.1] that simpler heuristic surface structure techniques, such as those discussed in this chapter, yield good practical results when compared to systems seeking to exploit deeper structures.

'Word recognition' can be viewed as a syntactic problem with important practical applications in word processing, man-machine interfaces, and the development of sophisticated input devices such as 'smart' optical scanners [10.3, 10.4]. Smart optical scanners are required to circumvent the errors introduced by the optical scanner's pattern recognition routines which, even when working at 99% efficiency, end up garbling one word in every two English sentences [10.5, 10.6].

'Word understanding', on the other hand, is usually considered to be a difficult semantic problem [10.7] with application to next generation relational database systems and embedded or robotic devices. Present applications of such semantic systems have involved isolating a word's root or base-word for use in sophisticated NLP systems [10.8, 10.9, 10.10, 10.11].

In pursuing work on mathematical models of English language word structure and usage this chapter presents some efficient techniques to enable a machine to recognize English language words and their grammatical structure. Such studies typically require that

one attempts to achieve simultaneously two goals. The first goal, which is to maximize the absolute size of the vocabulary covered by the model, requires that one analyses a large carefully compiled lexicon. The second goal is to assure the model's usefulness by somehow restricting its errors to rarely used words. This second objective requires access to probabilistic or statistical information on the frequency of use of the words in the lexicon.

The results presented here are based on our own previous work [10.12, 10.13, 10.14] as well as that of others [10.15, 10.16, 10.17, 10.18]. These results are mostly derived from an exhaustive analysis of a database encompassing the vocabulary of the Oxford Paperback Dictionary, OPD, [10.19] and the Oxford Spelling Dictionary, OSD, [10.20].

10.2 SYNTACTIC STRUCTURE

Markovian production rules expressed in Backus-Naur Form, BNF, have been used extensively as generic representations of syntactic structure. This formalism has been applied to many practical problems through developments in syntactic pattern recognition [10.21].

While they are very powerful, syntactic pattern recognition techniques suffer from a major methodological drawback. They require that a correct *a priori* structural model of the abstraction exists.

Furthermore the implementation of successful syntactic parsing routines requires not only the existence of an *a priori* structural model but also the availability of pattern recognition routines which can be used to isolate and correctly classify the model's features from raw data.

10.3 VOWEL NORMAL FORM: WORD LEVEL SYNTACTIC STRUCTURE

Vowel Normal Form (VNF) is a heuristic structural feature which has been developed [10.12, 10.13] to cluster and classify words on the basis of a single hybrid feature which has orthographic, phonetic

and probabilistic components. Figures 10.1 and 10.2 illustrate the importance of VNF classification. These figures demonstrate that the VNF classes found to be most frequently used in forming words of a given length form a structural kernel of the VNF frames empirically found to be most frequently used in forming words of greater length. As we shall see in the next section of this chapter VNF can also be used to isolate clusters of structurally similar words of a given length. It is subsequentially possible to build sets of rules for accurately reducing words in these well-populated VNF classes to their root or base word [10.11, 10.14].

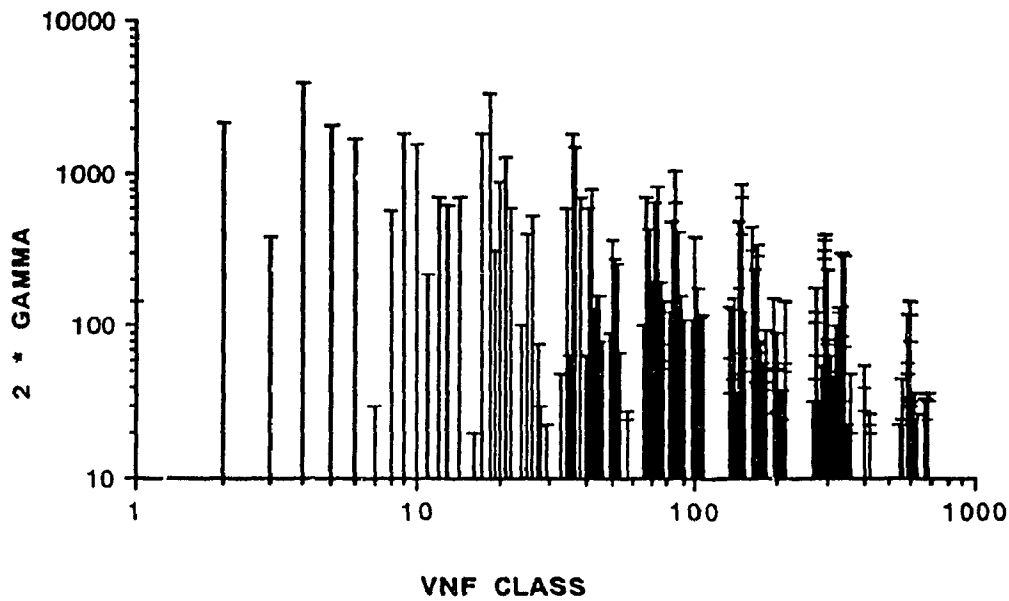

Figure 10.1 Filtered VNF density plot for all 2-, 3-,.. 12-letter-long valid English words defined in the OPD. Only those sets with 10 or more elements are depicted in this figure. Abscissa VNF class or structure specified as a base 10 number. Ordinate set size in words. This is a filtered image of Figure 4.1

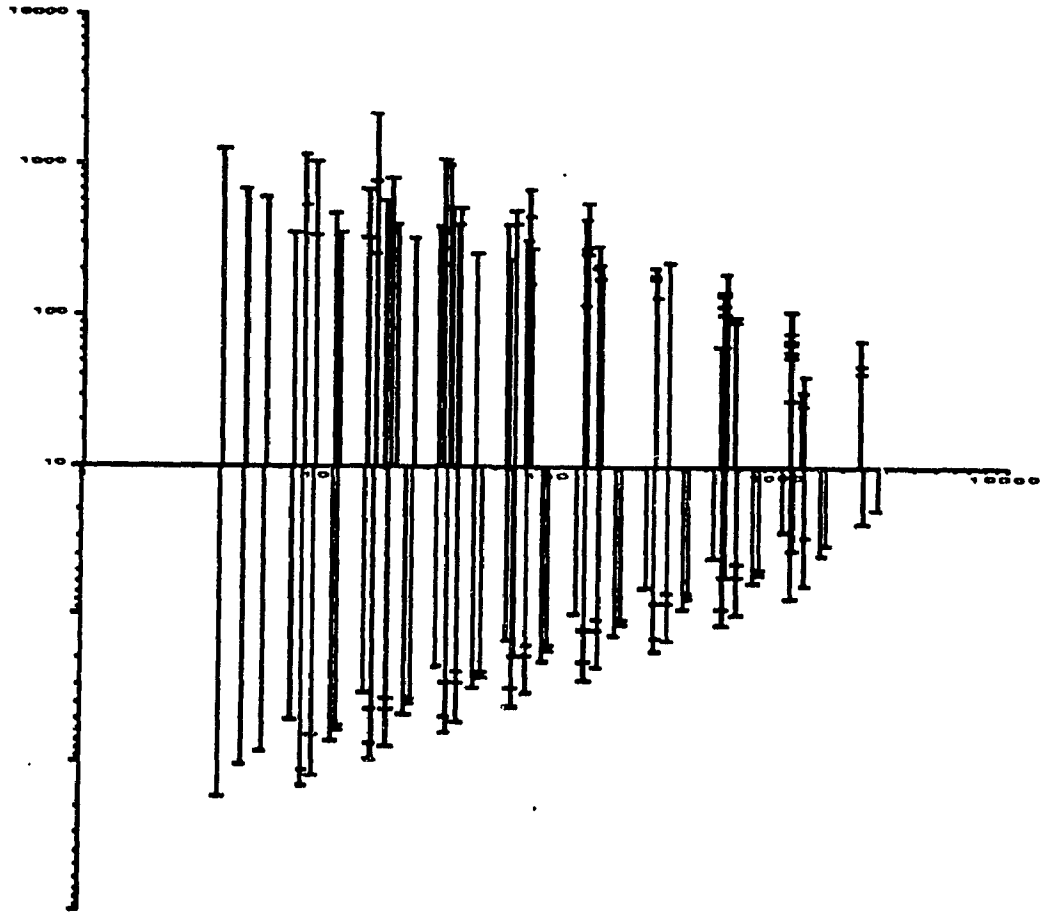

Figure 10.2 Superimposed Composite Image.

TOP-PLATE: Observed Top-Ten VNF frames found in 5-,.., 12-letter-long-words defined in the OPD [10.19] (see Figure 7.1). BOTTOM-PLATE: Predicted Top-Ten VNF frames (see Figure 7.3). This figure demonstrates that relatively few VNF frames account for the majority of the vocabulary structures found in English and that both the size and lexical structure of these principle frames are predictable .

10.4 METHODS

The lexicon used for this work was the Oxford Paperback Dictionary (OPD) [10.19]. This lexicon was the largest magnetically stored dictionary available to us. [9] A string processing routine which capitalized on the relatively rigid stylistic structure [10.23] of the entries in the OPD was developed to parse and extract all entries which are listed as parts of speech in this dictionary. This lexicographically ordered wordlist was then sorted by wordsize to yield, for example, a lexicographically sorted wordlist of 10-letter-long words. In order to group together all words of a similar structure, each wordlist was subsequently classified by its VNF. Wordlists of a given length and vowel normal form were then submitted to further syntactic analysis in order to produce the rulebases given in this chapter. While it is possible to produce a rulebase for each set of words given by its VNF grouping, it proves to be both more informative and efficient to construct rulebases which cover all words which share the same VNF suffixes. In this procedure, all VNF groups with common structured endings, such as **VCCV**, are clustered together to generate the rulebase for their suffixes.

While most of the work presented in this chapter is derived from the wordlists found in the OPD and the OSD, some auxiliary statistical data sources [10.24] were used to compute the expected, position-dependent letter-frequencies for words of a given length occurring in English text. This statistical data has been used primarily to help estimate the relative likelihood of an arbitrary suffix occurring in 10-letter-long-words.

10.5 WORD LEVEL GRAMMARS

Word level grammars have been developed by various authors [10.25, 10.26, 10.27, 10.28, 10.29] for use in such

---

9 This data was made available for the purposes of this study by the Oxford University Press through the kind support of Dr. Robert Burchfield, CBE, the editor-in-chief of the Oxford University Dictionaries.

environments as the UNIX spelling checker. These grammars are however context free and hence are both very limited in their application and very prone to error. The work presented in this chapter is principally the result of context sensitive refinements to early context-free approaches such as Paice's algorithm [10.25, 10.15]. The algorithms described here allow for both the sequential and parallel evaluation of a word using our rulebase. Earlier context free systems resemble simple expert systems based on production rules in that they do not have the ability to learn or modify their rulebase. The rulebases (presented in this chapter) were derived by a manual analysis of the system's wordlists. However, the process of clustering the various VNF groups into the major amalgamated subclasses used in this work is not a difficult one to automate.

Further work on the merits of using VNF as the principle word feature in the development of a discovery algorithm for the automated derivation of the system's context-sensitive rulebase is in preparation.

10.6 ALGORITHM

The principle design philosophy [10.30, 10.31] underlying the algorithm presented in this chapter is to incorporate the hierarchical constraints that logically exist within a set of domain-specific rules into a rulebase of context-sensitive production schemata.

The aim of this approach is to produce a uniframe [10.33, 10.34] system suitable for the reduction of natural language words to their stems or roots. Such a system may be used to reduce each member of a set of derived words, such as { *egoism, egoist, egotism, egotist, egotize, egotistic, egoistic, egomania, egomaniac, egotizing, egoistical, egotistical, egoistically, egotistically, egocentric, egocentricity, egocentrically* } and the hyphenated form *ego-trip* to its root: *ego* in this example. Besides reducing such semantic and syntactic derivations of a concept to a common stem, the algorithm may be easily modified to tackle the tasks of breaking or decomposing compound words such as 'wavelength' into 'wave + length'. It is also possible [10.14] to use this system to hyphenate derived words, for example 'ionization', would be hyphenated as 'ion-ization'. The

hyphenation procedure forced us to accept a standard or benchmark such as that recently published by the Oxford University Press [10.20] as the knowledge base for the algorithm. One of the consequences of undertaking this research has been the derivation of a comprehensive set of syntax rules for specifying natural language word structure and morphology.

This system's rulebase contains a set of context sensitive **IF...THEN** productions. These rules may be viewed as a record structure which contains five essential fields: **RULE_NUMBER**, **SUFFIX_STRING**, **REPLACEMENT_STRING**, **NEXT_RULE_NUMBER**, **PREREQUISITE_RULE_NUMBER**.

1 SELECT APPROPIATE RULESET ON THE BASIS OF THE WORD'S LENGTH AND VOWEL-NORMAL-FORM (**VNF**).

2 INITIALIZE THE VARIABLE RULE-NUMBER, **R**, TO RULE 1, r_1 . **R** := r_1 .

3 SEQUENTIALLY SCAN THE RULESET UNTIL ENCOUNTERING EITHER A TERMINATOR SYMBOL ● , OR THE FIRST SUFFIX RULE, r_i , WHICH MATCHES THE WORD'S ENDING.

4 TERMINATE IF A ● SYMBOL WAS ENCOUNTERED AT STEP 3.

5 REPLACE THE WORD'S ENDING IN ACCORDANCE WITH THE SPECIFIED RULE, r_i .

6 CONTINUE FURTHER ANALYSIS OF THE WORD BY RESETTING **R** , IN ACCORDANCE WITH THE DIRECTIVES OF THE **NEXT_RULE_NUMBER** FIELD OF r_i TO ONE OF THREE POSSIBLE CASES:

i) **R** := **R** + 1, REPEAT STEP 3 THROUGH STEP 6.

ii) **R** := r_j REPEAT STEP 3 THROUGH STEP 6.

iii) TERMINATE IF THE SYMBOL ^ IS ENCOUNTERED.

Figure 10.3 Sequential Algorithm.

```
PROCEDURE EVALUATE ( STRING, RULE_NUMBER )
  BEGIN
      IF ( RULE_NUMBER.SUFFIX_STRING  = STRING.SUFFIX_STRING )
            THEN
                  REPLACE ( STRING, RULE_NUMBER )
            ELSE

                  IF ( SUCC( RULE_NUMBER.SUFFIX_STRING )≠ ● )
                        THEN
                          EVALUATE ( STRING, RULE_NUMBER + 1 )
                        ELSE
                          TERMINATE
  END

PROCEDURE REPLACE ( STRING, RULE_NUMBER )
  BEGIN
      STRING := STRING  -  RULE_NUMBER.SUFFIX_STRING
                +  RULE_NUMBER.REPLACEMENT_STRING

      IF (  RULE_NUMBER.NEXT_RULE_NUMBER ≠ ^  )
            THEN
               EVALUATE ( STRING, RULE_NUMBER.NEXT_RULE_NUMBER )
            ELSE
                  TERMINATE
  END
```

Figure 10.4 Two recursive procedures for implementing the sequential algorithm given in Figure 10.3.

The algorithm, given in Figure 10.3, may be viewed as two procedures, as outlined in Figure 10.4 which call each other recursively in their evaluation of the system's rulebase. Each production-rule in our rulebase specifies the string substitutions appropriate to the reduction of a given **SUFFIX_STRING** by its **REPLACEMENT_STRING** field. In addition to this **IF** ⟨**SUFFIX_STRING** Found ⟩ **THEN** ⟨ Substitute **REPLACEMENT_STRING** for **SUFFIX_STRING** ⟩ script each rule specifies the control flow appropriate to the further sequencing of the production rules. The rulebase also contains information on which rule, if any, must by evaluated as the immediate prerequisite to the evaluation of a production rule. By maintaining this information on immediate prerequisites (**PREREQUISITE_RULE_NUMBER** Field) for every entry in the rulebase, [in addition to the control flow information needed to specify priority sequencing (**NEXT_RULE_NUMBER** Field)] this system may be executed in either a parallel or a sequential manner.

1 SELECT APPROPIATE RULESET ON THE BASIS OF THE WORD'S LENGTH AND VOWEL-NORMAL-FORM (VNF).

2 INITIALIZE **R** := C_1 TO INITIATE PROCESSING AT THE FIRST CLUSTER OF RULES, C_1.

3 EVALUATE ALL RULES IN C_1 TO FIND THE FIRST POSSIBLE MATCH OF A SUFFIX RULE TO THE WORD'S ENDING.

4 IF NO MATCH IS FOUND IN C_1 THEN EXECUTE THE EXCEPTION CLAUSE FOR THE RULE CLUSTER.

5 IF A POTENTIAL MATCH IS FOUND IN STEP 3 THEN CHECK THE PREREQUISITE CONDITIONS SPECIFIED BY THE RULE IN ITS **PREREQUISITE_RULE_NUMBER** FIELD

i) IF NO PREREQUISITES ARE SPECIFIED THEN EXECUTE STEP 6.
ii) IF A PREREQUISITE RULE EXISTS THEN EXECUTE STEP 8.

