

UNCLASSIFIED

AD NUMBER
AD354615
NEW LIMITATION CHANGE
TO Approved for public release, distribution unlimited
FROM Distribution authorized to DoD only; Administrative/Operational Use; DEC 1962. Other requests shall be referred to Rome Air Development Center, Attn: EMIIT, Griffiss AFB, NY.
AUTHORITY
AFRL/IFOIP ltr, 4 Jan 2007

THIS PAGE IS UNCLASSIFIED

UNCLASSIFIED

AD NUMBER
AD354615
CLASSIFICATION CHANGES
TO
unclassified
FROM
confidential
AUTHORITY
AFRL/IF Rome NY, ltr, 28 Aug 2006

THIS PAGE IS UNCLASSIFIED

UNCLASSIFIED

AD NUMBER
AD354615
CLASSIFICATION CHANGES
TO
confidential
FROM
secret
AUTHORITY
31 Dec 1974 per DoDD 5200.10 document marking

THIS PAGE IS UNCLASSIFIED

AD 3 5 4 6 1 5 L

DEFENSE DOCUMENTATION CENTER

FOR

SCIENTIFIC AND TECHNICAL INFORMATION

CAMERON STATION, ALEXANDRIA VIRGINIA



SECRET

NOTICE: When government or other drawings, specifications or other data are used for any purpose other than in connection with a definitely related government procurement operation, the U. S. Government thereby incurs no responsibility, nor any obligation whatsoever; and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use or sell any patented invention that may in any way be related thereto.

NOTICE:

THIS DOCUMENT CONTAINS INFORMATION
AFFECTING THE NATIONAL DEFENSE OF
THE UNITED STATES WITHIN THE MEAN-
ING OF THE ESPIONAGE LAWS, TITLE 18,
U.S.C., SECTIONS 793 and 794. THE
TRANSMISSION OR THE REVELATION OF
ITS CONTENTS IN ANY MANNER TO AN
UNAUTHORIZED PERSON IS PROHIBITED
BY LAW.

~~CONFIDENTIAL~~

AD354615

DECLASSIFIED

UNCLASSIFIED

FACT CORRELATION FOR INTELLIGENCE ANALYSIS. VOLUME 2.
ANALYSIS OF TECHNICAL PROBLEMS (U)

FEDERAL ELECTRIC CORP PARAMUS NJ

15 DEC 1962

Distribution authorized to DoD only; Administrative/Operational
Use; DEC 1962. Other requests shall be referred to Rome Air
Development Center, Attn: EMIIT, Griffiss AFB, NY. NOFORN

Declassified by:
AFRLIF
Donald W. Hanson
Director, Information Directorate,
Declassified on: August 25, 2006

Notice

There has been a classification
change to this document. It is the
responsibility of the recipient to
promptly remark it to indicate
change.

DECLASSIFIED

UNCLASSIFIED

~~CONFIDENTIAL~~

SECURITY NOTICE

Special Handling Required. Not Releasable to Foreign Nationals.

CLASSIFICATION NOTICE

This document is classified SECRET because it discusses in detail the limitations inherent in techniques for manipulating data available to present intelligence system environments and implies a course of technique development for future application to intelligence data handling environments.

AVAILABILITY NOTICE

Military agencies may request copies from DDC, all other qualified requests must be submitted through RADC (EMIIT), Griffiss AFB, NY.

PATENT NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

SECRET

RADC-TDR-62-461, Vol. II

15 December 1962

Final Report
(U) FACT CORRELATION FOR INTELLIGENCE ANALYSIS
Volume 2: Analysis of Technical Problems



International Electric Corporation

Route 17 and Garden State Parkway, Paramus, New Jersey

A SUBSIDIARY OF INTERNATIONAL TELEPHONE AND TELEGRAPH CORPORATION

Technical Report P-AA-TR-(0008)

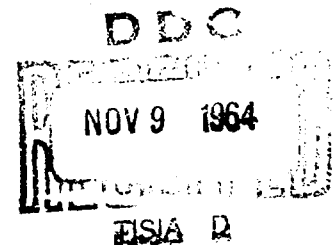
Security No: IEO 62-P-69

Copy No. 44 of 60 Copies

Contract No. AF 30(602)-2739

Project No. 4594, Task No. 459403

Prepared for
ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
United States Air Force
Griffiss Air Force Base New York



05403

DOWNGRADED AT 12 YEAR INTERVALS; NOT AUTOMATICALLY DECLASSIFIED. DOD EIR 5200.10.

This document contains information affecting the material defense of the United States, within the meaning of the Espionage Laws, Title 18, U.S.C., Section 793 and 794, the transmission or revelation of which in any manner to unauthorized persons is prohibited by law.

SECRET

3 S 4 15

CONFIDENTIAL

FOREWORD

This volume is the second of two volumes in a report analyzing the requirements for an applied research program for an automated system capable of correlating facts for intelligence analysis. This volume presents an analysis and evaluation of the technical problems pertaining to both equipment and techniques. (C)

This report was prepared by the International Electric Corporation (IEC)--a subsidiary of the International Telephone and Telegraph Corporation (ITT)--Paramus, New Jersey. The study leading to this report was performed under Contract No. AF 30(602)-2739 for the Intelligence and Electronic Warfare Directorate of the Rome Air Development Command, Griffiss Air Force Base, New York. (U)

This study was conducted within the Advanced Analysis Department under the direction of Jacques Harlow. The staff that performed the analytical and theoretical studies included Quentin A. Darmstadt, Dr. George Greenberg, Maralyn W. Lindenlaub, David M. Massie, Dr. Howard E. Smokler, Alexander Szejman, and Alfred Trachtenberg. (U)

The staff acknowledges the contribution of the authors cited as references and of the equipment manufacturers who supplied information pertaining to their present equipment design specifications and their future plans for equipment development. The report also contains original concepts developed by members of the staff while performing research activities sponsored by IEC. (U)

CONFIDENTIAL

SECRET

ABSTRACT

This report presents an analysis of the requirements for a Fact Correlation System--a specialized information storage and retrieval system oriented to the problem of correlating facts for intelligence analysis. Two aspects of the problem were analyzed: the equipment requirements necessary to implement a highly automated system, and the processing requirements necessary to transmute isolated items of information into an integral whole. The report includes a broad review and survey of existing processing techniques, primarily in the fields of linguistic analysis and adaptive learning, and a survey of both existing and experimental data processing equipment. This survey and analysis form the basis for a research plan for developing the techniques and equipment required to correlate facts automatically. (S)

This volume of the report contains a detailed analysis of techniques and equipment, which were discussed in general terms in Volume 1. The functions of linguistic transformation and adaptive learning are described, and existing research is reviewed in terms of its ability to fulfill these functions. Programming requirements are cursorily reviewed, since no specific problems have been discerned. An exhaustive analysis of equipment functions leads to a recommended system configuration in terms of existing components. Design principles and exploratory research have also been reviewed to anticipate possible areas of directed research activity for improving system operations. (U)

SECRET

TABLE OF CONTENTS

<u>Part</u>	<u>Title</u>	<u>Page</u>
	FOREWORD	iii
	ABSTRACT	v
	LIST OF ILLUSTRATIONS	xi
	LIST OF TABLES	xiii
I	<u>INTRODUCTION</u>	1
	A. Purpose	1
	B. Scope	3
	C. Background	3
	D. Organization of Report	4
II	<u>LINGUISTIC TRANSFORMATIONS</u>	5
	A. Statement of the Problem	5
	B. General Functional Requirements	5
	C. Review of Present Techniques	7
	1. Word Recognition	8
	2. Syntactic Analysis	9
	3. Semantic Analysis	21
	4. Automatic Dictionaries	35
	D. Research Approach	38
	1. Recognition of Well- or Ill-Formed Constructions	38
	2. Detection and Resolution of Ambiguity	42
	3. Generation of Maximally Useful Output	45
	4. Evaluation of the Adequacy of Language Analysis and Synthesis	47
	E. Summary	47
	F. References	48
III	<u>ADAPTIVE CORRELATION TECHNIQUES</u>	53
	A. Statement of the Problem	53
	B. General Functional Requirements	54
	1. General	54
	2. Ordering and Storage of Input Data	55

TABLE OF CONTENTS (Continued)

<u>Part</u>	<u>Title</u>	<u>Page</u>
III	3. Retrieval and Correlation of Data	56
	4. The Adaptive Nature of the System	57
	C. Review of Present Techniques	59
	1. General Classifications	60
	2. Perceptron Theory	64
	3. General Problem Solver	68
	4. Pandemonium	78
	5. Experiments in Machine Learning	82
	6. The Theory of Automata	87
	7. Summary and Conclusions	96
	D. Research Problems	98
	1. Ordering and Storage of Data	99
	2. Retrieval and Correlation of Data	104
	3. Learning Processes	112
	E. Summary	122
	F. References	123
IV	<u>COMPUTER PROGRAMMING</u>	129
	A. General	129
	B. Executive Monitoring and Control	132
	C. Parallel Programming	137
	D. Assemblers and Compilers	140
	1. Symbolic Language versus Machine Language	141
	2. Automatic Programming Systems	142
	3. Use of Subroutines	143
	E. Programming Languages	145
	1. Advantages of Programming Languages	145
	2. FORTRAN, ALGOL, and COBOL	146
	3. Comparison of Programming Languages	147
	4. Recursive Subroutines and ALGOL	148
	F. List Processing	149
	1. List Storage Techniques	151

TABLE OF CONTENTS (Continued)

<u>Part</u>	<u>Title</u>	<u>Page</u>
IV	2. Problem of List Erasure	153
	3. Types of List Structures	154
	4. List Processing Languages	155
G.	Storage Allocation Techniques	158
	1. Dynamic Control	159
	2. Algorithms	160
	3. Logic Structure Tables	161
H.	Adaptive Systems	162
I.	Composition of Programs	163
J.	Summary	166
K.	References	167
V	<u>EQUIPMENT EVALUATION</u>	171
A.	Objectives and Methodology	171
B.	Performance Criteria	173
	1. Volume of Data	173
	2. Speed	174
	3. Design	176
	4. Reliability	177
	5. Cost	178
C.	Parameters and Variables for Configurations	179
	1. First Method	179
	2. Second Method	180
D.	Analysis of Configurations	194
	1. Input Restrictions	194
	2. Basis Computer Processing Requirements and Limits	196
	3. Elimination of Unfeasible Equipment Configurations	198
	4. Possible Equipment Configurations	198
	5. Interpretation of Configurations	199
	6. Evaluation of System Configurations	212

TABLE OF CONTENTS (Continued)

<u>Part</u>	<u>Title</u>	<u>Page</u>
V	E. Comparative Operations	217
	1. Input Devices	218
	2. Output Devices	227
	3. Storage Devices	230
	4. Computing Units	248
	F. Development Requirements	254
	1. Speech and Pattern Recognition	256
	2. Storage Devices	260
	3. Computer Components and Techniques	278
	G. Summary	280
	H. References	282
	<u>DISTRIBUTION LIST</u>	285

LIST OF ILLUSTRATIONS

<u>Figure</u>	<u>Title</u>	<u>Page</u>
2-1	The Language Tree of Semantic Structure	31
3-1	State Graph for a Finite Automaton	90
4-1	Programming System for Fact Correlation	136
5-1	Reception Rate versus Time	181
5-2	Basic System Concepts	195
5-3	System Configuration 1	201
5-4	System Configuration 2	202
5-5	System Configuration 3	203
5-6	System Configuration 4	204
5-7	System Configuration 5	207
5-8	System Configuration 6	208
5-9	System Configuration 7	210
5-10	System Configuration 8	211
5-11	Recommended State-of-the-Art Configuration	214
5-12	Storage Capacity versus Access Time	216
5-13	Block Diagram of Matrix Reader	220
5-14	Character Recognition Sequence	221
5-15	General Purpose Flying Head	236
5-16	Thin Film Memory Planes	247
5-17	Synchronous and Asynchronous Operations	253
5-18	Gryotron Flip-Flop	273
5-19	Basic Hydraulic Logic Elements	277

LIST OF TABLES

<u>Table</u>	<u>Title</u>	<u>Page</u>
5-1	Input Equipments and Functions	182
5-2	Processing Time for a Single Document	185
5-3	Part I: Storage Allocations	187
	Part II: Storage Allocations	188
5-4	Part I: Total Processing Times	190
	Part II: Total Processing Times	191
	Part III: Total Processing Times	192
	Part IV: Total Processing Times	193
5-5	Drum Memories	239
5-6	Part I: Disc Files	241
	Part II: Disc Files	242
5-7	Tape Units	244
5-8	Tape Units	245

SECRET

I. INTRODUCTION

A. Purpose

The purpose of this report is to present the results of an analysis of the requirements for a Fact Correlation System--a specialized information storage and retrieval system oriented to the specific problem of correlating facts for intelligence analysis. Secondly, the report describes a general research plan for developing the techniques and equipment for implementing such a system. (S)

Although specifically oriented to intelligence analysis, this study considered the problem of fact correlation within a generalized framework. The basic assumption underlying this study was that information is more important than documents. It follows that it is the information about an event or an item of knowledge from a set of documents that must be stored and retrieved. This concept is equally applicable to a variety of problems in information flow and decision making as well as to the specific problem of intelligence analysis. (U)

The elements of functional requirements were analyzed and are presented within the context of a system concept. This system was defined as:

An information system is imbedded within an environment of data or intelligence information. The system consists of three operational functions--personnel, equipment, and techniques (including computer programming) together with the interactions among each function--required to interpret, rationalize, and understand communications from its environment.

The purpose of an information system is to extend the performance and effectiveness of individuals, including intelligence analysts, interacting within the frame of reference of the system.

The nature of this problem is such that a definitive solution cannot be

SECRET

SECRET

produced within a limited period of time. The primary purpose of this study, therefore, was to stipulate the research activity that would contribute to the solution of the problem. (S)

This volume of the report reviews the existing techniques of linguistic analysis and adaptive learning, which have been designated as the principal problems in developing an automated system for correlating facts. Language is the medium for expressing information or facts. Thus an automated system requires processes for discerning the properties of language, the function of linguistic analysis, and manipulating the elements of language, the function of adaptive learning. Besides reviewing existing techniques, the report postulates potentially fruitful areas of research, especially in semantic analysis, and anticipates the difficulties that are likely to arise between the formulation of a theoretical framework and the implementation of a system. (U)

A secondary aspect of this volume is a review of existing equipment within the scope of a system configuration. This configuration was selected after analyzing a variety of models developed from a series of estimated or assumed operational parameters. The initial equipment configuration is based upon components that are presently available or in the final stages of testing. The principles of equipment design were also analyzed together with experimental applications of these principles in anticipation of potential research to improve the operational characteristics of equipment within the next decade. (U)

SECRET

B. Scope

The analysis of the requirements for a Fact Correlation System is clearly limited. This study covered research that has been performed and that should be performed within the context of the conceptual framework of the system. This report does not purport to present substantive answers or formal solutions to the problem of correlating facts. (U)

The results of this study consist of two elements:

- (a) A Research Program Plan - a plan for specific research leading to a Fact Correlation System. The plan includes a procedure for conducting research, a perspective of the interrelationship among tasks, an over-view of program scheduling and phasing, expected results, and manpower requirements and qualifications.
- (b) An Analysis of Technical Problems - the analysis of techniques, equipments, and technical problems includes a discussion of alternate approaches, the recommended approach together with a rationale for this recommendation, and a listing and discussion of basic tasks.

These elements of the study are reported in terms of what has been done, what should be done, and why it should be done. The review of past activity is critical; but the critique is bounded by the framework established for the analytic study. The concept presented for a future system is feasible; yet the conceptual system may be revised, and should be, as research differentiates between the ideal and practical limits of theory. (U)

C. Background

The background for this analytical study is presented in Volume 1, Applied Research Plan. (U)

D. Organization of Report

This report consists of two volumes. The first contains a general analysis and description of requirements. The second describes available and exploratory techniques and equipment; it also anticipates several problem areas. (U)

Volume 1 presents the Applied Research Plan. The volume includes a discussion of basic concepts of information retrieval; a review of system requirements for a Fact Correlation System; and a general plan and schedule for research, development, and implementation leading to the installation of an operational system for correlating facts. (U)

This volume of the report contains a detailed analysis of techniques and equipment that are discussed generally in Volume 1. The functions of linguistic transformation and adaptive learning are described, and existing research is reviewed in terms of its ability to meet these functions. Programming requirements are cursorily reviewed; the major problems in this area can only be discerned after the nature of the programming tasks has been stipulated as a result of research in linguistic and learning. The review of equipment functions and capabilities evolves into a recommended system configuration. Design principles and exploratory research have also been reviewed to indicate areas of research activity that could be directed to the improvement of particular equipment functions. (U)

II. LINGUISTIC TRANSFORMATIONS

A. Statement of the Problem

The objective of the language transformation function of the Fact Correlation System is to produce a symbolic output more amenable to automatic correlation techniques than the ordinary language inputs to the system. This problem is not restricted to emphasis on any one level of language, but crosses the spectrum of linguistic levels of formation from the morphological through the syntactic, semantic, and logical to the factual. At each of these levels, the linguistic transformation processes must be supplied with techniques to:

- (a) Recognize well- or ill-formed constructions.
- (b) Detect and resolve ambiguous references and formations.
- (c) Generate maximally useful factual output, both explicit and implicit.
- (d) Evaluate the adequacy of linguistic analysis and synthesis.

The concept of levels indicates a strategy for studying significant aspects of linguistic analysis in clearly delimited segments of the over-all problem. (U)

B. General Functional Requirements

The language transformation processes should facilitate the development of an effective Fact Correlation System by allowing the system to accept sentences of ordinary language as inputs. It is assumed that such sentences will be in English. These processes will be designed to process linguistic inputs semi-automatically, with a minimum of pre-editing or annotating, directly from standard typographic forms if at all

possible. If foreign language inputs are desired, the problem is not altered in principle except for the insertion of a translation step. Dealing with non-textual inputs, however, is best considered as a separate, ancillary problem. (U)

The language transformation processes should produce a formal representation of the input text. The ultimate criteria for this formal representation are to possess the following characteristics:

- (a) Fidelity to the factual content of the linguistic inputs, including:
 - (1) Retention of all factual content, both explicit and implicit.
 - (2) No unjustified additions or elaborations to the factual content of the input. (Any extrapolation of information is an inferential process more properly assigned to the adaptive learning function.)
- (b) Checking of the linguistic input to assure that it does not contain unintended errors at the morphemic, syntactic, semantic, logical, and factual levels. (Obviously, these checks become increasingly difficult at successively higher levels of the linguistic scale; the resolution of this difficulty depends, in part, upon interaction with human beings.)
- (c) Resolution of apparent or actual ambiguous and ill-formed statements, including inconsistent and contra-factual linguistic inputs.
- (d) "Simple" rendering of factual input, including the reduction or elimination of unnecessary redundancy.
- (e) Compatibility with the processing requirements and capabilities of subsequent functions of the system. (U)

The complete automation of these requirements does not seem feasible within the current state-of-the-art. Even if the scanning of ordinary texts can be mechanized, two essential human decision functions are

envisaged for the language transformation processes:

- (a) Selecting (annotating and editing as necessary) the material to be processed.
- (b) Responding to machine queries on problems of formation and ambiguity. (U)

An ancillary requirement of the language transformation function is to enable the Fact Correlation System to communicate responses and requests to the users in a comprehensible form. For this purpose it will be necessary to develop question and sentence-generating processes. The essential theoretical groundwork for such computer-initiated communications may be established as an adjunct to the solution of the more difficult problem of analyzing input statements. (U)

C. Review of Present Techniques

The transformation of natural language to a form suitable for processing by an automatic Fact Correlation System requires that a computer be programmed to recognize ordinary written language at each of its successive levels of formation. The procedures must reject or correct possible ill-formed construction (misspelling, grammatical incorrectness, and meaninglessness) and clarify ambiguity at each level of recognition. (U)

The transformation process must be information-preserving. Since the language processed as data has no experiential reference for the computer, the information content of such data is a formal function of syntactic and semantic structure. The process preserves or represents those syntactic and semantic characteristics of the input that constitute the information. The transformation process performs a syntactic and semantic

analysis of input statements in natural language and transforms the language to another symbolic form that is equivalent to the original in information content. (U)

The problem of word recognition, the identification of letter sequences, is essentially distinct from the higher level problems of syntactic and semantic analysis; the latter, however, are closely related. Extensive research on recognition and analysis of natural language has been conducted in the last decade as part of the effort to develop techniques for machine translation. The analytic function required for machine translation is the same as that which would be required by a Fact Correlation System. This section reviews the techniques of linguistic analysis developed during the past decade; any techniques developed prior to 1950 are not seriously considered, since they tend to be inadequate in meeting the requirements imposed by mechanization. (U)

1. Word Recognition

Printed language is a linear sequence of a small number of signs. The recognition of an input statement in this linguistic form begins with the recognition of such signs--the lowest level of language structure. (U)

The initial recognition function is performed by a scanning device that reads input statements, which are subsequently transformed to an appropriate digital code. (It is assumed that such input statements will be typewritten or printed, single-spaced copy, perhaps as small as newsprint.) Words, or letter sequences, in the input statements are then compared to a dictionary stored by the computer. A word is identified

if such comparison indicates that the word is known by the computer; that is, if the word corresponds to a word in the dictionary at that time. In cases of morphological ambiguity, identification is determined by subsequent syntactic and semantic analyses. The entry describing a word in a computer dictionary for syntactic analysis is not a list of synonyms, but a description of the grammatical and semantic functions of the word, according to the methods of analysis used by the system. (U)

It is possible to program a computer to add new words to its vocabulary automatically, assuming that such words are correctly spelled. (An incorrectly spelled word would also appear as a new word; this problem may be resolved by having the computer request verification of a new word before accepting it.) The computer should be able to determine the grammatical function and semantic relationships of new words from an analysis of the sentences in which they appear. However, the delineation of a new word may require that most of the remaining words in a sentence were already contained in the dictionary. A large vocabulary could be built automatically from a relatively small one initially listed; there is a question about the size of the program and automatic procedures required to fulfill this function. The conclusion of researchers in mechanical translation is that the function is impractical; this conclusion is open to doubt. The moot issue is the relative cost of developing a complete description of a vocabulary by manual as compared to automatic methods. (U)

2. Syntactic Analysis

Extensive research has been devoted to developing methods of automatic

syntactic analysis, most of it in pursuit of techniques for the machine translation of natural languages. The methods of research, theoretical approaches, and resulting techniques vary widely. One of the primary research tasks in developing a Fact Correlation System will be to determine the method of analysis that can best be adapted to the needs of such a system. (U)

To devise a method of syntactic analysis requires the formulation of a theoretical grammar. Such grammars vary according to the purpose for which they are intended and according to the theoretical approach adopted. Some are intended as models for the production of all grammatically correct sentences in a language; others are designed to discover and describe the syntactic structure of any sentence in a language, which includes the determination that a given sentence is ill- or well-formed. The recognition of sentence structure and the generation of sentences are obviously related, although the two functions of grammar are usually treated separately in the pursuit of techniques. (U)

The accurate recognition of the sentence structure of input statements is the primary requirement for a Fact Correlation System. The generation of sentences is required by the final output stage of the system, but the structure of output sentences may be relatively simple. (U)

An obvious assumption in developing methods for automatic syntactic analysis is that any grammatically correct sentence has a grammatical structure that is algorithmically discoverable. An additional requirement for correlating facts is that the discovered structure be unique; that is,

that it be unambiguous. Whether or not all sentence structures are algorithmically discoverable has not been proved. The problems of syntactic analysis are complicated by the interrelationship of syntactic and semantic factors in natural languages; relatively little has been determined about the nature or the existence of semantic structure. The discovery of invariant and formal semantic structures may simplify the techniques of syntactic analysis, which have had to be extended and refined to solve problems of a semantic nature. No method of syntactic analysis yet devised has been capable of resolving all semantic ambiguities. (U)

Research in syntactic analysis is being conducted at many centers in the United States and abroad. Much of this research has been repetitive. Rather than review the work of each group separately, the major approaches to syntactic problems and the major contributions to each approach will be considered. The three most common methods of syntactic analysis existing today are phrase structure analysis, dependency analysis, and predictive analysis. These methods are reviewed in the following paragraphs. (U)

(a) Phrase Structure Analysis - Phrase structure analysis, also called immediate constituent analysis, is the most widely used of the three existing methods. The fundamental assumption of the method is that complete sentences, the ultimate constituents of properly formed language, are analyzable as sequences composed of one or more immediate constituents or phrases; each of these phrases has a definite structure of one of a limited number of types. (U)

The fundamental work in the phrase structure theory of

language formation was done by Harris and extended in the now classic work of Chomsky. Chomsky's work may be called formal theoretical linguistics, since it treats grammars as formal systems--sets of rules for generating the sentences of a language, English--and is concerned with the theory of grammars per se. (U)

Chomsky distinguishes three types of grammars: finite state grammars, phrase structure grammars, and transformation grammars. Of these, transformation grammars are the strongest and finite state grammars, the weakest; the strength of a grammar is determined by the variety of sentences it can generate. Chomsky demonstrates that finite state grammars, which treat language as generated by finite state Markov processes, are incapable of generating an adequate set of English sentences. (U)

A phrase structure grammar is a set of rules for the generation of complete phrases in a language. A transformation grammar is an extension of a phrase structure grammar, since it also generates a number of phrases. However, rather than generating all sentences by phrase structure rules, such a transformation grammar provides for the generation of kernel sentences; kernel sentences are then subjected to specific transformations, which produce related phrases. Both phrase structure and transformation grammars appear to be capable of generating almost all possible English sentences. However, the number of rules that an adequate phrase structure grammar requires is prohibitive. An equivalent transformation grammar requires fewer rules; thus, it is stronger and more practicable. (U)

The grammars described by Chomsky are sentence generating

grammars, while Harris formulated a recognition grammar limited to the recognition of transformation types. Generating grammars are important for recognition, however, since a generating grammar may serve as the basis for a recognition grammar; the latter are more complex, since decoding is more complicated than encoding. No complete generating or recognition grammars are known to have been formulated for any language. (U)

A model for generating English sentences has been formulated by Yngve. This model depends upon the concept that sentences are naturally produced in left-to-right order; that is, as linear sequences of words. Yngve also hypothesizes that the complexity of sentences in ordinary language is naturally limited by the capacity of a human being's temporary memory. His hypothesis includes a method for determining the complexity of any sentence. Yngve has produced a simple, yet effective, generation model to test his hypothesis. On the basis of the results he concluded that grammatically correct sentences could be generated, even though his initial model could not generate all possible English sentences; however, most of the sentences are meaningless. Yngve adds that the question of meaning is properly a semantic problem, not a grammatical problem. The resolution of this problem required a corresponding semantic model, which will generate meaningful sequences of words, in order to reduce or eliminate semantic ambiguities. (U)

Sidney Lamb's stratificational grammar is based upon phrase structure theory. Lamb develops his grammar from a complex analysis of language in terms of hierarchical formation strata, which depend upon

distinctions between morphological, lexical, and semantic characteristics of language. Lamb's grammar has been used in the formation of automatic dictionaries for mechanical translation. As applied to syntactic analysis Lamb's grammar appears not to be significantly different from other phrase structure theories. It is similarly dependent upon the use of empirically determined semantic distribution classes, which are described in a subsequent paragraph. (U)

The theory is phrase structure recognition grammars is being investigated by Hiz at the University of Pennsylvania, where his methods have been used as the basis for a program of mechanical grammatical analysis. (U)

Phrase structure analysis is the method most widely applied in mechanical translation studies in the United States at present. Mechanical translation is the only practical application of syntactic analysis to date, although research is being conducted on applications to information retrieval systems. Insofar as no method of syntactic analysis in use is complete--that is, that no method is capable of an unambiguous and unique parsing of all sentences in any language--no method is adequate for a Fact Correlation System. (U)

The most extensive application of the theoretical work of Chomsky and Yngve has been made by Pendergraft and his associates at the University of Texas. This development has used phrase structure grammars in formulating programs for the analysis and synthesis of language in mechanical translations from German to English. This program

has developed details of phrase structure theory in both linguistic and mathematical forms. (U)

The method of syntactic analysis used by Pendergraft depends upon information about the semantic functions of words in the sentences analyzed. This information is obtained empirically, and cognitive words are grouped in semantic classes. The inclusion of a word in a semantic class determines or limits the permissible range for using that word with other classes of words. Such classes of words, frequently called distribution classes, must be defined for any method of automatic syntactic analysis in order to resolve semantic ambiguities; these classes are also essential for synthesizing grammars in order to assure the generation of meaningful sentences. Distribution classes should, if possible, be determined from a formal theory of semantic structure. In the absence of such a theory, they must be determined empirically by either statistical studies of textual material, which may be performed by a computer, or an appeal to the knowledge of native speakers of the language. Empirically determined semantic classes will inevitably be similar to those derived from a formal theory of semantic structure; however, these classes will not provide algorithmic methods for resolving semantic ambiguity. This problem is discussed in the section on semantic analysis. (U)

Various forms of phrase structure theory have also been used by investigators at Georgetown University, the University of Illinois, and Wayne State University. (U)

(b) Dependency Theory - Dependency theory is the most common

alternative to phrase structure theory for describing sentence structure. Dependency grammar, in a simple form, is taught to most high school students. The theory of dependency grammars has been developed by Tesniere and Lecerf in Europe and has been further developed in this country by Hays, Harper, Edmundson, and their associates at the Rand Corporation. These techniques have been applied in developing techniques for Russian to English translation. (U)

According to dependency theory, the syntactic structure of a sentence can be analyzed in terms of a partial ordering of the words of the sentence; one word is independent and all the others depend upon it, directly or indirectly. The topological structure resulting from a dependency analysis is a tree with the independent terms at the apex node; the other words occupy dependent nodes. (U)

Dependency theory depends upon semantic information about the words in a sentence. Considerable effort has been expended by the Rand Corporation in determining the distribution classes of semantic types, and computer techniques have been devised to perform these functions. To limit the extent of the semantic problem, the study has been confined to the analysis of Russian scientific texts in physics. (U)

Dependency theory and phrase structure theory are both complex methods of sentence analysis; their degree of complexity is similar. Dependency theory permits a unique analysis of some sentences for which phrase structure theory provides several analyses; similarly, phrase structure theory yields a unique analysis for some sentences that are

not unique under dependency analysis. The relations between the two methods have been analyzed by Hays in terms of the topological characteristics of the sentence structures revealed by each method. (U)

Both methods of analysis are reportedly used by the Soviet Union for mechanical translation. The Russians have stressed the necessity for formal theoretical linguistics to support translation techniques; they have used phrase structure and dependency theory grammars according to the characteristics of the languages translated. (U)

(c) Predictive Syntactic Analysis - Predictive syntactic analysis differs from the previous two methods because it was developed specifically as a method of automatic syntactic analysis. Predictive analysis is based upon a continuous left-to-right scan of a sentence, a scanning process that corresponds to the way people speak, hear, or read sentences. Each word in the sentence is examined in order. For each word the possible syntactic role(s) in the sentence is predicted; the kinds of words and syntactic structures that can immediately follow the word are also predicted. As predictions are fulfilled and possibilities eliminated by successive words, the sentence structure is revealed. This method permits the analysis of sentences that are not autonomous; i.e., sentences dependent upon previous sentences for reference can be analyzed by retaining unfulfilled predictions until they are satisfied or by reiterating the process when necessary. (U)

Predictive analysis was first formulated by Ida Rhodes; it has been further developed theoretically and practically in its application

to the machine translation of Russian to English by Sherry, Oettinger, Bossert, Isenberg, Plath, Guiliano, and their associates at the Harvard Computation Laboratory. Oettinger and Sherry have developed a theoretical model that is similar in some respects to the phrase structure model of Chomsky. Syntactic analysis by the predictive method includes the simultaneous determination of phrase structures. (U)

Predictive analysis, like other methods available, is not yet capable of correctly analyzing all sentences. This incapacity occurs in part because a complete grammar has yet to be specified (serious problems arise in the analysis of indirect objects and coordinate conjunctions, for example). As a result, a complete set of computer programs has not been developed to apply the method. Sherry indicates that new rules to handle syntactic problems are easily incorporated into the program at the cost of great complexity. Some problems of semantic ambiguity, however, are not amenable to syntactic methods and require some form of semantic analysis. (U)

Predictive analysis is an empirical method. It is based upon the assumption that a language produced and probably interpreted as ordered sequences by human users should lend itself to similar analysis by machine. This method uses theoretical material from both phrase structure grammars and dependency theory. Predictive analysis appears to be simpler to apply than the methods based directly upon phrase structure or dependency theories, probably because it has been developed specifically as a machine technique. (U)

(d) The Operational Approach of Silvio Ceccato - The method of syntactic analysis developed by Silvio Ceccato and his associates at the University of Milan is a method specifically devised for mechanical translation. It is not based upon a theoretical grammar in the sense of Chomsky's grammars; nor is it essentially an empirical method like predictive analysis. Rather, Ceccato's method is based upon a philosophical theory of the nature of human thought. Ceccato contends that the structure of language can only be properly analyzed in terms of thought processes or, in his terms, the operations performed by the human mind and/or body in transforming an input statement into a response, which may be manifested by any mode of human activity. Such an analysis permits syntactically and morphologically dissimilar languages to be reduced to a common semantic base, which is itself pre-linguistic or, at least, extra-linguistic. (U)

The resulting method of syntactic analysis is similar to the predictive method. It employs a left-to-right scanning of sentences, again because this scanning procedure is the apparent method used by human speakers, and bases the analysis upon a series of successive predictions. Ceccato speaks of correlational networks and matrices, which are possible configurations predicted during the scanning process. One of the networks or matrices (the ideal) is finally determined to be the accurate syntactic structure. The theoretical basis of Ceccato's system is used to determine the predictions made during the analysis and also to determine the corresponding output in the translation process. The Ceccato system should be carefully studied and compared with the

method of predictive analysis. The two methods may prove to be complementary. (U)

Although it is called a semantic theory, Geccato's concept of language structures does not include a theory of formal semantic structure. It is thus dependent upon empirically determined semantic classes similar to those used by other methods of analysis and translation. (U)

(e) The Lattice Theory of Syntax - An interesting model of syntactic structure in terms of lattice theory has been developed by Parker-Rhodes and other researchers at the Cambridge (England) Language Research Unit. The model assigns positions at nodes of a lattice structure to syntactic substituents (the simplest syntactic units, usually words). Each node corresponds to a substituent type. The syntactic functions of substituent types are determined by positions in the lattice. (U)

A method of automatic syntactic analysis based upon the lattice theory has been formulated. The method appears to be quite effective, although it has not been extensively tested. The lattice model permits traditional subject-predicate distinctions, dependency theory analysis, and analysis in terms of phrase structures. The system has the advantage of a relatively simple mathematical structure. (U)

It remains to be determined whether lattice theory permits a simpler method of syntactic analysis than the method of predictive analysis. The predictive method has been much more extensively tested. (U)

(f) Commentary - Of the methods of syntactic analysis reviewed, the predictive method appears to be the simplest. The difference between theories of syntactic structure and methods of syntactic analysis must be distinguished. Obviously, the simplest method of analysis that is completely adequate is the most desirable method. The application of a powerful theory of semantic structure will probably simplify the problems of syntactic analysis. The recommended method of syntactic analysis will be the method that proves to be the most efficient in conjunction with the sense-value theory of semantic structure, which is described in the following section. (U)

3. Semantic Analysis

The processing of natural language data by computers depends upon the solution of fundamental semantic problems. In particular, it is necessary for automatic language analysis that computers be programmed to determine whether or not grammatically correct sentences are meaningful, to resolve syntactic ambiguities that depend upon semantic information, and to resolve semantic ambiguities automatically. Similarly, automatic synthesis of statements in natural language implies the production of meaningful sentences. (U)

The problem of programming computers to analyze and synthesize meaningful sentences is difficult. The problem has not been satisfactorily solved, perhaps because the criteria for determining that a sentence is meaningful have not been precisely defined. Little is known about the semantic structure of natural language, or even if natural language has invariant relations among terms that could be called a semantic structure. (U)

It is important to distinguish between two basic problems of meaning in language. One is the problem of the meaning of a word or sequence of words; this problem is a question of determining the denotation and connotation, extension and intension, of a word or phrase. It is a complex problem with many aspects--logical, linguistic, psychological. The other problem is that of determining the criteria for judging that a given sentence is meaningful or nonsensical. This question is obviously related to the previous question of meaning; but it may also be considered as essentially a formal problem, that of determining the criteria for judging that a given sequence of words is semantically acceptable. For purposes of computer processing of natural language, meaning must be considered a structural problem. For if a computer is to understand a language, such understanding must be purely formal. One cannot point to a red object, for instance, and thus demonstrate the meaning of the word "red;" a computer has no experience of redness. Rather, the understanding of a computer must be in terms of the significant use of a word in the language. That is, the meaning of a word for a computer is information about how to use the word significantly with other words. (U)

The meanings of words in a dictionary are, in fact, just such formal descriptions, given by listing synonyms and by citing examples of the correct (significant) use of the defined words. Such definitions are linguistic; they appeal to the user's knowledge of the grammar of the language and of the relation of at least some of the words to his experience. But, if a user who knew nothing about a language except the rules of its grammar were to attempt to learn the language from a dictionary,

he would learn only that certain words are considered meaningful--namely, those sequences given as examples of the correct use of the words defined. This knowledge of a language constitutes the limits within which a computer is restricted. (U)

If there are general rules that govern the semantic relations of terms in a language, the task of learning a language is facilitated. The learner, whether a human being or a computer, may then infer from knowledge of the use of a word in some meaningful sequences to its correct meaningful use in others. If there were no such general rules, learning a language would require learning the explicit use of every word with every other. Grammatical rules are general use rules; they restrict the uses or structural relations of words in certain ways. But they do not determine meaningful use in general. (U)

The most widely adopted approach to solving semantic problems is to classify the words of a language into semantic distribution classes; membership in a class then determines the permissible use of the words. Such classes may be determined empirically by examining an extensive corpus in a given language or intuitively by describing such classes. Both methods are used in practice. The empirical determination of distribution classes may be performed by a computer. This method has been adopted by the Rand Corporation and the University of Michigan in their joint mechanical translation research. The intuitive method, an appeal to the knowledge of native speakers, is used by Householder and Lyons at the University of Indiana. The classes determined by Ceccato's

group are basically determined by intuitive methods. Neither method of determining semantic classes is based upon a formal theory of semantic structure; neither establishes general criteria of meaningfulness for an entire language; and neither can provide a general method for resolving the persistent and fundamental problem of multiple meaning--the problem of ambiguity. (U)

The semantic accuracy of models for the production of sentences based upon the use of such distribution classes must be determined statistically. That is, the degree of semantic effectiveness of such models must be judged by the relative frequency of meaningful sentences produced. The difficulty of attaining high relative frequencies of meaningful sentences increases as the complexity of the vocabulary used is increased. (U)

The alternative approach to semantic problems is to formulate a general semantic theory from which the meaningful use of words can be derived. There have been relatively few such formulations; the only theory that approaches generality is the theory of sense values developed by IEC. (U)

K. Sparck Jones of the Cambridge Language Research Group has proposed a method of determining word use based upon the co-use of words in given contexts. She offers formal definitions of word-use and criteria for determining similarity of word-use. However, the criteria for determining whether or not the resulting semantic restrictions are satisfactory are intuitive; and her theory provides no method for

automatically resolving ambiguity. (U)

Peter Guberina has proposed a single formula governing the semantic relation of words in any meaningful sentence. Margaret Masterman has investigated the Guberina hypothesis, and R. M. Needham has reformatized the theory in traditional subject-predicate notation. It can be shown that Guberina has discovered some of the semantic properties described by the sense-value theory. The Guberina theory has not been fully developed; as it stands, it is not sufficiently powerful to provide a semantic structure for a language nor to resolve all cases of ambiguity. (U)

The Electro-Optical Systems Company has attempted to discover semantic properties of language by ranking words and phrases according to the relevance of meaning to a specified subject. Such relevance is determined by asking experts in the appropriate field to rank a set of words and word pairs on a scale according to the relevance of the concept represented by the words to the subject matter of the field. The resulting scale values are compared and analyzed by statistical methods, and words or concepts are assigned positions in a semantic classification space. The purpose of this attempt was to determine the semantic relations among such words to aid in the development of a Fact Correlation System. However, this method of approaching semantic problems deals primarily with words and secondarily with concepts rather than with terms of linguistic structure. The method provides no basis for a formal theory of semantic structure. The procedure fails to make the fundamental distinction between the meaning of words and the meaningfulness

of sentences. Unless this distinction is clarified, the problem of meaningfulness cannot be treated as a structural problem, and no formal solution can be found for the semantic problems involved in processing language automatically. (U)

In contrast to these approaches to semantic problems, research has been directed to the discovery of the fundamental semantic structure of natural language. The new theory of semantic structures provides a theoretical basis for the analysis and synthesis of meaningful sentences and for the automatic resolution of ambiguity. The theory thus offers solutions to some of the problems involved in the computer processing of natural languages. (U)

The theory of sense values demonstrates that ordinary English appears to obey fundamental rules that determine the use of words in sentences such that sentences are invariably meaningful if the rules are observed, nonsensical if they are violated. Furthermore, these rules precisely specify the presence of ambiguity in the meaning of a word and automatically resolve this ambiguity in such a way that the correct use of each meaning of a word is stipulated. (U)

The rules are simple, few in number, explicitly semantic, and general; they apply to an entire language, governing the meaningful use of words. The rules thus provide a relatively simple basis for analyzing sentences automatically. The semantic structure may be interpreted by a topological model in which words of the language are mapped upon the nodes of a topological tree (an acyclic graph). As the theory develops, it

will probably be necessary to use a topological structure that is more complex than a tree. It is convenient to refer to this theory as the sense-value theory of natural languages. (U)

The theory of sense values treats the problem of determining the meaningfulness of sentences as a problem concerning the permissible use of the words of a language to form such sentences; it is a theory of the use-relations of words. However, unlike grammatical theories of usage, which essentially stipulate the use of each word of a language, the sense-value theory provides for the determination of the meaningful use of a given set of words in all grammatically correct sentences that can be formed from the set, if the correct use of the words is known in a relatively small proportion of such sentences. In other words, given some information about the meaningful use of a group of words, the theory permits the correct inference of much more information about such use. Like any general theory, the sense-value theory permits prediction and generalization from given data. Since its rules are explicit and general, the rules can be applied invariantly by a properly programmed computer. (U)

The semantic rules also prohibit the formation of nonsensical sentences; that is, they not only provide for the correct use of words, but also prevent their misuse. And questions of ambiguity are automatically resolved by the theory. The rules invariably detect the presence of ambiguity in a sentence; the words responsible for the ambiguity are specified, and the different meanings of the ambiguous words--that is, their different correct uses--are also determined. (U)

The semantic structure of a language is based upon the knowledge of the universe that its users have experienced. Thus, the sentence, "The building is red" is meaningful; whereas, the sentence "The equation is hungry" is nonsense. This discrimination of sense and nonsense is valid because buildings are in fact the sort of thing that can be red, but equations are not the sort of thing that can be hungry. Language is thus a model of experience. (U)

One of the fundamental distinctions that must be made by a semantic theory is that language is a model of possibilities rather than of explicit facts. Since it makes sense to say "The building is red," it also makes sense to say "The building is not red;" similarly, since it does not make sense to say "Equations are hungry," it also is meaningless to say "Equations are not hungry." Some analysts of language have accepted the latter sentence as meaningful. But what, in fact, that sentence must mean, if it is accepted as meaningful, is that equations are not the sort of things that can be hungry. This interpretation, however, is not equivalent to "George is not hungry." For, it is meaningful to say that George is hungry. This distinction is obviously basic to the semantic analysis that provided only for the processing of factually true sentences would obviously be of little value: factual truth is continually changing. (U)

The semantic structure of a language is mapped on a language structure or tree that evolves from valid sentences in that language. If a group of simple sentences has the form S is P--for instance, "The building is red" and "The equation is hungry"--the cognitive words in each

The semantic structure of a language is based upon the knowledge of the universe that its users have experienced. Thus, the sentence "The building is red" is meaningful; whereas, the sentence "The equation is hungry" is nonsense. This discrimination of sense and nonsense is valid because buildings are in fact the sort of thing that can be red, but equations are not the sort of thing that can be hungry. Language is thus a model of experience. (U)

One of the fundamental distinctions that must be made by a semantic theory is that language is a model of possibilities rather than of explicit facts. Since it makes sense to say "The building is red," it also makes sense to say "The building is not red;" similarly, since it does not make sense to say "Equations are hungry," it also is meaningless to say "Equations are not hungry." Some analysts of language have accepted the latter sentence as meaningful. But what, in fact, that sentence must mean, if it is accepted as meaningful, is that equations are not the sort of things that can be hungry. This interpretation, however, is not equivalent to "George is not hungry." For, it is meaningful to say that George is hungry. This distinction is obviously basic to the semantic analysis of language. It is a distinction that has been generally neglected, but it is one upon which the sense-value theory is based. The semantic structure derived from the theory permits the analysis of meaningful sentences that may be factually true. Any model for semantic analysis that provided only for the processing of factually true sentences would obviously be of little value: factual truth is continually changing. (U)

The semantic structure of a language is mapped on a language

structure or tree that evolves from valid sentences in that language. If a group of simple sentences has the form S is P--for instance, "The building is red" and "The equation is hungry"--the cognitive words in each sentence form pairs that either are meaningful in such a sentence or are not.⁽¹⁾ Thus the pair "building, red" makes sense; the pair "equation, hungry" does not. All such pairs are taken from the set of sentences being analyzed. By analyzing the relations among these pairs, according to the rules of the theory, the sense relations of all the words in question may be determined; and the resulting semantic structure may be interpreted by a language tree, as shown in Figure 2-1. The various aspects of the semantic structure are interpreted in the following paragraphs. (U)

(a) Meaningful Relations - Any two or more words may be used meaningfully in the same simple sentence if and only if the nodes (branch-points) they occupy lie on a common ascending path (or, a common descending path) to the top of the tree. Permissible paths are thus determined by going along the branches of the tree in one direction only, either up or down from node to node. (U)

Thus, reading from the bottom up in Figure 2-1, the words on the first permissible path on the left of the language tree are "sunset,"

(1) This example is representative of a class of monadic predicates. Subsequent research has attempted to extend the theory to handle dyadic predicates. Eventually, tryadic predicates may be sufficient to resolve all relational problems occurring in English. These more complex predicates, or relations, require a structure that is more complex than the simple tree shown in Figure 2-1. It is pertinent to note, however, that the tree as represented is not a binary tree.

