

AD-A157 665

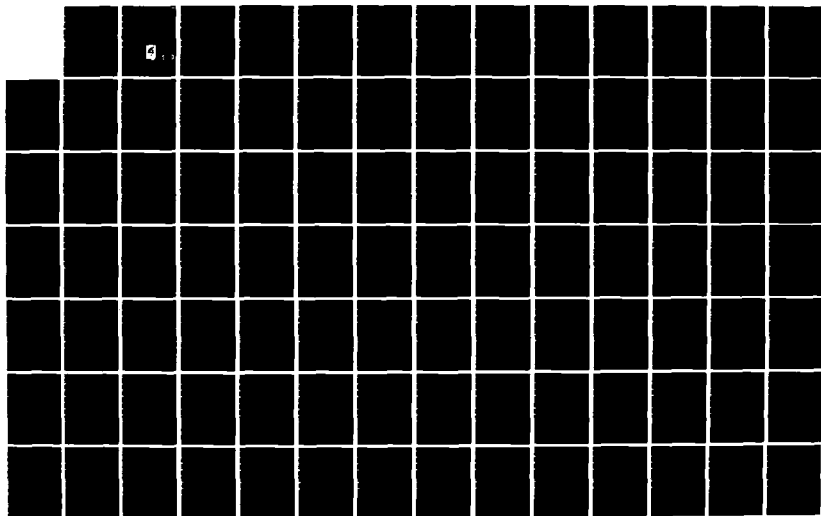
HUMAN COGNITION AND PERFORMANCE(U) CALIFORNIA UNIV SAN
DIEGO LA JOLLA INST FOR COGNITIVE SCIENCE
D A NORMAN ET AL. 01 MAY 85 N00014-79-C-0323

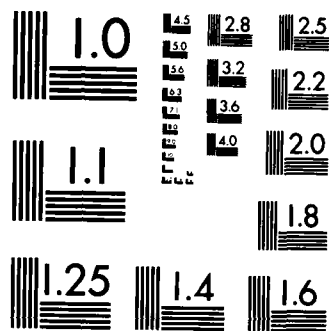
1/2

UNCLASSIFIED

F/G 5/10

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD-A157 665

**HUMAN COGNITION
AND PERFORMANCE**

**ONR Contract N-00014-79-C-0323
NR 667-437**

April 1979 - March 1985

Final Report

**COGNITIVE
SCIENCE**



DTIC FILE COPY

**DTIC
ELECTE**
AUG 07 1985
S D
E

INSTITUTE FOR COGNITIVE SCIENCE

UNIVERSITY OF CALIFORNIA, SAN DIEGO

LA JOLLA, CALIFORNIA 92093

This document has been approved
for public release and sale; its
distribution is unlimited.

85 7 25 006

**HUMAN COGNITION
AND PERFORMANCE**

**ONR Contract N-00014-79-C-0323
NR 667-437**

April 1979 - March 1985

Final Report

**Donald A. Norman and David E. Rumelhart
Principal Investigators**

**Donald R. Gentner
Skills Project Director**



This document has been approved
for public release and sale; its
distribution is unlimited.

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified			1b. RESTRICTIVE MARKINGS		
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution unlimited.		
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE					
4. PERFORMING ORGANIZATION REPORT NUMBER(S)			5. MONITORING ORGANIZATION REPORT NUMBER(S)		
6a. NAME OF PERFORMING ORGANIZATION Institute for Cognitive Science		6b. OFFICE SYMBOL (if applicable)	7a. NAME OF MONITORING ORGANIZATION		
6c. ADDRESS (City, State, and ZIP Code) C-015 University of California, San Diego La Jolla, CA 92093			7b. ADDRESS (City, State, and ZIP Code)		
8a. NAME OF FUNDING / SPONSORING ORGANIZATION Personnel & Training Research		8b. OFFICE SYMBOL (if applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N-00014-79-C-0323		
8c. ADDRESS (City, State, and ZIP Code) Code 442PT Office of Naval Research 800 North Quincy Street, Arlington, VA 22217			10. SOURCE OF FUNDING NUMBERS		
		PROGRAM ELEMENT NO.	PROJECT NO. NR 667-437	TASK NO.	WORK UNIT ACCESSION NO.
11. TITLE (Include Security Classification) Human Cognition and Performance: Final Report					
12. PERSONAL AUTHOR(S) Donald A. Norman, David E. Rumelhart, and Donald R. Gentner					
13a. TYPE OF REPORT Final Report		13b. TIME COVERED FROM 4/79 TO 3/85		14. DATE OF REPORT (Year, Month, Day) 1985 May 1	
15. PAGE COUNT 95					
16. SUPPLEMENTARY NOTATION					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB-GROUP	parallel distributed processing; human-machine interaction; motor skills; learning; interface design; typewriting; performance		
19. ABSTRACT (Continue on reverse if necessary and identify by block number)					
This report reviews work performed in the Cognitive Science Laboratory during the period from April 1, 1979 to March 31, 1985. The work has been focused on three main areas: parallel distributed processing, human-machine interaction, and skilled human performance. Summaries of these projects are presented.					
20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION		
22a. NAME OF RESPONSIBLE INDIVIDUAL			22b. TELEPHONE (Include Area Code)		22c. OFFICE SYMBOL

TABLE OF CONTENTS

INTRODUCTION	1
Overview of the Cognitive Science Laboratory	1
The Organization of this Report	2
PARALLEL DISTRIBUTED PROCESSING	3
A GENERAL FRAMEWORK FOR PARALLEL DISTRIBUTED PROCESSING	6
<i>The General Framework</i>	<i>6</i>
<i>A set of processing units.</i>	<i>8</i>
<i>The state of activation.</i>	<i>8</i>
<i>Output of the units.</i>	<i>9</i>
<i>The pattern of connectivity.</i>	<i>9</i>
<i>The rule of propagation.</i>	<i>11</i>
<i>Activation rule.</i>	<i>11</i>
<i>Modifying patterns of connectivity as a function of experience.</i>	<i>11</i>
<i>Representation of the environment.</i>	<i>13</i>
Classes of PDP Models	13
<i>Paradigms of Learning</i>	<i>13</i>
<i>Hierarchical Organizations of PDP Networks</i>	<i>16</i>
<i>Bottom-Up Processing</i>	<i>16</i>
<i>Top-Down Processing</i>	<i>17</i>
<i>Interactive Models</i>	<i>17</i>
<i>Synchronous versus Asynchronous Update</i>	<i>18</i>
Specific Versions of the General Parallel Activation Model	19
<i>Simple Linear Models</i>	<i>19</i>
<i>Linear Threshold Units</i>	<i>20</i>
<i>Brain State in a Box</i>	<i>22</i>
<i>Boltzmann Machines</i>	<i>23</i>
<i>Grossberg</i>	<i>27</i>
<i>Interactive Activation Model</i>	<i>27</i>
<i>Feldman and Ballard</i>	<i>28</i>
Sigma-Pi Units	29
References	31
HUMAN-MACHINE INTERACTION	32
USER CENTERED SYSTEM DESIGN	32
Introduction	34
A Pluralistic Field, a Pluralistic Book	34
<i>A Book of Questions, Not of Answers</i>	<i>34</i>
<i>Some Possible Dimensions</i>	<i>34</i>
The Structure of the Book	36
<i>Postscript</i>	<i>37</i>
Section I: User Centered System Design	38
Section II: The Interface Experience	39

Section III: Users' Understandings	40
<i>Bootstrapping</i>	42
Section IV: User Activities	42
<i>Technical Issues</i>	44
Section V: Toward a Pragmatics of Human-Machine Communication	45
Section VI: Information Flow	46
<i>Help Systems as an Extension of Memory</i>	48
<i>The Chapters</i>	49
Section VII: The Context of Computing	51
References	53
SKILLED MOTOR PERFORMANCE AT VARIABLE RATES	55
The Generalized Motor Program	56
Test for a Multiplicative Rate Parameter	56
<i>A Simple Mathematical Model</i>	57
<i>Problems With Existing Tests of Multiplicative Rate Parameter</i>	57
<i>The Constant Proportion Test</i>	57
<i>Test Results With Simulated Data</i>	58
Literature Review	59
Laboratory Tasks	59
<i>Arm movements.</i>	59
<i>Wrist twist.</i>	62
<i>Key press.</i>	62
Locomotion	63
<i>Cockroach.</i>	63
<i>Lobster.</i>	65
<i>Tortoise.</i>	65
<i>Cat.</i>	65
<i>Dog.</i>	65
<i>Human.</i>	66
Other Highly Practiced Skills	68
<i>Breathing.</i>	68
<i>Speech.</i>	68
<i>Handwriting.</i>	69
<i>Typing.</i>	70
<i>Music.</i>	72
Typewriting Data	72
General Method	72
Analyses	73
<i>Correlational analysis.</i>	73
<i>The constant proportion test.</i>	74
Experiments 1 and 2: Prose	74
<i>Method.</i>	74
<i>Results.</i>	74
Experiment 3: Random Words	76
<i>Method.</i>	76

<i>Results.</i>	76
<i>Experiment 4: Slow, Normal, and Fast Typing</i>	76
<i>Method.</i>	76
<i>Results.</i>	77
<i>Patterns of Change With Typing Rate</i>	77
<i>Simulation Model of Typing</i>	79
Multilevel Control of Timing	82
References	84
PUBLICATIONS	88
Technical Reports	88
Archival Publications	91

Accession For	
NTIS GRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	



INTRODUCTION

This report reviews work performed in the *Cognitive Science Laboratory* during the period April 1, 1979-March 31, 1985. The amount of work performed has been considerable and a single overview would be difficult. Instead, we present reports on the main foci of our efforts. Some work is left out, not because it was not important and significant, but because it represents more isolated pieces of research.

Overview of the Cognitive Science Laboratory

The Cognitive Science Laboratory is a part of both the *Institute for Cognitive Science* and the *Center for Human Information Processing* at the University of California, San Diego. Work in the laboratory is aimed at developing a deeper understanding of the theory and applications of human information processing. The research aims of the laboratory are broad, but all the work follows a common theoretical view of cognition, providing a cohesion that binds the separate studies together, despite the apparent wide differences in content. All of the work can be characterized as being *Studies of Human Cognition*. This is the main focus of the laboratory's efforts, and over the years the work has covered in-depth problems within the areas of perception, attention, learning, memory, language, thought, and action. Special emphasis has been placed on knowledge representation, and one major contribution of the laboratory has been within this area: the representational structure known as *Active Structural Networks*. Work in this general area is always in process, elaborated and modulated by the results and progress of work in the specialty areas. Three specialty areas can be talked about separately: parallel distributed processing, skilled human performance, and human-machine interaction:

- *Studies of Parallel Distributed Processing Mechanisms of Computation.* This work examines a new conceptualization of processing and representational structures. Under this view, computation takes place through the interaction of large numbers of elementary processing structures that activate and inhibit one another. The outcome of any computation is the result of competition and cooperation among a large number of simple processes. The representation of any specific knowledge is distributed across many structures. Representations are *additive* in the sense that different knowledge can be represented in the same set of processing units, new knowledge being added to the activation values of the old. This work promises to have major influences on a number of different areas, including basic models of the human information processing system, issues of representation, and the development of novel machine architectures for computational processes in general. The mechanisms can be described as *neurally-inspired*, for although they do not directly model brain structures, their properties are consistent with the neurological evidence. Preliminary models have already been applied to our research on typing, reading, information retrieval, and attention.

- *Studies of Skilled Human Performance.* These studies involve the theoretical and experimental investigation of skilled human behavior, including the changes that take place as a person acquires a skill. In recent years the effort has focused upon detailed investigation of novice and expert typing, but now includes the study of timing, general issues in the learning and practice of skilled performance, and the role of stress in performance.
- *Studies of Human-Machine Interaction.* This area has as its goal the determination of the cognitive principles that underlie human-machine interaction. This work, therefore, merges traditional work on Human Factors with Cognitive Psychology, the difference being the emphasis on cognitive and mental processing. As a result, much of the work concentrates upon the study of mental representations (*mental models*), memory structures (and limitations), principles of social interaction, and skilled human performance. Some of the work is based upon our theoretical descriptions of the mechanisms that give rise to human error and our models of human information processing structures; in addition, there is special emphasis on the work on parallel distributed systems, the roles of different knowledge structures, and the competing demands upon the person. The work in this group interacts heavily with our work in other areas, both by applying the insights gained from the theories and experimental findings, as well as adding to the other areas, for in the process of developing a theoretical structure for interaction with machines, it soon becomes obvious that there are gaps in the general understanding of cognitive functioning, gaps that must be filled before there can be successful application.

The Organization of this Report

During the contract period both the projects in Parallel Distributed Processing (PDP) and Human-Machine Interaction (HMI) have completed manuscripts for major books that review the work of the laboratory and also move the field forward in major ways. The books themselves provide the best summaries of our efforts in these two areas. Accordingly, we felt the best way of preparing these sections of the report was to prepare a summary of the books. The review of the PDP effort is presented by the chapter in the PDP book that best provides the overall summary. It is written by Rumelhart, Hinton, and McClelland and entitled "A General Framework for Parallel Distributed Processing." All three of the authors have contributed to the contract efforts. McClelland was closely associated with the work at UCSD until he moved to Carnegie-Mellon University at the beginning of the 1984/85 academic year. Hinton was a postdoctoral fellow in the Cognitive Science program at UCSD where this work initiated (see the book that resulted from the earlier Cognitive Science program at UCSD: Hinton & Anderson, 1981). Hinton is now at Carnegie-Mellon University.

The work on Human-Machine Interaction is summarized through the book introduction by Draper and Norman and its section introductions.

The work on study of Skilled Human Performance is summarized in a report prepared by Donald Gentner.

PARALLEL DISTRIBUTED PROCESSING

EXPLORATIONS IN THE MICROSTRUCTURE OF COGNITION

The following is a list of the chapters which will appear in the above titled two volume book to be published by MIT Press/Bradford Books, Cambridge, Massachusetts.

VOLUME 1: FOUNDATIONS edited by David E. Rumelhart and James L. McClelland.

I. The PDP Perspective

1. The Appeal of PDP
J. L. McClelland, D. E. Rumelhart, and G. E. Hinton
2. A General Framework for PDP
D. E. Rumelhart, G. E. Hinton, and J. L. McClelland
3. Distributed Representations
G. E. Hinton, J. L. McClelland, and D. E. Rumelhart
4. Discussion & Preview
D. E. Rumelhart and J. L. McClelland

II. Models of Basic Mechanisms

5. Feature Discovery by Competitive Learning
D. E. Rumelhart and D. Zipser
6. Introduction to Harmony Theory (Smolensky)
P. Smolensky
7. Learning and Relearning in Boltzmann Machines
G. E. Hinton and T. J. Sejnowski
8. Error-Driven Learning in Multilayer Networks
D. E. Rumelhart, G. E. Hinton, and R. J. Williams

III. Formal Analyses of PDP Models

9. An Introduction to Linear Systems
M. I. Jordan
10. The Logic of Activation Rules
R. J. Williams
11. The Delta Rule
G. Stone
12. P3: A Parallel Network Simulating System
D. Zipser and D. Rabin
13. Resource Requirements of PDP Networks
J. L. McClelland

VOLUME II: APPLICATIONS edited by James L. McClelland and David E. Rumelhart

IV. Models of Psychological Processes

Higher Level Processes

14. Schemata
D. E. Rumelhart, P. Smolensky, J. L. McClelland, G. E. Hinton

Perception, Learning, Memory, and Thought

15. TRACE Model of Speech Perception
J. L. McClelland and J. L. Elman
16. The Programmable Blackboard Model of Reading
J. L. McClelland
17. A Model of Distributed Memory and Amnesia
J. L. McClelland and D. E. Rumelhart
18. On Learning the Past Tenses of English Verbs
D. E. Rumelhart and J. L. McClelland
19. Thought Processes
D. E. Rumelhart, P. Smolensky, J. L. McClelland, G. E. Hinton

V. PDP Models & the Brain

- 20. Certain Aspects of the Anatomy and Physiology of the Cerebral Cortex
F. Crick and C. Asanuma
- 21. Five Questions about Cerebral Cortex
T. J. Sejnowski
- 22. Neural and Conceptual Interpretation of PDP Models(Smolensky)
P. Smolensky
- 23. Biologically Plausible Models of Place Recognition and Goal Location
D. Zipser
- 24. Neural Plasticity & Critical Periods
P. W. Munro

VI. Epilog

- 25. Reflections: Cognition and Parallel Distributed Processing
D. A. Norman
- 26. The Future
J. L. McClelland and D. E. Rumelhart

What makes people smarter than computers? This book suggests that the answer lies in the massively parallel architecture of the human mind. While thought and problem solving have a sequential character when viewed over a time frame of minutes or hours, we argue that each step in the sequence is the result of the simultaneous activity of a very large number of simple computational elements, each influencing others and being influenced by them. This book describes our work in developing a theoretical framework to the development of models of aspects of perception, memory, language, and thought.

Volume 1 lays the theoretical foundations of parallel distributed processing. Section 1 introduces the approach and the reasons why we feel it is a fruitful one. Section 2 describes several models of basic mechanisms with wide applicability to different problems. Section 3 presents a number of specific technical analyses of different aspects of parallel distributed models.

Volume 2 applies the approach to a number of specific issues in Cognitive Science and Neuroscience. Section 4 describes parallel distributed processing models of thought, speech perception, and reading, and models of aspects of memory and language acquisition. Section 5 discusses the relation between parallel distributed processing models and neurophysiology, and describes a number of models which are specifically addressed to neurophysiological data. Section 6 concludes the work with a discussion of the strengths and weaknesses of the approach and directions for the future.

A GENERAL FRAMEWORK FOR PARALLEL DISTRIBUTED PROCESSING

D. E. Rumelhart, G. E. Hinton, and J. L. McClelland

This chapter provides a general framework for PDP models and then shows properties of certain specific realizations of the general model.¹

The General Framework

It is useful to begin with an analysis of the various components of our models and then describe the various specific assumptions we can make about these components. There are eight major aspects of a parallel distributed processing model:

- (a) a set of processing units
- (b) a state of activation
- (c) an output function for each unit
- (d) a pattern of connectivity among units
- (e) a propagation rule for propagating patterns of activities through the network of connectivities
- (f) an activation rule for combining the inputs impinging on a unit with the current state of that unit to produce a new level of activation for the unit.
- (g) a learning rule whereby patterns of connectivity are modified by experience
- (h) an environment within which the system must operate.

Figure 1 illustrates the basic aspects of these systems. There is a set of processing units generally indicated by circles in our diagrams; at each point in time, each unit u_i has an activation value, denoted in the diagram as $a_i(t)$; this activation value is passed through a function f_i to produce an output value $o_i(t)$. This output value can be seen as passing through a set of unidirectional connections (indicated by lines or arrows in our diagrams) to other units in the system. There is, associated with each connection, a real number, usually called the *weight* or *strength* of the connection designated w_{ij} which determines the amount of effect that the first unit has on the second. All of the inputs must then be combined by some operator (usually addition) and the combined inputs to a unit, along with its current activation value, determine, via a function F , its new activation value. The figure shows illustrative examples of the function f and F . Finally, these systems are viewed as being plastic in the sense that the pattern of interconnections is not fixed for all time. Rather the weights can undergo modification as a function of experience. In this way the system can evolve. What a unit represents can change with experience and the system can come to perform in substantially different ways. In the following sections we develop an explicit notation for each of these components and describe some of the

1. We are, of course, not the first to attempt a general characterization of this general class of models. Amari (1977) and Feldman and Ballard (1982) are two papers with similarly general aims.

The pattern association paradigm is the typical learning situation for a linear model. There is a set of input units and a set of output units. In general each input unit may be connected to any output unit. Since this is a linear network, there is no feedback in the system nor are there internal units between the inputs and outputs. There are two sources of input in the system. There are the input patterns which establish a pattern of activation on the input units and there are the teaching units which establish a pattern of activation on the output units. Any of several learning rules could be employed with a linear network such as this, but the most common are the simple Hebbian rule and the delta rule. The linear model with the simple Hebbian rule is called the simple *linear associator* (c.f. Anderson 1970). In this case, the increment in weight w_{ij} is given by $\Delta w_{ij} = \eta a_j t_i$. In matrix notation, this means that $\Delta W = \eta T A^T$. The system is then tested by presenting an input pattern without a teaching input and see how close the pattern generated on the output layer matches the original teaching input. It can be shown that if the input patterns are orthogonal,⁴ there will be no interference and the system will perfectly produce the relevant associated patterns exactly on the output layer. If they are not orthogonal, however, there will be interference among the input patterns. It is possible to make a modification in the learning rule and allow a much larger set of possible associations. In particular, it is possible to build up correct associations among patterns whenever the set of input patterns are linearly independent. To achieve this an error correcting rule must be employed. The delta rule is most commonly employed. In this case, the rule becomes $\Delta w_{ij} = \eta (t_i - a_i) a_j$. What is learned is essentially the difference between the desired response and that actually attained at unit u_i due to the input. Although it may take many presentations of the input pattern set, if the patterns are linearly independent the system will eventually be able to produce the desired outputs.

The examples described above were for the case of the pattern associator. Essentially the same results hold for the auto-associator version of the linear model. In this case, the input patterns and the teaching patterns are the same and the input layer and the output layer are also the same. The tests of the system involve presenting a portion of the input pattern and having the system attempt to reconstruct the missing parts.

Linear Threshold Units

The weaknesses of purely linear systems can be overcome through the addition of nonlinearities. Perhaps the simplest of the nonlinear system consists of a network of linear threshold units. The linear threshold unit is a binary unit whose activation takes on the values {0,1}. The activation value of unit u_i is 1 if the weighted sum of its inputs is greater than some threshold Θ_i and zero otherwise. The connectivity matrix for a network of such units, like the linear system, is a matrix consisting of positive and negative numbers. the output function, f , is the identity function so that the output of a unit is equal to its activation value.

It is useful to see some of the kinds of functions which can be computed with linear threshold units which cannot be computed with simple linear models. The classic such function is the *exclusive or* (XOR) illustrated in Figure 4. The idea is to have a system which responds {1} if it receives a {01} or a {10} and responds {0} otherwise. The figure shows a network capable of this pattern. In this case

4. See Chapter 9 for a discussion of orthogonality, linear independence and the like.

updating at a time. Among other things, this system can help the stability of the network by keeping it out of oscillations which are more readily entered into with synchronous update procedures.

Specific Versions of the General Parallel Activation Model

In the following sections we will show how specification of the particular functions involved produces various of these models.

Simple Linear Models

Perhaps the simplest model of this class is the simple linear model. In the simple linear model activation values are real numbers without restriction. They can be either positive or negative and are not bounded. The output function, $f(a_i)$ in the linear model is just equal to the activation level a_i . Typically, linear models consist of two sets of units, a set of *input* units and a set of *output* units. (As discussed below, there is no need for internal units since all computation possible with a multiple step linear system can be done with a single step linear system.) In general, any unit in the input layer may connect to any unit in the output layer. All connections in a linear model are of the same type. Thus, only a single connectivity matrix is required. The matrix consists of a set of positive, negative and zero values, for excitatory, inhibitory values and zero connections, respectively. The new value of activation of each unit is simply given by the weighted sums of the inputs. For the simple linear model with connectivity matrix W we have

$$A(t+1) = WA(t).$$

In general, it can be shown that a linear model such as this has a number of limitations. In particular, it can be shown that nothing can be computed from two or more steps that cannot be computed by a single step. This follows because the above equation implies

$$A(t+1) = W^t A(0).$$

We can see this by proceeding step by step. Clearly,

$$A(2) = WA(1) = W(WA(0)) = W^2A(0)$$

It should be clear that similar arguments lead to $A(t+1) = W^t A(0)$. From this, it follows that for every linear model with connectivity matrix W , that can attain a particular state in t steps there is another linear model, with connectivity matrix W^t , that can reach the same state in one step. This means, among other things, that there can never be any computational advantage in a linear model of multiple step systems, nor can there ever be any advantage for allowing feedback.

Table 3

	Level 1 units				Level 2 units				Level 3 units			
	u1	u2	u3	u4	u5	u6	u7	u8	u9	u10	u11	u12
Level 1	u1											
"input"	u2											
units	u3											
	u4											
Level 2	u5											
"internal"	u6											
units	u7											
	u8											
Level 3	u9											
"output"	u10											
units	u11											
	u12											

It is sometimes supposed that a "single level" system with *no hierarchical structure* in which any unit can communicate with any other unit is somehow less powerful than these multilevel hierarchical systems. The present analysis shows that, on the contrary, the *existence of levels* amounts to a *restriction* on the general of free communication among all units. Such *nonhierarchical* systems actually form a superset of the kinds of *layered* systems discussed above. There is, however, something to the view that having multiple levels can increase the power of certain systems. In particular, a "one-step" system consisting of only input and output units and no communication between them in which there is no opportunity for feedback or for "internal" units is less powerful than systems with "internal" units and with feedback. Since, in general, hierarchical systems involve many "internal" units, some intralevel communication and some feedback among levels, they are more powerful than systems not involving such "internal" units. However, a system with an equal number of "internal" units, but one not characterizable as hierarchical by the communication patterns is, in general, of more potential computational power. We address the issue of "internal units" and "single step" versus "multiple-step" systems in our discussion of specific models below.

Synchronous versus Asynchronous Update

Even given all of the components of the PDP models we have described so far, there is still another important issue to be resolved in the development of specific models. That is the timing of the application of the activation rule. In some models, there is a kind of central timing pulse and after each such clock tick a new value is determined simultaneously for all units. This is a *synchronous update* procedure. It is usually viewed as a discrete, difference approximation, to an underlying continuous, differential equation in which all units are continuously updated. In some models, however, units are updated *asynchronously* and at random. The usual assumption is that each at each point in time each unit has a fixed probability of evaluating and applying its activation rule and updating its activation value. This latter method has certain theoretical advantages and was developed by Hopfield (1987) and has been employed in Chapter 6, Chapter 7, and Chapter 14. The major advantage is that since the units are independently being updated if we look at a short enough time interval only one unit is

divided up into 9 regions. The upper-left region represents interactions among level 1 units. The entries in the upper-middle region of the matrix represents the effects of level 1 units on level 2 units. The upper-right region represents the effects of level 1 units on level 2 units. Often bottom-up models do not allow units at level i effect units at level $i+2$. Thus, in the diagram we have left that region empty representing no effect of level 1 on level 3. It is typical in a bottom-up system to assume as well that the lowest level units (level 1) are input units and that the highest level units (level 3) are output units. That is, the lowest level of the system is the only one to receive direct inputs from outside of this module and only the highest level units affect other units outside of this module.

Top-Down Processing

The generalization to a hierarchical top-down system should be clear enough. Let us order the units into levels just as before. A top-down model then requires that the upper-right regions of the weight matrix be empty—that is, no lower level unit affects a higher level unit. Table 2 illustrates a simple example of a top-down processing system. Note, in this case, we have to assume a top-down

Table 2

		Level 1 units				Level 2 units				Level 3 units			
		u1	u2	u3	u4	u5	u6	u7	u8	u9	u10	u11	u12
Level 1	u1	within											
"input"	u2	level 1											
units	u3	effects											
	u4												
Level 2	u5	level 2				within							
"internal"	u6	affecting				level 2							
units	u7	level 1				effects							
	u8												
Level 3	u9					level 3				within			
"output"	u10					affecting				level 3			
units	u11					level 2				effects			
	u12												

input or "message" which is propagated down the system from higher to lower levels as well as any data input which might be coming directly into level 1 units.

Interactive Models

Interactive models are simply models in which there can be both top down and bottom up connections. Again the generalization is straightforward. In the general interactive model any of the cells of the weight matrix could be nonzero. The more restricted models in which information flows both ways, but in which information only flows between adjacent levels assume only that the regions of the matrix more than one region away from the main diagonal are zero. Table 3 illustrates a simple three level interactive model with both top-down and bottom-up input. Most of the models that actually have been suggested count as interactive models in this sense.

sometimes called unsupervised learning. Each different kind of unsupervised learning procedure has its own evaluation function. The particular evaluation procedures are mentioned when we treat these models. The three unsupervised learning models discussed in this book are addressed in Chapters 5, 6, and 7.

Hierarchical Organizations of PDP Networks

It has become commonplace in cognitive science to describe processes as "top-down," "bottom-up," "interactive" to consist of many stages of processing etc. It is useful to see how these concepts can be represented in terms of the patterns of connectivity in the PDP framework. It is also useful to get some feeling for the processing consequences of these various assumptions.