6 REPLACE THE WORD'S ENDING IN ACCORDANCE WITH THE SELECTED RULE.

7 CONTINUE FURTHER ANALYSIS BY EVALUATING ALL POSSIBLE SUBSEQUENT MATCHES BY RESETTING **R** IN ACCORDANCE WITH THE **NEXT_RULE_NUMBER** FIELD OF THE SELECTED RULE ONE OF TWO CASES CAN OCCUR.

i) **R** = C_K
ii) TERMINATE WHEN THE SYMBOL ^ IS ENCOUNTERED.

8 EVALUATE THE RULE CLUSTER C_J SPECIFIED IN THE **PREREQUISITE_RULE_NUMBER** FIELD EXECUTE STEP 4.

Figure 10.5 Parallel Algorithm.

The parallel algorithm, given in Figure 10.5, makes use of sequencing prerequisites (**PREREQUISITE_RULE_NUMBER** Field) to isolate and correctly compute necessary sequential dependencies. Pseudo-Code for such a system, conforming to the principles of mini-language Parallel [10.35], is given in Figure 10.6. A more detailed view of the rulebase is given in Figure 10.7, which outlines the Backus-Naur Form of its structure.

```
PROCEDURE EVALUATE ( STRING, RULE_NUMBER )
BEGIN
Flag := false
LOOP { Cluster 1 }
WHEN
      RULE_NUMBER_ONE.SUFFIX_STRING = STRING.SUFFIX_STRING
            => BEGIN
                  FLAG := true;
                  REPLACE ( STRING, RULE_NUMBER_ONE )
              END
WHEN
      RULE_NUMBER_TWO.SUFFIX_STRING = STRING.SUFFIX_STRING
            => BEGIN
                  FLAG := true;
                  REPLACE ( STRING, RULE_NUMBER_TWO )
              END
..
.
END LOOP;

IF ( FLAG = false ) => EXCEPTION_CLAUSE ( CLUSTER_ONE ) ;

END

PROCEDURE REPLACE ( STRING, RULE_NUMBER_ONE )
BEGIN
      STRING := STRING - RULE_NUMBER_ONE.SUFFIX_STRING
                + RULE_NUMBER_ONE.REPLACEMENT_STRING


IF ( RULE_NUMBER_ONE.NEXT_RULE_NUMBER ≠ ^ )
THEN
      EVALUATE ( STRING, RULE_NUMBER_ONE.NEXT_RULE_NUMBER )
ELSE
      TERMINATE
END

PROCEDURE REPLACE ( STRING, RULE_NUMBER_TWO )
  BEGIN
      STRING := STRING - RULE_NUMBER_TWO.SUFFIX_STRING
                + RULE_NUMBER_TWO.REPLACEMENT_STRING


IF ( RULE_NUMBER_TWO.NEXT_RULE_NUMBER ≠ ^ )
THEN
      EVALUATE ( STRING, RULE_NUMBER_TWO.NEXT_RULE_NUMBER )
ELSE
      TERMINATE
.
END
```

Figure 10.6 Pseudo-code for algorithm given in Figure 10.5.

```
<WORD-GRAMMAR>          ::=   <RULE-SET> ●
<RULE-SET>              ::=   <RULE> ;
                              | <RULE> ; <RULE-SET>
<RULE>                  ::=   <RULE-NUMBER> <ENTRY-POINT> <SUFFIX-
                              STRING> <REPLACEMENT-STRING>
                              <TRANSFER-LABEL> <COMMENT>
                              <PREREQUISITE-RULES>
<RULE-NUMBER>           ::=   <DIGIT>
                              | <DIGIT> <RULE-NUMBER>
<DIGIT>                 ::=   0 | 1| 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<ENTRY-POINT>           ::=   -
                              | <BLANK>
<SUFFIX-STRING>         ::=   - <PRIMARY-SEGMENT>
                              | | <BLANK> - <SECONDARY-SEGMENT>
<PRIMARY-SEGMENT>       ::=   | <CHARACTER-STRING> <BLANK>-<SECONDARY
                              SEGMENT>
                              | ●
<SECONDARY-SEGMENT> ::=       <CHARACTER-STRING>
                              | <TERMINATOR>
                              | <BLANK>
<CHARACTER-CLUSTER>  ::=      <LETTER>
                              | <LETTER> <LETTER-CLUSTER>
<CHARACTER-STRING>  ::=       <LETTER-CLUSTER>
                              | <LETTER-CLUSTER> <WILD-CARD-CLUSTER>
                              <LETTER-CLUSTER>
<LETTER>                ::=   A| B | C | D | E | F | G | H | I | J | K | L | M | N
                              | O | P | Q | R | S | T | U | V | W | X | Y | Z
<TERMINATOR>            ::=   ^
<WILD-CARD-CLUSTER> ::=       ?
                              |   ? <WILD-CARD-CLUSTER>
<BLANK>                 ::=
                              |   <BLANK>
<REPLACEMENT-STRING>::=       - <BLANK>
                              | <SIGN> <LETTER-CLUSTER> <BLANK>
                              | <SIGN> <LETTER-CLUSTER> <WILD-CARD-
                              CLUSTER> <LETTER-CLUSTER>
<TRANSFER-LABEL>        ::=   <RULE-NUMBER>
                              | <SIGNAL>
<SIGNAL>                ::=   <TERMINATOR>
                              | [   (  < RULE-NUMBER >  )
<SIGN>                  ::=   -
                              | +
<COMMENT>               ::=   <WORD-EXAMPLE> <ANALYSIS-RESULT>
                              <NOTE>
                              | <BLANK> <NOTE>
<WORD-EXAMPLE>          ::=   <LETTER CLUSTER>
<ANALYSIS-RESULT>       ::=   <LETTER-STRING>
                              | <BASE-STRING> - <LETTER-CLUSTER>
                              | <BASE-STRING> + <LETTER-CLUSTER>
<BASE-STRING>           ::=   <PREFIX> <LETTER-STRING>
                              | <LETTER-STRING>
<PREFIX>                ::=   <SCRIPT-LETTER>
                              | <SCRIPT-LETTER> <PREFIX>
```

```
<SCRIPT-LETTER>          ::=   a | b | c | d | e | f | g | h | i | j | k | l | m
                               | n | o | p | q | r | s | t | u | v | w | x | y | z
<PREREQUISITE-RULES>     ::=   ( < RULE-NUMBER > )
                               | ( < RULE-NUMBER ><PREREQUISITE-RULES> )
                               | <BLANK>
<NOTE>                   ::=   □
                               | +
                               | ☆
                               | ◆
                               | ▼
                               | ■
                               | ○
                               |
```

Figure 10.7 BNF of the Rulebase Structure.

CCCV ANALYSIS

RULE NUMBER	ENTRY POINT	PRIMARY SEGMENT	SECONDARY SEGMENT	REPLACEMENT STRING	NEXT RULE	WORD-EXAMPLE	ANALYSIS-RESULT	NOTE	PRE-REQUISITE
1		-CY	-	-	^	BANKRUPTCY	BANKRUPT		(8)
2		-RY	-	-	^	PLEASANTRY	PLEASANT		(8)
3		-LY	-	-	(8)	CONSTANTLY	CONSTANT		(8)
4		:	-ISH	-	16	SHEEPISHLY	SHEEP		
5		:	-LESS	-	15	FLAWLESSLY	FLAW	*	
6		:	-ING	-	10	UNERRINGLY	UNERR		
7		:	-^	-	^				
8		-Y	-	-	^	MATRIARCHY	MATRIARCH		
9		-^	-	-	^				
10	=>	-S	-	-E	^	ACCUSINGLY	ACCUSE		
11		-G	-	-E	^	GRUDGINGLY	GRUDGE		
12		-C	-	-E	^	MENACINGLY	MENACE		
13		-K	-	-E	^	STRIKINGLY	STRIKE		
14		-M	-	-E	16	BECOMINGLY	BECOME		
15	=>	-I	-	-Y	^	PITILESSLY	PITY		
16	=>	-MME	-	-M	^	SWIMMINGLY	SWIM		
17		-GG	-	-G	^	SLUGGISHLY	SLUG		
18		-PP	-	-P	^	SNAPPISHLY	SNAP		
19		-BB	-	-B		SNOBBISHLY	SNOB		
20		-NN	-	-N	^	STUNNINGLY	STUN		
21		-TT	-	-T	^	SKITTISHLY	SKIT		
22		-^	-	-	^				

Table 10.1 Rulebase for 10-Letter-Long-Words Ending In **CCCV.**

The first column in Table 10.1 specifies the rule number while the second column is used to denote or label the rules which are entry points in the schema. For example, the symbol => in column two of Rule 10 in Table 10.1, specifies that this rule is an entry point to a subset of this schema's rulebase.

The third column specifies the primary suffix segment. For instance the string **-RY** is the primary suffix specified by Rule 2 in Table 1. All primary and secondary segments are preceded by a - symbol. In Table 10.1 for example, both the primary suffix **-LY** given in column 3 of Rule 3 as well as the secondary suffix **-ISH** given in column 4 of Rule 4 are preceded by a - symbol. The symbol ^ is used to specify the last element of the list of primary suffixes and, hence, causes the procedure to terminate whenever it is encountered.

Column four is used to specify secondary suffix strings which are exposed as terminal suffixes only after the word's primary suffix string has been processed. For example in Table 10.1, the secondary suffix **-ING** specified in rule 6 is removable from the word *unerringly* only after the primary suffix **-LY** has been removed from the word by rule 3. The symbol ^ is used to specify the last element of the list of primary suffixes and, hence, causes the procedure to terminate whenever it is encountered.

PARTITIONS CONFORMING TO A HAND ANALYSIS OF TEN LETTER WORDS ENDING IN CVCV

RULE NUMBER	ENTRY POINT	PRIMARY SEGMENT	SECONDARY SEGMENT	REPLACEMENT STRING	NEXT RULE	WORD-EXAMPLE	ANALYSIS-RESULT	NOTE	PRE-REQUISITE
1		-ITY	-	-E	§(6)				
2		|	-CIL	-CLE	^				
3		|	-BILE	-BLE	^				
4		|	-UOSE	-UE	^				
5		|	^	-	^				
6		-ATE	-	-	§(15)	INTIMIDATE	INTIMID		
7		|	-TR	-TER	^	MAGISTRATE	MAGISTER	☆	
8		|	-IC	Y	^	TRIPLICATE	TRIPLY		
9		|	-MM	-ME	^	CONSUMMATE	CONSUME		
10		|	B	-	^	EXACERBATE	EXACER		
11		|	-AR		^	EXHILARATE	EXHIL		
12		|	-ST	+STATE	^	UNDERSTATE	UNDER+STATE	+	
13		|	-UR	-ER	^	INAUGURATE	INAUGER		
14		|	-^	-	^				
15		-LY	-	-	§(20)	ADEQUATELY	ADEQUATE		
16		|	-SIVE	-SE	^	INCISIVELY	INCISE		
17		|	-TIVE	-T	^	ABORTIVELY	ABORT		
18		|	-I	-Y	^	ORDINARILY	ORDINARY		
19		|	-^	-	^				

20	-ITE	-	-	‖(28)	TRIPARTITE	TRIPART			:
21	|	-SC	-	^	PLEBISCITE	PLEBI			:
22	|	-FIN	+FINITE	^	INDEFINITE	INDE+FINITE	+		:
23	|	-WR	+WRITE	^	UNDERWRITE	UNDER+WRITE	+		:
24	|	-CT	-	^	STALACTITE	STALA	□		:
25	|	-M	-	^	STALAGMITE	STALAG	□		:
26	|	-L	-	^	THEODOLITE	THEODO	□		:
27	|	-^	-ITE	^			○		:
28	-AGE	-	-Y	‖(31)	ASSEMBLAGE	ASSEMBLY			:
29	|	-FLY	-FLAGE	^	CAMOUFLAGE	CAMOUFLAGE	☆		:
30	|	-^	-	^					.
31	-SE	-	-	‖(38)					:
32	|	-MI	-	^	COMPROMISE	COMPRO			:
33	|	-WI	-	^	LENGTHWISE	LENGTH	◆		:
34	|	-CI	-	^	CIRCUMCISE	CIRCUM			:
35	|	-PRI	-	^	ENTERPRISE	ENTER			:
36	|	-E	-	^	JOURNALESE	JOURNAL			:
37	|	-^	-	^					:
38	-ORY	-	-E	‖(44)	PEREMPTORY	PEREMPT			:
39	|	-AT	-E	^	OBLIGATORY	OBLIGE			:
40	|	-SS	-SE	^	PROMISSORY	PROMISE			:
41	|	-IT	-E	^	REPOSITORY	REPOSE			:
42	|	-ST	+STORY	^	CLERESTORY	CLERE+STORY	+		:
43	|	-^	-	^					:
44	-INE	-	-	‖(50)	SACCHARINE	SACCHAR			:
45	|	-TW	+TWINE	^	INTERTWINE	INTER+TWINE	+		:
46	|	-T	-TINE	^	PHILISTINE	PHILISTINE	☆		:
47	|	-EL	-ELINE	^	MOUSSELINE	MOUSSELINE	☆		:
48	|	-L	+LINE	^	BORDERLINE	BORDER+LINE	+		:
49	|	-^	-INE	^	FIREENGINE	FIREENGINE	○		:
50	-CE	-	-	‖(55)					:
51	|	-STI	+STICE	^	INTERSTICE	INTER+STICE	+	(52)	:
52	|	-I	-Y	^	ACCOMPLICE	ACCOMPLY			:
53	|	-FRY	+FRICE	^	DENTIFRICE	DENTI+FRICE	+		:
54	|	-^	-CE	^	BIRTHPLACE	BIRTHPLACE	○		:
55	-OGY	-	-	‖(58)	MINERALOGY	MINERAL	□		:
56	|	-OL	-	^	GRAPHOLOGY	GRAPH	□		:
57	|	-^	-	^					:
58	-IZE	-	-	‖(62)	EVANGELIZE	EVANGEL			:
59	|	-T	-	^	STIGMATIZE	STIGMA			:
60	|	-OD	-	^	RHAPSODIZE	RHAPS			:
61	|	^	-	^					:
62	-ACY	-	-	‖(65)	INACCURACY	INACCUR			:
63	|	-IM	-	^	LEGITIMACY	LEGIT			:
64	|	-^	-	^					:
65	-IVE	-	-	‖(71)	EXHAUSTIVE	EXHAUST			:
66	|	-SAT	-SE	^	ACCUSATIVE	ACCUSE			:
67	|	-TAT	-TE	^	RECITATIVE	RECITE			:
68	|	-RAT	-R	^	PEJORATIVE	PEJOR			:
69	|	-PT	-B	^	ABSORPTIVE	ABSORB			.
70	|	-^	-	^					:
71	-URE	-	-	‖(74)	FORFEITURE	FORFEIT			:
72	|	-AT	-ARE	^	JUDICATURE	JUDICARE			:
73	|	-^	-	^					:
74	-ITUDE	-	-	^	INEPTITUDE	INEPT			:
75	-TOMY	-	+TOMY	^	EPISIOTOMY	EPISIO+TOMY	□		:
76	-ADO	-	-	^	AFICIONADO	AFICION			:
77	-TINA	-	-T	^	CONCERTINA	CONCERT			:
78	-TILE	-	-T	^	PROJECTILE	PROJECT			.
79	-MATIC		-M	^	SCHISMATIC	SCHISM	□		:
80	-SOME	-	+SOME	^	BURDENSOME	BURDEN+SOME	+		:
81	-OSIS	-	-US	^	THROMBOSIS	THROMBUS	□		:
82	-ICIDE	-	-	^	SPERMICIDE	SPERM	□		.
83	-ISSIMO	-	-O	^	PIANISSIMO	PIANO	□		:
84	-ETY	-	-	^	PERNICKETY	PERNICK			:
85	-IFY	-	-	^	DISQUALIFY	DISQUAL			:
86	-RAL	-	-RE	^	STRUCTURAL	STRUCTURE			:
87	-●	-	-	^					●

Table 10.2 Rulebase for 10-Letter-Long-Words Ending In **CVCV.**

Column five specifies the replacement string which is used to substitute either the primary or secondary suffix. This replacement string may be either the null string or any alphabetic string. In Rule 7 of Table 10.2, which handles10-letter-long-words ending in **CVCV**, the secondary string **-TR** is replaced by the string **-TER** to transform a word such as *magistrate* to *magister*. Rule 16, in Table 10.2, reduces the secondary string **-SIVE** by replacing it with **-SE** in order to reduce words such as *incisively* to *incise*. An example of a null-string replacement is found in Table 10.2, when Rule 10 replaces the secondary string **-B** by - (nothing) to tranform for example the word *exacerbate* into *exacer*.