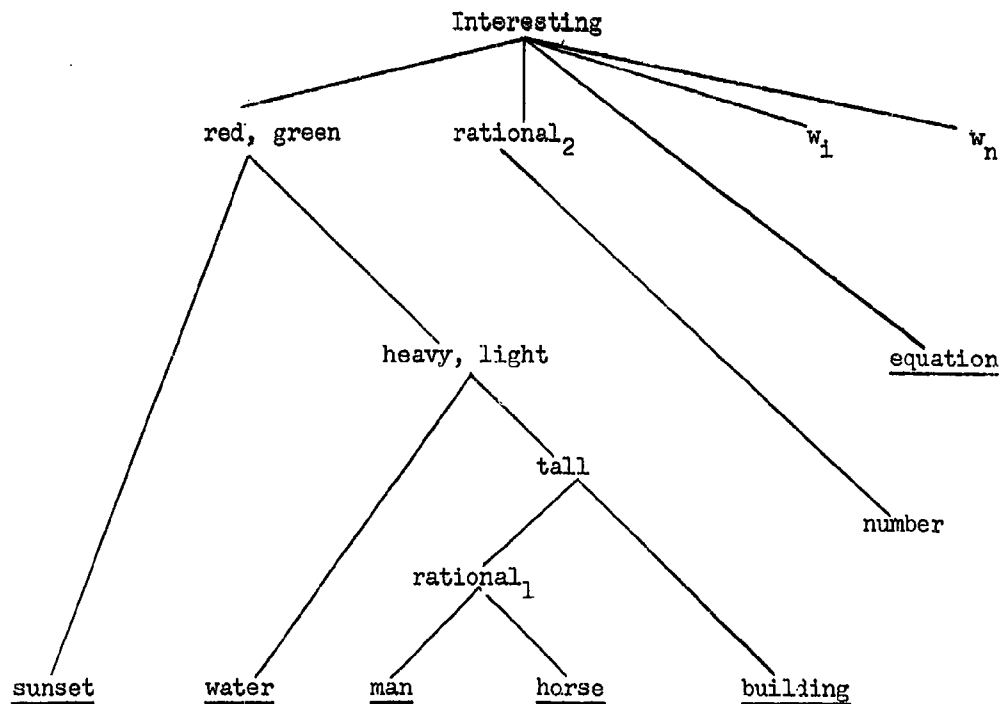


FIGURE 2-1. The Language Tree of Semantic Structure

"green," and "interesting." This set of words is called a use-set, a set of words that can be used meaningfully together. The possible simple sentences formable from these words are all meaningful; for instance, "The sunset is red" or "Red is interesting." (U)

On the same tree, "water" and "tall" do not belong to a common use-set; they do not lie on a common path reading up or down. The sentence "Water is tall" is not meaningful. Similarly, it follows that "The equation is interesting" is meaningful, while "The equation is red" is not. In general, the tree permits all possible meaningful combinations of the words used and permits no meaningless combinations. (U)

(b) Ambiguity - The word "rational" appears twice on the tree but with different subscripts. The different locations indicate that the word has different uses; thus, it has different meanings, since meaning in this treatment of language is a function of use. The locations indicate that "rational" is an ambiguous term. This discrimination is intuitively obvious; "rational" as used in "The man is rational" does not have the same meaning as in "The number is rational." However, the ambiguity indicated by different locations is determined from the uses of the word by the rules of the theory, not from an appeal to intuition. (U)

Not only is ambiguity determined by rules, but the different uses of the word are also explicitly stipulated by different locations; ambiguity is determined, and then resolved. Since the ambiguity is resolved by rules, it can be automatically resolved by a computer. If a word has several meanings, the rules assign it to several locations, one for each meaning; in effect then, each meaning of a word is treated as a different word, all of which have a common spelling. (The similarities in meaning of the two or more meanings of a single term can also be determined from their relative positions, if a sufficient number of terms are mapped on the language tree; the theory permits the interpretation of ambiguity as well as its specification.) (U)

(c) Sense Equivalence - On the illustrated language tree two of the nodes are occupied by two words: "red" and "green" appear together, as do "heavy" and "light." This combination indicates that

the two words have similar uses, thus similar or closely related meanings. In fact, if two words are synonymous or antonymous, they must occupy the same locations. Terms occupying the same node are defined as being sense equivalent. (U)

(d) Absolute Predication - The words in Figure 2-1 are all written in the affirmative form. That is, "interesting" is at the top node, and "uninteresting" does not appear. However, all the terms may be interpreted as absolute predicates; that is, the word "interesting" may be used to represent both "interesting" and "uninteresting," and "red" may signify both "red" and "not red." If it is meaningful to say that something is red, it is also meaningful to say that it is not red; and "not red" will occupy exactly the same tree location as "red." In terms of formal semantics, a word and its negation have the same uses and thus the same locations. The semantic language trees are not trees indicating class-inclusion, so-called Aristotelian trees; rather, they indicate semantic inclusion. An Aristotelian tree, which is based upon class relations, might indicate that all men are mortal. A semantic tree would indicate only that it is meaningful to say that all men are mortal, but that it is also meaningful to say that all men are not mortal, whether or not it is true. (U)

(e) Information Spectrum - Tree locations indicate the information weight of words, which is a function of their specificity. Thus the word "interesting," which appears at the top (root) node of the illustrated language tree, carries little information; it tells

little about anything to which it is applied. More information is given by the word "red," which is lower on the tree; maximum information is given by words such as "man" at the base of the structure. (U)

The information relations of words are also given by their locations. Factual trees, Aristotelian trees, are in fact being analyzed by correlation techniques to determine the information content of words and their related concepts, as in the work of Watanabe. The semantic structure is, however, more general than a factual structure and may be considered a tree of factual possibilities from which trees of factual actualities may be derived. (U)

(f) Logical Relations - The position of a word determines the possibility of its logical relation to other words. Unless two words can be used meaningfully together, they cannot be logically related; the semantic level of rectitude is logically prior to the level of logical consistency. The semantic structure of a language provides a basis for logical analysis. For instance, two words cannot have explicitly contradictory uses, such as "bachelor" and "husband," unless they are closely related semantically. (U)

(g) Memory Structure - Tree locations can be specified by numerical indices that also specify the relations of the words indexed. The tree structure of a language thus provides a basis for an indexing method that permits semantic structure to be explicitly indicated by the indices. That is, the index used to identify a word in memory devices and in language processing, can also indicate its semantic relation to the

entire language; this indexing technique may be an efficient and powerful device for the correlation of facts. (U)

The language structure as described is a simple one; the theory permits the construction of trees with as many words as desired. It is theoretically possible to construct semantic trees for an entire language. (U)

Although the illustrated sense relations were relatively simple, they are fundamental. More complex relations of words are also derivable from the same theory. For instance, binary relations between terms, as in the statement "The tree is taller than the house," and the tertiary relations, as in "New York is between Boston and Washington," are analyzable in terms of this theory. Continued research is necessary, however, to develop these concepts and their related structural forms more extensively. The sense-value theory is a new insight into the problem of semantics, but it requires further investigation. (U)

4. Automatic Dictionaries

An automatic language processing system requires an automatic dictionary. The automatic dictionary required by a Fact Correlation System would differ from a dictionary for mechanical translation only in that equivalent words in the target language would not be required. (U)

There are two basic factors in the design of an automatic dictionary: the functions to be performed by the dictionary, and the efficiency of computer programs in forming, storing, and using the dictionary. (U)

(a) Dictionary Functions - A dictionary for the language

transformation process performs two basic functions: it lists the words of the language to which the words in input statements can be compared in the word-recognition stage of the process; it provides information about the words as required for syntactic and semantic analysis. Since automatic dictionaries are not lists of synonyms or descriptions of etymology, they are really glossaries that describe the formal properties of words. (U)

The syntactic and semantic properties of listed words depend upon the methods of analysis applied by the system. The simpler the methods used, the less information required. One of the advantages of a formal theory of semantic structure integrated with a method of syntactic analysis is that the semantic information in a dictionary can be reduced to a numerical code describing positions in a topological model of semantic structures. (U)

A general theory of semantic structure may also provide a method for automatically determining the semantic functions of words added to the dictionary by analyzing their context. Similarly, it may be possible to determine the syntactic functions of new words automatically. The computer would then be capable of generating its own dictionary, using an initially stored basic vocabulary as the foundation for analyzing and deriving new entries. It remains to be determined whether such a method is feasible and, if so, what the basic vocabulary must be. (U)

(b) Dictionary Structure - the structure or form of the lists comprising an automatic dictionary is determined by the methods of

linguistic analysis. The capacity of the computer also presents restraints. (U)

A complete listing of all forms of all words, including plural forms, verb tenses, and other inflected forms is obviously impracticable. The methods of listing developed for mechanical translation solve the storage problem by splitting words into stems and affixes. The possible prefixes and suffixes for various types of words may be listed for each stem or for classes of word types. Such methods have been developed by Lamb and Jacobsen, the Rand Corporation, Oettinger, and others. Lamb and Jacobsen describe such a method, based upon Lamb's theory of language structure, that enables a computer with 32,000 words of core storage to list a vocabulary of hundreds of thousands of words with a look-up speed of more than 100 words per second. (U)

A basic problem in using automatic dictionaries is the efficiency with which word lists can be processed. This problem is related to the requirements of the methods of syntactic and semantic analysis that are used. A method of storage that provides fast processing is to list the vocabulary words and their variant forms separately. The second list contains the glossary descriptions. Word forms in the first list are accompanied by addresses locating appropriate descriptions in the glossary. Such a method permits the word list and the glossary descriptions to be organized independently. (U)

It is unlikely, for instance, that syntactic and semantic descriptions for an adequate vocabulary can be listed in core storage.

The Rand Corporation has devised a dictionary in which word forms or headings are stored in rapid access memory; the expositions of word function are stored on tape. The design of efficient programs will circumvent any disparity between the size of the dictionary and available core storage. (U)

The method of dictionary formation and organization adopted for a Fact Correlation System must be conditioned by the methods of linguistic analysis and the type of computer. The development of automatic dictionaries is sufficiently advanced so that it is not necessary to perform extensive research. An adaptation of one of the existing methods will probably suffice. (U)

D. Research Approach

In reviewing the problems of linguistic transformation, various levels of language and several areas of technique were enumerated. The following discussion considers an approach to research and development in fact correlation techniques in terms of the scope of each area of technique in relation to the various levels of language. (U)

1. Recognition of Well- or Ill-formed Constructions

The primary reason for requiring a language transformation process in a Fact Correlation System is the informal nature of ordinary discourse, which is not amenable to automatic correlation or processing techniques. The human users of language do not ordinarily rely on the conscious or explicit application of formation rules in generating or understanding statements. These rules are followed, but unconsciously; and the

manifestation of these rules is that they can be formulated formally.

To the extent that the system is to function automatically, reliance on judgment or intuitive understanding must, at least initially, be abandoned. The problem for the language transformation process is to discover and explicate the formation rules at each level of language in order to encode the information content of ordinary discourse in formal terms. (U)

(a) Morphology - At the morphological level, the simplest solution to the problem of formation rules is an extensional listing of the acceptable word forms in the input language.⁽²⁾ Such a list would not be closed, since it is desirable to be able to add terms to the system's vocabulary at will. Checking against such a list, however, serves the valuable purpose of eliminating troublesome problems that arise from misspelling--if every word that appeared in the input were accepted as being on the morphologically well-formed list. If a new term is introduced, a special set of instructions can insert it on the list; or a machine-generated query, occurring whenever an input term is not on the list, can resolve the question of genuineness or spuriousness.⁽³⁾

(2) Intensional rules for morphological formation are possible. Since such schemes are imperfect (all acceptable words may be generated, but many words not in the language may also be generated), and since word lists are useful for other purposes (syntactic and semantic coding or dictionaries), models of word generation are not pertinent except as possible paradigms for higher order processes.

(3) This technique--to ask for clarification--is a general system strategy whenever problems of ill-formation (or ambiguity) occur at any level of language. However, such questions generally have to be generated serially, since it is inappropriate to apply the formation criteria of higher levels if lower levels are ill-formed.

Thus extensive research on morphological formation is not anticipated. (U)

(b) Syntax - Rules for syntactic formation are not so easily developed. A complete extensional listing of sentences is theoretically impossible; its approximation is practically unfeasible. Fortunately, a great deal of theoretical and practical work has been done on this problem, especially as stimulated by the attempt to develop automatic translation between natural languages. (U)

While specific techniques differ in detail, the essential approach is to store syntactic information (e.g., noun, verb, etc.) about each word on the morphological list and to specify rules for the possible combination of various syntactic forms. Because of various difficulties, some of which may be resolvable semantically, there is as yet no perfect program for the automatic syntactic analysis of ordinary discourse. But progress has been achieved. In order to avoid duplication of effort, the best techniques available should be used for a Fact Correlation System rather than expending additional new research on the isolated problem of syntactic analysis per se.

(c) Semantics - Research on the problem of semantic formation is by contrast quite sparse. The concept of semantic well- or ill-formed constructions is best illustrated by Chomsky's sentence, "Green ideas sleep furiously." This sentence is syntactically well formed but semantically ill formed or meaningless. An approach to this problem is the theory of sense values, which can establish whether two terms may be sensibly used together by means of their location in the semantic

structure (see Section C.3). (U)

To date this approach is limited to the semantic evaluation of subject-predicate co-occurrence. A more general extension of the sense-value theory that will mesh with an adequate syntactic analysis will have to be developed in the research program. It will then be possible to encode positions in a semantic structure as well as syntactic form(s) for each term on the morphological list. A powerful lexicon will thus be available by means of which nonsensical, though syntactic, sentences can be detected. (U)

(d) Logic - The criterion for logical ill-formed constructions (absurdity) is simply stated: any statement that implies a contradiction is logically ill-formed. If a contradiction can be derived from a corpus, whether automatically or manually, then it is logically ill-formed. Unfortunately, an inability to derive a contradiction does not generally establish the logical well-formed construction (consistency) of a statement. The problem of consistency is a serious one for the Fact Correlation System as a whole. It is intuitively obvious that consistency tests must be limited in scope for the language transformation process, perhaps to a single sentence or a single paragraph of text. It is in the fact correlation portion of the system that consistency checks of a wider scope will be performed. (U)

(e) Fact - Ill- or well-formed constructions at the level of fact can be understood in terms of the truth or falsity of a statement. In general, the Fact Correlation System will be no more capable of

resolving factual issues empirically than a typical intelligence analyst-- both are restricted to working from information reported in documents; neither observes an event directly. The fact correlation portion of the system may, however, resolve matters of fact, which have not been directly inserted in the textual input, by means of principles of inductive inference and generalization. In a sense, such concern with facts is no more than an application of logic to a more extensive set of assertions than are explicitly stated in the input data. The language transformation process may also use certain limited principles about factual possibility (e.g., an object cannot be in two places at the same time), if not plausibility, which can be applied to evaluating the factual well- or ill-formed construction of a sentence. One of the research tasks is to develop an efficient scheme for such automatic verification of facts. (U)

2. Detection and Resolution of Ambiguity

The discussion of formation rules or criteria at various levels of language illustrates the first step that is necessary to transmute informal discourse into a formal representation. The discussion was limited to the available formation criteria that could filter the actual ill-formed assertions out of a corpus or clarify those statements that were apparently ill-formed. (U)

The transtion from informal to formal discourse cannot be achieved merely by applying filters at any given level of language. Well-formed terms or assertions may be ambiguous--that is, they may have been generated by alternate formation rules. Therefore, the terms may specify

a variety of relations and are thus subject to a variety of interpretations. One explanation is that ambiguity may be resolved by examining the context. Actually the process can be somewhat better articulated. As formation criteria are applied at successively higher levels of language, fewer of the possible interpretations at a lower level remain tenable. Ambiguity is thus gradually reduced and at some point may be completely eliminated.⁽⁴⁾ (U)

(a) Morphology - English, like most natural languages, is replete with homomorphs, words of multiple meaning and usage or, conversely, different linguistic types with the same symbolic form. An example of this problem is the multiple meaning of bore. A serious aspect of this problem is that many ambiguous references are not even listed in dictionaries and thus are not apparent from prearranged multiple morphologic lists. This restriction does not, however, detract from the value of such a list for reducing ambiguity. Sufficient work has been performed in this field so that an extensive research effort is not required. (U)

(b) Syntax - Some of the residual morphological ambiguity can be resolved if syntactic formation requirements place a given word in a given syntactic class. Thus, based purely on its location in a sentence, bore may have to be a noun rather than a verb; then a

(4) Complete automatization of such a procedure entails the risk that a genuinely ambiguous text will be misinterpreted because the computer "understands" the text "better" than the writer. It may, therefore, be desirable to retain considerable computer questioning about ambiguity that is, theoretically, resolvable.

substantial part of its ambiguity is removed. However, ambiguity is not completely resolved. Furthermore, purely syntactic programs are as yet, and probably inherently, incapable of resolving all syntactic ambiguity. Often the syntactic coding of the morphological list along with the criteria for syntactic analysis result in more than one acceptable parsing for a sentence. Since native users of language are inattentive to grammar, as opposed to meaning, there may be considerable research ingenuity necessary to arrange for a machine's questioning of a user in order to resolve syntactic ambiguity. (U)

(c) Semantics - Another part of the ambiguity that remains after syntactic analysis can be resolved at the semantic level. Consider a sentence in which the syntactic analysis establishes the term bore or an inflected variant as a verb. At least two meanings of bore are still possible. Suppose the object in context with bore is a gun. The sense-value criterion of well-formed construction would eliminate any interpretation of bore that are semantically only applicable to people. The semantic structure precludes the use of bore (in the sense of ennui) with guns and the use of bore (pertaining to rifling) with people. A second important contribution of the theory of sense values is its ability to detect equivocation in word use by mechanical procedures even though such equivocation is not listed in the dictionary. (U)

(d) Logic - The semantic criteria of the sense-value theory are not yet elaborate enough to span sentences. Furthermore, these criteria are not designed to cope with such problems as ambiguous pronoun references. To the extent that logical criteria span sentences,

these criteria can use further removed context to resolve ambiguities remaining after semantic screening. Even within sentences, logical criteria may be valuable in eliminating inappropriate interpretations of pronoun reference. (U)

(e) Fact - Not all ambiguities of pronoun references can be resolved without recourse to factual propositions in addition to textual material. For example: "He went to Chicago and at the same time he went to New York." An interpretation referring the pronoun to the same individual can only be rejected by referencing factual possibility. The level of fact may be especially important when the ambiguity is logical. Ordinary language, and even legal discourse, as demonstrated by Layman Allen,⁽⁵⁾ is quite imprecise in the use of logical connectives. It is often not clear on purely syntactic grounds whether or is to be understood in its inclusive or exclusive sense; whether if means only if; and even whether and really means or. If people disagree about the logical interpretation of a sentence, the argument is generally referred to requirements at the factual level. It may be possible to use a limited number of factual prepositions to resolve such logical ambiguities. (U)

3. Generation of Maximally Useful Output

In this area it is only possible to raise questions. Assuming that the input sentences have been analyzed--that the sentences have been parsed according to formation rules and all ambiguities resolved--

⁽⁵⁾ Allen, Layman, Toward a Procedure for Detecting and Controlling Ambiguity in Legal Discourse.

what form(s) should the output of the language transformation process take in order to be useful in further fact correlation? (U)

Should the morphological forms of the input be retained? If so, should these forms also be transformed into a more appropriate morphology in which a sentence of ordinary discourse is represented without one-one correspondence to the words of the input? If so, what input units should the output correspond with? (U)

Should the syntactic forms of the input be retained? The questions abound. (U)

To what extent can the semantic reference, as differentiated from the syntactic reference, of input words be communicated to the rest of the system? That is, is the fact that pencil refers to an object with a large number of characteristics communicated each time the word pencil occurs in the input data? Is such information stored in a semantic dictionary more elaborate than the semantic structure? If so, what are the limitations of scope on such a dictionary? For example, would the characteristic habitable by termites be listed for pencil? (U)

At the logical and factual levels, the questions to be asked will depend upon the scope of the formation rules used. If the logical survey of the language transformation process is larger than a sentence in scope, there is the question of the extent to which logical redundancy should be retained in the output. If the factual principles of possibility are extensive, it may be desirable to add appropriate ones to

the output even though they are not implicit in the input. (U)

4. Evaluation of the Adequacy of Linguistic Analysis and Synthesis

For each level of language, the analytic techniques are likely to remain imperfect for some time. In order to be able to improve available techniques, either by automatic adaptive adjustments or by reprogramming, it is necessary to develop methods for measuring success and failure. The problem is not simply one of program optimization in terms of speed and storage, but one pertaining to the degree of success or failure in achieving desirable results. An evaluation plan for measuring the ability of a Fact Correlation System to fulfill its requirements is an intrinsic part of any contemplated research activity. (U)

E. Summary

The problem of language transformation for a Fact Correlation System can be resolved theoretically and implemented operationally within five years. The only condition is that a reasonable level of research must be maintained. (S)

At the morphological level the problems are well understood and already essentially solved; English is not a heavily inflected language. At the syntactic level the theory of predictive analysis is well developed and reasonably effective. Other methods of analysis are also viable. The selection and extension of one of these approaches would be based upon its adaptability to a particular form of semantic analysis. (U)

At the semantic, logical, and factual levels considerable research remains. The theory of sense values appears to provide a substantial

theoretical framework for semantic analysis. Research on the logical and factual levels seem to be less critical for basic system functions, at least within the context of language transformation. These functions may be combined into a single research task oriented to the question of consistency in linguistic statements. The primary problems of eliciting explicit and implicit information from linguistic data and correlating the factual content of statements is properly the function of the learning processes. (U)

F. References

- (1) Bossert, W., The Implementation of Predictive Analysis, NSF-4, Sec. VIII, 1960.
- (2) Ceccato, Silvia, Linguistic Analysis and Programming for Mechanical Translation (AD 266384); University of Milan, Italy, June 1960.
- (3) Chomsky, N., "Three Models for the Description of Language," IRE Transactions on Information Theory, Vol. IT-2, Proceedings of the symposium on information theory, September 1956.
- (4) ----- Syntactic Structures, Mouton and Company, The Hague, 1957.
- (5) ----- "On Certain Formal Properties of Grammars" (AD 221311) Information and Control, Vol. 2, No. 2; June 1959.
- (6) Darmstadt, Quentin A., A Formal Development and Application of the Theory of Sense Values and Sense-Value Trees for Natural Languages
 - (a) "Part I: Foundations of the Theory of Sense-Values;" International Electric Corporation, Paramus, New Jersey, January 1962: 57 pp.
 - (b) "Part II: The Theory of Sense-Value Trees;" IEC, Paramus, New Jersey, February 1962: 128 pp.
 - (c) "Part III: The Application of Sense-Value Theory to the Analysis, Construction, and Verification of Sentences in Natural Language;" IEC, Paramus, New Jersey, March 1962: 43 pp.

- (7) Edmundson, H. P., and Hays, D. G., Studies in Machine Translation--2: Research Methodology, The Rand Corporation, December 1957.
- (8) Edmundson, H. P., Harper, K. E., and Hays, D. G., Studies in Machine Translation--1: Survey and Critique, The Rand Corporation, February 1958.
- (9) Electro-Optical Systems, Description of "Fact Correlation" Research, RADC Technical Report, 1962.
- (10) Garvin, P. L., Some Linguistic Aspects of Information Retrieval; Ramo-Wooldridge Laboratories, February 1961.
- (11) ----- Automatic Linguistic Analysis - A Heuristic Problem, (Paper 1); International Conference, Teddington, Middlesex, England: 16 pp.
- (12) Guiliano, V. E., Linguistic Information Processing, Report No. 1, Arthur D. Little, Inc., 1960.
- (13) E. Harper, K. E., Procedures for the Determination of Distributional Classes, (Paper 5); International Conference, Teddington, Middlesex, England.
- (14) Harris, Z. S., Methods in Structural Linguistics; Chicago, 1951.
- (15) ----- "Co-occurrence and Transformation in Linguistic Structure," Language, Vol. 33, No. 3; 1957.
- (16) Hays, David G., Grouping and Dependency Theories (AD 250237); The Rand Corporation, Santa Monica, California, 8 September 1960.
- (17) ----- On the Value of Dependency Connection, (Paper 3); International Conference, Teddington, Middlesex, England.
- (18) ----- Research Procedures in Machine Translation (AD 268643); The Rand Corporation, Santa Monica, California, December 1961.
- (19) Hiz, H., Steps Toward Grammatical Recognition; University of Pennsylvania.
- (20) Householder, F. W., Jr., and Lyons, J., Third Quarterly Progress Report on the Automation of General Semantics (AD 250499); Indiana University, November 1960.
- (21) ----- Fourth Quarterly Progress Report on the Automation of General Semantics (AD 253274); Indiana University, February 1961.

- (22) Josselson, H. H., and Jacobsen, A. W., Second Annual Report on Research in Machine Translation; Wayne State University: 1960
- (23) Lamb, S., Outline of Stratificational Grammar, University of California, Berkeley, 1962.
- (24) ----- "On the Mechanization of Syntactic Analysis," Paper 21, Proceedings of the First International Conference on Machine Translation; National Physical Laboratory, Teddington, England, 1961.
- (25) ----- , and Jacobsen, W. H., Jr., "A High-speed Large Capacity Dictionary System," Machine Translation, Vol. 6; November 1961.
- (26) Masterman, M. - Commentary on the Guberina Hypothesis, Cambridge Language Research Unit, 2 August 1961.
- (27) Oettinger, A. G., Foust, W., et al, "Linguistic and Machine Methods for Compiling and Updating the Harvard Automatic Dictionary," Preprints of Papers for the International Conference on Scientific Information; 1958.
- (28) ----- Automatic Language Translation: Lexical and Technical Problems; Harvard University, 1960.
- (29) Parker-Rhodes, A. F., A New Model of Syntactic Description, (Paper 34); International Conference, Teddington, Middlesex, England, 1961.
- (30) Pendergraft, E. D., Jonas, R. W., and Tosh, L. W., Machine Translation Study, Research on the Machine Translation of German (AD 251071), Sixth Quarterly Progress Report; University of Texas, October 1960.
- (31) ----- , and Tosh, L. W., Machine Translation Study (AD 255751); Seventh Quarterly Progress Report, January 1961.
- (32) ----- Machine Translation Study (AD 260254), Eighth Quarterly Progress Report; University of Texas, April 1961.
- (33) ----- Machine Translation Study (AD 273150), Tenth Quarterly Progress Report, University of Texas, October 1961.
- (34) Plath, Warren, Automatic Sentence Diagramming, (Paper 2); International Conference, Teddington, Middlesex, England.
- (35) Rhodes, I., "A New Approach to the Mechanical Syntactic Analysis of Russian" (unpublished report); National Bureau of Standards, 1959.

- (36) ----- Hindsight in Predictive Syntactic Analysis (AD 252659);
U. S. Department of Commerce, National Bureau of Standards,
30 November 1960.
- (37) Sherry, M. E., and Oettinger, A. G., "A New Model of Natural
Language for Predictive Syntactic Analysis," Proceedings of
the 4th London Symposium on Information Theory; London, 1960.
- (38) ----- Comprehensive Report on Predictive Syntactic Analysis,
Report NSF-7 on Mathematical Linguistics and Automatic Trans-
lation to the National Science Foundation by the Computation
Laboratory of Harvard University; 1961.
- (39) Taube, Mortimer - Meaning, Linguistic Structures and Storage
and Retrieval Systems (AD 120482); Documentation Incorporated,
Washington, D. C., February 1957.
- (40) Yngve, V. H., A Model and an Hypothesis for Language Structure,
Technical Report 369; M. I. T. Research Laboratory of
Electronics, Cambridge, Massachusetts, 1960.

III. ADAPTIVE CORRELATION TECHNIQUES

A. Statement of the Problem

It has become increasingly evident that the resources of a single human individual or even of many individuals organized into a system is not sufficient to meet the demands of modern decision making. For what is required in the process of decision making is a large amount of relevant information. To organize this information and present it to human decision makers requires the resources of a large-scale system that incorporates both men and machines in some systematic relationship. (U)

The objectives of this function are to specify a system capable of storing, retrieving, and correlating a large amount of information that is widely varied in content and interrelated both explicitly and implicitly. The information must be organized or structured in such a way that it is responsive to a wide variety of questions addressed to it. These questions are not restricted to questions about relationships explicitly stored in the system; they may pertain to relationships that are implied by these stored relationships. The system must be able to handle any relationships that are expressible in natural language, except for poetical or highly metaphorical relationships. (U)

The information to be stored and the questions to be answered are presented, as the result of pre-processing by the linguistic transformation processes, in a simple, unambiguous, non-redundant form. The Fact Correlation System must then process these inputs so that new information is retained for later use and so that questions are answered in a simple,

unambiguous, non-redundant form that may then be transformed by the linguistic transformation processes into acceptable English. (U)

B. General Functional Requirements

1. General

A system capable of achieving the objectives of a Fact Correlation System must be capable of performing functions that are essential to the completion of these objectives. This section formulates the conditions that an information processing system of this kind must satisfy in order to be an acceptable system. Two essential functions must be performed by the system so that it can correctly answer questions addressed to it:

- (a) The proper ordering and storage of input data.
- (b) The proper retrieval and correlation of stored data to answer questions.

In a more fundamental categorization these functions may be viewed simply as:

- (a) Putting information into the system.
- (b) Taking information out of the system.

The adaptive correlation processes operate within this frame of reference. (U)

It would be desirable to formulate a set of formal rules for data handling to perform these functions. However, not enough is known about the formal structure of complex information processes to be able to specify these rules, or even to say if the rules are specifiable. If such a set of formal rules could be specified, the preliminary work required for their specification might impose too great a price to pay for such a system. Therefore, it seems desirable to have the system develop techniques

for data handling as it processes information. If no formal rules can be developed to treat every case the system encounters, the system must devise empirical or heuristic methods for dealing with data. Such a system might be characterized as an adaptive or learning system. (U)

2. Ordering and Storage of Input Data

It is almost a triviality to demand that the system be capable of storing information in an ordered way; for in order for the system to operate upon information, the data must be organized so that data of a similar kind will be treated in the same way. Every known information processing system, including the human being, categorizes information so that it can be used by the processor in the necessary transformations--inductive, deductive, or inferential. Since the major tasks in this system will be performed by a machine, data must be organized in a structure accessible to machine operations. (U)

The processing of input information within the system must ensure that:

- (a) The information content of the input is correlated with respect to the categories of information already stored in the system.
- (b) The input is placed in the proper format so as to conform to the relevant data structures in memory.
- (c) The redundant information (in the sense of information already contained in the system) in the input is eliminated.
- (d) Contradictory information (in the sense of data contradicting information already in the system) is recognized, and action is initiated leading to resolve the contradictions.

The language transformation function will fulfill some of these tasks in ordering and structuring information; but the adaptive correlation processes must operate upon the structure and reformulate it with respect to all relationships within the information corpus. (U)

3. Retrieval and Correlation of Data

It is imperative that a Fact Correlation System be capable of responding to questions or requests about the information that has been received and transformed by the system. These questions must be correlated with data structures and operations in the same way the original information in sentences was correlated with the data structure. This task requires processes to determine the relevance of categories of stored information in terms of the intent of the question. In order to use the system's retrieval ability to the fullest extent, it may be necessary to phrase questions in accordance with theories of questioning, which are concerned with optimal ways of asking questions. (U)

Once the relevant data have been retrieved, various operations may be performed in order to ensure the consistency, accuracy, and completeness of an answer to the question completely. These operations may involve inductive and deductive processes, statistical correlations, and any other logical, mathematical, or heuristic techniques necessary. For a Fact Correlation System must be capable of providing its user with transformations of the information that has been received. Therefore, the system must be capable of performing inferential, generalizational, and associated functions to correlate the significant aspects of information. These

operations transform the data and increase the information content by obtaining information that was previously only implied by the existing data. Such transformations must be characterized precisely enough so that a man-machine system is capable of performing them. The transformation rules or techniques may be suggested by a variety of empirical and logical considerations. (U)

4. The Adaptive Nature of the System

If this system is to function effectively, it should be a learning or adaptive system. Many of the procedures for data structuring and data handling will be unknown by the system at first, so the system should be given rules for learning to structure and process data properly. Later the system should learn by enlarging and improving its techniques for data handling. Thus learning by the system includes:

- (a) Associating related concepts in such a way that degrees of association or relevance are determined.
- (b) Inferring new (implied) relationships among concepts and determining their relevance to the old relationships.
- (c) Correlating information requests with information available and requesting additional information to be transmitted by the user.

This list is not exhaustive; its extension depends upon increased knowledge about the actual processes of learning. (U)

To develop rules for data handling adaptively, the system must perform inferential functions such as associating, classifying, generalizing, hypothesizing, deducing, inducing, memorizing, and similar functions connected with learning. These functions, as applied to the system, are not identical to corresponding human functions, for they are not meant to

imply complex human behavior; but they are sufficiently similar in kind so that it is useful to apply the same terms to identify them. (U)

The following sections characterize those functions that appear to comprise the process of learning; by synthesizing these functions, a synthesis of the learning process would result. In other words, it has been hypothesized that certain simple functions, which are analogous in behavior to human functions, can be characterized so that in combination these functions form a model of the learning process. (U)

To be useful such functions must be characterized so as to be operationally definable within the context of system performance. These characterizations reflect the verbal nature of the problems; that is, these terms must be directly applicable to general problems involved in the manipulation of linguistic symbols. (U)

In developing rules for handling data, the system may initially develop incorrect ones because of a lack of pertinent or sufficient information. This circumstance is similar to human action in an unfamiliar area. Human beings can learn about subjects unfamiliar to them by asking other human beings about these subjects. The adaptive correlation process should also be able to ask questions about a subject in order to obtain sufficient information to formulate sound data handling rules. Thus the system would have the ability to gather information from its internal structure and its environment to re-enforce its primary task of answering questions. (U)

G. Review of Present Techniques

There are many adaptive or learning systems in existence. Some of these systems are computer programs, some are mathematical machines, and still others are physical machines. These systems use a variety of means to achieve diverse ends. This section summarizes some of the important characteristics of these systems, classifies and reviews existing techniques, and compares them with the adaptive learning requirements of a Fact Correlation System. (U)

Adaptive, learning, or self-organizing systems may be categorized generically within the field of artificial intelligence. This field is hampered in its development by a lack of unity and integration. The aims of this work are ambitious, but the state-of-the-art is primitive. Consequently, the field appears to be an ad hoc collection of problems, techniques, and points of view. This confusion is further complicated by the interdisciplinary nature of many of the theories and techniques. There is no agreement on a satisfactory name for the field. Theories that are more or less relevant to an adaptive Fact Correlation System have been referred to by various names such as cybernetics, automata theory, self-organizing systems, artificial intelligence, information sciences, and adaptive systems. No classification scheme neatly encompasses such a diverse collection of concepts. (U)

The literature that has been generated in this field may have some coherent focus; but, because of the diversity of both content and approach, its relevance is extremely difficult to evaluate. The ranges from the

banally general and trivial to the complicatedly detailed. Any attempt to review the work done in the field must begin with a methodological task of classification. The task is particularly difficult because of newness of the work. A profusion of points of view, of purposes, and of degrees of generality exists. Any classification scheme adopted may seem arbitrary; it may express the point of view of the reviewer rather than the inherent taxonomy of the field. (U)

Several criteria for the categorization are used in this survey. To some extent these criteria reflect a bias toward the field of automatic information retrieval or fact correlation. Basically, three main tendencies predominate in the field: the genotypic, the monotypic, and automata theory. (U)

1. General Classifications

(a) Genotypic Approach - The first trend reflects a bio-structural bias. Its characteristic is the interest in structure rather than behaviour. Usually the motivation for this approach is a desire to construct a brain analog or, at least, a structure with brain-like properties. Such an approach has been termed genotypic by one of its outstanding proponents, Dr. Frank Rosenblatt:

"In the genotypic approach the properties of the components may be fully specified, but the organization of the network is specified only in part by constraints and probability distributions, which generate a class of systems rather than a specific design. The genotypic approach, then, is concerned with the properties of systems which conform to designated laws of organization, rather than with the logical function realized by a particular system." (U)

Several important characteristics are implicit in this point of view. The first is probability theory, which plays an important and special role in the theory of brain-like studies. In an analysis of such models the chief interest is in the properties of the class of systems that is generated by particular rules of organization; such properties are best described statistically. In this context John Von Neumann states:

"Logic and mathematics in the central nervous system must structurally be essentially different from those languages to which our common experience refers... When we talk mathematics, we may be discussing a secondary language, built on the primary language truly used by the central nervous system. Thus, the outward forms of our mathematics are not absolutely relevant from the point of view of evaluating what the mathematical or logical language truly used by the central nervous system is. Whatever the system is, it cannot differ from what we consciously and explicitly consider as mathematics."

The essence of Von Neumann's concept that symbolic logic or Boolean Algebra, successful as they were in producing modern digital computing machines, are not suitable to describe the operations of the human brain. (U)

The second characteristic of the structural approach stresses the organization of basic rather than the more complex processes. It is from this aspect of the approach that the emphasis upon perception rather than logical processes derives. This feature of the structural approach detracts from its value to the information retrieval theorist. One of the more important tasks is to translate the results of the genotypic theory into the language of information retrieval or fact correlation theory. Specifically, an attempt must be made to generalize the self-organizing features of perceptron-like systems. The present concepts

are limited to the discrimination and generalization of special patterns; the development of useful applications implies an extension to the generalized recognition, discrimination, and generalization of abstract logical patterns. (U)

The genotypic approach stresses self-organization. In this context it is appropriate to introduce the distinction between learning and self-organizing systems. A learning system improves its behaviour with increased experience in accordance with some preselected criteria. A self-organizing system, on the other hand, formulates its own criteria. The basic distinction between learning and self-organizing systems is in the amount of information that each system possesses about its environment in its initial state. A self-organizing system starts with no information about the structure of its environment; it must elicit the information from the environment. In contra-distinction, a learning system starts with some initial information in the form of criteria for improvement or heuristic techniques applicable to a given environment. (U)

(b) Monotypic Approach - The second trend, which may be termed monotypic (after Rosenblatt), is characterized by its interest in behaviour rather than structure. Its first attribute is the strict definition of performance desiderata required from the model. A good description of this aspect of the monotypic approach is given by Culbertson:

"Neuro-anatomy and Neurophysiology have not yet developed far enough to tell us the detailed inter-connections holding within human or animal nets... We cannot start with specified nerve nets, and then... determine their properties. Instead it is the reverse problem which always occurs in dealing with organic

behaviour. We are given at best the vaguely defined properties of an unknown net and from these must determine what the structure of that net might possibly be."

The designer of this type of system thus specifies particular data-processing operations, input-output or stimulus-response functions, or remembering-regenerating operations. (U)

A second aspect of this approach is its interest in heuristic search techniques. The term heuristic is used in a number of senses depending upon the point of view of the system designer. For some designers heuristic techniques are those techniques characteristic of human problem-solving procedures; for others, any efficient problem-solving procedure falls into the category of heuristics. There are several typical research activities associated with heuristic techniques:

- (1) Setting up new problems as subgoals.
- (2) Replacing problems by new ones whose relation to the original is less immediate. These problems are not equivalent but only heuristically similar to the original, and their solution need not directly lead to the solution of the original.

In either case the machine must have facilities for recognizing problems as members of heuristic categories in order to select appropriate methods. (U)

The monotypic approach to the pattern recognition problem is sharply distinct from the genotypic approach. Monotypic theories attempt to construct a notion of similarity. The construction of such a concept hinges upon the solution of two problems: the possession of an appropriate measure of goodness of fit with respect to a template; the evolution of efficient and sensible matching procedures--i.e., the matching of a spatial

pattern by subjecting it to rotations or translations. (U)

(c) Automata Theory - Properly speaking the theory of automata does not belong to the same classification scheme as the two previous categories. Automata theory is not an approach towards the implementation of a learning or self-organizing system. Rather, it is a self-sufficient mathematical discipline for describing the relations between the structure and the behaviour of systems. Most of the results of the automata theory pertain to the problems of computability or unsolvability of certain functions. These theorems shed light upon the limits of performance to which all systems are subject. (U)

The following sections contain a brief description of various learning and/or self-organizing systems. Some of the systems exist only in theory; some have actually been implemented. These systems are representative of the advanced thinking in the field of artificial intelligence. (U)

2. Perceptron Theory

The perceptron theory, an example of the genotypic approach, is essentially the work of Dr. Frank Rosenblatt; his intention was to create a structure that exhibits brain-like properties. Of all the models of self-organizing systems, the perceptron has the distinct advantage of being the most theoretically developed. A respectable corpus of theorems has been derived from axiomatic foundations. (U)

Rosenblatt considers the brain's capacity for recognition or identification as its most distinctive characteristic.

"The most important advantage of a brain over a computer may well be its capacity for recognition or identification--its ability to say that a new set of data stimulus pattern, situation, or problem is similar to some class or type which it has previously encountered even in the absence of a specifically enumerated set of criteria. A machine which can recognize in this sense and form its own categories for identification of phenomena must be capable of generalization or concept formation in the same sense that these terms are used to describe human perceptual performance."

In attempting to construct a structure exhibiting these properties, Rosenblatt concludes that a system's memory must possess the following characteristics:

- (a) The exact reproducibility of remembered material must be abandoned in favor of an associative memory that stores only information essential for classification or discrimination of stimuli.
- (b) The notion of errorless retention must be abandoned in order to permit redundant use of the same unit.

"The appropriate connection will not be formed by assigning unique channels to every stimulus but by designing the system so that it responds to a statistical bias in large populations of impulses." (U)

Rosenblatt stipulates that the system should respond to stimuli in both a forced and a free way. A system's response is forced if there is an initial training period during which the trainer labels stimuli as belonging to one of the specified categories. On the other hand, the system's response is free if its responses are dictated by a spontaneous recognition of similarity.

The key structural concept underlying the perceptron theory is a

postulate originally stated by Hebb. Hebb's postulate was originally meant to provide a feasible description of the basic dynamics of brain processes.

"When an atom of a cell A is near enough to excite cell B and repeatedly or persistently takes part in firing it, some growth processes or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B is increased."

As a result of this postulate, Hebb suggests that "...a repeated stimulation of specific receptors will lead slowly to the formation of an 'assembly' of association area cells which can act briefly as a closed system after stimulation has ceased. This prolongs the time during which the structural changes of learning can occur and constitutes the simplest instance of a representative process."

The problem that confronted Rosenblatt was to construct a system that would embody Hebb's postulate and demonstrate the emergence of cell assemblies. The perceptron is constructed out three types of elementary cells, which are labeled S, A, and R for sensory, associative, and responsive, respectively. The layer of S-cells is sensitized by energy impinging from the external environment. If the energy flux impinging upon an S-cell exceeds a certain level, the S-cell fires a signal. (U)

In the simplest type of a perceptron, the S-layer is connected to the A-layer, which is connected to the R-layer. The connections between cells may be either excitative or inhibitive. For each distribution of stimuli over the sensory layer of cells there is a corresponding response; that is, the stimuli develop a pattern of activity over the associative layer and this pattern leads to a characteristic response configuration.

As a result of excitatory and inhibitory activities set up by the stimuli, a reverberatory pattern firing persists even after the stimuli ceases or is withdraw. This phenomenon is similar to the short term memory in human beings and is quite important in the perceptron theory. (U)

The set of all A-layer cells that fire when a stimulus S_1 is applied is called a super set. The strength of a signal emitted by any A-cell depends, however, upon its past history. In the simplest type of perceptron the A-cells that participate in a firing activity resulting in the response of a R-unit are reinforced; the strength of a signal emitted from the reinforced cell is increased. Thus the fact that a stimulus S_1 occurred changes the physical structure of the system. In Hebb's language, phase assemblies or groups of cells, which are permanently associated with a stimulus or a class of stimuli, emerge. Is there, then, functions defined over the subsets that possess especially interesting properties. (U)

The Theory of Statistical Separability investigates whether there is some simple parameter of the value distribution over the subsets that distinguishes the best response to S_t --a response that has been previously associated with S_t or with the class of stimuli that includes S_t as a member. The theory demonstrates that two such parameters exist:

- (a) The sum of the values in each subset.
- (b) The mean of the values in each subset.

It has been established on a rigorous mathematical basis that for certain simple types of perceptrons these parameters will exhibit stability independently of the manner in which the stimuli are presented. Even with a

single logical level, single layer of assembly cells, a perceptron is capable of a number of non-trivial performances involving selective attention and recall. (U)

A few critical remarks pertain to the application of perceptron theory in a Fact Correlation System. The perceptron is a device basically designed to perform classifying operation upon sensory inputs rather than upon the more abstract logical patterns. To perform useful functions with respect to a Fact Correlation System, the perceptron should be endowed with the capacity to operate upon linguistic inputs. It would have to perceive similarities and perform generalizations on inputs whose structure is not reflected by its sensory units. Whether this function can be performed is the problem of the future developments and extensions of the theory. (U)

The perceptron is essentially a wasteful system. It performs many unnecessary generalizations as well as many unnecessary classifications. Although generalization and recognition may be the basis of all mental processes, it is difficult to see how the perceptron could engage in more complex operations such as hypothesis formation. The perceptron is a primitive device, although it is an interesting one; a considerable amount of theoretical development separates it from being an efficient and intelligent data processor. (U)

3. General Problem Solver

The most interesting example of a monotypic or behaviour-simulating system is undoubtedly the General Problem Solver (GPS) developed by

Newell, Simon, and Shaw. The GPS is a system that attempts to parallel mental rather than brain processes; it stresses mental behaviour rather than brain structure. (U)

The major features of the existing program for the GPS that are worthy of discussion are:

- (a) The recursive nature of its problem solving activity.
- (b) The separation of problem content from problem solving technique as a way to increase the generality of the program.
- (c) The two general problem solving techniques that now constitute its repertoire: means-end analysis and planning.
- (d) The memory and program organization used to mechanize the program.
- (e) The capacity of the program to operate upon itself as an environment and then to produce programs better adapted to their tasks.

Newell, Simon, and Shaw are largely concerned with problems of heuristics, which they define simply as "...things that aid discovery." It is important to emphasize that "...systems of heuristics seldom provide infallible guidance. They give practical knowledge possessing only empirical validity." The main features of the GPS are methods for obtaining this practical knowledge. (U)

The theory of problem solving is concerned with discovering and understanding systems of heuristics. The fundamental entities within the domain of heuristics may be grouped into two categories: objects and operators. An operator is quite generally something that can be applied to certain objects to produce different objects. (U)

Objects can be characterized by the features that they possess and by the differences that can be observed between pairs of objects. These primitive terms are precisely the kind into which problems or, more exactly, problem solving could be resolved. Thus the typical stages in solving a problem are to find a way of transforming an object into another object, to find an object possessing a given feature, or to modify an object so that a given operator may be applied to it. For example, in the problem of proving theorems the objects are theorems while the operators are the admissible rules of inference. To prove a theorem is to transform some initial objects--the axioms--into a specified object--the desired theorem. (U)

The key to the approach to the GPS lies in insistence that the method of general heuristics may be separated from the specific task environment. Indeed, theoretically, this general method is equally applicable to all task environments under all problem-solving situations. The GPS is a program for working on tasks in an environment consisting of objects and operators. (U)

The principal components of the GPS are a set of goal types, which are the major units for organizing the problem solving process. There are essentially three types of goals:

- (a) To transform one object into another.
- (b) To apply an operator to an object.
- (c) To reduce a difference between two given objects.

For each goal there is a corresponding set of methods. If a goal is set,

one of the appropriate methods is chosen. The recursive structure of the GPS is due to the nature of methods used; for methods operate by establishing subgoals that demand that other methods be used in turn. The complex recursive interlacing of the methods and goals on many levels leads to the emergence of organized systems of heuristics. (U)

What methods does the GPS use? The same generality must apply to the specification of methods as to the specification of goals; otherwise, independence from a particular task environment is not attainable. Therefore, the goal of transforming an object into another object has the following set of methods:

- (a) Matching the two objects.
- (b) Discovering the difference between the two objects.
- (c) Establishing the subgoal of reducing the difference between the two objects by transforming one of the objects into another, thereby possibly obtaining a reduction in difference between the original objects.
- (d) Establishing a new transformation subgoal.

The set of methods associated with the goal of reducing the difference between two objects consists of:

- (a) Searching for an operator that is relevant to the difference.
- (b) If such an operator is found, establishing a new goal for the operator for the difference.

There is also a method associated with the goal of applying an operator to the object. It consists of:

- (a) Determining if the operator can be applied by establishing the transformation goal. This function will determine if the object can be transformed into the input form of the operator.

- (b) If successful, producing the output object.

The recursive interrelationship of goals is apparent. Transformation goals generate reduction goals and new transformation goals; reduction goals generate new application goals; and application goals generate new transformation and application goals. (U)

When the GPS is solving problems in a particular task environment, the performance program consists of two parts:

- (a) The GPS proper in which the goal types and methods are completely independent of subject matter and not modified in any way when the GPS is applied to a new task environment.
- (b) The specifications of the particular task environment, its objects, its operators, and its differences.

The learning programs may be applied to both of these component parts of the performance programs. That is, some learning programs must be directed towards obtaining knowledge about the new task environment to which the GPS might apply. The context of what is going to be learned by such a learning program is specific to a given task environment. On the other hand, learning programs may modify the GPS proper. A program of such nature would attempt to enrich the heuristics of GPS--perhaps by adding new types of goals to the existing ones. (U)

Newell, Simon, and Shaw attach greater importance to learning programs pertaining to new task environments. Indeed the GPS is not really equipped with any substantial capacity for modifying its own program. There is, however, a group of programs designed to improve the performance of the GPS in any new task environment. These programs are quite general in character; that is, they are independent of any

particular task environment. Only the content of what is learned is specific to a given task environment. (U)

True to their general approach, the creators of the GPS regard human learning situations as a model and a guide.

"Learning to characterize in an effective way the task environment is an important and prevalent kind of human learning. A problem solver who is experienced in a particular environment will notice features that will be unnoticed by an inexperienced person... After the problem solver has learned to recognize and attend to a useful set of differences, other things remain to be learned...., i.e., searching the list of operators until one is found that affects the...(relevant) difference."

This passage indicates that the essential problems of learning for GPS are:

- (a) Given the set operators and differences in a task environment, find a good set of connections associating relevant operators with several differences.
- (b) Given the object and operators in a task environment, find a good set of differences for that environment.
- (c) Find new operators other than the set initially given. (U)

The first of these problems is solved by the GPS learning program in the following fashion. An operator is applied to an object or a set of objects; the resulting set of objects is then compared to the original set and examined with respect to the set of all possible differences. A table is developed in which the rows represent operators; the columns, differences. If a one is placed at an intersection of a row and column, the difference represented by the column supplies to the operator represented by the row. From the way the learning program is constructed it is apparent that a sort of inclusion is performed. The tests take place

in a particular environment, yet the conclusions are general; i.e., they apply to the particular operator in all task environments. (U)

The second learning task may be characterized thus:

"...given the objects and operators in a task environment and a set of basic processes for detecting and discriminating features of objects, find a good set of differences between pairs of objects for the GPS in that environment."