Bottom-Up Processing

The fundamental characteristic of a bottom-up system is that units at level i may not affect the activity of units at levels lower than i . To see how this maps onto the current formulation it is useful to partition the coalitions of units into a set of discrete categories corresponding to the levels their inputs come from. There are assumed to be no coalitions with inputs from more than one level. Assume that there are L_i of units at level i in the system. We then order the units such that those in level L_1 are numbered $u_1 \dots u_{L_1}$, those in level L_2 are numbered $u_{L_1+1} \dots u_{L_1+L_2}$, etc. Then, the constraint that the system be a pure bottom-up system is equivalent to the constraint that the connectivity matrix, W , has zero entries for w_{ij} in which u_j is the member of a level no higher than u_i . This amounts to the requirement that the lower left hand region of W contains zero entries. Table 1 shows this constraint graphically. The table shows an example of a three level system with four units at each level.³ This leads to a 12x12 connectivity matrix and an A vector of length 12. The matrix can be

Table 1

	Level 1 units				Level 2 units				Level 3 units			
	u1	u2	u3	u4	u5	u6	u7	u8	u9	u10	u11	u12
Level 1	u1											
"input"	u2	within										
units	u3	level 1										
	u4	effects										
Level 2	u5											
"internal"	u6					within						
units	u7					level 2						
	u8					effects						
Level 3	u9											
"output"	u10									within		
units	u11									level 3		
	u12									effects		

3. In general, of course, we would expect many levels and many units at each level.

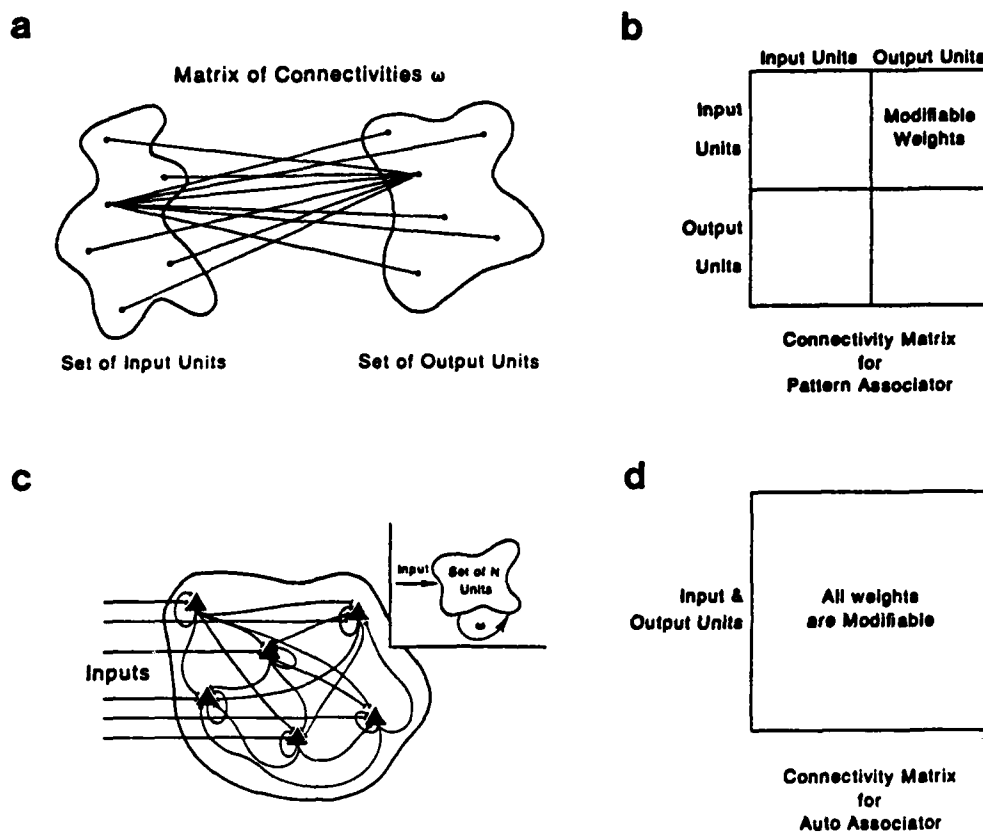


Figure 3. (a) Shows the basic structure of the pattern association situation. There are two distinct groups of units—a set of input units and a set of output units. Each input unit connects with each output unit and each output unit receives an input from each input unit. During training patterns are presented to both the input and output units. The weight connecting the input to the output units are modified during this period. During test patterns are presented to the input units and a response on the output units is measured. (after Anderson, 1977) (b) The connectivity matrix for the pattern associator. The only modifiable connections are from the input units to the output units. All other connections are fixed at zero. (c) The basic structure of the auto-association situation. All units are both input and output units. The figure shows a group of 6 units feeding back on itself through modifiable connections. Note that each unit feeds back on itself as well as on each of its neighbors. (after Anderson, Silverston, Ritz, & Jones, 1977). (d) The connectivity matrix for the auto-associator. All units connect to all other units with modifiable weights.

- (2) *Regularity discovery*, in which units learn to respond to "interesting" patterns in their input. In general, such a scheme should be able to form the basis for the development of feature detectors and therefore the basis for knowledge representation in a PDP system.

In certain cases these two modes of learning blend into one another, but it is valuable to see the different goals of the two kinds of learning. Associative learning is employed whenever we are concerned with storing patterns so that they can be re-evoked in the future. These rules are primarily concerned with storing the relationships among subpatterns. Regularity detectors are concerned with the *meaning* of a single units response. These kinds of rules are used when *feature discovery* is the essential task at hand.

The associative learning case generally can be broken down into two subcases—pattern association and auto-association. A *pattern association* paradigm is one in which the goal is to build up an association between patterns defined over one subset of the units and other patterns defined over a second subset of units. The goal is to find a set of connections so that whenever a particular pattern reappears on the first set of units, the associated pattern will appear on the second set. In this case, there is usually a *teaching input* to the second set of units during training indicating the desired pattern association. An *auto-association* paradigm is one in which an input pattern is associated with itself. The goal here pattern completion. Whenever a *portion* of the input pattern is presented the remainder of the pattern is to be filled in or completed. This is similar to simple pattern association, except that the input pattern plays both the role of the teaching input and of the pattern to be associated. It can be seen that simple pattern association is a special case of auto-association. Figure 3 illustrates the two kinds of learning paradigms. Figure 3A shows the basic structure of the pattern association situation. There are two distinct groups of units—a set of input units and a set of output units. Each input unit connect with each output units and each output unit receives an input from each input unit. During training patterns are presented to both the input and output units. The weight connecting the input to the output units are modified during this period. During test patterns are presented to the input units and a response on the output units is measured. Figure 3B shows the connectivity matrix for the pattern associator. The only modifiable connections are from the input units to the output units. All other connections are fixed at zero. Figure 3C shows the basic structure of the auto-association situation. All units are both input and output units. The figure shows a group of 6 units feeding back on itself through modifiable connections. Note that each unit feeds back on itself as well as on each of its neighbors. Figure 3D shows the connectivity matrix for the auto-associator. All units connect to all other units with modifiable weights. In the case of auto-association, there is potentially a modifiable connection from every unit to every other unit. In the case of pattern association, however, the units are broken into two subpatterns, one representing the input pattern and another representing the teaching input. The only modifiable connections are those from the input units to the output units receiving the teaching input. In other cases of associative learning the teaching input be more or less indirect. The problem of dealing with indirect feedback is difficult, but central to the development of more sophisticated models of learning. Barto & Sutton (1987) have begun a nice analysis of such learning situations.

In the case of regularity detectors, a teaching input is not explicitly provided, instead, the teaching function is determined by the unit itself. The form of the internal teaching function and the nature of its input patterns determine what features the unit will learn to respond to. This is

theorem has been proved. Still another variation has

$$\Delta w_{ij} = \eta a_i(t)(o_j(t) - w_{ij}).$$

This is a rule employed by Grossberg (1976) and a simple variant of which has been employed by us in Chapter 5. There are many variations on this generalized rule, and we will describe some of them in more detail when we discuss various specific models below.

Representation of the environment. It is crucial in the development of any model to have a clear model of the environment in which the model is to exist. In PDP models, we represent the environment as a time varying stochastic function over the space of input patterns. That is, we imagine that at any point in time, there is some probability that any of the possible set of input patterns is impinging on the input units. This probability function may in general depend on the history of inputs to the system as well as outputs of the system. In practice, most PDP models involve a much simpler characterization of the environment. Typically, the environment is characterized by a stable probability distribution over the set of possible input patterns independently of past inputs and past responses of the system. In this case, we can imagine listing the set of possible inputs to the system and numbering them from 1 to M . The environment is then characterized by a set of probabilities, p_i for $i = 1, \dots, M$. Since each input pattern can be considered a vector it is sometimes useful to characterize those patterns with nonzero probabilities as constituting "orthogonal" or "linearly independent" sets of vectors.² Certain PDP models are restricted in the kinds of patterns they are able to learn. Some being able to learn to respond correctly only if the input vectors form an orthogonal set, others if they form a linearly independent set of vectors and still others are able to learn to respond to essentially arbitrary patterns of inputs.

Classes of PDP Models

There are many paradigms and classes of PDP models that have been developed. In this section we describe some general classes of assumptions and paradigms. In the following section we describe some specific PDP models and show their relationships to the general framework outlined here.

Paradigms of Learning

Although most learning rules have roughly the form indicated above, we can categorize the learning situation into two distinct sorts. These are:

- (1) *Associative learning*, in which we learn to produce a particular pattern of activation on one set of units whenever another particular pattern occurs on another set of units. In general, such a learning scheme must allow an arbitrary pattern on one set of units to produce another arbitrary pattern on another set of units.

2. See Chapter 9 for explication of these terms.

- (1) The development of new connections.
- (2) The loss of existing connections.
- (3) The modification of the strengths of connections which already exist.

Very little work has been done on (1) and (2) above. To a first order of approximation, however, (1) and (2) can be considered a special case of (3). Whenever we change the strength of connection away from zero to some positive or negative value that has the same effect as growing a new connection. Whenever we change the strength of a connection to zero that has the same effect as losing an existing connection. Thus, in this section we will concentrate on rules whereby *strengths* of connections are modified through experience.

Virtually all learning rules for models of this type can be considered a variant of the *Hebbian* learning rule suggested by Hebb in his classic book *Organization of Behavior*, (1949). Hebb's basic idea is this. If a unit, u_j , receives a input from another unit, u_i , then, if both are highly active, the weight, w_{ij} , from u_j to u_i should be *strengthened*. This idea has been extended and modified so that it can be more generally stated as

$$\Delta w_{ij} = g(a_i(t), t_i(t)) h(o_j(t), w_{ij}),$$

where $t_i(t)$ is a kind of *teaching* input to u_i . Simply stated this equation says that the change in the connection from u_j to u_i is given by the product of a function, $g()$, of the activation of u_i and its teaching input t_i and another function, $h()$, of the output value of u_j and the connection strength w_{ij} . In the simplest versions of Hebbian learning there is no teacher and the function g and h are simply proportional to their first arguments. Thus we have

$$\Delta w_{ij} = \eta a_i o_j,$$

where η is the constant of proportionality representing the learning rate. Another common variation is a rule in which $h(o_j(t), w_{ij}) = o_j(t)$ and $g(a_i(t), t_i(t)) = \eta(t_i(t) - a_i(t))$. This is often called the *Widrow-Hoff* rule. However, we call it the *delta rule* because the amount of learning is proportional to the *difference* (or delta) between the actual activation achieved and the target activation provided by a teacher. In this case we have

$$\Delta w_{ij} = \eta(t_i(t) - a_i(t)) o_j(t).$$

This is a generalization of the *perceptron* learning rule for which the famous *perception convergence*

The pattern of connectivity is very important. It is this pattern which determines what each unit represents. As we shall see below many of the issues concerning whether "top-down" or "bottom-up" processing systems are correct descriptions or whether a system is hierarchical and if so how many levels it has, etc. are all issues of the nature of the connectivity matrix. One important issue which may determine both how much information can be stored and how much serial processing the network must perform, is the *fan in* and *fan out* of a unit. The *fan in* is the number of elements which either excite or inhibit a given unit. The *fan out* of a unit is the number of units affected directly by a unit. Note, in some cases we need more general patterns of connectivity. Specifying such a pattern in the general case is complex and will be addressed in a later section of this chapter.

The rule of propagation. We also need a rule which takes the output vector, $O(t)$, representing the output values of the units and combines it with the connectivity matrices to produce a *net input* for each type of input into the unit. We let net_{ij} be the net input of type i to unit u_j . Whenever only one type of connectivity is involved we suppress the first subscript and use net_j to mean the net input into unit u_j . In vector notation we can write $NET_i(t)$ to represent the net input vector for inputs of type i . The propagation rule is generally straight forward. For example, if we have two types of connections, inhibitory and excitatory, the net excitatory input is usually simply the weighted sum of the excitatory inputs to the unit. This is given by the vector product $NET_e = W_e O(t)$. Similarly, the net inhibitory effect can be written as $NET_i = W_i O(t)$. When more complex patterns of connectivity are involved, more complex rules of propagation are required. We treat this too in the final section of the chapter.

Activation rule. We also need a rule whereby the net inputs of each type impinging on a particular unit are combined with one another and with the current state of the unit to produce a new state of activation. We need a function, F , which takes $A(t)$, and the vectors NET_j for each different type of connection and produce a new state of activation. In the simplest cases, when F is the identity function and when all connections are of the same types, we can write $A(t+1) = WO(t) = NET(t)$. Sometimes F is a threshold function so that the net input must exceed some value before contributing to the new state of activation. Often, the new state of activation depends on the old one as well as the current input. In general, however, we have

$$A(t+1) = F(A(t), NET(t)_1, NET(t)_2, \dots)$$

the function F itself is what we call the activation rule. Usually, the function is assumed to be deterministic. Thus, for example, if a threshold is involved it may be that $a_i(t) = 1$ if the total input exceeds some threshold value and 0 otherwise. Other times it is assumed that F is stochastic. Sometimes activations are assumed to decay slowly with time so that even with no external input the activation of a unit will simply decay and not go directly to zero. Whenever $a_i(t)$ is assumed to take on continuous values it is common to assume that F is a kind of sigmoid function. In this case, an individual unit can *saturate* and reach a minimum or maximum value of activation.

Modifying patterns of connectivity as a function of experience. Changing the processing or knowledge structure in a parallel distributed processing model involves modifying the patterns of interconnectivity. In principle this can involve three kinds of modifications

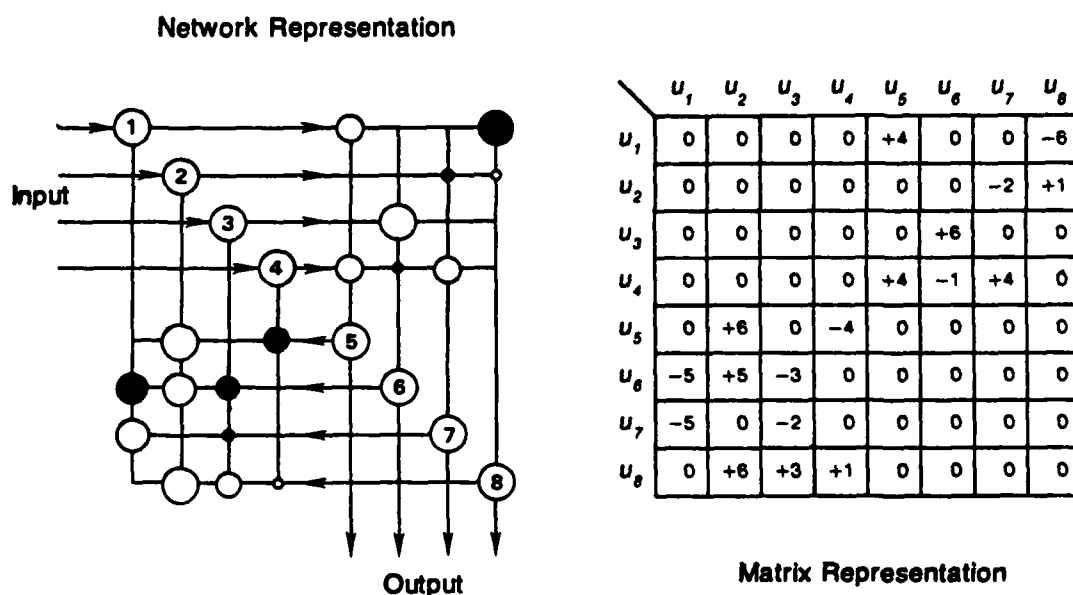


Figure 2. The connectivity of a network represented by a network drawing and in a matrix. The figure shows an eight unit network with units numbered from 1 to 8. Units 1 to 4 are input units. They receive inputs from the outside world and feedback from the output units—units 5 through 8. The connections among the units are indicated by the open and filled disks. The size of the disk indicates the strength of connection. Thus, the large black disk on the line connecting unit 1 to unit 8 indicates a strong inhibitory connection from 1 to 8. Similarly, the large open disk on the output line from unit 8 to unit 2 indicates that unit 8 strongly excites unit 2. The same connections are shown in the matrix representation on the left. The +6 in the row for u_8 and the column for u_2 indicates that unit 8 strongly excites unit 2. It will be noted that whenever there is a disk on a line connecting the output of one unit to the input of another in the network diagram there is a corresponding nonzero entry in the matrix. If the disk is filled, the entry in the matrix is negative. If the disk is open the entry is positive. The larger the disk the greater the magnitude of the entry in the matrix. It might also be noted that the connections in the network have been laid out to correspond to the entries of the matrix. The black disk in the upper left corner of the network corresponds to the -6 in the upper left corner of the matrix. Each disk in the network is in the corresponding position of its location in the matrix. The network would not have had to be drawn in that way, of course, and the matrix would still capture all of the connectivity information in the network. In general, because network drawings are difficult to work with we will often simply use the matrix representation to specify the pattern of connectivity.

implications of these various assumptions.

Output of the units. Units interact. They do so by transmitting signals to their neighbors. The strength of their signals, and therefore the degree to which they affect their neighbors, is determined by their degree of activation. Associated with each unit, u_i , there is an output function, $f_i(a_i(t))$, which maps the current state of activation $a_i(t)$ to an output signal $o_i(t)$ (i.e. $o_i(t) = f_i(a_i(t))$). In vector notation, we represent the current set of output values by a vector, $O(t)$. In some of our models the output level is exactly equal to the activation level of the unit. In this case f is the identity function $f(x) = x$. More often, however, f is some sort of threshold function so that a unit has no affect on another unit unless its activation exceeds a certain value. Sometimes the function f is assumed to be a stochastic function in which the output of the unit depends in a probabilistic fashion on its activation values.

The pattern of connectivity. Units are connected to one another. It is this pattern of connectivity which constitutes what the system knows and determines how it will respond to any arbitrary input. Specifying the processing system and the knowledge encoded therein is, in a parallel distributed processing model, a matter of specifying this pattern of connectivity among the processing units.

In many cases, we assume that each unit provides an additive contribution to the input of the units to which it is connected. In such cases, the total input to the unit is simply the weighted sum of the separate inputs from each of the individual units. That is, the inputs from all of the incoming units are simply multiplied by a weight and summed to get the overall input to that unit. In this case, the total pattern of connectivity can be represented by merely specifying the weights for each of the connections in the system. A positive weight represents an excitatory input and a negative weight represents an inhibitory input. As mentioned in the previous chapter, it is often convenient to represent such a pattern of connectivity by a weight matrix W in which the entry w_{ij} represents the strength and sense of the connection from unit u_j to unit u_i . w_{ij} is a positive number if unit u_j excites unit u_i ; it is a negative number if unit u_j inhibits unit u_i ; and it is 0 if unit u_j has no direct connect to unit u_i . The absolute value of w_{ij} specifies the *strength of the connection*.

Figure 2 illustrates the relationship between the connectivity and the weight matrix.

In the general case, however, we require rather more complex patterns of connectivity. A given unit may receive inputs of different kinds whose affects are separately summated. For example, in the previous paragraph we assumed that the excitatory and inhibitory connections simply summed algebraically with positive weights for excitation and negative weights for inhibition. Sometimes, more complex inhibition/excitation combination rules are required. In such cases it is convenient to have separate connectivity matrices for each kind of connection. Thus, we can represent the pattern of connectivity by a set of connectivity matrices, W_k , one for each *type* of connection. It is common, for example, to have two types of connections in a model, an inhibitory and an excitatory connection. When the models assume simple addition of inhibition and excitation they do not constitute different *types* in our present sense. They only constitute distinct types when they combine through some more complex rules.

alternate assumptions which have been made concerning each such component.

A set of processing units. Any parallel activation model begins with a set of processing units. Specifying the set of processing units and what they represent is typically the first stage of specifying an PDP model. In some models these units may represent particular conceptual objects such as features, letters, words or concepts in others they are simply abstract elements over which meaningful patterns can be defined. When we speak of a distributed representation, we mean one in which the units represent small, feature-like entities. In this case it is the pattern as a whole which is the meaningful level of analysis. This should be contrasted to a *one unit one concept* representational system in which single units represent entire concepts or other large meaningful entities.

We let N be the number of units. We can order the units arbitrarily and designate the i th unit u_i . All of the processing of a PDP model is carried out by these units. There is no executive or other overseer. There are only relatively simple units each doing its own relatively simple job. A unit's job is simply to receive input from its neighbors and, as a function of the inputs it receives, to compute an output value which it sends to its neighbors. The system is inherently parallel in that many units can carry out their computations at the same time.

Within any system we are modeling, it is useful to characterize three types of units *input*, *output* and *internal*—*input* units receive inputs from sources external to the system under study. These inputs may be either sensory input or inputs from other parts of the processing system in which the model is embedded. The *output* units send signals out of the system. They may either directly affect motoric systems or simply influence other systems external to the ones we are modeling. The *internal* units are those whose only inputs and outputs are within the system we are modeling.

The state of activation. In addition, to the set of units, we need a representation of the state of the system at time t . This is primarily specified by a vector of N real numbers, $A(t)$, representing the pattern of activation over the set of processing units. Each element of the vector stands for the activation of one of the units at time t . The activation of unit u_i at time t is designated $a_i(t)$. It is the pattern of activation over the set of units which captures what the system is representing at any time. It is useful to see processing in the system as the evolution, through time, of a pattern of activity over the set of units.

Different models make different assumptions about the activation values a unit is allowed to take on. Activation values may be continuous or discrete. If they are continuous, they may be unbounded or bounded. If they are discrete, they may take binary values or any of a small set of values. Thus in some models units are continuous and may take on any real number as an activation value. In other cases, they may take on any real value between some minimum and maximum such as, for example, the interval $(0,1)$. When activation values are restricted to discrete values they most often are binary. Sometimes they are restricted to the values 0 and 1 where 1 is usually taken to mean that the unit is active and a zero is taken to mean that it is inactive. In other models, activation values are restricted to the values $\{-1,+1\}$ (often denoted simply $\{-,+\}$). Other times nonbinary discrete values are involved. Thus, for example, they may be restricted to the set $\{-1,0,+1\}$, or to a small finite set of values such as $\{1,2,3,4,5,6,7,8,9\}$. As we shall see, each of these assumptions leads to a model with slightly different characteristics. It is part of the program of research represented in this book to determine the

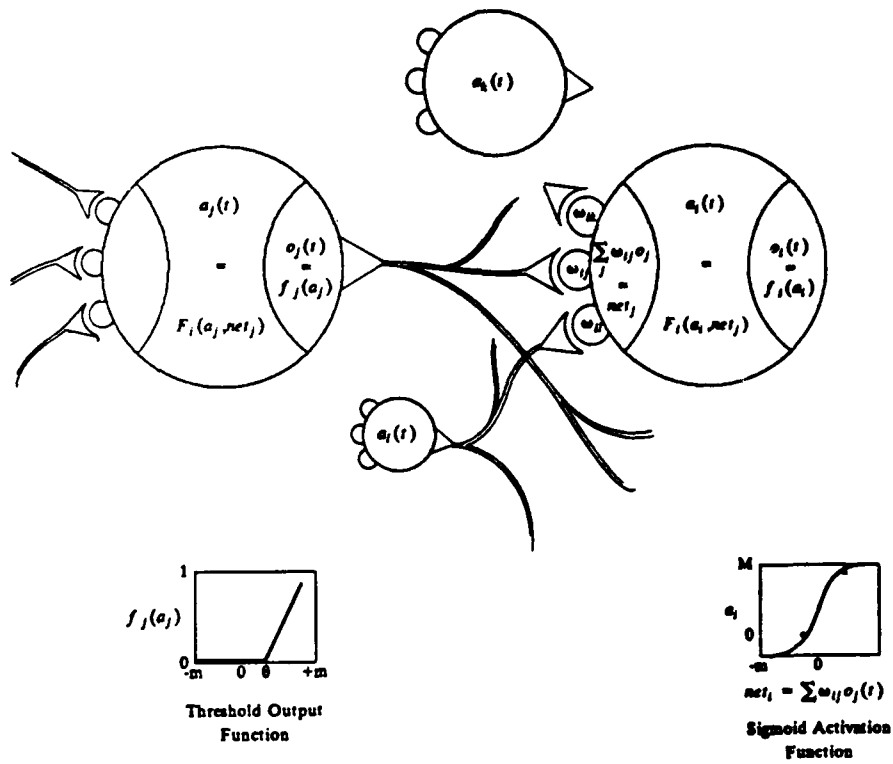


Figure 1. The basic components of a parallel distributed processing system.

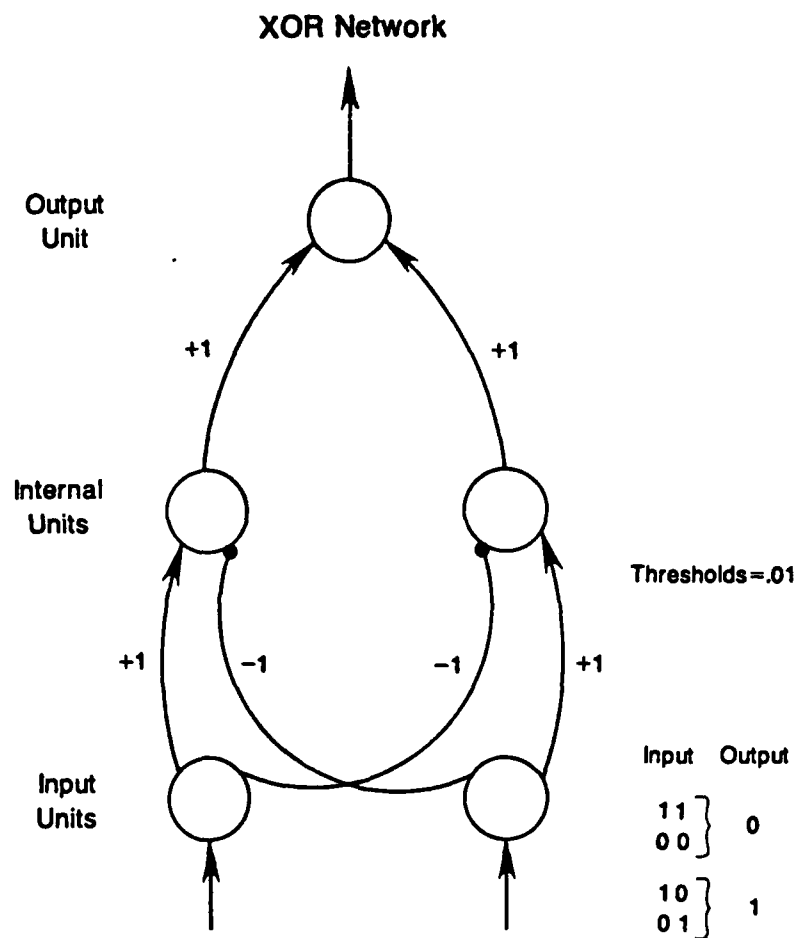


Figure 4. A network of linear threshold units capable of responding correctly on the XOR problem.

we require two layers of units. Each unit has a zero threshold and responds just in case its input is greater the zero. The weights are ± 1 . Since the set of stimulus patterns are not linearly independent this is a discrimination that can never be made by a simple linear model and cannot be done in a single step by any network of linear threshold units.

Although multilayered systems of linear threshold units are very powerful, and, in fact, are capable of computing any boolean function, there is no generally known learning algorithm for this general case. There is, however, a well understood learning algorithm for the special case of the *perceptron*. A perceptron is essentially a single layer network of linear threshold units without feedback. The learning situation here is exactly the same as that for the linear model. An input pattern is presented along with a teaching input. The perceptron learning rule is precisely the same form as the delta rule for error correcting in the linear model, namely, $\Delta w_{ij} = \eta(t_i - a_i)a_j$. Since the teaching input and the activation values are only zero or 1, the rule reduces to the statement that

- (1) Weights are only changed on a given input line when that line is turned on (i.e. $a_j = 1$).
- (2) If the system is correct on unit i (i.e. $t_i = a_i$) make no change on any of the input weights.
- (3) If the unit j responds zero when it should be one, increase weights on all active lines by amount η .
- (4) If the unit j responds one when it should be zero, decrease weights on all active lines by amount η .

There is a theorem, the perceptron convergence theorem, which guarantees that if the set of patterns are learnable by a perceptron this learning procedure will find a set of weights which allow it to respond correctly to all input patterns. Unfortunately, even though multilayer linear threshold networks are potentially much more powerful than the linear associator, the perceptron for which a learning result exists can learn no patterns not learnable by the linear associator. It was limitations on what perceptrons could possibly learn that led to Minsky and Pappert's (1968) pessimistic evaluation of the perceptron. Unfortunately that evaluation has incorrectly tainted more interesting and powerful networks of linear threshold and other nonlinear units. As we shall see in the course of this book the limitations of the one-step perceptron in no way applies to the more complex networks.