Column six specifies which rule should be analyzed next. By the definitions given in BNF in Figure 10.7, two possibilities can occur. In either case, word evaluation is immediately halted whenever the control-flow terminators ^ , or ● are encountered. Case 1: The ● symbol indicates that the word has not been modified by any of the context-sensitive rules specified by its VNF schema. Whenever the procedure terminate' inder ● the word's ending was not matched by any of the suffix strings specified by the schema. In such cases the word is either an exception conforming to previously unencountered rules or it is simply a misspelled word! Case 2: Termination under ^ . All further analysis of a word halts whenever a ^ symbol is encountered. Termination under such circumstances means that the word has been analyzed and most likely modified. Under most circumstances the word has been reduced to a smaller base form by the application of a short sequence of production rules. However, in certain well defined circumstances, a set of production rules will be applied to a word which eventually returns it to its original form and notes it to be an exceptional case. Seven classes of exceptional cases are denoted in column nine of our schema whenever termination occurs under ^ .

Column nine is used to specify that an exceptional case has been encountered. Such exceptions are crudely characterized into seven linguistically meaningful subclasses. This field is used to state which

of the seven auxiliary rule types {✚, ☆, ◆, ▼, ■, ○, ❒ } best describes the exception.

A ✚ symbol denotes compound or concatenated wordform, such as *understate* (Figure 10.14, Rule 12), while an ☆ symbol is used to denote words whose replacement string is longer than the string deleted from them. An example of such a rule is given in Table 10.4 for 10-letter-long-words ending in **CVVC** by Rule 9 which would replace the secondary suffix **-ET** by **-ETE** to reduce for example the word *discretion* to *discrete*.

The ◆ symbol is used to denote suffixes which behave semantically as qualitative operators such as **-LESS** (Table 10.3, Rule 8) or **-ETTE** (Table 10.5, Rule 16) in words such as *effortless* or *maisonette*.

A ▼ symbol is used to denote an exceptional case in an otherwise valid context sensitive generic rule. Such exceptions may specify a unique word or class of words that share an exception which may be correctly handled by an auxiliary rule. For example while the suffix **-ITE** may be removed from most words in the class **CVCV** , given in Table 2, an auxiliary rule (such as Rule 23 in Table 10.2) must check that **-ITE** was not preceeded by **-WR-** in order to prevent producing 'mutant strings' from words such as *underwrite*.

The symbol ■ is used to specify those words which conform to the analysis criteria but are simply unanalyzable with the given VNF rulebase. Take for example the word *xxy* which is hyphenated as *xx-y* while its VNF homomorph the word *xxy* is hyphenated as *x-xy*. In that these two words share the same VNF structure and are hence structurally homomorphic it is impossible to specify a single VNF rule which will corrrectly hyphenate both words.

The symbol ○ is used to denote a word that has had its primary suffix returned to its reduced form after no further reductions were found to apply to its reduced form. For example the suffix **-INE** removed from the word *fireengine* by Rule 44 in Table 10.2 for 10-letter-long-words ending in **CVCV** is returned to its original form by a replacement string when termination occurs under ^ in Rule 49. An ○ type rule specifies that a word was returned to its original form after none of the secondary suffixes specified by the schema Rules 45,

46, 47, and 48 were found to apply to its reduced form. In this case while Rule 44 was correctly applied to *fireengine* neither Rules 45, 46, 47, nor 48 were applicable and thus Rule 49 was invoked, if and only if, all other alternatives were exhausted. The string *fireeng* is returned to its original wordform by applying a replacement string which simply reversed Rule 44.

The symbol ❒ is used to denote technical or scientific words which conform to the very rational set of rules [10..36] used to coin such words from Greek, Latin or other language bases. Take for example the word *episiotomy* which as noted in column 9, Table 10.2, Rule 74 conforms to such rules [10.36] where *episio* is derived from the greek *episio* meaning "region of pubes; vulva" and *tom* , which is used as the suffix **-TOMY**, meaning "cut".

Column seven gives an example of a word meeting the rule's specification while column eight illustrates the result of applying the rule to the word used as an example in column seven. For example, column 7 of Rule 59 in Table 10.3 specifies that the word *boyishness* would be reduced as shown in column 8 to its stem *boy*. This reduction is the result of the sequential application of Rule 1, Rule 3, Rule 53, and finally Rule 59.

RULE NUMBER	ENTRY POINT	PRIMARY SUFFIX	SECONDARY SUFFIX	REPLACEMENT STRING	NEXT RULE	WORD-EXAMPLE	ANALYSIS-RESULT	NOTE	PRE-REQUISITE
1		SS	-	-	§(13)				
2		|	-INE	-Y	^	UNTIDINESS	*un* TIDY		(3(9))
3		|	-NE	-	53	YELLOWNESS	YELLOW		(9)
4		|	-DRE	+DRESS	^	NIGHTDRESS	NIGHT+DRESS	+	(9)
5		|	-STRE	+STRESS	^	SEAMSTRESS	SEAM+STRESS	+	(9)
6		|	-PRE	-PRESS	^	DECOMPRESS	*decom* PRESS	O	(9)
7		|	-GRE	-GRESS	^	TRANSGRESS	*trans* GRESS	O	(9)
8		|	-LE	-	^	EFFORTLESS	EFFORT	◆	(9)
9		|	-E	-	^	STEWARDESS	STEWARD		
10		|	-FLO	+FLOSS	^	CANDYFLOSS	CANDY+ FLOSS	+	
11		|	-GLA	+GLASS	^	FIBREGLASS	FIBRE+GLASS	+	
12		|	-^	-	^				
13		-NT	-	-	§(32)				
14		|	-CE	-	^	IRIDESCENT	*tri* DES		(21)
15		|	-AME	-	^	REARMAMENT	*re* ARM		(17(21))
16		|	-IME	-Y	^	EMBODIMENT	*em* BODY		(17(21))
17		|	-ME	-	^	ENDEARMENT	*en* DEAR		(21)
18		|	-TE	-TENT	^	OMNIPOTENT	*omni* POTENT	O	(21)
19		|	-VE	-VENT	^	CIRCUMVENT	*ctr cum* VENT	O	(21)
20		|	-GE	-	^	ASTRINGENT	ASTRIN		(21)
21		|	-E	-	53	ANTECEDENT	*ante* CED		
22		|	-TA	-	^	ACCOUNTANT	*ac* COUNT		(26)
23		|	-PLA	-PLANT	^	TRANSPLANT	*trans* PLANT	O	(26)
24		|	-PA	-PANT	^	DISCREPANT	*dis* CREPANT	O	(26)
25		|	-SA	-SE	^	UNPLEASANT	PLEASE		(26)
26		|	-CHA	-CHANT	^	DISENCHANT	*dis en* CHANT	O	(26)
27		|	-RA	-RANT	^	RESTAURANT	RESTAURANT	O	(26)
28		|	-A	-	^	COMMANDANT	COMMAND		
29		|	-FRO	+FRONT	^	WATERFRONT	WATER+FRONT	+	
30		|	-MI	+MINT	^	PEPPERMINT	PEPPER+MINT	+	
31		|	-^	-NT	^	TRIUMPHANT	TRIUMPHANT	O	

32		-ING	-		§(40)	UNYIELDING	un YIELD			;
33		|	-EN	-	^	SWEETENING	SWEET			;
34		|	-Z	-ZE	^	STARGAZING	STARGAZE	☆		;
35		|	-L	-	^	CHANGELING	CHANGE			;
36		|	-DD	-D	^	FORBIDDING	for BID			;
37		|	-COM	+COMING	^	UNBECOMING	un BE+COMING	+		;
38		|	-SPR	+SPRING	^	HAIRSPRING	HAIR+SPRING	+		;
39		|	-^	-	^					;
40		-ST		-	§(46)					;
41		|	-I	-	^	MISOGYNIST	miso GYN			;
42		|	-MO	+MOST	^	BOTTOMMOST	BOTTOM+MOST	+		;
43		|	-BE	+BEST	^	SECONDBEST	SECOND+BEST	+		;
44		|	-LU	+LUST	^	WANDERLUST	WANDER+LUST	+		;
45		|	-^	-ST	^	WATERCREST	WATERCREST	○		;
46		-ISM	-	-	^	JOURNALISM	JOURNAL			;
47		-CS	-	-	§(53)					;
48		|	-BATI	-BAT	^	AEROBATICS	aero BAT	□	(49(50))	;
49		|	-ATI	-	^	RHEUMATICS	RHEUM	□	(50)	;
50		|	-TI	-T	^	GYMNASTICS	gym NAST	□		;
51		|	-TRI	-T	^	OBSTETRICS	ob STET	□		;
52		|	-^	-CS	^	NUCLEONICS	NUCLEONICS	○		;
53	=>	-SH		-	§(61)					;
54		|	-BRU	+BRUSH	^	PAINTBRUSH	PAINT+BRUSH	+		;
55		|	-BRA	+BRASH	^	WATERBRASH	WATER+BRASH	+		;
56		|	-DA	+DASH	^	BALDERDASH	BALDER+DASH	+		;
57		|	-FLE	+FLESH	^	HORSEFLESH	HORSE+FLESH	+		;
58		|	-FI	+FISH	^	CUTTLEFISH	CUTTLE+FISH	+		;
59		|	-I	-	^	BOYISHNESS	BOY		(59)	;
60		|	-^	-	^					;
61		-UL		-	§(64)	FRAUDULENT	FRAUD			;
62		|	-F	-	^	JOYFULNESS	JOY	◆		;
63		|	-^	-	^					;
64		-●	-	-	^					●

Table 10.3 Rulebase for 10-Letter-Long-Words Ending In **CVCC.**

Column ten is used to specify an essential but infrequently used constraint on the application of every rule in the schema. Column ten is used to enforce priorities whenever they are needed to ensure the correct sequential application of some set of related rules. Typically these constraints result from the inclusion of rules which detect the presence of embedded sub-strings in a schema; consider for example the abstract case of 'γ' or '$\beta\gamma$' in the string '$\alpha\beta\gamma$'. Under such circumstances the rulebase may be evaluated in the correct systematic order by using the information contained in column ten. As a more specific example consider Rules 1, 2, 3 and 8 given in Table 10.1. The intent of this schema is that the primary suffixes **-CY, -RY** and **-LY** specified by Rules 1, 2 and 3 must be evaluated before checking to see

if the word's ending matches the suffix string **-Y** specified in Rule 8. The entry found in column ten for Rules 1, 2, and 3 informs us that these rules must be evaluated before Rule 8 in order to meet this schema's intent. The constraint specified by column ten allows the rulebase to be used and maintained in an order that does not explicitly conform to the sequential dependencies that are concomitant with the necessary conceptual prerequisites that underlie such linguistic schema. Column ten is used to specify adherence to the topological orderings that constrain the schema.

The sequential order in which syntactic rules are applied is specified by the value of the next-rule-number found in column 6 of Table 10.1. The next-rule-number may specify termination under ^ or continuation of the evaluation process. Further computation continues by either transferring the evaluation process to some specified rule-number or simply evaluating the next rule.

The symbol | is used to instruct the system to continue with its sequential evaluation of our rulebase. If the | symbol is encountered in column 3 during the sequential search of the rulebase, such as that given in Table 10.1, a process of reverse chaining is used to transfer control flow to the rule-number referenced after the | symbol in column 6 of the nearest preceeding rule. Whenever a | symbol is encountered in column 5 the subsequent set of rules is encapsulated in a rectangular structure which blocks all rules bounded by the | symbol and the next ^ termination symbol. This blocking device is used solely to clarify the structure of the schemata presented in this chapter. The context of these schemata is easily depicted by nesting all 'secondary' or 'auxiliary' suffix rules under their dominant primary production rule and graphically encapsulating the set of rules into a block. Whenever two or more rules in such schemata share a common set of secondary rules, the 'common set' is blocked and treated conceptually as a 'subroutine' which may be accessed by a particular rule which serves as an entry point to an encapsulated or blocked structure. All such subroutine structures used within these schemata conform explicitly to the previously outlined constructs and notations. Thus it is possible to have both primary and secondary suffix rules within a subroutine.

The inference algorithm adopted in this work is based on the concept of the implicit sequential evaluation of a set of rules. This implicit strategy is the default sequencing process which can be overridden whenever a rule's suffix string matches a word's ending and column 5 specifies either termination under ^ or continued evaluation at the rule-number specified. Under all other circumstances the dominant process of sequential evaluation of the schema's rulebase is adopted until either a string-match is found; a | symbol is encountered; or termination occurs under the ● symbol. Whenever a | symbol is encountered the implicit default control-flow process is interrupted and control-flow is transferred to the rule specified by the second term of the nearest preceeding | symbol in column 5. Under such circumstances the | symbol is ignored and control flow is transferred to the rule specified by the second term of the control-flow label given in column 5 of these schemata. Under normal circumstances this procedure involves simply querying column five of the preceeding rule. However, in order to ensure the robustness of these schemata whenever control-flow is transferred under the | symbol, a backward chaining process is invoked until a preceding rule containing the necessary | operator in column five is encountered. This simple procedure protects the system from important errors which could otherwise occur by accidently directing the start of the evaluation of a schemata's subroutine to begin at a rule which was not declared, *a priori*, to be an entry point. Rules skipped in such an erroneous manner would never be evaluated and can be viewed as unreachable nodes in the schema's sequential state graph.

A sequential state graph, such as that drawn in Figure 10.8 from the schema given in Table 10.1, is a decision-to-decision point, (D-D), graph [10..37, 10..38] in which each rule or node can specify two or fewer control-flow transfers and hence has an outdegree of < 2 and an arbitrary indegree.

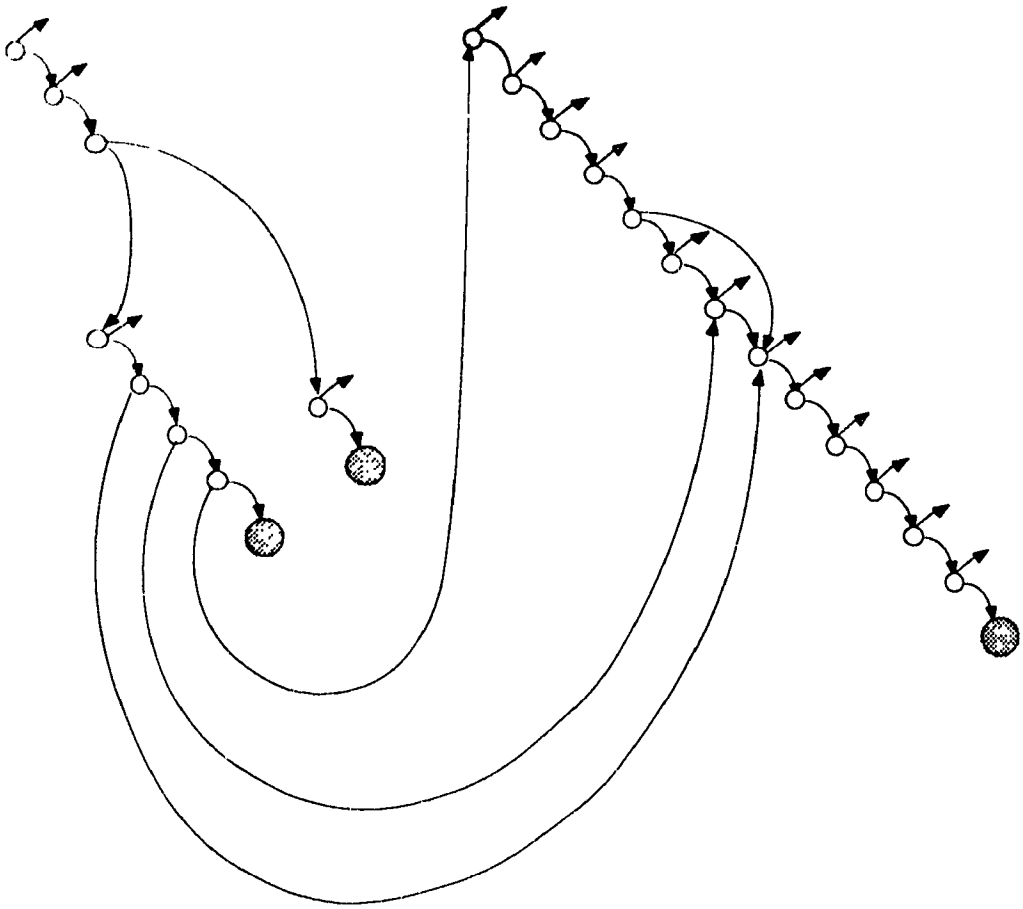

Figure 10.8 Sequential State Graph for the schema given in Table 10.1 which depicts the rulebase for 10-letter-long words ending in **CCCV**. (CASE II behavior)

Figure 10.8 is drawn in a manner that depicts the implicit underlying 'cascadence' or waterfall approach used in the design of this system's rulebases. Every rule is represented by a circle in these graphs. Those rules which direct the procedure to terminate under either ^ or ● are denoted by larger shaded circles. Thus both Rule 8 and 10 in Figure 10.8 and Table 10.1 instruct the system to terminate the evaluation of the word in question. Rule 24 also instructs the system to terminate evaluation of the word in question. Termination from Rule 8 occurs after a partial match of the word's ending by the suffix-string specified by Rule 3 in Figures 10.8 and Table 10.1. Termination specified by Rule 10 occurs whenever every primary suffix specified in

the rulebase given in Table 10.1 has been queried with no success. Thus termination at Rule 10 occurs under the ● symbol. Termination at Rule 20 however occurs only after at least a partial match of the suffix's terminal sub-string. Termination at Rule 20 always occurs as the result of the evaluation of a subroutine or sub-rulebase specified by Rules 11 through 20 (R11..20). This subroutine may be reached from four different paths: { (R11..20), (R11..15, R18..24), (R17..24), (R18..24)}. All of these paths originate from a single cluster or block of rules which is only reachable from Rule 3.