The successful fulfillment of objects implied in this task depends upon the existence of a suitable programming language capable of generating object programs for detecting differences. The difference program language (DPL) developed for this purpose has two parts, a general part with a number of processes to operate on sets and lists and a specific part with particular processes for each task environment. The general part of the DPL consists of seventeen processes. Out of these basic seventeen processes the DPL is capable of constructing programs whose chief distinction is their abstraction; that is, these programs start with detailed comparisons and gradually remove distinctions by the successive application of DPL operations. (U)

It is possible to generate learning programs with the DPL processes on a trial and error basis. Such a procedure would, however, violate the principles of the GPS.

"...we have no reason for a priori confidence--that the learning will be accomplished within a reasonable time span. The space of possible differences is large and human beings guide their trial and error searches through it with a variety of heuristics. If the learning program is to operate in real time, it must make use of additional information in selecting and testing candidates for the set of differences."

Newell, Simon, and Shaw conceive of the learning program itself as a

problem solver. Thus the GPS may be used recursively to generate its own learning program. In this sense the GPS has the property of reflexivity that is involved in self-organization. (U)

An application to the GPS to a general learning situation is illustrated in the following schematic and oversimplified presentation. Let the original task environment be labeled the A-environment, the logic environment. The A-environment comprises A-objects, A-operators, and A-designations. The learning problem is to find a good set of A-differences. Let the B-environment be the environment of the learning problems; the B-objects are the sets of A-differences. The learning program is then defined as the searching for B-objects that result in good problem solving in the A-environment. Improving the performance of the learning program may then be stated as a task of creating B-operators (operators for modifying sets of A-differences) and B-differences (tests for comparing B-objects) and of discovering how to state the learning goal as the B-goal. The B-operators work upon sets of A-differences. They may add A-differences to a set, delete A-differences from a set, or modify existing members of a set. (U)

A new level of generality is achieved by this application of the DPL to the GPS itself. The system now is endowed with the capacity to define new differences extensionally. The learning task generates its own goals, the B-goals. The highest B-goal for the learning process is a criterion for a good set of A-differences. Ultimately a good set of differences is one that is effective for solving problems in the A-environment. But to permit intelligent learning, other ways must be

found to characterize a good set of differences so that the GPS can evaluate any improvement it is achieving in its attempt to learn. (U)

Newell, Simon, and Shaw provide the B-environment with a set of criteria for a good set of differences. The rationale for these criteria can be established if:

"...the differences are the diagnostic tests that GPS uses to determine what operators it should apply in a given situation. The diagnosis will be most efficient if each difference points to the application of one and only one operator, if each difference affects one and only one difference, if the effect of an operator is predictable, and if a difference is always detectable between non-identical objects."

To weld all the separate reduce difference goals in the B-environment into a single effective goal, a priority order for the differences must be established. The GPS attempts to improve the set of differences by first attacking the lower priority criteria; it always returns to the higher priority criteria if the lower criteria are no longer satisfied after the set has been modified. (U)

In summary, the learning task is defined in the environment whose objects are sets of differences in the task environment. The operators in the environment of the learning task are entities that permit the manipulation of the objects--i.e., the sets of differences in the task environment. The differences in the learning environment correspond to the various criteria for the creation of good sets in the original task environment. (U)

The GPS stands out as the most ambitious and most successful attempt to construct a self-organizing adaptive system. Any criticisms

of the GPS are essentially directed towards its inability to use the enormous store of information at the disposal of human beings. This information could be elicited from men by a system oriented towards such learning procedures. The criticism may be briefly restated: the GPS is capable of learning, but it is not capable of being taught. Therein lies its chief weakness. Future elaborations of the GPS should then attempt to include the capability of learning by eliciting information actively from those entities that possess it. Success in such venture will depend upon the extensiveness and flexibility of communications between the GPS and external intelligence. Some important advances in the theory of meaning may provide a system like GPS with another dimension to its learning capability. (U)

On another level, the concept of differences, which the heuristics of GPS uses almost exclusively, is overemphasized. More extensive research into the nature of intelligent processes will probably lead to other heuristic patterns, which could be more powerful. (U)

Finally, the GPS is self-limited. Its search for subgoals leads through a predetermined pattern, even if the predetermination is implicit and accidental. Once a subgoal has been established, its contribution to the primary goal is fixed; there is a single return path. The resolution of this problem is easy to state but difficult to solve: Once a subgoal has been attained, the information and functions so discovered should designate the next function to be performed; the learning of one aspect of a problem should lead directly to another aspect anywhere within

the complex of the problem, and not necessarily to the primary goal. The solution of this problem is necessary if the GPS is to become a non-deterministic learning process. (U)

4. Pandemonium

An interesting model of a process that adaptively improves its ability to handle certain pattern recognition problems, which cannot be adequately specified in advance, is Pandemonium, designed by Dr. Oliver Selfridge. The GPS model is based upon a concept of heuristic processes that parallel human mental operations; Dr. Selfridge's model relies upon concepts inherent in the theory of communication. (U)

In communication systems the measure of similarity between two messages is an integral of the squared absolute value of their difference. Two messages are then identical if the first variation of this integral vanishes. This concept is defined as:

$$\delta \int |M_1(T) - M_2(T)|^2 dt = 0$$

where M_1 and M_2 are some continuous messages and T is time. This idea is the basis of the model for Pandemonium. (U)

Pandemonium is organized in a four-leveled structure. On the lowest level the elements, facetiously called data-demons, may be thought of as a sensorium. These elements transmit the data to the next higher level of elements, which perform computational functions on the data. These computational functions essentially evaluate certain properties of the data. The output of these functions serves as an input to the next

level of elements. The task of the third level is cognitive; that is, each of the elements weights the inputs as the evidence for computing the similarity of data to a specific pattern. The inputs might be the angles between lines or the presence of implication signs in logical formulas. Then an element on the cognitive level might look for squareness or for a picture with objects resembling missiles. The cognitive element looking for the property of squareness attaches a greater weight to a computational function that measures the angles between intersecting lines than a function that recognizes circles. The cognitive element emits a signal whose magnitude is controlled by the closeness of the data to the representative pattern. Finally, on the highest level a decision element selects the element that emits the strongest signal. (U)

This description of the operation of Pandemonium does not indicate that the system is adaptive. Indeed, the actual model simply recognizes Morse code characters and is not capable of learning. However, the model was designed to exhibit adaptive learning properties, and Dr. Selfridge has indicated how the system may be extended to become an adaptive process. (U)

There are several possible lines of development for Pandemonium to acquire adaptive properties. The first pertains to the weighting of the computational elements by the cognitive elements. The weighting is being altered so as to maximize the score of the system. How then is the score obtained? The system could be monitored, at least initially, so as to record the results of small changes in weight assignments in terms of the number of correct responses. Such a procedure is essentially an application

of the hill climbing method, whereby small changes leading to desirable results are continued. The problem of optimizing the weight assignment may be viewed as a problem of finding a set of values (weights) λ_j^i such that the signals generated by the cognitive elements,

$$d_i = \sum_{j=1}^n \lambda_j^i \delta_j$$

lead to optimum results. Here δ_j stands for a computational function. (U)

Another method of extending the system to include adaptive properties pertains to the selection of new computational elements. In general, the selection of the set of the computational elements is not necessarily the most suitable for the performance of a given set of tasks. There are essentially two techniques available for the formation of new computational elements and the elimination of inefficient ones. The first technique attaches scores to the computational elements in accordance with the degree of their participation in the cognitive functions of the system:

$$w_i = \sum_j \lambda_j^i$$

where λ_j^i is, as previously, the weight that the j^{th} computational function assumes in the i^{th} cognitive process. The generation of more suitable assemblies of the computational elements occurs by eliminating the elements with low scores and generating new functions by mutating the survivors and reapportioning the weights. Such an approach usually requires that the computational function be represented in some canonical form so that the insertion of changes does not have the effect of rewriting

the program. The second technique is simply to generate the new logical functions from the existing computational elements. (U)

Several remarks are pertinent to the suitability of this model for a Fact Correlation System. First, the cognitive processes necessary for the execution of the system's tasks are assumed as given. This assumption implies that for complex fact correlation tasks the repertory of the cognitive processes or the assembly of cognitive elements must be immense. Such an approach is inherently inefficient. A system in which the cognitive functions are synthesized out of simpler elements would be much more economical. (U)

There is at present no indication that the frequency count method of eliminating inefficient computational functions and forming new computational functions on the basis of probabilistic rearrangements is a feasible, let alone economical, method of endowing the system with adaptive properties. Such unintelligent processes waste large amounts of information in the environment of the system. This information could be used as a guide for forming new cognitive and computational elements. (U)

Finally, Pandemonium is capable of acts of recognition. Recognition, however, is only one of the constituent elements entering into the set of cognitive processes required for fact correlation. Pandemonium is not capable, for example, of performing complex inferential processes; neither does it seem plausible that a Pandemonium-like system, based essentially upon a recognition capability, will be able to perform semantic and syntactic operations that a Fact Correlation System must be

capable of performing. (U)

5. Experiments in Machine Learning

Pandemonium and the GPS both emphasize what might be called the cognitive properties of learning processes. The GPS, for example, strives to obtain a suitable description of properties of the task environment before attempting a solution. The problem of constructing programs that will implement the solution is a relatively minor problem. In contrast to this approach, the learning machine of R. M. Friedberg and experiments with machine learning by H. Campaigne assume a different approach. Both men attempt to evolve a technique for efficiently generating new programs independently of information about the task environment. (U)

One way of grasping the essential difference between the two approaches is to visualize two spaces S_1 and S_2 . S_1 is a problem space; S_2 is a solution space. A description of a problem in S_1 corresponds to the selection of a point in it. A description of a class of problems corresponds to the selection of a subspace of S_1 . Now, imagine a bundle of lines originating with every point in S_1 and leading to S_2 . Thus there is a one-to-many mapping of the problem space into the solution space. Only a few or possibly one of the lines within the bundle will lead to the desired solution. Nevertheless, a fair definition of the problem drastically reduces the region within the solution space where the search for the right solution must be conducted. This gain in the economy of search processes within the solution space is balanced by the necessity for a more intensive search for the point or subspace in the problem space. However, considerations of economics and efficiency are not necessarily

paramount; at least from theoretical point of view, other criteria may be more important. Such criteria may include the capabilities for self-organization or learning. (U)

The essential point of view adopted by Friedberg is:

"If...the machine is to try out methods (of solution) and select better ones, we must present it a priori with a well defined universe of methods from which it must choose those to be tried. If this universe is small, then the 'inventiveness' of the machine is severely limited and the value of the methods that it develops depends more on our astuteness in choosing a universe containing good methods than on the ability of the learning procedure to pick the best method from among these in the universe... In order to give the learner 'free hand' we should present it with a universe which, although well defined, is so large and varied that we are not even acquainted with the forms of all the methods it contains... We must expect that in any universe so uncensored the majority of methods are useless."

Friedberg considers it essential to devise a scheme for classifying methods of solutions (programs) without knowing what these programs do a priori. In this way the success or failure of a method will be interpreted as a reflection not only on that method but on all related methods. A large universe of methods will then be sifted in a relatively small number of trials. (U)

Rather arbitrarily, Friedberg assigns all programs with a certain instruction in the same location to the same class. A given program may, however, belong to several classes. Friedberg defends this approach by saying that:

"Form and intent...are related quite discontinuously in the compact economical programs that programmers write, but a learning machine would probably develop much more inefficient programs in which many irrelevant instructions were scattered among instructions that were essential to the intent. Among such

programs, slight changes in form might very well correspond to slight changes in intent so that the programs falling into the same classes tended to perform similar acts."

There is an obvious objection to this method; there is no clearly recognizable continuum of intents for the machine programs. A change in one construction may transform a useful program into a nonsensical one. (U)

To test the plausibility of his assumptions, Friedberg designed a series of hypothetical computers that are modified by a second element in the system, the Teacher. A third element of the system, the Learner, evaluates the different instructions that appear in computer's program on different occasions. The experiment was run by simulating these three elements on the IBM 704 computer. The computer element (named Herman in the last experiment) had a simple logic such that every number of 14 bits was a meaningful instruction and every sequence of 64 instructions constituted a program. (U)

The repertory of the computer consisted of four 14-bit instructions. The program consisted of 64 instructions stored in consecutive locations; the program operated upon stored data in a separate set of 64 consecutive locations. The object of the experiment was for the computer to transfer the input data into output locations according to a scheme unknown to either Herman or the Learner. Before each trial the Teacher placed bits chosen at random into certain data locations. A trial continued until either the last or sixty-fourth instruction was reached or a preset time limit was exceeded. (U)

The Learner monitors instructions currently occupying a given location. Since there can be 2^{14} different instructions in a single location, it is impractical to keep a record of each; instead, two instructions are on record at any time for each location. For each location one of the two instructions is active; the other, inactive. The Learner can change the program in two ways: it can interchange active with inactive instructions, and it can replace an instruction at random--that is, one of the 128 instructions is replaced by a new configuration of bits. Both methods are regulated by two parameters, a success number and a state number. These parameters play a crucial role in Herman's learning process. (U)

The success number indicates how well an instruction has served over a sequence of trials. If a program is deemed successful by the Teacher, the success number of each active instruction is incremented by one or two. If a new instruction is randomly introduced, its initial success number is set to a value related to the average value of the success numbers of other instructions in the program. The average success number is kept approximately constant by a down-scaling procedure that effectively depresses the success number of instructions not recently involved in successful trials. (U)

The state number is a mechanism that governs the interchanges between active and inactive instructions. Each time a failure is reported, one location is subjected to criticism. The absolute difference between its state number and the success number of its inactive instruction

becomes the new state number; if the old state number was less than the success number of the inactive instruction, the latter becomes active. (U)

The results of the experiments with Herman produced a particular learning pattern. There were three stages in the experiment. During the first stage of 60,000 trials, actual success improved from zero to slightly less than 50 percent. The lack of success was attributable primarily to time failures. During stage two of 90,000 trials success hovered around 45 percent. "One may contend that during stage two the Learner was improving Herman's program in a way that did not immediately increase the frequency of success but that gradually increased the probability that further modifications would result in a perfect program." In stage three, the last 50,000 trials, the learning process was essentially complete and Herman was successful on 100 percent of the trials. (U)

The random generation of computer programs represents a basically different approach to the learning processes. The primary features of this approach are:

- (a) Freedom from the necessity of exploring a task-environment in detail prior to an attempted solution.
- (b) Severance of dependence between the methods of solution used by the computer and the patterns inherent in stored programs.
- (c) Independence from human bias towards certain methods of solution inherent in the systems relying upon heuristic search patterns.
- (d) Increase originality in the sense of a system's ability to generate novel methods of solution.
- (e) Extreme inefficiency.

It is difficult to evaluate the advantages and disadvantages of the two approaches on the basis of existing examples. All of them, including Friedberg's machine and experiments by Campaigne and Amarel, are either too specialized or too primitive to cast much light upon the status of such techniques. (U)

It is worth pointing out, however, that the two approaches are not completely exclusive. A compromise is possible whereby an exploratory investigation of the task environment will impose restrictions upon the generation of a method of solution; but these restrictions will not necessarily force a particular solution as a function of problem description. If the compromise solution is acceptable, then the key problem seems to be one of integrating the two approaches. The pertinent questions to explore are:

- (a) The proper means of transmitting descriptions of a task environment to the program generating elements of the system.
- (b) The evolution of techniques for program generation based upon a partial description of the environment.
- (c) The relation between the originality of the system and the degree of dependence of program generation upon the prior exploration of the task environment.

The final critique is a different kind. Despite any theoretical potential for generating programs, the question of efficiency remains a paramount problem. (U)

6. The Theory of Automata

There is no uniformity in the way the term automaton is used. Perhaps the best way to characterize those devices that are regarded

as automata is to say that their purpose is to transform information; at least, these devices transmit or receive information. Thus, in addition to machines that behave intelligently and computers, telephone switching networks, meters of all kinds, and servo-mechanisms are conveniently studied as application of automata theory. (U)

The theory of automata has a special subject matter, and not all the knowledge about devices classified as automata comes under this heading. The subject matter is abstract; it deals with logical or mathematical consequences of precisely formulated concepts. The basic terms used by the theory in the description of automata are: input, output, input history, and state. An input is an entity capable of assuming values specified from a certain range of values. The same definition applies to output. The inputs are, however, beyond the control of the automaton. The value of the output may be considered as an entity associated with a state of the automaton and a particular value of an input. (U)

An automaton may then be conveniently described as a set of quadruplets of the form $[S_i S_j q_m p_n]$, where S refers to a state or internal configuration of the automaton, q is a particular input, and p is an output. An automaton that is in a state S_i and receives an input q_m will produce an output p_n and pass into a new state S_j . An automaton may be probabilistic or deterministic according to whether the output is determined uniquely or probabilistically. The probabilistic automaton may be formally described as a set $[S, P_s(S), q, P_p(p)]$, where P_s and P_p are random variables defined on sets of states and outputs, respectively. (U)

Automata may be further classified according to their growth potential. An automaton that grows but cannot become arbitrarily large is no more powerful than a non-growing automaton. If there is a continuum of values for each input and output, then the automaton is continuous; otherwise, an automaton is called synchronous if its input and output history could be described completely as occurring during discrete moments of time. The history of non-synchronous automaton cannot be described in this manner because its action is understood only in terms of the continuity of time. (U)

The theory of automata deals with two main areas. The first is the study of rather general structural and behavioral properties of automata. Most of these problems are not problems of efficiency of design nor of efficiency of construction, although there are some exceptions. An existence theorem about automata satisfying certain conditions would be a typical example of work in the first area. The second domain is the study of various behavioral descriptions of automata. An algorithm converting one kind of behavioral description into another is an example of work in this area. (It is evident that the division in the field of automata theory parallels the division established at the beginning of this section on "Adaptive Correlation Techniques.") (U)

Since there are five categories for classifying automata, there are thirty-two possible subclasses. Only some of these subclasses have received attention in the literature. The fixed, discrete, synchronous, deterministic, finite, and state automaton is the most elementary of all

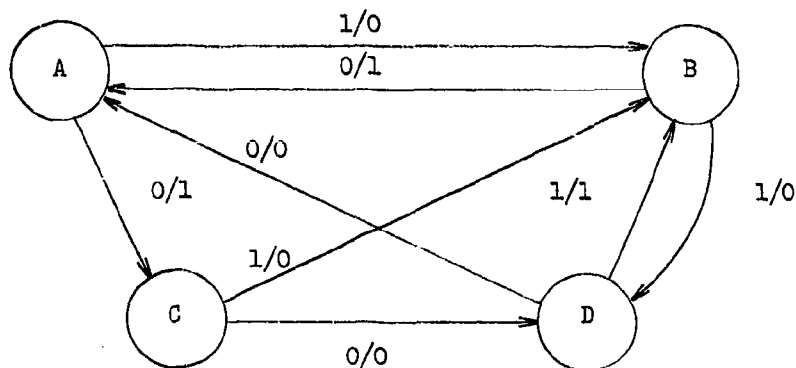


FIGURE 3-1. State Graph of a Finite Automaton

the logically possible types; it has also been the most thoroughly investigated. To facilitate the study of such mathematical machines, a device called a state graph has been introduced. A state graph is a diagram showing the connections between states and their input-output relations, as shown in Figure 3-1. If the automaton starts in state A and its input is 1, then its output is 0 and the next state is B. If the input history is 1101 and the automaton starts in state B, then the output sequence is 01110 and the final state of the automaton is B. (U)

An important concept in connection with state graphs is reduction. Two state graphs are equivalent if every input sequence yields two identical output sequences. Many theorems in automata theory deal with the problem of distinguishing between inequivalent automata. Thus, Moore and Ginsburg have investigated the question of the length of experiments required to distinguish between two inequivalent automata. (U)

If the term information identifies the input and the output of an automaton, then an automaton may be considered as a device for transforming

information. In a deterministic automaton the input information must in a sense contain the output information, since the output information is determined by the nature of the input information. The question then may be posed in reverse: For what automata does the output information contain the input information? The methods for handling such problems are suggested in a paper by Huffman. (U)

Of special interest in connection with adaptive systems are probabilistic automata or machines. Unfortunately, this branch of the theory has attracted the least attention. Most of the research has been limited to the problem of proving that some input-output transformations are possible only if probabilistic machines are used. (U)

In his work de Leeuw proved that probabilistic machines are in a sense capable of no more than deterministic machines. De Leeuw formulated the problem in terms of the enumeration of a set of symbols printed by the output of a machine. For every symbol s that the output can logically print there will be a degree of probability r that s will eventually be printed by a given probabilistic machine. If for some probabilistic machine S is the set of all symbols that is printed with at least a probability p , then S is p-enumerable. The precise formulation of the question then is: Are there any sets that are p-enumerable but not 1-enumerable? The answer is negative. (U)

G. F. Rose has developed a probabilistic interpretation of deterministic machines. Consider a machine that receives n , ordered numbers $x_1 \dots x_n$ as its inputs and prints some other n ordered numbers

$y_1 \dots y_n$ as its outputs. Rose defines a degree of probability associated with a given computational process as follows: If for every n -tuple of x 's there are at least m of the x 's such that $f(x_i) = y_i$, then the machine computes f with a probability m/n . The justification of this definition lies in the fact that, in general, there may be no way of knowing for which m of the n x 's the function has been correctly computed. (U)

Some of the fundamental work in the behavioral aspect of the theory of automata has been done by Kleene. Kleene approaches automata theory by representing sets of input sequences. Consider, for example, the class of all automata with one binary input and one binary output each with values of 1 or 0. For every finite input sequence of 0's and 1's, a given member of the class reacts with an output 1 or an output 0. Thus every automaton breaks down the set of all finite sequences into two disjoint sets--those to which it reacts with an output 1 and those to which it reacts with an output 0. An automaton represents that set of input sequences to which it reacts with 1. The problem is to find sets of sequences of 0's and 1's that are representable by finite automata. The solution of this problem hinges upon the concept of a regular set of sequences or regular events in Kleene's terminology. The concept of a regular set is defined recursively. Without adding the detailed technicalities of exposition, it may be stated that the recursive construction of regular sets proceeds on the basis of operations of concatenation and iteration of sets of sequences. Kleene is able to prove that only the class of all regular sets can be represented by finite automata. (U)

Much interest has been generated by the questions of effective computability. The notion of effective computability goes back to the work of Post and Turing and antedates modern automata theory. A function is effectively calculable if some mechanical means or set of rules exists for calculating it. The quest for an effectively calculable function is often described as a quest for an algorithm in many mathematical theories. Indeed, the concept of an algorithm is more general than the concept of a calculable function; algorithms for constructing a matrix or a graph with a certain property can be describe. However, all algorithms can be expressed as effectively computable functions. The technique of expressing an algorithm as an effectively computable function is given by the process of Gödel numbering. This technique is elaborated in the works of Davis and Kleene. (U)

Computability theory is largely concerned with the problems of decidability (in the computational sense). A decision procedure for an infinite set of questions, each having yes or no as an answer, answers any of the questions when the procedure is applied. If a set of questions has no decision procedure, it is undecidable. Typical of the theorems in this area is a theorem asserting that there is no decision procedure for whether two Turing machines compute the same function. A number of theorems of this kind are proved by Wright, Elgot, Büchi, and others. (U)

One of the most important concepts in the theory of computability is that of a Turing machine, which is a mathematical representation of a digital computer. A Turing machine consists of an infinite tape divided

into successive squares and a machine proper that is fixed in size and capable of reading and writing on the tape. The machine proper without the tape is a finite automaton; but the machine as a whole is a growing automaton, since generally the tape may grow without bounds in at least one direction. The machine proper has a finite number of states. At any time it must be in one of these states and it must be scanning a square of the tape. Depending upon its present state and the symbol on the scanned square, the action of the machine is determined. Its action consists of either doing nothing at all, erasing the scanned symbol and writing a new symbol, and/or moving to the right or left. Various other mathematical machines have been invented since the appearance of Turing's work; the most interesting among them is Dr. Wang. However, anything computable on a Turing machine is also computable on a Wang machine, and vice versa. (U)

From point of view of adaptive systems the theory of growing automata is particularly interesting. Intuitively, the connection between growth and learning seems to be a close one. Since the theory of automata is suitable for investigating the limits of a machine capability, the first problem of growing automata is: What can the growing automata do that finite automata cannot do? One way to formulate this problem mathematically is in terms of one binary-input, one binary-output automata. Any such automaton effects a transformation of an input sequence into an output sequence. The question then is:

- (a) What is the class of transformations realized by a set of fixed automata?

(b) What is the class of transformations realized by a set of growing automata?

Answers to these questions were attempted by Aiserman, Nerode, and others. Unfortunately, this work is presently limited to a careful formulation of the problem. (U)

The topic of automatic pattern recognition has also been treated from point of view of the theory of automata. The precise formulation of the problem of pattern recognition is obtained by asking a following question: For a given set of finite sequences of 0's and 1's, is there an automaton that recognizes members of the set? Recognition means that automaton reacts in one way if the sequence is a member of the set and in another distinguishable way if the sequence is not a member of a set. Several classes of automata have been investigated for their suitability in recognition processes. The first is a finite automaton that scans sequences once from left to right. This problem was studied by Ginsberg. An automaton of this kind is capable of recognizing only regular sequences (in a sense that regular was defined in the discussion of Kleene's work). (U)

The second type of finite automaton that was studied is able to go back and forth over the tape, but it is not permitted to mark or erase the tape. Such automata have been investigated by Rabin and Scott. These authors prove that such an automaton is also able to represent regular events. However, the simplest automaton of this kind that is capable of representing a regular set is simpler than the simplest automaton that scans only from the left. Simplicity in this sense is measured in terms of the number of states of an automaton. (U)

Finally, there is a finite automaton that is allowed to go back and forth over the tape and to erase and write other symbols on the tape. In this manner the machine is able to represent sets of sequences that are not regular. Such a machine is called a linear-bounded automaton; its main characteristic is the variability of the amount of memory in its possession. Specifically, the amount of memory is a linear function of the length of the sequence. Robert McNaughton ventures the hypothesis, without proof, that any task of interest to structural linguistics could be performed by a linear-bounded automaton. (U)

The general critique of these theories pertains despite their theoretical interest and fundamental importance. The theory of automata has relatively little interest for a pragmatic designer of intelligent data-processing systems. The basic reason is that the theory of automata deals with the limits of what can be done rather than with constructive questions in the mathematical sense. In other words, the theory of automata aims at existence theorems. With the present state of computer art, questions of more practical interest are those dealing with the problems of how to perform certain types of tasks. Computer design and programming concepts are still far removed from a stage where the problems concerned with the limits of the possible raise issues of paramount importance. (U)

7. Summary and Conclusions

This review has attempted to summarize theoretical concepts and to present examples of the major avenues of approach to the construction of

learning and self-organizing systems. In view of the diversity of efforts on the one hand and newness of the research on the other, no systematic taxonomy of the field is as yet possible. The taxonomy impressed upon the material in this review is, therefore, somewhat arbitrary. The criteria used to evaluate various systems are slanted toward the requirements for correlating facts. (U)

Five major approaches have been distinguished:

(a) Genotypic Approach - The characteristic feature of this type of approach is the attempt to devise structures with brain-like properties. This approach is not particularly suitable for the needs of a Fact Correlation System. (U)

(b) Monotypic Approach - The common characteristic of this approach is interest in behavior rather than structure. Three applications of this approach were reviewed:

- (1) General Problem Solver - This technique is modelled after the methods used by human beings in solving problems. The focal point of this technique is to endow the system with a capacity for analyzing the environment within which it is supposed to operate. The ideas of Newell, Simon, and Shaw are quite pertinent to a Fact Correlation System. However, more emphasis should be placed upon learning by instruction and upon eliciting information from human sources on an as needed basis.
- (2) Pandemonium - Pandemonium, essentially a monotypic counterpart to the genotypic perceptron, is basically a recognition device. Because of its monotypic approach this system is better suited to the problem of recognizing abstract patterns. The system is capable of more complex recognition tasks than the genotypic models. This flexibility is payed for by a reduced capacity for self-organization.

- (3) Experiments in Machine Learning - This technique stresses the generation of novel methods for solving problems. The main interest in such systems derives from their ability to solve problems independently of human knowledge as to how solution is to be obtained, provided that the criteria for knowing when the problem is solved are known. The techniques used at the present time are extremely primitive. Nevertheless, the experiments in random program generation are important and valuable; they should be pursued conjointly with other two monotypic approaches. (U)

(c) Theory of Automata - The theory of automata deals primarily with the body of existence theorems pertaining to idealized computers or mathematical machines. This theory has little to contribute towards the construction of learning or self-organizing systems and is of no particular interest to the development of a Fact Correlation System. (U)

The three monotypic systems complement each other in many ways. It would be instructive to list these complementary features side by side. Whereas the GPS stresses differences between objects, Pandemonium stresses the degree of similarity. The GPS is concerned with the problem of recognition. Whereas both the GPS and Pandemonium explore the problem space prior to solution, the random program generators approach the solution space directly. In view of this complementarity it is expected that the learning feature of a Fact Correlation System might incorporate elements of all three approaches. (U)

D. Research Problems

Generally, there has been little research activity in the problem areas directly associated with the techniques necessary to perform the

functions of the adaptive correlation task. This section discusses the scope and magnitude of the major problem areas in relation to the functional requirements of a system for correlating facts. The following discussion also outlines the research tasks necessary to resolve these problems and develop the essential processing techniques. (U)

1. Ordering and Storage of Data

One of the problems in devising a system for correlating facts automatically is to determine the structures in which data or information is ordered. These structures will be specified in some degree by the functional requirements, operations, and processes of adaptive correlation. It may be necessary to order data in several different structures, each adapted to particular operations or to particular storage requirements. (U)

The structures in which data are ordered may be essential to the information content of the data; in this sense, the information content is a function of the form of its presentation. The information contained in natural language sentences is partially a function of the order of the words in the sentence. The same information may also be conveyed by another order--for example, the tree structure resulting from syntactic analysis in terms of dependency relations. This tree structure is an information-preserving transformation of the original sentence structure. Similarly, mathematical transformations may be described by either equations or matrices. Operations upon data may also be defined in terms of an ordered structure of data as a transformation or extension of a specific structure or as a re-allocation of positions within a structure. Operations

defined in this manner lend themselves readily to computer processing. (U)

The input to the correlation stage is the product of the language transformation stage, a transformation of natural language data. The form of this data after language transformation is determined by both the transformation process and the requirements of the initial operations of the correlation stage. Since the information content of language is partially a function of syntactic and semantic structures, it may be necessary or desirable to retain some of the structure of the original input, either explicitly or implicitly, in the transformed version. The initial operation of the correlation system may assign such data to a particular data structure for subsequent processing and storage. (U)

The methods and structure or configuration used to store data can also serve as the basis for correlating data. The location of data in a storage structure can be used to indicate relationships among the data; inversely, relationships among stored data can be inferred from their relationships or juxtaposition with a data structure. (U)

There are many ways to order data and structure information. New methods can also be devised and may be necessary. Several general methods should be considered. (U)

(a) Logical Systems - Certain types of sentences in natural language can be readily translated into formulas in a system of formal logic. The logical system employed is determined by several factors, including the capacity of the system to represent the original statements

and the formal characteristics of the system. Such a formalization of data, when possible, may be necessary for inferential operations performed by the system. It should be noted that in a Fact Correlation System it may be essential to retain information about places and times of specific events and reports. Such information is not usually treated in logical systems. The application of logical systems is a well developed art, however, and should be able to meet many of the demands for structuring particular characteristics of data. (U)

(b) Set Theoretical Methods - Certain data are easily translated into the language of set theory. Set theoretical methods are particularly useful in performing logical and mathematical operations upon data that can be described within the bounds of theoretical restrictions. Set theory is readily translated into the terms of formal logic; thus an additional degree of flexibility could be introduced to any system employing both methods. However, set theoretical methods may be either awkward or inappropriate for certain kinds of data and correlation processes. For instance, it is questionable that political and historical data could be readily couched in set theoretical terms. (U)

(c) Topological Configurations - Topological graphs or point sets are useful in presenting certain kinds of information. Trees and lattice structures have proved useful as maps for describing syntactic and semantic structures and for displaying class relationships. Such maps may also prove to be efficient structures for storing and processing information derived from natural language. Topological structures are readily represented by numerical codes, thus lending themselves easily

to computer processing. These structures also offer a basis for treating certain processing operations as topological transformations. (U)

(d) Lists and Matrices - Some types of data are readily stored and processed as lists or matrices. Lists may be defined by a great variety of methods; the techniques for processing lists are now becoming widespread as discussed in Part IV. Matrices, which may be translated into lists, provide powerful structures for the cross-correlation of various kinds of data. (U)

Methods of listing data, including efficient methods of indexing, have been devised for computer processing and storage. (U)

(e) Simplified Language - The possibility should be recognized that a simplified version of the original natural language input to a system may be a most efficient form for storing certain kinds of data. Such a simplified language might be the output of the language transformation process. (U)

(f) Data Structuring - A great deal of research has been performed in this field. Much of this work has been designed to meet the needs of particular systems; therefore, is restricted by the needs and special logical characteristics of that special domain. Some work on a set-theoretical type language for information structuring has recently been suggested (April 1962 Communications of the ACM), but that language seems to be too general to be valuable for a system to correlate facts. (U)

One project that seems to have potential for the future is

the work of Chomsky, Green, and Wolf on Baseball. Essentially, the stored data are the record of all games, together with their scores, played by all baseball teams in one league during specified months of several specified years. These data are stored in tables--for example, under the class month all games together with their scores and the participants are arranged by days. These data can be arranged under other categories so long as the classification is exhaustive and exclusive. Questions are analyzed grammatically so that whatever their grammatical form, key phrases are arranged into specification lists. The lists then order the processing functions for retrieving data. Thus, for a restricted class of questions--namely, those about games played during a particular year--of a certain order of complexity, all questions--and there are a great number of them--are answered. (U)

This program is valuable in several ways. First, it suggests a way to analyze questions. Second, it suggests that in any information processing system that is useful, there must be a data base correlated with the linguistic forms, at least a certain level. The program is, however, essentially a static one. It can only deal with a certain amount of well characterized information. In other words, there are no learning mechanisms built into the data structuring as required for a system for correlating facts. (U)

A fundamental requirement for any data structure used to store data in an adaptive system is that such a structure permits the reordering of data in response to the adaptive functions of the system. Such a structure

should facilitate the re-allocation, addition, modification, and deletion of data with a minimum of processing. (U)

The associational functions of the correlation process also depend upon the data structures used. Certain associations can be derived as a function of the data structure. The assignment of data to a location in a structure immediately establishes the possibility of association to data in related or juxtaposed positions. (U)

There are, obviously, essential research tasks involved in determining the data structures to be used by a system for correlating facts. The structures chosen are directly related to the operations to be performed by the system. The actual execution of such operations can be greatly facilitated by the structures used. (U)

2. Retrieval and Correlation of Data

(a) General - The Fact Correlation System must deal with the functional elements of fact in such a way that a sequence of operations upon these elements or upon concatenations of these elements produces the requested information. What is desired is information explicitly or implicitly contained in the facts that are received by the system. Thus, ultimately, logical implications, generalizations, correlations, and even logical appraisals of the original facts (credulity measures and ordering relations) may be the results of operations upon the data. (U)

The requirements for the performance of operations upon the data parallel, at least in part, those for the structuring of data. These operations must be defined so that data can be recombined into forms that

are not explicitly formed in the original data. Such processing operations must be specified in relation to the structure of data. (U)

These retrieval and correlation processes must gather relevant material from the data structure so that it may be operated upon and used to answer questions. Some of these operations are based upon statistical analyses of the data. Mathematical correlations can be found between two or more facts that might indicate a close relationship that the system might otherwise have missed. (U)

(b) Derivation of Implied Relationships - Other correlative operations on data will be necessary. The system will be expected to derive logical relationships existing among data contained in its memory. In addition to logical inferences (deductions), the system is expected to perform other inferential processes (inductions). Such inductive inferences differ from deductive inferences in two important respects:

- (1) The relationships derived are not necessarily valid.
- (2) Not all the rules of inductive reasoning are explicitly formalized at the present time. (U)

Implied relationship is a generic term for all relationships not explicitly contained in a system. Such relationships must be derived by means of inferential processes; i.e., deductions, inductions, and statistical correlations. The term implied relationship includes relationships derived on the basis of inductive, or non-rigorous, inferential processes. Such relationships are by their nature not as well defined as relationships obtained deductively. The system must, therefore,

be endowed with the capacity of estimating the degree of credibility of such derived relations and the degree of relevance to other information. On the basis of such estimates the system may accept or reject the derived conclusions. (U)

Since the set of implied relationships is not well defined, the system must arbitrarily limit the range of derivable relationships. It cannot be expected that the system will attempt to derive all the implied relationships that lie within a specified range without being requested to do so, either directly or indirectly, in terms of a question. On the other hand, some of the implied relationships might be so important to the functioning of the system that they ought to be derived even without any initiating query. The system must, therefore, possess a set of decision algorithms for determining at which point it must stop its inferential activities. (U)

It is necessary to state the criteria employed to select the relationships the system will derive. While the set of explicit relationships stored in the memory of the system is well defined, the corresponding set of implicit relationships is not. The derived implicit relationships depend not only upon the set of explicit relationships, but also upon the techniques of derivation. These techniques may, in turn, depend upon the nature of the formal or informal inferential methods as well as upon other factors--e.g., the richness of associations--less amenable to precise description. Because of these factors it may be questioned whether the notion of the set of all implicit relationships derivable from the data is

meaningful. From a practical viewpoint, some limitations upon the range of implicit relationships must be imposed. (U)

The criteria for the limitations that are to be imposed upon the system's ability to derive implicit relationships ought to include:

- (1) Only implicit relationships possessing potential utility to the users of the system should be derived.
- (2) The system should not try to derive implicit relationships of so complex a nature that the attempt is likely to end in failure.
- (3) The limitations should be flexible enough to leave room for learning.

The system ought to be able to increase the range of derivable implicit relationships as it obtains more input information or elicits more information about its environment from the user. The criterion for the selection of derivable relationships, which includes all three of these characteristics, is:

The system is only concerned with those implied relationships that can be derived in response to a definite procedure specified by the user.

This principle may be considered as the organizing principle of the system. (U)

There are several points that will clarify the meaning of this principle. In addition, the adoption of this principle has certain implications for the learning processes that will take place in the system. The phrase, ".....in response to a definite procedure specified by the user," does not mean that the user is obliged to supply the directives that could be directly translated into programs--that is a sequence of action resulting in an output consisting of the appropriate implicit

relationships. Neither does it mean that such a specification must be supplied to the system initially. (U)

The principle simply states that the user knows how to go about solving the problem embodied in a query addressed to the system; he knows how to solve the problem in terms of human mental processes. Moreover, the principle does not require the user to state the procedure formally. The concept of knowing how to go about solving problems implies no more than that the user know enough about his own procedures to answer questions about his approach to the problem. (U)

(c) Theory of Questioning - In order to optimize the retrieval ability of a system, the user should question the system within the framework of a theory of questioning. The development of a theory of questioning has occasioned considerable scientific interest within the last decade. In part such an interest is related to problems of retrieving information, for even a cursory examination of questioning indicates that it plays an important role in the retrieval of information. Every pragmatically important question has a correct answer associated with it. Such a correct answer is a statement that provides a person with information--knowledge that he did not possess at the time that he asked the question. The statement may be true or false and still fulfill this criterion. (U)

Given a framework of this kind, the theory of questions requires a development along two parallel lines, the semeiotic and the methodology of questions. (U)

(d) Semeiotic of Questions - Questions are a type of linguistic structure. Composed as they are of words, questions have meaning. Such meaning may be even more complex than the meaning of declarative statements, since questions may also be logical functions of such meanings. (U)

There are two possible ways to investigate the meaning of a question. A question may be correlated with a class of statements, any one of which is a correct answer to the question. In this sense, the question defines the scope of possible answers; it is neither responsive nor meaningful to answer the question "What time is it now?" with the statement "The Parthenon is located in Anthens, Greece." On the other hand, there are questions that do not define the kind of statement that is a correct answer. Consider the question "How many horns does a unicorn have?" "There are no such things as unicorns," is as correct an answer as, "A unicorn has one horn." In other words, a question may pragmatically admit unclarity about the boundaires of a subject. Only procedurally correct questions request information within a framework of concepts and statements accepted as true by both the questioner and the informer. (U)

This realization that a question is related to a given state of knowledge requires further exploration. It is clear that a question is meaningful only if the questioner refers to a set of interrelated concepts either explicitly or implicitly. When a questioner asks "What time is it?" he knows that the answer is a set of numbers that have a certain order-- for example, later than. But it remains a problem whether some concept must be assumed explicitly or implicitly for any question to be meaningful.

It may be that in order for a question to be meaningful, some restriction of its scope must be present. (U)

The meaning of a complex term is not only determined by its relationship to non-linguistic factors, but also by its logical relationship to other terms. The meaning of questions is in part specified by their logical or syntactical relationship to other questions. What is required, then, is a formal logic of questions. Such a logic would rigorously formulate:

- (1) The syntax of a formal language into which questions in natural language are translatable.
- (2) The rules of deduction for such a language.
- (3) The theorems concerning logical relations formulatable in such a system.

It seems that the language in which the logic is formulated may be constructed out of declarative sentences by the use of an undefined logical operator.⁽¹⁾ Logical functions analogous to deduction can then be defined. In any system the correlation between questions and permissible answers must be formally modeled by mapping a question on a set of sentences. Semantically, at least, the range of variables should also be specified for answers that are specifiable for standard types of questions. (U)

In addition to logical deducibility that would be studied by such a calculus, there is another dimension of logical analysis. This area pertains to the relative complexity of questions. It may be, for

(1) See Harrah, D., "The Logic of Questions and Answers," Philosophy of Science and "The Logic of Questions," Proceedings of Congress of Philosophy.

example, that in a certain context a Why question is translatable into a finite set of How questions. In this context, Why questions are more complex than How questions. But there are many types of questions. In addition, there are disjunctive and conjunctive questions as well as general and particular questions. This brief discussion indicates that a logical theory is necessary to consider problems of this kind systematically. (U)

Once a formal analysis of questions has been developed, it will provide insight into the methodology of questions. If the questions that imply other questions are known or are reducible to other questions, then it is easier to develop strategies for sequencing questions so as to obtain maximum information for a minimum set of questions. It is advantageous for any information processing system to allow this condition to be fulfilled. (U)

(e) Strategy of Questioning - Besides purely logical and formal considerations, there is a problem of the strategy or heuristic of interrogation. This problem centers on the problem of efficiency and purposefulness in interrogation. The main objective is to relate the formal characteristics of questioning to intentions that the questioner (man or machine) may have. From the nature of the problem it is evident that, unlike the inquiry into formal properties of questions, this investigation is mainly concerned with sequences of questions. (U)

There are two types of goals that can be associated with the procedure of interrogation. The first is the desire to obtain more factual information. A simple example of this type of interrogation is: "How many

people reside in Rome?" The second type of goal is to obtain a better understanding of a certain area of inquiry. This objective may be related to the interrogator's perception of gaps in the flow of information or to his lack of understanding of the information. Efficient and intelligent questioning depends upon the precision with which the interrogator can pinpoint the kind of information he wants as well as upon his ability to formulate the appropriate sequences of questions. (U)

The objective of the theory of questioning is to establish procedures for an interrogator to discern the intention of his interrogations. The theory is not psychologically oriented. The problem is not to correlate subjective states of mind with the objectives of the questioning process. The theory seeks to associate the properties of sets of information with the rational formulation of interrogative intentions. These intentions are then fulfilled if the sequence of questions is appropriate for its purpose. (U)

3. Learning Processes

(a) General - The ordering of data and the retrieval and correlation of data depend upon initially specified rules for data handling. These rules may not be the only rules for data handling necessary for the proper and efficient operation of a Fact Correlation System. The system must be able to acquire new rules and modify old rules as it continues to process information. The acquisition for learning of rules may be divided into two categories. (U)

One category of learning includes processes based upon

success-failure criteria. In processes of this kind the system attempts to improve its performance without an interchange of complex questions with the user. If the criteria for adequate performance are not not satisfied, the system seeks to improve its performance solely on the basis of its store of data and its own experience. (U)

The second category of learning includes processes based upon the system's attempts to elicit information pertinent to the formation of adequate processing rules from the user. Such learning processes are more complex than those in the first category. In addition to being able to use its own experience, the system must be able (or learn how) to question human beings and to use human guidance. Both kinds of learning involve associational and inferential processes as well as basic adaptive mechanisms. (U)

(b) Adaptive Behavior and the Learning Process - It has been stated that the Fact Correlation System could be characterized as an adaptive or learning system. The learning process can be viewed as a function of a homeostatic system; the system is motivated toward learning certain additional processes that are necessary for the continued existence of the system. (U)

A homeostatic system is a system whose goal is to survive as a functioning entity in an environment that could possibly destroy it. The system brings certain processes into action when changes in the environment cause internal changes in the system. The homeostatic system can be characterized as being potentially in a finite but perhaps large

number of internal states. Some of these states are stable; that is, the system can exist indefinitely as an entity in such a state. The remaining states are unstable. When the system is in an unstable state, processes are initiated that tend to bring the system back into a stable state; for, if the system persists in the unstable state, the system may cease to exist as a properly functioning entity. (U)

A homeostatic system has other goals--the realization of its primary functions. When these functions are being carried out successfully, the system remains in a stable state. If the system is not functioning properly, it enters an unstable state and corrective processes are initiated. These processes attempt to return the system to a stable state by correcting difficulties in system functioning. The system learns to associate certain subsidiary goals with its primary goals. The association may become so strong that some of the subgoals become, in effect, primary goals. (U)

Biological systems, which are examples of homeostatic systems, are motivated toward a certain type of behavior. If motivation is applied strictly to goal-directed activity, it can be applied to more general homeostatic systems. This conclusion is warranted since such systems are motivated to remain in stable states and to correct malfunctions because their basic goals and stabilizing mechanisms direct them into certain kinds of behavior. Thus the system is motivated to adapt to its environment and to fulfill its goals and subgoals. (U)

Adaptive behavior can now be characterized as the behavior

involved in achieving the goals of a homeostatic system--the attempts to survive and function properly in an environment that could possibly disrupt or destroy the system. Adaptive behavior involves the selection of specific processes to attain particular goals. In addition, an evaluation of the effectiveness of the selected adaptive process is implied; on the basis of this evaluation, the parameters of the process may be modified. (U)

Learning, as defined by ordinary language, generally means that an organism gains a knowledge or understanding of a procedure (practical or theoretical) or a fact. The process of learning is such that, judged by some observer, the knowledge or understanding the organism has of the procedure or fact at the end of the process is greater than it had at the beginning. It is important to note that some observer, even if that observer is himself the participant in learning, must be postulated in order to understand the notion of learning operationally. This notion of learning is quasi-behavioristic; it includes an observer and implies an appropriate evaluation process. (U)

This use of the word learning would create problems if applied to a general homeostatic system. It would then be necessary to define knowing and understanding. These terms are entangled in implications of awareness and consciousness, which lead to questions that are not legitimate at this stage of system development. Thus the following use of learning and related terms does not involve the questions of awareness and consciousness; and terms such as knowing and understanding are avoided. These terms are reserved for more complex human behavior. (U)

Learning can then be characterized as a change in adaptive behavior by a homeostatic system in attempting to adapt to new environmental conditions. This process of change implies an evaluation of the behavior either by the system itself or by an external observer. In other words, the effectiveness of the adaptation and the efficiency of the process must be evaluated so that any necessary corrective action can be initiated. Corrective action modifies the adaptive processes to make them more effective. (U)

This discussion is concerned with a general homeostatic system. However, it is directly applicable to an adaptive information retrieval system if the goals are verbal goals and the behavior of the system is verbal behavior. The goals of the Fact Correlation System are to store and process information in order to answer questions. The system adapts as its environment changes--as it receives more information and more questions. If the environmental changes are too drastic--if completely new and unrelated information is received--the system modifies its storage and processing techniques, changes its behavior, and reprocesses the information. If the reprocessing is successful, the system remembers the functions it has performed and has learned how to process this new kind of information. If it is unsuccessful, it again modifies its processing; but it has learned that certain methods cannot be used with certain types of information. (U)

The learning process includes reinforcement. As some adaptive processes prove successful in particular cases, their chance of being

chosen in similar cases increases; if the processes are unsuccessful, their chance of being chosen in similar cases decreases. The processes are then either positively or negatively reinforced. Reinforcement also involves the adjustment of parameters of specific adaptive processes. As certain values are found more effective for particular applications, the tendency for them to be chosen in similar applications increases. (U)

The homeostatic model of learning should be put into mathematical or quasi-mathematical terms such as those used in automata theory. This formalization would clarify the model in several respects:

- (1) The simulation of the homeostatic system by a computer program might be feasible after the mathematical formalization of homeostasis has been completed.
- (2) The empirical realization of the operations of association, classification, and generalization might be investigated. This investigation would include computer simulation of these functions in terms of comparative operations--same as, different from, and similar to.