Brain State in a Box

The brain state in a box model was developed by Anderson (1977). This model too is a close relative of the simple linear associator. There is, however, a maximum and minimum activation value associated with each unit. Typically, units take on activation values in the interval $[-1,1]$. The brain state in a box (BSB) models are organized so that any unit can, in general be connected to any other unit. The auto-associator illustrated in Figure 3 is the typical learning paradigm for BSB. Note that with this pattern of interconnections the systems feeds back on itself and thus the activation can recycle through the system in a positive feedback loop. The positive feedback is especially evident in Anderson &

Moser's (1981) version. Their activation rule is given by

$$a_j(t+1) = a_j(t) + \sum w_{ij} a_i(t)$$

if a_j is less than 1 or greater than -1 . Otherwise, if the quantity is greater than 1, $a_j = 1$, and if it is less than -1 , $a_j = -1$. That is, the activation state at time $t+1$ is given by the sum of the state at time t and the activation propagated through the connectivity matrix provided that total is in the interval $[-1,1]$. Otherwise it simply takes on the maximum or minimum value. This formulation will lead the system to a state in which all of the units are at either a maximum or minimum value. It is possible to understand why this is called a brain-state-in-a-box model by considering a geometric representation of the system. Figure 5 illustrates the "activation space" of a simple BSB system consisting of three units. Each point in the box corresponds to a particular value of activation on each of the three units. Thus, each such point corresponds to a state of the system. dimensional space in which the first coordinate corresponds to the activation value of the first unit, the second coordinate corresponds to the activation value of the second unit and the third coordinate corresponds to the activation value of the third unit. Thus, each point in the space corresponds to a possible state of the system. The feature that each unit is limited to the region $[-1,1]$ means that all points must lie somewhere within the box whose vertices are given by the points $(-1,-1,-1)$, $(-1,-1,+1)$, $(-1,+1,-1)$, $(-1,+1,+1)$, $(+1,-1,-1)$, $(+1,-1,+1)$, $(+1,+1,-1)$, and $(+1,+1,+1)$. Moreover, since the system involves positive feedback, it is eventually forced to occupy one of these vertices. Thus, the state of the system is constrained to lie within the box and eventually, as processing continues, is pushed to one of the vertices. Of course, the same geometric analogy carries over to higher dimensional systems. If there are N units the state of the system can be characterized as a point within this N dimensional hypercube and eventually the system ends up in one of the 2^N corners of the hypercube.

Learning in the BSB system involves auto-association. In different applications two different learning rules have been applied. Anderson and Moser (1981) applied the simplest rule. They simply allowed the system to settle down and then employed the simple Hebbian learning rule. That is, $\Delta w_{ij} = \eta a_i a_j$. The error correction rule has also been applied to the BSB model. In this case we use the input as the teaching input as well as the source of activation to the system. The learning rule thus becomes $\Delta w_{ij} = \eta(t_i - a_i)a_j$ where t_i is the input to unit i and where a_i and a_j are the activation values of the system after it has stabilized in one of the corners of the hypercube.

Boltzmann Machines

Another more recent development is the *Boltzmann* machine developed by Hinton and Sejnowski (1987). The Boltzmann Machine is discussed fully in Chapter 7, however, it is useful to understand the basic idea and how it relates to the general class of models under discussion here. To begin, the Boltzmann machine consists of binary units which take on the values $\{0,1\}$. The units are divided into two categories, the *visible* units corresponding to our input and output units and the *hidden* units corresponding to our internal units. In general, any unit may connect to any other unit. However, there is a constraint that the connections must be symmetric. That is, the $w_{ij} = w_{ji}$. In the Boltzmann Machine, there is no distinction between the output of the unit and its activation value. The activa-

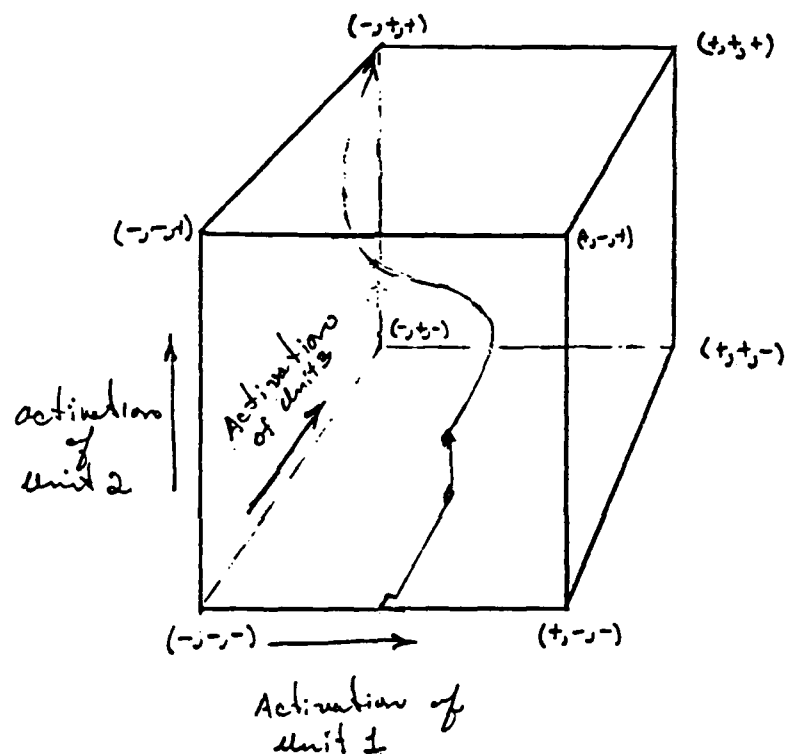


Figure 5. The state space for a three unit version of a BSB model. Each dimension of the box represents the activation value of one unit. Each unit is bounded in activation between $[-1,1]$. The curving arrow in the box represents the sequence of states the system moved through. It began at the black spot near the middle of the box and, as processing proceeded moved to the $(-, +, +)$ corner of the box. BSB systems always end up in one or another of the corners. The particular corner depends on the start state of the network, the input to the system and the pattern of connections among the units.

tion values are, however, a stochastic function of the inputs. That is,

$$p(a_i(t)=1) = e^{-\frac{(\sum_j w_{ij} a_j + \eta_i - \theta_i)/T}{1}}$$

where η_i is the input from outside of system into unit i , θ_i is the threshold for the unit and T is a parameter, called *temperature*, which determines the slope of the probability function. Figure 6 shows how the probabilities vary with various values of T . It should be noted that as T approaches zero the individual units become more and more like linear threshold units. In general, if the units exceeds threshold by a great enough margin it will always attain value 1. If it is far enough below threshold, it always takes on value zero. Whenever the unit is above threshold, the probability that it will turn on is greater than 1/2. Whenever it is below threshold the probability that it will turn off is greater than 1/2. The temperature simply determines the range of uncertainty as to whether it will turn on or off. This particular configuration of assumptions allows a formal analogy between the Boltzman Machine and thermodynamics and allows the proof of theorems concerning its performance as a function of the temperature of the system. This is not the place to discuss these theorems in detail, suffice it to say that this system, like the BSB system, can be viewed as attaining states on the corners of a hypercube. There is a global measure of the degree to which each state of the system is consistent with its input. The system moves into those states which are maximally consistent with the input and with the internal constraints represented by the weights. It can be shown that as the temperature approaches zero the probability that the system attains the maximally consistent state approaches 1. These results are discussed in some detail in Chapter 7.

The learning scheme associated with the Boltzman Machine is somewhat more complex than the others. In this case the learning events are divided into two classes. During one class a set of patterns are randomly presented to the *visible* units and the system is allowed to respond to each in turn. During this phase of learning the system is environmentally driven. During this phase, a simple Hebbian rule is assumed to apply so that $\Delta w_{ij} = \eta a_i a_j$. Note, since activations take on values of zero and 1 this says that the weight is incremented by an amount η whenever unit i and j are on, otherwise no change occurs. During the second phase of learning the system is allowed to respond for an equal period of time in a so-called free running state in which no inputs are presented. Since the system is stochastic, it will continue to respond even though no actual stimuli are presented. During this phase, a simple anti-Hebbian rule is employed, $\Delta w_{ij} = -\eta a_i a_j$. The intuition is roughly that the performance during the environmentally driven phase is determined by both the pattern of interconnections and by the environment. The performance during the free-running phase is determined only by the internal set of connections. To correctly reflect the environment, we should look at its performance due to the environment plus internal structure and then subtract out its performance due to internal structure alone. This is actually quite a powerful learning scheme, it can be shown that if a portion of the input units are turned on after the system has learned, it will complete the remaining portion of the visible units with the probability that those units had been present in the stimulus patterns given the subpattern that had been turned on. These issues are again addressed in Chapter 7.

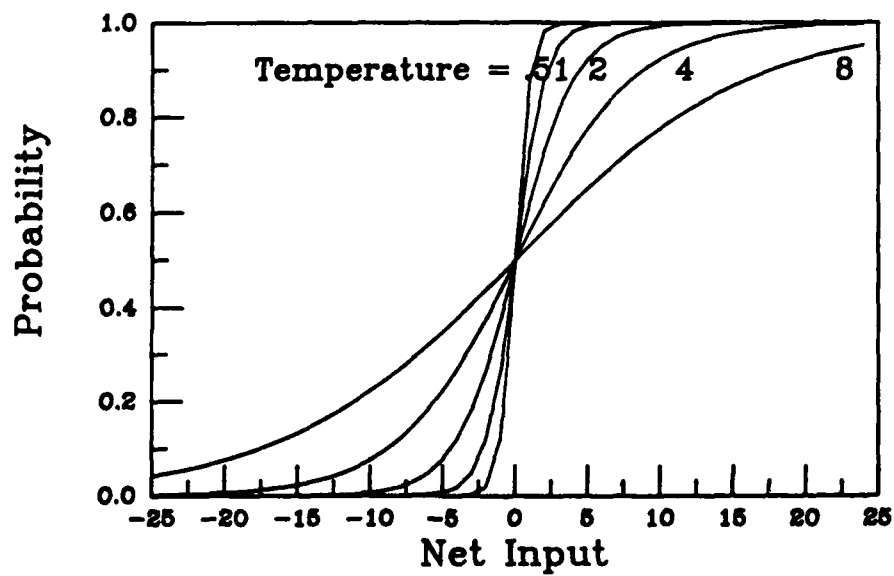


Figure 6. Probability of attaining value 1 as a function of the distance of the input of the unit from threshold. The function is plotted for several values of T .

Grossberg

Stephen Grossberg has been one of the major contributors to models of this class over the years. His work is complex and contains many important details which we cannot review here. We will instead simply describe some of the central aspects of his work and show how it relates to the general framework. Perhaps the clearest summary of Grossberg's work appears in Grossberg (1980). Grossberg's units are allowed to take on any real activation value between a minimum a maximum value. The output function is, in many of Grossberg's applications, a threshold function so that a given unit will affect another unit only if its activation level is above its threshold. Moreover, Grossberg argues that the output function must be a *sigmoid* or S-shaped function of the activation value of the unit. Grossberg's activation rule is rather more complex than the others we have discussed thus far in that excitatory and inhibitory inputs don't simply sum, but appear separately in the activation rule. Grossberg has presented a number of possible activation rules, but they typically have the form

$$a_j(t+1) = a_j(t)(1-A) + (B - a_j(t))net_{ej}(t) - (a_j(t) + C)net_{ij}(t)$$

where A is the decay rate, B represents the maximal degree of excitation of the unit and C is much smaller in magnitude than B and represents the maximal amount the unit can be inhibited below the resting value of 0. Grossberg generally assumes that the inhibitory inputs come from a kind of recurrent inhibitory field in which the unit is embedded and the excitatory inputs come from the unit itself and from another level of the system.

Grossberg has studied learning in these networks over a number of years and has studied several different learning schemes. The learning rule he has studied most, however, is similar to the one analyzed in Chapter 5 and is given by

$$\Delta w_{ij} = \eta a_i(o_j - w_{ij}).$$

Grossberg has applied this and similar learning rules in a number of cases, but a review of these applications is beyond the scope of the present discussion.

Interactive Activation Model

The interactive activation model of McClelland and Rumelhart (1981) and Rumelhart and McClelland (1982) had units which represented visual features, letters and words. Units could take on any value in the range [min,max]. The output function, was a threshold function such that the output was zero if the activation were below threshold and equal to the difference of the activation value and the threshold if the activation were above threshold. The interactive activation model involve a connectivity pattern in which units are organized in layers such that an element in a layer connect with excitatory connections with all elements in the layers above and below which are consistent with that unit and connects negatively to all units in the layers above and below which are inconsistent with that unit. In addition, each unit inhibits all units in its own layer which are inconsistent with the unit in

question. Thus, the interactive activation model is a kind of positive feedback system with maximum and minimum values for each unit, like the BSB model. The information coming into each unit is weighted (by the interconnection strengths) and summed algebraically to yield a "net input" to the unit. Let $net_j = \sum w_{ij} a_i$ be the net input to unit j . This net input, is then combined with the previous activation value to produce the new activation value according to the following activation rule.

$$a_j(t+1) = a_j(t)(1-\Theta) + \begin{cases} net_j(max - a_j(t)) & net_j > 0 \\ net_j(a_j(t) - min) & otherwise \end{cases}$$

where Θ is the decay rate of the activation given no input. In words, the new activation value is given by the old activation value properly decayed, plus (or minus) a factor which pushes toward the minimum or maximum value depending on the magnitude of the net input into the unit. This activation rule is similar to that employed by Grossberg, except in this formulation the excitation and inhibition are algebraically combined.

The interactive activation model was designed as a model for a processing system and our goals were to show how we could account for specific aspects of word perception. Thus, there was no specific model of learning proposed to explain where the particular network we assumed came from. As we shall see, much of the work on learning reported in this book has been aimed at giving plausible accounts of how such a network might have been learned. (See especially [CompLearn] and [Harmony]).

Feldman and Ballard

Feldman and Ballard (1982) have proposed a framework for what they dub *connectionism*. The units have continuous activation values, which they call *potential* which can take on any value in the range $[-10,10]$. Their output function is a kind of threshold function which is allowed to take on a small number of discrete integer values ($0 \leq o_i \leq 9$). They have proposed a number of other unit types each with a somewhat different activation rule. Their simplest unit type is what they call the *P-Unit*. In this case the activation rule is given by

$$a_j(t+1) = a_j(t) + \beta net_j(t)$$

Once the activation reaches its maximum or minimum value it is simply pinned to that value. Decay is implemented by self inhibition. Feldman & Ballard also have a *conjunctive* unit similar to our *sigma-pi* units described below. Feldman (1981) has also considered learning. In general the approach to learning offers more machinery than is available within our current framework. In practice, however, the learning rules actually examined are of the same class we have already discussed.

Sigma-Pi Units

Before completing our section on a general framework, it should be mentioned that we have sometimes found it useful to postulate units which are more complex than those described to this point in this chapter. In our descriptions thus far, we have assumed a simple additive unit in which the net input to the unit is given by $\sum w_{ij} a_i$. This is certainly the most common form in most of our models. Sometimes, however, we want multiplicative connections in which the output values of two (or possibly more) units are multiplied before entering into the sum. Such a multiplicative connection allows one unit to *gate* another. Thus, if one unit of a multiplicative pair is zero, the other member of the pair can have no effect, no matter how strong its output. On the other hand, if one unit of a pair has value 1 the output of the other is passed unchanged to the receiving unit. Figure 7 illustrates several such connections. In this case, the input to unit A is the weighted sum of the products of units B & C and units D & E. The pairs, BC and DE are called *conjuncts*. In this case we have conjuncts of size 2. In general, of course, the conjuncts could be of any size. We have no applications, however, which have required conjuncts larger than size 2. In general, then, we assume that the net input to a unit is given by the weighted sum of the products of a set of individual inputs. That is the net input to a unit is given by $\sum w_{ij} \prod a_{i_1} a_{i_2} \cdots a_{i_k}$ where i indexes the conjuncts impinging on unit j and $u_{i_1}, u_{i_2}, \cdots, u_{i_k}$ are the k units in the conjunct. We call units such as these *sigma-pi units*.

In addition to their use as gates, sigma-pi units can be used to convert the output level of a unit into a signal which acts like a *weight* connecting two units. Thus, assume we have the pattern of connections illustrated in the figure. Assume further that the weights on those connections are all 1. In this case, we can use the output levels of units B and D to, in effect, set the weights from C to A and E to A respectively. Since, in general, it is the weights among the units which determine the behavior of the network, sigma-pi units allow for a dynamically programable network in which the activation value of some units determine what another network can do. Indeed, we use this feature to advantage in Chapter 5.

In addition to its general usefulness in these cases, one might ask whether we might not sometime need still more complex patterns of interconnections. Interestingly, as described in Chapter 10 we will never be forced to develop any more complex interconnection type, since sigma-pi units are sufficient to mimic any function monotonic of its inputs.

Sigma Pi Units

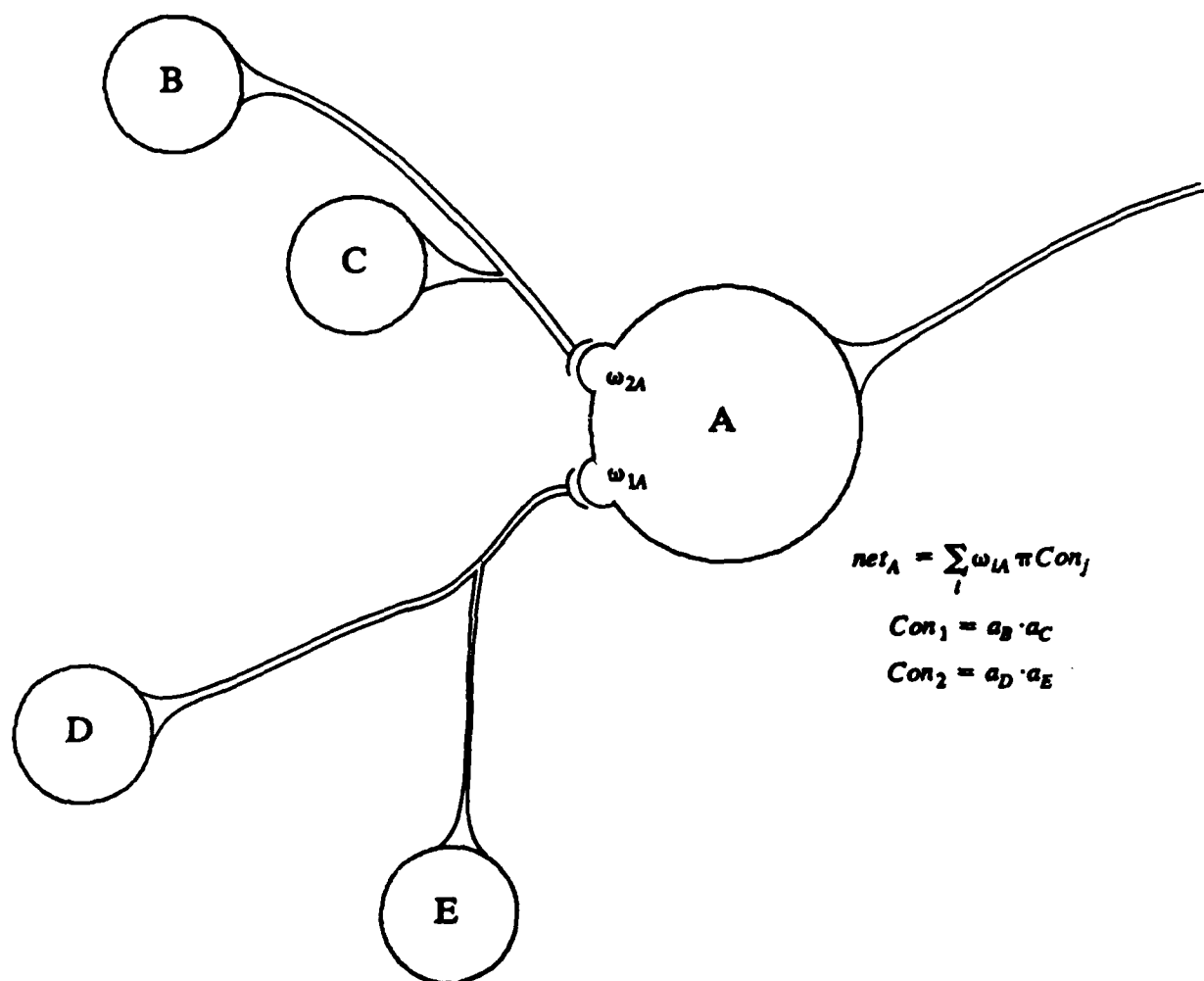


Figure 7. Two conjunctive input to unit A from the conjunct B & C and D & E. The input to unit A is the sum of the product of the outputs of units BC and DE.

References

- Anderson, J. A. (1977). Neural models with cognitive implications. In D. LaBerge & S. J. Samuels (Eds.), *Basic processes in reading: Perception and comprehension*. Hillsdale, NJ: Erlbaum.
- Anderson, J. A., & Mozer, M. (1981). Categorization and selective neurons. In G. E. Hinton & J. A. Anderson (Eds.), *Parallel models of associative memory*. Hillsdale, NJ: Erlbaum.
- Anderson, J. A., Silverstein, J. W., Ritz, S. A., & Jones, R. S. (1977). Distinctive features, categorical perception, and probability learning: Some applications of a neural model. *Psychological Review*, 84, 413-451.
- Barto, Andrew G., & Sutton, Richard (1981). Landmark learning: An illustration of associative search. *Biological Cybernetics*, 42, 1-8.
- Feldman, J. A., & Ballard, D. H. (1982). Connectionist models and their properties. *Cognitive Science*, 6, 205-254.
- Grossberg, S. (1976). Adaptive pattern classification and universal recoding, I: Parallel development and coding of neural feature detectors. *Biological Cybernetics*, 23, 121-134.
- Grossberg, S. (1980). How does the brain build a cognitive code?. *Psychological Review*, 87, 1-51.
- Hebb, D. O. (1949). *The organization of behavior*. New York: Wiley.
- Hinton, G. E., Sejnowski, T. J., & Ackley, D. H. (1984). *Boltzmann machines: Constraint satisfaction networks that learn* (Tech. Rep. No. CMU-CS-84-119). Pittsburgh, PA: Carnegie-Mellon University.
- Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences USA*, 79, 2554-2558.
- McClelland, J. L., & Rumelhart, D. E. (1981). An interactive activation model of context effects in letter perception: Part 1. An account of basic findings. *Psychological Review*, 88, 375-407.
- Minsky, M., & Papert, S. (1969). *Perceptrons*. Cambridge, MA: MIT Press.
- Rumelhart, D. E., & McClelland, J. L. (1982). An interactive activation model of context effects in letter perception: Part 2. The contextual enhancement effect and some tests and extensions of the model. *Psychological Review*, 89, 60-94.

HUMAN-MACHINE INTERACTION

USER CENTERED SYSTEM DESIGN

NEW PERSPECTIVES ON HUMAN-COMPUTER INTERACTION

The following is a list of chapters to appear in the above titled book edited by Donald A. Norman and Stephen W. Draper to be published by Lawrence Erlbaum Associates, Hillsdale, New Jersey.

Introduction

Stephen W. Draper and Donald A. Norman
(With a postscript by Clayton Lewis)

I User Centered System Design

- 1 Architectural Design: An Analogy
Kristina Hooper
- 2 Issues in Design: Some Notes
Liam J. Bannon
- 3 Cognitive Engineering
Donald A. Norman

II The Interface Experience

- 4 Interface as Mimesis
Brenda K. Laurel
- 5 Direct Manipulation Interfaces
Edwin Hutchins, James D. Hollan, and Donald A. Norman
- 6 Notes on the Future of Programming
Andrea A. diSessa

III Users' Understandings

- 7 User Understanding
Mary Riley
- 8 Understanding What's Happening in System Interactions
Clayton Lewis
- 9 Naive Theories of Computation
David Owen
- 10 Models of Computation
Andrea A. diSessa
- 11 Knowledge-Based Interface Design
William Mark

IV User Activities

- 12 The Structure of Users' Activities
Allen Cypher
- 13 Psychological Issues in Support of Multiple Activities
Yoshiro Miyata and Donald A. Norman
- 14 Communication Paradigms for Window Systems
Rachel Reichman (Adar)

V Toward a Pragmatics of Human-Machine Communication

- 15 There's More to Interaction Than Meets the Eye: Some Issues in Manual Input
William Buxton
- 16 Display Managers as the Basis for User-Machine Communication
Stephen W. Draper

VI Information Flow

- 17 Answers First, Then Questions
David Owen
- 18 Helping Users Help Themselves
Claire O'Malley
- 19 Helping Users Help One Another
Liam J. Bannon
- 20 Designing for Error
Clayton Lewis and Donald A. Norman
- 21 Computer-Mediated Communication
Liam J. Bannon

VII The Context of Computing

- 22 From Cognitive to Social Ergonomics and Beyond
John Seely Brown

Introduction

A Pluralistic Field, a Pluralistic Book

What is this field of Human-Computer Interaction? People are quite different from computers. This is hardly a novel observation, but whenever people use computers, there is necessarily a zone of mutual accommodation and this defines our area of interest. People are so adaptable that they are capable of shouldering the entire burden of accommodation to an artifact, but skillful designers make large parts of this burden vanish by adapting the artifact to its users. To understand successful design requires an understanding of the technology, the person, and their mutual interaction, and that is what this book is about.

The computer can be thought of from the perspective of its technology—from the field of computer science. Or it can be thought of as a social tool, a structure that will change social interaction and social policy, for better or for worse. It can be thought of as a personal assistant, where the goals and intentions of the user become of primary concern. It can be viewed from the experience of the user, a view that changes considerably with the task, the person, the design of the system. The field of human-computer interaction needs all these views, all these issues, and more besides. Studying these various perspectives can involve many disciplines: computer science, psychology, artificial intelligence, linguistics, anthropology, and sociology—the cognitive sciences. We are prepared to take on board any discipline, any approach that helps. It is a pluralistic field, so this is a pluralistic book.

A Book of Questions, Not of Answers

This is a book about the design of computers, but from the user's point of view: User Centered System Design. The emphasis is on people, rather than technology, although the powers and limits of contemporary machines are considered in order to know how to take that next step from today's limited machines toward more user-centered ones. This book is about the directions in which we must move to get there and does not follow traditional paradigms from any of the contributing disciplines. It is not a book on how to do things. It does not cover techniques and tools. Nor is it a book of fantasies, of possible dream worlds. Instead, it treads an intermediate path, neither detailed science nor flights of fancy. Think of it as a book of ideas, of analytical techniques described for their purpose, not for the details of their methods. Think of it as a book from which to derive the new directions in which we must move. Think of this as a book of questions, not a book of answers.

Some Possible Dimensions

This book is primarily an expression of a pluralistic approach, but if it has a common theme—a unity in its diversity—it is that human-computer interface design is not one small aspect of the main business of software design, nor will it be illuminated (let alone "solved") by a single methodology or technical innovation. To begin with, we do not wish to ask how to improve upon an interface to a program whose function and even implementation has already been decided. We wish to attempt User Centered System Design, to ask what the goals and needs of the users are, what tools they need, what kind of tasks they wish to perform, and what methods they would prefer to use. We would like to

The primary value of adopting an information flow perspective is that it causes us to step back from any prior assumptions about the pre-eminent importance of learning, of conceptual knowledge, or of the role of certain "official" channels such as manuals and tutorials. Instead the perspective encourages us to look at the amount of information vital to users that is picked up, used, and forgotten as well as at the importance of sources such as colleagues that do so much of the work supposedly done by manuals. Hence, the information flow perspective suggests that a designer's task in this area is first to calculate *what* information must be conveyed to users (the information content needed) and then to orchestrate the possible information sources to achieve this, perhaps designing special delivery systems. Some people promote slogans or codes of practice such as "good systems must have on-line help" and "always write a manual entry." Others have proposed rules for screen design, technical writing styles, and so on. Before we are ready to consider these assumptions and approaches to the effectiveness of particular information delivery systems, we should look at the broader perspective of the information flows actually important in current systems—both their content and the sources now supplying them.

A second aspect of the information flow perspective is to consider opportunities for replacing a requirement for users to know something by timely delivery of the needed information. In the simplest case, a menu might list available commands. However, there is also scope for aiding not just simple knowledge of items like names, but also understanding. By providing examples ("recipes") an action can be described to the user in such a way that it can be performed even if the user understands little about it. This is important whenever completion of an interaction is more important than education of the user. Thus, timely information delivery can stand in for memory, saving planning and understanding. This view can be developed by regarding the design of help systems as an attempt to design an extension of the user's memory.

Help Systems as an Extension of Memory

A simple but interesting viewpoint for considering help systems is to regard them as memory aids, for conserving the user's memory resources. Thus, we could consider all help systems as forming an extended external memory for the user—one kind of mind amplifier. From the cognitive point of view there is a spectrum of retrieval types. The best situation is when users can remember the item needed effortlessly. The next best is when they find the needed item already displayed. Third best is when they must ask. It is only third best because this counts as an interrupting activity, causing suspension of their current activity (see the discussions in Section IV of this book). This is the distinction made in the literature between active and passive help—between whether the user has to prompt delivery or not. Fourth best (i.e., worst) is when the user doesn't know how to ask. The activity requiring the information will have to be suspended indefinitely, and either a random search will ensue or the user must just wait to stumble on the answer. This is analogous to when you know you should know something—a name perhaps—but you just can't recall it. The information is there, but inaccessible.

The ideal, in fact, is not just to reduce the effort of seeking information but (subjectively) to abolish it—to achieve where possible the smooth flow of information. A familiar instance of this occurs in every display editor which revolves around the manipulation of the cursor's position. Users seldom note the subtask of retrieving the cursor position. Instead, they simply edit; moving the cursor to wherever it should go. Adequate information flows from the display at every moment. We should

what they know by an active use of external sources of information (Draper 1984). Thus information flow remains important for all users, although a shift is to be expected in the relative importance of different sources of information as a user gains experience: from a dependence on other people to a facility for interrogating manuals and source code, and for learning from experimentation.

There are two main topics for study here: identifying the information *contents* and identifying information *sources* or *delivery systems*.