Figures 10.8 and 10.10 give the sequential and parallel flowgraphs of the evaluation procedure for the rulebase specified in Table 10.1.

The north-eastwardly directed arcs in the schema graph depicted in Figure 10.8 specifies termination under the ^ symbol whenever the rule's guard is satisfied. All nodes in such decision graphs must also have a south-eastwardly directeu arc which is determined as a consequence of the inference scheme adopted in this model. Our inference scheme defaults to the evaluation of the next rule whenever the present guard is not matched (ie: the rule fails to fire). Consider for example Rule 3 in Figures 10.8 and Table 10.1. When its guard is matched Rule 3 fires and Rules 4 through 7 are subsequently evaluated in sequence for secondary matches. Rule 8 is invoked only if Rule 3 fired and all of its subservient rules R4..7 were inapplicable. However if both Rule 3 and Rule 5 fired, then Rule 24 invoked termination only when all of its subservient rules R18..23 were inapplicable. The case is considerably more complex when Rule 3, Rule 7, Rule 15 and R18..23 do not apply. When Rule 3 and Rule 7 apply and Rule 24 invokes termination then Rule 16 and Rule 17 may or may not have been evaluated. When termination occurs at Rule 16 or Rule 17 then both Rule 3 and Rule 7 must have been satisfied while if Rule 17 specified termination then either (Rule 3 and Rule 7) or (Rule 3 and Rule 6) may have been satisfied.

Within this nondeterministic paradigm it is impossible to precisely determine the exact path that led to a specific termination point [10.35, 10.39]. Consider Rule 18 which may have been reached by immediate transition from Rule 17, Rule 15 or Rule 5 and which

can lead to at least one more further word reduction through satisfaction of R18..23 or termination at Rule 24 with no further word reduction. Similarily Rule 17 could be reached by a single transition from either Rule 6 or Rule 16. Rule 6, in turn, was arrived at by satisfying Rule 3 while Rule 10 was arrived at by first satisfying Rule 3, and subsequently satisfying Rule 7.

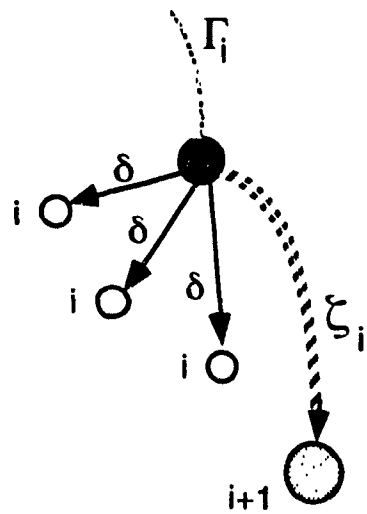

Figure 10.9 A 'Prefect' and its components.

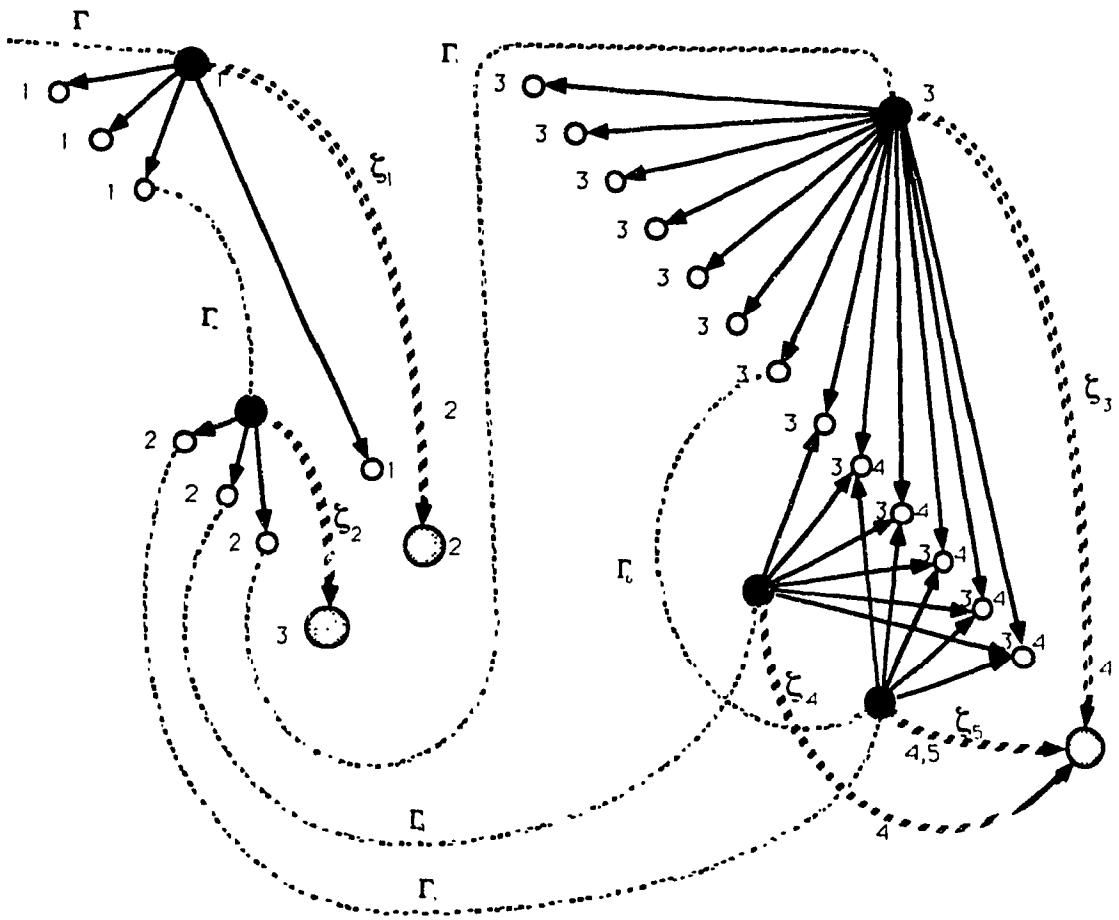

Figure 10.10 Parallel state graph for the schema given in Table 10.1 which depicts the rulebase for 10-letter-long words ending in **CCCV**.

Such sequential representations are consistent with more conventional views of computation. A functionally equivalent view of this computation is depicted in parallel state graph drawn in Figure 10.9 for the schema outlined in Figure 10.8 and Table 10.1. Figure 10.10 demonstrates the conceptual ease with which the schema depicted in Figures 10.8 and Table 10.1 may be evaluated by parallel processing. In addition to the two types of circular nodes introduced in Figure 10.8, there is a need to introduce another type of construct, in Figures 10.9 and Figure 10.10. The symbol, , denotes an adjuvant administrative process referred to as a 'prefect'. A is depicted as communicating with the schemata graphs through three operationally distinct routes denoted $< \Gamma, \zeta, \delta >$ each of which are described below. A prefect has eight basic functions which are used to coordinate parallel processing in a schema graph :

Hail: queries all of the prefect's subservient processes.

Log: catalogues all subservient processes that responded to **Hail.**

Listen: polls each of the **Log**ged processes until they respond to a **Delagated** task.

Delegate: broadcasts a task to all of the prefect's subservient processes.

Compile: uses **Listen** and **Log** to update the status of the prefect's **Log**ged processes. When **Compile** detects the failure of all **Log**ged processes it terminates **Wait**.

Respond: uses **Listen** to flag the success of a **Delegated** process. When **Respond** detects success it then immediately terminates the **Wait** process and the prefect procedure.

Wait: suspends its own processing until either **Compile** or **Respond** terminates **Wait.**

Forward: whenever **Wait** is terminated by **Compile** the **Forward** procedure is used by the prefect to terminate, continue processing or default to exception handling or error processing rules.

The behaviour of a prefect is outlined as follows:

The prefect ⅀ is invoked by a request which is depicted in Figure 10.9 as being passed-by-value on the path or route denoted by Γ. This request may have originated from either an external calling procedure, such as that denoted by S in Figure 10.9, or as the result of a rule firing. An example of the latter case is depicted in Figure 10.10 where $\Gamma 3$ is invoked by firing Rule 7 in Figure 10.10. Γ paths are denoted in the schemata outlined in Figures 10.9 and 10.10 as dashed lines. A query referred to as 'Hail' is used to initiate a handshake routine (Hail & Log) which determines and notes the availability of δ rules. Paths from the prefect to its δ rules are denoted by dotted lines in the schemata outlined in Figures 10.9 and 10.10. 'Log' is effectively an initialization procedure, which establishes how many and which rules are engaged by a prefect. This information is needed to guarantee termination under the default procedure, 'Forward', which is routed on a ζ path. Once initialized the prefect ⅀ broadcasts or Delegates its request to each of the sequestered δ rules filed by Log. The prefect then assumes a Wait state which prevails under Listen until the δ rules respond to the Delegated task that was broadcast to them. δ rules report to prefects in either one of two possible ways. The first possibility occurs when all engaged δ rules return a declaration to the prefect that they are inapplicable. The second possibility occurs when one of the δ rules queried returns a declaration through Respond that it has fired. Whenever a δ rule informs a prefect that it was applicable and hence fired, the prefect procedure immediately terminates Wait and the evaluation procedure proceeds in accordance with the applicable δ rule. If on the other hand, all sequestered δ rules inform the prefect, through Listen & Log, that they were inapplicable then Compile terminates the prefect's Wait state. The prefect then uses its exception route ζ to Forward the result of its computation. Whenever processing continues under the perfect's exception route ζ the prefect, in fact, required two sequentially distinct computational steps to proceed with its analysis. The first of these the α process is inherently a parallel nondeterministic process, while the second which is referred to as the β process involves simple sequential computation. The total number of computational steps, ($\Sigma\alpha + \Sigma\beta$), needed to arrived at a

given rule in our parallel schema are marked on each 'arc' of the five prefects ≡ 1, ≡ 2, ≡ 3, ≡ 4, ≡ 5 found in Figure 10.10. Exception rules , ζ, are denoted by double lines in the schemata illustrated in Figures 10.9 and Figure 10.10. Under most circumstances ζ routes are used to specify termination under ^.

In their simplest conceptual form Γ routes require a single passed-by-value parameter whereas ζ routes require a single passed-by-result parameter and δ paths require both a passed-by-result parameter and a passed-by-value parameter.

The parallel evaluation of the rulebase given in Table 10.1 terminates under ^ in three cases (see Figure 10.10). The first of which occurs under Rule 10 and involves a single prefect and two steps of computation, ($1\alpha + 1\beta$). The second case occurs under Rule 8 and involves two prefects and three sequential steps, ($2\alpha + 1\beta$), of processing while the third case under Rule 24 may be reached in either four ($3\alpha + 1\beta$), or five ($4\alpha + 1\beta$), steps of processing which involved either three or four prefects. For instance at ≡ 1 in Table 10.1, if Rule 3 is satisfied and then Rule 7 is satisfied at ≡ 2 followed by Rule 17 firing at ≡ 3 and none of the five rules: R19..23 of the concomitant rules specified by ≡ 4 are satisfied, then Rule 24 is reached by a ζ route under the exception clause of ≡ 4 which specifies termination under ^. Similarily Rule 24 may be reached from ≡ 1 by satisfying Rule 3 and Rule 6 at ≡ 2 followed by enacting the termination clause ^ associated with ≡ 5. The longest parallel computation encounterable under these conditions is thus ($4\alpha + 1\beta$).

The number of sequential steps, or cycles, in the parallel computation needed to arrive at each node in Figure 10.10 is given as the depth of computation on each arc incident to rules R1..24 in Figure 10.10.

While both the sequential and parallel algorithms may be implemented with many different control structures, using either static or dynamic memory allocation techniques, the recursive versions given in Figures 10.4 and 10.6 perhaps best capture the simplicity and power of the schemata reported here. The system's rulebase may be maintained in an optimum probabilistic order by a

number of computational steps, ($\Sigma\alpha + \Sigma\beta$), needed to arrived at a given rule in our parallel schemae are marked on each 'arc' of the five prefects ≣ 1, ≣ 2, ≣ 3, ≣ 4, ≣ 5 found in Figure 10.10. Exception rules , ζ, are denoted by double lines in the schemae illustrated in Figures 10.9 and Figure 10.10. Under most circumstances ζ routes are used to specify termination under ^.

In their simplest conceptual form Γ routes require a single passed-by-value parameter whereas ζ routes require a single passed-by-result parameter and δ paths require both a passed-by-result parameter and a passed-by-value parameter.

The parallel evaluation of the rulebase given in Table 10.1 terminates under ^ in three cases (see Figure 10.10). The first of which occurs under Rule 10 and involves a single prefect and two steps of computation, ($1\alpha + 1\beta$). The second case occurs under Rule 8 and involves two prefects and three sequential steps, ($2\alpha + 1\beta$), of processing while the third case under Rule 24 may be reached in either four ($3\alpha + 1\beta$), or five ($4\alpha + 1\beta$), steps of processing which involved either three or four prefects. For instance at ≣ 1 in Table 10.1, if Rule 3 is satisfied and then Rule 7 is satisfied at ≣ 2 followed by Rule 17 firing at ≣ 3 and none of the five rules: R19..23 of the concomitant rules specified by ≣ 4 are satisfied, then Rule 24 is reached by a ζ route under the exception clause of ≣ 4 which specifies termination under ^. Similarily Rule 24 may be reached from ≣ 1 by satisfying Rule 3 and Rule 6 at ≣ 2 followed by enacting the termination clause ^ associated with ≣ 5. The longest parallel computation encounterable under these conditions is thus ($4\alpha + 1\beta$).

The number of sequential steps, or cycles, in the parallel computation needed to arrive at each node in Figure 10.10 is given as the depth of computation on each arc incident to rules R1..24 in Figure 10.10.

While both the sequential and parallel algorithms may be implemented with many different control structures, using either static or dynamic memory allocation techniques, the recursive versions given in Figures 10.4 and 10.6 perhaps best capture the simplicity and power of the schemae reported here. The system's

self-organizing scheme [10.40 , 10.41] which ensures that the most likely suffixes are dynamically assigned high priority **RULE_NUMBER**s and, hence, evaluated first by a sequential search scheme. Sequential search on a frequency-ordered list guarantees O(Lg(N)) behavior [10.42]. In that the most likely occuring suffix can be statically determined from statistical data [10.12, 10.14] it is seldom that the overhead encountered in maintaining a self-organizing structure is truly warranted [10.42].

10.7 RESULTS

For the sake of brevity we will restrict the discussion here to the analysis of our results obtained for all 10-letter-long-words listed in the OPD. These results are however typical of those obtained for words of other lengths and VNF [10.14].

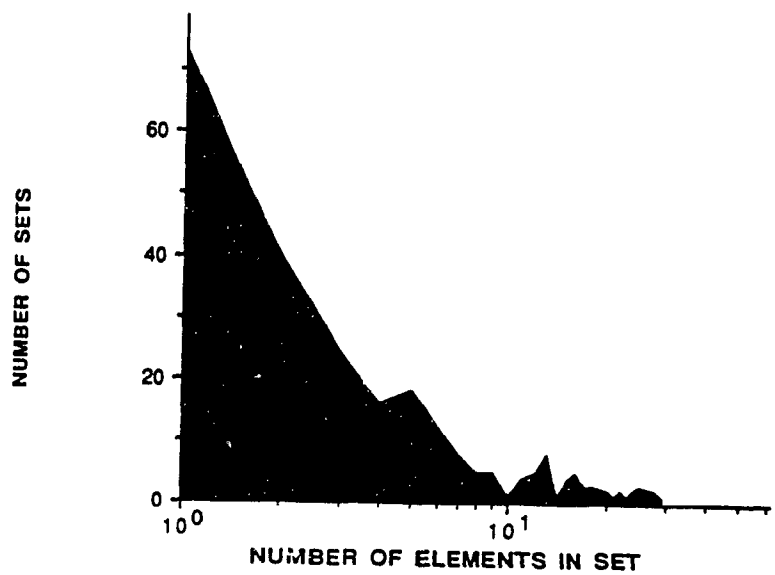

Figure 10.11 Least densely populated sets for 10-letter-long words. This plot depicts the number of VNF sets of a given size. Abscissa: Rank-ordered set size. Ordinate: Number of VNF sets found.