One point is clear: the mathematical formalization of a homeostatic model would be concerned with delineating those aspects of the concept that are feasible. (U)

(c) Association, Classification, and the Problem of Relevance -

A Fact Correlation System must be able to associate or relate various items in a manner similar to the association processes of human beings. These items may be facts, concepts, relationships, things, or areas of knowledge. To determine the relationship of concepts in memory and to measure the strength of these relationships is perhaps the basic function of the system. Without the ability to associate related concepts, the system has no basis for further processing. As its experience increases,

the system must learn to associate concepts that human beings relate and learn to evaluate the strength of concepts. (U)

The function of association, the linking or grouping of two or more items, is basic to the learning process. Implied in the association process is the prior identification or recognition of an item. This identification or recognition process is based upon similarity and difference criteria that establish whether an item was identical or roughly similar to an item or several items in memory. Once an item is identified, its associated items can also be located in memory. The items in memory may have been linked because of temporal or spatial proximity, explicit instructions from the human user, or rules for associating items given to or developed by the system. (U)

Items may be strongly related in one context but weakly linked in another. Thus the problem of association includes the problem of relevance--the degree of significance one item has in relation to another within a specific contextual framework. One item is more or less relevant to another in a specific context. In addition, only certain aspects of an item may be relevant to another within this context. Thus, in one context a group of items may be relevant; in another context the group of related items may have changed radically. Association and relevance may, therefore, be analyzed in terms of the types of significant relationships that occur. (U)

Association means the grouping or linking of two or more items on the basis of predetermined criteria. Association includes the linking of both normally related and normally unrelated terms. Classification,

structural grouping, operational grouping, and functional grouping are all types of association. A simple kind of association may contain normally unrelated items grouped on the basis of a predetermined list such as: apple, chair; pencil sharpener, submarine. (U)

Classification means the grouping of two or more items based upon a predetermined notion of similarity among their attributes. Thus, classification implies that the items have a number of attributes in common. The items are classified on the basis of similarity and difference standards. For example, members of the animal kingdom are classified into various phyla; phylum members are further classified into different classes. Cat, dog, and alligator are classified together as vertebrates; with further classification they are separated into mammals and reptiles. (U)

Structural relationships involve spatial relationships; i.e., part-whole and part-part relationships. The relationship of room to building or of the planet Earth to the solar system illustrate the part-whole relationship. The relationship of roof to wall or the relationship of Earth to moon illustrate the part-part relationship. The inclusion of one structure within another is indicated by structural relationships. (U)

Classification involves the kind of item; structural analysis involves parts of an item; and operational relationship involves stages of change. Operational relationships deal with temporal relations and involve phases of complex operations; they may be considered structural relationships in the time domain. The relationship of inning to baseball game and of the Normandy Invasion to the Battle of Stalingrad are

operational relationships. (U)

Functional relationships involve structures and operations and their functions and purposes. They relate structures or parts of structures to their function, operations to their function, or structures to operations involving those structures. Some examples are: the relationship of atomic bombs to destruction, intercontinental ballistic missiles to war, or the aging of whiskey to the removal of undesirable substances in whiskey. (U)

(d) Inferential Processes - The system for correlating facts must be able to infer information that is implicitly contained in other information. The system must learn to perform simple inferences that human beings perform in the course of everyday data handling; ultimately, the system should be able to perform inferences too difficult for human beings. In addition, the system must use these inferential processes as part of the set of processes by which it learns ways to order, retrieve, and correlate data. The two general types of inference are induction and deduction. (U)

Deductive inference is the derivation of necessary conclusions from given premises. This derivation follows formal rules of deductive inference. Deduction is a formal procedure independent of the empirical content of statements. Deduction may be subdivided into two categories, immediate inference and extended inference. Immediate inference is the derivation of a conclusion directly from one premise. Extended inference involves deduction from two or more premises. Since deduction is a formal

procedure, deductions from true premises always yield true conclusions. (U)

Inductive inference is generally defined as the derivation of conclusions that are not necessarily true, given true premises, but for which the premises provide less than conclusive evidence. There are a variety of forms of inference that are considered inductive: generalization, inference by analogy, inference from part to whole, inference from group to member, and hypothesis formation. (U)

(e) Interrogation of the User - There are two categories of learning; one simple, the other complex. The complex type of learning depends upon the system's ability to elicit pertinent information from human monitors. The most efficient method of eliciting information is to question the individual in possession of the required information. The system then, must be equipped with the capacity to ask pertinent questions and to avail itself of the responses provided by human monitors. (U)

The art of asking proper questions is taken for granted by human beings. In a machine the rules of questioning must be carefully specified. The process of eliciting information by questioning assumes different forms depending upon the degree of similarity between the interrogator and responder. One of the research tasks, therefore, must deal with the effects that the dissimilarities between mental and data processing processes impose upon interrogations. (U)

The problem areas that require research in the interchange of information between men and machines include:

- (1) The effect of linguistic divergence between human beings and machines upon the structure and strategy of interrogations.
- (2) The dependence of the strategy of interrogation upon judgments of reliability and relevance. The interrogating processes between human beings rely heavily upon the quality of judgment concerning the reliability and relevance of information. The fact that machines are inferior to human beings may impose modifications upon the strategy of questioning.
- (3) The effect of different storage capacities and speeds of data handling upon the efficiency of questioning. The nature of human interrogative processes may be a function of particular human capabilities.

This simple statement of these problems may obscure the difficulty involved in their resolution. Little is known about the question and answer process as it pertains to man; some insight into these processes must be gained before they can be systematically organized and programmed as computer operations. (U)

E. Summary

There are two major aspects to the operation of the adaptive correlation processes. They are the ordering and storage of input data and the retrieval and correlation of stored data. Since not enough is known about complex information processes to specify the formal rules required for these operations, the system must be able to modify techniques for data handling as it processes information. Thus the system must be a learning system. (U)

A Fact Correlation System must organize data into a structure that reflects relationships among data by the nature of its logical organization. This structure should indicate that information is relevant to other information by its format without further processing by the system. It

must be easily accessible by both input and retrieval processing. In addition, it must be modifiable as the system receives new information, modifies its processing, or learns more about the information by correlative procedures (U)

The system must retrieve and correlate relevant material from the data structure and operate on this material with inductive, deductive, and statistical techniques to be able to respond properly to questions. In order to aid the system in answering questions, it may be necessary to phrase and sequence questions in accordance with an optimal questioning logic developed under a theory of questioning. (U)

The learning function of the system includes the associating of related concepts through groupings based upon classification, structural relationships, operational relationships, and functional relationships. It also includes the inferring of relationships, which may be implicit in the data stored by the system, by processes of induction and deduction. The system will learn from its experience by using primitive learning processes; it will also learn by questioning the human user. (U)

F. References

- (1) Aizerman, M. A., "Finite Automata," Automatika i Telemekhanika, Vol. 21; 1960: pp 156-163, 248-254.
- (2) Allpart, F. H., Theories of Perception and the Concept of Structure; John Wiley, New York, 1955.
- (3) Alt, Franz L. (editor), Advances in Computers, Vol. 2; Academic Press, New York, 1961.
- (4) Annett, John, The Role of Knowledge of Results in Learning: A Survey (AD 262-937); U. S. Naval Training Device Center, Port Washington, New York, May 1961: 53 pp.

- (5) Babcock, M. L., Reorganization by Adaptive Automation (AD 233-187); Electrical Engineering Research Laboratory, University of Illinois, Urbana, Illinois, January 1960.
- (6) Bellman, Richard, Dynamic Programming, Intelligent Machines, and Self-Organizing Systems (AD 276-533); The RAND Corporation, Santa Monica, California, June 1962.
- (7) Burks, Arthur W., The Logic of Fixed and Growing Automata; Engineering Research Institute, University of Michigan, Ann Arbor, Michigan, August 1957.
- (8) ----- "Theory of Logical Nets," Proceedings of the IRE; 1953.
- (9) Bremermann, H. J., The Evolution of Intelligence--The Nervous System as a Model of its Environment (AD 201-143); Washington University, Seattle, Washington, July 1958; 81 pp.
- (10) Campaigne, Howard, "Some Experiments in Machine Learning," Proceedings of the Western Joint Computer Conference; 1959.
- (11) Copi, I. M., Elgot, C. C., and Wright, J. B., "Realization of Events by Logical Nets," Journal of the ACM; New York, 1958.
- (12) David, Joseph R., Contributions to Perceptron Theory (AD 243-899); Cornell Aeronautical Laboratory, Buffalo, New York, June 1960.
- (13) Davis, M., Logemann, George, and Loveland, Donald, A Machine Program for Theorem Proving (AD 269-378); New York University, New York, June 1961.
- (14) De Leeuw, K., Moore, E. F., Shannon, C. E., and Shapiro, N., "Computability by Probabilistic Machines," Automata Studies; Princeton University Press, Princeton, New Jersey, 1956.
- (15) Elgot, C. C., and Buchi, J. R., "Decision Problems of Weak Second-Order Arithmetic and Finite Automata," Notices of the American Mathematical Society; 1958: p 834; 1959: p 48.
- (16) Farley, Belmont, G., Self-Organizing Models for Learned Perception (AD 242-332); Lincoln Laboratory, MIT, Lexington, Massachusetts, 1960: 25 pp.
- (17) Farley, Belmont G., and Clark, W. A., Activity in Networks of Neuron-Like Elements (AD 238-760); Lincoln Laboratory, MIT, Lexington, Massachusetts, 1960.
- (18) Friedberg, R. M., "A Learning Machine: Part I," IBM Journal of Research and Development, Vol. 2; IBM, Poughkeepsie, New York, January 1958.

- (19) Gelernter, H. L., and Rochester, N., "Intelligent Behavior in Problem Solving Machines," IBM Journal of Research and Development, Vol. 2; IBM, Poughkeepsie, New York, October 1958.
- (20) Ginsburg, S., "On the Length of the Smallest Uniform Experiment which Determines the Terminal States of a Machine," Journal of the ACM; New York, 1958: pp 266-280.
- (21) ----- "A Technique for the Reduction of a Given Machine to a Minimal State Machine," IRE Transactions; 1959: pp 346-355.
- (22) Gorn, S., Common Programming Language Task (AD 248-110); Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia, June 1960.
- (23) Greene, Peter, In Search of the Functional Units of Perception: An Outline (AD 268-009); Committee of Mathematical Biology, University of Chicago, Chicago, Illinois, June 1961.
- (24) Hebb, H. O., Organization of Behavior; John Wiley and Sons, New York, 1949.
- (25) Huffman, D. A., "Canonical Forms for Information Loss-less Finite State Automata," Nuovo-Ciurento (Supplement); 1959: pp 397-405.
- (26) Kleene, S. C., Introduction to Metamathematics; D. Van Nostrand and Co., Princeton, New Jersey, 1952.
- (27) ----- "Representation of Events in Nerve Nets and Finite Automata," Automata Studies; Princeton University Press, Princeton, New Jersey, 1956.
- (28) McNaughton, R., and Yamada, H., "Regular Expressions and State Graphs for Automata," IRE Transactions on Electronic Computers; New York, 1960: pp 39-57.
- (29) Minsky, M., "Steps Toward Artificial Intelligence," Proceedings of the IRE; New York, January 1961.
- (30) Moore, E. F., "Gedanken Experiments on Sequential Machines," Automata Studies; Princeton University Press, Princeton, New Jersey, 1956.
- (31) Murago, Saburo, Preliminary Study of the Probabilistic Behavior of a Digital Network with Majority Decision Elements; RADC, Rome, New York, August 1960.

- (32) Nerode, A., "Linear Automaton Transformations," Proceedings of the American Mathematical Society; 1958: pp 541-544.
- (33) Newell, A., Shaw, J. C., and Simon, A. H., Report on General Problem Solving Program, P-1584; The RAND Corporation, Santa Monica, California, January 1959.
- (34) ----- The Simulation of Human Thought (AD 235-801); The RAND Corporation, Santa Monica, California; December 1960.
- (35) Quillian, M., The Elements of Human Meaning: A Design for an Understanding Machine; Social Science Department, Presbyterian St. Luke's Hospital, Chicago, Illinois, 1961.
- (36) Rabin, M. O., and Scott, D., "Finite Automata and their Decision Problems," IBM Journal of Research and Development, Vol. 3; IBM, Poughkeepsie, New York, 1959: pp 114-125.
- (37) Rosenblatt, Frank, Neurodynamics; Spartan Press, Washington, D. C., 1962.
- (38) ----- The Perceptron: A Theory of Statistical Separability in Cognitive Systems (AD 204-706); Cornell Aeronautical Laboratory, Buffalo, New York, January 1958: 268 pp.
- (39) ----- On the Convergence of Re-enforcement Procedures in Simple Perceptrons (AD 234-874); Cornell Aeronautical Laboratory, Buffalo, New York, February 1960: 56 pp.
- (40) Ross, John, Human Meaning: A Partial Model and Its Implications (AD 259-526); Princeton University, Princeton, New Jersey, May 1961.
- (41) Selfridge, Oliver G., Pandemonium: A Paradigm for Learning (AD 236-251); Lincoln Laboratory, MIT, Lexington, Massachusetts, April 1960: 12 pp.
- (42) Shannon, C. E., and McCarthy, J., Automata Studies; Princeton University Press, Princeton, New Jersey, 1956.
- (43) Shannon, C. E., "Game Playing Machines," Journal of the Franklin Institute, Vol. 206; Franklin Institute, Philadelphia, 1959: pp 447-453.
- (44) Solomonoff, R. J., "An Inductive Inference Machine," IRE National Convention Record; New York, 1957.
- (45) Turing, A. M., "Can a Machine Think?" The World of Mathematics (J. R. Newman, Ed.) Vol. 4; Simon and Shuster, New York, 1960.

- (46) ----- "On Computable Numbers with an Application to the
Entscheidung Problem," Proceedings of the London Mathematical
Society; London, 1936: pp 230-265.
- (47) von Neumann, John, Computers and the Brain; Yale University
Press, New Haven, Connecticut, 1958.
- (48) Wang, H., "Towards Mechanical Mathematics," IBM Journal of
Research and Development Vol. 4; IBM, Poughkeepsie,
New York, January 1960.
- (49) Yovits, M. C., and Camerson, S., Self-Organizing Systems;
Pergamon Press, New York, 1960.
- (50) Yovits, M. C., Jacobi, G. T., and Goldstein, G. D., Self-
Organizing Systems - 1962; Spartan Press, Washington,
D. C., 1962.

IV. COMPUTER PROGRAMMING

A. General

A computer must be considered in terms of both the actual physical equipment and the computer programs. An unprogrammed computer is merely a complicated electronic network, as useless as a random stack of electron tubes, until its behavior is organized. A program changes the computer into a highly specialized calculating tool. A program is a systematic set of instructions that defines the operations of a data processing system under every conceivable combination of circumstances within a particular frame of reference. (U)

A serious drawback in the application of modern data processing systems has been the cost and time consumed in programming the computers. Since computers had to be instructed step by step on the most basic level, each programmer found himself repeating in general many processes that another programmer using the same machine had to do for himself, namely, getting information into the machine, operating on it, and printing the results. In spite of the many similarities in programs, there was virtually no possibility of one programmer's making use of another's work. In the past several years, however, a variety of programmatic tools have been developed for the programmers, users, and operators of large-scale digital computers. historically, one of the first of these was the symbolic assembly program. When computers were first developed, a difficulty existed in translating problems into the limited artificial language of the machine. The development of automatic programming systems--such as assemblers, translators, and compilers--helped to resolve this problem. (U)

Regional addressing and symbolic addressing allowed a programmer to write a symbol to refer to an address but to delay the assignment of absolute addresses until all requirements for memory space were available. In regional addressing, addresses are assigned according to a scheme of origin cells that control the relationship between regional symbols and absolute addresses; addresses are assigned by the programmer after the entire routine has been written. In symbolic addressing, addresses are assigned according to an arbitrary dictionary created after the routine is finished. Both addressing concepts were programming conventions that in no way changed any of the engineering characteristics of the computer. (U)

An assembly program performed the transition from regional and symbolic addresses to absolute addresses. The first assembly programs performed nothing more than a one-to-one mapping of symbolic instructions to machine instructions. Later assemblers had a variety of facilities, including pseudo-instructions, a symbolic representation of information composed of a group of characters having the same general form as a computer instruction, but never executed by the computer as an actual instruction. Additional improvements include user-defined macro operations, a type of pseudo-instruction that specifies a certain sequence of machine words, and the ability to modify the assembly program itself. Input-output conversion and formatting packages were also developed, as well as subroutine libraries accessible to the assembly programs. (U)

The advent of algebraic and data processing languages and their associated compilers represented a major advance in programming. These languages provided the user of a computer with a means of expressing

himself in easy-to-understand, common-sense terms, thus relieving him of the need to understand many of the details of the computer's operation. Programming methods that previously differed from computer to computer now became part of an automatic system that could be provided for many different computers. The programmer was able to communicate with the machine in a simplified notation that more closely resembled the language of the problem than that of the machine. These programs assigned the computer the task of assigning memory locations and of translating mnemonic codes into machine-language programs. Monitor programs were devised to tie together assembly programs, input-output packages, subroutine libraries, and compilers and to process a series of jobs automatically. (U)

Most of the languages and processors developed thus far have been oriented toward either data processing or scientific computer applications. Most of these aids have been designed to service individual programmers and do not include automatic facilities for communication between programmers and programs within a computer. (U)

At present there is a large class of information-processing problems whose needs transcend the limits of existing programmatic tools. This class of problems includes: modeling problems such as input-output analysis; transportation and scheduling problems; mathematical modeling; control and communication systems; information retrieval and library systems; language translation; and automatic programming systems. These problems are characterized by the following properties: large and/or indefinite amounts of data; a large variety of data elements and data structures; complex, logical-type manipulation of data elements; imperfectly defined

solution schemes requiring frequent modifications to programs; and large modular programs requiring frequent reorganization of their components. (U)

A compiler operating within the framework of the ordinary executive or monitor systems would be insufficient for the construction of these types of information processing programs. Such compilers require a much broader environment--one that includes, in addition to the monitoring functions, facilities for filing programs introduced by the compiler, facilities for filing data to be operated on by these programs, and facilities for permitting any programmer to reference and incorporate any other programs or data existing in the file. (U)

The programming system needed for handling complex problems such as those involved in analyzing intelligence data requires, in addition to an effective compiler, an executive monitoring and control function that maintains a constant surveillance over all operations, initiates input-output activity, and maintains a smooth flow of information for maximum system efficiency. (U)

B. Executive Monitoring and Control

A programming system with an executive routine and control programs is required in order to perform the many tasks in a Fact Correlation System. The executive program is the agent for coordinating the interrelated functions of personnel, equipment, and programs within the over-all system. To achieve operational flexibility the programming system must be organized to perform specific tasks in response to either internal or external demands for processing functions. Since the sequence of demands is unordered, the

executive program requires a facility for the dynamic allocation of storage media. Dynamic storage allocation implies that the computation process is interrupted as necessary in order to assign codes to storage elements, set parameters, or load new codes, thus permitting memory to be assigned in small sections and working storage to be assigned as a function of computational requirements. Problems in the area of information retrieval and fact correlation readily lend themselves to dynamic allocation, since the working storage requirements and the routines to be executed are determined during the course of computation. Dynamic storage allocation also plays an important part in multiprogramming, since the set of programs involved has unpredictable running times and requires varying amounts of storage. (U)

Storage allocation, the linkage between data and its location in memory, consists of two functions--assignment and communication. The requirement that storage allocation be performed dynamically during the execution of programs is based upon the characteristics of the programming system needed for the fact correlation problem. These characteristics are:

- (a) The programs are composed of a number of subprograms and the precise composition, organization, and sequencing of subprograms may be unknown before execution time. It must be possible to execute programs on a demand basis.
- (b) The programs may reference a large variety of data elements as well as a large variety of data structures.
- (c) The files of programs and data must be organized and the data stored and retrieved automatically by the system at execution time.
- (d) The programs may be self-organizing.
- (e) The programs will be interrupted from time to time in order to interrogate the environment or respond to a request from the environment.

In a complex programming system of this nature, several tasks may be executed simultaneously. To achieve simultaneity requires an executive program that determines priority and assigns control to tasks that are to be performed concurrently. The executive program is cognizant of all active and inactive or interrupted tasks and of the priority and precedence of each task. It is also aware of the location of all data and programs available to the system. The executive program acquires these capabilities through the several smaller programs that are a part of it, such as an interrupt program, an input-output control program, a priority interpretation program, and a dynamic allocation program. (U)

The control programs are subordinate to the over-all executive program and their purpose is to coordinate, monitor, evaluate, and define the logical functions being performed by the processing programs. The control programs can be considered in a master-slave context--the processing programs that do the actual work such as data movement, scaling, calculation, comparisons, or conversions are slaves to the master control program. Thus the control programs become the flexible heart of the programming system and allow for countless arrangements of program modules to serve required functions. (U)

The operation of the system is basically cyclical, a cycle being defined as that time period or that sequence of program operations necessary to accomplish all the system functions applicable at that particular time. A cycle can then be considered as a series of computer operations that are non-identical in successive iterations, flexible in duration, and variable in function; but ultimately the cycle returns the program system

to the same basic starting point. A large unit of cycled programs may be comprised of various and varying smaller cyclic program units. This concept admits the validity of changing system objectives (sometimes planned, sometimes dynamically dictated) while maintaining a continuous operation. Transitions present no problem because the system is designed to undergo dynamic change. (U)

Figure 4-1 shows a flow chart of a general organization for a programming system for the Fact Correlation System. Input data is received by the input-output control program, which notifies the executive of the arrival of new input. The dynamic allocation program assigns storage locations as needed, and a map of storage location assignments is maintained so that this information is always available to the executive. The executive then transfers control to the proper control program to process the input data. If an input error is detected, control is transferred to the error routine. If a query is received at any time during processing operations, the interrupt program is activated. If the control program requires additional information from the environment, the input-output control is activated and the additional information requested. (U)

Conflicting demands by the control programs for operational processing are resolved by the priority interpretation program of the executive. The purpose of the executive is to maintain the continuous and efficient operation of the system. An interrupt does not necessarily mean that all activity ceases while the query or request for information is attended to by operational personnel; other processing tasks continue. The priority interpretation program continually checks the task list and determines which

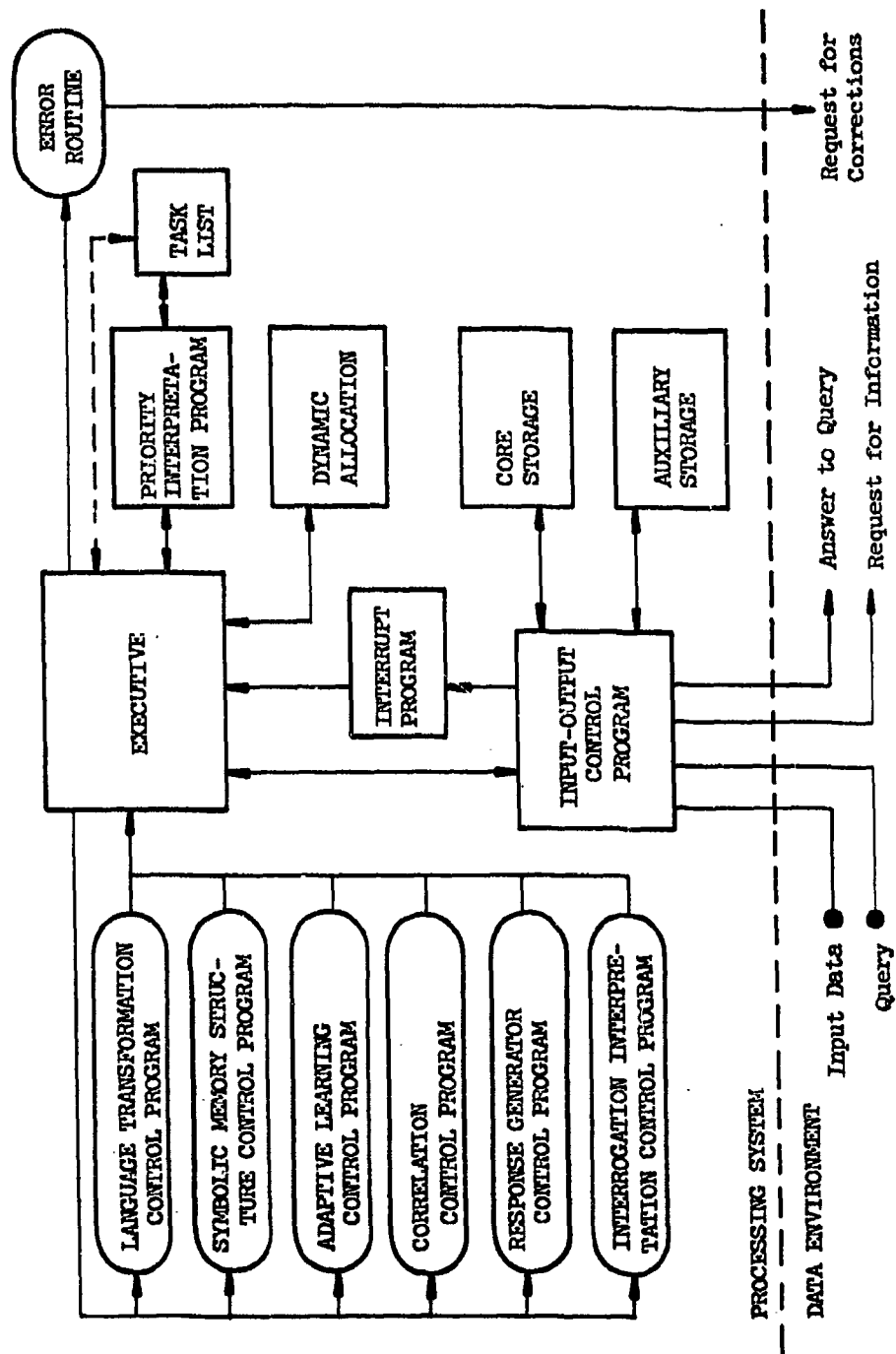


FIGURE 4-1. Programming System for Fact Correlation

tasks are waiting to be performed. If at any point in the processing of a program an error is detected, the error routine is immediately activated; regular processing of that program continues after the error has been processed. (U)

In the early stages of a Fact Correlation System, there will be a great deal of interplay between the processing function and its environment--the remaining functions of the system. However, as the techniques of analyzing and correlating information are improved, this requirement is expected to decrease proportionately until the ultimate goal of a fully automated system is attained. Full automation does not preclude continuous interaction between man and the computer--there will always be something more to learn. (U)

C. Parallel Programming

With parallel programming techniques it is possible to overlap various tasks being performed at any given time. Time-sharing between programs shares the use of the central computer, core storage, and peripheral devices. While one program is delayed because it is waiting for the operation of a peripheral device, another program in the machine can be processed. Operations that would otherwise have to be performed off-line can be performed by the central computer even though the computer is doing other work. Full lock-outs built into the equipment make it impossible for one program to refer unintentionally to a part of the machine or a peripheral device allotted to another program. (U)

The program being executed by a computer, insofar as the computer

itself is concerned, consists entirely of the instruction or instructions being executed at any given moment. The computer is not concerned with whether or not the current instruction is associated with the preceding or succeeding instructions. In this sense any time-sharing operations can be called multiple programming. However, the common use of the term implies that the system has features that simplify the task of the programmer in attempting to maximize the amount of time-sharing possible. Multiple or parallel programming, therefore, is a method of processing more data in a given amount of time by a given amount of equipment through time-sharing. (U)

In general, the problem may be stated: Given a multiple facility with autonomous units having time-sharing and space-sharing capabilities, execute a program or a set of programs so as to optimize an objective function. This function may be minimum time, minimum number of instructions, maximum utilization of facilities, or any combination of these variables. In general, a sequential program is considered and blocks of instructions are allocated to the autonomous units, according to their total or partial availability, under the control of a central processor. The sequencing procedure is concerned with observing priority requirements and minimizing queuing problems, while facilitating communication between autonomous units and avoiding destructive interaction between programs or program segments. This procedure is an essential part of an executive system that would be required for a Fact Correlation System. (U)

Parallel programming superimposes a new level of logical complexity on both the system designer and the programmer. The factors involved are:

- (a) Additional control logic to facilitate switching between programs.
- (b) A program to monitor the execution of the set of programs present in the computer at any given time.
- (c) New programming systems to provide for the incorporation of additional program control data into the system.

The system must be based upon the concurrent and mutually coordinated design of equipment, the executive program, and the programming system.

The ultimate system should have the following characteristics:

- (a) Programs entered or stored in the system for execution are independent of any absolute machine addresses.
- (b) Program priorities and precedence to permit the choice of an appropriate program with minimum delay when alternatives are possible.
- (c) An executive system to keep track of the status of each program being executed.
- (d) Any control or memory register or other system element not currently being used by a given instruction must be available for use, if required, by an instruction from another program being executed concurrently.
- (e) A set of interrogatory and imperative interrupts must be included in the equipment design, and the machine logic should be able to recognize these demands for attention.
- (f) A direct communication link must exist between any two system elements that may need to transfer information between themselves.
- (g) Sufficient micro-programmability must be incorporated to exact a minimum of redundancy and a maximum of information per instruction and to permit one instruction to use the idle portion of the duty cycle of any other instruction.
- (h) A minimum of additional execution time should be incurred by the executive and control programs to run more than one program.
- (i) Data reordering and intra-system data transfer should be at a minimum.

(j) Programming should be kept simple.

To a large extent these characteristics entail the development of new techniques. Research per se is not essential, although a development program would require creative skill and ability in the organization of computer programs. (U)

Little attention has been given to the problem of restructuring a program to make it more adaptable for the assignment of sets of instructions to various units. Rather than following a serial procedure, in which the next consecutive instruction or set of instructions is assigned to an appropriate and available unit, the aim should be to restructure and reorganize a given program to take full advantage of the potential afforded by a multiple facility. A procedure should be developed by which the computer itself can automatically restructure a program and devise a sequence in accordance with given information. At the present time, models and analytic solutions do not exist for complex sequencing problems, although solutions for simple cases have been developed. Various heuristic methods and Monte Carlo and linear programming techniques have been employed to arrive at optimal or near optimal solutions, but the computations required are lengthy and the cost of obtaining a solution may exceed the savings. (U)

D. Assemblers and Compilers

The preparation of a program for an electronic computer requires the following steps:

- (a) Formulation of the problem and formulation of the way in which it is to be solved.
- (b) Allocation of storage for data and programs.

(j) Programming should be kept simple.

To a large extent these characteristics entail the development of new techniques. Research per se is not essential, although a development program would require creative skill and ability in the organization of computer programs. (U)

Little attention has been given to the problem of restructuring a program to make it more adaptable for the assignment of sets of instructions to various units. Rather than following a serial procedure, in which the next consecutive instruction or set of instructions is assigned to an appropriate and available unit, the aim should be to restructure and reorganize a given program to take full advantage of the potential afforded by a multiple facility. A procedure should be developed by which the computer itself can automatically restructure a program and devise a sequence in accordance with given information. At the present time, models and analytic solutions do not exist for complex sequencing problems, although solutions for simple cases have been developed. Various heuristic methods and Monte Carlo and linear programming techniques have been employed to arrive at optimal or near optimal solutions, but the computations required are lengthy and the cost of obtaining a solution may exceed the savings. (U)

D. Assemblers and Compilers

The preparation of a program for an electronic computer requires the following steps:

- (a) Formulation of the problem and formulation of the way in which it is to be solved.
- (b) Allocation of storage for data and programs.

(c) Selection and incorporation of library routines.

(d) Machine coding of programs; i.e., expressing the programs in the language of the particular computer on which the problem is to be solved.

The first of these steps, adapting the problem for solution on a computer, is much the same whether the programs are written for a particular computer or for any computer in general. The remaining steps are more dependent upon the nature of a particular computer. The way to solve a computational problem must be stated precisely. The question then is: What is the best method of stating the way in which a problem shall be processed on a computer? Another way of phrasing this question is: What language should be used to state the problem? This question becomes a problem of dividing labor between man and machine. There are many answers to this problem, depending upon such factors as the computer (large or small), the class of problems, and even the taste and philosophy of the personnel involved. All these factors must be taken into account in designing a system for fact correlation. (U)

1. Symbolic Language Versus Machine Language

A good, high-level programming language with a good, efficient compiler is almost always more efficient than machine-language programming. A machine-language program is a sequence of binary numbers that instruct a computer to perform a particular task. A symbolic language program is a representation of a machine-language program in a form that is more convenient to human beings. The symbolic language is sufficiently like machine language to permit the programmer to use all the facilities of the computer that would be available to him if he were coding directly in a

machine language. If the programmer must use the machine language directly in preparing his program, he must exercise scrupulous care to ensure that he has not committed any of many possible programming errors, such as computing with values obtained from an incorrect location in memory or setting up the wrong qualifications to enter or leave a repetitive portion of the program. When an automatic programming system is used, these housekeeping responsibilities are assumed by the computer. (U)

2. Automatic Programming Systems

Various types of automatic programming systems are available for almost all present-day computer systems. These programming aids enable the programmer to write his coding in a convenient notation that is not necessarily related to the language of the computer, thus simplifying the programming task and minimizing errors. In automatic programming systems the translation from pseudo-code, the language in which the programmer writes, to computer code, the language used internally by the computer, is performed by the computer under the direction of an assembler or compiler. (U)

As sophistication in programming has increased, computers have been given the capability to understand algebraic notation. The relative complexity of algebra requires that this function be a programming innovation rather than an equipment design, at least for the present. A translation program accepts the language that describes some computation. These translation programs, which are functionally similar to assembly programs but more complex, are called compilers or translators. An assembler transforms symbolic-language programs into machine-language programs, as does a compiler; it differs from a compiler in that the symbolic languages used with

the assembler are closely related to the language of the computer. The source language used with a compiler resembles the technical language in which problems are stated by human beings. A compiler is a master routine that transforms a program expressed in some form of artificial code into computer code before the computation is started. When transforming, the compiler may be required to perform the following functions: convert, select, generate, allocate, assemble, and record. (U)

Compilers have several advantages over assemblers. The language of the compiler is easier to learn and use; programming is faster; there is less chance of programming errors; and most errors can be detected by the compiler. The disadvantage of compilers, however, is that they are generally so organized as to use the memory capacity of the computer inefficiently. Compilers also tend to be innately rigid, a condition that restricts the type of problem that can be solved and the method of solution. (U)

3. Use of Subroutines

One method used by automatic programming systems to conserve programming effort is to store coded subroutines in a library file. Almost from the beginning of programming, subroutines have been used to extend the effective vocabulary of operations available to a programmer, conserve memory space, avoid duplicate coding of repetitive routines, serve as a standard means of communicating among programmers, and justify the effort of developing more efficient routines. In addition, routines have been written more and more in segmented, subroutine form in order to add flexibility in altering programs and to parcel out a large job among several programmers. (U)

As constructed, a computer has a repertoire of instructions, each one initiating a single step in the total computation. However, it is more desirable to work with larger units of processing and to preserve or conserve the effort spent in constructing such units from individual instructions. The subroutine has been the standard solution to this problem. In connection with compilers and assemblers, libraries of subroutines can be called into any program automatically. From a linguistic viewpoint, a subroutine is an abbreviated device whereby a name can be assigned to a set of instructions and used in its place. (U)

Systems of subroutines can be developed and called into action by means of a pseudo-code. A program called an interpreter is used to decode these pseudo-instructions and execute their corresponding subroutines. An interpretive routine carries out problem solutions by decoding instructions written in a pseudo-code, selecting and executing an appropriate subroutine to perform the functions, and proceeding to the next pseudo-instruction. An interpretive routine, unlike a compiler, performs its functions as it decodes the pseudo-code. An interpretive routine is essentially a closed subroutine that operates successively on an indefinitely long sequence of program parameters, the pseudo-instructions and operands. A disadvantage of such routines is that interpretation is performed at a great cost in speed, since each pseudo-instruction requires several machine instructions for interpretation. IPL-V (Information Processing Language) is an example of a pseudo-code and is described further in Section F.4, List Processing Languages. (U)

E. Programming Languages

Programming languages are well established as a tool in the use of electronic digital computers. Among the programming languages considered to be most useful are those that may be described as machine independent and problem oriented. The first phrase implies that the programming language does not vary with the computer; the second, implies that the only effort required of a human programmer is to state the problem in the programming language. These programming languages are called problem oriented languages or POL's. (U)

Among the many possible programming languages, there is a class of languages with the following properties:

- (a) They do not contain elements that refer to particular components of computers; e.g., index registers or accumulators. They describe a computing process in terms of abstract operations--addition, multiplication, comparison--that have to be executed with units of information such as numbers or bits.
- (b) They are sufficiently precise and restricted to be mechanically translatable into machine programs.
- (c) They are rich enough to describe a set of problems and their solutions and are reasonably close to the user's ordinary language of description and solution.

In this class are such languages as FORTRAN, MATH-MATIC, ALGOL, and COBOL. (U)

1. Advantages of Programming Languages

Certain benefits should be derived from the use of high-level programming languages. In program planning some languages, such as those incorporating decision tables, can be used to replace flow charts. The planner can compile the final version of his plan as part of the program. Programming languages can be used for communicating between programmers

and those having the work programmed or among different programmers working on the same large job. This communication is often accomplished by having the data and common terms available to all the programmers. The speed and accuracy of coding can be increased through programming languages that relieve programmers of the burden of machine-language coding. Faster and more accurate program checkout can be achieved because fewer errors are written initially and the error detecting procedures are more comprehensive. Programming languages also provide a substantially non-machine oriented statement of the problem solution, which can then be easily converted to another machine. (U)

If a programming language is to provide these benefits, it should satisfy certain requirements. It should not be difficult to learn; it should not be overly oriented toward a particular computer; and it should include such features as list processing, variable length logical records on secondary storage, program storage overlays, character and bit manipulation, and facilities to set up subroutines or to organize programs into subroutines. In reviewing the existing POL's for possible use in a Fact Correlation System, these requirements should pertain as evaluation criteria. Of course, the possibility exists that as new or unforeseen problems arise, techniques other than those satisfying these requirements may be needed. (U)

2. FORTTRAN, ALGOL, and COBOL

The FORTRAN (FORmula TRANslating) system is an automatic coding system that accepts a source program written in the FORTRAN language and produces a machine-language object program ready to be run on a computer.

The object programs produced by FORTRAN are generally as efficient as those written by an experienced programmer. The purpose of the FORTRAN language is to describe computational processes in mathematical terms. The basic concepts used for describing calculating rules are the well-known arithmetic expressions that contain as constituents, numbers, variables, and functions. By applying the rules of arithmetic composition, such expressions compound self-contained units of the language into explicit formulae, called assignment statements. ALGOL, a similar language oriented to mathematical expressions, is a powerful tool for the communication of algorithms and effective computing techniques. (U)

COBOL (Common Business Oriented Language), a problem oriented language that is a subset of normal English, and is suitable for expressing the solution to data processing problems. The syntax of COBOL is close to English syntax. The language structure falls into three categories: procedure statements tell how to manipulate the data; data descriptions name and explain the elements of information; environment descriptions provide the link between the data involved and the physical hardware media. Since differences in computers make it impossible to achieve a completely independent machine language, each category contains a decreasing amount of compatibility across different computers, the environment category being completely computer dependent. (U)

3. Comparison of Programming Languages

FORTRAN and ALGOL are generally described as being machine independent and problem oriented. FORTRAN is a machine-oriented macro-language; it is actually not machine independent, although it is implemented on a variety of

computers. ALGOL is not essentially a machine language, but is rather a problem statement language whose structure is particularly well-suited for translation by existing computers. A comparison of FORTRAN and ALGOL leads to the conclusion that ALGOL is the more powerful and general language; it allows the user to write more comprehensive problems in the source language. ALGOL is also freer of restrictions and exceptions and offers better annotation capabilities. FORTRAN persists, however, because it has been operational for a long time and is a tried and tested language that has been used successfully in many applications. (U)

The current COBOL and ALGOL families of languages are elaborate and sophisticated. Their generality is accompanied by a profusion of rules and regulations for writing programs; hence, these languages are more difficult to learn to use effectively than simpler languages. ALGOL, with its many special symbols, is difficult to learn to read; COBOL procedure statements are easy to read, but the data description statements are difficult to understand. The simplified English syntax of COBOL makes it rather verbose; many more characters must be written in the program than would actually be necessary to describe the function. The programmer has been relieved of machine details only to be faced with the problem of spelling words correctly and learning the exact meaning of some 256 key words in a new vocabulary. A word often has several meanings depending upon its context. This language is not English, but a pseudo-English with its own specific grammar. (U)

4. Recursive Subroutines and ALGOL

The task of an ALGOL translator program is to transform ALGOL

statements into a machine-language program that will perform the intent of the programmer's description of the problem. Many of the constituents of ALGOL are defined recursively. For example, in order to process a statement, there must be a statement subroutine; and in order to process an arithmetic expression, an arithmetic expression subroutine. Since a constituent may be another arithmetic expression, the simple arithmetic expression subroutine must be able to call upon itself. This inversion must be performed without losing information on the former level; a subroutine that can call upon itself in this way is a recursive subroutine. The ALGOL translator must therefore consist of a set of recursive subroutines, each of which can call upon other subroutines as well as itself. (U)

F. List Processing

There is a class of problems that are difficult to program in either classical machine languages or the higher level languages. One characteristic of these problems is that their solution requires a prior scheme for organizing memory, while the task of organizing memory cannot be completed until the problem has been solved. Symbolic assemblers and compilers were developed to solve problems for which it was merely convenient to postpone the organization of memory. However, problems that involve highly dynamic reorganizations of storage as part of their solution require something beyond the compiler. (U)

After analyzing this class of problems and trying various methods of solution, it was found that list techniques provided a feasible solution and that the problems were readily programmable in list language. It is possible that better techniques may be discovered, but at present list

techniques provide an answer; hence, a new emphasis is beginning to emerge in the form of machine designs that make list processing easy, natural, and economical. The characteristics of the problems involved in correlating information from intelligence data place them in the category of problems solvable by list processing techniques; therefore, it is expedient that such techniques be investigated for possible application in a Fact Correlation System. (U)

Recent attempts to program computers for complex problems have been motivated by the desire to extend the capabilities of computers in understanding, thinking, and learning. The results of this work will form a basis for some of the functions of a Fact Correlation System, particularly where the analysis of natural languages is concerned. Most of the work in this area has been focused on formalized tasks such as chess playing and theorem proving. To program a computer to play chess or prove theorems poses communication problems between the programmer and the machine. These problems are far more difficult than those of expressing formal algebraic manipulations. The programmer must communicate to the machine his incomplete knowledge of how to behave in these complex situations. In these programs the entire pattern of data, both structure and content, evolves during the course of processing; hence, it is necessary to have the program construct its own data structure dynamically. The basis of list techniques is:

- (a) A list is a set of words tied together by having the address of each word in the list recorded in the word that occurs just prior to it in the list. Each word in the list contains two addresses: one gives the item of information at this location in the list; the other gives the link to the next word.

- (b) A list structure is a unit of data larger than the list.
It is compounded from lists by having the names of some lists occur as items of information on other lists.

Lists and list structures have been used as the basis of providing a unit of data larger than the single number. (U)

Considerable work is being performed in constructing programming languages using lists, most of it in connection with work on heuristic programs. Heuristic methods do not necessarily produce the best solutions to the problems to which they are applied. Some heuristic procedures simply attempt to construct a feasible solution to the problem; others perform transformations on feasible solutions to obtain others that are sometimes better solutions. The recent work of Newell, Simon, and Shaw as well as other men less well known has demonstrated the utility of heuristic methods and opened a whole new field of research. The prime concern of this work is to determine how human beings solve problems and to impart this ability to computers. (U)

1. List Storage Techniques

Digital computer design has been strongly oriented toward increased speed and facility in the arithmetic manipulation of numbers. However, a realization of the ultimate capacity of such machines, along with an understanding of the techniques of information processing, has led to the development of computer programs that deal largely with purely symbolic entities and with processes that are logical rather than arithmetic. There is a growing number of systems that arbitrarily manipulate symbolic expressions. The IPL systems of Newell, Simon, and Shaw, the LISP system of McCarthy, the Threaded List modification of Perlis, and the Multi-List system of Gray

and Prywes are all primarily based upon the fact that symbol manipulation can be carried out by connecting and disconnecting two-sided or branching elements to form lists and list structures. (U)

The intermediate data generated by such programs are generally unpredictable in their form, complexity, and length. Arbitrary lists of information may contain an arbitrary number of items or sublists as data. If a block of storage sufficient to contain some reasonable maximum amount of information were assigned to each possible list beforehand, all available rapid-access storage would be quickly exhausted and the organization of information would be rigidly prescribed. A program failure could easily occur if some list exceeded its allotted space, although much of the remaining storage was empty. (U)

Newell, Shaw, and Simon simulated a list-processing machine by programming a type of associative memory in which lists of arbitrary length and organization could be generated by annexing registers from a common store. This program afforded a substantial increase in programming flexibility at the cost of an apparent decrease in usable high-speed storage. There was a real decrease in the speed of indexing consecutive items in a list, since consecutive items of data in a list are not in consecutive memory registers but are connected by a string of location words. These location words determine the organization of the data; in addition, they can carry several bits of useful data and can be used to annex a given item on several different lists, thus eliminating repetitions and redundant data. Consequently, as the number and complexity of the generated lists increase, the density of useful information stored in the memory approaches one word per register. (U)

2. Problem of List Erasure

The primary merit of list storage is that the allocation of storage space for data is synthesized with the actual generation of the data. This scheme achieves a sort of local optimization, since each basic item of data occupies a minimal amount of space. A second advantage is that data having multiple occurrences need not be stored in more than one place in most cases. Hence, there is an overlapping or intersection of lists; the use of this overlapping may be regarded as a step toward the global optimization of storage allocation. However, the overlapping of lists leads to complications in list erasure. If a list is no longer required, just those parts that do not overlap other lists should be erased. No general method has been developed to perform this operation short of surveying all the lists in memory. However, several methods have been proposed. (U)

In McCarthy's method individual lists are abandoned; that is, the lists are disconnected from a base register rather than erased. When the list of available storage is exhausted, all registers currently employed in non-abandoned lists are surveyed. The complement of this set of accessible registers is then returned to the list of available storage. This method has two disadvantages. First, the time required to complete the reclamation process is nearly independent of the number of registers reclaimed; hence, its efficiency diminishes as memory approaches the saturation point. Second, the method requires that a bit be reserved in each word to tag accessible registers during the survey of accessibility. Since the sign bit would generally be used for this purpose, a further loss of efficiency occurs if signed integers or floating-point numbers are being used. (U)

Gelernter, Hansen, and Gerberich adopt the convention that each list component is owned by exactly one list and borrowed by all others in which it appears. A priority bit is then used in each location word to indicate whether the list entry is owned or borrowed. The erasing routine erases only those parts of a list that are owned. This technique is adequate so long as no list is erased that owns a part that has been borrowed by some list not yet erased. This restriction may present a problem, depending upon its application. (U)

A third method of list erasure is proposed by Collins. This method allows the arbitrary interspersal (in lists) of words containing reference counts. A two-bit type code then appears in each location word, one value of which serves to identify a reference count as such. (U)

3. Types of List Structures

Lists that can be searched efficiently tend to be difficult to change efficiently, and vice versa. Some lists are never changed; some are never searched. If a list must be both searched and changed, the conflict can often be resolved by devoting more storage space to the list. If there is insufficient storage space available, the problem is to implement a workable compromise between a search-oriented list and a change-oriented list design. (U)

A typical example of a search-oriented list is the sequence list; an example of a change-oriented list is the singly linked or push-down list. Push-down lists with an associated push-down counter are often used to store information required in the use of recursive subroutines. Since

each subroutine must return a push-down counter to the same state that existed when the subroutine started, information is stored on a last-in, first-out basis. The same list may also be used for temporary working space within the subroutine. A single push-down list is theoretically sufficient, although several are generally used. (U)

Threaded lists are an addition to list structure techniques. A threaded list is a list structure in which the last element of each list specifies the location of the head of the list of which it is the terminal member. The advantage of this structure is the addition of threads from the end of each sublist of a list to the next word on the list. This technique permits simple, efficient sequencing operations that will process each element of every sublist without recourse to push-down lists. In contrast to the usual list structures, a threaded list of complex structure essentially requires no more storage space for housekeeping information during sequencing operations than does a simple list. All the structural information is immediately accessible at the point in the list structure where it is applicable. The threaded list technique corresponds to the representation of all computable functions iteratively rather than recursively. Explicit program-controlled sequencing often permits simpler coding and processing. Iterative definitions require little or no increase in complexity over the more customary recursive definitions. (U)

4. List Processing Languages

In an associative memory the descriptors required to retrieve requested information are set in the control unit. In an effort to develop more intelligent machines, Newell, Simon, and Shaw have developed a series

of information processing languages (the IPL series) so that a machine can perform its own associations. The IPL series of programming languages was developed as an aid in constructing problem-solving programs, based upon the adaptive cut-and-try methods characteristic of human behavior, as a research tool in the study of heuristic problem-solving. (U)

The IPL languages were developed to handle problems with the following characteristics:

- (a) The problems basically involve the manipulation of symbols with other than numerical meaning and in other than algebraic systems.
- (b) The particular storage requirements of the problem-solving program cannot be specified in advance. Complex data structures are developed as the program proceeds.
- (c) The relationships between elements of data are restructured during the program's operation. New associations must be represented and old ones deleted.
- (d) The problem-solving process is naturally expressed at several levels of discourse, each built upon the lower levels.
- (e) The problem-solving procedure will be modified frequently as the program is developed and tested.