Examples of kinds of information content:

- File names,
- Little recipes for the key strokes to get some neat or useful effect,
- Getting into a system or subsystem (e.g., how to invoke an editor)
- A minimum set of commands
- Advanced commands
- Where to get information
- Efficiency—neater ways to do certain things
- How things work
- Strategies for combining system programs (functions) for user tasks
- Recognizing information when it's there
- How to use information you have
- How to recover from errors
- What is commonly done on this system (i.e., standard user tasks and standard plans for accomplishing them)

Examples of information delivery systems:

- Printed manual entries
- On-line manual entries
- "Cheat Sheets" (summaries of commands)
- Menus
- Owen's "DYK" System ("Did You Know": See Chapter 17)
- People—replying to specific queries
- People—fortuitous pickup (looking over someone else's shoulder)
- Monitors that examine your input and periodically make suggestions about new commands, etc.

There are an unlimited number of possible delivery systems, since they can vary in who supplies the information (a public bulletin board versus system designer documentation), in whether the information is specifically requested or spontaneously generated (error messages, monitors), in the access structures (i.e., by command name, English description, etc.), in how much information about the user is consulted in selecting the presentation (none, or records of past requests, of commands vocabulary displayed in practice, etc.).

In Chapter 16, Draper begins by introducing the term "inter-referential I/O" as a generalization of the ways in which the inputs and outputs to a system can refer to each other. This allows for a common language and history between a user and a system, allowing both to refer to the same concepts readily and easily. These are critical aspects of the "first-personness" concept of Laurel (Chapter 4) and the Direct Engagement notion of Hutchins, Hollan, and Norman (Chapter 5). Draper suggests that systematic support for inter-referential I/O could have important benefits. First, it could provide better tools for support of user-machine dialogue. Second, it should allow for error correction and recovery through cooperative interaction, by allowing the user to elicit clarifications and explanations. These are points taken up by other chapters: The ability to interact gracefully with the system to take care of error is an important component of the discussion by Lewis and Norman in Chapter 20 and in the discussion of "repair" by Brown in Chapter 22.

Draper's chapter, like that by Buxton, again points out the interaction between apparently low-level details and high-level concepts. On one hand, the concept of inter-referential I/O is a low-level suggestion (keep the records needed to interpret pointing to objects on the screen and other references to previous I/O) but, on the other hand, it adds up to a transformation of the whole quality of the interaction and supports repair to the dialogue and smooth transitions between "levels."

These chapters can be seen as steps towards a pragmatics of human machine interaction. The notion however cries out for further exploration: There may be a whole perspective (not just hints) waiting to emerge on the "pragmatics" or "ethnomethodology" of human-computer communication that would explain what are now isolated, curious observations.

Section VI: Information Flow

Information Flow is meant to be a suggestive term, representing the flow of information that takes place among the systems, documents, and users of a system. At the least, the term "information flow" is used to emphasize the importance of a unified approach that embraces information sources other than manuals, including other people, system displays, all system messages, and prompts. In line with the emphasis of this book on new perspectives, the chapters in this section emphasize relatively neglected aspects of information flow.

Many issues in the field of Human-Computer Interaction revolve around the flow of information within the system comprising user(s), machine, and the relevant surroundings (which generally include printed manuals, phone lines to other users or experts, and colleagues near or far). On a fine scale information flow analysis concentrates on all the items of information that must reach users during the execution of their given goals in order for them to be achieved. On a broader scale, information flow analysis concentrates on all the flows of information that affect a user over time.

It is obvious that information flow analysis includes for all users attention to feedback, such as the echoing of characters and the movement of cursors, and, for novices, how they pick up all the necessary facts. With large and complex systems, even very experienced users know only a minority of the large set of commands: Their expertise lies, not in their having learned enough to immediately solve any problem, but in their having become skilled in gathering information and supplementing

Section V: Toward a Pragmatics of Human-Machine Communication

In linguistics it is traditional to separate the analysis of language into semantics and syntax. Those who study language, however, have long noted interesting phenomena that do not fall neatly under these divisions (and which are less amenable to formal analyses). These additional subject areas are known by various names, but the term "pragmatics" seems most common. Pragmatics addresses issues such as why the response "Yes" is not appropriate as an answer to the question "Can you pass the salt?" and why the response "I'm not wearing a watch" can be appropriate as an answer to the question "What is the time?" Analogous issues arise in human-computer communication, and so we suggest labeling them with the same term: *pragmatics*.¹

The two chapters in this section, in effect, deal with pragmatics, although neither mentions the term. Other chapters too are connected, most notably the previous chapter, Reichman's "Communication Paradigms for Window Systems," with its concern for how a person might signal shifts between activities smoothly and naturally. A characteristic of pragmatic phenomena is that, although they often are concerned with "large scale" structures such as user activities or conversation, they are typically expressed in "low-level" things such as mouse clicks, or words such as "but" and "hello." Don't be fooled: "But" and "hello" may seem non-entities syntactically and have insignificant meaning, but their role in structuring a conversation is crucial.

In Chapter 15, Buxton demonstrates that critical attention to the details of the input mechanisms are essential if we are to optimize interface design. He points out that current systems are designed with a paucity of input devices—often only a keyboard—thereby severely restricting their effectiveness. He goes on to discuss how even apparently small variations within a class of input device changes their effectiveness for different tasks considerably. The chapter might at first appear to be about the mechanics of input devices, but it deals with more than that. The chapter is really about the pragmatics, about subtle features that can seriously affect the nature of the interaction.

Buxton demonstrates quite convincingly that too little attention has been placed on the varieties of input devices (existing and potential) and on the critical details of their operation. He draws upon his considerable expertise as a professional musician (especially electronic music) to argue that we should realize that people have five major output mechanisms which could therefore use five input devices: two hands, two feet, and voice. There is no reason why we could not use all five simultaneously, and he suggests that at least we should develop systems that encourage use of the two hands simultaneously, each on a different form of device. (We could perhaps add a sixth—eye movements—but effective eye movement sensors are not yet practical.) Buxton restricts himself to simple kinds of inputs. He does not discuss the important variety of output devices that are possible, and the paucity of experimentation with sound as output (not speech—sound) is much to be regretted (see, for example, the PhD thesis by Bly (1982). Although Buxton is fully aware of the power of audio signals as data, he deliberately restricted the topics in this chapter to make his points: (a) We have seriously neglected the details and pragmatics of input devices; (b) Details do matter; and (c) We can do better, much better.

1. The term "pragmatics" has been proposed before in this connection (see Buxton, 1981) but the idea seems not to have yet gained a foothold.

to current window managers, would support users' new conceptualizations of the human-machine interface.

Technical Issues

All three chapters depend on the fact that many systems today make a start toward supporting multiple user activities. The most frequently discussed technique is the use of windows. However windows are not, technically, the heart of the matter.

One important issue is whether one user may have more than one interactive process active at once. Multiple computer jobs are clearly possible, for that is the essence of time-sharing, but with most of today's smaller, personal computers, only one interactive task can be run at a time. Multiple processing is possible on systems such as Berkeley UNIX that permit "suspending," "backgrounding," or "detaching" a process and then "resuming," "foregrounding," or "reattaching" to it later, thus allowing one user at a single terminal to alternate among several interactive processes without terminating and re-starting them. The great advantage is that the system context of each process is preserved. Windows add the preservation of the current output state of each process, which can then serve to preserve some of the user's mental context for the associated activity. Windows also serve as a visual reminder of the existence and identity of unfinished activities.

In the world of personal computers, windows face stiff competition from "integrated" systems, systems that achieve by other means the most important aims of a window system—to make it easy, for users whose complex tasks require multiple sub-activities, to switch rapidly and easily among multiple parts (programs) of the system, transferring the results of one system into the data structures of another. Whether the system has windows or not, the overall activity will only succeed if the two programs share a common representation with which to pass data back and forth—"integrated" systems stress this inter-program data transfer compatibility. Although windows are a leading technique in supporting multiple user activities, they are neither necessary nor sufficient, and, as discussed in Reichman's chapter, they are not yet supporting users' activities correctly.

From the computer science point of view, the two technical issues a designer must be concerned with in order to support multiple concurrent user activities on one system are, one, allowing the user to move between interactive programs without losing the context built up within each (this is especially important when the two programs are not part of the same user activity), and, second, ensuring that separate programs can be used together to pursue a single activity with an easy transfer of data among them. From the psychological point of view, the technical issues concern support for interruptions, reminders, and resumption of activities, all of which could be viewed as support for the natural and efficient saving and resumption of context. The chapters in this section provide valuable insights into the nature of real activities and into what will be required to support them in a smooth, even if tacit, manner.

To our knowledge, these aspects of real tasks have seldom been discussed, let alone supported (or, in the case of displacement activities, discouraged) by computer systems. The three chapters in this section discuss these aspects of real user activities and suggest several different ways in which computer systems might support the performance of multiple activities. Cypher discusses the situation from the computer's point of view, showing how system support could be provided. Miyata and Norman discuss the issues from the psychological point of view. And Reichman shows how the analogy to similar situations in conversation provides useful insight into the problem and shows how analyses of conversational gambits can suggest techniques for allowing people to switch among tasks and, perhaps more important, to resume where things left off with a minimum of disruption.

The three chapters in this section are motivated by the understanding that people usually have many things to do, some of which are related to each other and some which are not. Some tasks take such a long time that other things are also done along the way. Unexpected events occur frequently, interrupting on-going activities. As a result, normal activities are filled with simultaneous or overlapped tasks, with things deferred until later, or planned tasks awaiting an appropriate time to be done. These factors make behavior complex, even if each activity is relatively simple.

The basic phenomena of interleaved tasks and the basic modes of computer support for people's activity structures are introduced in Chapter 12 by Cypher, *The Structure of Users' Activities*. Cypher approaches the problem from the point of view of the user who is faced with the need to deal with multiple activities that simultaneously compete for attention. He treats the competing activities as interruptions of the ongoing activity and discusses ways to aid the user to transform the simultaneous demands into a linear sequence that can be dealt with one at a time. He discusses the varying ways in which activities may need to share contexts, and establishes in detail the point that people's computer usage is not, in general, in a simple one-to-one relationship with their activities. Good support for activities must therefore allow flexible groupings of processes. He then makes some proposals for activity support in interfaces, particularly in connection with the function of reminding.

The chapter by Miyata and Norman, Chapter 13, supplements the one by Cypher, expanding upon some of the issues and pointing out the relevance to some traditional studies within psychology. Cypher's analysis was based upon the activity flow to both the user and the computer. Miyata and Norman take the psychological point of view, centering upon what is going on inside the person. They focus more heavily on humans' abilities to pursue multiple activities and the effects of the relevant cognitive limitations. They thus develop a more detailed analysis of reminding, including the distinction between the signaling component and the descriptive component of a reminder, and hence arrive at proposals for supporting this in interfaces.

Reichman, in Chapter 14, analyzes how users use window systems in an attempt to support their multiple activities. She discusses the developmental trend of computer interfaces leading up to windows that has influenced users to expect more from their systems and to act as if they were having a two-party conversation with the computer. In conversation we switch topics, usually signaling this to the other person by simple clue words such as "well anyway," without recourse to explicit meta-linguistic references to topic. Reichman argues that interfaces need to achieve a parallel smoothness of use and that the lack of such mechanisms in existing systems shows up in various annoying and persistent errors. She then presents a set of contextual, navigational, and visual techniques that, if added

explore consequences of choices of concepts. It can furthermore support the interpretation of user inputs.

Bootstrapping

Many of the chapters in this section touch on the concept of *bootstrapping* when dealing with how new users can get started at acquiring understanding. The phrase "pulling yourself up by your own bootstraps" applies with some force to the problem facing new users. It is of course physically impossible to lift oneself by one's own bootstraps, and there is a corresponding philosophical problem of how one can come to know anything without first already knowing something. However, many of the problems that new users face hinge upon the fact that in order to understand the new concepts they have been exposed to, they need already to understand part of those concepts. All the chapters in this section are concerned with the kinds of understanding that users must and do acquire, but it is the consideration of new users that forces the issue of "bootstrapping." For instance, Owen and Lewis, in different ways, are both concerned with the way users interpret their experience of events (and "errors") at the interface. On the one hand, experience can lead to new understanding and a change in concepts. On the other hand, the interpretation of events and, hence, the possibility of new understanding, depends in part on the user's existing concepts. Owen suggests that the first seed of a new piece of understanding must be delivered "Answers first, then questions": The initial information must be delivered as an unsolicited display, because the user doesn't yet know enough to formulate and pursue precise questions.

Section IV: User Activities

The point of the three chapters in this section is that people interleave their activities—and do so in many different ways. Real tasks can take hours, weeks, or even months to perform. As a result, other tasks must of necessity intervene. Some tasks are so important they are allowed to interrupt others. Some tasks require the generation of subtasks, sometimes to satisfy prerequisite conditions, sometimes as corequisites, sometimes simply as relevant, but not necessary components. And then, some tasks are so onerous that people, sometimes deliberately, allow *displacement activities* to take over.

On a broad scale, people and computers often do several things at once: A person can cook dinner and carry on a conversation; A computer can execute multiple processes (time-sharing) while performing input-output operations on its peripherals. However both computers and humans use an important part of their capacities *linearizing*: carrying on several tasks by switching among them, so that at any instant only one of them is actually being done, even though on a broader scale they are all being done in parallel.

Linearizing carries with it a bookkeeping burden. Any activity carries with it a "context"—all the things that are particular to this instance of the activity. Each time an activity is suspended this context must be stored: Each time it is resumed, the context must be recalled. Thus, linearizing requires support for remembering and recalling. In addition, the fact that an activity is still waiting for completion must itself be remembered and recalled.

concerns how well the representation reflects the actual behavior of the system, and integration reflects to what extent is the representation tied to other components of a user's knowledge. Riley is concerned with the nature of the understanding that people form of the systems they are learning. A major issue is how to present the information so that it becomes understood—which is where the concepts of internal coherence, validity, and integration become important. Her other concern is with the robustness of the knowledge so formed, robust in the sense that it will not collapse in the face of unexpected outcomes.

The discussion by Lewis (Chapter 8) draws attention to the role of explanations and their construction in understanding and learning about a system. Lewis emphasizes false explanations, explanations that satisfy the learners that they truly understand and follow what is happening, but that in fact sometimes bear surprisingly little relationship to the truth. In some sense, Lewis shows what happens when the knowledge structures are *too* robust—so robust that they survive and prosper despite repeated evidence that they are erroneous.

Owen (Chapter 9) proposes that we should consider the idea of a "Naive Theory of Computation," analogous to the "naive physics" that underlies our understanding of physical events. This is another way of characterizing a kind of informal knowledge underlying a user's practical understanding. He also stresses that the problem of how to help users acquire it may be by making things visible: by "Answers First then Questions" (a suggestion he follows up in detail in Chapter 17) and by Direct Manipulation or Visible Computation design techniques that make a system self-revealing, rather than relying on users taking the initiative and asking for explanations.

diSessa (Chapter 10) offers a typology of the kinds of understanding users may have—"structural," "functional," and "distributed." He stresses that understanding is, in practice, often much less systematic than may be presupposed by researchers, and the examples he gives of each model type illustrate this. diSessa also shows that the common notion in Human-Computer Interaction of a "mental model" is much too simple: There are different kinds of models, each kind having quite different functional and computational properties.

Several chapters in the book talk about the need for a good, consistent design model and the need for it to be reflected in a consistent, coherent System Image (see especially Chapters 3 by Norman and 7 by Riley). Mark (Chapter 11) accepts this challenge, but points out that this consistency is difficult to achieve unaided, especially when the system to be designed is large and complex, involving a team of programmers working over a period measured in months or years. As a result, he proposes a design tool, one based upon modern knowledge representation techniques from Artificial Intelligence (the knowledge representation language KL-ONE by Brachman, 1978).

Mark starts from the assumption that a consistent system image is a good thing. He describes a system that helps programmers develop an explicit and consistent "system image model"—a formal model of the behavior of a system as observable from the user interface. The subsequent implementation is then constrained by requiring programmers to link each concept in the model to a specific software data object: The relationship between model concepts must hold between the software objects. The knowledge representation technique is used to define similarity of concepts as a basis for consistency in the interface, and its implementation gives a support system to allow designers to

We somehow have to separate the concepts of immediate visual feedback from other components of interactive computing. Visual programming and immediate representation are of crucial importance, but this is not all there is to the experience of directness. The nature of the interaction between one's thoughts and the structures available in the computing environment are also crucial. Hutchins, Hollan, and Norman speak of the Gulfs of Execution and Evaluation, gulfs comprised of the distances between mental concepts and the physical requirements and displays of the computational system. diSessa shows how different programming environments offer quite different tools and powers to the user. Surface appearance is only a small part of the story, although it is an essential component.

The phrase "direct manipulation" has become a catchword, connoting great ease of use, visual depiction of the events upon the screen, and specification of computer operation by moving icons around on a screen, perhaps supplemented by pop-up menus. But both chapters in the section that analyze the concept—Chapter 5 by Hutchins, Hollan, and Norman and Chapter 6 by diSessa—show this view to be simplistic. Hutchins, Hollan, and Norman point out that the concept of directness is complex, involving different concepts for the directness of specifying the action to be performed and the process of evaluating the result. Moreover, there is a directness between a person's intended meaning and the commands and displays available on the system ("semantic directness") and a directness of the actions that are to be performed and the input actions required or the depiction of the result on the displays ("articulatory directness"). Finally, the feeling of "direct engagement," the concept of which Laurel speaks and which, implicitly, is what is believed by many to be the hallmark of "direct manipulation," occurs only under special conditions, when both the Gulfs of Execution and Evaluation are bridged, and when there is inter-referential constancy between the objects manipulated at the input and viewed at the output, i.e., input and output representations are unified.

The nature of programming environments is further explored in the chapter by diSessa, who examines in detail the underlying principles of programming, from conventional languages to "device programming" and "direct manipulation." diSessa illustrates the way that programming languages borrow from and count upon analogies with natural language and experience, illustrating his point with reference to the language, Boxer, he has helped develop at MIT.

Section III: Users' Understandings

This section concerns the understanding users can have of a system. It starts with three chapters that attempt to characterize the *nature* of understanding relevant to a user of an interface: Chapters 7, 8, and 9 by Riley, Lewis, and Owen. Then we turn to diSessa and his focus upon the (non-computational) sources of knowledge brought to bear by users. He discusses three types of understanding: structural, functional, and distributed. These three types of understanding differ in their inherent nature and in the applications to which they mainly relate. We conclude the section with a chapter that applies ideas on the nature of understanding to interface design: the chapter by Mark. This is only one dimension underlying the topic however. Others include the origin of users's understanding, its nature, and how to support it in system design.

Riley (Chapter 7) argues that knowledge relevant to the understanding of a topic has three major components: internal coherence, validity, and integration. Internal coherence reflects how well the components of the representation that is to be learned are related in an integrated structure, validity

are precise enough to lead to design rules. The trick is to develop the engineering guidelines in such a way so as not to lose sight of the overall goals of cooperative, enjoyable interaction—of pleasurable engagement, to use the term coined by Laurel in Chapter 5. Norman's chapter introduces a wide range of topics that are dealt with further in subsequent chapters.

Section II: The Interface Experience

This section of the book contains chapters that get directly at the question of the quality of the user's experience. This is of course the ultimate criterion of User Centered System Design, but most workers approach it obliquely in various ways such as exploring the implementation techniques, or applying existing cognitive concepts. These chapters attempt more direct analyses. The spirit of the enterprise is expressed by the image of computer usage described by Laurel in her chapter: *Interface as Mimesis*. Laurel suggests we think of the computer as a stage, capable of letting the user experience a world. What we want is to figure out how to design things so that we, as users, can enter the world. When we read or watch a play or movie, we do not think of ourselves as interpreting light images, we become a part of the action: we imagine ourselves in the scenes being depicted. We have a "first person experience." So too with well constructed video arcade games and with simulation tools. Well, why not with computers?

In this section we start with Laurel's notion of *Mimesis*, a concept borrowed from that under-credited developer of computational concepts, Aristotle. Aristotle was a dreamer, but like proper dreamers, he provided deep analyses of his ideas in ways that are fruitful to consider today, a mere 2000 years later. Hutchins, Hollan, and Norman follow up with an analysis of a new form of interface, one that Ben Shneiderman has called "direct manipulation," an interface idea made practical by the folks at the Xerox Palo Alto Research Center, and made popular by the folks at Apple, in the Macintosh computer. Direct manipulation offers great promise of being a route toward Laurel's "first person experience," but it also provides traps for the unwary. The Hutchins, Hollan, and Norman chapter offers detailed analyses of what might really be going on. After that diSessa, speaking from his experience in the design of languages for children of all ages—first LOGO, then Boxer—discusses the future of programming, and the directions in which the programming world might move.

There have been major changes in interaction in computers in general and with programming in particular, with the advent of highly interactive, personal computers. One change came about naturally with the introduction of screen editors (text editors that provide automatic and full display of the contents of the text being manipulated). These editors instantly reveal the changes to the text, thus giving the illusion of working directly with the objects on the screen, an illusion that is more difficult to sustain with earlier generations of line or character-oriented editors. A similar revolution is occurring in the development of computer systems and in programming, perhaps most prominently with the programming conventions used in spreadsheets. Again, the hallmark is continual visual presence, a self-revealing display showing at all times the state of the system. Not very many programming systems have all these properties. Important examples are the Smalltalk programming system (Goldberg, 1984) and Boxer, the system described by diSessa.

As the title indicates, this book is a collection of a variety of new ideas about human-computer interaction: new interpretations of phenomena, ideas about phenomena in other areas that may be relevant here, and (yes) some ideas about widgets, or at least widgets-in-the-large, such as knowledge-representation technology. We don't think much of this will be of direct use in design, unlike wisdom about methods, rules, and widgets. Rather, we hope it will indicate new directions that design might take, or that experimental designs might explore.

We think this is important. Empirical methods and quantitative rules don't expand the design space in which we work (except when empirical study reveals a problem that we really are forced to solve). Riding the flow of widgets takes us into new territory, but not necessarily the territory we want to be in. We hope some of the ideas we discuss here will lead to new kinds of interfaces, kinds that are new in some way that we want.

These ideas are not intended to supplant the familiar wisdom. With some differences in balance, we all endorse all of those viewpoints on design, and the forms of knowledge that go with them. No one should design an interface without knowing the technology, without thinking about keystroke counts, or without planning for measurement and redesign. So this book is a supplement, not a substitute.

Clayton Lewis

Section I: User Centered System Design

Chapters 1 and 2 of this section raise some issues concerning the nature of design. The essay by Hooper (Chapter 1) draws a number of interesting parallels between the design of architectural and computational artifacts and searches for some lessons we might learn from the successes and, perhaps more important, the failures of the older design profession. The chapter does not provide a list of principles of design or even a set of guidelines for design that can be adopted by members of the human-computer interaction community, but it does show that "movements" and slogans in design can become overarching, with the result that the ultimate purpose of the design artifact is overlooked. Concern for the needs of the users should be primary.

The short note by Bannon (Chapter 2) expands on this issue, arguing for a socially conscious design process that is embedded in a full understanding of the users' needs. He looks at the interaction between the design and use of technology, stressing that artifacts can and do have an effect on the people who use or inhabit them and that building usable and useful systems requires that attention be paid to user needs at a variety of levels, from ergonomic concerns to organizational issues. The use of design as a social and cultural force is a contentious issue raised in both these chapters and returned to in the concluding chapter by Brown.

In the last chapter of this section, Norman discusses the applied side of Cognitive Science, what he calls *Cognitive Engineering*. He is concerned with putting together the parts of the formal science that will make a science of interface design possible. This requires, says Norman, some formal models of people and of interaction, models that need only be approximations of the ultimate theory, but that

and the interactions between users (some of the chapters on information flow) are considered. Finally, we finish by considering computer tools as they fit into and affect the social structures within which users live and work—the last chapter of Section VI by Bannon and the concluding chapter of the book, the chapter by Brown in Section VII.

Stephen W. Draper
Donald A. Norman

Postscript

A user interface functions at the intersection of many different kinds of things: people, machines, tasks, groups of people, groups of machines, and more. These different things contribute diverse constraints and opportunities to the design process. None of them, not even machines, is captured by adequate theories, that is, theories adequate to tell the designer what's going to happen when a design is put to work. Two results of this state of affairs are that design is very hard, and that there are many ideas about how it should be approached.

One view, argued by Gould and Lewis (1985), is that design must be treated as fundamentally empirical. Designers must work hard to learn as much as possible about the users of the system and the work they will do with it. They must assume that their initial design ideas, even given this background information, will be wrong, and plan for repeated redesign. They must base these redesigns on empirical measurements of the success of the design, made on actual use of an implementation, a prototype, or mockup.

A second view, represented in the influential work of Card, Moran, and Newell (1983), is that the chaos of interface design can be reduced over time to a kind of approximate order, in which quantitative rules are discovered that make useful predictions about the good and bad features of potential design.

A third view, perhaps never stated but implicit in most real design work, is that interface design is driven by technology. Improvements in interfaces stem from improvements in the size and quality of displays, or the invention of new widgets for moving cursors, or new ways to arrange information on the screen. Designers should keep up with the news and think about how to exploit these new possibilities.

Each of these viewpoints has its corresponding design wisdom. Empirical design needs methods: ways to learn about users, ways to detect, measure, and diagnose the problems in a design. Card, Moran, and Newell want those quantitative rules. Widgeteers want to know about new widgets. But this book supplies very little of any of this.

intentions and thoughts. On a larger scale are the chapters on users' activity structures and on the nature of interaction among people, between people, between people and systems, and systems and society.

These different kinds of analysis are often associated with yet another cause for multiple approaches: differences in methodology, for example in the use of experimental techniques, of implementing trial systems, observing and questioning users, or simply doing theoretical analyses. For instance, Mark's chapter revolves around an implementation, whereas that by Miyata and Norman involves applying existing work in psychology to the concerns of Human-Computer Interaction. But the chapters do not stress methodology, and we rejected the idea of organizing by methodological dimensions.

The Structure of the Book

Following the chapters in Section I that survey the field as a whole, we jump to what we see as medium scale approaches. This is Section II, "The Interface Experience." The studies in this section are traditional in that they focus on one user, one task, and one machine, but their emphasis on the subjective experience of the user is not conventional. This section deals with a phenomenologically oriented approach that directly asks the ultimately central question "what is the experience like for the user?" In the end, that is the basic question underlying all user-centered design. However, asking it does not necessarily or directly lead either to useful theory or to practical design methods which is why only a few chapters take this approach.

The next section of the book, Section III on "Users' Understandings," we move to a finer scale. One approach in this section is to start with a design or implementation technique and pursue its consequences which, of course, one hopes will ultimately have beneficial effects for the user. For instance, in his chapter, Mark describes an aid for designers that ensures a consistent use of concepts in an interface. That this will benefit users is assumed rather than directly investigated in that work. Yet another kind of approach in this section offers various ideas about the nature and content of how a user comes to understand a system. The hope, of course, is that better theories will lead to more informed and hence better interface design, but again that is an indirect consequence. This section of the book in fact shows still further diversity in the kinds of argument found to be interesting: Whereas some authors (e.g., Riley) concentrate on the kinds of content users' understandings can have, diSessa concentrates on the different origins that understanding can have (e.g., loose analogies with natural language, as opposed to a detailed "functional" model from a different instantiation of the same task domain).

We next move to the larger scale chapters, going from an analysis of user activity structures (Section IV), to the pragmatic aspects of human-machine interaction (Section V), to the considerations of the flow of information among people and between person and systems (Section VI), and then to the societal and cultural impact of design (Section VII). This approach has let us zoom progressively outward to larger and larger scales in which multiple tasks are considered (the section on user activities)

start with the users, and to work from there.

Granted the premise of User Centered System Design, though, what follows? The more we study it, the bigger the subject seems to become. Pluralism is the result of the piecemeal recognition of more and more important aspects to the subject. We are at the point (in the mid 1980s) of realizing just how much bigger the problem is than has usually been acknowledged, but we are not within sight of a grand synthesis or a unifying theory. This book offers "perspectives"—pluralistic voices laying claim to your attention. The authors contributing to this book interacted to a considerable extent during its writing. As a result, many mutual connections have been found and are mentioned in the chapters, but nothing like a single synthesis has yet been constructed. The main message remains that pluralism is necessary and appropriate at this stage of the field. The chapters reflect this pluralism implicitly, not by design.

This book has been difficult to organize, for the essence of pluralism is that there is no single *best* approach, no single dominant dimension, no single approach that fully organizes the issues to be faced or the solutions and perspectives that are offered.

One approach, a rather traditional one, is to start with considerations of the person, the study of the Human Information Processing Structures, and from this to develop the appropriate dimensions of the user interface. Many of the chapters in this book can be considered to be developments of this approach.

Another approach is to examine the subjective experience of the user and how it might be enhanced. When we read or watch a play or movie, we do not think of ourselves as interpreting light images, we become a part of the action: We imagine ourselves in the scenes being depicted. We have a "first person experience." So too with well-constructed video arcade games and with simulation tools. Well, why not with computers? The ideal associated with this approach is the feeling of "direct engagement," the feeling that the computer is invisible, not even there, but rather what is present is the world we are exploring, be that world music, art, words, business, mathematics, literature—whatever your imagination and task provide you.

A different approach would be to focus upon the social context of computing, on the fact that computers are meant to be tools for carrying out tasks, tasks that are normally done in association with (or for the benefit of) other people. This approach focuses upon social interaction, upon the nature of the workplace, about the kinds of assistance people get from one another rather than from manuals or formal instruction, and on the ways that we might aid and assist this process. This is a perspective in which the issue is the use that can be made of interfaces in designing human work, jobs, offices, social interactions—even society itself—i.e., a complete subordination of interface design to social concerns.

All of the above approaches are included in this book. After considering a half dozen ways of organizing the chapters we decided to group the chapters in terms of the scale or grain of analysis adopted. We see approaches that focus on one user pursuing one task on one machine, as representing only one point on a range of possible scales of analysis. On a finer scale we have studies that examine hand motions, keystrokes, and the information-processing structures of actions and perceptions,

aim for a similar subjective effect in delivering all information. Currently not enough is understood to tackle this systematically.