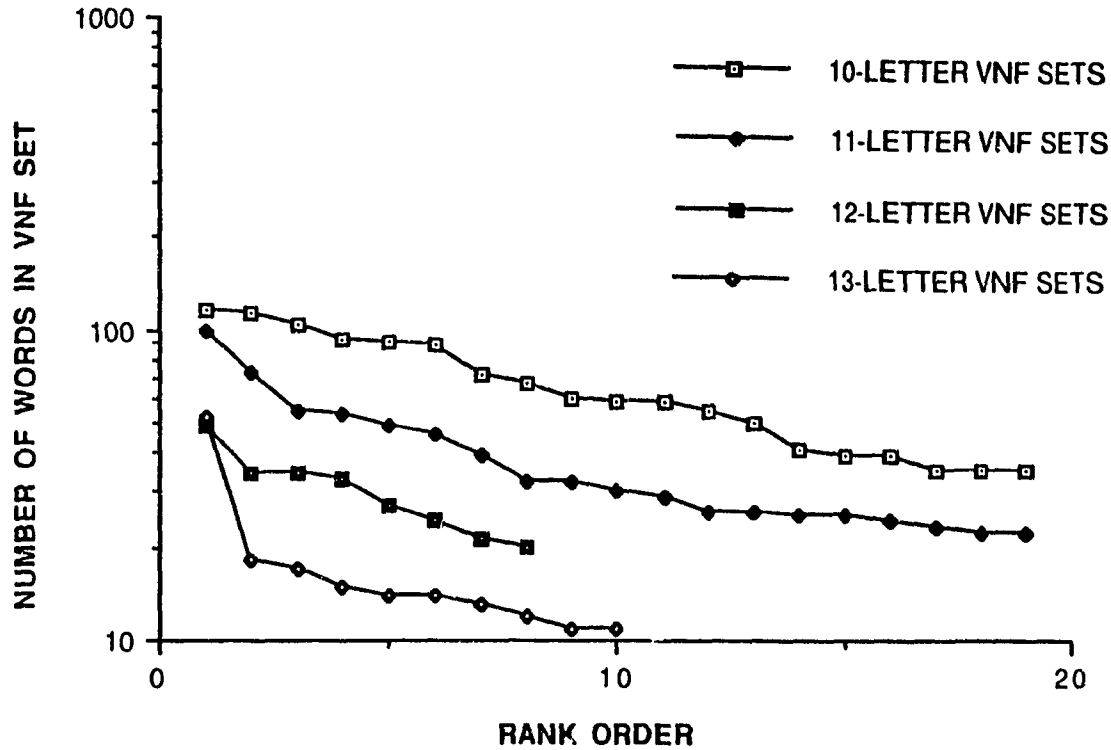

Figure 10.12 Most densely populated VNF sets for 10-letter -long words. This plot depicts the number of VNF sets of a given size. Abscissa: Rank-ordered set size. Ordinate: Number of VNF sets found.

The first observation, which can be drawn from Figures 10.11 and 12, is that relatively few of the possible 1,024 VNF groups describing 10-letter-words are densely populated, while over seventy VNF groups are sets with but a single element. In all, over 990 VNF groups are needed to account for all ten-letter-words found in the OPD.

The simple product of the VNF group's rank, ρ, and its frequency, f, is a constant for VNF sets with a small number of elements ($f\rho$ = 75.8, σ=9.85 for 10-letter-long-words computed for ρ =1..5) This result is in accordance with Zipf's law of rarely occurring or sparsely populated sets [10.43]. A similar analysis of the twenty

most densely populated VNF sets demonstrates their decay in size is accountable as a simple exponential function of rank.

A simple analysis of the largest nineteen VNF groups given in Figure 10.12 illustrates that six 4-letter-long suffix groups: { **CVCV**, **CVCC**, **CVVC**, **VCCV**, **VCVC**, **CCVC** } of respective size: { 310, 290, 261, 239, 108, 35 } and composed of the following number of VNF sets: { 4, 5, 3, 4, 2, 1 } encompasses the suffix structure of all the most densely populated VNF groups found for 10-letter-long-words listed in the OPD.

The rulebases given in Tables 10.2, 10.3, 10.4, 10.5, 10.6 and 10.7 are derived for suffixes for 10-letter-long-words ending in { **CVCV**, **CVCC**, **CVVC**, **VCCV**, **VCVC**, **CCVC** }.

PARTITIONS CONFORMING TO A HAND ANALYSIS OF TEN LETTER WORDS ENDING IN CVVC

RULE NUMBER	ENTRY POINT	PRIMARY SEGMENT	SECONDARY SEGMENT	REPLACEMENT STRING	NEXT RULE	WORD-EXAMPLE	ANALYSIS-RESULT	NOTE	PRE-REQUISITE
1		-ION	-	-	↓(13)	ADMONITION	ADMONIT		
2		|	-VIS	+VISION	^	TELEVISION	TELE+VISION	+	
3		|	-MOT	+MOTION	^	LOCOMOTION	LOCO+MOTION	+	
4		|	-MPT	-ME	^	ASSUMPTION	ASSUME		
5		|	-RPT	-RB	^	ABSORPTION	ABSORB		(6)
6		|	-IPT	-IBE	^	ASCRIPTION	ASCRIBE		
7		|	-TENT	-TAIN	^	ABSTENTION	ABSTAIN		
8		|	-IZAT	-	^	IONIZATION	ION		(11)
9		|	-ET	-ETE	^	DISCRETION	DISCRETE	☆	
10		|	-UT	-UTE	^	DEVOLUTION	DEVOLUTE	☆	
11		|	-AT	-	^	ALIENATION	ALIEN		
12		|	-^	-	^				
13		-UT		-	↓(18)				
14		|	-ABO	+ABOUT	^	ROUNDABOUT	ROUND+ABOUT	+	
15		|	-NA	+NAUT	^	JUGGERNAUT	JUGGER+NAUT	+	
16		|	-O	-	^				
17		|	-^	-	^				
18		-AN		-	↓(26)				
19		|	-EAN	-	^	ANTIPODEAN	ANTIPOD	□	
20		|	-CE	-	^	CRUSTACEAN	CRUSTA	□	
21		|	-GI	-GY	^	THEOLOGIAN	THEOLOGY	□	
22		|	-ICI	-Y	^	BEAUTICIAN	BEAUTY		
23		|	-TRI	-	^	PEDESTRIAN	PEDES		
24		|	-U	-	^	GARGANTUAN	GARGANT		
25		|	-^	-	^				
26		-UM	-	-	↓(31)				
27		|	-E	-	^	PERITONEUM	PERITON	□	
28		|	-RI	-	^	OPPROBRIUM	OPPROB	□	
29		|	-I	-Y	^	PROSCENIUM	PROCENY	□	
30		|	-^	-	^				

31	-AL	-	-	‖(35)				:
32	|	-U	-E	^	INDIVIDUAL	INDIVIDE		:
33	|	-I	-	^	JANITORIAL	JANITOR		:
34	|	-^	-	^				:
35	-IES	-	-Y	‖(38)	HUMANITIES	HUMANITY		:
36	|	-RY	-	^	TOILETRIES	TOILET		:
37	|	-^	-	^				:
38	-ER	-	-	‖(44)				:
39	|	-E	-	^	COMMANDEER	COMMAND		:
40	|	-I	-Y	^	HUMIDIFIER	HUMIDIFY		:
41	|	-U	-	^	CATALOGUER	CATALOG		:
42	|	-TE	-T	^	CHARIOTEER	CHARIOT		:
43	|	-^	-	^				:
44	-OUS	-	-	‖(54)	GRATUITOUS	GRATUIT		:
45	|	-MO	-	^	SYNONYMOUS	SYNONY	◆	:
46	|	-VO	+VOUS	^	RENDEZVOUS	RENDEZ+VOUS	+	:
47	|	-OR	-Y	^	MELODOROUS	MELODY		:
48	|	-VELL	-VEL	^	MARVELLOUS	MARVEL		:
49	|	-CUL	-CLE	^	MIRACULOUS	MIRACLE		:
50	|	-MIN	-ME	^	VOLUMINOUS	VOLUME		:
51	|	-ATR	-	^	IDOLATROUS	IDOL		:
52	|	-REN	-ER	^	GANGRENOUS	GANGER		:
53	|	-^	-	^				:
54	-IAR	-	-	^	UNFAMILIAR	UNFAMIL		:
55	-EON	-	-	^	CURMUDGEON	CURMUDG		:
56	-CHAUN	-	+CHAUN	^	LEPRECHAUN	LEPRE+CHAUN	+	:
57	-POID	-	-	^	ANTEROPOID	ANTERO	□	:
58	-OID	-	-	^	RHEUMATOID	RHEUMAT	□	:
69	-IED	-	-Y	^	STRATIFIED	STRATIFY		:
60	-SCHAUM	-	+SCHAUM	^	MEERSCHAUM	MEER+SCHAUM	+	:
61	-●	-	-	^				●

Table 10.4 Rulebase for 10-Letter-Long-Words Ending In **CVVC.**

PARTITIONS CONFORMING TO A HAND ANALYSIS OF TEN LETTER WORDS ENDING IN VCCV

RULE NUMBER	ENTRY POINT	PRIMARY SEGMENT	SECONDARY SEGMENT	REPLACEMENT STRING	NEXT RULE	EXAMPLE	REDUCED STRING	COMMENT
1		-LE	-	-	‖	-	-	
2			-B	-B	39	-	-	
3			-NAC	+NACLE	^	TABERNACLE	TABER+NACLE	+
4			-EL	-ELLE	^	IMMORTELLE	*im* MORT	●
5			-DRIL	+DRILLE	^	ESPADRILLE	ESPA+DRILLE	+
6			-VIL	+VILLE	^	VAUDEVILLE	VAUDE+VILLE	+
7			-BEET	+BEETLE	^	STAGBEETLE	STAG+BEETLE	+
8			-^	-	^			
9		-TY	-	-	‖	-	-	
10			-L	-L	43	DISLOYALTY	*dis* LOYAL	
11			-^	-	^			
12		-TE	-	-	‖	-	-	
13			-L	-L	43	-	-	
14			-TAN	-TANTE	^	DILETTANTE	DILET	●
15			-AN	-E	^	CONFIDANTE	*con* FIDE	
16			-ET	-ETTE	^	MAISONETTE	MAISON	●
17			-FAS	+FASTE	^	TOOTHFASTE	TOOTH+FASTE	+
18			FOR	-FORTE	^	PIANOFORTE	PIANO	●
19			-^	-	^			
20		-LY	-	-	‖	DISCREETLY	*dis* CREET	
21			-UL	-	47	-	-	
22			-US	-	51	-	-	
23			-ED	-	56	DESIGNEDLY	*de* SIGN	
24			-IAR	-Y	^	FAMILIARLY	FAMILY	
25			-^	-	43			
26		-CY	-	-	60	-	-	
27		-CE	-	-	60	-	-	

28		-HY	.	.	!	-	-		
29			-PAT	+PATH	^	OSTEOPATHY	OSTEO+PATH	+	
30			-GRAP	+GRAPH	^	TOPOGRAPHY	TOPO+GRAPH	+	
31			-SOP	+SOPH	^	PHILOSOPHY	PHILO+SOPHY	+	
32			.^	.	^				
33		-RY	.	.	!	-	-		
34			-AT	-	^	PSYCHIATRY	PSYCHI		
35			-BER	+BERRY	^	STRAWBERRY	STRAW+BERRY	+	
36			-MAR	+MARRY	^	INTERMARRY	INTER+MARRY	+	
37			.^	.	^				
38		-^	.	.	^				
39	=>	-IB	.	.	^	INSENSIBLE	*in* SENS		
40		-IAB	.	-Y	^	UNDENIABLE	*un* DENY		(42)
41		-VAB	.	-VE	^	ACHIEVABLE	ACHIEVE		(42)
42		-AB	.	-	^	UNBEATABLE	*un* BEAT		
43	=>	-TICAL	.	-S	^	EROTICALLY	EROS		
44		-BAB	.	-B	^	ABSORBABLY	*ab* SORB		
45		-ICALLY	-	-	^				
46		.^	.	.	^				
47	=>	-F	.	.	^	YOUTHFULLY	YOUTH		
48			-I	-Y	^	FANCIFULLY	FANCY		
49			.^	.	^				
50		-^	.	.	^				
51	=>	-CIO	.	-H	^	LUSCIOUSLY	LUSH		(52(54)
52		-IO	.	-	^	SPURIOUSLY	SPUR		(54)
53		-TRO	.	-TER	^	DEXTROUSLY	DEXTER		(54)
54		-O	-	-	^	COVETOUSLY	COVET		
55		.^	.	.	^				
56	=>	-IT	-	-IT	^	SPIRITEDLY	SPIRIT		
57		-RR	.	-R	^	OCCURRENCE	OCCUR		
58		-LL	.	-L	^	EXCELLANCE	EXCEL		
59		-^	.	.	^				
60	=>	-AN	.	.	!	EXUBERANCE	*ex* UBER		
61			-IZ	-US	^	COGNIZANCE	COGNUS		(68)
62			-UR	-URY	^	LUXURIANCE	LUXURY	•	
63			-I	-Y	^	COMPLIANCE	*com* PLY		
64			-TT	-T	^	ADMITTANCE	*ad* MIT		
65			-STR	-STRANCE	^	MONSTRANCE	MONSTRANCE	•	
66			.^	.	^				
67		-EN	-	.	56	ABSTINENCE	*ab* STIN		
68		-DES	-	-DE	^	RECRUDESCE	*re* CRUDE		
69		.^	.	.	^				

Table 10.5 Rulebase for 10-Letter-Long-Words Ending In **VCCV.**

VCVC HAND ANALYSIS

RULE NUMBER	ENTRY POINT	PRIMARY SEGMENT	SECONDARY SEGMENT	REPLACEMENT STRING	NEXT RULE	WORD-EXAMPLE	ANALYSIS-RESULT	NOTE	PRE-REQUISITE	
1		-IC	-	-	‖(9)	AUTOCRATIC	AUTOCRAT			;
2		|	-MET	-	^	ARITHMETIC	ARITH			;
3		|	-AT	-	^	EMBLEMATIC	EMBLEM			;
4		|	-ET	-Y	^	APOLOGETIC	APOLOGY			;
5		|	-OM	-OMY	^	UNECONOMIC	ECONOMY	☆		;
6		|	-TIF	-CE	^	SCIENTIFIC	SCIENCE			;
7		|	-GEN	-	^	PHOTOGENIC	PHOTO			;
8		|	-^	-	^					;
9		-AL	-	-	‖(24)	ABORIGINAL	ABORIGIN			;
10		|	-TIC	-SE	^	ELLIPTICAL	ELLIPSE		(16)	;
11		|	-DIC	-	^	PERIODICAL	PERIOD		(16)	;
12		|	-RIC	-ER	^	THEATRICAL	THEATER		(16)	;
13		|	-NIC	-	^	RABBINICAL	RABBI		(16)	;
14		|	-FIC	-F	^	PONTIFICAL	PONTIF		(16)	;
15		|	-MIC	-MY	^	ANATOMICAL	ANATOMY		(16)	;
16		|	-IC	-Y	^	ANARCHICAL	ARARCHY			;
17		|	-TYP	-TYPE	^	ARCHETYPAL	ARCHETYPE	☆		;
18		|	-IAC	-	^	DEMONIACAL	DEMON			;
19		|	-ER	-	^	PERIPHERAL	PERIPH			;
20		|	-DUR	-ED	^	PROCEDURAL	PROCEED			;
21		|	-IT	-	^	CONGENITAL	CONGEN			;
22		|	-ICID	-I	^	FUNGICIDAL	FUNGI	□		;
23		|	-^	-	^					;
24		-ED	-	-E	‖(28)	INEBRIATED	INEBRIATE			;
25		|	-NE	-N	^	UNLEAVENED	UNLEAVEN			;
26		|	-ULATE	-	^	ACIDULATED	ACID			;
27		|	-^	-	^					;
28		-OR	-	-E	‖(32)	SUPERVISOR	SUPERVISE			;
29		|	-ATE	-E	^	DESECRATOR	DESECRE			;
30		|	-ITE	-E	^	COMPETITOR	COMPETE			;
31		|	-^	-	^					;
32		-MAN	-	+MAN	‖(36)	JOURNEYMAN	JOURNEY+MAN	+		;
33		|	-WO	+WOMAN	^	SALESWOMAN	SALES+WOMAN	+		;
34		|	-HUM	+HUMAN	^	SUPERHUMAN	SUPER+HUMAN	+		;
35		|	-^	-	^					;
36		-ER	-	-	‖(39)	MALINGERER	MALINGER			;
37		|	-IZ	-	^	LIQUIDIZER	LIQUID			;
38		|	-^	-	^					;
39		-ES	-	-	^	ABORIGINES	ABORIGIN			;
40		-●	-	-	^					●