The fact that the program is modified during development reflects the use of the computer as a means of studying and learning about the problem. Consequently, the program must permit easy modification at various levels and with a minimum of interaction with the rest of the program. (U)

IPL-V is a formal language in terms of which information can be symbolized and processes can be specified for manipulating the information. IPL-V allows two kinds of expressions: data list structures that contain information and routines that define the information processes. (U)

McCarthy of MIT developed a list language called LISP, List Processor. This language, like IPL-V, uses lists for both data and routines. However, externally LISP uses a horizontal notation rather than the vertical list notation of IPL-V. In IPL-V list structures are represented horizontally with the aid of parentheses. (U)

In his program for proving theorems in plane geometry, Gelernter developed a list processing adjunct to FORTRAN called FLPL, Fortran List Processing Language. FLPL was developed by adding a series of subroutines to the FORTRAN system. It uses lists only for data; using FORTRAN produces the machine codes for routines. (U)

Another language that uses list-structure techniques is COMIT, a user-oriented, general-purpose, symbol-manipulation language. COMIT was originally conceived for research in mechanical translation, but features have since been added to convert it to a programming language. Developed at MIT primarily by Yngve, COMIT is especially convenient for problems ranging from mechanical translation to information retrieval, and from theorem proving to the maintenance of predominantly non-numerical files. (U)

The COMIT system consists of a two-pass compiler that translates the COMIT language into a machine-oriented notation that is then processed interpretively. Two built-in features are a dictionary search scheme and a simple search scheme that uses criteria such as class inclusion and context. COMIT manipulates strings as data; hence, it can deal with problems of natural language text concerning operations on strings of letters, words, phrases or sentences. The core storage capacity limits the number

of constituents that can be in the computer at any time. A constituent is defined as a letter or word of text, an algebraic symbol, or a retrieval item consisting of a document name, document number, or a set of descriptors. Storage allocation is automatic: in the compiler tables are automatically moved to new locations when they overflow; in the interpreter the free storage space is available as needed through list-structure techniques. Error detection and correction is carried on by the compiler. Although COMIT does not allow actual programming in English, it does include many of the patterns of natural language. (U)

Gray and Prywes have developed a Multi-List concept in which each item of data appears only once in an addressable memory; but each item has a number of descriptors and control information that assign the data to a number of separate lists. The Multi-List organization of memory has two parts:

- (a) A tree structure in which the nodes of the tree consist of memory locations for storing descriptors and the addresses of other nodes. The addresses provide the branches between the nodes. Successive addressing leads from a single entry point of the tree to an appropriate exit corresponding to a list defined by descriptors in the retrieval request.
- (b) A multi-association area where the file, organized in lists, is actually stored. This area is crisscrossed by the lists originating from the tree structure.

This technique requires a large amount of storage but has a high-speed access and retrieval time, as a function of program organization, and is particularly useful in heuristic systems. (U)

G. Storage Allocation Techniques

There is a need for developing automatic and semi-automatic methods

for assigning operational and information entities to available storage. Some work has already been done in this area; other work is in progress, but most of it is heuristic. (U)

There are two basic ways of handling the physical storage in computers:

- (a) By a fixed sequence of storage cells in which each cell has a unique numerical address.
- (b) By a referencing or linking scheme in which the address of a desired cell is obtained from some other cell or combination of cells.

Most computer storages are constructed according to the first method. Instances of the second method are generally found in automated schemes that modify the inflexibility inherent in the first method. These referencing or linking techniques are found in a hardware form or as integral parts of programming systems as evidenced by index registers, indirect addressing, list or tree memories, and related techniques. (U)

1. Dynamic Control

There are two basic ways of storing information in a computer. One method is to store the information directly in suitably coded form in the storage element of the computer. A somewhat less known but useful way is to assign the dynamic location of control to a program. Actual control is located in a particular place in a program at any given time only if certain branches have been taken, and these branches are taken only if certain conditions are satisfied. Hence, the current position of control in a program implies that these conditions are satisfied; the location of control is in this sense a device for remembering certain facts. Because of this

situation the program can call up from auxiliary storage only that data necessary for the operation of the program. All other data can be effectively ignored. (U)

2. Algorithms

The hard core of real storage allocation problems are difficult to formulate and solve rigorously. As a result, the only recourse has been to develop algorithms that look as if they will yield reasonable answers to problems. The more that is learned about techniques for solving problems, the better the chance for success. (U)

Recently, considerable interest has been shown in obtaining solutions to combinatorial problems. There are two reasons for this interest: practically, the advent of the computer has motivated the formulation of problems that previously were solvable only in primitive ways; theoretically, the methods of linear programming have been used to prove some classical theorems about combinatorial problems and to provide algorithms for their solution. Storage allocation problems, in most cases, belong to the class of combinatorial problems, which are characterized by the fact that their solutions form discrete sets of points. (U)

At present, few practical methods for obtaining solutions to problems of this type satisfy any objective criterion precisely. However, various algorithmic methods have been used to solve combinatorial problems representing real situations by producing usable solutions that satisfy the objective criterion reasonably, although not exactly. Sampling is one such method; a subset of the possible solutions is generated by some

random selection rule and the best of these solutions is selected for implementation. Algorithms of this type are called heuristic. (U)

3. Logic Structure Tables

Logic tables provide a method of expressing the logic required in procedures, operations, systems, and circuits; these tables can be readily compiled into computer programs. This capability has led to certain conventions that maximize their usefulness and establish a common language. A complete table is a compact expression of many paths in parallel; the table immediately shows the similarities, differences, and direction of each path. By the discipline of the technique, the table must include all possible paths and must eliminate ambiguities. (U)

The logic of a problem can always be written out in words and sentences, but this procedure necessitates a sequential consideration of alternative paths rather than the more desirable simultaneous analysis. Such techniques as flow charts, symbolic logic, and algebra are used, but these techniques are primarily sequential methods of expressing logic. Logic tables, however, are two-dimensional in nature and permit the expression of both the sequential and parallel aspects of logic; this characteristic uniquely suits them to the development and expression of the logic of a problem. TABSOL, TABular Systems Oriented Language, is an example of a logic structure table program that provides a type of decision table. It depicts, by means of tables, the relationships of logical decisions written in terms of conditions to be satisfied and the subsequent action to be taken. (U)

H. Adaptive Systems

In developing programs to perform the various tasks required for fact correlation, the problems of adaptive systems must be considered. The study of adaptation involves the study of both the adaptive system and its environment--that is, the study of how systems can generate procedures that enable them to adjust efficiently to their environments. The adapting system must be able to generate any method or procedure capable of an effective definition; the set of all effectively defined procedures are identified with the set of all programs of some suitably specified universal computer. Unrestricted adaptability would require that the adaptive system be able to generate initially any of the programs of some universal computer, assuming that nothing is known about the environment initially. The process of adaptation can then be considered as a modification of the generation process as information about the environment is accumulated. (U)

The problem is not one of producing some particular program; rather, it is the ability to generate a particular population of programs. The generated population of programs can then act upon another population of programs (the environment) in an attempt to produce solutions. The adaptive system must be able to compare generation procedures as to their efficiency in producing solutions. Holland⁽¹⁾ suggests a method for comparing two generation procedures confronted by the same environment in terms of their ratings. Adaptation within this context involves two concurrent processes: sampling the environment to produce estimates of the

(1) Holland, J. H., Outline for a Logical Theory of Adaptive Systems; ACM Journal, Vol. 9, No. 3, July 1962.

environment; and modifying the generation procedure to obtain the generation procedure with the highest rating relative to the current estimate. (U)

To implement the rating technique, each adaptive system must have a central control over its associated generation procedure. This control, or supervisory program, gathers the threads of control at a single point. Adaptation then, according to Holland's method, is based upon the differential selection of supervisory programs. The more successful a supervisory program is in terms of its problem-solving ability, the more predominant it will become in a population of supervisory programs. Differential selection presupposes that the supervisory programs must be capable of self-duplication and that the rate of duplication of a supervisory program must be determined by the effectiveness of the problem-solving programs it controls. (U)

In this procedure the problem of adaptation concerns the modification of a free generation procedure by the control program acting on the environment. This modification can only be carried out in terms of available information about the environment, which can only be obtained through trials or experiments performed on the environment. (U)

I. Composition of Programs

A Fact Correlation System may be required to compose programs on an as needed basis. Such capability will ensure that the system deals adequately with categories of tasks not specified in advance. (U)

The composition of programs as needed involves several major problems.

The composition of programs requires the selection of a programming basis; that is, ordered sets of instructions that serve as basic units out of which larger programming units may be synthesized. There are several criteria that govern the selection of elementary programming units:

- (a) Their selection should reflect available information about the way useful programs are composed.
- (b) Since information about the composition of programs is continuously accruing to the system, the programming basis may be expected to change with time. (U)

The process of composing a program involves the exchange of information between the user, the executive function, and the programming system or compiler. The user prescribes the program at a level of specification. The programming system translates from the level of specification into a program written in basic machine language. This function can be accomplished only if the system is able to elicit more information from the user through the executive function. The executive function assists the programming system either passively or actively. Passive assistance consists of answering machine queries pertaining to the elucidation of specifications furnished by the user. Active assistance takes the form of guiding the programming system in the composition of programs. The executive function performs this task by answering questions pertaining to the suitability of tentatively composed programs; these questions may, in turn, be referred to the user for confirmation. (U)

The ability to compose programs or order a series of subroutines is predicated upon the programming system's capacity to interrogate the user. The successful formation of questions depends upon the system's ability to

understand adequately the purport of the composed program. This process may be conceived as the inverse of the problem of constructing a programming language; that is, given a program, the objective is to construct a derived sequence of statements that will communicate the meaning of the program to the executive function. (U)

The basic programming units will increase in complexity and size as the programming system acquires more experience. This development may be visualized as follows. If the system started with no information whatsoever about the structure of useful programs, every instruction in the machine's repertory would be equally likely to combine with any other instruction. With the acquisition of experience, some combinations of instructions would be ruled out; others would be almost always inseparable; still other sets of instructions would be likely to occur in certain types of programs but not in others. The meaning of such sets of instructions depends upon the programs in which these sets occur and upon the function of the instructions in the programs. Upon reaching maturity, the system will be able to avail itself of complex sequences of instruction as building blocks in the construction of programs. These sequences, moreover, will be modifiable in accordance with sets of experiential rules. (U)

The theoretical feasibility of composing programs from basic instructions leads to a practical means for implementing this concept. Experience about the sets of combined instructions may be gained in programming. These sets and their interrelationships may then be included as constituents of the programming system. The formation of more complex programs from the basic sets is then a function of requirements specified by the user and

interpreted by the programming system. In its broadest sense this process is self-organizing; but its feasibility depends upon the ability to exchange information about the organization of behavior between the programming system and its environment--specifically, the user. (U)

J. Summary

The success of any system that includes a computer among its elements depends upon the programs that organize the behavior of the computer. The programs must fulfill the processing functions imposed by system requirements. But the effort required to write programs, especially for complex operations, must also be simplified. (U)

The organization of programs, including the executive function, is a specific task in the process of developing a Fact Correlation System. A comprehensive review of programming languages, list processing techniques, and storage allocation techniques has indicated that such a system can be developed through an extension of current techniques, except that these techniques require a greater degree of sophistication. List processing techniques seem particularly promising for use in the analysis of natural language. An effective compiler will also be required for the system. The development of a compiler will depend to some extent upon the characteristics required to write programs for linguistic analysis and adaptive learning. Further research in developing compilers depends entirely upon the nature of the functions imposed by these requirements. However, extensive research problems are not anticipated. The ability to compose programs on a self-organizing basis may require special research. (U)

There has been a tendency in the past toward single address hardware and three plus zero address pseudo-codes. Although this type of instruction has been adequate for most applications, it would be advantageous to have a variable instruction, variable word-length machine. This design feature would enable a single machine (or pseudo-machine) word to encompass all of a complete sentence. A development of this nature would be particularly useful for a Fact Correlation System. (U)

K. References

- (1) Bottenbruch, H., "Structure and Use of ALGOL 60," ACM Journal, Vol. 9, No. 2; April 1962.
- (2) Burroughs Corporation, An Introduction to ALGOL 60; Company literature.
- (3) Cantrell, Harry N., "Where Are Compiler Languages Going?" Datamation, Vol. 8, No. 8; August 1962.
- (4) Cantrell, H. N., King, J. N., and King, F. E. H., "Logic-Structure Tables," Communications of the ACM, Vol. 4, No. 6; June 1961.
- (5) Cheatham, T. E., and Collins, G. O., Programmed Parallel Computation Sequences; Technical Operations, Inc., Burlington, Massachusetts, August 1960.
- (6) Cheatham, T. E., Collins, G. O., and Leonard, G. F., "CL-1, An Environment for a Compiler," Communications of the ACM, Vol. 4, No. 1; January 1961.
- (7) Collins, George E., "A Method for Overlapping and Erasure of Lists," Communications of the ACM, Vol. 3, No. 12; December 1960.
- (8) Emery, J. C., "Modular Data Processing Systems Written in COBOL," Communications of the ACM, Vol. 5, No. 5; May 1962.
- (9) Fortran Assembly Program (FAP) for the IBM 709/7090, IBM Bulletin J28-6098-1; 1961.
- (10) Gelernter, H., Hansen, J. R., and Gerberich, C. L., "A Fortran-Compiled List-Processing Language," ACM Journal, Vol. 7, No. 2; April 1960.

- (11) General Electric Corporation, GECOM - General Compiler; Company literature.
- (12) Ginsburg, S., and Rice, H. G., "Two Families of Languages Related to ALGOL," ACM Journal, Vol. 9, No. 3; July 1962.
- (13) Grau, A. A., "Recursive Processes and ALGOL Translation," Communications of the ACM, Vol. 4, No. 1; January 1961.
- (14) Gray, H. J., Prywes, N. S., et al., The Multi-List System, Technical Report No. 1, Vol. 1; The Moore School of Electrical Engineering, University of Pennsylvania, November 1961.
- (15) Hibbard, Thomas N., "Some Combinatorial Properties of Certain Trees with Applications to Searching and Sorting," ACM Journal, Vol. 9, No. 1; January 1962.
- (16) Holland, John H., "Outline for a Logical Theory of Adaptive Systems," ACM Journal, Vol. 9, No. 3; July 1962.
- (17) Kelley, J. E., "Techniques for Storage Allocation Algorithms," Communications of the ACM, Vol. 4, No. 10; October 1961.
- (18) McMahon, James T., "ALGOL versus FORTRAN," Datamation, Vol. 8, No. 4; April 1962.
- (19) Newell, Feigenbaum, et al., The Elements of IPL Programming, IPL-V Manual; The RAND Corporation, Santa Monica, California, 1960.
- (20) Newell, A., and Tonge, F. M., "An Introduction to Information Processing Language V," Communications of the ACM, Vol. 3, No. 4; April 1960.
- (21) Perlis, A. J., and Thornton, C., "Symbol Manipulation by Threaded Lists," Communications of the ACM, Vol. 3, No. 4; April 1960.
- (22) Remington-Rand Corporation, MATH-MATIC, Remington Rand Automatic Programming System; Company manual.
- (23) Richards, Paul, Parallel Programming; Technical Operations, Inc., Burlington, Massachusetts, August 1960.
- (24) Ross, Douglas T., "A Generalized Technique for Symbol Manipulation and Numerical Calculation," Communications of the ACM, Vol. 4, No. 3; March 1961.
- (25) Sammet, Jean E., "Basic Elements of COBOL 61," Communications of the ACM, Vol. 5, No. 5; May 1962.

- (26) Schwartz, Eugene S., "An Automatic Sequencing Procedure," ACM Journal, Vol. 8, No. 4; October 1961.
- (27) Weizenbaum, J., "Knotted List Structures," Communications of the ACM, Vol. 5, No. 3; March 1962.
- (28) Yngve, Victor H., "COMIT as an I-R Language," Communications of the ACM, Vol. 5, No. 1; January 1962.

CONFIDENTIAL

V. EQUIPMENT EVALUATION

A. Objectives and Methodology

In order to establish equipment requirements for a Fact Correlation System, the capabilities of various existing equipments and some experimental devices were analyzed and evaluated. The purpose of this study was to determine the ability of each equipment to fulfill the input-output, processing, and storage and retrieval functions demanded by a system for analyzing intelligence data. In establishing basic equipment requirements for a system, it became apparent that certain areas would require further development in order to achieve an efficient system; an efficient system is defined as one that has no time lag between computer operations and handles the estimated flow of information without loss of speed or degradation of system performance. With those areas that indicated a need for immediate development defined, the study was then extended to include an analysis of research being conducted in all areas and a prediction of what may be expected within the next five to fifteen years. (C)

A data processing system begins with a set of objectives specified for the system. Then the data needs, the data flow throughout the system, and the means for achieving the objectives are determined; next, these factors are translated into requirements for each function of the system. These requirements are established as criteria that the equipment in the system should meet in order to attain a continuous flow of information from input, through processing, to output. The volume of information to be handled by the system will cover a range of possible values; similarly, each equipment selected must be capable of handling the maximum volume of

CONFIDENTIAL

input at speeds compatible with the remainder of the system--a balance of input time and processing time improves efficiency. Equipment reliability and costs also play a part in the selection of appropriate equipment. (U)

The first step in this analysis was to establish performance criteria as a standard for evaluating the effectiveness of the equipment. These criteria led to the elimination of some equipment as being inadequate for the system because of such factors as speed, capacity, or reliability. (U)

The second step was to set up a range of parameters and to evaluate various equipment configurations within this parametric framework. The problem was approached in two ways:

- (a) Starting from the environment, a range of input parameters (rates of data reception in terms of pages per minute or characters per second) was established; and the rates at which these data would travel through the system were calculated for various combinations of input, processing, storage, retrieval, and output equipment or techniques.
- (b) Starting from within the system and working outward from the processing, storage, and retrieval of the data, the input and output rates needed to satisfy the processing rates of the system were computed. In this case also, several values were considered for the processing, storage, and retrieval rates.

The purpose of both these approaches was to isolate the bottlenecks in the system and to establish the best combination of equipments and techniques to solve the problem efficiently and economically. (U)

The third step was to analyze these configurations and to select optimum components for the system; i.e., those components that came closest to satisfying all the requirements of the system. (U)

The final step was to analyze the optimum system selected from the configurations, determine the areas of greatest deficiency, and indicate the equipment that is expected to be available in the near future to help resolve these deficiencies. (U)

The one-step-beyond the final step was a study of the long-range research in data processing equipment and techniques; the purpose of this was to establish the foundations for forecasting the system of the future. (U)

B. Performance Criteria

In establishing the performance criteria for the equipment required in a Fact Correlation System, the following factors were considered:

- (a) Volume of data--a minimum and a maximum value with a range of values falling between them.
- (b) Speed--the operational rates of the individual components and their relation to the speed of the remainder of the system.
- (c) Design--the desirable and undesirable features of various types of components.
- (d) Reliability--the stability and durability of the individual components and their contribution to over-all system reliability.
- (e) Cost--a balancing of cost versus time and efficiency. (U)

1. Volume of Data

In designing a system, consideration is necessarily given to the amount of data the system is expected to handle, including an allowance for future increases in data flow. Most present day data processing systems are designed modularly and allow for future expansion. In establishing the volume of data handled as a performance criteria, maximum and

SECRET

minimum values were established for a range of parameters. Each equipment configuration was then subjected to these input parameters in order to determine the ability of various equipments to handle increasing amounts of data. (U)

The input parameters chosen for this study were based upon the following assumptions:

- (a) Intelligence data input is single-spaced elite type or newsprint with 48 lines per page and 72 characters per line.
- (b) Data is received at an average rate of 2,000 pages per day (83.33...pages per hour; 1.388...pages per minute).
- (c) The maximum data input rate is 12,000 pages per day (500 pages per hour; 8.33...pages per minute).

On the basis of these assumptions the following input parameters were selected:

<u>Case No.</u>	<u>Pages per Minute</u>	
	<u>Single Spaced</u>	<u>Double Spaced</u>
1	1	2
2	8	16
3	20	40

The selected parameters cover the assumed input range and extend well beyond the range in order to preclude contingencies caused by increases in the volume of data after a system becomes operational. (S)

2. Speed

Feasible speed is a function of such factors as the need for the information, the configuration of the data processing equipment, and the number and skills of people required to achieve various rates of data

SECRET

flow. It is pointless to enter data at a rate faster than the rate at which it can be processed. If output is required almost instantaneously, then real-time processing of information at high speeds is required; if output can be batched, then the necessary speeds will not be so great. Present day computers operating at microsecond speeds are capable of performing many instructions per unit of time, but the over-all processing time depends upon the length of the program and the types of operations to be performed. Presently available storage devices provide large capacity storage coupled with high-speed random accessing. The greatest problem in the speed of existing equipment is in the input-output area. (U)

Much emphasis is being placed on research and development into new techniques and equipment for handling input and output data at rates comparable to those of the computers. Reading devices promise increased input speeds by permitting typewritten documents to be directly read into a computer at speeds of 10,000 characters per second or greater. This procedure is a significant improvement over keypunching, where data is punched at an average rate of approximately 7,000 characters per hour. (Keypunched data must also be placed into a card reader and transferred to magnetic tape before being input into the computer.) Another bottleneck exists where there is a need to abstract data from documents before it can be used as input. Any routine process that depends largely upon human operators is relatively slow and prone to errors. (U)

Increases in processing speeds and efficiency can be achieved by techniques such as parallel operations in which several instructions can be carried on simultaneously by overlapping memory cycles, asynchronous

operations, or parallel programming. As research in the field continues, new equipment will be developed to attain even greater speeds with increased reliability. (U)

3. Design

Several desirable features may influence the selection of one equipment in preference to a similar equipment lacking these features, everything else being equal. For example, such features as automatic positioning of documents, automatic consecutive numbering of documents, or concurrent production of several input media in a single operation are useful in input equipment. The ability to read multiple type fonts and single-spaced type are desirable features for reading devices. Error detection, error correction, and low error rates are equally desirable in all input and output equipment. (U)

Flexibility is another design feature that is highly desirable. Input, output, and storage devices that are limited in their use to only one or two computers or that require excessive additional equipment to be compatible with other computers are less desirable than flexible, readily adaptable equipment. (U)

The concept of parallel operations should play an important role in the choice of a computer. Many presently available computers afford some parallel operating capabilities with time sharing of the computer memory in such operations as table lookups, input, or output. Several investigations are currently being conducted in the area of parallel processing--the sharing of a common memory and all peripheral equipment by two or more processor

units--and parallel programming, the use of a buffered processor by time-sharing its central control and its memory between two or more object programs under control of an executive system. (U)

4. Reliability

The problem of reliability covers three areas: equipment, components, and systems. Equipment reliability is measured in terms of available time and accuracy. Available time is the amount of time that the equipment is in operation, except for time used in scheduled or unscheduled maintenance. Modular design and, in some instances, duplication of equipment may contribute substantially to the reliability of the system. (U)

Ensuring the accuracy of input data is a formidable problem. There are several ways to solve this problem, and further analysis will be required to discover the most efficient and economical method. For example, computers may perform part of the data-input function by incorporating error detection and correction routines into their programs. This technique may be useful, since the assumed function of the system is to process information received in the form of ordinary English sentences. Whenever discrepancies arise and self-correcting programs fail, the system should request correct information from an analyst. In addition, it may be necessary to incorporate as much error detection as feasible into the input equipment itself. Error detection as close to the source as possible is desirable so that corrective actions can be taken immediately. Automatic interlocks, validity checks, and warning signals may be included as part of the input equipment. (U)

The probability of a complete electronic system functioning as intended is computed by multiplying the probabilities of the individual components and connections composing the system. For example, if a component has a rated reliability of 90 percent and seven of these components are connected to form a system, the over-all reliability of the system would be $.90 \times .90 \times .90 \times \dots$ or .48; that is, the reliability of the system would be 48 percent, assuming 100 percent reliability for the connections between the components. However, the connections are usually less reliable than the components. Therefore, a reduction in the number of either the components or the interconnections within a system will improve the over-all reliability of a system faster than the improvement in the reliability of individual components. (U)

Molecular electronics, a new concept of design in electronic systems, seeks to integrate into a solid block of material the functions performed by one or several electronic circuits. If a given function is performed within a single solid block of material, interconnections are eliminated completely. This design concept is one of several that promise a higher degree of reliability for future systems. (U)

5. Cost

In justifying the cost of a piece of equipment, it is necessary to take many factors into account. For example, if a piece of equipment is to replace a human operation, one must consider the number of human operators that would have been required to perform the function at the same speed as the machine, their aggregate salaries, and the difference in error rate and correction time. Similarly, the use of built-in error

detection in input equipment is only justified if the cost is balanced by the saving in processing time. (U)

Another cost factor is the cost per bit in storage devices. As the capacity of the device increases, the cost per bit must necessarily decrease; otherwise, the increased storage will be priced out of the picture for most users. Increased speed will permit some increase in cost per bit, if the increased speed is mandatory and if it affords an over-all savings. (U)

C. Parameters and Variables for Configurations

1. First Method

For the first method of approach in establishing equipment configurations, the rates (in characters per second) at which documents are being received for input into the system were calculated using the selected parameters for Cases No. 1, 2, and 3 (see page 174). Assuming 72 characters per line and 48 lines per page, the number of characters per page is 3456. Therefore:

<u>Case No.</u>	<u>Pages per Minute</u>	<u>Characters per Minute</u>	<u>Characters per Second</u>	
1	1	3,456	57.6	
2	8	27,648	460.8	
3	20	129,128	1152.0	(U)

The calculations performed in setting up configurations were based upon a single document of 10,000 characters. The time to receive a document in each of the three cases is:

<u>Case No.</u>	<u>Character Reception Rate (seconds)</u>	<u>Document Reception Rate (seconds)</u>
1	57.6	173.6
2	460.8	21.7
3	1152.0	8.7

These figure indicate that in Case No. 1 a document of 10,000 characters is received by the system every 173.6 seconds (or nearly 3 minutes); in case No. 2, every 21.7 seconds; and in Case No. 3, every 8.7 seconds. These three cases are represented graphically in Figure 5-1. The continuous curve allows for extrapolations to different combinations of reception rates. (U)

In Table 5-1 the rates in characters per second are given in Column 2 for equipment listed in Column 1. The amount of time required for the equipment to handle a single document of 10,000 characters is shown in Column 3.⁽¹⁾ Columns 4, 5, and 6 indicate the number of units of equipment (or human beings) that would be needed to handle documents received in a continuous flow at the rates established for the three test cases. In some instances it is immediately apparent that a particular equipment or a human operator is unsatisfactory for handling the assumed volume of data. (U)

2. Second Method

For the second method of approach in establishing equipment configurations, parameters were selected to represent the length of computer

⁽¹⁾ The equipment selected for use in the configurations is representative of the state-of-the-art but is by no means exhaustive.

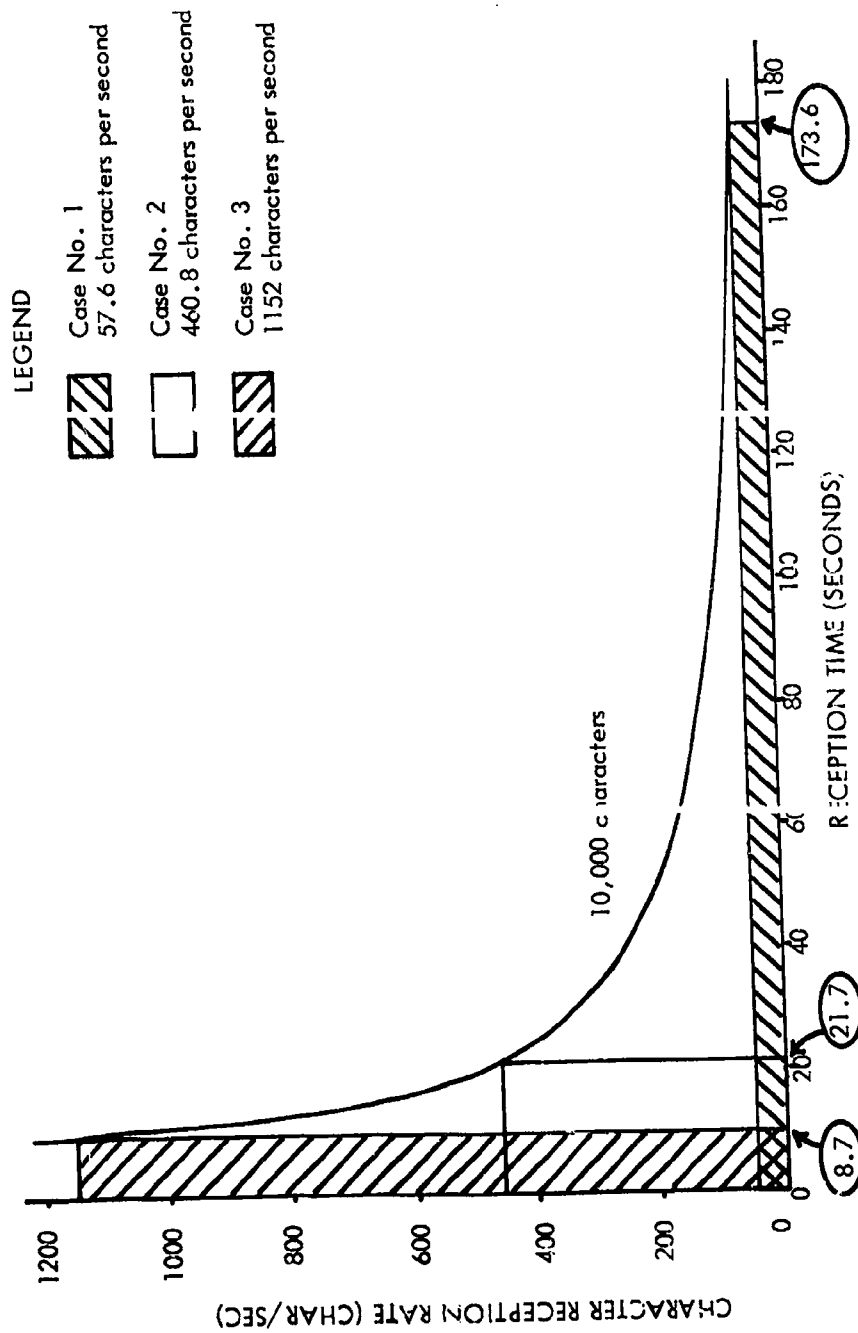


FIGURE 5-1. . . Reception Rate versus Time for 10,000 Characters

TABLE 5-1. INPUT EQUIPMENTS AND FUNCTIONS

EQUIPMENT OR FUNCTION	RATE (CHARACTERS PER SECOND)	TIME TO HANDLE ONE DOCUMENT* (SECONDS)	NUMBER OF UNITS NEEDED		
			CASE NO. 1 (57.6 CHAR- ACTERS PER SECOND)	CASE NO. 2 (460.8 CHAR- ACTERS PER SECOND)	CASE NO. 3 (1152 CHAR- ACTERS PER SECOND)
Abstracting	100,000 characters/ day	48 minutes	17 people	133 people	331 people
Keypunching	2	500	3	23	58
Card Reader No. 1 (250 cards/minute)	300	33.3	1	2	4
Card Reader No. 2 (800 cards/minute)	960	10.4	1	1	2
Card Reader No. 3 (2000 cards/minute)	2400	4.13	1	1	1
Card Reader No. 4 (3000 cards/minute)	3600	2.8	1	1	1
Optical Scanner	180	55.5	1	3	7
Reading Machine	6000 to 10,000	1.6 to 1	1	1	1
Magnetic Tape Unit No. 1	60,000	0.167	1	1	1
Magnetic Tape Unit No. 2	120,000	0.083	1	1	1

*One document - 10,000 characters

programs. The various tasks to be performed upon the input were analyzed and the number of instructions required to process a single document at one time were estimated. The following estimates are not indicative of the total number of instructions required to perform various combinations of each task.

<u>Task</u>	<u>Instructions (per document)</u>
Language Transformation	
--Letter Sequences	2,000
--Grammatical Sequences	6,000
--Semantic Analysis	6,000
--Logical Consistency	3,000
Symbolic Memory Structure	2,000
Adaptive Learning	10,000
Correlation Techniques	6,000
Sentence Generator	4,000
Executive Routine	<u>1,000</u>
Average instructions per document	40,000

Since it is difficult to conceive an accurate idea of the size of a program for correlating facts, and since the programming techniques and types of computing devices are additional variables in the problem, two additional parameters were chosen to represent upper and lower bounds for the average number of instructions. The three selected parameters are:

<u>Case No.</u>	<u>Instructions (per document)</u>	
1	25,000	
2	40,000	
3	100,000	(U)

In considering the processing of an input item--e.g., a single document of 10,000 characters--allowance was made for reiteration of portions

of the program, since the number of instructions in the program does not account for such programming techniques as looping, counting, indexing, or similar operations. The number of operations required to process one document was computed according to the formula: $6.4 \text{ (Instructions)} = \text{Number of Operations}$. Hence, for the estimated number of instructions:

<u>Case No.</u>	<u>Instructions (per document)</u>	<u>Operations (per document)</u>	
1	25,000	160,000	
2	40,000	256,000	
3	100,000	640,000	(U)

The next consideration was the type of computer and computer program. The evaluations were based upon a high-speed computer using an arithmetic-type program, a high-speed computer using a logical-type program, a medium-speed computer, and a low-speed computer. These four types are represented by the following operations rates:

<u>Type No.</u>	<u>Type of Computer</u>	<u>Operations (per second)</u>
1	High Speed Arithmetic	191,000
2	High Speed Logical	114,500
3	Medium Speed	28,000
4	Low Speed	3,500

The processing time for a single document of 10,000 characters, assuming programs of 25,000, 40,000, and 100,000 instructions, is given in Table 5-2. These processing times are based upon operations performed from a high-speed magnetic core memory; therefore, the next problem is to correct these figures on the basis of high-speed storage capacity available and auxiliary storage capacity obtained via drum, disc files, or magnetic tape. (U)

TABLE 5-2. PROCESSING TIMES FOR A SINGLE DOCUMENT

	NUMBER OF INSTRUCTIONS	NUMBER OF OPERATIONS PER DOCUMENT	PROCESSING TIME PER DOCUMENT (SECONDS)			
			TYPE NO. 1	TYPE NO. 2	TYPE NO. 3	TYPE NO. 4
Case No. 1	25,000	160,000	0.84	1.4	5.7	45.7
Case No. 2	40,000	256,000	1.34	2.24	9.0	73
Case No. 3	100,000	640,000	3.35	5.5	22.9	183

Type No. 1	High Speed Arithmetic	191,000 operations per second
Type No. 2	High Speed Logical	114,500 operations per second
Type No. 3	Medium Speed	28,000 operations per second
Type No. 4	Low Speed	3,500 operations per second

To obtain the storage requirements for each of the three cases of computing parameters, the number of words of input for a single document (assumed to be approximately 2,000 words for a 10,000 character document) was added to the number of instructions in the program. The following storage requirements resulted:

<u>Case No.</u>	<u>Storage Locations</u>
1	27,000
2	42,000
3	102,000

For programs processed on Types No. 1 or No. 2 computers, core storages of 16K, 32K, and 64K were assumed; for Type No. 3, 16K and 32K storage; and for Type No. 4, 8K and 32K storage. Where these conditions failed to provide adequate storage capacity to meet the estimated requirements, a disc and a tape storage were added for Types No. 1 and No. 2, and drums and tapes were used for Types No. 3 and No. 4. (These assumptions do not include storage capacity required for other functions.) (U)

The storage allocations for the different configurations are listed in Table 5-3. The total processing time is a combination of the processing time in core plus the access time and read/write time for the auxiliary storage device. The total access time for the auxiliary storage device is the access time multiplied by a factor F, representing the number of times the device is referenced. The factor was computed as follows:

$$F = \frac{\text{Total Storage Requirement}}{\text{Core Storage Requirement}} = \text{integer plus a fraction}$$

---The integer is the factor F if there is no fraction.

---Add one to the integer to obtain F if there is a fraction.

TABLE 5-3 (PART I). STORAGE ALLOCATIONS

COMPUTING DEVICE	CORE STORAGE	AUXILIARY STORAGE	AUXILIARY STORAGE CAPACITY (WORDS)	AUXILIARY STORAGE MAXIMUM ACCESS TIME (SECONDS)	AUXILIARY STORAGE READ/WRITE RATE (WORDS PER SECOND)	COMMENTS RELATIVE TO CASES*
Type No. 1 and Type No. 2	16K	---	---	---	---	Inadequate all 3 cases
	32K	---	---	---	---	Adequate Case 1 only
	64K	---	---	---	---	Adequate Cases 1-2 only
	16K	tape	720K	128	10,000	Adequate all 3 cases
	32K	tape	720K	128	10,000	Adequate all 3 cases
	64K	tape	720K	128	10,000	Adequate all 3 cases
	16K	disc	8300K	0.21	10,000	Adequate all 3 cases
	32K	disc	8300K	0.21	10,000	Adequate all 3 cases
	64K	disc	8300K	0.21	10,000	Adequate all 3 cases

*Case No. 1 requires 27K storage capacity
Case No. 2 requires 42K storage capacity
Case No. 3 requires 102K storage capacity

TABLE 5-3 (PART II). STORAGE ALLOCATIONS

COMPUTING DEVICE	CORE STORAGE	AUXILIARY STORAGE	AUXILIARY STORAGE CAPACITY (WORDS)	AUXILIARY STORAGE MAXIMUM ACCESS TIME (SECONDS)	AUXILIARY STORAGE READ/WRITE RATE (WORDS PER SECOND)	COMMENTS RELATIVE TO CASES*
Type No. 3	16K	---	---	---	---	Inadequate all 3 cases
	32K	---	---	---	---	Adequate Case 1 only
	16K	tape	500K	192	2,500	Adequate all 3 cases
	32K	tape	500K	192	2,500	Adequate all 3 cases
	16K	drum	8K	0.035	10,000	Case 1 requires 2 drums
	32K	drum	8K	0.035	10,000	Case 2 requires 2 drums
Type No. 4	8K	---	---	---	---	Inadequate all 3 cases
	32K	---	---	---	---	Adequate Case 1. only
	8K	tape	1000K	96	10,000	Adequate all 3 cases
	32K	tape	1000K	96	10,000	Adequate all 3 cases
	8K	drum	3K + 16K	0.0085	10,000	Case 1 requires 3 drums
	32K	drum	16K	0.0085	10,000	Adequate Cases 1-2

*Case No. 1 requires 27K storage capacity
Case No. 2 requires 42K storage capacity
Case No. 3 requires 102K storage capacity

The total read/write time is obtained by subtracting the amount of core storage from the total storage requirement and multiplying this quantity by a factor of 0.2 (based upon a read/write rate of 10,000 words per second). Therefore, the total processing time is:

$$\begin{array}{rclcl} \text{Total} & & \text{Core} & & \\ \text{Processing} & = & \text{Processing} & + & F \left[\begin{array}{c} \text{Auxiliary} \\ \text{Storage Access} \\ \text{Time} \end{array} \right] & + & \text{Read/} \\ \text{Time} & & \text{Time} & & & & \text{Write} \\ & & & & & & \text{Time} \end{array}$$

For example: assume a configuration consisting of a 16K core storage with a processing time of 0.84 seconds for a single input document and an auxiliary magnetic tape with an access time of 128 seconds. Then for Case No. 1, which requires a storage capacity of 27K:

$$(a) \quad F = \frac{27}{16} = 1 + \text{fraction} = 2$$

$$\begin{aligned} (b) \quad \text{Total Access Time} &= F(\text{Access Time}) \\ &= 2(128) \\ &= 256 \text{ (seconds)} \end{aligned}$$

$$\begin{aligned} (c) \quad \text{Read/Write Time} &= 0.2 (27-16) \\ &= 2.2 \text{ (seconds)} \end{aligned}$$

$$\begin{aligned} (d) \quad \text{Total Processing Time} &= 0.84 + 256 + 2.2 \\ &= 259.04 \text{ (seconds)} \end{aligned}$$

Table 5-4, Parts 1 through 4, lists the total processing time for each combination of core storage and auxiliary storage in the three selected cases using the four specified types of computing devices. (U)

The input times in Table 5-1 and processing times in Table 5-4 were used in the analyzing various configurations as described in the following sections. (U)

TABLE 5-4 (PART I). TOTAL PROCESSING TIMES

STORAGE REQUIREMENTS	COMPUTING DEVICE	CORE STORAGE	AUXILIARY STORAGE	PROCESSING TIME IN CORE (SECONDS)	ADDITIONAL PROCESSING TIME		TOTAL PROCESSING TIME (SECONDS)
					Access Time	Read/Write Time (Seconds)	
Case No. 1 27K Storage Requirement	Type No. 1	32K	none	0.84	none		0.84
		64K	none	0.84	none		0.84
		16K	tape	0.84	256	2.2	259.04
		16K	disc	0.84	0.42	2.2	3.46
Case No. 2 42K Storage Requirement	Type No. 1	64K	none	1.34	none		1.34
		16K	tape	1.34	384	5.2	390.54
		32K	tape	1.34	256	2	259.34
		16K	disc	1.34	0.63	5.2	7.17
		32K	disc	1.34	0.42	2	3.76
Case No. 3 102K Storage Requirement	Type No. 1	16K	tape	3.35	896	18	917.35
		32K	tape	3.35	512	14	529.35
		64K	tape	3.35	256	8	267.35
		16K	disc	3.35	1.47	18	22.82
		32K	disc	3.35	0.84	14	18.19
		64K	disc	3.35	0.42	8	11.77

TABLE 5-4 (PART II). TOTAL PROCESSING TIMES

STORAGE REQUIREMENTS	COMPUTING DEVICE	CORE STORAGE	AUXILIARY STORAGE	PROCESSING TIME IN CORE (SECONDS)	ADDITIONAL PROCESSING TIME Access + Read/Write Time (Seconds)	TOTAL PROCESSING TIME (SECONDS)
Case No. 1 27K Storage Requirement	Type No. 2	32K	none	1.4	none	1.4
		64K	none	1.4	none	1.4
		16K	tape	1.4	256 + 2.2	259.6
		16K	disc	1.4	0.42 + 2.2	4.02
Case No. 2 42K Storage Requirement	Type No. 2	64K	none	2.24	none	2.24
		16K	tape	2.24	384 + 5.2	391.44
		32K	tape	2.24	256 + 2	260.24
		16K	disc	2.24	0.63 + 5.2	8.07
		32K	disc	2.24	0.42 + 2	4.66
Case No. 3 102K Storage Requirement	Type No. 2	16K	tape	5.6	896 + 18	919.6
		32K	tape	5.6	512 + 14	531.6
		64K	tape	5.6	256 + 8	269.6
		16K	disc	5.6	1.47 + 18	25.07
		32K	disc	5.6	0.84 + 14	20.44
		64K	disc	5.6	0.42 + 8	14.02

TABLE 5-4 (PART III). TOTAL PROCESSING TIMES

STORAGE REQUIREMENTS	COMPETING DEVICE	CORE STORAGE	AUXILIARY STORAGE	PROCESSING TIME IN CORE (SECONDS)	ADDITIONAL PROCESSING TIME Access + Read/Write Time (Seconds)		TOTAL PROCESSING TIME (SECONDS)
Case No. 1 27K Storage Requirement	Type No. 3	32K	none	5.7	none		5.7
		16K	tape	5.7	384	8.8	398.5
		16K	2 drums	5.7	0.070	2.2	7.97
Case No. 2 42K Storage Requirement	Type No. 3	16K	tape	9.0	576	20.8	605.8
		32K	tape	9.0	384	8.0	401.0
		16K	4 drums	9.0	0.140	5.2	14.34
		32K	2 drums	9.0	0.070	2.0	11.07
Case No. 3 102K Storage Requirement	Type No. 3	16K	tape	22.9	1344	72	1438.9
		32K	tape	22.9	768	56	846.9
		16K	11 drums	22.9	0.385	18	41.285
		32K	9 drums	22.9	0.315	14	37.215

TABLE 5-4 (PART IV). TOTAL PROCESSING TIMES

STORAGE REQUIREMENTS	COMPUTING DEVICE	CORE STORAGE	AUXILIARY STORAGE	PROCESSING TIME IN CORE (SECONDS)	ADDITIONAL PROCESSING TIME		TOTAL PROCESSING TIME (SECONDS)
					Access Time	Read/Write Time (Seconds)	
Case No. 1 27K Storage Requirement	Type No. 4	32K	none	45.7	none		45.7
		8K	tape	45.7	384	3.8	433.5
		8K	2 drums 1 - 8K 1 - 16K	45.7	0.0255	3.8	49.5255
Case No. 2 42K Storage Requirement	Type No. 4	8K	tape	73.0	576	6.8	655.8
		32K	tape	73.0	192	2.0	267.0
		8K	3 drums 3 - 16K	73.0	0.051	6.8	79.851
		32K	16K drum	73.0	0.0085	2.0	75.0085
Case No. 3 102K Storage Requirement	Type No. 4	8K	tape	183	1248	18.8	1449.8
		32K	tape	183	384	14	581.0
		8K	6 drums 6 - 16K	183	0.101	18.8	201.901
		32K	5 drums 1 - 8K 4 - 16K	183	0.0765	14	197.0765

D. Analysis of the Configurations

1. Input Restrictions

Prior to selecting equipment for a system, the functions to be performed by the system must be analyzed. These functions may be performed by man, by a combination of man and machine, and by a combination of man, machine, and techniques, as illustrated in Figure 5-2, the basic system concept. The human function is to create, integrate, organize, interrelate, sift, and condense information as well as to disseminate it. Man can be assisted in some of these functions by machines and techniques that are able to perform such tasks as integrating, organizing, interrelating, sifting, and condensing data. (U)

Abstracting and keypunching are two tasks that require a human operator. Abstracting is a slow process (approximately 30 pages of abstracted material per day per person). By including this function as an input task, an immediate bottleneck occurs. Although information retrieval systems have been developed to abstract documents (by using information culled from the contents of a document), such techniques might require additional processing equipment. However, if the documents used as input into the Fact Correlation System are concise, it is feasible to read them into the system in their entirety rather than to abstract them. (U)

The keypunching process requires a human operator for each piece of equipment. The process is extremely slow-roughly two cards (160 characters or less) per minute. In addition, keypunched data must be verified and corrected, and the cards must still be transcribed to

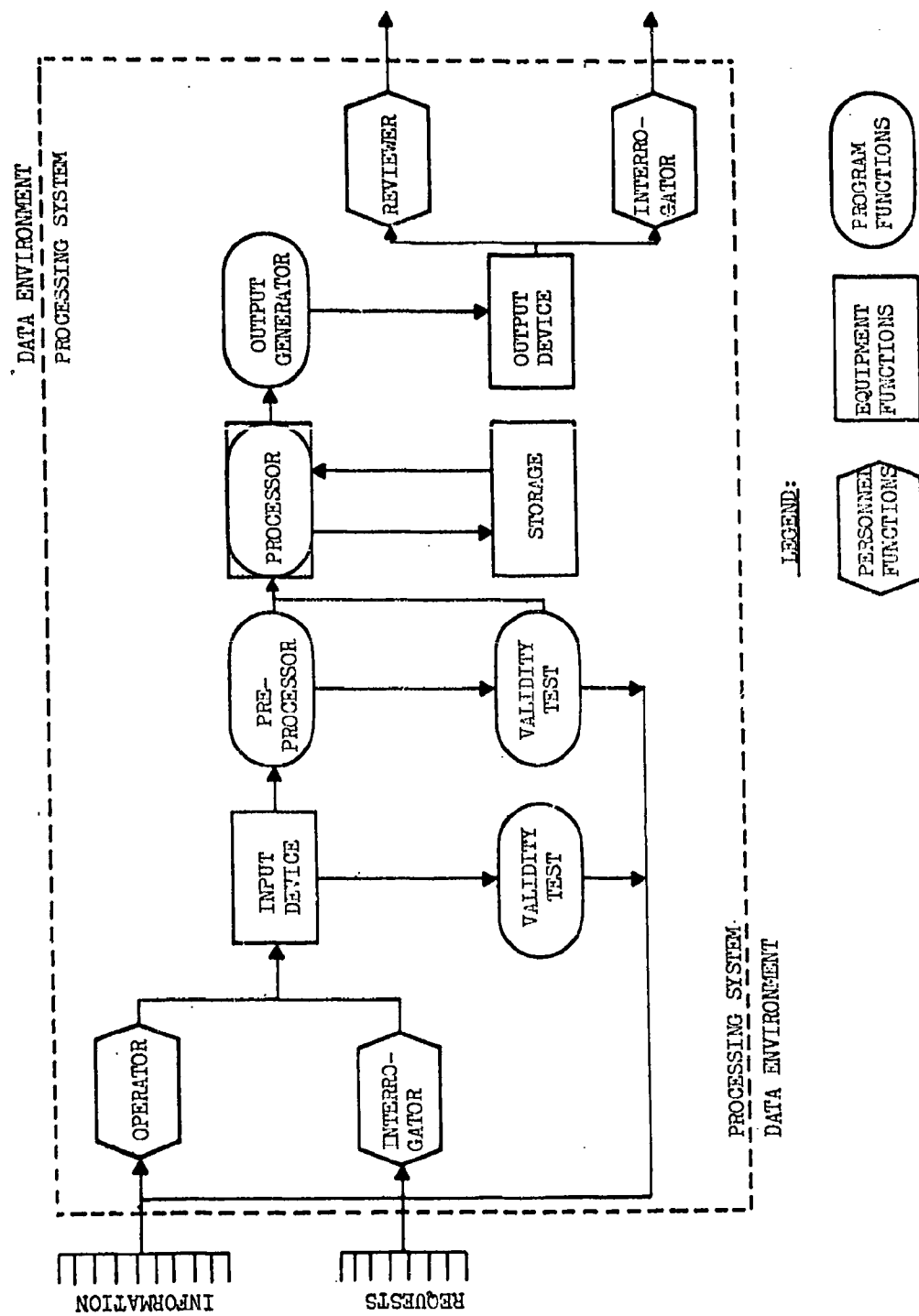


FIGURE 5-2. Basic System Concept

magnetic tape or read into a computer directly. When keypunching is included as part of the preparation of input data, the over-all processing time for the input averages approximately ten minutes per document, as will be illustrated in the configurations. (U)

2. Basic Computer Processing Requirements and Limits

The high-speed computers available today are more than adequate for most processing requirements; they are limited not by their own speed and logic, but rather by the speed and capacity of the input, output, and storage devices required for an operational system. Processing speeds in microseconds (and sometimes in nanoseconds) and developments in parallel operations, real-time processing, and multiple addressing are but a few of the recent advancements in digital computers. In addition, more sophisticated programming techniques have increased the efficiency and extended the range of applications of high-speed computers. In designing or selecting a data processing system, therefore, it is sufficient to choose one from among the many available digital computers having essentially similar or comparable capabilities. (U)

(a) General Characteristics - The high-speed, general-purpose digital computers being considered in this study will have the following general characteristics:

- (1) Parallel binary logic
- (2) Parallel processing
- (3) Transistorized circuits
- (4) Indirect addressing

- (5) Automatic indexing
- (6) Automatic interrupt
- (7) Multiple input-output operations
- (8) Real-time operational capability (U)

(b) Processing Speeds - The computing devices considered in this study have a basic cycle time in microseconds with add times ranging from 4 to about 36 microseconds. A typical computing device in the 4 microsecond range has the following capabilities for fixed-point operations:

- (1) Addition or subtraction--4.36 microseconds (229,000 operations per second).
- (2) Multiplication--4.36 to 30.52 microseconds (average rate of 39,500 operations per second).
- (3) Division--6.54 to 30.52 microseconds (average rate of 32,700 operations per second).
- (4) Logical operations--2.18 or 4.36 microseconds (458,000 or 229,000 operations per second).