The cursor example depends on the display being constantly updated without explicit commands from the user—the technique of *autodisplay*. Autodisplay poses two major problems: what to display, and whether the user will pick up the information displayed. The compiler example illustrates the first problem. Since error codes are useful only as pointers to other better descriptions of the problem, it is a safe bet to print the latter. On the other hand automatically displaying a piece of source code or a possible diagnosis is only as good as the reliability of the compiler at knowing which line of the code needs to be changed or knowing that a set of symptoms has only one possible cause. The second problem may be called "receptivity"—whether the user will pick up relevant information if it is displayed. One of us once watched the other discover that half his files seemed to be missing. It took several minutes before someone else pointed out that he was logged on to a different machine than he thought. Yet all the time the name of the machine was being displayed in every prompt, only three character widths to the left of the fixation point as he frantically typed commands. The machine name was not noticed, even though the user had himself designed that prompt with exactly this sort of problem in mind. This is a common kind of problem: failing to see a solution that is literally staring us in the face, even when we are actively searching for a solution. When the information presented does not fit the current hypothesis it may be ignored. Until we understand what governs receptivity a lot better, designing information flow will be a patchy business at best, and it will be hard to design information delivery systems that make users feel that the system's help facilities are an extension of their own memories.

The Chapters

Owen, in Chapter 17, reminds us that we often learn by first discovering information, then determining what question it answers—the inverse of the what is thought to be the standard method of seeking information. Owen calls this approach "*answers first, then questions*." The point is that on many occasions we come across an idea or piece of information more or less by chance, and then recognize it as interesting or relevant in some way: i.e., an answer to a dormant question. He illustrates this mode of information flow in everyday life and presents a fantasy of one style of advanced graphics-intensive user interface. He describes the exploration of this method in an experimental facility he calls DYK ("Did you Know?") DYK offers assorted factual tidbits on demand, providing a novel, but quite effective, method of conveying some kinds of information to the user population. (Furthermore his DYK facility is fun to use, making it an attractive displacement activity as well as a valuable source of information about computer usage.)

Another reason for the importance of "answers first, then questions" as an information delivery mode emerges from considering a tacit assumption behind most conventional help systems. The most common way to store and present information follows a database model: all the stored items have the same structure but different details. An example is the telephone directory: All entries (items) show a name, an address, and a phone number. Computer databases follow this rule, with standardized formats for its records. Computer help systems and manuals also usually have this form, with one entry per command and various standard parts to each entry.

To ask a question of such systems—to formulate a query—enquirers must know how to transform their original need for information into queries the system can respond to. This requires a knowledge of the query language syntax, of the structure of the database, and of the indexing system. For instance, to find a phone number you must know the person's name: in the case of standard telephone directories, knowing only the address will not work because they are indexed by name. A first improvement of retrieval systems is to provide more and different indices to the same database—for instance, computer system manuals can be indexed by function as well as by command name. These improvements, however, do not overcome the problem that the enquirer must know the standard structure of the information and the legal queries, in order to use the index or phone directory. (We should remember also that many people prefer to ask another rather than to use a directory themselves—phone companies mount active campaigns to move their customers away from asking information operators for help.)

O'Malley (Chapter 18) concentrates on the more usual case when the user initiates a query to the system: *"questions first, then answers."* She asks why users so often prefer to ask other people for help rather than to use manuals. She suggests that a large part of the answer stems from the fact that to formulate a question for such systems, you already need to know many things.² This means that asking a question is in principle a multi-step process. Failure in any step means failure of the whole. The apparent superiority of human help is often due to its robustness—the people one consults can aid in filling in the missing steps and in reformulating the question in an appropriate fashion. Thus in both measuring the success of a delivery system, and in designing improved ones, we need to identify all the steps and ensure that they are all supported by one means or another. It is not enough to provide answers, we must also support the formulation of questions.

Bannon focuses upon a relatively neglected aspect of information flow in human-machine interaction—the role of social interactions. In his first chapter in this section, Chapter 19, Bannon focuses on user-user information flows, arguing that they will always be an important part of the total pattern of information flow within the larger system of computers plus user community, despite present and future advances in system-to-user delivery methods. He suggests that facilities could and should be installed on computers to facilitate (not to replace) user-to-user flows—on-line human help, as it were. This chapter is supplementary to the one by O'Malley, for it argues that human assistance is such a pervasive aspect of information finding and computer usage that systems ought to be designed with this in mind, to supplement and to extend the ability to get help from others.

Lewis and Norman (Chapter 20) discuss the possible responses of a system when it detects a problem: a class of system-to-user information flow, initiated by the system. This chapter is a product of the intensive workshop we held to review draft chapters of the book. There, Lewis and Norman discovered their common interests in the manner that contemporary systems handle cases where the system cannot interpret the information given to it. "Error," says the system: "naughty user." In fact, the error is just as much on the part of the system for failure to understand as it is on the part of the user. The main point is that there is a breakdown in the pattern of information flow. In normal conversation, as Lewis and Norman point out listener and speaker do not jump to assign blame for a failure to

2. As Miyake and Norman (1979) once put it in the title of their paper: "To Ask a Question, One Must Know Enough to Know What Is Not Known."

understand. Instead, the listener simply does the best job possible at forming an appropriate interpretation and seeks help when that is not possible. Lewis and Norman discuss the range of responses open to the system and its designers to errors by the user. (These are points raised again in Chapter 22 by Brown).

In the last chapter of this section (Chapter 21), Bannon again focuses on user-user information flows, but in a different sense than in his earlier chapter: not as part of information flow *about* the system, supplementing the documentation, but as an end in itself, for which the computer merely acts as a medium. He reviews some of the most common forms of computer-mediated communication (e.g., electronic mail and computer conferencing), and then some ways of distinguishing the effects the media have on the human interactions they support. He also shows how one might design an office by deliberately selecting some set of communication tools to fulfill the needs of the office members. This chapter adopts the largest scale of any in this section: that of augmenting and shaping human interactions and environments by the selection of technology. Bannon reminds us of the social component and consequences of human-computer interaction and discusses ways that the computer can be used as a tool to improve and enhance social interaction. All too often today the computer is thought of as an individual tool, used by a person in relative isolation from all others. But just the reverse is really the case. First, we need other people in order to learn how to use computers most effectively (the point of almost all the chapters in this section). Second, computers provide enhanced communication abilities, both synchronous and asynchronous. And third, the computer is a new tool that can enhance participatory work, allowing for shared workspaces, and shared communication channels, so that our ideas can immediately be made available to the others with whom we work, allowing for increased interaction and creativity.

Section VII: The Context of Computing

This final section has only one chapter, one that in part functions as a sort of summary statement. We organized the book by "grain size," by the level at which the topics were approached. We think it only fitting that we conclude the book at the largest grain size, with a chapter that considers the way one might use computational tools to affect society, to change the tasks that we do and the way that we view them. This section, and the single chapter within it, address many of the issues raised in the book, but with the explicit attempt to exploit them by designing tools for society, tools that could change thinking, problem solving, writing, and communication. A good way to put this chapter into perspective is by reviewing the entire book, commenting upon how Brown uses the points developed in the book in his far-ranging analysis of social settings and in his suggestions for social tools.

Section I provides an overview of *User Centered System Design*. Hooper (Chapter 1) reviews the design philosophy of architecture, suggesting that we in Human-Computer Interaction can learn from the history of that field. Bannon (Chapter 2) raises several issues concerning the role of the designer in society and the need for a responsible design. In Chapter 22, Brown continues this discussion. He suggests that the Bauhaus School in architecture, the influential design school in Germany just before the second world war that took social change as an essential theme of design, might provide a good model for our field as well. The Bauhaus school and its impact was raised and discussed by both Hooper and Bannon, but not necessarily with great favor. Brown asks "how designers who take up this

challenge might form a movement akin to the Bauhaus School of architecture, in which physical artifacts were designed explicitly to facilitate social change." This is both an exciting challenge and a major problem. But as the experience with the Pruitt Igoe Housing Project in St. Louis vividly points out (see Chapters 1 and 2), it is dangerous to make social policy statements prematurely. We certainly do not know enough about social factors to understand the full implications of the artifacts that we create. Bannon summarized the issue this way (in an electronic mail interchange among Bannon, Brown, Hooper, and Norman on this topic):

I am still struggling with the dilemma of how to reconcile the role of the designer as both creator of artifacts that can effect positive change in society and the obverse side of this, which is where the designer creates artifacts that, though designed with "good intentions," are totally unsuited to the needs of people.³

A recurring debate throughout the book is the view of the computer as a "tool." Norman (Chapter 3) makes this one of his themes, that the computer should be viewed as a tool, that it should serve in the interests of the user, neither dictating the work nor taking control, but rather acting together with the user to get better mastery of the task domain. Laurel, in Chapter 4, argues against the notion of computer as tool, but using "tool" in a different sense, as a intermediary that might stand in the way between a person and the task or experience. These two points of view are reconciled if the tool is invisible, transparent, so that the user notices the product and the task, not the tool being used to get there—a point of view consistent with the arguments for Direct Manipulation in Chapter 5. Brown comes down strongly in favor of the computer-as-tool view, and he shows that the proper tools can change one's perception of the task. Indeed, he argues that the best way to induce social change in a large design community is to provide tools carefully crafted to affect the way designers think of their task. (Brown uses as an example the design of copiers in his own organization, but the clear intent is to generalize to design in general, in much larger contexts).

An interesting commentary on the role of computer-based tools in society and how they might be crafted with concern for the social implications comes from the Scandinavian experience in constructing tools for newspaper composition, using high-quality video displays. These tools are being designed by an innovative design institute, one partially funded by Labor Unions and Corporations: Norman, Chapter 3, reviews briefly some of their work. (See Ehn & Kyng, 1984.) A brief overview of the entire project can be found in Howard, 1985. It turns out that even the intense concern for social impact and explicit support by labor unions during development did not prevent difficulties when the systems were actually installed. This emphasizes the point that we should be very cautious in making social commentaries or in introducing technological artifacts designed for a particular social purpose: We risk tampering with the complexities of a delicate cultural process with insufficient knowledge and experience.

3. Those interested in pursuing these issues might start with the several chapters in this book that discuss social change, and might reflect upon Tom Wolfe's (1981) treatment of the lessons from Architecture or upon Giedion's general critique of architecture (1965).

"User Understanding," the theme of Section III of this book, provides a major theme for Brown in this chapter. He argues strongly that users form mental models of the systems with which they interact, and he echoes Norman's concern from Chapter 3 that the system itself provide the necessary information to support the development of an effective model (what Norman called the *System Image*). Here, Brown builds upon the work in Cognitive Science on understanding (reviewed by Riley, Chapter 7), along with the discussion by Lewis (Chapter 8) of the way that people find explanations of their experiences. Brown argues strongly that people need to construct explanations of the systems that they use, that they build upon a "naive theory of computation" (in the sense described by Owen, Chapter 9), and that these play major roles in the way that people recover from difficulties—"repair" the model and procedures being used. Here he borrows heavily from the experiences of natural language, where repair and communication take place continually and unobtrusively, points that serve as the themes for Chapters 14 and 20 by Reichman and Lewis and Norman.

The concept of "Information Flow"—the theme of Section VI—is central to Brown's chapter, so much so that we debated adding this chapter to that section. In many ways that is what the chapter is about: the flow of information between person and computer system, between people, among social groups and organizations. Brown takes the strong position that most tasks are interactive, that we need more tools to support this interaction, and that technology now offers a way to provide these tools. Design, writing, communicating—all are areas that can benefit from enhanced tools. Tools as artifact, not only of technology, but of society and culture. The theme that users must help one another is dominant here, reflecting the arguments made by Owen, O'Malley, and Bannon (Chapters 17, 18, 19, and 21). Brown points out the differences that occur when learning is done in isolation as opposed to with other people. "Eye contact" is critical for Brown, and he illustrates a case in which failure to be within sight of others learning a system was detrimental for the learner. As Bannon points out, in the design of computer (and other) systems, the social organization of work, and even the physical layout of the office may be as important as the manner by which the system was programmed.

Brown's chapter—and therefore this review—does not touch upon all of the book. Regardless of how you feel about his discussion, we are certain that it will be provocative, raising important issues about the role of our field within the context of computing, indeed, within the context of society. It is a fitting way to close the book.

References

- Brachman, R. (1978). *A structural paradigm for representing knowledge* (Tech. Rep. 3605). Cambridge, MA: Bolt, Beranek, & Newman.
- Ehn, P., & Kyng, M. (1984). *A tool perspective on design of interactive computer support for skilled workers*. Unpublished manuscript, Swedish Center for Working Life, Stockholm.
- Giedion, S. (1965). *Space, time, and architecture* (4th ed.). Cambridge, MA: Harvard University Press.

Goldberg, A. (1984). *Smalltalk-80: The interactive programming environment*. Reading, MA: Addison-Wesley.

Howard, R. (1985, April). UTOPIA: Where workers craft new technology. *Technology Review*, pp. 42-49.

Miyake, N., & Norman, D. A. (1979). To ask a question, one must know enough to know what is not known. *Journal of Verbal Learning and Verbal Behavior*, 18, 357-364.

Wolfe, T. (1981). *From Bauhaus to our house*. New York: Washington Square Press.

SKILLED MOTOR PERFORMANCE AT VARIABLE RATES¹

Donald R. Gentner

Motor skills can be performed with a range of overall durations, and a detailed study of performance at different rates provides information about the representation and execution of motor skills. An analysis of data from a variety of motor skills reported in the literature and a detailed study of skilled typewriting shows that a central, generalized motor program with a multiplicative rate parameter usually does not fit observed performance. A multi-level control model, in which performance is the composite product of central and peripheral control, provides a better account of performance data.

Skilled motor performance is based on a combination of innate capabilities and learning in the environment. Actions such as walking, for which relatively little learning is required to produce competent behavior, lie at one extreme of a continuum. At the other extreme of the continuum are activities such as playing a violin or flying an airplane, which, although capitalizing on existing motor capabilities, require hundreds of hours of learning to reach expert performance. In all cases, however, flexibility is a striking characteristic of expert performance. Experts are able to modify their actions to accommodate their intentions and the changing task demands. For example, a basketball player does not learn to shoot baskets from just a few locations, but instead is able to shoot from anywhere near the basket, from standing and jumping positions, around defending players, and so forth. Thus, the skills that a basketball player acquires can be applied in varied and novel situations.

The neural bases of motor skills are usually described in terms of a motor program, but this concept has been given widely differing meanings over the past fifty years. The concept originally was originally used action sequences as centrally controlled patterns that functioned with only minor involvement of sensory input, in contrast with the prevailing view of action sequences as stimulus-response reflex chains (Lashley, 1951; Keele, 1968). The view that motor programs made little use of sensory input was given support by the finding that animals could make the coordinated rhythmic leg movements typical of locomotion even when their spinal cords and sensory nerves were cut (see Grillner, 1985 for a review). It is now generally recognized, however, that normal motor behavior is based on a collaboration of perceptual, cognitive and memory processes in the brain, reflexes and pattern generators in the spinal cord, and sensory input (see, for example, Keele & Summers, 1976; Prinz & Sanders, 1984; Schmidt, Zelaznik, Hawkins, Frank, & Quinn, 1979). For the purposes of this discussion, the

1. The ideas in this paper have developed over several years, and have benefited greatly from conversations with numerous colleagues. I especially wish to thank Ian Abramson, John Donald, Mike Jordan, Ronald Knoll, Donald Norman, David Rumelhart, Richard Schmidt, Diane Shapiro, Saul Sternberg, Hans-Leo Teulings, Paulo Viviani, and Alan Wing.

concept of a motor program will refer to the centrally stored representations used in the performance of action sequences.

The simple concept of a central motor program, however, is not sufficient to account for the observed flexibility of motor performance. It seems implausible that a separate motor program would be stored for every variation of the action, and there would also be no way to perform novel variations of an action. The generalized motor program provides a direct account of flexibility in performance.

The Generalized Motor Program

The generalized motor program model was originally proposed by Schmidt (1975) and has since been discussed by numerous authors (Carter & Shapiro, 1984; Frohlich & Elliott, 1984; Klein, Levy, and McCabe, 1984; Schmidt, 1982; Shapiro & Schmidt, 1982). A generalized motor program is roughly analogous to a computer program. Just as a computer program can produce different outputs when it is invoked with different parameter values, a generalized motor program also has variable parameters and motor performance will vary depending on the values of the motor program parameters. It is generally assumed that there is a direct relation between a parameter of the motor program and a feature of the observed behavior. Evidence for a generalized motor program can be obtained, therefore, by observing behavior under a variety of conditions, and searching for patterns in the data that can be related to an underlying parameter in the motor program. An alternative strategy is to search for invariances in behavior, which would indicate those aspects of a movement that are not the product of variable parameters.

The most frequently cited candidates for parameters to the generalized motor program are overall duration (or rate) and force (Schmidt, 1984). For example, Carter and Shapiro (1984) trained subjects to perform a series of four wrist movements with a total duration of about 600 msec. Then they asked the subjects to perform the movements as fast as possible. They reported that the overall duration for the four movements decreased by about 100 msec for the fast trials, and that the durations of the individual movements all decreased by the same ratio. The finding that the individual movement durations maintained a constant ratio was cited as evidence for a generalized motor program with a multiplicative rate parameter. Similarly, Hollerbach (1981) found that when one subject wrote the word *hell* in either large or small script, the timing of vertical accelerations was similar, suggesting that force of movement could be varied as an independent parameter.

Test for a Multiplicative Rate Parameter

The most widely cited evidence for a generalized motor program comes from studies of how action sequences change with changes in overall duration (Schmidt, 1982, p. 311). If the timing of an action sequence is determined by a generalized motor program with a multiplicative rate parameter, then the durations of all the components of the sequence should maintain a constant proportion as the overall duration of the sequence changes. For convenience, I will refer to this model of a generalized motor program with a multiplicative rate parameter as the proportional duration model. This paper is intended to be a critical review of the evidence for a proportional duration model. First, I present a statistical test for the presence of a multiplicative rate parameter. This test is then used to examine the evidence for a proportional duration model cited in the literature and to test performance

data collected from expert typists.

A Simple Mathematical Model

To make the argument clearer, consider an action sequence that can be decomposed into a set of components. For example, the movement of a leg during one walking cycle can be decomposed into a support component when the foot is touching the ground, and a swing component when the leg is swinging forward. As the total duration of the step cycle increases, the duration of the support and swing components will probably change also. The prediction of the proportional duration model is that, even though all the durations may change, the duration of each component will remain a constant proportion of the total duration. For example, if the swing component lasts 0.4 s when the total cycle duration is 1.0 s, the swing component should last 0.6 s when the total cycle duration is 1.5 s. Specifically, with a multiplicative rate parameter, the ratio d_i/T should be constant over all instances of the action sequence, where d_i is the duration of the i th component and T is the total duration of the sequence.

Problems With Existing Tests of Multiplicative Rate Parameter

Although a number of published reports have attempted to examine behavior in relation to a multiplicative rate parameter, a critical test of the theory has not emerged thus far. There are two main weaknesses in published reports.

First, the data examined are often averaged over instances and over subjects. The proportional duration model describes the timing of individual action sequences. It is true that if the proportional duration model holds for all the individual instances, it will also hold for the averaged data, but the reverse case is not necessarily true. The practice of averaging over subjects is especially problematical. The generalized motor program is necessarily specific to an individual, and data that has been averaged over subjects cannot be directly related to the model.

Second, in the cases where individual performances were examined, the papers present only a very limited number of examples. What is needed is a statistical method that permits analysis of individual sequences, and allows summary of these analyses over many different sequences and subjects.

The Constant Proportion Test

The constant proportion test is a simple and direct test of the proportional duration model. The basic idea behind the test is that although the total duration for an action sequence may change, the proportion of time occupied by a given component should remain constant. Therefore, if d_i is the duration of the i th component of an action sequence and T is the total duration of the sequence, then when d_i/T is plotted against T , the corresponding linear regression line should have a slope of zero. The constant proportion test thus simply consists of determining whether the slope of the linear regression is significantly different from zero for d_i/T versus T . When a large number of intervals are being examined, the results of the test can be conveniently summarized over many such linear regression analyses. Specifically, if the criterion of significance is taken to be $p < .05$, then in a large series of such analyses we would expect only about 5% of the linear regression slopes to be significantly

different from zero if the proportional duration model is valid.

Test Results With Simulated Data

The constant proportion test was verified by using it to analyze simulated data generated according to two simple models of timing. Data for the multiplicative rate parameter model were generated by the equation:

$$d_{in} = r_n D_i + e_{in} \quad (1)$$

Data for the additive rate parameter model were generated by the equation:

$$d_{in} = D_i + r_n + e_{in} \quad (2)$$

In these expressions, d_{in} is the observed duration of the i th component in the n th instance of the sequence; r_n is a normally distributed rate parameter, varying for each instance but constant for all components within an instance; D_i is the mean duration of the i th component; and e_{in} is a normally distributed random error, differing for each component and instance of the sequence. Each simulated instance consisted of five intervals, and groups of twenty instances were analyzed with the constant proportion test.

The results are summarized in Table 1. The observed variability in durations is the resultant of

Table 1

Analyzing simulated data with the Constant Proportion Test

Proportion of standard deviation		Multiplicative rejection rate	Additive rejection rate
Rate parameter	Random error		
90%	10%	5.0%	80.8%
75%	25%	5.0%	69.2%
50%	50%	6.6%	34.0%
33%	67%	7.8%	16.8%
25%	75%	8.2%	12.6%
10%	90%	9.2%	13.4%
0%	100%	11.0%	8.2%

Note. Simulated data were generated for 20 instances of a 5 component sequence. Each test was repeated 100 times.

variability from the changing rate parameter and variability from the random error term. Each value in the table is based on 100 repetitions of the constant proportion test on a set of 20 instances. Figure 1

shows one example of the test with data generated by the additive model (Equation 2). Table 1 shows how the relative contributions to the observed variability affects the test rejection rate. With the multiplicative model (Equation 1), the rejection rates are all near the expected level of 5%. With the additive model (Equation 2), the rejection rates are very high when most of the variability comes from the rate parameter term, but the rejection rates approach the 5% chance level when most of the variability comes from the random error term.

There are several technical issues that should be mentioned with respect to the constant proportion test. First, as shown in Table 1, when the variability in durations is entirely due to the random error term, the rejection rate is approximately 10%, rather than the expected 5%. The increased rejection rate is caused by the fact that the total duration is the sum of the component durations and the slope of the linear regression line is not exactly zero. Because the random error term has the same standard deviation for each interval in the sequence, short intervals will have relatively larger variability than longer intervals. Thus the relative duration of short intervals will increase with overall duration, whereas the relative duration of long intervals will decrease. The exact rejection rate obtained with the constant proportion test depends on the relative mean duration of the component intervals and the number of instances sampled. The mean durations (100,150,200,75,130) and sample size (20) used for the results in Table 1 were chosen to be representative of the typing data presented later in this paper.

The second technical issue concerns whether the constant proportion test should be applied to intervals between components, or to times measured from the start of the sequence. The constant proportion test will work with either intervals or times; the choice should depend on the details of the timing model being tested. A serial model of timing would generally indicate use of intervals, whereas a parallel model would generally indicate use of times. The typing data discussed later were tested using both intervals and times. The rejection rate was slightly higher with times than with intervals. This probably reflects an earlier finding that typing data fit a serial model of timing better than a parallel model of timing (Gentner, 1982). Also, the relative durations of the intervals in an action sequence are not completely independent, because they must add up to one, and the slopes from the constant proportion test must add up to zero.

Literature Review

This section is a critical review of data in the literature that are relevant to the proportional duration model. Whenever practical, I have analyzed these data with the constant proportion test. The first section examines laboratory tasks in which the subjects practiced the task for a few hours. Next, studies of animal and human locomotion are reviewed. The final section takes up other highly practiced tasks, typically based on thousands of hours of practice.

Laboratory Tasks

Arm movements. Several experimenters have studied arm movements in the laboratory. One of the original suggestions for a multiplicative rate parameter came from the work of Armstrong (1970). Armstrong had subjects move a lever with a series of elbow flexions and extensions, while attempting to match a target pattern of displacement versus time. After four days of practice, subjects became fairly good at this task, but the interesting finding was that timing errors within an individual trial

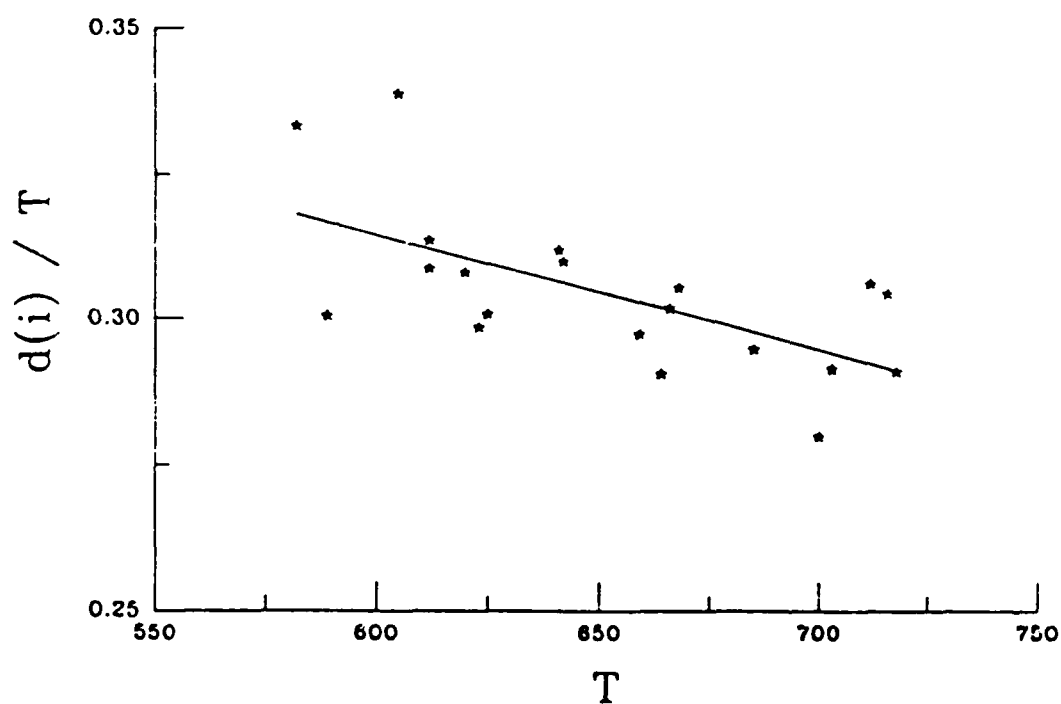


Figure 1. The constant proportion test with data generated by the additive model (Equation 2). An additive rate parameter and random error contribute equally to variability in the duration. In this case the slope of the linear regression line was significantly different from zero.

appeared to be related. In particular, Armstrong found that the time from the start of the movement to the second displacement peak was highly correlated with the time from the start of the movement to the third displacement peak. The correlation was .88 in the first experiment with a 3 s movement (Figure 8 in Armstrong, 1970, p 25) and .79 in a second experiment with a 4 s movement (Figure 13 in Armstrong, 1970, p 34). Of course, we would expect some correlation between these two times, because the time to reach the third peak includes the time to reach the second peak. I analyzed Armstrong's data to determine if these positive correlations reflected more than the common time interval. The first approach was to assume that the time to the third peak represented a total duration, and use the constant proportion test to determine if the relative time to the second peak is constant. In Armstrong's first experiment, the relative time to the second peak increased from 53% of a 1.8 s total duration to 60% of a 3.3 s total duration (t [for slope $\neq 0$] = 2.18, $p < .05$). In the second experiment there was no significant change in the relative time to the second peak (t [for slope $\neq 0$] = -1.83, n.s.).

The second approach to analyzing the data was to compare the time interval from the start of movement to the second peak (0-2 interval) with the time interval from the second to third peaks (2-3 interval). If the proportional duration model holds, these intervals should be highly correlated within a trial, and should maintain a constant ratio across trials. These intervals had a significant correlation ($r = .31$, $p < .05$) in the first experiment (Figure 8 in Armstrong, 1970, p25), but their ratio decreased from 1.07 to .50 as interval 0-2 changed from .8 to 1.9 s (t [for slope $\neq 0$] = -7.5, $p < .001$). The results were even more negative for the second experiment (Figure 13 in Armstrong, 1970, p 34); the intervals had an insignificant negative correlation ($r = -.07$, n.s.) and their ratio decreased from .61 to .26 as interval 0-2 changed from 1.4 to 2.7 s (t [for slope $\neq 0$] = -8.6, $p < .001$). In sum, then, the studies of Armstrong (1970) do not support a proportional duration model. The basic flaw in Armstrong's analyses was that the correlation Armstrong observed between the times from the start of movement to the second and third displacement peaks is primarily due to their shared component (the time from start of movement to the second peak) rather than any consistent change in movement rate. That is, a and $a+b$ will be positively correlated, even if a and b are completely independent.

This re-analysis of Armstrong's data also illustrates the conservative nature of the constant proportion test. Because the time from the start of movement to the second peak and the time from the second to third peaks was completely uncorrelated in the second experiment, there was no consistent trend in the relative time to the second peak, and the constant proportion test did not detect a violation of the proportional duration model. In mathematical terms, if a and b are uncorrelated durations, then $a/a+b$ will not change consistently as a function of $a+b$. Thus the constant proportion test can only reject data that has a consistent, non-proportional change with overall duration.