Table 10.6 Rulebase for 10-Letter-Long-Words Ending In **VCVC.**

PARTITIONS CONFORMING TO A HAND ANALYSIS OF TEN LETTER WORDS ENDING IN CCVC

RULE NUMBER	ENTRY POINT	PRIMARY SEGMENT	SECONDARY SEGMENT	REPLACEMENT STRING	NEXT RULE	WORD-EXAMPLE	ANALYSIS-RESULT	NOTE	PRE-REQUISITE
1		-PLED	-	-PLE		UNEXAMPLED	UNEXAMPLE		(8)
2		-GLED	-	-GLE	^	BEDRAGGLED	BEDRAGLE		(8)
3		-VED		-VE	^	UNDESERVED	UNDESERVE		(8)
4		-GED	-	-GE	^	UNABRIDGED	UNABRIDGE		(8)
5		-CED	-	-CE	^	UNBALANCED	UNBALANCE		(8)
6		-ED	-	-	‖(13)				
7		|	-PP	-P	^	WORSHIPPED	WORSHIP		
8		|	-LL	-L	^	UNEQUALLED	UNEQUAL		
9		|	-SS	-S	^	NONPLUSSED	NONPLUS		
10		|	-TT	-T	^	HALFWITTED	HALFWIT		
11		|	-C	-CE	^	PRONOUNCED	PRONOUNCE	☆	
12		|	-^	-	^				
13		-AL	-	-	‖(17)	MONUMENTAL	MONUMENT		
14		|	-CHR	-CHER	^	SEPULCHRAL	SEPULCHER	☆	
15		|	TR	-TRA	^	ORCHESTRAL	ORCHESTRA	☆	
16		|	-^	-	^				
17		-ER	-	-	‖(40)				
18		|	-PP	-P	^	WORSHIPPER	WORSHIP		
19		|	-ERN	-	^	NORTHERNER	NORTH		
20		|	-BB	-B	^	LANDLUBBER	LANDLUB		
21		|	-LL	-L	^	VICTUALLER	VICTUAL		
22		|	-MM	-M	^	PROGRAMMER	PROGRAM		
23		|	-NN	-N	^	FORERUNNER	FORERUN		
24		|	-MAND	-MANDER	^	SALAMANDER	SALAMANDER	○	
25		|	-GETH	-GETHER	^	ALTOGETHER	ALTOGETHER	○	
26		|	-AFT	+AFTER	^	THEREAFTER	THERE+AFTER	+	
27		|	-BURG	+BURGER	^	BEEFBURGER	BEEF+BURGER	+	
28		|	-MAST	+MASTER	^	POSTMASTER	POST+MASTER	+	
29		|	-FIND	+FINDER	^	STARFINDER	STAR+FINDER	+	
30		|	-SIST	+SISTER	^	HALFSISTER	HALF+SISTER	+	
31		|	-MOTH	+MOTHER	^	STEPMOTHER	HALF+MOTHER	+	
32		|	-FATH	+FATHER	^	STEPFATHER	STEP+FATHER	+	
33		|	-LADD	+LADDER	^	STEPLADDER	STEP+LADDER	+	
34		|	-FING	+FINGER	^	FOREFINGER	FORE+FINGER	+	
35		|	-LETT	+LETTER	^	NEWSLETTER	NEWS+LETTER	+	
36		|	-COPT	+COPTER	^	HELICOPTER	HELI+COPTER	+	
37		|	-MINIST	+MINISTER	^	ADMINISTER	AD+MINISTER	+	
38		|	-^	-	^				
40		-EN	-		‖(43)	DISHEARTEN	DISHEART		
41		|	-GARD	+GARDEN	^	ROCKGARDEN	ROCK+GARDEN	+	
42		|	-^	-	^				
43		-IC	-	-	‖(51)	OPTIMISTIC	OPTIMIST		
44		|	-LIST	-L	^	FATALISTIC	FATAL		
45		|	-NOST	-NOSE	^	DIAGNOSTIC	DIAGNOSE		
46		|	-CENTR	+CENTER	^	CONCENTRIC	CON+CENTER	☆	
47		|	-METR	+METRIC	^	BAROMETRIC	BARO+METRIC	□	
48		|	-AST	-AR	^	SCHOLASTIC	SCHOLAR		
49		|	-MALM	-	^	OPHTMALMIC	OPHT+MALMIC	□	
50		|	-^	-	^				
51		OR	-	-	‖(54)	INSTRUCTOR	INSTRUCT		
52		|	-LL	-L	^	CHANCELLOR	CHANCEL		
53		|	-^	-	^				
54		-NIK		-	^	KIBBUTZNIK	KIBBUTZ	◆	
55		-FUL			^	DELIGHTFUL	DELIGHT	◆	
56		-THES		TH	^	BEDCLOTHES	BEDCLOTH		
57		ES		-E	^	SPECTACLES	SPECTACLE		
58		ENDUM		-E	^	REFERENDUM	REFERE	□	
59		-GRAM	-		^	CRYPTOGRAM	CRYPTO-GRAM	+	
60		-PLET	-	-U	^	QUADRUPLET	QUADRU	□	
61		SHIP	-	-	^	CENSORSHIP	CENSOR+SHIP	+	
62		MAN	-	-	^	GROUNDSMAN	GRANDS+MAN	+	
63		BOX	-	-	^	CHATTERBOX	CHATTER+BOX	+	
64		CRAT	-	-	^	ARISTOCRAT	ARISTO+CRAT	+	
65		-SLIP	-		^	PILLOWSLIP	PILLOW+SLIP	+	
66		-STAT	-		^	THERMOSTAT	THERMO	□	
67		-●	-	-	^				●

Table 10.7 Rulebase for 10-Letter-Long-Words Ending In **VCVC.**

Four archtypical schemata graphs are encountered in our analysis of 10-letter-long word endings.

The first and simplest of these which will be referred to as a 'cascade' graph involves simple sequential control flow transfer either throught the evaluation of a string of primary suffix rules or through the evaluation of at most one block of nested secondary rules. Transfer to a block of secondary, or auxillary, rules within a cascade graph is equivalent to a call to a local or nested subroutine with embedded scope. Control flow within a cascade graph may also be transferred to a non-local subroutine. An example of a cascade schema graph is illustrated in Figure 10.13 which is drawn from Table 10.6. This sequential schema graphs can be characterized in terms of four parameters (P, Q, SP, SQ) where P is the number of primary nodes and Q is the number of secondary nodes in the graph depicting the schema, while SP is the number of primary subroutine nodes and SQ is the number of secondary subroutine nodes. For instance the cascade graph drawn from Table 10.6 has P = 8 , Q = 33 and therefore an average of six secondary or auxillary rules per primary suffix. In this example SP = 0 and SQ = 0. Figure 10.14 illustrates a parallel version of the cascade schema graph given in Figure 10.13. The parallel structure given in Figure 10.14 exhibits ($2\alpha + 1\beta$) behaviour. In fact all simple cascade graphs (with SP = 0 and SQ = 0) possess ($2\alpha + 1\beta$) behaviour. Tables 10.2, 10.4, 10.6 and 10.7 are all simple cascade graphs which structurally differ from each other only in terms of their (P, Q, SP, SQ) values and the location and distribution of the Q and SQ nodes.

The second archtype, case II, is illustrated in Table 10.1 and Figures 10.8 (P = 5, Q = 4, SP = 14 , SQ = 0) and Figure 10.9 ($1\alpha + 4\beta$). This case is characterized by simple sequential execution with the optional capacity to shift forward or skip the evaluation of a block of primary rules, such as R4..R8 in Figure 8, or subroutine rules, such as R16..R17 in Figure 10.8. Such time-warped behaviour in case II structures is unidirectional. Case II graphs differ from simpler cascade graphs by their ability to direct a shift forward in control flow. A common block of rules, such as R18..R24, may thus be reached from any number of paths. In this example SP = 14 and SQ = 0.

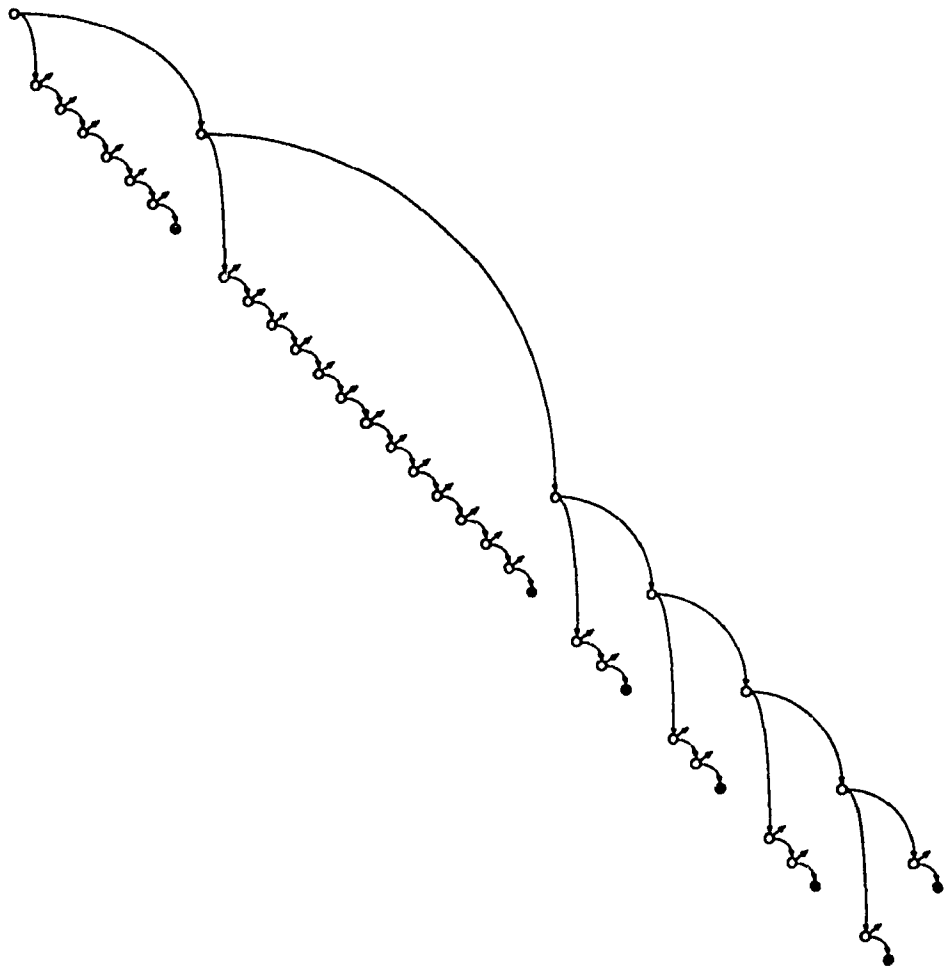

Figure 10.13 Sequential Cascade Schema Graph constructed for the rulebase given in Table 10.6 for 10-letter-long words with suffix strings of the form **-VCVC.** (CASE 1 BEHAVIOUR)

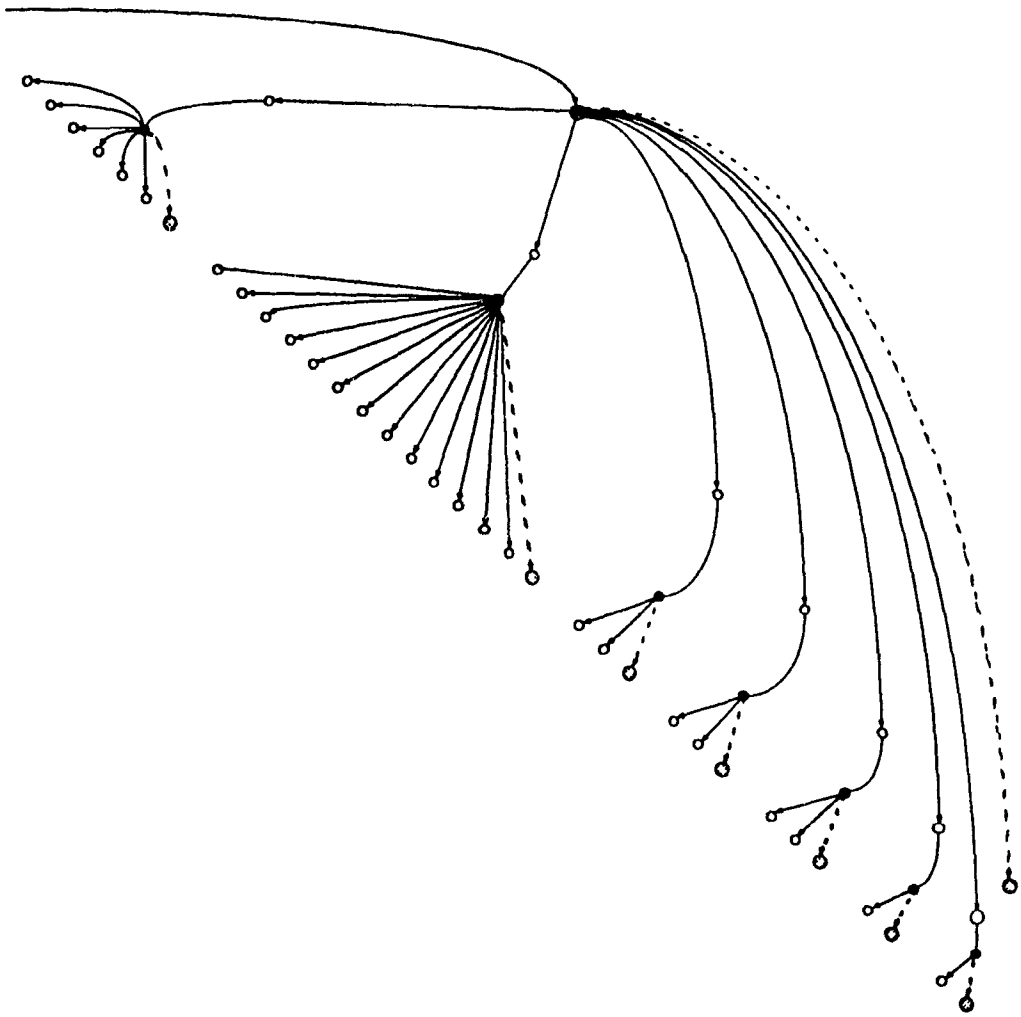

Figure 10.14 Parallel Cascade Scheme Graph constructed for the rulebase given in Table 10.6 for 10-letter-long words with suffix strings of the form **-VCVC.** (CASE 1 BEHAVIOUR)

The third archtype, case III, is characterized by simple sequential execution which may be enhanced by the ability to skip forward from the schema's secondary or auxilary ruleset to a node in the primary ruleset. This type of control flow behaviour is similar in nature to that exhibited by a restricted-exit construct. An example of case III behaviour is depicted in Figure 10.15 (P = 9 , Q = 55, SP = 0, SQ = 0) which is drawn from Table 10.3. In this example control flow transfer from two different secondary nodes (R3 & R21) is directed to a common primary rule, R53. Such control flow behaviour is conceptually similar to allowing multiple exits from a single subroutine. The parallel version of Figure 10.15 exhibits ($3\alpha + 1\beta$) behaviour.

The fourth archtype, case IV, is characterized by simple sequential execution which may be enhanced by the ability to skip forward from anywhere within the schema's primary or secondary rulesets to a node in one of the schema's subroutines or common block rulesets. Whenever a node is used as a common entry point to a block of rules in Case IV schema graphs a further restriction is placed on the schema's control flow. This restriction is that a common entry point may be reached only by control flow transfer from either a set of primary nodes or a set of secondary nodes. Under no circumstances may a common entry point be reached from a mixture of both primary and secondary nodes. An example of case IV behaviour is depicted in Figure 10.16 (P = 9 , Q = 29, SP = 23, SQ = 8) which is drawn from Table 10.5. In this example control flow transfer from three different secondary nodes (R10, R13 & R25) is directed to a common primary subroutine rule, R43. Similarly in this example control flow transfer from two different primary nodes (R26 & R27) is directed to a common primary subroutine rule, R60. This type of restriction is conceptually similar to placing scope restrictions on a program's structural subcomponents. The parallel version of Figure 10.16, which is given in Figure 10.17, exhibits ($4\alpha + 1\beta$) behaviour.