A typical computing device in the 36 microsecond range may be characterized as follows for fixed-point operations:

- (1) Addition or subtraction--36 microseconds.
- (2) Multiplication--80 microseconds.
- (3) Division--128 microseconds. (U)

(c) Internal Data Representation - The representation of data within the computer varies. One type of machine may have single address instructions with word lengths of 26 binary digits (bits) and over 200 different operation codes. Another computer may have a 28-bit word, multiple addressing, and less than 50 basic instructions. These differences can be balanced by the techniques employed in programming the

various computers; however, computers with longer words are generally more efficient, especially in processing linguistic data. (U)

3. Elimination of Unfeasible Equipment Combinations

Over 1000 configurations could be combined from the individual equipments listed in Tables 5-1 and 5-4. Therefore, it is expedient to reduce the number of configurations to a practical number while still giving a representative concept of various possible systems and their limitations. A review of the tables of data reveals that certain cases may be eliminated for various reasons. (U)

(a) Table 5-1 - The abstracting function performed by human beings was eliminated from the configurations because this process required excessive time and personnel as compared with all other processes and equipments. Even the slowest computer would be kept idle for significant periods of time while waiting to receive documents abstracted at the rate of one every 48 minutes. (U)

(b) Table 5-4 - All cases based upon core storage without any additional storage medium were eliminated, since the largest core storage, 64K, would not satisfy the storage requirement of Case No. 3 (102K). (U)

4. Possible System Configurations

By combining the equipments and functions of Table 5-1, it is possible to obtain a number of different input configurations. The following combinations have been selected for consideration:

- (1) Keypunching, Card Reader No. 1 and Tape No. 1 or No. 2.
Keypunching, Card Reader No. 2 and Tape No. 1 or No. 2.
Keypunching, Card Reader No. 3 and Tape No. 1 or No. 2.
Keypunching, Card Reader No. 4 and Tape No. 1 or No. 2.
- (2) Optical Scanner and Tape No. 1.
Optical Scanner and Tape No. 2.
- (3) Reading Machine and Tape No. 1 or No. 2.
(1.6 second)
- (4) Reading Machine and Tape No. 1 or No. 2.
(1.0 second)
- (5) Reading Machine as direct input to computer.

The possible processing configurations selected for consideration are as follows:

(1)	Type No. 1: 16K plus tape	Type No. 3: 16K file tape
	Type No. 2: 16K plus tape	Type No. 4: 8K file tape
(2)	Type No. 1: 16K plus disc	Type No. 3: 16K file drums
	Type No. 2: 16K plus disc	Type No. 4: 8K file drums
(3)	Type No. 1: 32K plus tape	Type No. 3: 32K file tape
	Type No. 2: 32K plus tape	Type No. 4: 32K file tape
(4)	Type No. 1: 32K plus disc	Type No. 3: 32K file drums
	Type No. 2: 32K plus disc	Type No. 4: 32K file drums
(5)	Type No. 1: 64K plus tape	
	Type No. 2: 64K plus tape	(U)

5. Interpretation of the Configurations

An input configuration was combined with various processing configurations as shown in Figures 5-3 through 5-10. Three reception rates for documents are indicated in the diagrams together with the corresponding number of equipments required for each case. The time to handle a single

document is invariant regardless of the number of equipment used. The type of processor is shown with three processing times corresponding to programs of 25,000, 40,000, and 100,000 instructions (see Section C.2). Each processing configuration consists of the four selected types of processors plus a combination of core and auxiliary storage. (U)

(a) System Configurations 1 to 4 - Keypunches, card readers, and magnetic tape units were used as input media in system configurations 1 to 4 illustrated in Figures 5-3 to 5-6.⁽²⁾ A card reader capable of processing 250 cards per minute was used in conjunction with a tape unit with a rate of 60,000 characters per second in system configurations 1 and 2. The total time required to process a single 10,000-character document is 533.5 seconds. (This calculation alternatively means that an input document is available for processing every 533.5 seconds.) Using a faster card reader (3000 cards per minute) and a faster tape unit (120,000 characters per second) in system configurations 3 and 4 resulted in a total input time of 502.88 seconds. These two input configurations were combined with processing configurations 1 and 3. The first processing configuration uses 16K core and tape for processor Types Nos. 1, 2, and 3 and 8K core plus tape for processor Type No. 4. The second processing configuration uses 32K core and tape for all 4 types of processors. (U)

(2) System configurations include combinations of input and processing configurations. In the following discussion the system configuration numbers corresponds to the enumeration of the block diagrams. The numbers for processing configurations reference the lists in Section D.4. The input configurations are considered as entities within each diagram.

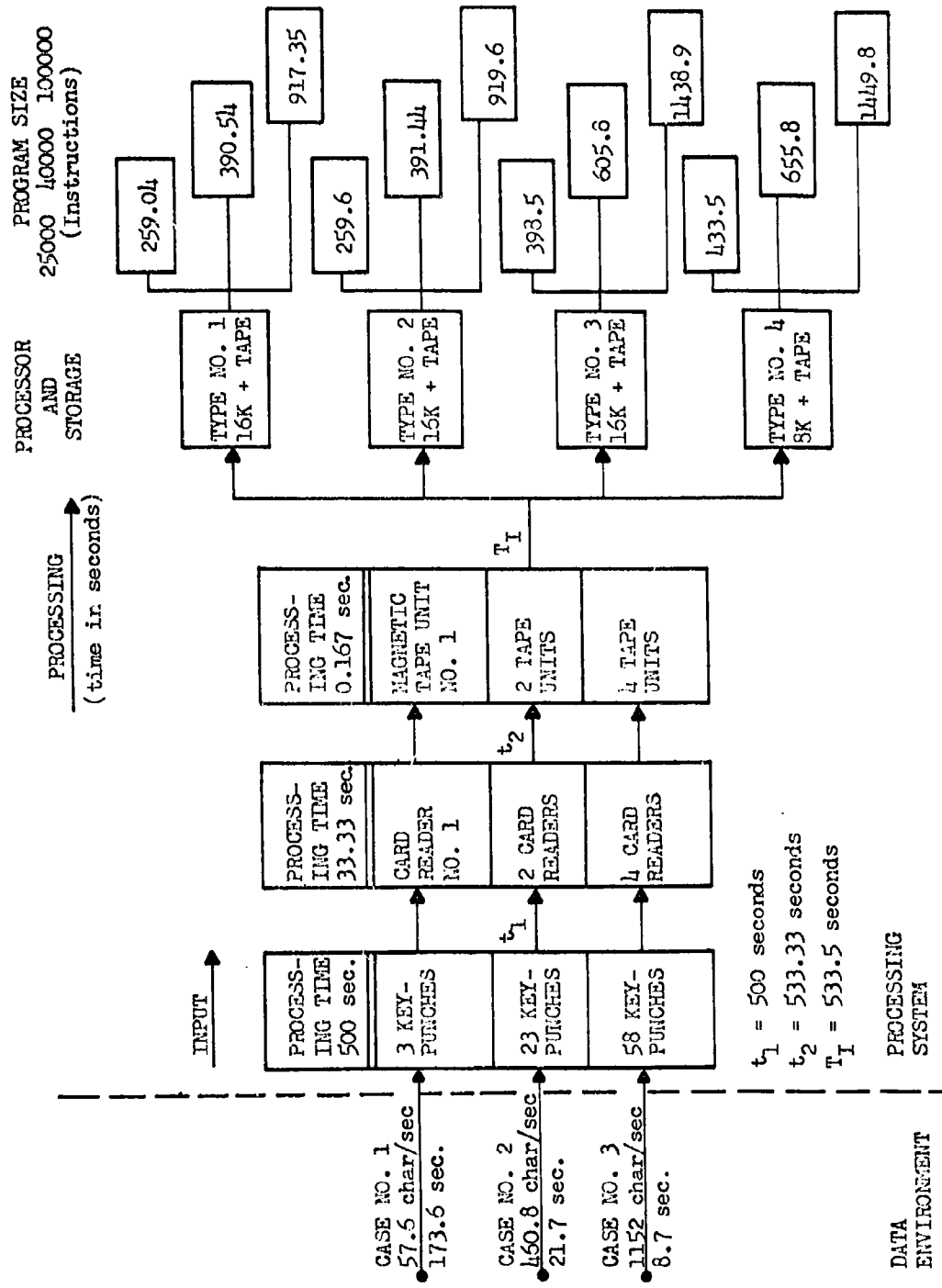


FIGURE 5-3. System Configuration 1

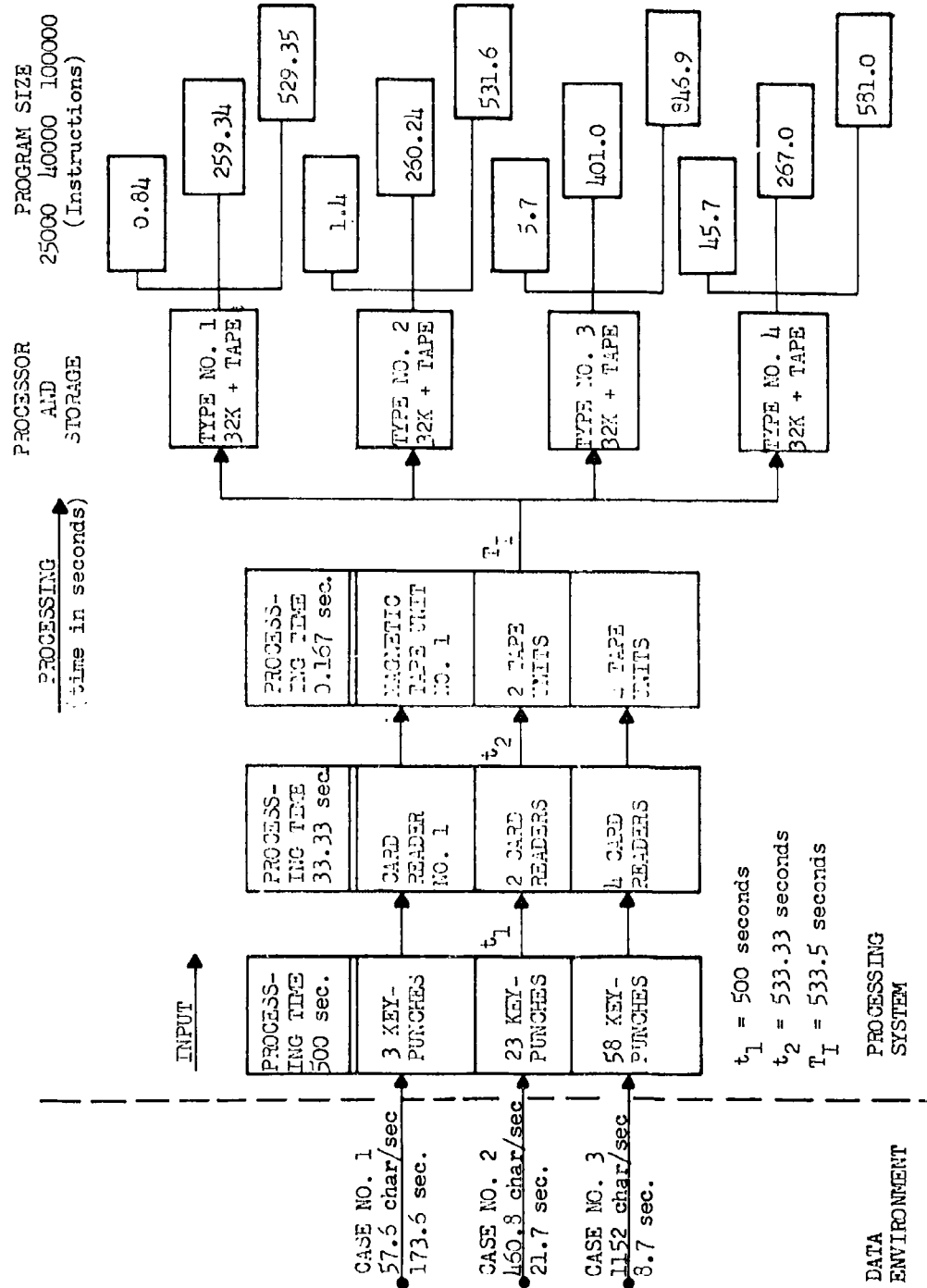


FIGURE 5-4. System Configuration 2

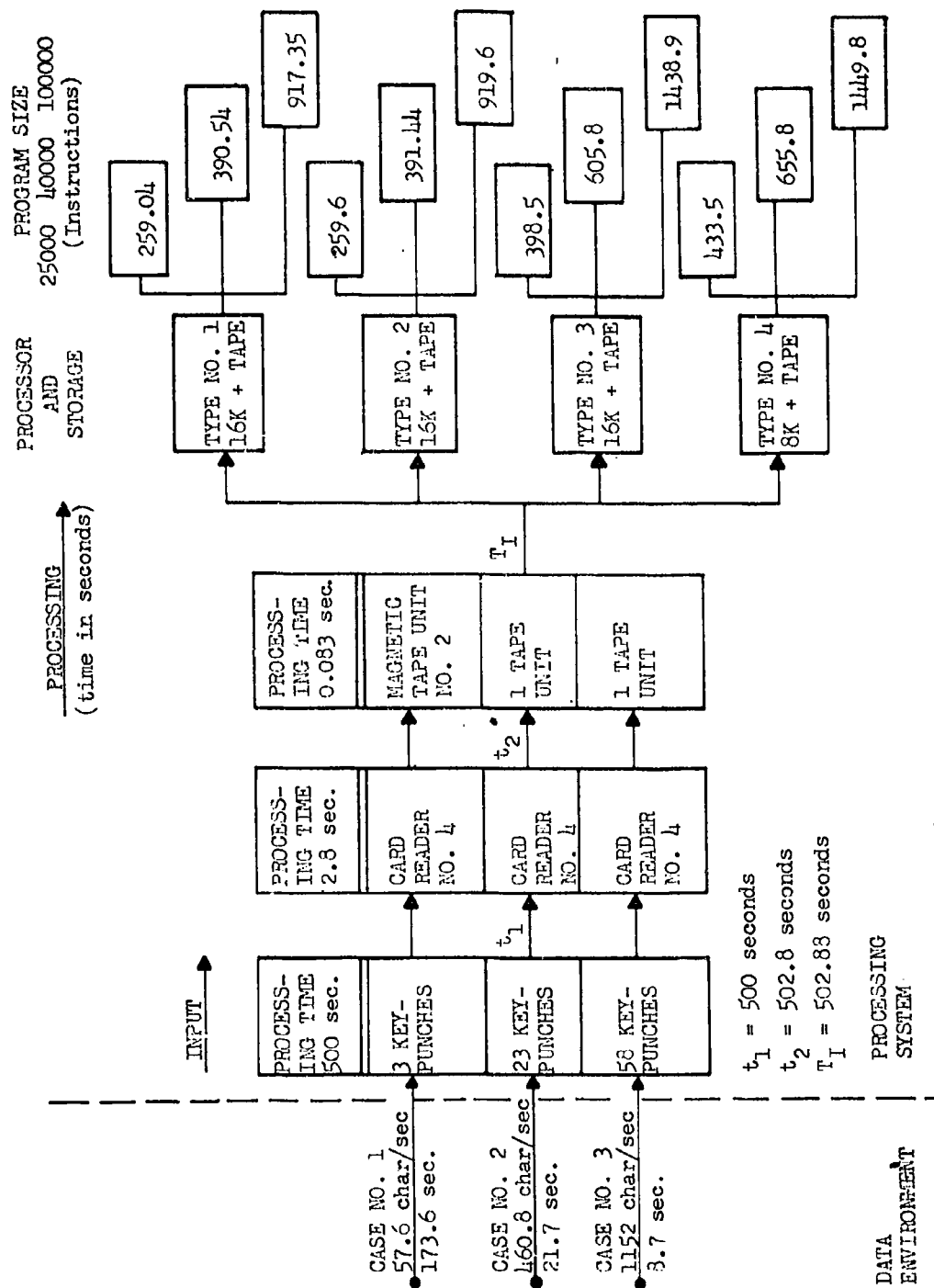


FIGURE 5-5. System Configuration 3

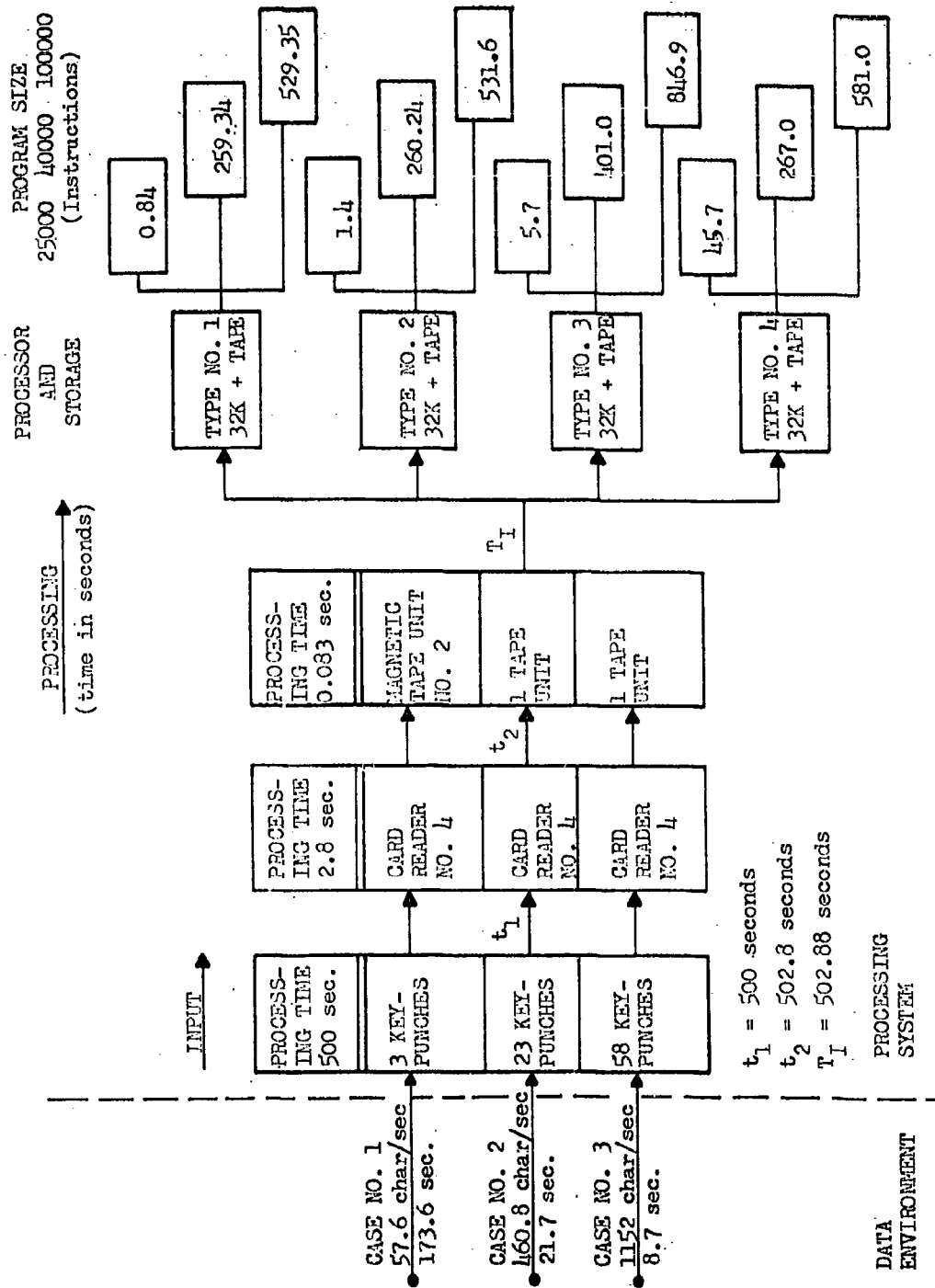


FIGURE 5-6. System Configuration 4

In system configurations 1 and 3, the processing times and input times are related as follows:

- (1) For processor Types Nos. 1 and 2 the computer would be idle approximately 50 per cent of the time for a 25,000 instruction program, 20 per cent of the time for a 40,000 instruction program; the computer would be unable to handle input data at the rate of every 503 or 533 seconds if a 100,000 instruction program were used.
- (2) For processor Types Nos. 3 and 4 the computer would be idle approximately 20 per cent of the time for a 25,000 instruction program; the computer would be incapable of handling the larger programs at the prescribed rate. (U)

In system configurations 2 and 4, the processing times and input times are related as follows:

- (1) For a 25,000 instruction program, the processing times are small (0.84 to 45.7 seconds) because the amount of core storage (32K) is sufficient to handle the program without the use of tape. Hence, for all four types of processors the idle time would be so great as to be impractical.
- (2) For a 40,000 instruction program for processor Types Nos. 1, 2, and 4 the computer would be idle approximately 50 per cent of the time; for processor Type 3, it would be idle about 20 per cent of the time.
- (3) The 100,000 instruction program would be handled efficiently in these configurations by processor Types Nos. 1 and 2. For Types Nos. 3 and 4 the input rate would exceed the processing time. (U)

Processing configurations 2, 4, 5, and 6 are not illustrated in combination with this input configuration, since their processing times are from 2.5 to 500 times as fast as the input time; hence, the computer would be idle from 50 to 90 per cent of the time. Card readers Nos. 1 and 4 were selected for these input configurations, since they

provide upper and lower bounds to the input time for card reader. A tape unit with twice the speed (120,000 characters per second) would decrease the input time by approximately 0.08 second, a relatively insignificant amount; hence, it was not used in these system configurations. (U)

(b) System Configurations 5 and 6 - The input media for system configurations 5 and 6 shown in Figure 5-7 and 5-8, were an optical scanner and magnetic tape unit with a total input time of 180.17 seconds. Processing configurations 2 and 4 were used in combination with these devices. Time comparisons are as follows:

- (1) For system configuration 5 the three different size programs can be processed in less than the time required for input on all four types of processors, with the exception of the 100,000 instruction program on processor Type No. 4. This case would require approximately 22 seconds more processing time than input time. All these cases would require that the computer be busy only 3 to 40 per cent of the time.
- (2) For system configuration 6 the 25,000 instruction program requires core storage only. The two larger programs require disc or drum storage in addition to 32K core storage. The larger programs use only a small percentage of the computer time except for the 100,000 instruction program when run with processor Type No. 4. (U)

Processing configurations 1 and 3 were not illustrated in combination with the optical scanner and tape input configuration because the processing times for these configurations exceeds the input time per document; hence, the computer would be unable to handle the input at the prescribed rate of one document every 180 seconds. (U)

Processing configuration 6 was omitted because the processing speed was so great that the computer would be idle a high percentage of

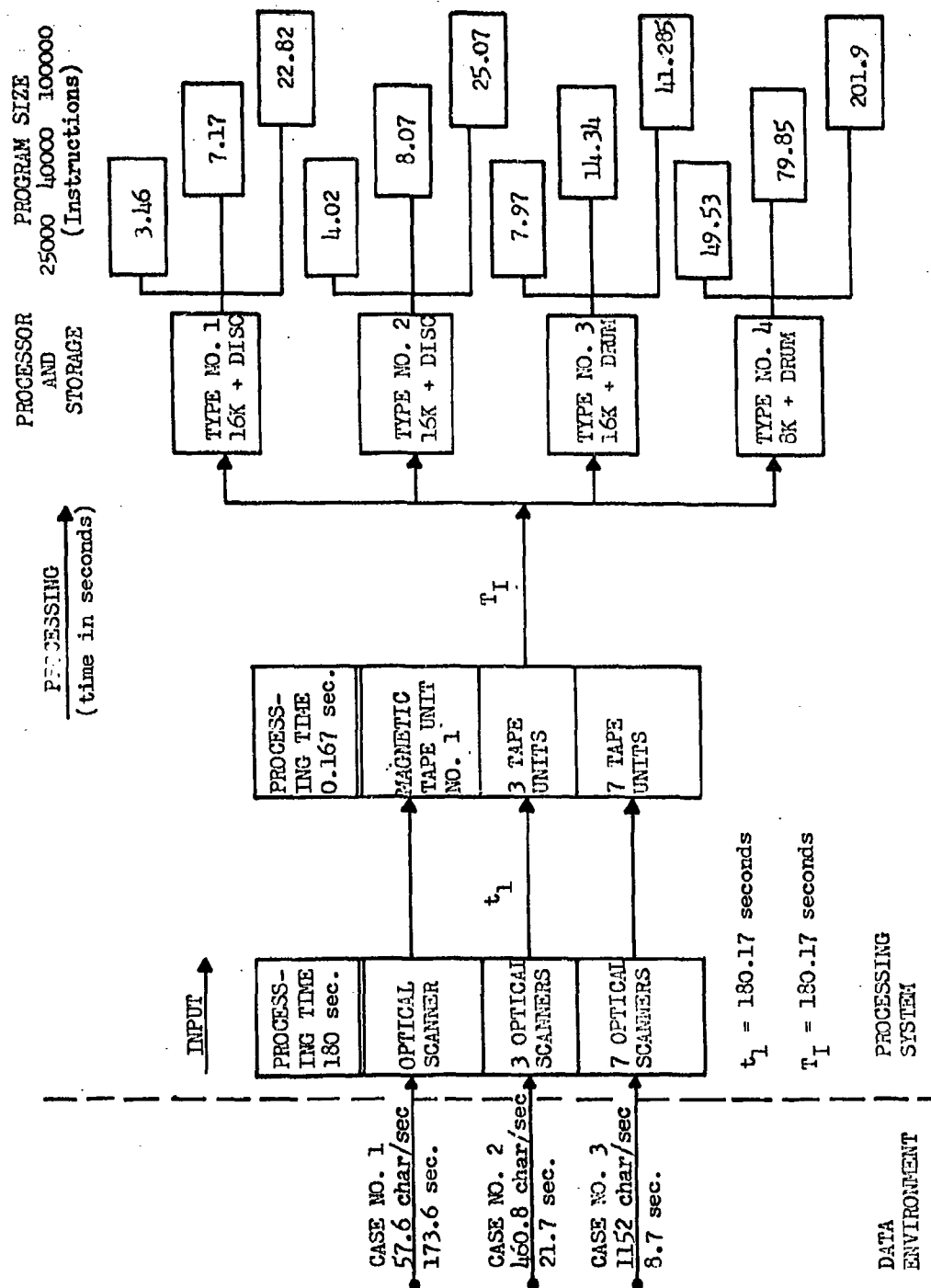


FIGURE 5-7. System Configuration 5

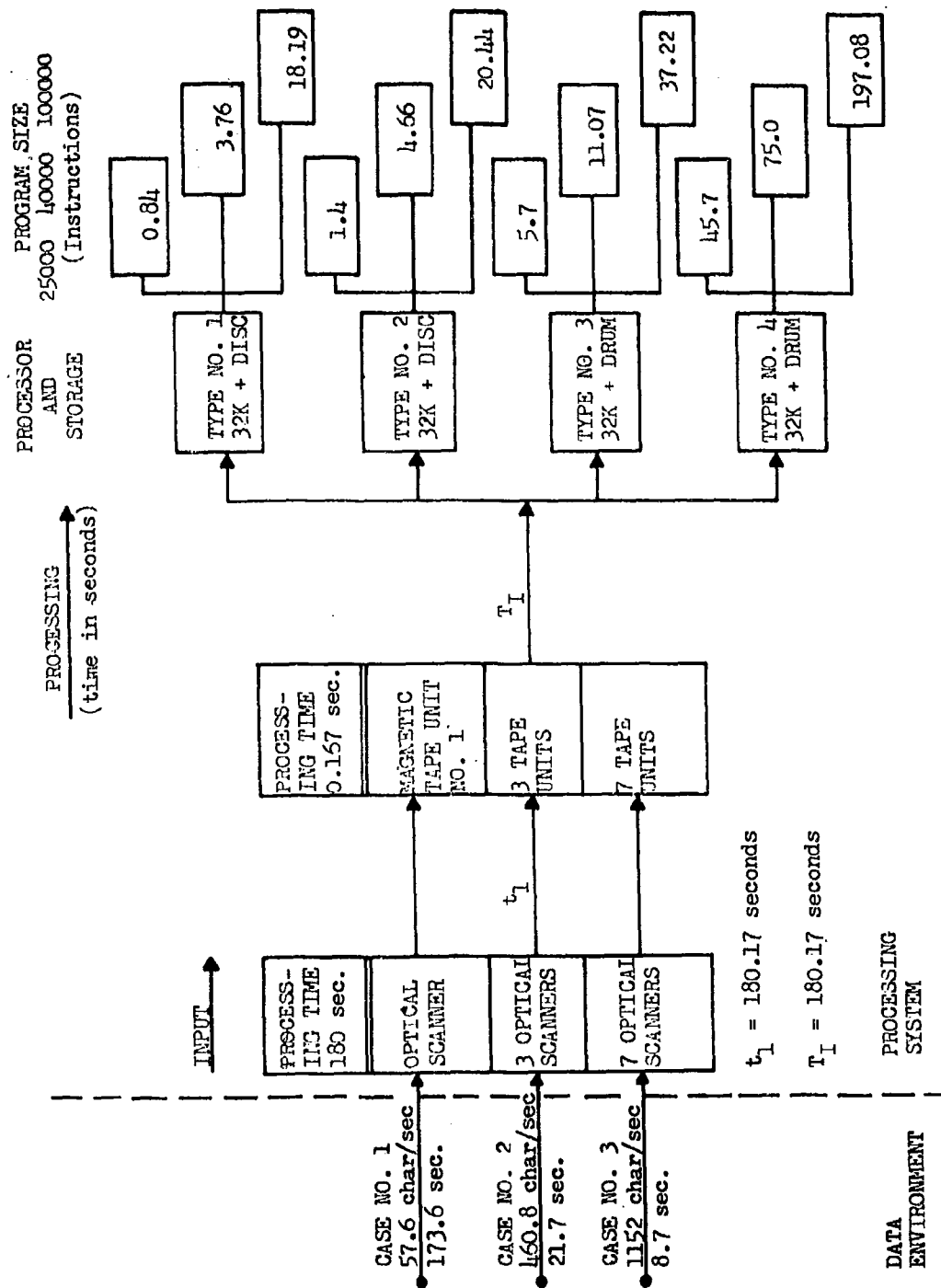


FIGURE 5-8. System Configuration 6

the time, even for the largest program. For processing configuration 5, only core storage would be needed in the first two cases, leaving the computer idle most of the time. With the 100,000 instruction program the computer would be unable to handle the input at the prescribed rate. (U)

(c) System Configurations 7 and 8 - A reading machine and magnetic tape unit were used as input devices in system configurations 7 and 8 shown in Figures 5-9 and 5-10. The reading machine processes a document and reads it on tape in 1.6 seconds. Adding the tape processing time to the input time gives a total input processing time of 1.77 seconds per document. Processing configuration 4 and 6 were combined with this input configuration with the following results:

- (1) In system configuration 7, 32K core storage was used with either drum or disc for auxiliary storage. For processor Types Nos. 1 and 2 the smallest program (25,000 instructions) handles the input efficiently. The larger programs require a longer processing time than input time, as do processor Types Nos. 3 and 4 with all three program sizes.
- (2) Configuration 8 requires 64K core storage plus disc storage, a combination appropriate only to the high-speed processor, Types Nos. 1 and 2. The 100,000 instruction program requires more processing than input time. The smaller programs work efficiently with this amount of core storage without additional disc storage. (U)

Processing configurations 1, 2, 3, and 5 were not illustrated in combination with these input devices because an excessive amount of processing time is required for all three program sizes. Processing configuration 5 is similar to processing configuration 6; it requires only core storage for the smaller programs and is incapable of handling the input at the prescribed rate with a 100,000 instruction program. (U)

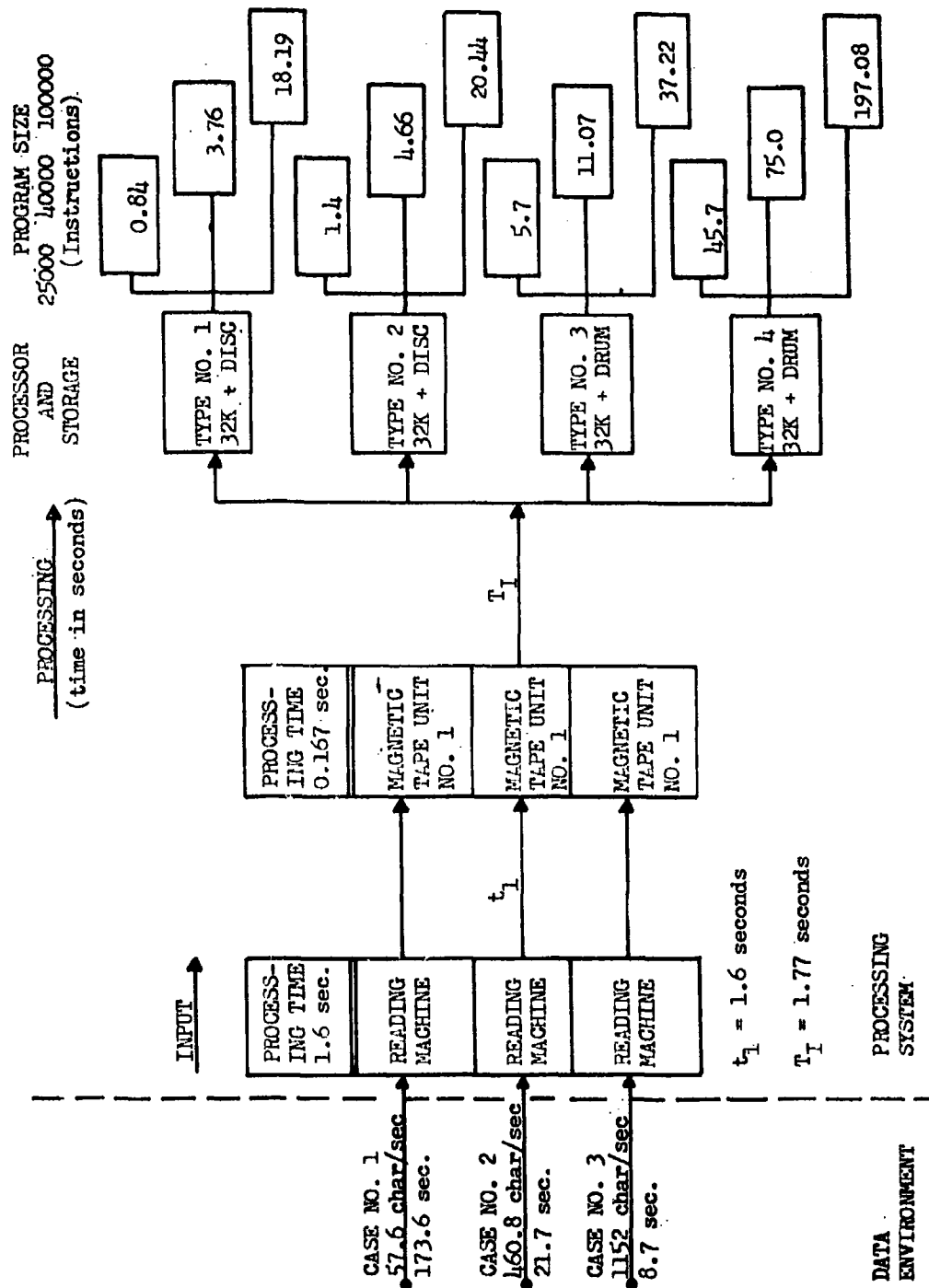


FIGURE 5-9. System Configuration 7

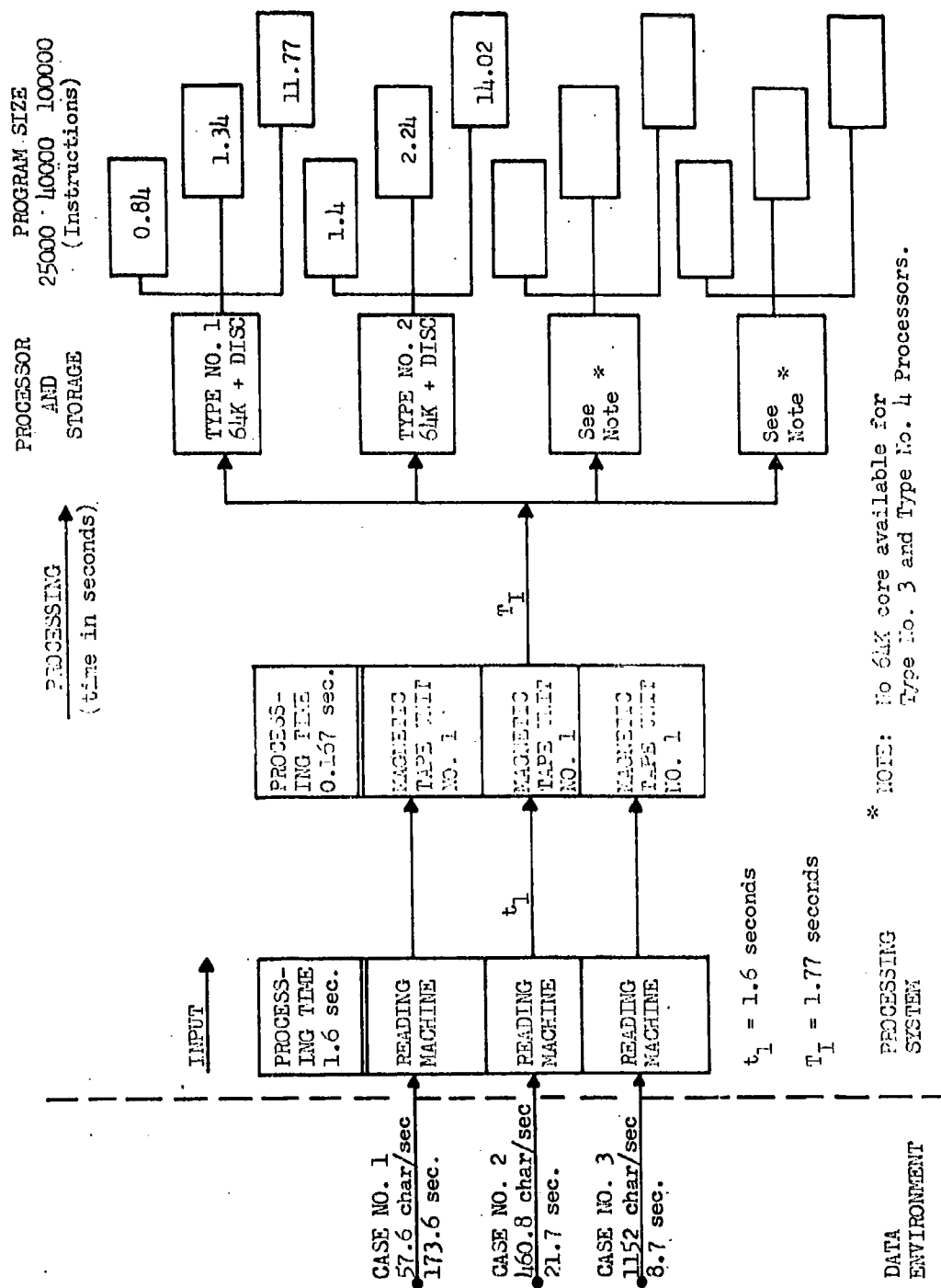


FIGURE 5-10. System Configuration 8

6. Evaluation of System Configurations

In evaluating the system configurations there are several points to consider. First, a balance must be maintained between the input and processing rates within a system. If the input configuration is slow, as in the first four system configurations, it is inefficient to use a high-speed processor with high-speed access to stored information because the computer would be idle most of the time. It is similarly inefficient to enter information at rapid rates that the processor is incapable of handling. (U)

A secondary consideration is the amount of equipment required to handle the problem. For example, the first four system configurations illustrate the keypunching problem. In order to maintain a steady flow of input data, 23 or 58 keypunch machines are required for Cases Nos. 2 and 3, respectively. In calculating the cost of the input subsystem for the worst case, the price of 58 keypunches, 4 card readers, 4 tape units, and salaries for 58 keypunch operators must be included. When these factors are taken into account, the price of a single, comparatively inexpensive reading machine may be readily justified. (U)

The optical scanner and tape input combination used in system configurations 5 and 6 provides a considerable time saving over the keypunch and the card reader operations. When used with a low-speed computer and drum storage, this combination balances fairly well with the processing rate. However, for the fastest document reception rate, Case No. 3, seven optical scanners and seven tape units are required. (U)

CONFIDENTIAL

The reading machine used in the last two system configurations is the fastest means of entering input data. Although the reading machine is combined with a tape unit in system configurations 7 and 8, this machine can read information into the computer directly. A high-speed computer with high-speed storage access is required to balance the system. (U)

In all system configurations extra storage space is available for filing new information derived from each document. The use of magnetic tape units for auxiliary storage when the amount of core storage is inadequate tends to produce a bottleneck in processing because of the comparatively slow access time for tape. However, with parallel processing techniques, it is possible to overlap operations and reduce the time during which the computer waits for information to be read from tape. Disc files and drum storage provide faster access; but their cost is higher, and their capacity per unit is lower. Disc files appear to be inherently less reliable because of their large number of mechanical parts. (U)

(a) Recommended System Configurations - The system configurations illustrated in Figures 5-3 through 5-10 lead to the conclusion that the optimum state-of-the-art configuration requires a reading machine for input, a high-speed processor, a large core storage capacity, and high-speed access auxiliary storage. Figure 5-11 illustrates such a configuration based upon a high-speed processor with an arithmetic-type instruction (Type No. 1), a 64K core storage, and a disc storage unit to provide maximum processing speed with a minimum of input time. (C)