Zelaznik, Schmidt, Gielen, and Milich (1985) examined simple horizontal arm movements to a target, with varying movement distances and movement times. In their first study, subjects were given about 500 practice trials at each time and distance condition. The movements were 10, 20 and 30 cm long, with movement times of 150, 200 and 250 ms. Data from individual subjects and instances were analyzed. Zelaznik et al. (1985) found that although the durations of the positive and negative accelerations were linearly related to the movement's duration, the times of peak acceleration and deceleration were not. In fact, they found that the time of peak acceleration was constant and independent of the movement duration, and same was true of the time of peak deceleration with

respect to the end of the movement. A second experiment that focused on the effect of movement times confirmed these results. Zelaznik et al. (1985) conclude that "the maintenance of relative timing is not an invariant feature of motor control" (p. 22) in this task.

Wrist twist. Shapiro (1976) had subjects learn a timed sequence of seven wrist twists in 265 trials over 3 days. A Pattern group attempted to match a given pattern of time and angle targets with a total movement time of 1600 ms. There was also a Control group that made the same movements attempting only to match the total time. After completion of the learning trials, there were 15 trials of the movement from memory, and 15 sped-up trials in which subjects were instructed to make the movements as fast as possible while disregarding any learned timing pattern. The data were averaged over subjects and trials. In a combined analysis of Pattern and Control groups, Shapiro found no significant difference between the memory and sped-up trials, but did find significant differences in the duration proportions for several of the components. The author concluded that the results supported a proportional duration model, but was able to hold this view only by combining the Pattern and Control groups and proposing that "the first part of [the] movement sequence is programmed and the second half is programmed separately" (p. 23). The entire movement sequence does not maintain relative timing, and this is especially evident when the Pattern and Control groups are analyzed separately.

Carter and Shapiro (1984) reported a similar experiment in which subjects learned a sequence of four wrist twists. After 600 trials in three days, subjects were asked to ignore the timing they had learned, and to make the response as rapidly as possible while maintaining spatial accuracy. Data from the last 10 learning trials were compared with data from the 10 sped-up trials. The data were averaged over trials and subjects. The average total duration was 570 ms for the final learning trials, compared with 461 ms for the sped-up trials, and the averaged proportional durations of the components were not significantly different in the two conditions. However, in addition to the problem with comparing averaged data, it should be noted that none of the subjects were able learn the original target movement. The target durations varied by a factor of almost 2 (200 msec for the first component and 110 msec for the second component), but subjects made all movements with approximately equal durations (e.g., 135 ± 16 ms for the first component and 150 ± 8 ms for the second component). The sped-up trials also had approximately equal durations for all components. This suggests that the timing was determined by some simple physical constraint rather than a central motor program, or that subjects adopted a pattern of equal timing for all movements. In either case, there is little evidence for the proportional duration model.

Key press. Summers (1975) trained subjects to make a series of nine keypresses with a specified timing pattern and then asked them to make the keypress sequence as fast as possible while ignoring the previously learned timing. The training consisted of 473 repetitions of the sequence over 3 days. Flashing lights above the keys indicated the target timing, which was a repeating sequence: 500-500-100 ms for one group of subjects and 500-100-100 ms for another group of subjects. At the end of training, there was a test condition in which subjects pressed the same sequence of keys as rapidly as possible for a total of 110 repetitions. Data were combined across subjects and instances for analysis. Although subjects were told that maintenance of the previous timing was no longer important or necessary in the test condition, some remnant of the previous timing was apparent, at least at the beginning of the test condition. By the end of the test condition, subjects in the 500-500-100 group had completely lost the previous timing pattern, but subjects in the 500-100-100 group still showed remnants

were three letter words, which did not have separated interstroke intervals, and the interstroke intervals of alternating-hand, three letter words had a strong negative correlation for many typists. For example, the correlation between the *th* and *he* interstroke intervals in the word *the* ranged from $-.67$ to $+.54$ across the typists, with an average correlation of $-.29$. The within-word correlation of $.20$ for the separated intervals suggests that about 33% of the observed standard deviation of interstroke intervals (20% of the variance) was due to changes in typing rate.

In accord with Experiment 1, there was a 31% overall rejection rate in the constant proportion test. The rejection rate for individual typists ranged from 15% to 50%, once again not supporting the proportional duration model.

Experiment 3: Random Words

Method. The text for Experiment 3 was a set of 69 words, from 5 to 11 letters in length. The words were presented in 20 blocks, each block consisting of a different random arrangement of the words. Presenting the text as random words rather than prose appears to have no effect on the performance of skilled typists (Fendrick, 1937; Shaffer, 1973; West & Sabban, 1982).

Results. The results from Experiment 3 are shown in the third section of Table 4. They are based on a total of 688 words (4505 interstroke intervals), with an average of 16 instances per word. The range of durations for a word averaged 33% of the mean duration for that word. The within-word correlation of $.12$ for the separated intervals suggests that about 27% of the observed standard deviation of interstroke intervals (12% of the variance) was due to changes in typing rate.

In close accord with the results from Experiments 1 and 2, there was a 36% overall rejection rate in the constant proportion test. The rejection rate for individual typists ranged from 24% to 48%, all well above the rate expected with the proportional duration model.

Experiment 4: Slow, Normal, and Fast Typing

One possible objection to the preceding experiments is that the range of typing speeds was not wide enough, and any effects of a multiplicative rate parameter were swamped by other effects. Note that these "other effects" cannot be just random error because, as illustrated in Table 1, even a large random error with a small multiplicative rate parameter would still produce a rejection rate of approximately 5%. These "other effects" would thus have to be systematic effects of typing rate on interstroke intervals. In any event, Experiment 4 was an attempt to increase the relative effect of a rate parameter by explicitly asking the typists to type at slow, normal or fast rates.

Method. The text was based on a set of 8 sentences (99 words, 611 characters), all in lowercase and without punctuation. Each page to be transcribed was a different random arrangement of these sentences. Ten typists were asked to type 6 pages as slow as they could comfortably type, 6 pages at their normal rate, and 6 pages as fast as possible. The typing rate alternated for successive pages of text.

Table 4

Typist	Wpm	Words	Correlations		% Rejection
			All	Sep	
Experiment 1: Prose					
t01	77	108	0.13	0.20	28
t02	60	109	0.31	0.23	34
t03	71	108	0.12	0.13	34
t04	62	109	0.10	0.13	31
t05	61	108	0.25	0.18	42
Mean	66		0.18	0.17	34
Experiment 2: Prose					
t01	90	19	-0.20	0.15	26
t02	66	19	0.28	0.29	30
t03	76	19	0.04	0.15	50
t04	75	19	0.01	0.28	43
t05	61	19	0.28	0.33	39
t06	82	19	0.16	0.21	17
t07	110	19	-0.26	0.18	33
t08	112	19	-0.06	0.24	15
t09	86	18	-0.12	0.14	26
t10	85	19	0.03	0.20	33
t40	92	18	-0.06	0.03	29
Mean	85		0.01	0.20	31
Experiment 3: Random words					
t60	64	68	0.28	0.25	43
t61	101	69	0.02	0.08	31
t62	69	69	0.10	0.11	37
t63	84	69	0.08	0.09	40
t64	92	69	0.06	0.09	37
t65	68	69	0.14	0.11	37
t66	72	69	0.09	0.08	37
t67	69	69	0.14	0.14	24
t68	66	69	0.18	0.16	48
t69	66	68	0.06	0.08	26
Mean	75		0.11	0.12	36
Experiment 4: Varied rate					
t70	71	67	0.69	0.67	34
t71	69	67	0.67	0.67	42
t72	71	68	0.64	0.61	33
t73	81	68	0.47	0.49	56
t74	64	68	0.84	0.83	43
t75	101	69	0.94	0.95	72
t76	74	69	0.41	0.44	34
t77	69	69	0.92	0.93	35
t78	65	69	0.79	0.79	28
t79	53	69	0.53	0.51	33
Mean	72		0.69	0.69	41

For each word examined, correlations were calculated between pairs of interstroke intervals across all instances of the word. All combinations of interstroke intervals were used, but to avoid undue bias toward long words, a mean correlation was calculated for each word, and these correlations were then averaged to produce a grand mean correlation for each typist. Parallel models of timing production can produce negative correlations between adjacent intervals (see Gentner, 1982; Wing, 1980), and could affect the interpretation of correlations that included adjacent intervals. Therefore, the mean correlation between all non-adjacent intervals was also calculated. As it turned out, neither within-word averaging nor exclusion of adjacent intervals made any appreciable difference in the analyses.

The constant proportion test. The constant proportion test has already been described. Each interstroke interval was tested to determine if the relative duration of the interstroke interval remained constant as a function of the overall word duration. The measure reported is the percentage of tests that reject a constant proportion with $p = .05$ or less. Interstroke intervals, rather than times, were used with the constant proportion test because typing data fits a serial model of timing better than a parallel model of timing (Gentner, 1982). When the constant proportion tests were repeated with times since the first letter of the word, the rejection rates were slightly higher.

Experiments 1 and 2: Prose

Method. In Experiment 1, five typists transcribed a collection of magazine articles adapted from *Reader's Digest* totaling 9371 words (55,572 characters) at their normal typing rates. In a similar study about a year later, the five typists from Experiment 1 and six additional typists participated in Experiment 2, in which they transcribed one of the magazine articles from Experiment 1, with a total of 2067 words (12,208 characters).

Results. The results from Experiment 1 are shown in the first section of Table 4. They are based on a total of 542 words (1841 interstroke intervals), with an average of 22 instances per word. On average, the range of durations for a word was 38% of the mean duration for that word. The within-word correlation of .17 for separated interstroke intervals suggests that about 31% of the observed standard deviation of interstroke intervals (17% of the variance) was due to changes in typing rate.

When tested with the constant proportion test, the hypothesis of a constant relative duration was significantly rejected for 34% of the interstroke intervals. The lowest rejection rate for any typist was 28%. This rejection rate is, of course, much higher than the 5-10% rate expected with the proportional duration model. It is interesting to note that the rejection rate is also much higher than the 17% rejection rate found in the simulation studies with a 33% contribution of an additive rate parameter (see Table 1), indicating a serious deviation from the behavior expected with a simple rate parameter.

Experiment 2 included more typists, but fewer words than Experiment 1. The results from Experiment 2 are similar to the results from Experiment 1 and are shown in the second section of Table 4. They are based on a total of 207 words (584 interstroke intervals), with an average of 18 instances per word. On average, the range of durations for a word was 49% of the mean duration for that word. In this experiment, the mean within-word correlation for all interstroke intervals was much lower than the correlation for only separated intervals. This was because 9 of the 19 words examined

Analyses

The data used for these analyses were the interstroke intervals of words that occurred several times in the text. In each study, the words used were all words occurring at least ten times in the original text, at least 3 letters in length, and consisting of only lowercase letters. The data for a given word were the interstroke intervals within the word; for example, the four interstroke intervals within a five letter word. All analyses were based on data from individual typists. Because the analyses can be strongly affected by outlying data, two procedures were used to eliminate atypical instances of words. First, instances containing an interstroke interval greater than 400 ms (typically about 9% of the instances) were eliminated. Second, instances containing an interstroke interval more than 3 standard deviations away from the mean for that interval (another 5% of the instances) were eliminated. Finally, if less than five instances of the word remained, the word was not used for that typist.

Correlational analysis. When a word is typed repeatedly in the course of transcribing a text, the interstroke intervals for a given digraph vary from one instance to another. For example, in 20 repetitions of the word *system*, the interstroke intervals of one typist for the *sy* digraph varied from 89 to 394 ms (mean = 152, *SD* = 77). How much of this variation is due to long term variations that affect the whole word, such as changes in typing rate, and how much is due to short term variations that affect only a single keystroke? A study of correlations can help answer this question. Assume that the variability of an interstroke interval is composed of two terms: 1) a common term, common to all interstroke intervals in the word, and 2) an independent term, different for all the interstroke intervals. The previous question can then be rephrased to ask: What proportion of the observed variability in interstroke intervals comes from the common term, as opposed to the independent term? If the common and independent terms are normally distributed, the answer is simple. The proportion of variance due to the common term is equal to the correlation of two interstroke intervals over instances of the word.² For most purposes, the proportion of standard deviation due to the common term is more meaningful than the proportion of variance. The relation between correlation and standard deviation is more complicated. If *p* is the proportion of standard deviation due to the common term, then the correlation coefficient is given by the following sigmoidal function.

$$r = \frac{p^2}{p^2 + (1 - p)^2} \quad (3)$$

Even if this simple model with normally distributed common and independent terms is not completely valid, a correlational analysis can still provide an approximate measure of the proportion of variability that is constant within an instance of a word.

2. In an earlier paper (Gentner, 1982) I stated that an mean correlation between intervals of .2 indicated that rate changes accounted for 4% (r^2) of the variance. This was an error. A correlation of .2 suggests that 20% of the variance is due to rate changes.

Music. Music is different from the previous tasks because, with music, the control of timing is an explicit part of the performance. Musical notation directly specifies the timing structure of the piece, although performers may deviate from regular timing for expressive effect (Shaffer, 1984).

Michon (1974) analyzed tape recordings of a performance of *Vexations* by Erik Satie for evidence of changes in the temporal pattern with changes in overall duration. *Vexations* is aptly named, because the performer must repeat a five part motif 840 times. The entire performance took about 19 hours. The duration of the individual parts varied roughly between 15 and 25 s, and Michon found that this duration was the major factor accounting for deviations from the prescribed timing. That is, the temporal structure of the performance varied with the rate of playing. However, Michon did not present any quantitative evidence for this conclusion.

Clarke (1982) also examined performances of *Vexations* by Erik Satie. Two graduate musicians each performed about an hour of the piece on the beautifully instrumented piano in Shaffer's laboratory. Clarke divided up the repetitions into six groups based on overall duration of each part, and found significant effects of part duration. In particular, the faster instances (about 13 s duration) tend to become slower as they proceed, and the faster instances (about 22 s duration) tend to become slower. The constant proportion test did not show any significant changes in relative duration for the quarter note intervals from subject M (Figure 3A in Clarke, 1982, p 8), but 6 of 11 quarter note intervals showed a significant change in relative duration for subject G (Figure 3B in Clarke, 1982, p 8).

Typewriting Data

In addition to analyzing data from the literature, I have applied the constant proportion test and other analyses to a extensive body of data that I have collected over the past five years from expert typists.

General Method

All subjects were professional typists recruited from the university and local businesses. They typically typed 14 hours per week (range = 2 to 25), and their median typing speed was 71 words per minute (range = 53 to 112) measured over approximately 40 minutes of typing and uncorrected for errors. Subjects were paid for participation. The five subjects in Experiment 1 were also subjects in Experiment 2. Otherwise, subjects participated in only one experiment.

In Experiment 1, subjects typed at a Hazeltine 1500 computer terminal connected to a minicomputer that recorded keypresses and the corresponding times. Typed characters were displayed on the terminal screen. The typists were all familiar with this terminal, having used it in conjunction with the campus word-processing system. In Experiments 2 through 4, subjects typed on a high quality, electronic keyboard (Microswitch 51SD12-4) with a layout identical to the IBM Selectric typewriter and a similar "feel". The keyboard was connected to a microcomputer that recorded keypresses and the corresponding times with an accuracy of 1 ms. Typed characters were displayed on a CRT screen in front the the typist. The text to be transcribed was presented as printed or typewritten copy in a convenient place next to the keyboard. After a warmup period, typists were asked to transcribe the text at their normal, rapid rate and to ignore any errors they might make.

Table 3

Interstroke Intervals and Times in Typewriting

Unit	% at short duration	% at long duration	t (slope $\neq$ 0)	Significance
<i>enclosed</i> (42 instances; total duration = 843 to 1177 ms)				
Interstroke intervals				
en	10.8	11.3	.69	n.s.
nc	19.1	20.5	1.89	n.s.
cl	9.3	13.0	5.66	< .001
lo	18.5	14.7	-6.57	< .001
os	10.8	8.7	-5.44	< .001
se	10.7	14.2	5.64	< .001
ed	20.8	17.7	-5.37	< .001
Time since first letter				
n	10.8	11.3	.69	n.s.
c	29.9	31.7	2.62	< .02
l	39.2	44.7	6.33	< .001
o	57.7	59.4	2.26	< .05
s	68.5	68.1	-.62	n.s.
e	79.2	82.3	5.37	< .001
<i>trouble</i> (27 instances; total duration = 845 to 1218 ms)				
Interstroke intervals				
tr	21.6	18.5	-3.57	< .002
ro	14.1	19.2	4.83	< .001
ou	17.0	13.9	-2.54	< .02
ub	14.7	18.7	2.74	< .02
bl	17.4	14.9	-1.92	n.s.
le	15.1	14.8	-.37	n.s.
Time since first letter				
r	21.6	18.5	-3.57	< .002
o	35.8	37.7	1.85	n.s.
u	52.8	51.6	-.87	n.s.
b	67.5	70.3	2.72	< .02
l	84.9	85.2	.37	n.s.

Note. Data for the word *enclosed* is from Terzuolo and Viviani (1980).

Data for the word *trouble* is from Terzuolo and Viviani (1979).

repeated instances of the word. (This prediction is closely related to the constant proportion test used in the present paper, but I now feel that the constant proportion test is a more direct test of the proportional duration model than the test for a constant ratio of intervals.) The constant ratio analysis was restricted to the pairs of intervals with significant positive correlations, but even then, 60% of the pairs deviated significantly from a constant ratio. The results from both tests were taken as evidence against the proportional duration model.

In summary, the studies of handwriting have yielded mixed results. The data of Hollerbach (1981) and Wing (1978) do not fit the proportional duration model, whereas the data of Viviani and Terzuolo (1980) support the proportional duration model.

Typing. In addition to the study of rapid speech mention earlier, Sternberg, Knoll, Monsell, and Wright (1983) also examined the rapid typing of short letter sequences. Like their finding with short speech sequences, the interstroke intervals of short typewriting sequences increase as the length of the sequence gets longer. In order to determine whether these increases in the interstroke interval were evenly distributed over the interval, Sternberg et al. examined the index finger trajectories in alternating-hand sequences. They divided the trajectory into five components, and found that the increase in the interstroke interval (actually it was two interstroke intervals, because they examined an alternating hand sequence) was entirely localized in one component of the trajectory. The "late lift" component increased from 80 ms (36% of the trajectory) to 240 ms (62% of the trajectory), as the number of letters in the sequence increased from 3 to 5 and the total trajectory duration increased from 220 to 390 ms. When tested with the constant proportion test, the relative duration changed significantly for 3 of the 5 components. The results are very similar to their results from short speech sequences: the changes were localized to one component of the sequence, and therefore to not fit the proportional duration model.

Probably the most commonly cited evidence for the rate parameter model is the work of Terzuolo and Viviani (1979, 1980; Viviani and Terzuolo, 1980) on typing. Terzuolo and Viviani examined the performance of expert typists during continuous typing to determine how the keystroke times within a word change as the overall duration of the word changes. Terzuolo and Viviani found that when the keystroke times for repeated instances of a word are plotted in the proper manner, the times for each letter appear to radiate from a common origin, thus showing that relative timing was maintained as the overall duration of the word changed. Terzuolo and Viviani have presented these radial plots for two words: 42 instances of the word *enclosed* (Figure 6A in Terzuolo and Viviani, 1980, p 1092; and Figure 2A in Viviani and Terzuolo, 1980, p 527) and 27 instances of the word *trouble* (Figure 1A in Terzuolo and Viviani, 1979, p 115). Although these radial plots are intuitively clear and appealing, there is no quantitative measure to determine how well the data fit the radial lines, or to summarize results over a large number of words and typists. In fact, as shown in Table 3, when analyzed with the constant proportion test these data do not support the proportional duration model. For both words, the majority of interstroke intervals did not maintain the same relative duration as the overall word duration changes. Similar results were obtained when the data for the two words was analyzed in terms of time since the first letter rather than in terms of interstroke intervals. Thus, the typing data of Terzuolo and Viviani do not support the proportional duration model.

In an earlier paper (Gentner, 1982), I examined data from prose transcription by a group of expert typists for evidence of the proportional duration model. Two predictions of the proportional duration model were tested on data from repeated words in the text. The first prediction was that two different interstroke intervals within a word should be positively correlated over repeated instances of the word. For example, if the *wi* interval in *wish* is longer than average for a particular instance of the word, the *th* interval in that instance should tend to be longer also. Of the 1,517 pairs of intervals examined, only 15% had significantly positive correlations. The proportional duration model makes a second, stronger prediction: that two interstroke intervals within a word should tend to maintain a constant ratio over

In other analyses of these EMG data, Tuller, Kelso, and Harris (1981) found that "no measured interval was found to vary systematically with changes in speaking rate" (p. 73), and that "the temporal relationships between these muscles were not systematically affected by changes in speaking rate" (p. 73).

In a related study, Tuller, Harris, and Kelso (1981) found that, contrary to the proportional duration model, overlap between articulator muscle activities was independent of acoustic syllable duration (Figure 4 in Tuller et al., 1981, p. 48). The proportional duration model predicts that all aspects of speech production should scale proportionally in time. The results of Tuller, Kelso, and Harris (1982) and Tuller, Harris, and Kelso (1981) are consistent with the finding by Kozhevnikov and Chistovich (1965) that speech events at the level of the syllable do not correspond to the proportional duration model.

Sternberg, Monsell, Knoll, and Wright (1978) have found that when subjects perform a rapid action sequence, the rate of performance is dependent upon the number of units in the sequence. For example, the time to speak a digit is about 8 ms longer in a sequence of 5 digits than in a sequence of 4 digits. Sternberg, Knoll, Monsell, and Wright (1983) inquired whether this change in speaking rate was equally distributed through the word or was localized to a particular portion of the word. They had subjects rapidly speak sequences of two-syllable words, such as *copper* and *token*. The data were combined over instances, words, and subjects. When the acoustic waveform for each word was decomposed into six components, the increase in duration was almost entirely a result of an increase in duration of the second vowel, which increased from 17 ms (8% of the word) to 42 ms (17% of the word), as the number of words in the sequence increased from 2 to 5 and the total word duration increased from 216 to 249 ms. When the speech data (Figure 25 in Sternberg et al., 1983, p. 16d) were tested with the constant proportion test, the relative duration changed significantly for 4 of the 6 word components.

Handwriting. Hollerbach (1981) reports vertical acceleration profiles for the pen of a subject writing the word *hell* at three different rates (Figure 28 in Hollerbach, 1981, p. 154). If the letter boundary is taken to be the moment of zero acceleration at the bottom of the stroke, then the relative durations for three of the 4 letters changed significantly as the word duration changed from about 800 ms to about 2000 ms (all values of t [for slope $\neq 0$] > 6.6 , $p < .05$).

Viviani and Terzuolo (1980) asked people to write letters of the alphabet at different speeds and recorded the pen positions with a digitizing table. They report velocity profiles for nine instances of a letter written by one person (Figure 4 in Viviani & Terzuolo, 1980, p. 529). Analysis of their data with the constant proportion test indicated that none of the intervals between velocity maxima and minima showed a relative change as the letter duration changed from 245 to 680 ms (first maxima to last maxima), thus supporting the proportional duration model.

Wing (1978) examined timing within single handwritten letters (v, n, w, and m). Based on correlations of individual instances, he concluded that "these data do not show overall time scaling of successive segments" (p. 164).

Other Highly Practiced Skills

This final section of the review focuses on other highly practiced skills. Some are innate actions, such as breathing; others are learned skills in which the subjects have had thousands of hours of practice, such as piano playing.

Breathing. Clark and von Euler (1972) studied the relationships between depth of a breath and the duration of the inspiratory and expiratory phases in cats and humans. They found an approximately linear relation between the inspiration and expiration durations. Clark and von Euler also state that the durations were approximately proportional for humans, but the durations were not proportional for cats. Analyses of data from individual breaths for one human and one cat with the constant proportion test confirmed these findings. For the human (Figure 9a in Clark & von Euler, 1972, p. 286), the relative duration of the inspiration phase remained at 47% as the total duration changed from 1.2 to 3.7 s (t [for slope $\neq 0$] = -.04, n.s.). In contrast, for the cat (Figure 9b in Clark & von Euler, 1972, p. 286), the relative duration of the inspiration phase increased from 38% to 50% as the total duration changed from 1.4 to 2.4 s (t [for slope $\neq 0$] = 18.1, $p < .001$).

Speech. Kozhevnikov and Chistovich (1965) studied how the components of speech vary as the overall speed changes. They had subjects repeat phrases with overall durations ranging from 1 to 3 s, and determined the duration of various components from measurements of speech articulator movement. They examined the relative duration of words, syllables, and vowels and consonants. The data were averaged over instances and subjects. When analyzed with the constant proportion test, the relative duration of words (Figure 3.5 in Kozhevnikov and Chistovich, 1965, p. 83) changed significantly in 2 of 9 cases. This rejection rate is within chance (binomial distribution, $p = .23$). The relative duration of syllables in the word *topila* (Figure 3.7 in Kozhevnikov and Chistovich, 1965, p. 86) did not change significantly in any of 9 cases. In contrast to behavior of words and syllables, however, the relative duration of consonants (Figure 3.9 in Kozhevnikov and Chistovich, 1965, p. 88) decreased from 36% for a total phrase duration of 800 ms to 29% with a total phrase duration of 3000 ms (t [for slope $\neq 0$] = -12.7 $p < .001$). This finding that the relative duration of consonants and vowels change as the rate of speaking changes is commonly reported. Thus the constant proportion test does not describe speech at the syllable level, but does provide a good mode of longer units of speech.

Tuller, Kelso, and Harris (1982) recorded EMG patterns associated with tongue, lip and jaw movements while words were spoken with different speeds and stress. They searched for linear relationships between a component of the EMG pattern and a longer period of activity by calculating correlations for onset, offset and peak amplitude of EMG for muscles associated with vowel and consonant production. Data were averaged over instances before analysis. Eight of the nine relationships reported showed a wide variety of positive and negative correlations. Only one relationship showed a consistently high correlation: in the sequence V_1CV_2 , the time from onset of V_1 to onset of the consonant C was highly correlated with the time between onset of V_1 and V_2 . Although a high correlation indicates a linear relation, but not necessarily a proportional relation, examples of single-subject data for the one case with high correlations (Figure 4 in Tuller et al., 1982, p. 467) were accepted by the constant proportion test in 11 of 12 cases.

each speed (Figures 2A and 2B in Grillner et al., 1979, p 179). The constant proportion test could not be applied because data were plotted only for slowest and fastest running and walking speeds, but I did compare these extremes. The support phase occupied an average of 58% of a 915 ms cycle at the slowest walking speed, and 49% of a 630 ms cycle at the fastest walking speed. The within-subject difference in proportion was significant (t [for slope $\neq 0$] = -8.1, $p < .001$). Similarly, the support phase occupied an average of 45% of a 725 ms cycle at the slowest running speed, and 25% of a 530 ms cycle at the fastest running speed. The within-subject difference in proportion was significant (t [for slope $\neq 0$] = -8.6, $p < .001$).

Grillner et al. (1979) also present data for two of their eight subjects on the duration of the flexion phase at the knee joint during running and walking. The data are for individual subjects and step cycles. The duration of the flexion phase was linearly related to the step cycle duration, but it was not proportional to step cycle duration; that is, when flexion or extension duration is plotted against step cycle duration, the fitted straight line cannot be extrapolated through the origin. These conclusions were confirmed when the data were analysed with the constant proportion test. For one subject (Figure 1A in Grillner et al., 1979, p 178) the relative duration of the flexion phase decreased significantly as the walking step cycle increased from 620 ms to 918 ms (t [for slope $\neq 0$] = -3.56, $p < .01$), and the relative duration of the flexion phase also decreased significantly as the running step cycle increased from 519 ms to 758 ms (t [for slope $\neq 0$] = -2.73, $p < .02$). For a second subject, one of two with different slopes for running and walking when flexion duration was plotted against step cycle (Figure 1B in Grillner et al., 1979, p 178), the relative duration of the flexion phase did not change significantly as the walking step cycle increased from 553 ms to 897 ms (t [for slope $\neq 0$] = 1.24, n.s.), whereas the relative duration of the flexion phase decreased significantly as the running step cycle increased from 482 ms to 705 ms (t [for slope $\neq 0$] = -5.14, $p < .001$).

Lee, Lishman, and Thomson (1982) measured the strides of skilled long jumpers during their run-up to the takeoff board. They were primarily interested in how the jumpers adjusted their gait to land just in front of the takeoff board for the jump, but they report data from the individual step cycles that was analyzed with the constant proportion test. The data reported were for individual jumpers for each stride before takeoff, but averaged over 12 runs (Figure 4 in Lee et al., 1982, p 453). Data for the last stride before takeoff was omitted from these analyses because it was atypical. Subject MN's support phase duration varied from 47% of a 218 ms step cycle to 60% of a 272 ms step cycle (t [for slope $\neq 0$] = 5.0, $p < .001$). Subject VW's support phase duration varied from 48% of a 225 ms step cycle to 63% of a 281 ms step cycle (t [for slope $\neq 0$] = 4.7, $p < .001$). Subject FM's support phase duration varied from 50% of a 220 ms step cycle to 60% of a 246 ms step cycle (t [for slope $\neq 0$] = 2.0, n.s.). Thus, the step cycles of two of three jumpers are not consistent with the proportional duration model.

Winter (1983) reports ankle, knee and hip joint angle curves for people walking at their natural pace (571 ms mean step cycle), as well as other groups walking at a slow pace (708 ms) or a fast pace (493 ms). The data were averaged over 14 to 16 subjects within each pace group and over an unstated number of step cycles for each subject. Plots of the average joint angles against the normalized stride cycles appear remarkably similar for the three pace groups, but there is insufficient information for any statistical test of the proportional duration model.