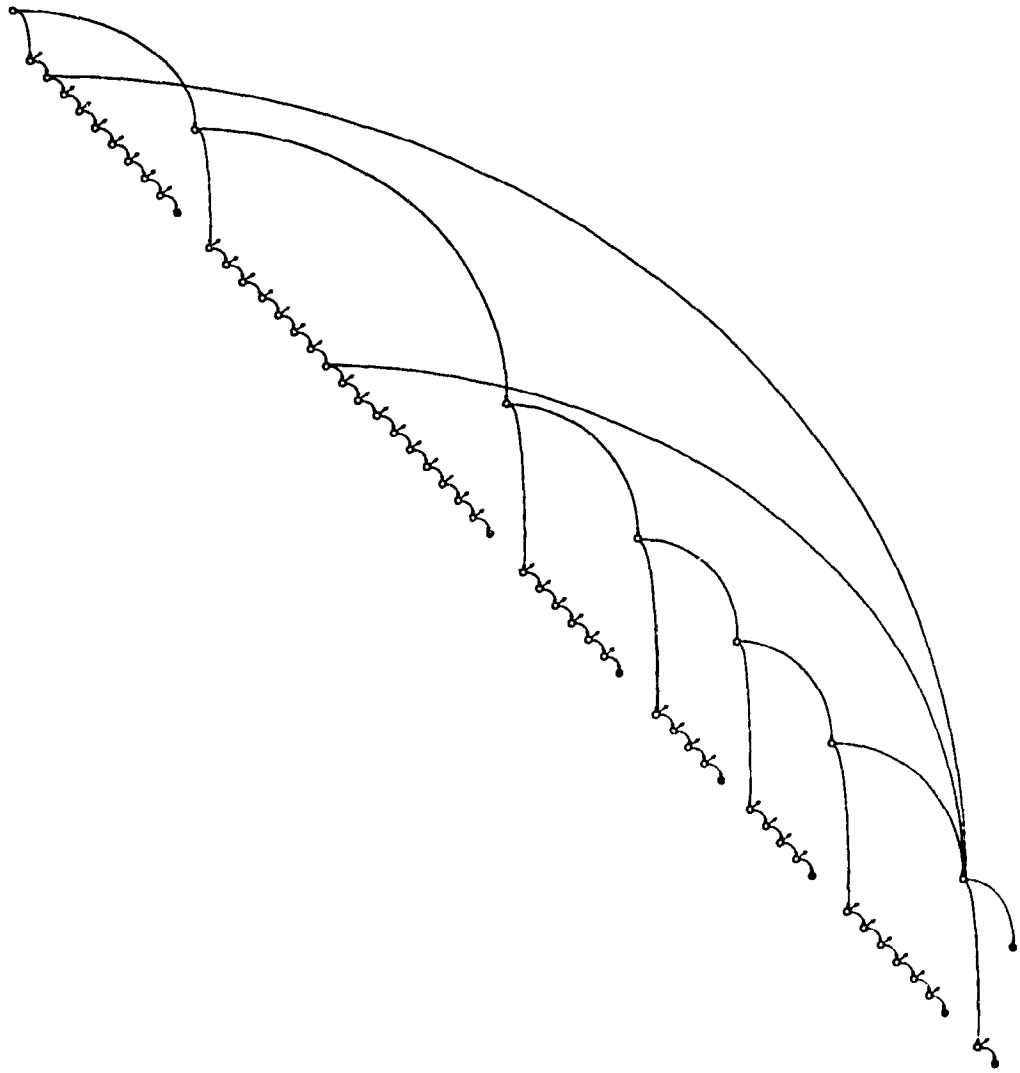

Figure 10.15 Sequential Cascade Schema Graph constructed for the rulebase given in Table 10.3 for 10-letter-long words with suffix strings of the form **-CVCC.** (CASE III BEHAVIOUR)

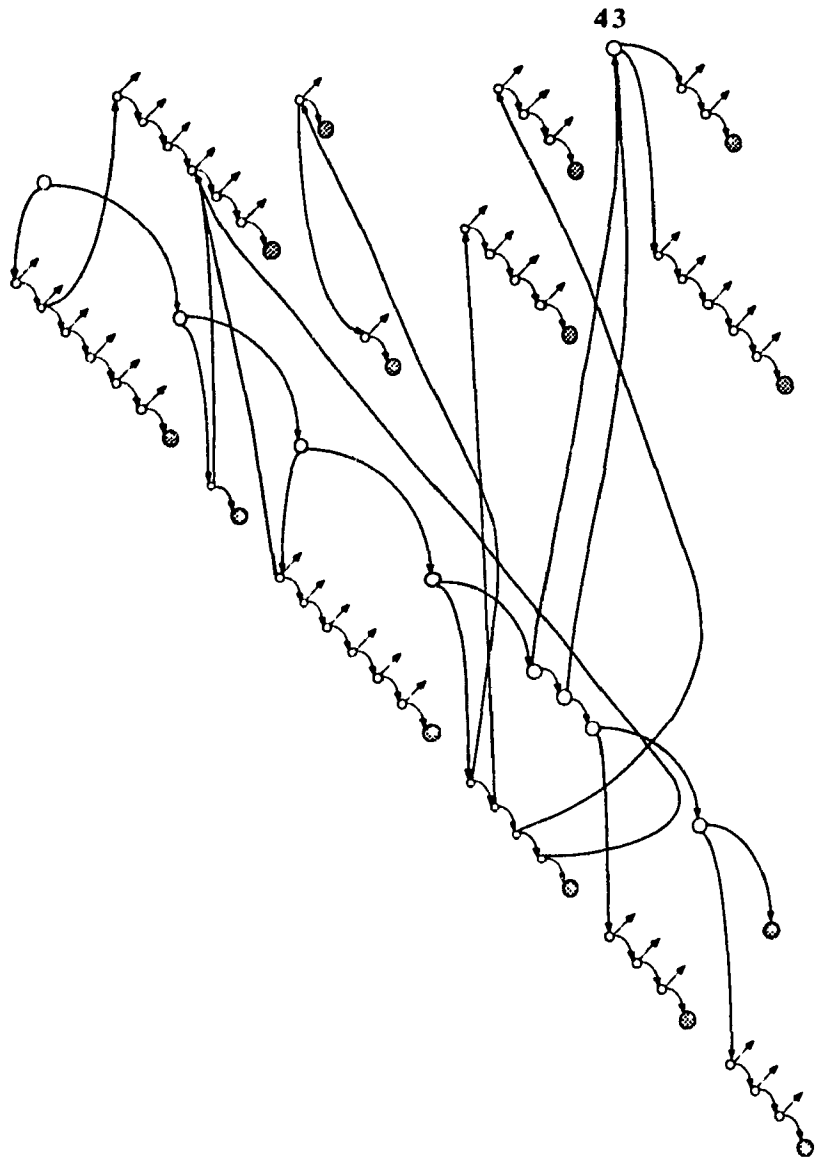

Figure 10.16 Sequential Cascade Schema Graph constructed for the rulebase given in Table 10.5 for 10-letter-long words with suffix strings of the form **-VCCV.** (CASE IV BEHAVIOUR)

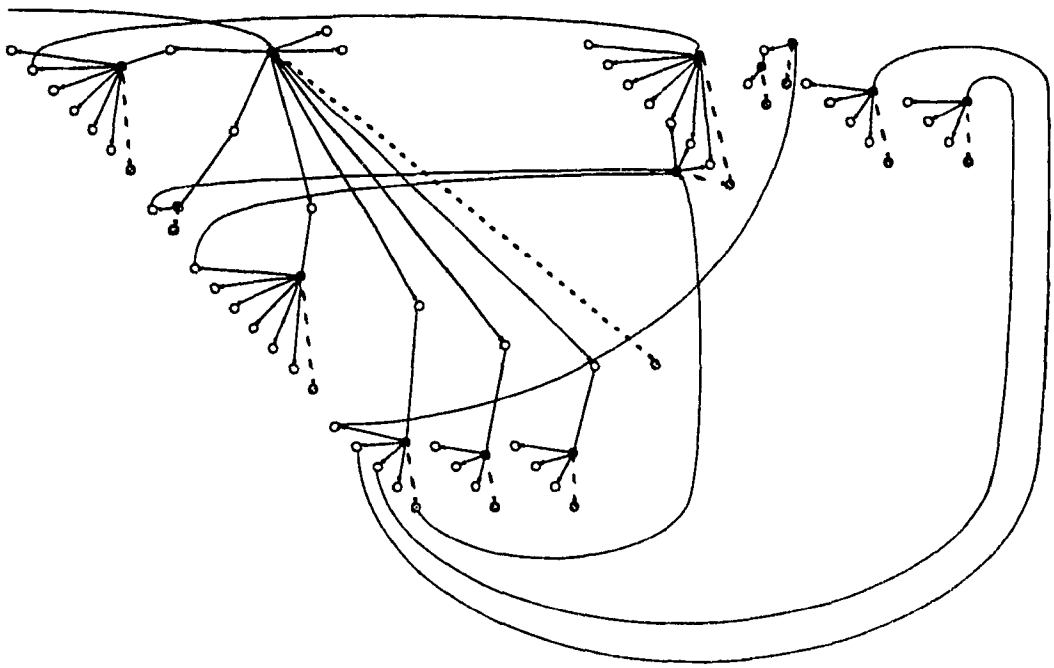

Figure 10.17 Parallel Cascade Scheme Graph constructed for the rulebase given in Table 10.5 for 10-letter-long words with suffix strings of the form **-VCCV.** (CASE IV BEHAVIOUR)

VNF WORD ENDING	WORD	PREFIX	FIRST ROOT	CONJUNCTION	SECOND ROOT	SUFFIX	NOTE
CVVV							
	GONORRHOEA		GONO		RRHOEA		
CCCC							
	FOOTLIGHTS		FOOT		LIGHT	S	
	THOUSANDTH		THOUSAND			TH	
	WAVELENGTH		WAVE		LENGTH		
CCVV							
	CATAFALQUE		CAT	A	FAL	QUE	
	CHIMPANZEE						
	HUMORESQUE		HUMOR			ESQUE	
	POINSETTIA						
	PRESENTDAY		PRESENT		DAY		
	QUARTERDAY		QUARTER		DAY		
	RIFTVALLEY		RIFT		VALLEY		
	SCREENPLAY		SCREEN		PLAY		
	STATUESQUE		STATU(E)			ESQUE	
	STRATHSPEY		STRATH		SPEY		
	UNBIRTHDAY	UN	BIRTH		DAY		
VCVV							
	AFRORMOSIA		AFRO		RMOSIA		
	BUDGERIGAR		BUDGERI		GAR		
	COMMUNIQUE		COMMUNI			QUE	
	CORNUCOPIA		CORNU		COPIA		
	DIPSOMANIA	DIPSO	MANIA				
	DIPHTHERIA	DIPH	THERIA				
	ESCALLONIA						
	HULLABALOO		HULLA		BALOO		
	IMPRESARIO	IM	PRESA		RIO		
	MONTBRETIA		MONT		BRETIA		
	PARAPLEGIA	PARA	PLEGIA				
	PASSAGEWAY		PASSAGE		WAY		
	ROTISSERIE		ROTISS			ERIE	
	SATURNALIA		SATURN			ALIA	
	TRAVELOGUE		TRAVE(L)		LOGUE		
	UNDERVALUE	UNDER	VALUE				
	XENOPHOBIA	XENO	PHOBIA				
VCCC							
	AFTERBIRTH	AFTER	BIRTH				
	AFTERWARDS	AFTER	WARDS				
	BELONGINGS		BE		LONG	INGS	
	BIRTHRIGHT		BIRTH		RIGHT		
	COELACANTH		COELAC		ANTH		
	CARTWRIGHT		CART		WRIGHT		
	CHILDBIRTH		CHILD		BIRTH		
	CLOUDBURST		CLOUD		BURST		
	DISTRAUGHT	DIS	TRAUGHT				
	EIGHTEENTH		EIGH(T)		TEEN	TH	
	FISTICUFFS		FIST	I	CUFF	S	
	FLASHLIGHT		FLASH		LIGHT		
	FORTHRIGHT		FORTH		RIGHT		
	FOURTEENTH		FOUR		TEEN	TH	
	FLOODLIGHT		FLOOD		LIGHT		
	GREENFINCH		GREEN		FINCH		
	HENCEFORTH		HENCE		FORTH		
	HOTCHPOTCH		HOTCH		POTCH		
	INDISTINCT	IN	DISTINCT				
	LINEAMENTS		LINE	A	MENTS		
	MAKEWEIGHT		MAKE		WEIGHT		
	NINETEENTH		NINE		TEEN	TH	
	NORTHWARDS		NORTH		WARD	S	
	OVERWEIGHT	OVER	WEIGHT				
	PENNYWORTH		PENNY		WORTH		
	PLAYWRIGHT		PLAY		WRIGHT		
	SACROSANCT	SACRO	SANCT				
	SHIPWRIGHT		SHIP		WRIGHT		
	SIDEBOARDS		SIDE		BOARD	S	
	SOUTHWARDS		SOUTH		WARD	S	
	SHOREWARDS		SHORE		WARD	S	
	THIRTEENTH		THIR		TEEN	TH	
	TORCHLIGHT		TORCH		LIGHT		
	UNDERWORLD	UNDER	WORLD				
	UNDERPANTS	UNDER	PANT			S	
	WATERWINGS		WATER		WING	S	
	WATERWORKS		WATER		WORK	S	
	WATERTIGHT		WATER		TIGHT		

HAND ANALYSIS OF ALL WORDS FOUND IN RARELY USED 10-LETTER LONG WORDS ENDING IN CVVV, CCCC, CCVV, VCVV and VCCC.

Table 10.8 List of exceptions found in the OPD for 10-letter-long words ending in **CVVV, CCCC, CCVV, VCVV, VCCC.**

The remaining classes { **CVVV**, **CCCC**, **CCVV**, **VCVV**, **VCCC** } as seen in Table 10.8 are small, rarely used, sets of words which are mostly compound words. Such sets and their elements are best treated as exceptional cases which can be best handled by a table look-up procedure. The most densely populated suffix groups such as **CVCV**, **CVCC** and **CVVC** are however both simply and efficiently handled by our rulebase system.

In an analysis of the algorithm's performance, it is important to remember that compiled statistics such as those given in Figure 10.18, for the position-dependent, letter-frequencies found in 10-letter-long-words, allow us to initialize the order of evaluation of a set of given sequential rules in a manner that optimizes the efficiency of sequential search. For instance, by using the data compiled in Figure 9.16, for the most frequently encountered 10-letter-long-words, we would rank order the primary suffixes { **-CY**, **-RY**, **-LY** } found in Table 10.1 as { **-LY**, **-CY**, **-RY** }[10]. In that both the 'depth of recursion' and the 'branch-factors' found in rulebases needed to span the OPD is modest, the applications of dynamic optimization techniques to the maintenance of the rulebase is unwarranted. The process of rank ordering the rulebase on the bases of static statistics which were compiled for a representative sample set is more than adequate. Procedures such as these are referred to as robust when they yield sufficing solutions for a wide range of sample sets.

10.8 IMPLEMENTATION

The schemata presented in this chapter have been easily implemented on a number of systems in various high level languages, such as Pascal and Prolog. On-going research will investigate them

---

10 In this case for example the rank of the letter C, $\rho(C)$, is 6 while the rank of the letter R, $\rho(R) = 10$ and $\rho(l) = 5$ when occuring at position nine of 10-letter-long-words and thus $\rho(\text{-LY}) > \rho(\text{-CY}) > \rho(\text{-RY})$.

further within a language independent programming environment called ABL/W4 [10.43].

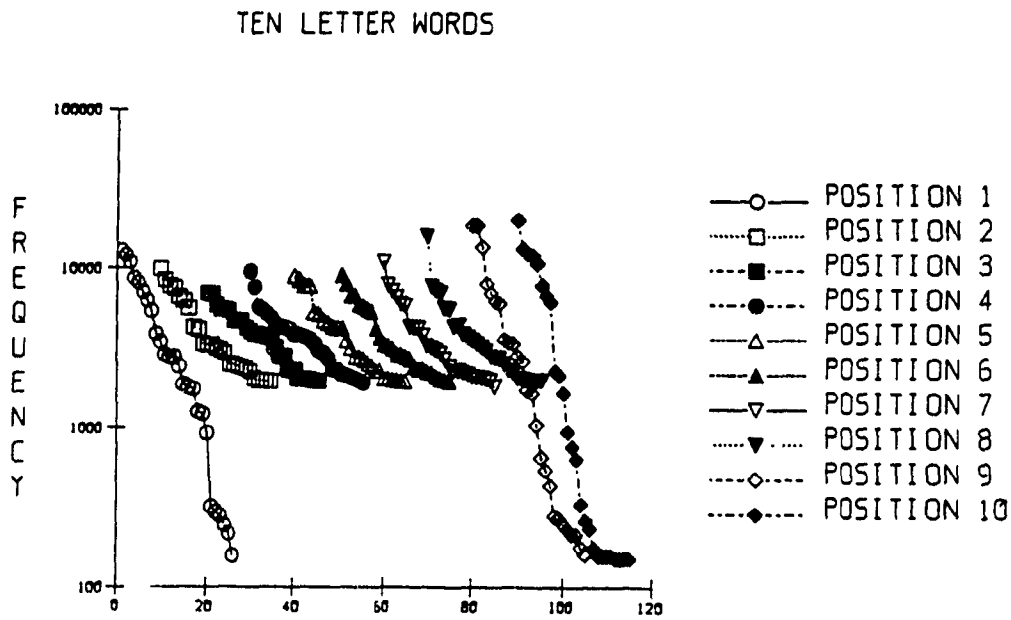

Figure 10.18 Position-dependent, letter-frequencies found in a sample of 10-letter-long words [10.24]. The following list gives the rank ordered letter sequences for each possible position in 10-letter-long words.