(b) Constraints and Limitations - A reading machine is the

CONFIDENTIAL

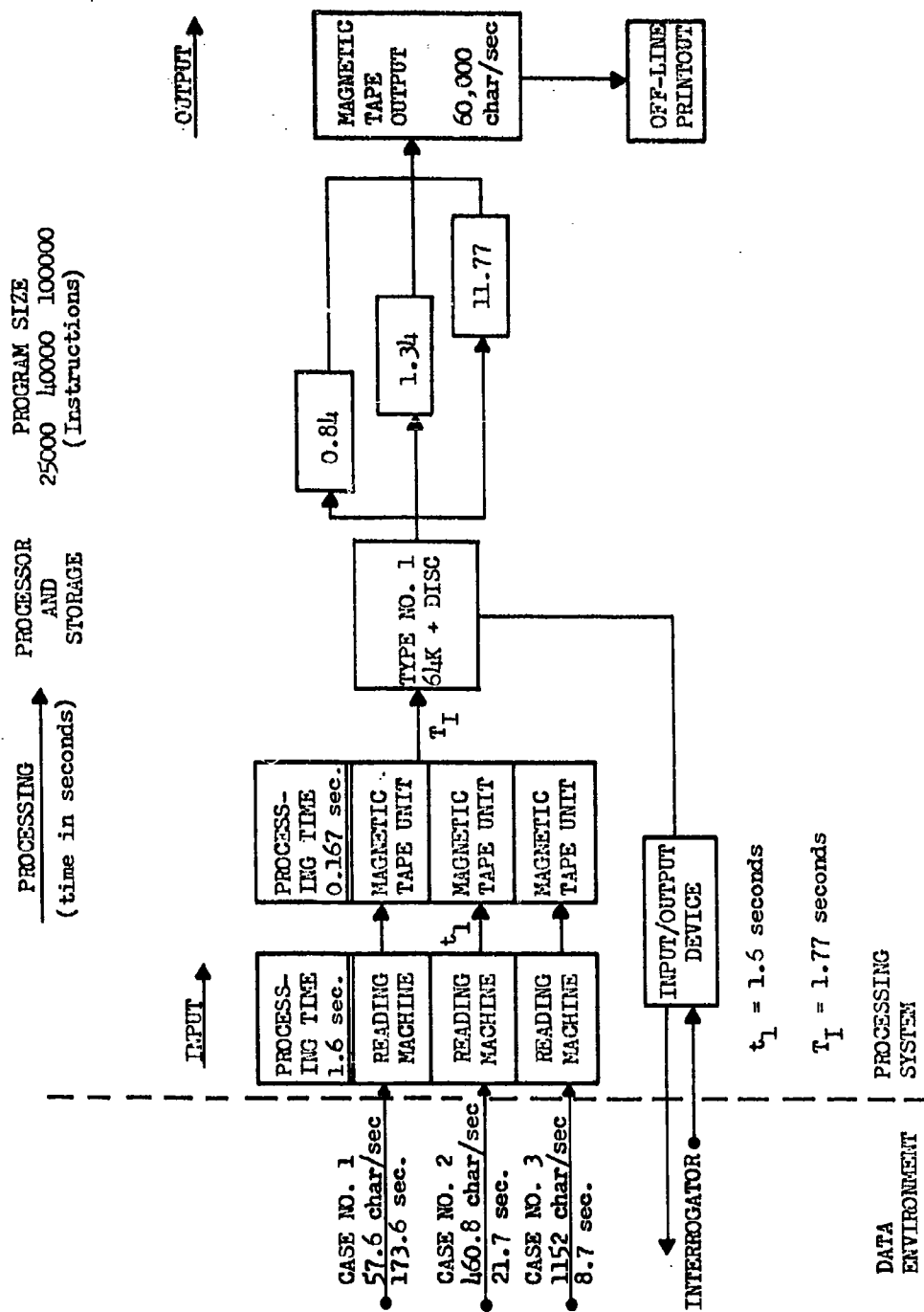


FIGURE 5-11. Recommended State-of-the-Art Configuration

CONFIDENTIAL

fastest means for entering input data into the system. Although such devices have been developed to operate at the indicated speeds, there are several limitations in using these devices. The reading machines are based upon character recognition techniques that presently require a relatively high quality printed document to ensure accurate reading. If documents are rejected or errors noted for correction by the machine or subsequent processing, the input speed will be reduced considerably. Although a device capable of reading documents and transforming them into input suitable for a computer is ideal, only continued development will produce a reading machine that is highly reliable and capable of handling all types of printed documents with a minimum of errors and rejections. (C)

The graph in Figure 5-12 depicts the storage capacity of different types of storage devices versus access time. Core storage access time is in the microsecond range as compared to the slower access times of drums, disc files, and magnetic tapes; however, disc files and magnetic tapes have a greater storage capacity than drums. This limitation indicates a need for developing internal storage devices with greater capacity. Research in thin film memories, modulated coupling devices, and similar components may provide a solution to this problem. A large high-speed random-access internal memory for the computer would help to eliminate the need for auxiliary storage devices. (U)

(c) The Question of Output - The eight configurations that have been illustrated did not show the means for output. In systems where the volume of output is great, the output data is generally written on magnetic tape to be processed off-line by a printer or by a microfilm

CONFIDENTIAL

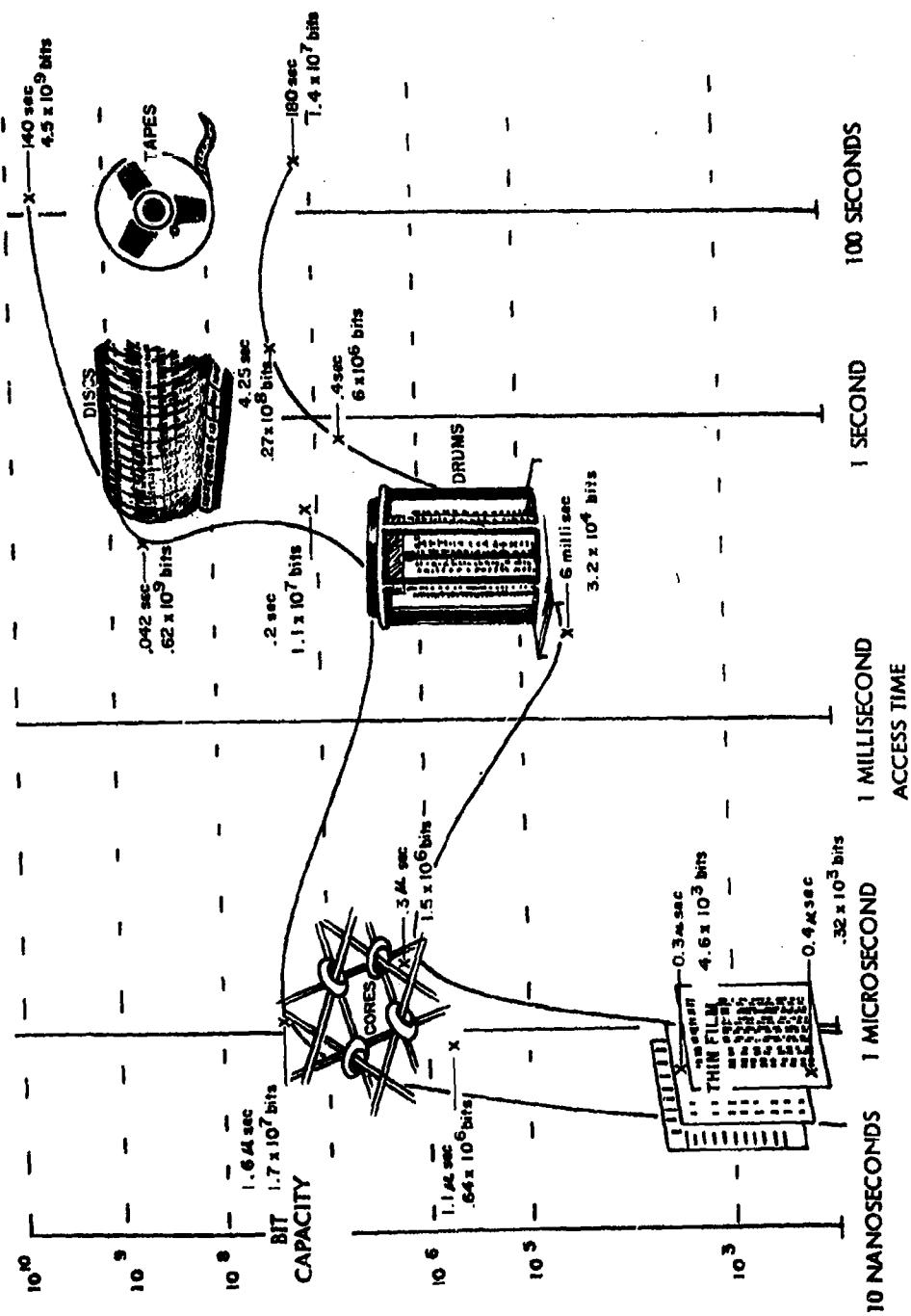


FIGURE 5-12. Storage Capacity Versus Access Time

processor. It has been assumed that output will not equal input in volume or in rate so that off-line processing of output from magnetic tape will not interfere with the operational efficiency of the system. (U)

An intermediate input-output device will be required in a Fact Correlation System to allow an analyst to interrogate the system or the system to interrogate an analyst when it has a problem to resolve. This device can be a keyboard and an on-line printer, a cathode-ray display system, or a direct display system capable of handling relatively small amounts of data at any one time. If such an inquiry becomes necessary during the processing of a document, a time lapse will occur while the processor asks for information from an analyst. This time lapse has not been considered as part of the document processing time because it is not an expected condition that will occur for each document. The length of time will also vary, depending upon the number of questions asked and the difficulty in resolving the problem that has arisen. (U)

E. Comparative Operations

The four phases of data processing--input, processing, storage and retrieval and output--are interdependent; therefore, any change in one phase affects the others. Computer users pay a high price when they do not design the processing system before selecting a computer. Computers have been selected because they had faster magnetic tapes, high arithmetic speeds, or faster printers than other computers, even though these features were really not germane to the processing application. The equipment must be considered in terms of its application as well as in terms of its relation to other equipment in the system. (U)

1. Input Devices

Data input is defined as the generation, collection, and transmittal of data to a computer for automatic processing. Since data input techniques and equipment are an integral part of a data processing system, the selection of input equipment to handle the data depends upon computer-related attributes--the volume of transactions processed, computer speed, time between computer-processing cycles, and the type of input data the computer can accept. Five basic considerations in selecting input devices are: the equipment reliability (measured in terms of available time and accuracy); volume of data to be handled by the system; input time requirements; equipment costs; and design features of the equipment. (U)

The development of devices that automatically read documents by optical, electronic, or magnetic sensing improves data input recording. Typed documents have in the past been unusable as computer inputs, although they provided information in clear readable form. Consequently, personnel have had to read the typewritten, printed, or handwritten documents, abstract the important information from them, and reproduce this information on punched cards or punched tape, often introducing errors into the data in the process. However, with advances in the techniques of sense-reading, typed documents will ultimately be usable as direct input for the computer. (U)

(a) Reading Machines - One type of character recognition machine for converting typewritten or printed information into a form suitable for electronic data processing systems is the matrix comparison

reader. This machine uses an electronic pattern matching technique limited only by the speed of paper handling. No mechanical scanning or other mechanical features are involved. A block diagram of the matrix reader is shown in Figure 5-13. (U)

In the optimum pattern matching process shown in Figure 5-14 the output of a bank of photocells is sampled as follows:

- Step 1: - Documents are fed one by one past a high-speed paper transport under an optical head at constant speed. The optical head projects images of segments of a moving character to a bank of photo sensitive cells.
- Step 2: - Preamplifiers boost the electrical signals and four level quantizers detect white from three densities of black.
- Step 3: - The "first black" signal from the quantizers stimulates a sampling time signal from control logic circuits. Samples are taken sequentially as the character moves past the optical head.
- Step 4: - The storage matrix receives signals indicating the sampling of a character has been completed and the shift register moves the memorized image to the lower edge.
- Step 5: - Recognition circuits select the best voltage match and determine the character. The recognized character is transferred in coded form to buffer storage and then transferred to magnetic tape, punched paper tape, punched card, or computer memory.

The recognition circuits electronically compare with the storage matrix master matrices consisting of an individual matrix for each character. Since comparisons are performed in parallel, the process is extremely fast; the reader operates at instantaneous reading rates of 10,000 characters per second, although the effective reading rate is somewhat lower. (U)

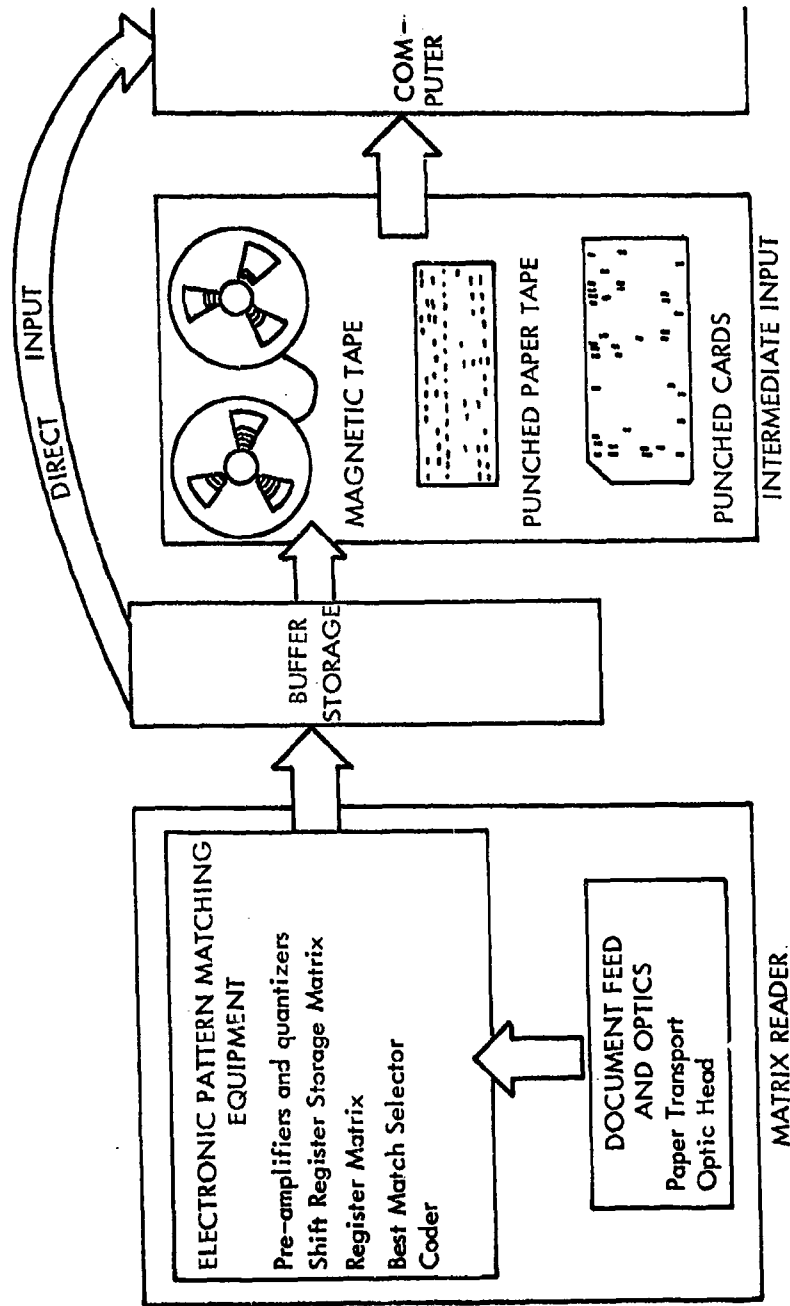


FIGURE 5-13. Block Diagram of Matrix Reader

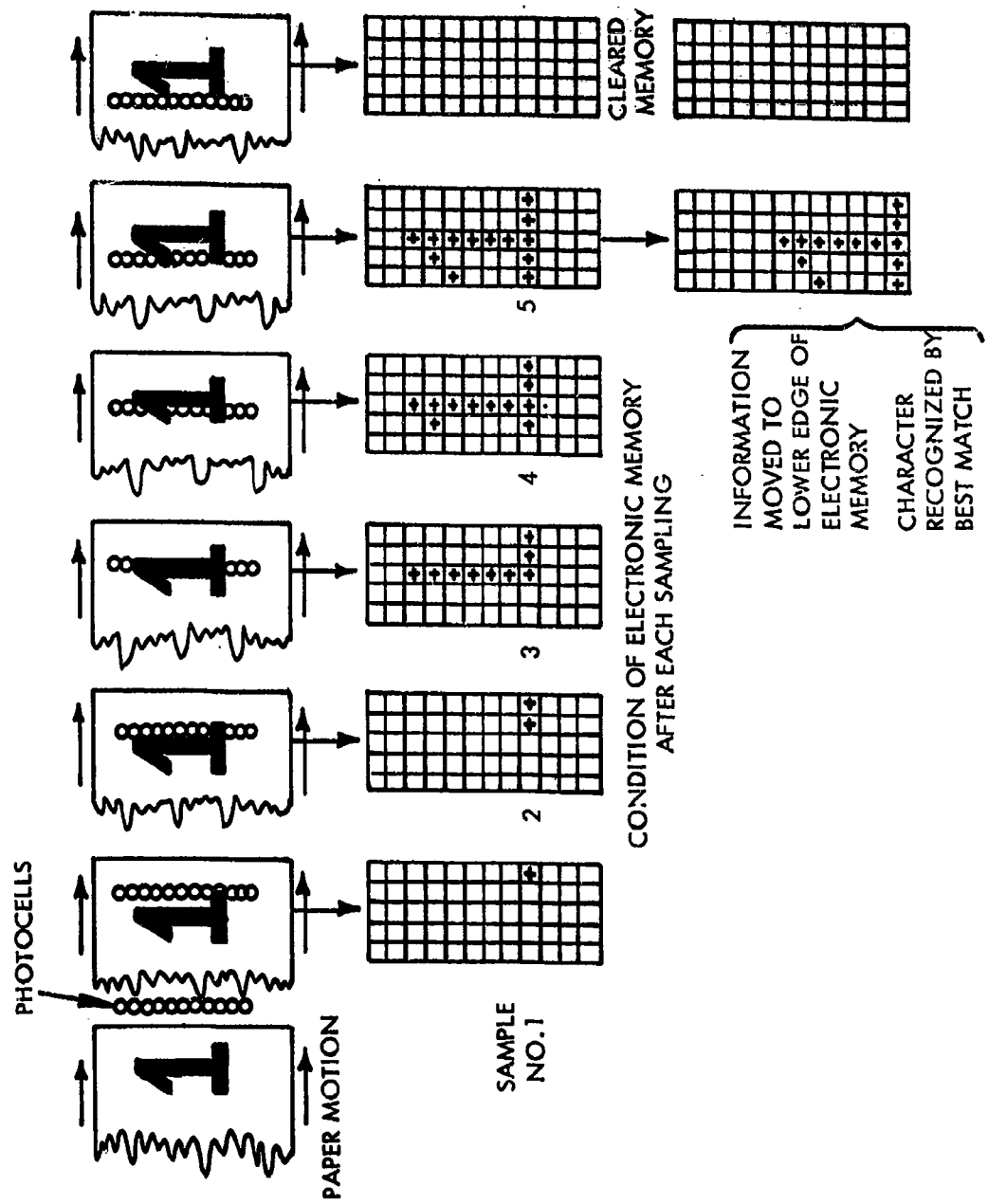


FIGURE 5-14. Character Recognition Sequence

The reading machine is flexible. Various font styles and sizes, including common typewritten upper- and lower-case or printed fonts, can be accommodated simultaneously since the master matrices are custom manufactured for each machine. Single-spaced elite type of newsprint can be read provided the characters do not touch. It is possible to accommodate more than one alphabet, if this degree of flexibility were desired. For example, documents printed in Russian could be read by a matrix reader with the characters in the Cyrillic alphabet. (U)

The resolving power and logic built into the machine is such that the reading of relatively poor reproductions, such as carbon copies and relatively imperfect print, is possible. Some smudging or minor defects in the original characters are tolerable. The machine does not recognize the character "A" because it meets certain specific requirements, but because it meets the characteristics of the character "A" better than the characteristics of any other letter or number. Thus an imperfect character may still be the best match. (U)

Reliability is a function of the quality of print read by the machine. Machines have been built to read typewritten information with reject rates of less than 1 character in 10,000 and error rates of less than 1 character in 100,000. There are a number of techniques that may be built into the machine to increase its reliability. Two independent comparisons for each character can check the character successively positioned at the bottom and the top of the storage matrix for each character. (U)

Additional requirements can be imposed to verify the reliability

of the machines. For example, if the two highest voltages are too close to each other, the observed character can be rejected as uncertain. By adjusting the relative values for acceptance or rejection, the error rate can be decreased by simply increasing the proportion of rejected characters-- i.e., by using a more stringent criteria for acceptance of a character. The choice of error rates would depend upon the font being read, the significance of errors in the material, and the cost of reprocessing rejects. A mark indicating the rejected line can be printed on the document, so that the error can be quickly located by an operator; then a correction can be sent to the system. (U)

Other procedures may also be used to improve the reliability of the recognition system. For example, a standard pre-printed alphabet can be read between every two documents, thereby checking the machine's accuracy before beginning the next document. This check alphabet can be printed in both heavy and light density ink so that the tolerance of the machine can be continuously checked against itself. (U)

The redundancy of information in a document can be used to increase accuracy. If a field is numeric, the alphabetic matrices can be temporarily disabled so that only 10 voltages have to be compared. This procedure appreciably decreases both the reject and error probabilities. It is possible to program the recognition procedure so that errors occurring in standard or unique lines of a text can be corrected automatically. (U)

The reading machines described here are constantly being improved

and developed. A machine capable of reading 10,000 characters per second has been built but is not in production. The readers are generally built to suit an individual user's requirements. A more complex machine, such as one that reads alpha-numeric information in both upper and lower case type, might require as much as a year for delivery. (U)

(b) Optical Scanners - Optical scanners employ a high-speed, high-resolution electromechanical scanner to sweep over the data image just as a television camera scans a picture image. Each letter or number is scanned many times, with each scan representing a vertical slice of the character. A photo-multiplier converts each scan into video pulses. A complete set of video pulses contains all the information about a letter or number in electronic form. The video pulses are analyzed to identify the character and to activate a punching or magnetic recording device. The character, is resolved into a bar code; the relative length, location, and orientation of the bars are the significant parameters. Each character differs from any other character by at least two bars so that the chance of errors is small. The code is simple enough; defective or partially obliterated characters are often repaired with pencil and eraser. (U)

Optical scanners read numbers, capital letters, common punctuation marks and special symbols. Scanners have been built to read mixed cases, but not as a standard item. A variety of type fonts may be read; however, the scanner is not able to read multiple fonts. A special type font, especially suited to optical scanning systems, helps to compensate for the wide variety of print quality in the ordinary data source. This

type font increases the reliability and efficiency of optical scanning systems. (U)

Optical scanners are considerably slower than the matrix reader. Data is read at a rate of six 32-character lines per second or two and one-half full lines per second. Ordinary documents can be read and punched on paper tape at the rate of 240 characters per second; this speed can be increased to 300 characters per second if magnetic tape is used for recording. (U)

In addition to the basic function of reading and transforming characters, optical scanners may have other capabilities that contribute to their usefulness in a data processing system:

- (1) Accumulating--the scanner can add and subtract, recognize signs, and produce algebraic totals, both positive and negative. Totals can be transferred to the output record, printed on a lister, or both.
- (2) List printing--to monitor batch detail and summary information list, printing can be provided on continuous tape with information supplied from scanned or accumulated data.
- (3) Batch heading--to identify every output record, information read from header documents can be included in every output record resulting from the reading of all documents in that batch.
- (4) Mark scanning--preprinted mark guides on a document, in pen or pencil, can be optically read and converted to machine language.
- (5) Programming--flexibility in reading and output formats can be achieved through programming. Wired control panels select the program steps, each of which performs a function such as reading and storing a digit, filling zeros in storage, or locating and testing a check digit.

Presently available models of optical scanners read upper case letters,

numerals, and common punctuation marks and symbols. Their repertoire could be extended to include lower case letters and exotic alphabets. (U)

(c) Card Readers - The majority of card readers are electro-mechanical devices. Cards are placed in a hopper, mechanically moved through a card feed unit under reading brushes that sense the information punched on the cards, and placed in a stacker. The electrical impulses sensed by the brushes are transferred to a buffer storage where they are translated and transmitted to magnetic tape units. Checking features compare the record read from the card against that transmitted to tape. Card readers operate at speeds ranging from 250 to 800 or more cards per minute. The higher speed card readers are sometimes used for direct input to a computer. (U)

A special card reader has been developed to fulfill the need for reading cards at speeds more closely matching those of some computers. This reader uses photo-sensing in both reading and timing operations. Photo-diodes time the reading cycle and provide a constant series of checks. Horizontal diodes in a circuit supply an output whenever a change from dark to light is detected; the vertical diodes supply an output only when changes from light to dark are detected. (U)

The card reader has a simple mechanical design so that there are no major maintenance problems. The cards are fed from a 4000-card hopper through the photo-sensing system and into a removable drawer. Cards can be read at speeds from 400 to 3000 cards per minute. The card reader may be used on-line with a computer or off-line for card-to-tape conversion. (U)

(d) Tape Units - The magnetic tape unit is an intermediary input device that requires another input device to record data on tapes. Tape units operate at speeds from 7000 or less to more than 360,000 characters per second; hence, tape units come closer to matching computer speeds than any other input device. However, since tape units require some other device for recording data, their use as input devices depends upon the combined character handling rate. (U)

Since magnetic tapes are also a prime storage device, they will be discussed further under the heading of Storage Devices. (See also Table 5-7 for a comparison of tape units and different kinds of tape.) (U)

2. Output Devices

(a) Printers - High-speed printers print 1000 or more full lines of alpha-numeric characters per minute and 2000 or more lines of numeric characters per minute. These printers are capable of both on-line and off-line performance. Lower speed printers have rates of 150 or 200 lines per minute and may prove entirely adequate in some applications where greater speeds are not required. (U)

Two basic types of printers are available: one uses a print roll or drum; the other, a wire printer. Both types of printer are capable of printing off-line from magnetic tape or on-line under computer control. The former type consists of a constantly revolving print roll or drum with engraved columns of characters. Print hammers are triggered by input pulses that convert the data source information into printing signals. An intermediate storage unit retains data required for at least

one complete print line until the program unit sends it to the print hammer drive circuits. The program unit controls the printing cycle by synchronizing the functioning of auxiliary components with the printer operation. (U)

The wire type printer receives data through a control and storage unit in sequential order from tape. Each character is printed as a pattern of dots formed by the ends of small wires arranged in a 5 by 7 rectangle. Character configuration is determined by the arrangement of the projecting wires terminating in the print head; selected wires are pressed against an inked fabric ribbon to print the characters on paper. The print unit consists of 30 to 60 print heads spaced four character positions apart. To print even lines, the unit moves from left to right four times in four subcycles; each head prints four adjacent characters. In printing odd line, the print unit moves through a reverse process. Printing in both directions conserves printing time. (U)

A combination high-speed printer and plotter has also been developed. This device receives information from magnetic tape and plots up to 6000 points per second in 10 simultaneous curves while printing annotations and drawing grid lines on output paper moving at 10 inches per second. If this device is only used as a high-speed printer, it is capable of printing 4000 lines per minute with 100 characters per line. (U)

(b) Microfilm - A microfilm system converts machine language to alpha-numeric characters on microfilm at speeds compatible with the computer itself. The system converts magnetic pulses into visible alpha-numeric or special characters and records them on microfilm at an average

recording rate of 20,000 characters per second. The system uses an electronic camera with 16-mm film with a capacity of 1000 feet of film per magazine. This amount of film can contain the decoded information from as many as 80 reels of 2400-foot magnetic tape. The microfilm images can be viewed on the face of a cathode ray tube, reproduced as a paper enlargement, or stored as microfilm records. The system operates off-line from magnetic tape or on-line under computer control. Microfilm output is especially advantageous in systems having a great volume of output data, as in information storage and retrieval applications. (U)

(c) Display Devices - Digital display devices are another means of presenting computer generated data. Displays may be generated on cathode ray tubes or projected on wall screens. In a typical cathode ray tube system, a keyboard on an input-output control console permits an operator to compose messages, correct typing errors, and transmit the corrected message to a computer, disc file, tape unit, or magnetic drum. The console contains a high-speed magnetic core memory from which the data is displayed on the scope. Symbol generation rates exceed 150,000 per second. (U)

Various display devices are capable of plotting points and drawing vectors as well as generating alpha-numeric characters of several different sizes. Although display devices are certainly a form of output device, they are actually much more, since they also have input and storage facilities. These devices are especially useful as on-line intermediary input-output devices in cases where information monitoring, interrogation,

and man-machine communications are an integral part of the data processing system. (U)

(d) Multiple Stylus Recorder - Another method of output from a computer or magnetic tape is the multiple stylus recorder. This unit is a facsimile recorder that marks moist electrolytically treated paper by applying current. The device has 1024 writing styli that press 11-inch wide paper against a reciprocating steel strip with paper moving at 5 or 10 inches per second. Currents are applied through the style to the bar. The electrolytic process removes metal from the bar and deposits it on the paper as dots 0.01 inch in diameter. (U)

3. Storage Devices

The functions of the storage unit are to receive digital data and control information, to retain this data unchanged, and to return this data to the remainder of the system upon request. In describing storage devices the following definitions apply:

- (a) Storage capacity--the number of available addresses for word storage.
- (b) Bit capacity--the number of words multiplied by the number of bits per word.
- (c) Read access time--the elapsed time from the moment when the system requests a stored word until the system receives that word from the storage unit.
- (d) Write cycle time--the elapsed time between the storage of two words in different addresses.

Storage units are classified according to operating mode, order of access, and permanence of the stored data. (U)

There are three types of operating mode to consider; the first is the addressable mode, the most common operating mode of storage units. This mode has a write and a read phase. Data is retrieved in this mode on the basis of the address of the word in storage. A second operating mode is the content addressed mode. This mode also has two phases, but the read phase is different because all or a portion of the stored digital data is the input of the read phase rather than the address. The input word is compared with all stored words serially until a match is found. Output is the remainder of the stored word or a unit record of a given number of stored words following the input data word. There is also a method for quizzing all storage registers in parallel for a match, as in associative storage units. The third mode is the buffer mode, which also operates in both a write and read phase. However, in the write phase, only the data word and control signal are inputs, no address being specified; in the read phase, data words are read out in the order in which they were stored--first in, first out. The newer push-down buffers have a last in, first out order. (U)

The order of access of a storage unit may be either cyclic, serial, or random. In cyclic access the addresses are arranged serially, traversed in one direction, and the first follows immediately after the last. An example of this type of access is the magnetic drum and the disc file. In determining read access time for cyclic access, the average read access time is considered; that is, half the minimum value plus the maximum value-- $0.5 (\min + \max)$. Serial access is a serial progression from any address to any other, passing all intervening addresses; this order is

often reversible. The elapsed time depends upon the number of intervening addresses. For single address inputs at random, the average access time must be considered. For example, for a non-reversible read magnetic tape, the average access time is the time required to read the entire tape divided by two. If a large block of addresses are requested simultaneously, the rate of data transfer must be considered; that is, the elapsed time includes the time to move from one address to the next address and to read the amount of data stored in that address. An example of this type of access is tape and card storage. Random access generally takes the least elapsed time but requires the most equipment and expense. In random access the input address is completely independent of the past history of addresses; that is, any storage register can be addressed with equal elapsed time without affecting the time required to retrieve data from the next address. Core storage is an example of random accessing. (U)

Stored data may be erasable, fixed, or volatile. If the data are erasable, then data can be entered under control of the system; hence, both the read and write phases are permitted. Both phases usually require operator intervention; the storage unit is changeable. Fixed data are essentially built into the storage registers and cannot be changed without a major revision of the equipment. These units are called recall or read only storage units. One advantage of such units is that they save the expense of write equipment and avoid accidental erasure. Data storage is volatile if the stored data is vulnerable to loss through power failure or other mishap, as, for example, in magnetostriuctive and fused quartz delays. With reliable power sources, this type of storage is not a drawback. (U)

(a) Magnetic Core Storage - The magnetic core is a high-speed random access, erasable storage unit that exploits magnetic hysteresis in ferromagnetic materials as its basic mechanism. The core presents a closed flux path that is switched from one stable state to another by passing a current on one or more wires through the aperture of the core to produce a magnetomotive force sufficiently above the threshold. Reversing the direction of the current produces the opposite state. The switching speed is a function of the magnitude of the applied field and is inversely related to the amount or volume of material switched. (U)

The most common core sizes are 80 by 50 mils and 50 by 30 mils. Smaller sizes would produce higher speeds (in the nanosecond range), but there is a physical limit to the size that can be practically handled. The use of two cores per bit increases the speed and temperature tolerance of core storage; it also equalizes the load on access outputs and has in some instances permitted non-destructive read-out. However, this design feature is achieved at an increased cost per bit. The moderate cost, high reliability as a function of speed, and simplicity of its concept have contributed to the widespread use of magnetic core memories in present-day computers. (U)

(b) Magnetic Drums - Another common method of exploiting magnetic hysteresis is in magnetic drums. A probe or reading head is passed through the demagnetizing field in air over a magnetic surface to permit sensing of the remanent flux state without actually switching it. The rate of change of flux induces a voltage capable of being sensed in the read head. This operation produces a non-destructive read-out that

can be electrically altered by passing a current through a similar (or the same) sensing head at the appropriate time and in the appropriate direction. A coating of ferrite material less than 1 mil thick is applied to the supporting surface for easy transport to the vicinity of the head. (U)

The read access time is limited only by the access circuit. The mechanical speed with which the read head and storage register can be brought together and the maximum frequency limit of the sensing equipment are the limits of reading speed. Packing densities of 22,000 bits per square inch have been achieved, and laboratory experiments have shown that more than 3,000 bits per inch are possible under single sensing heads that can be stacked 500 to the inch. (U)

A recent development in both magnetic drums and disc storage units is the flying head. There are three general classifications of flying heads: the contact start-stop flying head; the externally-actuated flying head; and the noncontact start-stop flying head. The contact start-stop head starts in contact with the drum surface and is lifted by the creation of a laminar film of air in the immediate area of the rotating drum surface. This type of head has the following disadvantages: start-stop wears the head and coating; increased starting power is required because of the friction drag of the heads against the coating; low temperatures may produce frost adhesion between head and coating; the head has a natural low resonant frequency. (U)

The externally-actuated head is a noncontact start-stop head that is physically moved into flying position by pneumatic, hydraulic,

electrical, or mechanical devices each time the drum starts. This type is suitable for disc files where hydraulic power is already available. (U)

There are two types of noncontact flying heads, the Bernoulli head and the general purpose flying head. The Bernoulli head operates by having an aerodynamic shape that causes it to fly toward the drum surface as the drum accelerates to operating speed. The equalization of forces as the head approaches the surface and rides on the laminar film surrounding the coating causes the head to fly stably within a few ten-thousandths of an inch of the coating. This type of head has been successfully used in airborne and military applications. (U)

The general purpose flying head is a parallel reed type that incorporates an aerodynamic surface on a conventional cylindrical head body to develop a flying force between the head and the recording surface. The cylindrical head body is supported on the free ends of a pair of spring reeds that urge the head toward the drum surface, as shown in Figure 5-15. A shoe of self-lubricating material is lapped flat and secured on the end of the head. This shoe lifts the head away from the drum and protects the drum surface from damage by accidental head-drum collisions during adjustment or servicing. An adjustment screw acts against the outer spring reed to vary the spring force urging the head toward the drum. As the force is increased, the head flies closer to the drum; as it is decreased, the head flies further away. The movement of the head toward the drum surface is limited by an adjustable stop screw. The stop screw is set while the drum is in the dynamic condition so as to provide a head-to-drum spacing that is slightly greater than the centrifugal and thermal

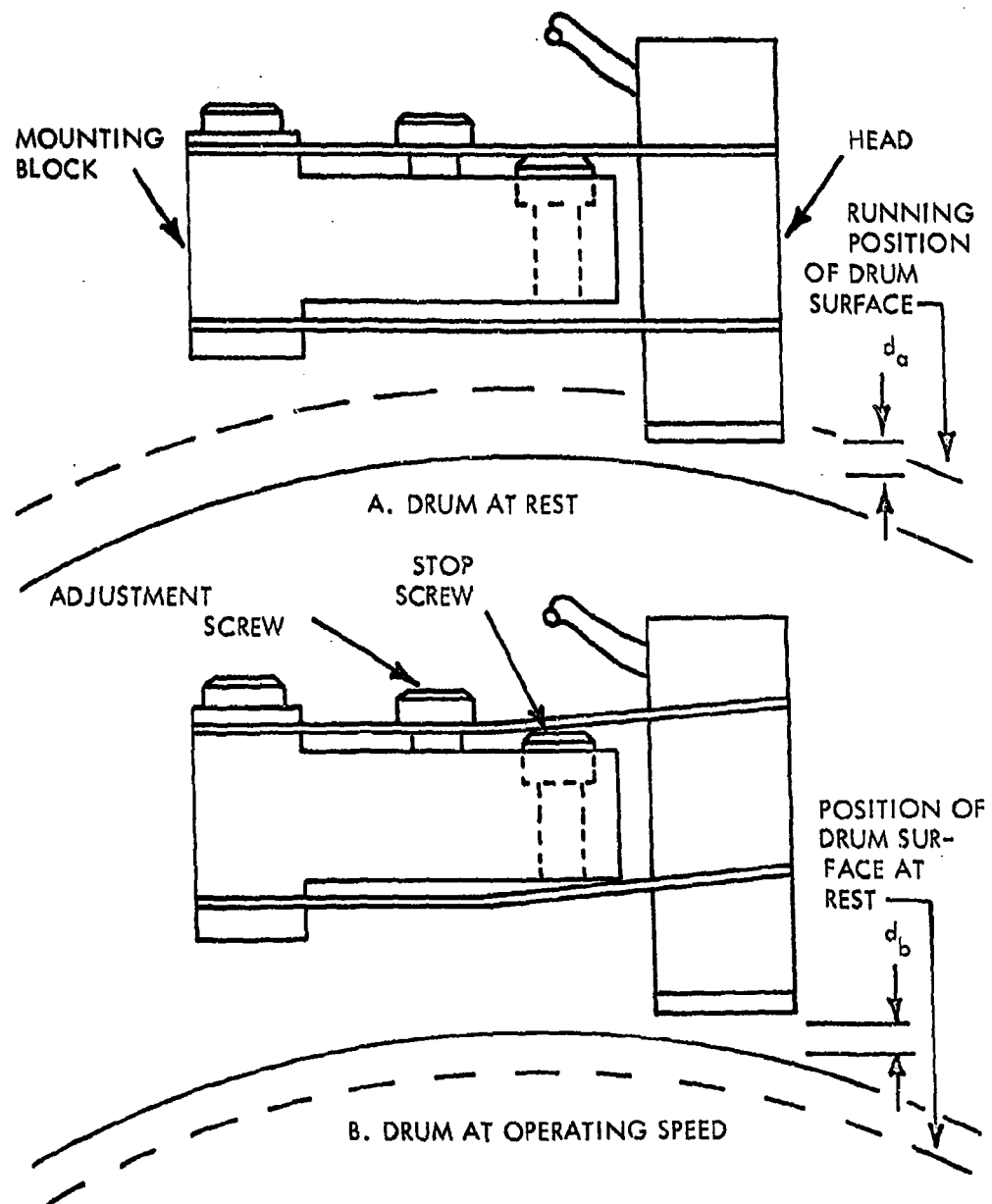


FIGURE 5-15. General Purpose Flying Head

expansion of the drum. Hence, the heads are not in contact with the drum when it is started. As the drum accelerates and expands, the drum surface approaches the heads but does not contact them because of the lift provided by the laminar film of air surrounding the drum. (U)

The general purpose flying heads can be used in both general storage and register applications. In a general storage configuration, a number of heads are supported on a common mounting block; in a register configuration, individual fixed and adjustable mounting blocks are used. (U)

A drum file with a 198.6 million bit capacity employs two drums with two magnetic head assemblies attached to a single linear track access mechanism. There are 2000 tracks per drum and 50 tracks per inch. The drums rotate at a speed of 870 revolutions per minute and have a bit density of 650 bits per inch. The head positioning time is a maximum of 550 milliseconds. The heads are aerodynamically floated at a distance of 0.0002 inch from the drums' surfaces. The two-drum configuration doubles the amount of information available to the access mechanism. (U)

Magnetic drum storage differs from magnetic core storage; the drum has dynamic cyclic access rather than static access. Cyclic access reduces the amount of read-write circuits per unit of storage capacity far below that of static connections. The total number of storage cells that connect with a single read-write channel during a single cycle of access is called a group of storage cells. A storage unit may then be cyclic but have separate read-write circuits for each group. The group is accessed statically even though the storage unit is cyclic, thus

permitting random accessing of groups. Alternatively, a single read-write channel has a dynamic, serial connection with the groups, as in the flying-head drums. (U)

Although magnetic drums have been used for the high-speed working storage unit of some relatively slow computers, their longer access time has relegated them primarily to the role of back-up storage units. The various types of drum storage units are compared in Table 5-5. (U)

(c) Disc Storage Units - Disc storage units are another example of dynamic access storage units based upon the principle of magnetic hysteresis. Although similar in principle to drums, they are capable of achieving far greater speeds. Disc storage units provide random accessing of groups of storage cells in much the same manner as drums. Early disc storage units had only one read-write channel so that the head moved linearly in two dimensions but not on several discs simultaneously. Speed was increased by including multiple heads per arm and one for each disc face, thus greatly reducing both the mechanical distance the heads must travel and the access time to a group. By simultaneously positioning the arms on each surface, parallel read-out of individual words has been achieved. The read-write heads generally are the airborne type, such as the flying heads, and are loaded against the disc by pneumatic pressure. The boundary layer of air on the rotating disc supports the heads and prevents actual contact with the surface. (U)

Present-day disc storage units have a wide range of parameters. The number of discs vary from one to sixty-four with disc diameters varying

TABLE 5-5. DRUM MEMORIES

DRUM MEMORY UNIT	MAXIMUM rpm * rps	NUMBER OF TRACKS PER DRUM	NUMBER OF BITS PER TRACK	TOTAL CAPACITY (bits)	MAXIMUM ACCESS TIME (msec)
Bryant 5005	24000	85	2048	174,080	
Bryant 5008	18000	150	2048	306,200	
Bryant 5014	12000	280	2048	569,344	
Bryant 7505	12000	85	3072	261,220	
Bryant 7508	8000	150	3072	460,800	
Bryant 7514	6000	280	3072	860,160	
Bryant 10000	6000	85	4096	348,160	
Bryant 10008	6000	150	4096	614,400	
Bryant 10014	3600	280	4096	1,116,880	
Bryant 10019	3600	420	4096	1,720,320	
Drum File (2 drums)	870	2000	--	198,600,000	550
RW 400 Drum	3600	--	--	57,344	8.5
Univac FH-880	1800*	768	30720	23,592,960	17
Univac FH-500	3500*	384	25600	9,830,400	8.5
IBM 733 Drum	--	--	--	589,824	35

between 24 and 39 inches. The number of heads range from one to 288 heads per storage unit. Maximum access times for different disc units vary from 100 to 800 milliseconds, and character storage capacities vary from 5 to 176 million characters. The various types of disc storage units are compared in Table 5-6. (U)

(d) Magnetic Tape Storage - Digital magnetic tape recording has provided large capacity, medium-access time storage of information in digital computing systems for more than a decade. During this period, continued emphasis has been placed on increasing the total amount of information stored on a fixed length of tape and increasing the instantaneous transfer rate for writing or reading. (U)

Information is recorded on magnetic tapes in parallel bit, serial character format. This parallel multi-channel operation is the distinctive performance characteristic of magnetic tape recording, since it permits faster transfer rates than single channel systems. However, this principle does set a limit in extending the capabilities of tape units. Bit packing densities have increased from densities of 100 bits per inch to 300 and 400 per inch, and high-density systems range from 556 to more than 1,000 bits per inch. The highest rates are practical only when reliability is a less important factor in system design. (U)

One tape recording system manufacturer has initiated work toward the development of new techniques. Interchannel timing problems have been eliminated by independent self-clocking parallel recording channels that remove the time displacements of skew in the system electronically. A

TABLE 5-6 (PART I). DISC FILES

DISC FILE	NO. OF DISCS	DISC DIAMETER (INCHES)	SPEED (REVOLU- TIONS PER MINUTE)	TRACKS PER DISC SURFACE	NO. OF HEAD POSI- TIONERS	NO. OF HEADS	CHARAC- TER STORAGE (MILLIONS)	MAXIMUM ACCESS TIME (MILLI- SECOND)	READ/RECORD RATE (CHAR- ACTERS PER SECOND)	BIT STORAGE CAPACITY (MILLIONS)
Telex I	16	31	1200	256	16	64	22	250		155
Telex II	64	31	1200	256	64	256	88	200		622
Bryant 4000	24	39	900/1200	768	40	288	100	100		720
IBM 350 (RAMAC)	50	24	1200	100	1 or 2	1 or 2	10	800		
IBM 1301	50	24	1800	250	50	100	50/56	210		
IBM 1405 (RAMAC)	50	24	1200	200	1, 2 or 3	2, 4 or 5	20	800	25,000	

TABLE 5-6 (PART II). DISC FILES

DISC FILE	NO. OF DISCS	DISC DIAMETER (INCHES)	SPEED (REVOLU- TIONS PER MINUTE)	TRACKS PER DISC SURFACE	NO. OF HEAD POSI- TIONERS	NO. OF HEADS	CHARAC- TER STORAGE (MILLIONS)	MAXIMUM ACCESS TIME (MILLI- SECOND)	READ/RECORD RATE (CHAR- ACTERS PER SECOND)	BIT STORAGE CAPACITY (MILLIONS)
IBM 7300 Models 1-2	50 50			100 200	3 3		6 to 48 digits	100 to 850		
RCA High Speed Data Disc File							22-176	150	transfer rate 32,000	
RCA Low Speed Record File	128			20			4.6		2,500	
Honeywell 460-1 and 460-2	6 12						32 72	170 170		
Honeywell 460-3 and 460-4	18 24						108 144	170 170		

system employing this technique provides instantaneous data transfer at the rate of 121,000 ten-bit characters per second by recording 1,100 bits per inch at a tape speed of 110 inches per second. This system has high reliability and an exceptionally low bit drop-out rate despite a data transfer rate and storage capacity several times greater than that of conventional systems. (U)

Another manufacturer has introduced a sealed tape cartridge that contains an 1800-foot reel of tape, on which 25 million characters can be recorded, and an empty take-up reel. The cartridge is placed in the tape unit, a button is pressed, and the unit automatically opens the cartridge, engages the tape, and starts processing--all within 20 seconds. (This cartridge eliminates operator problems in handling the tape itself and thus increases the life of the tape.) (U)

Tables 5-7 and 5-8 compare some of the different tape units and the various types of tape available. (U)

(e) Thin Film Memory - Great increases in speed of switching have been demonstrated in devices exploiting the rotational mode of switching. In this type of device--thin film, for example--the entire magnetic domain is rotated to the new orientation simultaneously, so that switching time is limited only by the rise time of the excitation energy. (U)

Films deposited on a substrate show a preferred or easy direction of magnetization, shown by a square hysteresis loop, under the

TABLE 5-7. TAPE UNITS

TAPE UNITS	DENSITY (BITS PER INCH)	TAPE SPEED (INCHES PER SECOND)	REWIND SPEED (INCHES PER SECOND)	HIGH SPEED REWIND (MINUTE)	START/STOP TIME (MILLISECOND)	CHARACTER RATE (CHARACTERS PER SECOND)
IBM 729 II	200 or 556	75	75	1.2	10.8	15,000/ 41,667
IBM 729 IV	200 or 556	112.5	112.5	1.2	7.3	22,500/ 62,500
IBM 7330	200 or 556	36	36	2.2		7,200/ 20,016
Ampex	300	1 and 150	230		3.0	45,000
Ampex	200 or 556	75			3.3/1.8	
Control Data 606	200 or 556	150	150	2.0 (unload)	4.0	30,000/ 83,400
Honeywell High Density	555					111,000
Honeywell Super Density	777					155,000
Potter High Density	1,200	100	200		4.5	120,000

COMPARISON OF LOW SPEED AND HIGH SPEED TAPE REALS
(Number of records per reel)

Recording Density (bpi)	Record Length (# of Characters per Record)					
	6	30	60	300	1800	3000
200	36,460	31,600	27,085	12,640	2,916	1,805
556	37,375	35,373	33,069	22,046	7,127	4,624

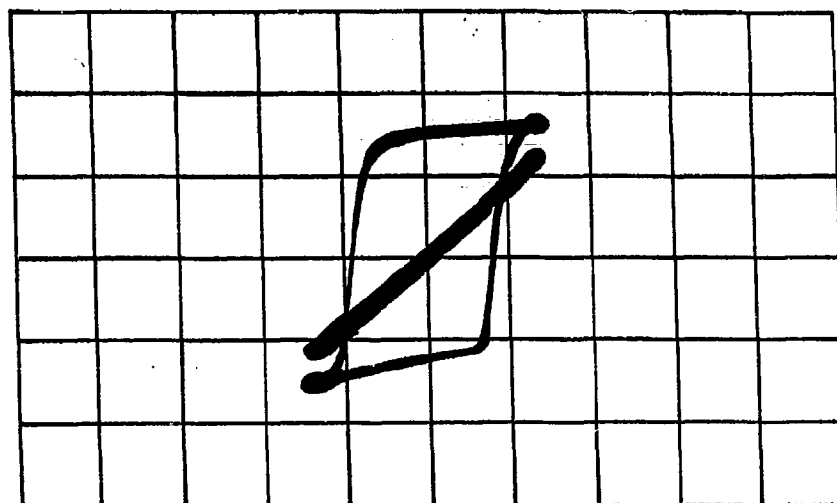
NOTE: 2370 feet of usable tape with 3/4 inch record gaps is assumed. All figures are approximate.