Table 2

Duration of Step Cycle Phases in the Cat

Phase	% at Short Duration	% at Long Duration	t (slope $\neq 0$)	Significance
Trot (12 instances, total duration = 345 to 620 ms)				
E1	22	16	-1.51	n.s.
E2	19	12	-2.74	< .02
E3	27	53	4.63	< .001
F	31	18	-3.24	< .01
Gallop (8 instances, total duration = 253 to 382 ms)				
E1	28	29	.10	n.s.
E2	14	13	-.30	n.s.
E3	18	18	-.13	n.s.
F	39	41	.15	n.s.

Note. Data are from Goslow, Reinking, and Stuart (1973).

Human. Shapiro, Zernicke, Gregor, and Diestel (1981) filmed experienced runners while running and walking on a motor driven treadmill. They measured the duration of the four phases of the Phillipson step cycle at four walking speeds and four running speeds. Data were combined over subjects and instances before analysis. The authors report that there was a significant effect of speed on the proportion of the step cycle occupied by the components, but most of the change was between gaits, and they did not find a significant difference in relative durations within the walking or running gaits. These results are not directly relevant to the rate parameter model, however, because Shapiro et al. (1981) examined proportions as a function of speed of locomotion, rather than as a function of total step cycle duration. In fact, although the average step cycle durations for walking have a reasonable range of about 33%, the range of step cycle durations for running is only 4% and 3 of the 4 durations are within 1% (see Figure 3A in their paper), and thus the running data do not provide information about how the components of the step cycle change with changes in the total duration. When the data for walking (from Figures 3 and 4 in Shapiro et al., 1981, pp. 41-42) were analyzed with the constant proportion test, the mean relative duration of three of the components of the Phillipson step cycle did not change significantly with total cycle duration, but the fourth component, E3, increased from 48% of a 954 ms cycle to 54% of a 1320 ms cycle (t [for slope $\neq 0$] = 6.9, $p < .01$). When the walking data (Figure 3 in Shapiro et al., 1981, p. 41) were analyzed in terms of swing and support phases, there was no change in the relative duration of the support phase (t [for slope $\neq 0$] = 2.4, n.s.). The net results, then are somewhat mixed. With one exception (the E3 phase in walking), the analyses support the proportional duration model, but the main problems remain: the data were combined across subjects and the running data do not span a significant range of cycle durations.

Grillner, Halbertsma, Nilsson, and Thorstensson (1979) studied humans walking and running on a treadmill at velocities from 1.7 to 8.3 m/s. They recorded joint movement with a knee goniometer, and mounted foot switches on the subjects' shoes to record the duration of foot contact with the ground. Cycle duration data are reported for seven individual subjects, but are averaged over ten instances at

Delcomyn found that, on average, the support phase occupied about 51% of the step cycle when the cycle time was 50 msec, but increased to 71% of the cycle when the cycle time was 400 ms. The data presented in Delcomyn's (1971) article are for individual instances of step cycles; it is unclear whether the data is for one or several cockroaches. I analysed data for individual instances of step cycles (from Figures 5A and 5B in Delcomyn, 1971, p449) with the constant proportion test. The support phase duration of leg R_3 increased from 47% of a 52 ms step cycle to 70% at a 440 ms step cycle (t [for slope $\neq 0$] = 8.3, $p < .001$). The support phase duration of leg R_2 increased from 51% of a 53 ms step cycle to 78% at a 465 ms step cycle (t [for slope $\neq 0$] = 9.5, $p < .001$). These analyses support Delcomyn's conclusion that "both forward [swing] and rearward [support] movements of the legs relative to the body decreased in duration as the insect's rate of forward progression increased, but at different rates" (p. 452).

Lobster. Macmillan examined walking in the American lobster, recording movements on videotape and also recording electromyograms from selected muscles. Two of Macmillan's figures report data from individual instances of movement that I was able to analyze with the constant proportion test. Both sets of data were consistent with the proportional duration model. The proportional duration of the swing phase measured from the videotape record (Figure 21a in Macmillan, 1975, p 31) did not change significantly as the step cycle time increased from 1000 ms to 1800 ms (t [for slope $\neq 0$] = 1.31). Similarly, the proportional duration of the depressor bursts measured from the electromyograph record (Figure 29 in Macmillan, 1975, p 31) did not change significantly as the step cycle time increased from 600 ms to 1200 ms (t [for slope $\neq 0$] = -.18).

Tortoise. Williams (1977) analyzed motion picture films of walking tortoises. She reports that "a change in the forward speed of locomotion was accomplished by an approximately proportionate change in the duration of all portions of the cycle" (p. 54). No data or statistical analyses are reported.

Cat. Goslow, Reinking, and Stuart (1973) analyzed high-speed motion pictures of cats moving freely at speeds from 1 to 16 miles/hr (.4 to 7 m/s). Cats use three gaits over these speeds: walk (up to .7 m/s), trot (.7 to 2.7 m/s) and gallop (above 2.7 m/s). Across this range of speeds and gaits, Goslow et al. (1973) found that the duration of the support phase remained approximately constant at 200 ms, whereas the duration of the swing phase decreased from about 500 ms at the slowest speeds to less than 100 ms at the highest speeds (Figure 3 in Goslow et al., 1973, p 12). They also reported the durations of the four phases of the Phillipson step cycle (Figure 4 in Goslow et al., 1973, p 13). The data were for individual instances, combined over nine cats. There were insufficient data from the walking gait, but it was possible to analyze the data from the trot and gallop gaits with the constant proportion test. These analyses are summarized in Table 2. The proportional duration model does not fit Goslow et al.'s data for the trot gait, but the proportional durations of the step cycle phases are remarkably constant for the gallop gait.

Dog. Arshavskii, Kots, Orlovskii, Rodionov, and Shik (1965) measured the leg joint angles of dogs running on a treadmill. The joint angle records were used to calculate the durations of the support and swing phases of the step cycle (Figure 3A in Arshavskii et al., 1965, p 741). The data presented were individual instances from one dog. Analysis with the constant proportion test did not support the proportional duration model. The support phase duration increased from 45% of a 420 ms step cycle to 69% of a 765 ms step cycle (t [for slope $\neq 0$] = 5.9, $p < .001$).

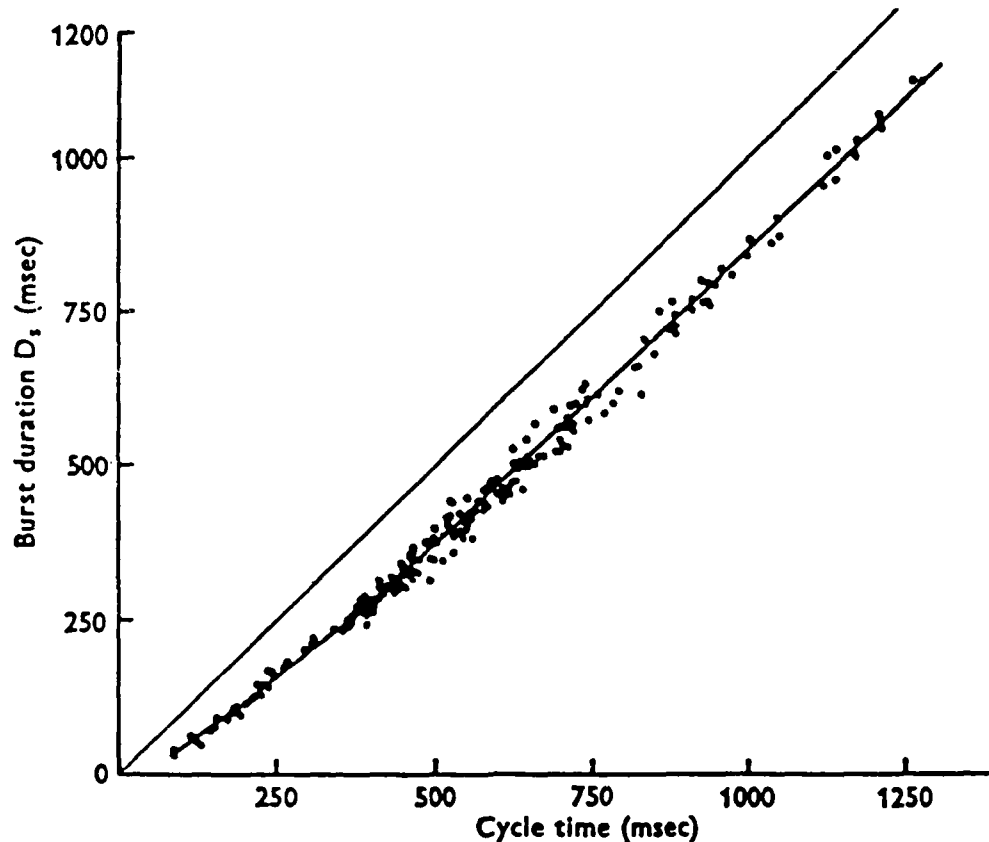


Figure 2. Variation in the burst duration of motor axon D_s with cycle time for a cockroach. The dashed line is an example of how the data might look if the depressor duration maintained a constant proportion of the total cycle time. The difference between the line through the data points and the diagonal line gives an approximate measure of the burst duration of levator motor axon 5. *Note.* From "Central programming and reflex control of walking in the cockroach" by K. G. Pearson, 1972, *Journal of Experimental Biology*, 56, p 182. Copyright 1972 by The Company of Biologists Limited. Reprinted by permission.

of the previous timing pattern. Although Summer's (1975) results are often cited in support of the proportional duration model, the results actually strongly contradict such a model. The ratio of slow to fast interkeypress intervals in the final block of training for the 500-500-100 groups was $579/252 = 2.3$, but the corresponding ratio in the test block was $292/246 = 1.2$. Similarly, the ratios for the 500-100-100 group was $421/184 = 2.3$ in the final block of training and $283/185 = 1.5$ in the test block. It appears that the fast intervals was at a floor, and the only effect of speeding up is to speed up the slow intervals. Proportional durations were not preserved in the test condition, as required by the proportional duration model.

Glencross's (1973) study of subjects turning a hand crank has been cited by several authors (e.g. Shapiro, 1976; Zelaznik et al., 1985) as providing support for the proportional duration model. However, this study compared performance across subjects, each of whom turned the crank at a single speed, and therefore the results are not relevant to the rate parameter model, which describes how an individual performs at differing rates.

Locomotion

In contrast to motor tasks practiced for a few hours, locomotion is largely innate and has been studied in a wide variety of species.

Cockroach. Most of the studies in the literature report behavioral data because it is the easiest data to collect. A few investigators, however, have examined data from neural recordings. Pearson (1972) presents an impressive study based on single motor neuron recordings in cockroaches. Pearson measured the burst durations in levator and depressor motor axons. Figure 2 shows the burst duration of the depressor motor axon from a remarkable cockroach that varied its total step cycle from 84 ms to 1265 ms. The observed depressor burst durations clearly do not maintain a constant proportion of the total cycle duration. The constant proportion test indicated that the proportion of the total cycle occupied by the depressor bursts increased significantly as the total duration increased (t [for slope $\neq 0$] = 22.3, $p < .001$). The depressor bursts occupy 42% of the step cycle when the cycle time is 84 ms, but this proportion increases to 87% of the step cycle when the cycle time is 1265 ms. In fact, for cycle times greater than 400 ms, changes in the depressor durations account for almost the entire increase in cycle time, and the duration of the levator bursts remains approximately fixed.

When the constant proportion test was applied to data from another cockroach (Figure 5 in Pearson, 1972, p 181), none of the durations showed evidence of a multiplicative rate parameter. The duration of levator axon 5 decreased from 46% of the step cycle at a 150 ms step cycle to 34% at a 500 ms step cycle (t [for slope $\neq 0$] = -7.8, $p < .001$). The duration of levator axon 6 decreased from 29% of the step cycle at a 86 ms step cycle to 19% at a 500 ms step cycle (t [for slope $\neq 0$] = -6.8, $p < .001$). And the duration of depressor axon D, increased from 40% of the step cycle at a 98 ms step cycle to 68% at a 530 ms step cycle (t [for slope $\neq 0$] = -14.4, $p < .001$).

Pearson's observations of motor neuron activity correspond nicely to the behavioral observations of cockroach walking by Delcomyn (1971). Delcomyn recorded the walking movements of cockroaches with a high speed motion picture camera. He found that cockroaches used a single, alternating triangle gait at all speeds of locomotion from 5 to 80 cm/s, corresponding to step cycles of 400 to 45 ms.

Results. The typists were able to vary their rate on demand. Table 5 lists their typing rates for

Table 5

Typing Rate in Experiment 4

Typist	Slow	Normal	Fast
t70	44.2	73.5	80.8
t71	49.0	69.0	78.8
t72	48.4	70.9	75.5
t73	61.7	81.0	83.9
t74	38.4	63.9	86.1
t75	34.1	100.5	105.4
t76	60.7	74.2	80.3
t77	25.1	68.5	75.3
t78	37.8	65.3	69.8
t79	37.2	53.3	56.5
MEAN	43.7	72.0	79.2

the three experimental conditions. Not surprisingly, typists were more effective at slowing down from their normal rate than at speeding up. For these analyses, data from all three conditions were pooled; the three conditions just served as a device to encourage a wide range of word durations. The cutoff point for eliminating outlying instances was increased from the normal 400 ms to an interstroke interval of 700 ms because of the many long interstroke intervals in the slow condition.

The results from Experiment 4 are shown in the last section of Table 4. They are based on a total of 685 words (4505 interstroke intervals), with an average of 18 instances per word. The range of durations for a word was much larger than in the previous experiments: an average of 107% of the mean duration for the word. The within-word correlations were also much larger, ranging from .44 to .95 for separated interstroke intervals, reflecting the larger changes in typing rate. The average within-word correlation of .69 for the separated intervals suggests that about 60% of the observed standard deviation of interstroke intervals (69% of the variance) was due to changes in typing rate.

Once again, the hypothesis of a constant relative interstroke interval had a high rejection rate; in this experiment the mean rejection rate was 41%. The rejection rate for individual typists ranged from 28% to 72%. Thus, increasing the range of typing rates actually led to a stronger rejection of the proportional duration model.

Patterns of Change With Typing Rate

If, as has been demonstrated, the interstroke intervals of skilled typists are not described by a proportional duration model, is there any systematic pattern of rate change, or are these data merely the result of random variation? There are three strong arguments indicating that expert typists vary their rate. First of all, Experiment 4 clearly demonstrates that typists can vary their typing rate over a large range on demand. Second, the correlational analyses show that even when typing at normal rate, fluctuations in rate occur that persist for at least the duration of a word. Third, an autocorrelation study indicated that while typing at their normal rate, some typists have rate changes that persist for

many words. The word durations from Experiment 3 were normalized by dividing the observed duration by the mean duration for that word and typist, and then the autocorrelation coefficient was determined for lags of 1 to 50 words. The extent of rate changes was conservatively estimated by the lag before the first negative autocorrelation. Individual differences between typists were very pronounced. Rate changes persisted for only 1 word for two of the typists; 3 to 6 words for two typists; 13-31 words for three typists; and more than 50 words for the remaining three typists. So, for at least some typists, there are normally slow changes in typing rate that persist for many words.

Given that typing rates can change, are there any regularities in the details of how performance changes with overall rate? David Rumelhart and I analyzed the interstroke intervals of typists who transcribed normal prose under fast and slow conditions. In the fast condition, they were told to assume they were typing a rough draft and to type as fast as possible, without concern for errors. In the slow condition, the typists were told to assume they were typing a final copy and to type carefully and minimize errors. For purposes of analysis, the letter-letter digraphs were divided into four classes based on the fingers used in touch typing: doubles, such as *dd*; 1-finger digraphs, non-doubles typed by a single finger, such as *de*; 2-finger digraphs, typed by different fingers on the same hand, such as *de*; and 2-hand digraphs, typed by different hands, such as *de*. The results of the analysis are shown in Table 6. Interstroke intervals for 2-hand digraphs had the largest relative increase, followed by 2-finger

Table 6

Typist	Wpm fast	Wpm slow	% Increase in ISI (slow-fast)			
			Double	1finger	2finger	2hand
t02	72.5	59.1	7.6	6.7	20.0	28.8
t03	80.3	72.9	1.5	5.4	11.7	12.9
t04	84.3	73.1	4.1	9.7	9.2	16.8
MEAN	79.0	68.4	4.4	7.3	13.6	19.5

and 1-finger and double digraphs, in descending order. The mean absolute change in interstroke intervals followed the same pattern. This, of course, is in direct opposition to the proportional duration model, which predicts that because the interstroke intervals of 2-hand digraphs are shortest, they should show the least absolute change in duration. Instead, 2-hand digraphs (mean = 109 ms) increased by an average of 22 ms, whereas doubles (mean = 185 ms) increased by an average of only 7 ms. An analysis of variance confirmed that digraph class had a significant effect, $F(3,6) = 8.57, p = .014$.

The same pattern of change was found when data from the normal and fast conditions in Experiment 4 were compared. The mean percent decrease in median interstroke intervals was 3.9, 6.8, 10.0, and 12.3 for doubles, 1-finger, 2-finger, and 2-hand digraphs respectively. The significant effect of digraph class was again confirmed by an analysis of variance, $F(3,27) = 11.18, p < .001$.

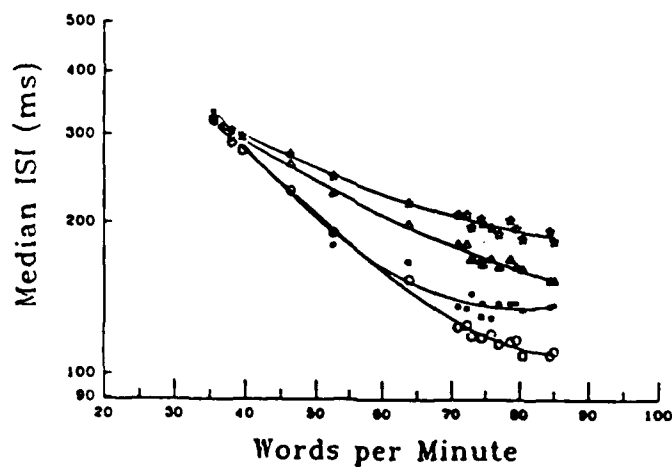
The picture changes somewhat, however, when data from the slow condition is included in the analyses. The pattern of results across all three conditions is most easily seen by examining performance of individual typists. The overall typing rate and the median interstroke intervals for each digraph class were calculated separately for each page of text (99 words) in Experiment 4. Figure 3

shows these results for two typists who had a wide range of typing rates. The median interstroke intervals are plotted on a logarithmic axis in Figure 3, so that if they maintain a constant relative proportion, the lines should be parallel. The lines are clearly not parallel, illustrating the previous finding that the data from this experiment do not fit the proportional duration model. Figure 3, however, shows two patterns that were observed in this study. The first pattern was that at the lowest typing rate, the median interstroke intervals for all digraph classes were similar, but they diverge as the typing rate increases. The spread in the median interstroke intervals (as measured by the standard deviation of the medians) was positively correlated with the typing rate. This was true for all 10 typists on a relative basis (mean correlation across typists = .96) and for 9 of 10 typists on an absolute basis (mean of correlation across typists = .70). The second pattern was that at the highest typing rates, the median interstroke intervals for double and 1-finger digraphs change very little relative to the medians for 2-finger and 2-hand digraphs. This, of course, is similar to the finding from the rough draft/final copy study (see Table 6), but now it is clear that the relative amount of change is dependent on the absolute rate.

Simulation Model of Typing

Another approach to understanding the control of timing is to construct a simulation model of performance and investigate the model's predictions of timing. Rumelhart and Norman (1982) developed a simulation model of a skilled typist that did not have any central timing control. Instead, the observed timing of the model's keystrokes is the result of competition and cooperation between efforts to type several letters at once. Similar to skilled typists, the model's interstroke intervals reflect the constraints of the hands and fingers and the layout of the typewriter keyboard. Although the model normally attempts to type several letters at once, the letters are normally produced in the correct serial order because each letter inhibits every following letter to some extent, and the initial letter thus usually has the highest activation level. When the level of inhibition between successive letters is very high, the model essentially types just one letter at a time. As the level of inhibition is lowered, however, the model attempts to type several letters at once. In this case, movements to type successive letters can overlap in time, especially when successive letters are typed by separate fingers or hands, and the overall typing rate increases. Figure 4 shows how the mean interstroke intervals decrease with decreasing levels of inhibition. Notice that the different digraph classes do not maintain constant relative durations, which would be indicated by parallel lines on this logarithmic plot. Doubles do not speed up at all because there is no advantage in moving the hand for the second keystroke; 2-hand and 2-finger digraphs speed up the most because the model can easily overlap movements on different hands and fingers; and 1-finger digraphs speed up slightly because the model can move its hand into position for the second keystroke while typing the first. This pattern is similar to the results from skilled typists for changes near their normal rate (for example, see Table 6) and also matches the pattern of changes in interstroke intervals observed during acquisition of typing (Gentner, 1983). Note, however, that the effect of changing the inhibition level in the typing simulation does not completely match the pattern of results when skilled typists change over wide ranges as in Experiment 4 (see Figure 3).

Typist t70



Typist t74

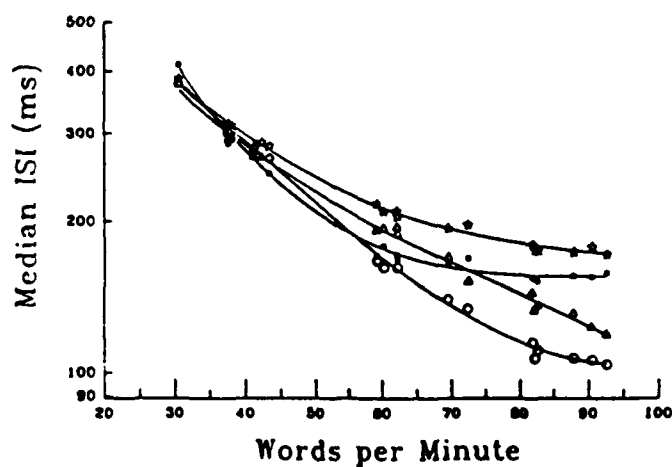


Figure 3. Median interstroke intervals for the digraph classes as a function of overall typing rate. The vertical axis is logarithmic, so that if proportional duration model fit this data, the lines would be parallel. Instead the interstroke intervals are similar at low rates and quite different at high rates. As typing rate increases, the interstroke intervals for some digraph classes continue to decrease, whereas the interstroke intervals for doubles appear to reach a minimum level.

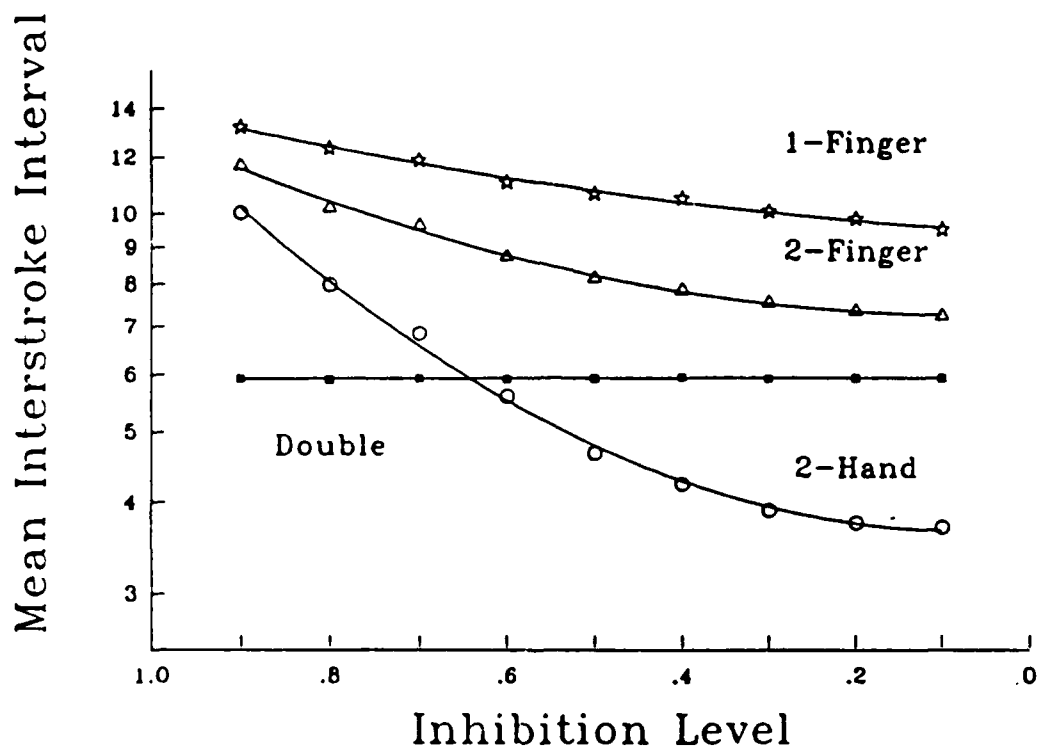


Figure 4. Results from the typing simulation model. Mean interstroke intervals for the digraph classes as a function of the level in inhibition between letters. As the inhibition level decreases, the overall typing rate increases, but the interstroke intervals for the different digraph classes decrease at different rates.

Multilevel Control of Timing

Models of timing, such as the proportional duration model, seem to be based on the idea that motor performance is determined solely by central control. With the proportional duration model, a generalized motor program is stored in long term memory, and with the specification of a multiplicative rate parameter, maps directly onto the timing of behavior, in much the way that the performance of an ideal robot is a direct reflection of its program. At the other extreme is a model, such as the typing simulation of Rumelhart and Norman (1982), in which the observed timing is determined primarily by peripheral constraints, such as the structure of the hands. With this model, any central control is very indirect, for example by changing the inhibition levels between successive letters. Of course, humans and other creatures can control timing with great precision when the task demands it, and there is no reason to suspect that people could not use central control to maintain constant relative intervals if that is required by the task, as in musical performance. But for most action sequences, the detailed relative timing of components is not a direct part of the task.

Consider the case of a cockroach walking at different rates. The results of Delcomyn (1971) and Pearson (1972) imply that a motion picture of a fast cockroach, when played back in slow motion, would not match the performance of a slow cockroach. Why should this be? The only way for cockroach to walk slower is by decreasing the angular velocity of its leg during the support phase of the step cycle when the leg is on the ground. Therefore the movement of the leg is must be directly related to the speed of walking. In contrast, the leg is not in contact with the ground during the swing phase, and therefore the angular velocity during the swing phase is not directly constrained by speed of walking. In fact, unlike the case for the support phase, the angular velocity for the swing phase is relatively constant, perhaps to be close to a minimum energy for the swing phase or maintain stability by having more feet on the ground when possible. Thus, the ratio of angular velocities for the swing and support phases (and thus the ratio of durations) changes as the walking speed of the cockroach changes. In general, when the details of timing are not an explicit part of the task, they may be determined by other considerations, such as minimizing energy consumption.

The basic point here is that motor performance is an inseparable part of perceptual and cognitive processes, the physical properties of the body, and the environmental context of the task. The timing of any particular instance of motor performance is determined by the interaction of these central and peripheral processes with environmental task. Results from typing illustrate this multi-level control of timing. At the central level, we have seen that typists can vary their overall typing rate. Typists also show effects due to the frequency of words and letter sequences (Laroche, Gentner, & Grudin, 1985; Shaffer, 1973; West & Sabban, 1982). At the peripheral level, finger and hand constraints determine the interstroke intervals of different digraph classes when typing at high speeds. And constraints at both central and peripheral level interact with the task environment, in this case including issues such as the particular text to be typed and the layout of the typewriter keyboard. The relative contribution of all these factors to the final performance depends on the situation. In Experiment 4, when typists were working at the slowest rates, their timing appears to have been determined by a regular central timekeeper and all digraph classes were typed at the same rate. As the typists speeded up, however, peripheral constraints began to play an important role and double and 1-finger digraphs reached their limiting rate, whereas 2-finger and 2-hand digraphs, where overlapped movement is easier, continued to get faster.

Novice typists show a related progression over the course of learning to type. At first their performance appears to be limited by central processes. Instead of following a central timekeeper and typing all digraph classes at the same rate as a slow expert does, novices appear to be limited by planning and memory processes. Novices type 1-finger, 2-finger, and 2-hand digraphs at the same rate, but doubles (where there is no need to retrieve a new location for the second key) are typed twice as fast as the other digraphs (Gentner, 1983). As their skill level increases, novices gradually speed up their typing, and in the process, the control of timing shifts to the peripheral processes typical of expert typists.

Control of timing is thus determined at several levels in the perceptual-cognitive-motor system, and the nature and relative importance of these control levels shifts with skill acquisition and the task environment.