Position 1= {C, P, I, A, S, E, D, R, M, T, O, F, U G, H, L, B, N, W, V, Q, Y, U, Z, K,X}

Position 2= {O, E, N, S, A, R, I, U, H, C, X, P, M, T, L, V, D, G, B, Y, F, Q, W, K, J, Z}

Position 3= {N, P, S, R, T, E, D, C, I, M, O, A, J, L, F, V, G,.U, B, Y, H, W, Q, K, X, Z}

Position 4= {E, I, R, T, O, C, S, A, L, U, P, N, M, D, F, V, H, G, K, Q, B, J, W, Y, Z, X}

Position 5={E, R, I, S, O, T, A, U, N, L, C, V, G, P, M, H, D, B, F, Y, W, K, X, Z, J, Q}

Position 6= {I, T, A, S, E, R, N, C, X, O, D, U, P, M, L, G, H, B, F, V, W, Q, Y, K, Z, J}

Position 7= {T, I, A, M, S, E, U, R, L, O, D, N, C, H, P, V, G, Y, B, F, Q, X, W, Z, K, J}

Position 8= {I, E, B, N, O, T, Z, L, A, D, R, C, S, U, V, P, H, F, G, M, W, K, Q, Y, X, J}

Position 9= {E, N, O, A, L, C, T, J, I, R, S, U, H, V, G, D, M, Q, P, W, Z, F, B, K, Y, X}

Position 10= {S, N, E, Y, D, T, L, G, R, C, M, H, A, P, K, I, O, U, F, W, X, Z, B, J, Q, V}

10.9 VERIFICATION AND VALIDATION

The process of validation and verification of knowledge-bases or expert systems is an important and difficult process [10.45, 10.46]. In our model we need to assess those cases in which the system derived a reduced form which is in disagreement with that of a human expert.

Fortunately, the task of compiling a comprehensive list of primary and secondary hyphenation points for most of the words found in the OPD was undertaken on a case-by-case basis by a group of linguists and etymologists at Oxford in 1986 [10.20]. The approach taken by these scholars was to render justice, including considerations of historical precedence, to each and every word in the lexicon. In this manner, these experts avoided the problem of attempting to establish a set of non-contradictory syntax rules for word morphology. The existence of the OSD [10.20] is fortuitous in that it provides a 'benchmark' on which to base the algorithm's performance at word-hyphenation.

The OSD may also be used as the basis on which to verify a derivation's root or stem. This latter process is however more complex and requires value judgements, which, while complex, are nonetheless apparently obvious to the native speaker.

When using the OSD [10.20] as a benchmark, one finds that the rulebases given in this chapter reduce most words to their correct root. The rulebases reduce words to a wrong root or stem in very few cases. The system performs accurately in its analysis of the vast majority of all 10-letter-long-words found in the OPD.

10.10 DISCUSSION

In classical expert systems it is often the choice of the domain features which are used to access and establish a rulebase, that determines the system's overall utility and efficiency [10.45, 10.47]. Even if one modifies an expert system's rulebase to take into account any natural hierarchy, that exists within its rulebase, one is confronted

with essentially the same 'feature selection problem' that has plagued pattern recognition research for decades [10.48, 10.49].

Clearly the intuitiveness, expressiveness, simplicity and applicability of **IF...THEN** production rules to natural language word syntax is an excellent example of the power of this approach and the subtle difficulties that can occur in such systems. While a set of **IF...THEN** production rules would appear, in principle, to be eminently applicable to natural language word morphology, the contextual dependencies of such rules make it a very demanding task to determine features which can be accurately used to specify which rules apply to an arbitrary word. The brute force approach would simply tabulate a case-by-case analysis of each word in the entire lexicon. It is this latter approach that led to the publication of the OSD [10.20]. While the OSD is a very valuable reference source it is a simple databank which does not provide us with any understanding of the processes which underlie English language word morphology. The brute force approach is best reserved for use in situations where it is impossible to produce a set of consistent, context-sensitive, syntax rules.

In spite of the merits of the philosophical objections to reductionistic efforts such as the derivation of natural language structure, the immense pedagogical and technical practicalities of such rulebases would surely guarantee the acceptance of a schema of comprehensive syntax rules for English.

The simplicity of the schemata described in this chapter is the result of the mapping of a Chomsky type 1, context-sensitive, grammar to a set of readable **IF...THEN** rules which provide an easily understood representation of English Language syntax that is concise enough to be easily learned and quickly verified or validated. The use of context sensitive blocks of rules is essential to this task in that this approach limits the scope and enhances the understandability of the process.

The inclusion of explicit control flow, and the use, of a hierarchy of rule-sets, empowers the system with a clear and simple semantic notation for generic schemata with the computational facilities needed to clearly manage the symbolic complexity of English. These system endowed features are important in that it is not reasonable to expect a

simple orthogonality in the actual rulebase. English language grammar rules are not like sets of independent, mutually, exclusive, non-interactive **IF...THEN** production rules encountered in many successful expert systems. The success of many expert systems is attributable in part to their application in a constrained and modeled environment where the logical equivalent of the superposition principle holds [10.34].

The system described in this chapter is not constrained by such principles. Of course, the inclusion of control flow in the rule-base reduces the modularity of the rule-base and also increases the possibility of generating side-effects when the rule-base is updated. This hopefully minor deficit in design is balanced by a clearer understanding of the impact of the rule-interpreter's control flow on the rule-base in its evaluation of specific cases. This approach is particularly applicable to sets of rulebases each of which contain very few rules. The opacity encountered by the interaction of control-flow on rule-based systems is less of a problem in our formulation since the rule-base itself is process oriented. The simplicity of our inference system is the result of a basic model [10.13] of the morphological processes encountered in English language word structures and their derivations.

The inference model used in this work is very simple. The model may be operationally described in terms of two processes. First, it applies the rules causing suffix removal and then it checks for 'mutant-strings' which would have resulted from the application of generic rules to an exceptional word. On the rare occasions when a mutated string is detected, the application of a rule that led to the mutation is reversed and the word is noted to be an exception to the system's rule-set. In such exceptional cases further processing of the word is terminated. If the system encounters no exceptions processing continues by checking for any further possible suffix reductions. The system described here restricts the possibility of 'state-space-explosions' by restricting the 'outdegree' of its schema graph to 2.

Many expert systems have established [10.50, 10.45] rule priority schemes in an attempt to avoid conflict resolution. Conflict

resolution [10.34, 10.51] is less of a problem in our system in that it is assumed that a given word has a unique root or stem and thus our schema does not incorporate a set of sufficing solutions. The use of priority in our system is thus primarily restricted to the sequencing of rules which guarantees that the longest possible suffix strings consistent with the data is checked first.

10.10 CONCLUSIONS

The method and results presented in this chapter demonstrate that a system based on the application of a set of simple context-sensitive rule bases is sufficient to efficiently and accurately derive the word-root or base of larger English language words.

10.11 REFERENCES

[10.1]. see 1.34

[10.2]. see 1.35

[10.3]. see 3.9

[10.4]. S Srihari, ed., "Computer Text Recognition Correction," IEEE Computer Society Press, Piscataway, New Jersey, 1985.

[10.5]. see 1.62

[10.6]. see 6.6

[10.7]. G. Barton, R. Berwick, E. Ristad, Computational Complexity and Natural Language, MIT Press, Cambridge, Massachusetts, 1987.

[10.8]. G. Blank, "A Finite and Real-Time Processor for Natural Language," Commun. ACM 32, Vol. 10, pp. 1174-1189; Oct. 1989.

[10.9]. G. Hirst, Semantic Interpretation and the Resolution of Ambiguity, Cambridge University Press, London, UK, 1988.

[10.10]. R. F. Simmons, "Semantic Networks: Their Computation and Use for Understanding English Sentences," Computer Models of Thought and Language, R. Schank and K. Colby, eds., Freeman, San Francisco, CA., 1973.

[10.11]. G. Pullman, "Syntactic and Semantic Parsability," In Proceedings of COLING 84, Stanford University, pp. 34-40; July 1984.

[10.12]. see 1.49

[10.13]. see 1.50

[10 14]. K. S. O'Mara, T. Fancott, S. Hyder, "An Artificial Intelligence Procedure for Extracting Natural Language Syntax Rules at the Lexical Level", manuscript in preparation

[10.15]. D. Armon, K. S. O'Mara, "Semantic Meaning & the Extraction of Word-Roots from English Text by a Context Sensitive Refinement of Paice's Algorithm," J. MN. Academy of Science, Vol. 53, 3, 1988.

[10.16]. see 1.54

[10.17]. see 1.63

[10.18]. D. E. Appelt, Planning English Sentences, Cambridge University Press, NY., 1985.

[10.19]. see 3.2

[10.20]. see 3.12

[10.21]. see 1.42

[10.22]. see 8.22

[10.23]. see 3.14

[10.24]. see 3.11

[10.25]. see 2.19

[10.26]. R. Zhu, T. Takaoka, "A Technique for Two-Dimensional Pattern Matching," Commun. ACM 32, Vol. 9, 1110-1120; September, 1989.

[10.27]. J. L. Peterson, Computer Programs for Detecting and Correcting Spelling Errors, Commun. ACM 23, Vol. 12, pp. 676-687; December, 1980.

[10.28]. J. L. Peterson, A Note on Undetected Typing Errors, Commun. ACM 29, Vol. 7, pp. 633-637; July 1986.

[10.29]. J. Bentley, Programming Pearls: A Spelling Checker, Commun. ACM 28, 4, pp. 456-462; May 1985.

[10.30]. J. Yen, "Gertis: A Dempster-Shafer Approach to Diagnosing Hierarchical Hypotheses," Commun. ACM 32, Vol. 5, pp. 573-585; May 1989.

[10.31]. R. Davis, D, Lenat, eds., Knowledge-Based Systems in Artificial Intelligence, McGraw-Hill, New York, N. Y., 1981.

[10.32]. S. Tu, M. Kahn, M. Musen, J. Ferguson, E. Shortliffe, L. Fagan, "Episodic Skeletal-Plan Refinement Based on Temporal Data," Commun. ACM 32, Vol. 12, pp. 1439-1455; December 1989.

[10.33]. see 2.76

[10.34]. see 1.4

[10.35]. H. Ledgard, M. Marcotty, The Programming Language Landscape, SRA, Chicago, pp. 394-415; 1981.

[10.36]. see 3.17

[10.37]. M. R. Paige, "Program Graphs, an Algebra, and their Implications for Programming," IEEE Transactions on Software Engineering, Vol SE-1, 3, pp. 286-291; September 1975.

[10.38]. M. R. Paige, "On Partitioning Program Graphs," IEEE Transactions on Software Engineering, Vol SE-3, 6, pp. 386-393; November 1977.

[10.39]. T. Fancott, W. M. Jaworski, K. S. O'Mara, On the Canonical Representation of Control Flow, Proceedings IEEE Montech-Compint 1987, # 87CH2518-9, pp. 347-354; November 1987.

[10.40]. D. W. Jones, "An Empirical Comparison of Priority-Queue and Event-Set Implementations," Commun. ACM. Vol. 28, 4, pp. 300-311; April 1989.

[10.41]. W. J. Hendricks, "An Account of Self-Organizing Systems," SIAM J. Comput., Vol. 5, 4, pp. 715-723; December 1976.

[10.42]. J. L. Bentley, C. C. McGeoch, "Amortized Analyses of Self-Organizing Sequential Search Heuristics," Commun. ACM Vol. 28, 4, pp. 404-411; April 1989.

[10.43]. see 1.61

[10.44]. see 1.38

[10.45]. F. Hayes-Roth, "Rule-Based Systems," Commun. ACM, Vol. 28, 9, pp. 921-932; September 1985.

[10.46]. D. G. Bobrow, S. Mittal, M. Stefik, "Expert Systems: Perils and Promise," Commun. ACM, Vol. 29, 9, pp. 880-894; September 1986.

[10.47]. see 1.53

[10.48]. R. Reddy, L. Erman, R. Fennell, R. Neely, "The HEARSAY Speech Understanding System: An Example of the Recognition Process," IEEE Trans. Computers C-25, pp. 427-431; 1976.

[10.49]. S. Pinker, Visual Cognition, MIT Press, Cambridge, Massachusetts, 1985. Expert system quote (domain features)

[10.50]. J. McDermott, A. Newell, J. Moore, "The Efficiency of Certain Production System Implementations," D. Waterman, F. Hayes-Roth, eds., Pattern-Directed Inference Systems, pp. 177-199, Academic, London, 1978.

[10.51]. J. McDermott, C. Forgy, "Production System Conflict Resolution Strategies," D. Waterman, F. Hayes-Roth, eds., Pattern-Directed Inference Systems, pp. 177-199 Academic, London, 1978.

CHAPTER ELEVEN
CONCLUSIONS

11.1 RESULTS

The basic fundamental results obtained from this work are:

1) There are fundamental patterns underlying English language word structure at the orthographic level.

2) A classification and clustering scheme referred to as Vowel Normal Form (VNF) is a powerful tool for classifying English word structure. Its great advantage is simplicity, but this quality limits classification to a single letter level and can not accommodate multiletter combinations such as qu or ch.

3) A simple prefix code underlies the relationship between the major word structures of various sizes found throughout the English language lexicon listed in the Oxford Paperback Dictionary.

4) The prefix code structure of English language word structure assures band-filtering effects which may be exploited by simple pattern recognition routines.

5) A single two parameter model is sufficient to predict the size of the major VNF word group structures found in the Oxford Paperback Dictionary.

6) The prefix code structure model, when coupled with the two parameter set-size model predicts both the structure and size of the major VNF frames found in the lexicon.

7) A form of directed graph, referred to as a WORD-WEB, is sufficient to represent all words of a given VNF set.

8) Context-sensitive rule base schema are sufficient to reduce longer words, such as 10-letter-long words to their root words or base component.

9) The frequency of occurrence of words may be computed as the product of word-length and position-dependent letter frequencies for the most frequently occurring smaller words listed in the Oxford Paperback Dictionary.

11.2 FURTHER RESEARCH

Further work is needed to determine if:

1) These results are expandable to larger English language lexicons such as the Oxford English Dictionary.

2) These models and results apply to languages which are closely related to English such as French, Spanish and German.

3) These results indicate support of Chomsky's theory that the human language center is genetically endowed.

4) These results can be directly applied to other forms of animal communication such as those encountered in dolphins and the higher apes.

5) The Prefix Model presented in this work is both necessary and sufficient for predicting English language word structure.

6) The VNF Set Size Model presented in this work is both necessary and sufficient for predicting English language word structure.

7) The frequency of occurrence of words may be computed as the product of word-length and position-dependent letter frequencies for the most frequently occurring 5- to 10-letter-long-words listed in the Oxford Paperback Dictionary. This frequency could be expected to be influenced to a certain extent by domain-specific characteristics, for example technical terms in legal or medical texts.

Furthermore the following 'what' questions need to be answered:

1) What type of statistical distribution is concomitant with the observed rank-ordered set-size function found to underlie English language VNF word structures.

2) What is needed to extend these models and their results to languages, such as Turkish, where pronunciation and phonetics are mapped as one-to-one functions unto a word's written form.

3) What is needed to extend these models and their results to iconographic languages, such as Chinese, or to languages which do not use explicit vowel representation, such as Arabic or Hebrew.

4) To what degree can these results predict the origin and evolution of natural languages such as English.

5) To what degree can these results be used to predict the primitive archetypes underlying modern day languages in general and English in particular.

6) What characteristics make the English language VNF group **CVCCVC** make it an aberrant outlyer.

7) To what degree do the two models developed in this thesis establish the basis components of a physical symbol system needed to model and predict English language word structures and their usage.

11.3 BASIC ASSUMPTIONS

This work has been undertaken under the following three basic assumptions:

1) While the spoken word is fundamental to our understanding of English, the written word is sufficient for an analytical analysis of English language word structure.

2) The Physical Symbol System Hypothesis first presented by Newell and Simon is necessary and sufficient for generalized intelligent action.

3) The English lexicon can be described as a relatively simple physical symbol system.