References

- Armstrong, T. R. (1970). *Training for the production of memorized motor patterns* (Tech. Rep. No. 26). Ann Arbor: University of Michigan, Human Performance Center.
- Arshavskii, Y. I., Kots, Y. M., Orlovskii, G. N., Rodionov, I. M., & Shik, M. L. (1965). Investigation of the biomechanics of running by the dog. *Biofizika*, 10, 665-672.
- Carter, M. C., & Shapiro, D. C. (1984). Control of sequential movements: Evidence for generalized motor programs. *Journal of Neurophysiology*, 52, 787-796.
- Clark, F. J., & Euler, C. von (1972). On the regulation of depth and rate of breathing. *Journal of Physiology*, 222, 267-295.
- Clarke, E. F. (1982). Timing in the performance of Erik Satie's 'Vexations'. *Acta Psychologica*, 50, 1-19.
- Delcomyn, F. (1971). The locomotion of the cockroach, *Periplaneta americana*. *Journal of Experimental Biology*, 54, 443-452.
- Fendrick, P. (1937). Hierarchical skills in typewriting. *Journal of Educational Psychology*, 28, 609-620.
- Frohlich, D. M., & Elliott, J. M. (1984). The schematic representation of effector function underlying perceptual-motor skill. *Journal of Motor Behavior*, 16, 40-60.
- Gentner, D. R. (1982). Evidence against a central control model of timing in typing. *Journal of Experimental Psychology: Human Perception and Performance*, 8, 793-810.
- Gentner, D. R. (1983). The acquisition of typewriting skill. *Acta Psychologica*, 54, 233-248.
- Glencross, D. J. (1973). Temporal organization in a repetitive speed skill. *Ergonomics*, 16, 765-776.
- Goslow, G. E., Reinking, R. M., & Stuart, D. G. (1973). The cat step cycle: Hind limb joint angles and muscle lengths during unrestrained locomotion. *Journal of Morphology*, 141, 1-42.
- Grillner, S. (1985). Neurobiological bases of rhythmic motor acts in vertebrates. *Science*, 228, 143-149.
- Grillner, S., Halbertsma, J., Nilsson, J., & Thorstensson, A. (1979). The adaptation to speed in human locomotion. *Brain Research*, 165, 177-182.
- Hollerbach, J. M. (1981). An oscillation theory of handwriting. *Biological Cybernetics*, 39, 139-156.
- Keele, S. W. (1968). Movement control in skilled motor performance. *Psychological Bulletin*, 70, 387-403.

- Keele, S. W., & Summers, J. J. (1976). The structure of motor programs. In G. E. Stelmach (Ed.), *Motor control: Issues and trends* (pp. 109-142). New York: Academic Press.
- Klein, R., Levy, S., & McCabe, J. (1984). The parameter preferences of acquired motor programs for rapid, discrete movements: I. Transfer of training. *Memory & Cognition*, 12, 374-379.
- Kozhevnikov, V., & Chistovich, L. (1965). *Speech: Articulation and perception*. (English translation: J.P.R.S. Washington, D.C., No. JPRS 30543).
- Larochelle, S., Gentner, D. R., & Grudin, J. (1985). *Frequency effects in the transcription of native and foreign language texts*. Manuscript in preparation.
- Lashley, K. S. (1951). The problem of serial order in behavior. In L. A. Jeffress (Ed.), *Cerebral mechanisms in behavior* (pp. 112-136). New York: Wiley.
- Lee, D. N., Lishman, J. R., & Thomson, J. A. (8). Regulation of gait in long jumping. *Journal of Experimental Psychology: Human Perception and Performance*, 448-459.
- Macmillan, D. L. (1975). A physiological analysis of walking in the American lobster (*homarus americanus*). *Philosophical Transactions of the Royal Society of London*, 270, 1-59.
- Michon, J. A. (1974). Programs and "programs" for sequential patterns in motor behavior. *Brain Research*, 71, 413-424.
- Pearson, K. G. (1972). Central programming and reflex control of walking in the cockroach. *Journal of Experimental Biology*, 56, 173-193.
- Prinz, W., & Sanders, A. F. (Eds.) (1984). *Cognition and motor processes*. Berlin: Springer-Verlag.
- Rumelhart, D. E., & Norman, D. A. (1982). Simulating a skilled typist: A study of skilled cognitive-motor performance. *Cognitive Science*, 6, 1-36.
- Schmidt, R. A. (1975). A schema theory of discrete motor skill learning. *Psychological Review*, 82, 225-260.
- Schmidt, R. A. (1982). *Motor control and learning*. Champaign, IL: Human Kinetics Publishers.
- Schmidt, R. A. (1984). *The search for invariance in skilled motor behavior* (Report No. 11/1984). Bielefeld, Federal Republic of Germany: Research Group on Perception & Action at the Center for Interdisciplinary Research (ZiF), University of Bielefeld.
- Schmidt, R. A., Zelaznik, H. N., Hawkins, B., Frank, J. S., & Quinn, J. T. Jr. (1979). Motor-output variability: A theory for the accuracy of rapid motor acts. *Psychological Review*, 86, 415-451.
- Shaffer, L. H. (1973). Latency mechanisms in transcription. In S. Kornblum (Ed.), *Attention and*

- performance IV* (pp. 435-446). New York: Academic Press.
- Shaffer, L. H. (1984). Timing in solo and duet piano performances. *The Quarterly Journal of Experimental Psychology*, 36A, 577-595.
- Shapiro, D. C. (1976). A preliminary attempt to determine the duration of a motor program. In D. M. Landers & R. W. Christina (Eds.), *Psychology of sport and motor behavior I* (pp. 17-24). Champaign, IL: Human Kinetics.
- Shapiro, D. C., & Schmidt, R. A. (1982). The schema theory: recent evidence and developmental implications. In J. A. S. Kelso & J. E. Clark (Eds.), *The development of movement control and coordination* (pp. 113-150). New York: John Wiley.
- Shapiro, D. C., Zernicke, R. F., Gregor, R. J., & Diestel, J. D. (1981). Evidence for generalized motor programs using gait pattern analysis. *Journal of Motor Behavior*, 13, 33-47.
- Sternberg, S., Knoll, R. L., Monsell, S., & Wright, C. E. (1983). *Control of rapid action sequences in speech and typing*. Paper presented at the annual meeting of the American Psychological Association.
- Sternberg, S., Monsell, S., Knoll, R. L., & Wright, C. E. (1978). The latency and duration of rapid movement sequences: Comparisons of speech and typewriting. In G. E. Stelmach (Ed.), *Information processing in motor control and learning* (pp. 117-152). New York: Academic Press.
- Summers, J. J. (1975). The role of timing in motor program representation. *Journal of Motor Behavior*, 7, 229-241.
- Terzuolo, C. A., & Viviani, P. (1979). The central representation of learned motor patterns. In R. E. Talbott & D. R. Humphrey (Eds.), *Posture and movement* (pp. 113-121). New York: Raven Press.
- Terzuolo, C. A., & Viviani, P. (1980). Determinants and characteristics of motor patterns used for typing. *Neuroscience*, 5, 1085-1103.
- Tuller, B., Harris, K. S., & Kelso, J. A. (1981). *Articulatory motor events as a function of speaking rate and stress* (Haskins Laboratories: Status Report on Speech Research SR-65).
- Tuller, B., Kelso, J. A. S., & Harris, K. S. (1981). *Phase relationships among articulator muscles as a function of speaking rate and stress* (Haskins Laboratories: Status Report on Speech Research SR-65).
- Tuller, B., Kelso, J. A. S., & Harris, K. S. (1982). Interarticulator phasing as an index of temporal regularity in speech. *Journal of Experimental Psychology: Human Perception and Performance*, 8, 460-472.
- Viviani, P., & Terzuolo, C. (1980). Space-time invariance in learned motor skills. In G. E. Stelmach & J. Requin (Eds.), *Tutorials in motor behavior* (pp. 525-533).

- West, L. J., & Sabban, Y. (1982). Hierarchy of stroking habits at the typewriter. *Journal of Applied Psychology*, 67, 370-376.
- Williams, T. L. (1977). Tortoise locomotion. *Journal of Physiology (London)*, 270, 54-55.
- Wing, A. M. (1978). Response timing in handwriting. In G. E. Stelmach (Ed.), *Information processing in motor control and learning* (pp. 153-172). New York: Academic Press.
- Wing, A. M. (1980). The long and short of timing in response sequences. In G. E. Stelmach & J. Requin (Eds.), *Tutorials in motor behavior* (pp. 469-486).
- Winter, D. A. (1983). Biomechanical motor patterns in normal walking. *Journal of Motor Behavior*, 15, 302-330.
- Zelaznik, H. N., Schmidt, R. A., Gielen, S. C. A. M., & Milich, M. (1985). *Kinematic properties of rapid aimed hand movements*. Manuscript submitted for publication.

PUBLICATIONS

Technical Reports

- ONR-8001. Donald R. Gentner, Jonathan Grudin, and Eileen Conway. *Finger Movements in Transcription Typing*. May 1980.
- ONR-8002. James L. McClelland and David E. Rumelhart. *An Interactive Activation Model of the Effect of Context in Perception: Part I*. May 1980. Also published in *Psychological Review*, 88, 5, pp. 375-401, 1981.
- ONR-8003. David E. Rumelhart and James L. McClelland. *An Interactive Activation Model of the Effect of Context in Perception: Part II*. July 1980. Also published in *Psychological Review*, 89, 1, pp. 60-94, 1982.
- ONR-8004. Donald A. Norman. *Errors in Human Performance*. August 1980.
- ONR-8005. David E. Rumelhart and Donald A. Norman. *Analogical Processes in Learning*. September 1980. Also published in J. R. Anderson (Ed.), *Cognitive skills and their acquisition*. Hillsdale, NJ: Erlbaum, 1981.
- ONR-8006. Donald A. Norman and Tim Shallice. *Attention to Action: Willed and Automatic Control of Behavior*. December 1980.
- ONR-8101. David E. Rumelhart. *Understanding Understanding*. January 1981.
- ONR-8102. David E. Rumelhart and Donald A. Norman. *Simulating a Skilled Typist: A Study of Skilled Cognitive-Motor Performance*. May 1981. Also published in *Cognitive Science*, 6, pp. 1-36, 1982.
- ONR-8103. Donald R. Gentner. *Skilled Finger Movements in Typing*. July 1981.
- ONR-8104. Michael I. Jordan. *The Timing of Endpoints in Movements*. November 1981.
- ONR-8105. Gary Perlman. *Two Papers in Cognitive Engineering: The Design of an Interface to a Programming System and MENUNIX: A Menu-Based Interface to UNIX (User Manual)*. November 1981. Also published in *Proceedings of the 1982 USENIX Conference*, San Diego, CA, 1982.

- ONR-8106. Donald A. Norman and Diane Fisher. *Why Alphabetic Keyboards Are Not Easy to Use: Keyboard Layout Doesn't Much Matter*. November 1981. Also published in *Human Factors*, 24, pp. 509-515, 1982.
- ONR-8107. Donald R. Gentner. *Evidence Against a Central Control Model of Timing in Typing*. December 1981. Also published in *Journal of Experimental Psychology: Human Perception and Performance*, 8, pp. 793-810, 1982.
- ONR-8201. Jonathan T. Grudin and Serge Larochelle. *Digraph Frequency Effects in Skilled Typing*. February 1982.
- ONR-8202. Jonathan T. Grudin. *Central Control of Timing in Skilled Typing*. February 1982.
- ONR-8203. Amy Geoffroy and Donald A. Norman. *Ease of Tapping the Fingers in a Sequence Depends on the Mental Encoding*. March 1982.
- ONR-8204. LNR Research Group. *Studies of Typing from the LNR Research Group: The role of context, differences in skill level, errors, hand movements, and a computer simulation*. May 1982. Also published in W. E. Cooper (Ed.), *Cognitive aspects of skilled typewriting*. New York: Springer-Verlag, 1983.
- ONR-8205. Donald A. Norman. *Five Papers on Human-Machine Interaction*. May 1982. Also published individually as follows: Some observations on mental models, in D. Gentner and A. Stevens (Eds.), *Mental models*, Hillsdale, NJ: Erlbaum, 1983; A psychologist views human processing: Human errors and other phenomena suggest processing mechanisms, in *Proceedings of the International Joint Conference on Artificial Intelligence*, Vancouver, 1981; Steps toward a cognitive engineering: Design rules based on analyses of human error, in *Proceedings of the Conference on Human Factors in Computer Systems*, Gaithersburg, MD, 1982; The trouble with UNIX, in *Datamation*, 27, 12, November 1981, pp. 139-150; The trouble with networks, in *Datamation*, January 1982, pp. 188-192.
- ONR-8206. Naomi Miyake. *Constructive Interaction*. June 1982.
- ONR-8207. Donald R. Gentner. *The Development of Typewriting Skill*. September 1982. Also published as Acquisition of typewriting skill, in *Acta Psychologica*, 54, pp. 233-248, 1983.
- ONR-8208. Gary Perlman. *Natural Artificial Languages: Low-Level Processes*. December 1982. Also published in *The International Journal of Man-Machine Studies*, 20, pp. 373-419, 1984.
- ONR-8301. Michael C. Mozer. *Letter Migration in Word Perception*. April 1983. Also published in *Journal of Experimental Psychology: Human Perception and Performance*, 9, 4, pp. 531-546, 1983.

- ONR-8302. David E. Rumelhart and Donald A. Norman. *Representation in Memory*. June 1983. To appear in R. C. Atkinson, G. Lindzey, & R. D. Luce (Eds.), *Handbook of experimental psychology*. New York: Wiley (in press).
- ONR-8303. The HMI Project at University of California, San Diego. *User Centered System Design: Part I, Papers for the CHI 1983 Conference on Human Factors in Computer Systems*. November 1983. Also published in A. Janda (Ed.), *Proceedings of the CHI '83 Conference on Human Factors in Computing Systems*. New York: ACM, 1983.
- ONR-8304. Paul Smolensky. *Harmony Theory: A Mathematical Framework for Stochastic Parallel Processing*. December 1983. Also published in *Proceedings of the National Conference on Artificial Intelligence, AAAI-83*, Washington DC, 1983.
- ONR-8401. Stephen W. Draper and Donald A. Norman. *Software Engineering for User Interfaces*. January 1984. Also published in *Proceedings of the Seventh International Conference on Software Engineering*, Orlando, FL, 1984.
- ONR-8402. The UCSD HMI Project. *User Centered System Design: Part II, Collected Papers*. March 1984. Also published individually as follows: Norman, D.A. (1984), Stages and levels in human-machine interaction, *International Journal of Man-Machine Studies*, 21, 365-375; Draper, S.W., The nature of expertise in UNIX; Owen, D., Users in the real world; O'Malley, C., Draper, S.W., & Riley, M., Constructive interaction: A method for studying user-computer-user interaction; Smolensky, P., Monty, M.L., & Conway, E., Formalizing task descriptions for command specification and documentation; Bannon, L.J., & O'Malley, C., Problems in evaluation of human-computer interfaces: A case study; Riley, M., & O'Malley, C., Planning nets: A framework for analyzing user-computer interactions; all published in B. Shackel (Ed.), *INTERACT '84, First Conference on Human-Computer Interaction*, Amsterdam: North-Holland, 1984; Norman, D.A., & Draper, S.W., Software engineering for user interfaces, *Proceedings of the Seventh International Conference on Software Engineering*, Orlando, FL, 1984.
- ONR-8403. Paul Smolensky and Mary S. Riley. *Harmony Theory: Problem Solving, Parallel Cognitive Models, and Thermal Physics*. April 1984. The first two papers appear in *Proceedings of the Sixth Annual Meeting of the Cognitive Science Society*, Boulder, CO, 1984.
- ONR-8404. Donald R. Gentner. *Expertise in Typewriting*. April 1984. To appear in M. T. Chi (Ed.), *The nature of expertise* (in press).
- ONR-8405. Michael C. Mozer. *Inductive Information Retrieval Using Parallel Distributed Computation*. May 1984.
- ONR-8406. David E. Rumelhart and David Zipser. *Feature Discovery by Competitive Learning*. July 1984. Also published in *Cognitive Science*, 1985, 9, 75-112.

AD-A157 665

HUMAN COGNITION AND PERFORMANCE(U) CALIFORNIA UNIV SAN
DIEGO LA JOLLA INST FOR COGNITIVE SCIENCE
D A NORMAN ET AL. 01 MAY 85 N00014-79-C-0323

2/2

UNCLASSIFIED

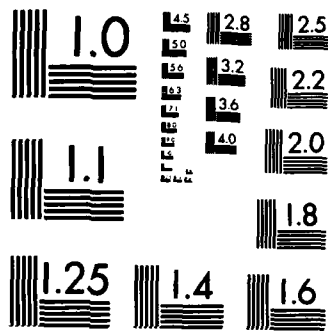
F/G 5/10

NL

END

FIMED

DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

Archival Publications

- Miyake, N., & Norman, D. A. (1979). To ask a question, one must know enough to know what is not known. *Journal of Verbal Learning and Verbal Behavior*, 18, 357-364.
- Norman, D. A. (1979). Copycat science. In R. A. Cole (Ed.), *Perception and production of fluent speech*. Hillsdale, NJ: Erlbaum.
- Norman, D. A. (1979). Perception, memory, and mental processes. In L. G. Nilsson (Ed.), *Perspectives on memory research*. Hillsdale, NJ: Erlbaum.
- Norman, D. A., & Bobrow, D. G. (1979). Descriptions: An intermediate stage in memory retrieval. *Cognitive Psychology*, 11, 107-123.
- Norman, D. A. (1980). Cognitive engineering and education. In D. T. Tuma & F. Reif (Eds.), *Problem solving and education: Issues in teaching and research*. Hillsdale, NJ: Erlbaum.
- Norman, D. A. (1980). Teaching, learning, and the representation of knowledge. In R. E. Snow, P. A. Federico, & W. E. Montague (Eds.), *Aptitude, learning, and instruction: Vol. 2. Cognitive process analyses of learning and problem solving*. Hillsdale, NJ: Erlbaum.
- Norman, D. A. (1980). Twelve issues for cognitive science. *Cognitive Science*, 4, 1-32.
- Norman, D. A. (1980). What goes on in the mind of the learner? In W. J. McKeachie (Ed.), *Cognition, college teaching, and student learning*. San Francisco: Jossey-Bass.
- Norman, D. A. (1980). Why people make mistakes. *Reader's Digest*, 117, 103-106.
- Perlman, G. (1980). Data analysis programs for the UNIX operating system. *Behavior Research Methods and Instrumentation*, 12, 554-558.
- Rumelhart, D. E. (1980). On evaluating story grammars. *Cognitive Science*, 4, 313-316.
- Rumelhart, D. E. (1980). Schemata: The building blocks of cognition. In R. Spiro, B. Bruce, & W. Brewer (Eds.), *Theoretical issues in reading comprehension*. Hillsdale, NJ: Erlbaum.
- Erickson, T. D., & Mattson, M. E. (1981). From words to meaning: A semantic illusion. *Journal of Verbal Learning and Verbal Behavior*, 20, 540-551.
- McClelland, J. L., & Rumelhart, D. E. (1981). An interactive activation model of the effect of context in perception: Part 1. An account of basic findings. *Psychological Review*, 88, 375-407.
- Norman, D. A. (1981). Categorization of action slips. *Psychological Review*, 88, 1-15.
- Norman, D. A. (Ed.). (1981). *Perspectives on cognitive science: The La Jolla Conference*. Published

jointly by Erlbaum and Ablex. Norwood, NJ: Ablex. Hillsdale, NJ: Erlbaum.

Norman, D. A. (1981). A psychologist views human processing: Human errors and other phenomena suggest processing mechanisms. *Proceedings of the International Joint Conference on Artificial Intelligence*, Vancouver.

Norman, D. A. (1981). The trouble with UNIX: The user interface is horrid. *Datamation*, 27, 138-150.

Norman, D. A., & Rumelhart, D. E. (1981). The LNR approach to human information processing. *Cognition*, 10, 235-240.

Rumelhart, D. E. (1981). Analogical processes in learning. In J. R. Anderson (Ed.), *Cognitive skills and their acquisition*. Hillsdale, NJ: Erlbaum.

Rumelhart, D. E., & Norman, D. A. (1981). An activation-trigger-schema model for the simulation of skilled typing. *Proceedings of the Berkeley Conference of the Cognitive Science Society*.

Rumelhart, D. E., & Norman, D. A. (1981). Parallel associative models of memory: Where do they fit? In G. Hinton & J. Anderson (Eds.), *Parallel associative models of memory*. Hillsdale, NJ: Erlbaum.

Gentner, D. R. (1982). Acquisition of typewriting skill. *Acta Psychologica*, 54, 233-248.

Gentner, D. R. (1982). Evidence against a central control model of timing in typing. *Journal of Experimental Psychology: Human Perception and Performance*, 8, 793-810.

Gentner, D. R. (1982). Testing the central control model of typing: Comments on the reply by Viviani and Terzuolo. *Journal of Experimental Psychology: Human Perception and Performance*, 8, 814-816.

Miyake, N. (1982). Mental models. *Psychology*, 3 (3), 12-19. (In Japanese.)

Norman, D. A. (1982). Comments on cognitive engineering, the need for clear "system images," and the safety of nuclear power plants. *Proceedings of IEEE Standards Workshop on Human Factors and Nuclear Safety*. New York: IEEE.

Norman, D. A. (1982). How to make computer systems that people will like to use: Steps toward a cognitive engineering. *Proceedings of the Western Electronic Show and Convention (Wescon/82)*.

Norman, D. A. (1982). *Learning and memory*. San Francisco: Freeman. (Translations: Japanese, in progress.)

Norman, D. A. (1982). Steps towards an cognitive engineering: Design rules based on analyses of human error. *Proceedings of the Conference on Human Factors in Computer Systems*, Gaithersburg, MD.

- Norman, D. A. (1982). The trouble with networks. *Datamation*, xx, 188-192.
- Perlman, G. (1982). Data analysis in the UNIX environment: Automated experimental design specification for data analysis programs. *Proceedings of the XIV Conference on the Interface*, Troy, NY.
- Perlman, Gary, (1982). Experimental control programs for the UNIX operating system. *Behavior Research Methods and Instrumentation*, 14, 417-421.
- Rumelhart, D. E., & McClelland, J. L. (1982). An interactive activation model of the effect of context in perception: Part 2. The contextual enhancement effect and some test and extensions of the model. *Psychological Review*, 89, 60-94.
- Gentner, D.R. (1983). Keystroke timing in transcription typing. In W. E. Cooper (Ed.), *Cognitive aspects of skilled typewriting*. New York: Springer-Verlaag.
- Gentner, D. R., Grudin, J., Larochelle, S., Norman, D. A., & Rumelhart, D. E. (1983). A glossary of terms, including a classification of typing errors. In W. E. Cooper (Ed.), *Cognitive aspects of skilled typewriting*. New York: Springer-Verlag.
- Grudin, Jonathan T. (1983). Error patterns in skilled and novice transcription typing. In W. E. Cooper (Ed.), *Cognitive aspects of skilled typewriting*. New York: Springer-Verlag.
- Larochelle, Serge (1983). A comparison of skilled and novice performance in discontinuous typing. In W.E. Cooper (Ed.), *Cognitive aspects of skilled typewriting*. New York: Springer-Verlag.
- Mozer, Michael C. (1983). Letter migration in word perception. *Journal of Experimental Psychology: Human Perception and Performance*, 9, 531-546.
- Norman, D. A. (1983). Design principles for human-computer interfaces. In A. Janda (Ed.), *Proceedings of the CHI '83 Conference on Human Factors in Computing Systems*. New York: ACM.
- Norman, D. A. (1983). Design rules based on analyses of human error. *Communications of the ACM*, 4, 254-258.
(Also translated into Japanese and printed in *Psychology* (1983), 41, No. 8, 62-69.)
Originally printed in *Proceedings of the Conference on Human Factors in Computer Systems*. (March, 1982). Gaithersberg, MD: Institute for Computer Sciences and Technology, National Bureau of Standards, U.S. Department of Commerce.
- Norman, D.A. (1983). Some observations on mental models. In D. Gentner & A. Stevens (Eds.), *Mental Models*. Hillsdale, NJ: Erlbaum.
- Norman, D.A. (1983). Theories and models in cognitive psychology. In E. Donchin (Ed.), *Cognitive psychophysiology*. Hillsdale, NJ: Erlbaum.
- Norman, D. A. (1983). Worsening the knowledge gap: The mystique of computation builds

- unnecessary barriers. *Annals of the New York Academy of Sciences*, 246, 220-233.
- Norman, D. A., (1983, September). The DVORAK revival: Is it really worth the cost? *cpnews: Consumer Products Tech Group: the Human Factors Society*. 8, No. 3, 5-7.
- Norman, D. A. (1983, October). The computer always rings twice. *Psychology Today*, pp. 46-50.
- Norman, D. A., & Rumelhart, D. E. (1983). Studies of typing from the LNR Research Group. In W. E. Cooper (Ed.), *Cognitive aspects of skilled typewriting*. New York: Springer-Verlag.
- Rumelhart, D. E., & McClelland, J. L. (1983). Interactive processing through spreading activation. In C. Perfetti & A. Lesgold (Eds.), *Interactive processes in reading*. Hillsdale, NJ: Erlbaum.
- Compton, M. (1984, November). Don Norman speaks out: An interview with the UNIX system's best known critic. *UNIX Review*, pp. 38-49, 94-96.
- Draper, S.W., & Norman, D. A. (1984). Software engineering for user interfaces. *Proceedings of the 7th International Conference on Software Engineering*. Orlando, FL.
- Gentner, D. R., & Norman, D. A. (1984, March). The typist's touch. *Psychology Today*, pp. 66-72.
- Norman, D. A. (1984). Cognitive engineering principles in the design of human-computer interfaces. In G. Salvendy (Ed.), *Human-Computer Interaction: Proceedings of the First USA-Japan Conference on Human-Computer Interaction (Honolulu, August 1984)*. New York: Elsevier.
- Norman, D. A. (1984). Stages and levels in human-machine interaction. *International Journal of Man-Machine Studies*, 21, 365-375.
- Perlman, Gary, (1984). Natural artificial languages: Low-level processes. *International Journal of Man-Machine Studies*, 20, 373-419.
- Riley, M. S., & Smolensky, P. (1984). A parallel model of (sequential) problem solving. *Proceedings of the Sixth Annual Conference of the Cognitive Science Society*. Boulder, CO.
- Rumelhart, D. E. (1984). Understanding understanding. In J. Flood (Ed.), *Understanding reading comprehension*. Newark, DE: International Reading Association.
- Smolensky, P. (1984). The mathematical role of self-consistency in parallel computation. *Proceedings of the Sixth Annual Conference of the Cognitive Science Society*. Boulder, CO.
- Bannon, L. J., & O'Malley, C. E. (1985). Problems in evaluation of human-computer interfaces: A case study. In B. Shackel (Ed.), *INTERACT '84: First Conference on Human-Computer Interaction*. Amsterdam: North-Holland.
- Draper, S. W. (1985). The nature of expertise in UNIX. In B. Shackel (Ed.), *INTERACT '84: First*

Conference on Human-Computer Interaction. Amsterdam: North-Holland.

- Norman, D. A. (1985). Stages and levels in human-machine interaction. In B. Shackel (Ed.), *INTERACT '84: First Conference on Human-Computer Interaction*. Amsterdam: North-Holland.
- O'Malley, C. E., Draper, S. W., & Riley, M. S. (1985). Constructive interaction: A method for studying user-computer-user interaction. In B. Shackel (Ed.), *INTERACT '84: First Conference on Human-Computer Interaction*. Amsterdam: North-Holland.
- Owen, D. (1985). Users in the real world. In B. Shackel (Ed.), *INTERACT '84: First Conference on Human-Computer Interaction*. Amsterdam: North-Holland.
- Riley, M. S., & O'Malley, C. E. (1985). Planning nets: A framework for analyzing user-computer interactions. In B. Shackel (Ed.), *INTERACT '84: First Conference on Human-Computer Interaction*. Amsterdam: North-Holland.
- Rumelhart, D. E., & Zipser, D. (1985). Feature discovery by competitive learning. *Cognitive Science*, 9, 75-112.
- Smolensky, P. Monty M. L., & Conway, E. (1985). Formalizing task descriptions for command specification and documentation. In B. Shackel (Ed.), *INTERACT '84: First Conference on Human-Computer Interaction*. Amsterdam: North-Holland.
- McClelland, J. L., & Rumelhart, D. E. (in press). *Parallel distributed processing: Explorations in the microstructure of cognition. Vol. 2: Applications*. Cambridge, MA: MIT Press/Bradford Books.
- Norman, D. A., & Draper, S. W. (in press). *User center system design: New perspectives in human-computer interaction*. Hillsdale, NJ: Erlbaum.
- Rumelhart, D. E., & McClelland, J. L. (in press) *Parallel distributed processing: Explorations in the microstructure of cognition. Vol. 1: Foundations*. Cambridge, MA: MIT Press/Bradford Books.
- Rumelhart, D.E., & Norman, D.A. (in press). Representation in memory. In R.C. Atkinson, R.J. Herrnstein, G. Lindzey, & R.D. Luce (Eds.), *Handbook of Experimental Psychology*. NY: Wiley.

ONR Distribution List

E. Aiken
Navy Personnel Research
& Development Center
San Diego, CA 92152

Robert Bachman
ONR, Q-043
University of California, San Diego
La Jolla, CA 92093

Susan Chipman
Personnel & Training Research (Code 442PT)
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Charles E. Davis
Personnel & Training Research (Code 442PT)
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Craig Fields
Advanced Research
Projects Agency
1400 Wilson Blvd.
Arlington, VA 22209

Eugene Gloye
ONR Branch Office
1030 E. Green St.
Pasadena, CA 91101

James Hollan
Code 13
NPRDC
San Diego, CA 92152

James Lester
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Harold F. O'Neil, Jr.
Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

Joe Psotka
Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

Michael Shafto
Personnel & Training Research (Code 442PT)
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

John Tanguay
Program Manager
Life Sciences Directorate
Bldg 410
AFOSR/NL
Bolling Air Force Base
Washington, DC 20332

Jack Thorpe
Advanced Research Projects
Agency
1400 Wilson Blvd.
Arlington, VA 22209

Wallace Wulfeck
Code 13
NPRDC
San Diego, CA 92152

END

FILMED

9-85

DTIC