TABLE 5-8. TAPE UNITS

KINDS OF TAPE	MATERIAL COATING AND BASE		SPECIAL FEATURE	WIDTHS (INCHES)	LENGTHS (FEET)
Scotch 457	.38 mil oxide	.65 mil tensilized polyester	2 times record- ing time	$\frac{1}{4}$; $\frac{1}{2}$; $\frac{3}{4}$; 1	2400 4800 9600
Scotch 458	.38 mil oxide	1.5 mil polyester	maximum strength	$\frac{1}{4}$; $\frac{1}{2}$; $\frac{3}{4}$; 1	1250 2500 5000
Scotch 459	.38 mil oxide	1.0 mil polyester	50% more record- ing time	$\frac{1}{4}$; $\frac{1}{2}$; $\frac{3}{4}$; 1	1800 3600 7200
Audio- tape	mylar 150 cellulose- acetate		tensile strength 28 lb. 10 lb.	$\frac{1}{2}$ $\frac{1}{2}$	2400 2400
Compu- tape	.35 mil .5 mil 1.0 mil 1.5 mil	1.45 .92 polyester	556 or 800 bits per track inch	$\frac{1}{2}$ $\frac{3}{4}$ 1	1800 2400 3600 4800
Ampex		1.0 mil mylar		1	3600

influence of a magnetic field. Perpendicular to this easy direction, called the hard direction, the film shows a linear loop, as illustrated in Figure 5-16. The wall coercive force and the rotational saturating force are obtained from these two characteristics. Any magnetic square-loop material can be used as a storage element in a random-access memory. The unique geometry, directional magnetic properties, and rotational remagnetization process are characteristics that must be considered in the design of thin-film memories. Their behavior can be compared with a magnetic dipole having two states, both parallel to the preferred or easy direction. These states represent the storage of a 1 or 0 in each thin film element. Three conductors are associated with each bit. The word drive conductor is parallel to the preferred direction; the information and sense conductors are parallel to the hard direction. (U)

The information conductor is split in order to reduce the mutual capacitance between the information and sense conductors and to reduce eddy currents induced in the information conductor by the word drive current. A current in the word drive conductor generates a transverse field. If this field is greater than the rotational saturating force in the hard direction, it rotates the magnetic moments or dipoles to the hard direction, thus inducing a sense signal. This signal is positive if the rotation originated from the 1 state; negative, if from the 0 state. To magnetize a selected area of thin film to the 1 or 0 state, two magnetic fields perpendicular to each other are applied--the drive field in the hard direction, the information field in the easy. The resultant field lies between them oriented toward either 1 or 0 depending upon the



SQUARE AND LINEAR LOOPS

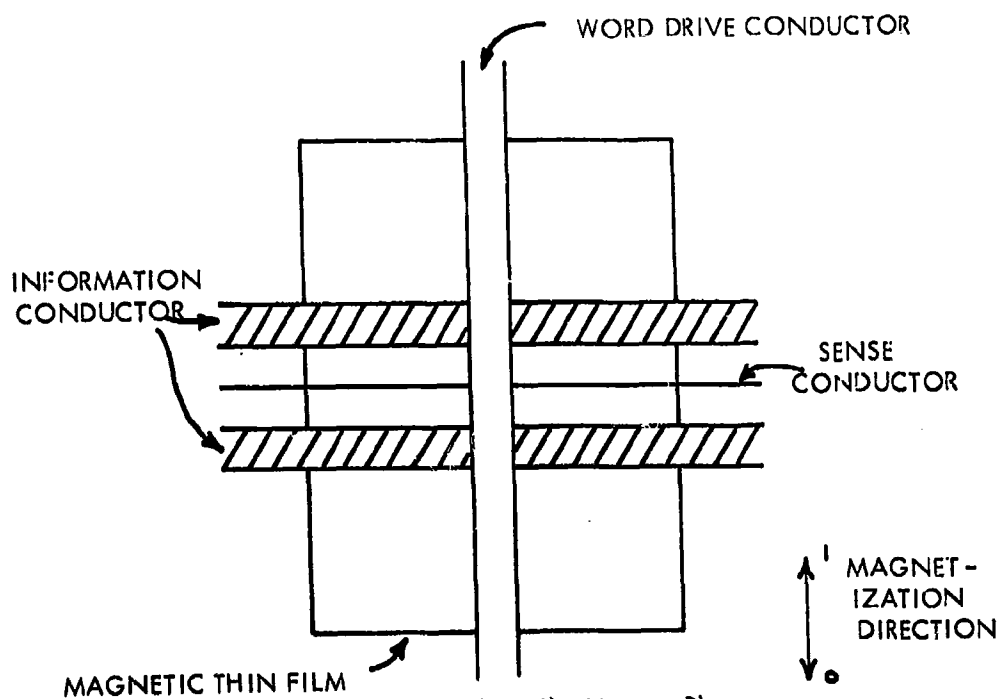


FIGURE 5-16. Thin Film Memory Planes

direction of the information field. Removal of the drive field allows the dipoles to fall to the desired 1 or 0 state, after which the information field can be terminated. (U)

One thin film memory has an access time of 300 nanoseconds (0.3 microseconds) for 128 thirty-six bit words. An experimental thin film memory operates at an access time of 60 nanoseconds. The experimental memory holds 256 words of 72 bits each--a total of 18,432 bits--each occupying a rectangular area of 0.12 by 0.026 inch. The thin film bits are 0.0002 inch thick and the entire planar array measures 4 by 8 inches with a packaging density of 576 bits per square inch. High-speed transistor circuits produce nanosecond drive pulses to switch the film in less than 5 nanoseconds. The simple design of thin films inherently allows for production in large quantities at a fraction of the cost of wired ferrite cores. (U)

4. Computing Units

(a) Elements of a Computer - A digital computer is a device for manipulating symbols. The principal element of a computer is an arithmetic unit, which performs the four operations of addition, subtraction, multiplication, and division at speeds ranging from 0.1 to 0.000001 second. The addition of a memory unit and a control unit produces the automatic digital computer. (U)

The purpose of discussing computing units is to isolate the part of a computing system that contains the arithmetic, logical, and control functions from the input, output, and storage units. The

following definitions clarify this distinction:

- (1) Arithmetic unit--the section of a computer where arithmetic and logical operations are performed on information.
- (2) Control unit--the section of a computer that directs the sequence of operations, interprets the coded instructions, and initiates the proper signals to the computer circuits for the execution of instructions. (U)

Storing and retrieving information is an essential function of a computer. The problem of allocating storage is one of assignment: programs and data must be assigned space within the available storage medium of the computer. This allocation may be performed by two techniques: pre-planned storage allocation or dynamic storage allocation. These methods are not mutually exclusive, and in many cases a combination of the two provides the most efficient system. (U)

(b) Parallel Operations - Parallel processing or multiprocessing, which is related to parallel programming, seeks to achieve parallel operations with equipment, rather than with programs, through the time-sharing of one or more memory units by a number of processors or arithmetic units. Parallel processing is generally considered when greater computer speeds are the primary design requirement. Serial computers may not be sufficiently fast to solve particular problems; thus it is important to study the potential and the associated problems of parallel computers. (U)

The studies of parallel operations performed on available computers have only permitted operations of different types--such as table look-ups or input-output--to occur in parallel. A more difficult problem is one in which operations of any type are permitted to occur in

parallel; hence, it may be assumed that all operations are of the same type and will require the same equipment. Assuming a number or array of processors connected in parallel to a single memory and allowing some registers for sequencing or queuing, the problem of paralleling operations can theoretically be achieved. Exploration of this potential might prove valuable in developing a system capable of retrieving various types of information simultaneously. (U)

Parallel operations have been achieved in different ways. One high-speed computer provides overlapping operations with a core storage divided into stacks of 4096 words, each with independent access to the computing unit. These stacks are interleaved in pairs--even addressed words in one stack, odd in the other--so that access to consecutive words of the store is not limited by the core cycle time. Instructions are drawn from the store in pairs. The logic of the arithmetic unit is so arranged that the instruction (or decoding) cycle of the next operation occurs during the execution cycle of the previous operation. Access to storage and the arithmetic unit is arranged so that an instruction dependent upon the results of a previous operation will wait until the preceding instruction has been completed. (U)

Another high-speed computer handles parallel operations in its instruction unit. The primary function of this unit is to fetch and prepare instructions for execution by the arithmetic units; secondarily this unit executes a large number of the machine instructions, including the indexing and branching instructions. Preparing an instruction prior to its execution may include using an index to compute the

operand location, fetching of the operand, and handling a number of error and interrupt situations. The speed of the computer is largely the result of the independent and simultaneous operation of the major elements of the computer. By overlapping the operations of the arithmetic, instruction, and memory units, a floating-point arithmetic operation is performed at a rate of two instructions per microsecond. Additional research is needed to attain an operating rate approaching that of the fastest components of the system. The limit normally imposed by the slowest component is largely overcome by an effective use of parallelism. (U)

Another approach to the problem of parallel operations is parallel programming. The state-of-the-art shows few computers with built-in parallel processing capabilities; hence, it may be advantageous to simulate this facility with an advanced programming system. Facilities for paralleling input-output operations are available in most modern computers. Many of the benefits of parallel computation sequences may be derived by using a sequential computer with parallel input-output facilities in conjunction with appropriate programming techniques. (U)

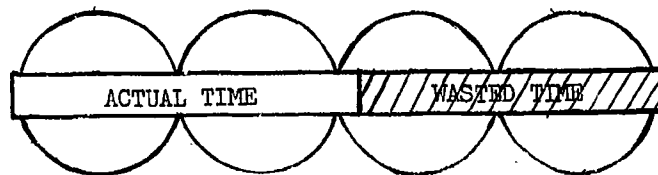
(c) Synchronous and Asynchronous Computers - Some computers are designed so that input-output operations occur while the computer is processing information. This feature is a type of asynchronous operation that is better termed multiple processing. (U)

In a synchronous computer, each data processing operation is synchronized by the pulses produced by a master clock. A unit of time is allotted for each step, and the computer cannot proceed to the next

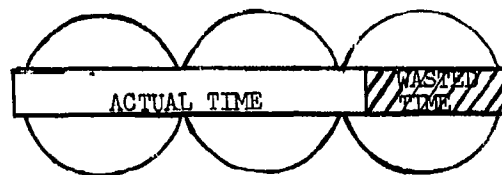
step until it receives a pulse from the clock. Such a computer has a basic instruction timing cycle of several steps. All instruction cycles are regulated by the number of steps needed to complete the longest instruction. For example, if the longest addition takes four steps per cycle and a specific add instruction took fewer steps, the computer would have to wait until the end of the fourth step before another instruction could begin. (See Figure 5-17, Part A.) (U)

The wasted time in fixed cycle instructions led to the development of the variable synchronous computer. This machine tests whether the instruction has been completed after each step of a cycle. For instance, if a particular add instruction requires two cycles, the next instruction could begin at the start of the third step instead of waiting the full number of steps needed for the longest addition. However, an instruction could not start within a step. These instructions are called asynchronous. (See Figure 5-17, Part B.) (U)

In asynchronous machines, computing speed can be gained by letting the adder proceed to the next operation as soon as all carries have propagated, rather than waiting a fixed time that corresponds to the longest possible time to propagate a carry. One method is to signal the completion of carries. The average longest carry is computed for different word lengths by considering all possible cases of distribution and allowing a correction for the delay that occurs at the place where the longest carry chain originates. Such an arithmetic unit makes optimum use of the inherent speed of its components. In completely asynchronous computers, there is no waiting for the completion of a



A. SYNCHRONOUS INSTRUCTION



B. ASYNCHRONOUS INSTRUCTION



C. COMPLETELY ASYNCHRONOUS OPERATION

FIGURE 5-17. Synchronous and Asynchronous Operations

step or a cycle; immediately upon completion of an operation, the next operation begins. (See Figure 5-17, Part C.) This procedure not only speeds up the over-all system operation, but it allows for the addition of new and faster components without major system redesign and rewiring. (U)

F. Development Requirements

An optimum configuration for a Fact Correlation System was selected and analyzed in Section D. As research and development continues, some of the equipment selected may become obsolete as more sophisticated and powerful equipment is introduced. (U)

The problem of development requirements can be divided into two areas:

- (a) Development anticipated as feasible within the next three years.
- (b) Research and development results that may be realized within the next ten to twenty years.

Any development that may be realized within the next several years will be based upon known concepts--the research has been essentially completed. For the longer term the primary requirement is to perform basic research to resolve physical problems or to establish a sound theoretical framework for conceptual problems. (U)

In the near future, improvements in input devices may be expected, since the need for faster and more efficient methods of handling large amounts of input data has already become apparent. Foremost in this area are devices capable of reading printed matter reliably at high speeds, even allowing for some degradation of print. Presently available devices

are hindered most by mechanical limitations such as their inability to handle paper at sufficiently high speeds. The development of efficient abstracting techniques is also in order, so that this slow meticulous process performed almost exclusively by human operators may be relegated to a high-speed computer. (U)

Another device needed for retrieving information is a parallel random access memory. Such a device would be capable of interrogating a large corpus of data simultaneously instead of sequentially. In present systems a program must either find the address of the information desired and then read the information from that location or scan the entire memory sequentially until the correct information is found. In a parallel random access memory, information could be retrieved in one memory cycle regardless of the size of the memory or corpus. (U)

Advanced research may produce associative or search memory systems distinguished by the rational spacing of information and by a considerable decrease in access time. In such systems, the storage and retrieval of information would not be achieved by referencing fixed memory cell addresses; the process will be performed according to the operatively variable attributes of the information groups. The effect of random access will be attained regardless of the location of the information. (U)

Another important theoretical problem for future resolution is the automatic discernment of the structural properties of information and the formation of concepts and opinions in machines. The most pressing problems are discernment, automatic perception, correction of printed text,

clarification of noise-distorted speech, and recognition of other forms of information. The formation of concepts and opinions is closely related to adaptation and, in this regard, it is important to investigate the physical structure of biological phenomena and attempt to simulate them. (U)

Microminiature elements, thin films, and the discovery of new physical principles should pave the way to producing more powerful computers at decreased costs. Microminiaturization of cybernetic machines promises to create radically new conditions for the construction of computers. These devices may approximate the storage capacity and information handling ability of the human brain. (U)

The following sections outline some areas of research activity both in the United States and abroad with some indications as to what has been done to date and what is expected in the future. (U)

1. Speech and Pattern Recognition

Speech analysis, pattern recognition, and allied processes are considered one of the branches of cybernetics. Speech recognition is still a complex problem for which no successful solution has been found. Most developments have been limited in operation by requirements for a specific voice, speech tempo, pronunciation, accent, or other similar factors. In experiments, words are pronounced in declamatory clear style as contrasted to conversational speech. (U)

A recent development in speech recognition is an experimental device that responds to spoken commands. It presently recognizes and

responds to 16 words (including 10 digits); with improvement in measurements and logic it is expected to be able to respond to as many as 10,000 word patterns. (A word pattern consisting of a group of syllables and/or words is the minimum meaningful unit of language recognized by the machine.) Most ordinary variations in speech rates and differences in pitch and inflection do not affect the machine's recognition ability. It responds almost as well to singing as to normal speaking tones, is not baffled by whispering or shouting, and operates in the presence of a considerable amount of ambient noise. This device is based upon the discovery that speech sounds can be identified mechanically by a unique feature of the speech wave arising from its phase structure. This phase-dependent asymmetry property in the wave is not recognizable by the human ear, but voice recognition machines using the property may be made simpler and more reliable. (U)

In pattern recognizing apparatus, a continuous image may be perceived by its discretization into a number of elements, in a manner similar to the functioning of the retina of the human eye. Information about the degree of illumination and the position of elements may then be analyzed by the machine for further details of recognition. The set of coordinates of the elements forming the contour of the image is called the coordinate description of the image. Different coordinate descriptions of the same image will be received by the retina of the apparatus, depending upon the size and position of the projection of the image upon it. (U)

One approach to the problem of pattern recognition is a combination

of analog and digital techniques. The initial input of a character is straight-forward; a flying spot is used to scan the pattern at a rate of 3000 characters per second, and the output signal wave is converted to bits indicating black or white areas. The result of the scan is 200 bits, 20 vertical by 10 horizontal. A two-dimensional 200-bit shift register stores this image of the original pattern as it develops. The system is limited at present to 14 characters. Horizontal and vertical centering by means of flip-flop registers places each character image in a unique standard position. Then the salient features of the pattern are extracted by determining the character's shape through an analysis of first and second differentials of current flow at a point. The process is an analog function. (U)

A pattern recognition program for hand-printed characters has been developed. An unknown input is pre-processed by filling in holes and eliminating local noise. Then it is supplemented by three rotated replicas. The four representations of the pattern are subjected to a set of 28 different tests. These tests examine patterns for such factors as cavities and a number of intersecting lines. The output of each test is a string of up to 12 digits, ranging from 0 through 7. These outputs are then compared with summary lists of outputs written during a previous accumulation of experience, and a weighted average of the inverse probabilities corresponding to the observed outcomes from the separate tests is computed. The program gives the input the name corresponding to the highest of these averages. The program has been tested on a ten-pattern alphabet of hand-printed capital letters. Sixty examples of each letter

(on the average) were split into two halves. One half was then used to accumulate information in the summary lists; the other half was used to test the program's abilities. Eighty seven per cent of the test inputs were correctly identified. (U)

A character recognition machine that converts printed character to punched cards at rates up to 240 characters per second has been developed. The accuracy claimed is fewer than one in 10,000 rejects and one in 1,000,000 errors. The system includes a scanner, memory, logic, and document handler. The sensor uses a flying-spot scanner and a photo-electric pick-up; it pre-scans each character twice to establish peak white and black levels and x-y limits for centering the character. The final scan transfers character information into ferrite-core storage. (U)

Work concerned with conditioned reflexes is closely related to developments in character recognition. One approach uses the multistep characteristic of ferrite cores in a matrix with the sensors connected to one axis and reactors to the other. By repeatedly exposing the sensors to a situation, a conditioned reflex is developed; when the reactors are interrogated, a learning-type reaction is obtained. In addition to character recognition, applications of this technique would be useful in information retrieval and automatic speech recognition. (U)

Research in pattern recognition being carried on in the Soviet Union includes the recognition of three-dimensional objects. (U)

Speech and pattern recognition has been considered in terms of input to a computer. However, an optical scanning device for output

is also possible. Such a device would convert signals into the appropriate characters and deposit them on paper by means of a suitable technique such as photography. A mechanical problem remains--the skewing of the characters on the paper or other output media. (U)

2. Storage Devices

The research being performed on storage techniques and devices is diversified. It includes an investigation of many physical principles and their possible applications based upon the exploitation of the electronic, electromagnetic, thermal, photoscopic, and other properties of various materials. Research is also being performed in hydraulic logic, microminiature components, and molecular electronics. The following paragraphs first present the principles being investigated and then discuss some of the experimental devices being developed. (U)

(a) Principles - Increases in speed of switching have been demonstrated in devices exploiting the rotational mode of switching. The entire magnetic domain is rotated to the new orientation simultaneously, as opposed to domain wall motion. Rotational switching is generally realized in thin film spots, cylinders, and sheets of a nickel-iron alloy similar to Permalloy. The Fluxlok technique demonstrates one example of high-speed rotation in ferrite cores. The switching time of these devices is limited only by the rise time of the excitation energy. (U)

The Curie point effect establishes another way to store data in a magnetic medium. The entire area of a thin continuous film of material is magnetized to the saturation normal to its surface. An

electron beam is allowed to impinge on an elemental area (about 0.002 inch in diameter) of the film. The temperature of the spot is thus raised above the Curie point and the area is effectively demagnetized. Read-out can be effected either electrically by an electron mirror microscope or optically by the Kerr magneto-optic effect. (U)

Ferroelectric materials are a class of ionic compounds with dielectric properties that are analogous to the properties of ferromagnetic materials. Ferroelectric materials exhibit electric hysteresis in polarization versus electric field intensity. Devices have been fabricated both from artificially grown single crystals and from polycrystalline ceramics of either single ferroelectric compounds or mixtures of compounds. Thin films of ferroelectric materials have also been developed. Potentially small size and planar geometry permits many devices on a single substrate with large output signals available. Information is stored by coincident voltage and read-out by voltage pulses of radio-frequency sensing. So far, this technique has proved disappointing because of the instability of available materials and the lack of a true threshold field. Switching speeds will not be as fast as those of ferromagnetic devices, unless extremely thin ferroelectric films can be developed, since ion displacement is slower than spin rotation. However, the high impedance and nonlinearity of ferroelectric devices has been advantageous in conjunction with electroluminescent storage and display devices. (U)

The extreme speeds (switching speeds less than one nanosecond) of tunnel diodes and their small physical size has led to considerable

effort to design reasonably large capacity, high-speed memory around them. The problems of sensing the stored bit and the high cost per bit are still to be resolved. (U)

Relays were among the first devices used as storage cells. Their slow speed has made them almost obsolete. New interest in relay circuits has been generated by the possibility of manufacturing micro-miniature relay devices with electron-beam matching. One type of signal delay not yet fully exploited is spin echo, the use of nuclear spin phenomenon or organic materials in a strong stationary magnetic field. Data are entered via a solenoid around the material and are returned in a mirror image in time about the recall pulse. One serious disadvantage is that data cannot be re-entered immediately after reading, since a settling time is required before a cell can be re-used. (U)

The presence or absence of deposited silver, as in a photographic plate, may feasibly store more than a million bits per square inch. The means of access to determine the presence or absence of a particular bit or group of bits in such high-density storage is an unresolved technological and economic problem. The two most successful attempts to exploit the high-density possibilities of photographic plates have been a flying-spot storage and the Photoscopic Disc Reader. The capacity is not limited by the ability of photographic films to resolve spots; the limit is imposed by the inability of the flying-spot scanner to produce and position the light spots in phosphorus with sufficient brightness and accuracy. The spot brightness is limited by the risk of burning the screen; techniques for rotating the tube face have been

used to move the spot over the phosphor without burning while still holding it in a fixed position relative to the storage. A servo loop is employed around the spot-positioning circuits in order to achieve the required accuracy. (U)

The permanent-magnet twistor storage units rely on the presence or absence of small permanent magnets glued to a replaceable card. Card capacitor store and ferrite rod store use the presence or absence of capacitors and inductors to modify the coupling at the intersection of word selection lines and digit lines. (U)

Thermoplastic recording uses the principle of material deformation. An electric charge is deposited on the thermoplastic material in accordance with the desired information pattern. The material is then softened thermally and the equilibrium of two forces (electric and surface tension) deforms the surface, which is then refrozen. The principal means of reading out the stored information takes advantage of the differential defraction that the distorted surface causes to a collimated beam of light in a Schlieren optical system. It is conceivable with electron optics to deposit as many as eighty million bits per square inch, if the read-out is to be optical. The limitations on storage density will probably be similar to those on photographic stores, which are limited by the light source of flying-spot scanner. One principal advantage of thermoplastic over photographic recording is the electrical changeability. However, to erase and change a given bit requires the heating and softening of a large number of bits in a given area, and the

time required is considerably longer than a read-out cycle. (U)

Photochromic storage phenomenon is a reversible photographic process in which three separate wave lengths of light are employed--one to write or initiate high absorbance, the second to read or sense the absorbance, and a third to erase or eliminate the absorbance. Photochromic storage cells have been made in the laboratory from organic compounds whose absorbance to a particular wave length of light may be switched to a high value by previous illumination. This process is reversible, the absorbance being erased by subsequent illumination at a different wave length. Densities in excess of 10^6 cells per square inch are practical. The problems encountered include the accession to any type of photographic storage and the need for variation in the wave length of the light source. (U)

Three principal considerations limit the maximum storage capacity of any unit: technical feasibility, ultimate reliability, and the cost per bit to construct the unit. The cost per bit for small capacity storage units (10^4 bits) may be as high as one dollar or more per bit. Hence, high-speed storage, including circuit aggregates rather than discrete elements, can be employed. Increasing the storage capacity to 10^6 or 10^7 bits means that costs of a few tens of cents per bit are reasonable, even at the sacrifice of some speed. Magnetic cores with one or two cores per bit and an access time of 1 to 10 microsecond are in this category. It is technically feasible to increase storage capacity to 10^8 or 10^9 bits, but this increase would be expensive. Storage capacities greater than 10^8 bits require the cost per bit to range from a few

cents to a fraction of a cent per bit. Some possible techniques for reducing costs are devices such as the twistor store, thin films, and ferrite aperture plates. Costs may also be reduced by relaxing requirements such as speed or by eliminating high speed in writing and having only high-speed reading. (U)

Technical feasibility is an important consideration also. The accumulation of partial disturbance in a coincident-current magnetic core unit rapidly increases beyond the output signal; techniques such as checkerboarding and strobing the output for optimum times have been developed to extend this range. Another limitation is the amount of high speed drive available from a single access output. Increased speeds and capacity both mean increased back voltages; hence, more power is necessary per access output. Load sharing and other redundancy techniques have shown that the addition of one set of redundant windings in a magnetic core access switch can double the power output. Since resistance is exactly zero in superconductive devices, the possibility of eliminating the accumulation of a large number of minute noise sources is admitted. Cryogenics gives hope of large capacity random access stores, especially in the fixed or read-only mode. (U)

Transition from small to large capacity static storage units puts an additional strain on component reliability. Components that fail once in 10^8 hours are considered to be fairly reliable; however, in the case of storage capacities of 10^8 or more bits, components with this order or reliability would result in failures several times per hour, even with

only one component per bit. Therefore, either superreliable components are required or one or more of the following design characteristics are necessary: distributed storage, redundancy in access, and error correcting codes. (U)

The maximum speed of a storage unit is based upon three factors (for a particular static access storage unit): the reaction time of the storage register; the propagation and other miscellaneous times that complete the cycle time of the unit; and the limitation caused by heating and power dissipation. The reaction time for magnetic core storage registers is generally the switching time of the cores. It is possible to shorten the switching time by providing high driving currents; however, in coincident-current memory or any unit of more than one read dimension, the read selection ratio cannot generally exceed 2 without complex wiring. Switching usually takes 1 microsecond for the linear select units. This speed can be reduced by overdrive to approximately 0.2 microseconds for reading, but the write time remains the same. There are several ways to reduce the reaction time. One is to reduce the physical size of the core; another, to reduce the size electrically by means of partial switching. Another powerful means of increasing the speed or decreasing the reaction time of the storage cells is through rotational switching of the magnetic domains as opposed to domain wall motion. This technique is exploited in thin films principally, but has been demonstrated in cores by the Fluxlok technique and in strong overdrive techniques. Rotational switching modes are limited in their reaction time only by the rise time of the excitation field. In any storage unit in which rotational switching is

used, the time required to complete a read or write cycle is almost completely dependent upon the access equipment and is, therefore, almost independent of the storage mechanism. A need for detailed studies of access equipment still exists. (U)

Propagation time and miscellaneous time losses include the switching times involved in the input address register; propagation time through decoders, drivers, and access switches; physical propagation time caused by a size alone from access equipment to register; delay time in strobing and sensing; and propagation time from sensing equipment to the input-output register. Physical size is important in high-speed storage units since signal information travels only 8 to 10 inches per nanosecond; for a 10 nanosecond cycle the entire distance from the input-output register to the farthest storage register must be only a few feet. (U)

Coincident current memories require high uniformity in core-to-core properties and local differences in temperatures. Repeated addressing of nearby registers or unequal cooling can cause malfunctions. Increased packing densities required for reducing propagation times dissipate larger amounts of energy per unit volume. (U)

(b) Experimental Devices - Magnetostrictive delay lines are the most common means of exploiting signal delay in currently available storage units. They are essentially a rod or wire of an iron-nickel alloy that tends to change its physical dimensions in a magnetic field. The read-write transducers are merely solenoids to produce the pulsed magnetic field. Bit information is propagated as a single (or double)

pulses of mechanical stress. The line and circuits are simple, reliable, and inexpensive. Cyclic access is used and the units are competitive with drums. (U)

The flying-spot storage uses photographic plates for data storage and a flying-spot-scanner, photo-multiplier for access and sensing. It operates in the addressable mode and is a fixed bit changeable store with pseudo-random access. The number of bits per word determines the required number of plates, and the number of bits per plate determines the number of registers. A separate optical system is used with each plate, so the image of a cathode ray tube face is individually focused on each plate. Each plate has a separate photo-multiplier. A particular register is accessed by positioning the electron beam inside the cathode ray tube. The phosphorus screen converts the electron-beam to a light beam that is focused on each bit plate simultaneously for parallel read-out. Present systems store 131,000 seventy-six bit words with 5 microsecond read-access time and 64,000 seventy-six bit words with 4 microsecond rate. The maximum capacity is limited by the number of spots that can be generated on the cathode ray tube face. (U)

The photoscopic storage device also employs photographic storage and a flying spot scanner. The data are stored on a single annulus of a revolving glass plate and the unit is accessed cyclically. A cathode ray tube and a movable lens are used radially for track selection. Storage capacity is 21×10^6 bits; average read-access time, 25 milliseconds. (U)

A thermoplastic plate storage unit with flying-spot scanner access is a device in which a light beam from a cathode ray tube is directed at one plate at a time and bits are read serially by scanning the beam. It is an erasable store under control of a parent system. Capacity is 18×10^7 bits; time, 1 millisecond. (U)

In a permanent magnet twistor storage the presence of a permanently magnetized spot on the card is detected by its biasing effect on the twistor element, which prevents switching in one mode and aids switching in another mode. The absence of such a spot permits the normal switching of the twistor. Such storage units, at present, are designed in the linear select fashion (S_p approaching infinity), but this design is imposed to relax the square-loop requirements of the twistor rather than to answer any fundamental need. (U)

The Kilburn rod used in the Atlas MUSE computer must be designed with linear select with an infinite read selection ratio. Ferrite rods are then placed in the vicinity of the intersection of the word and digit lines. The presence or absence of these rods modifies the coupling for a semi-permanent store. An ingenious device has been constructed for moving the rods rapidly in or out of position pneumatically. Writing is essentially serial access; as the machine travels, the track places rods in their correct position. Reading is random access with a cycle time of 0.2 microseconds. (U)

The capacitor card is also a linear select arrangement in which the coupling between the intersection of word lines and digit

lines is modified by the presence or absence of a small capacitor. The cards are essentially punched paper cards with a metalized foil sandwiched into it. The grounded foil forms a Faraday shield between selection lines and digit lines, preventing coupling at intersections. Where the shield is punched out, a capacitive coupling of approximately three picofarads is formed. Read times are random-access to linearly selected words that may be either the columns or rows. The time factor varies with the size of a storage unit: 0.1 microseconds for storage of 10^4 bits; 1 microsecond for 10^6 bits; and 10 microseconds for 10^7 bits. (U)

In Europe, random access memories such as the Carousel memory permit access to any of over five million stored decimal digits in less than 2 seconds on the average. The memory contains 64 small reels of 8-channel magnetic tape mounted in 2 concentric circles on a wheel. A given reel of tape is read by indexing the wheel so that the selected tape is at the bottom. A weight attached to the end of the tape drops down past an air-gap read-write head to an unwinding bin. Photo-electric sensors control start, stop, and rewind. A fully loaded wheel can be replaced in 10 seconds. (U)

Another approach is the K-10 memory. The storage capacity is equal to a standard tape transport with a 1000-meter tape, but its access time is only 2.5 percent of the time required by a standard transport. The K-10 memory has ten bins, each containing 100 meters of tape. Each bin has an associated read-write head and a drive mechanism. Tapes rest with their mid-point over the head and are detected photoelectrically.

A typical application uses a crossbar switch, four tape control units, and nine K-10 units for simultaneous reading or writing on any four of 90 tapes. (U)

A Soviet development is a capacitive memory used for permanent information; i.e., tables, dictionaries, or control program. Information patterns are stored on a number of metal-coated paper sheets bonded together under pressure and hermetically sealed. Each 195 by 255 millimeters stores 512 bits, and a stack of 500 sheets together with selection logic occupies 0.027 cubic meters. Thus a cubic meter stores 9×10^6 bits. The read-out rate is 30,000 sheets per second, a speed that can be improved by a factor of ten with improved fabrication techniques. (U)

Wired-core memories are especially popular in Europe. A wire is threaded through a core matrix in a pattern that represents the stored information. Advantages are: nondestructive readout, high speed (usually less than 1 microsecond switching time), and low cost. These memories are used to store various fixed or seldom-changed data; e.g., start-up sequences, basic input-output programs, sub-routines, and even some operating programs. This technique is used also in microprogramming and industrial control. E-shaped wire cores are used: one slot stores a 1; another, a 0. With 52 cores and 52 diodes a storage capacity of 256 words is achieved on a single plug-in card with an access time of 1 microsecond. (U)

Fixed rod memories have the advantages of wired core memories plus flexibility of modification. Small magnetic rods are inserted in a wire mesh. An 8192-word memory with a 48-bit word has an access time of 0.15 microsecond. (U)

Some thin film memories now use a new alloy called Gyrallloy, which is deposited by vacuum evaporation on an oxidized aluminum substrate. The presence of a conductor so near the film gives a good signal-to-noise ratio and reduces drive impedance, since magnetic flux cannot penetrate into the conductor in the duration of the selection pulse. No harmful damping is produced in the magnetic reversal of this arrangement. Gyrallloy is deposited in a continuous film rather than spots. An insulating layer is applied over the film and printed-circuit copper conductors form the read winding. (U)

Thin magnetic film experiments have shown that information can be written, read, and regenerated in approximately 10 nanoseconds, a speed that approaches the light barrier. The light barrier occurs when the transit time of a signal from one side of the computer to the other becomes appreciable in terms of the speed of an arithmetic unit. For example, propagation across a distance of 30 centimeters takes 1 nanosecond, and dimensions may be 10 to 100 times greater than 30 centimeters. Therefore, numerous complexities will occur in designing faster computers; the ultimate arithmetic speed may be limited to 1 nanosecond for addition. The problems that remain to be solved for thin films are the reproducible production of matrices and the designing of an efficient access system. It is difficult to conceive and develop physical mechanisms for selecting, writing, and reading at speeds on the order of 1 nanosecond. Despite the small quantity of magnetic material present in any storage element, the output is still quite large as a consequence of the fast rise times of the interrogating pulses. (U)

The Cryotron is a device designed to apply the principle that in the presence of an applied magnetic field the temperature of incidence of superconductivity is reduced. Since niobium is less sensitive to the destruction of its superconductivity by applied fields than tantalum, a Cryotron flip-flop is possible, as shown in Figure 5-18. The circuit is cooled to 4.2 degrees Kelvin in boiling liquid Helium. Without current,

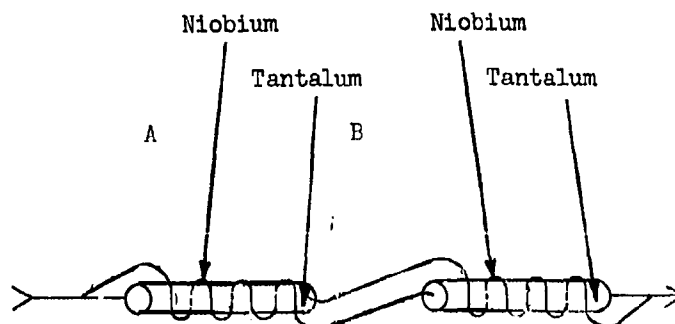


FIGURE 5-18. Cryotron Flip-Flop

all the materials are superconducting. As an applied current is increased, a point is reached where the field generated by the niobium forces the tantalum wire to assume its normal ohmic resistance. This point occurs in one side of the circuit before the other. Thus, two stable states are possible and the device behaves as a flip-flop. The speed of the original Cryotron is relatively slow (10 milliseconds), but it is possible to decrease this speed to 1 microsecond or less with vacuum deposited thin film elements. A variation of this technique is possible by vacuum depositing a thin superconducting film upon an insulating substrate. In this film are D-shaped openings separated by a narrow bridge. Data are inserted and read by means of wires along the bridges to which current is applied.

A current can flow in either direction along the bridge so that circulating currents can be produced by the magnetic fields generated by the wires. These persistors are reported to have operating times of 10 nanoseconds. (U)

Electron-mirror-microscopy may be the future equivalent of the file drum. The reading process is static, depending upon the deflection of the electron beam in an electron microscope by the magnetic field of the data recorded on a magnetic film. The inherently great magnifications power of the electron microscope makes the reading of closely packed data easy; the electron beam can be deflected electronically, thus facilitating high speeds. Non-destructive read-out is still present. Meyer has shown how information may be written on a magnetic medium with an electron beam by magnetizing the medium initially via an external field and heating small regions above the Curie point by a sharply focused electron beam.⁽³⁾ After cooling, an inversion effect takes place to enable the electron mirror microscope to detect the previously heated spots. Information densities of 10^5 bits per centimeter squared with writing speeds of 10^6 bits per second are possible. (U)

(c) Molecular Electronics - Molecular electronics is a new concept that seeks to integrate the functions performed by electronic circuits into a solid block of material. The objective is to rearrange the internal physical properties of the solid in such a way that phenomena occurring within or between domains of molecules will perform a function

(3) Meyer, Ludwig. - Electron Mirror Microscopy of Patterns Recorded in Magnetic Tape and Magnetic Writing with an Electron Beam.

ordinarily achieved by an assembly of electronic components. In addition to decreasing size and weight, increasing reliability, and reducing power requirements, molecular blocks could execute tasks now too complex to be performed economically by conventional methods. (U)

There have been three main approaches to solid-state microcircuits:

- (1) Microcomponent assembly--the construction of micromodules from microcomponents, miniaturized electronic components. Advantages are small size, light weight, and ruggedness; disadvantages are the number of parts and soldered connections and the problem of assembling such tiny components.
- (2) 2 - D Microminiaturization--the elimination of the thickness dimension by forming the active and passive circuit elements directly upon a thin insulating wafer or substrate. Capacitors, resistors, inductors, and other electronic components are processed with their interconnections upon an insulating ceramic wafer by a film depositing technique. This method is a step beyond microcomponent assembly.
- (3) Functional electronic blocks--the simple block diagram of electronics replaces the detailed circuit diagram as the working blueprint for constructing the electronic system. Each functional electronic block is a subsystem; together with a minimum number of connections they form the complete system. If the system is not complex, a single block may comprise the system.

Simple functional electronic blocks perform such functions as audio and video amplification, oscillation, multivibrator behavior, phase shifting, switching, and frequency discrimination. Semiconductor materials, particularly germanium and silicon, have been used to construct the blocks. Techniques employed in fabrication are: junction fabrication, etching, plating, alloying, cutting, masking, and film deposition. Success in molecular electronics rests upon a basic understanding of the innate

properties of materials and the ability to modify and exploit them usefully. A major advance has been the development of dendritic growth; germanium and silicon crystals are grown in the form of long, thin, optically flat, perfectly surfaced strips suitable for finished semiconductor devices. A second method is epitaxial growth, the vapor deposition of layers of silicon upon a silicon substrate, a technique useful for multiple junctions. (U)

Another new component is based upon the quantum mechanical tunneling of electrons into a vacuum. The estimated switching time is 0.1 nanosecond. This high-speed process may be suited for the economical production of a one cubic inch data processing system having 10^{11} active components. The solution of technological problems for such a system will probably take at least 15 to 20 years. (U)

Research in microelectronics includes work in the following fields:

- (1) Packaging techniques--including work on micromodules, which are ceramic wafers 0.31 inches square, each holding a circuit element (including diodes or transistors). Wafers are stacked. The thin ceramic wafers used to fabricate thermionic diodes are triodes designed to use ambient heat and internally dissipated power to heat the cathode.
- (2) Integrated circuits--this work studies evaporation techniques, including single crystal semiconductor substrates to produce active and passive circuit elements.
- (3) Functional electronic blocks--research into the production of complete circuits--particularly, the problems of interconnecting elements. Electron beam machining is used to fabricate high density (10^8 elements per cubic foot) thin film field-emission devices.

Continued research in these fields is essential so long as computing speed

is an essential requirement. The need for speed is strictly a function of the type of problems that require solution. But if the problems are such that months or years of computing are needed with existing computers, then microelectronics must be developed. The reason is simple: as the light barrier is approached, the distance between functional elements must be reduced in order to attain any additional increments in speed. (U)

Hydraulic elements can take several forms. As a spool valve, logical signals in the form of high and low pressure (corresponding to voltage/no-voltage levels in electronic binary logic) are applied to the inputs. In the diagram shown in Figure 5-19, a static medium pressure

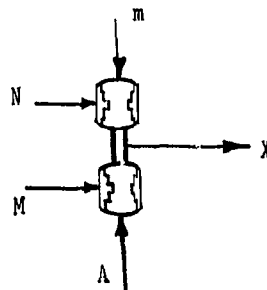


FIGURE 5-19. Basic Hydraulic Logic Elements
(3 inputs, 1 output)

is applied to m. Thus, pressure at A determines the position of the valve, which, together with input at M or N, defines the pressure at X. If high pressure is regarded as a logical 1 and low pressure as a 0, then $X = MA' + NA$. The logical capability of the hydraulic element is therefore greater than that of a single transistor. (U)

By providing a feedback path from X to A, a bi-stable element

is created. These two simple configurations, the gate and bi-stable device, are the basis for all other hydraulic logic elements. Gating networks, shift registers, counters, matrices, and multivibrators have been built and operated successfully. However, hydraulic logic presents many serious design problems. Elements behave like electronic counterparts in many respects. But transient peaks occur when current in an inductive circuit is suddenly stopped. There are also many physical effects related to fluid flow. Inertia is critical in determining response time, and conductor lengths must be equal for two parallel-fed elements to operate synchronously. However, the high reliability and long life of hydraulic components make the technique attractive. (U)

3. Computer Components and Techniques

The research projects described for storage devices are equally applicable for computer components, since the computer and its memory are intrinsically related. In the area of arithmetic and control units, new developments described in the fields of microminiature and molecular electronics and in hydraulic logic are applicable. As new elements are developed, they are likely to be exploited in all phases of computer equipment. (U)

An example of the application of these techniques is a saturated transistor switch. Increasing the speed of an adder is possible by reducing carry propagation time. If the addend digits at a given position are unequal, then the outgoing carry is identical to the incoming carry. Thus, a switch that allows a carry signal to propagate rapidly will save time. Even if the time required to set the switch is longer,

the switches in all positions can be set simultaneously, whereas the carry must propagate sequentially. If a saturated transistor is used as a switch, the impedance from emitter to collector is low and almost symmetrical under suitable circumstances. The propagated carry signal may be either a rising or falling edge, unless the carry is reset to 0 after each addition. The contribution of each switch to the carry propagation time can be reduced to 0.5 nanoseconds. (U)

Research and development in pattern recognition may also be directly related to computer development, especially in cases where there is an interplay between the pattern recognition device and the computer. Such a combination would facilitate research in automatic photo-interpretation. A single aircraft or satellite on a reconnaissance mission takes thousands of photographs that must be scanned, catalogued, and filed. If the pattern recognition device sampled with 100 strobes per inch and discriminated 16 levels of intensity, 250,000 bits of information distributed about each level would be recognized by a 5 by 5 inch photograph. With special logic circuits developed for this task, the computer could perform operations leading to the automatic analysis of the significant features in the photo. The computer could then store the photographic data in a special storage device. Subsequent photographs with the same coordinates could then be matched with the original; if significant differences were noted, they would be printed in parallel by a special printer. Ultimately, the analysis could include an estimate of the possible reason for the difference to aid an analyst in recognizing events of military significance. (U)

G. Summary

The objective of this equipment study was to establish basic equipment requirements for a Fact Correlation System. To attain this objective required an investigation of the capabilities of existing equipments. The various types of equipment were analyzed with respect to their ability to fulfill a set of assumed performance criteria relating to the volume of input, speed of operations, design features, reliability, and cost. (U)

The configurations of equipment were based upon parametric values and variables for the following factors:

- (a) Input reception rates.
- (b) Input processing rates.
- (c) Computer processing rates.
 - (1) Size of core memory.
 - (2) Size of program.
 - (3) Speed of computer.
 - (4) Type of auxiliary storage.

These configurations were then analyzed and evaluated on the basis of the following considerations:

- (a) Maintaining a balance between the input and processing rates within the system.
- (b) Determining the amount of equipment required to handle a particular function.
- (c) Establishing the minimum input and processing speeds required.
- (d) Ascertaining the ability of equipment to perform their assigned functions within a system framework.

The recommended state-of-the-art configuration selected requires a reading machine for input, a high-speed computer with a 64K core memory, and an auxiliary disc storage unit for processing (see Figure 5-11). (U)

The recommended configuration is essentially available now. The only research or development needed is in implementing the direct reading device. The full configuration should be operational within five years. (U)

The remainder of the equipment study investigated the basic principles and exploratory research in data processing equipment and techniques. This survey reveals the importance of developing improved input devices, including equipment and techniques for voice and pattern recognition. The need for speed is not pressing, but substantial improvements can be achieved with better switching circuits, storage units, and a critical reduction in size. Storage devices and access to them may be improved by using such techniques as thin films, molecular elements, and photoscopic storage. The ultimate storage unit for a Fact Correlation System would enable the entire file to be interrogated with a single set of instructions. (U)

The equipment requirements for a system ten to fifteen years from now depends upon the functional needs for linguistic transformation and adaptive learning. Once these requirements have been specified, several of the basic principles or exploratory developments may enable these needs to be fulfilled. (U)

H. References

- (1) Anelex Corporation; company literature on high-speed printers.
- (2) Bazilevskiy, Yu., Cybernetic Machines (AD 268055); Translation Services Branch, Foreign Technology Division, WP-AFB, Ohio, 29 November 1961.
- (3) Booth, Andrew D., "The Future of Automatic Digital Computers," Communications of the ACM, Vol. 3, No. 6; June 1960.
- (4) Bryant Computer Product; company literature on flying heads, magnetic storage drums, and disc files.
- (5) Burroughs Corporation; company literature on thin film memory.
- (6) Cheatham, T. E. and Collins, G. O., Programming Parallel Computation Sequences; Technical Operations Incorporated, Burlington, Massachusetts, August 1960.
- (7) Computing Reviews, Vol. 2, No. 2; March-April 1961: Review No. 630.
- (8) ----- Vol. 2, No. 4; July-August 1961: Review No. 914.
- (9) ----- Vol. 2, No. 5; September-October 1961: Reviews Nos. 1033, 1051, 1052, 1054, 1064.
- (10) ----- Vol. 2, No. 6; November-December 1961: Reviews Nos. 1135, 1136, 1197, 1203, 1204.
- (11) ----- Vol. 3, No. 1; January-February 1962: Review No. 1328.
- (12) ----- Vol. 3, No. 2; March-April 1962: Review No. 1621.
- (13) Data Display, Incorporated; company brochure.
- (14) European Information Technology (AD 250143); Auerbach Electronics Corporation, Philadelphia, Pennsylvania, 15 January 1961.
- (15) Farrington Electronics Incorporated; company literature and private correspondence pertaining to optical scanners.
- (16) Gabor, Andrew, and Comstock, George E., Performance Characteristics of High Density Digital Magnetic Tape Recording System; Potter Instrument Company, Incorporated, Plainview, New York, (Preprint of paper presented at 1961 IRE convention).

- (17) Goldberg, J., Green, M., Heckler, H., Van De Riet, E., Singleton, R., Frei, E., Multiple Instantaneous Response File; RADC-TR-61-233; Stanford Research Institute, Palo Alto, California, 1961.
- (18) Hayes, Dr. Robert M., Mathematical Models for Information Systems Design and a Calculus of Operations, RADC Contract No. AF30(602)-2111; Magnavox Research Laboratory (Contractor), Electrada Corporation, Los Angeles, California, Subcontract No. LX-17726, 27 October 1961.
- (19) Mathieu, J., and Barlen, S., Guiding Principles for Time and Cost in Documentation Work, TIL/T.4966; Germany, 1958.
- (20) Meyer, L., "Electron Mirror Microscopy of Patterns Recorded in Magnetic Tape," and "Magnetic Writing with an Electron Beam," Journal of Applied Physics, Vol. 29; 1958.
- (21) Millers, S. W., Fundamental Investigation of Digital Computer Storage and Access Techniques, RADC-TR-61-117A; Stanford Research Institute, Palo Alto, California, May 1961.
- (22) "Molecular Electronics," Computers & Automation, Vol. XI, No. 3; March 1962.
- (23) Newell, Feigenbaum, et al., "The Elements of IPL Programming," IPL V Manual; The Rand Corporation, Santa Monica, California, 1960.
- (24) Philco Corporation; company literature on asynchronous computers.
- (25) Pollack, Solomon L., The Role of Data Input in Automatic Data Processing Systems, Research Memo, RM-2681; USAF Project RAND, 9 December 1960.
- (26) Rabinow Engineering Company, Incorporated; company literature and private correspondence pertaining to matrix reading machines.
- (27) ----- Character Recognition Machines (Principles of Operation and 5 X 9 Fonts Report; Rockville, Maryland, January 1962.
- (28) Recordak Corporation; company literature on the DACOM system.
- (29) Richards, Paul, Parallel Programming; Technical Operations Incorporated, Burlington, Massachusetts, August 1960.
- (30) "Shoebox - A Voice Responsive Machine," Datamation, Vol. 8, No. 6; June 1962.

SECRET

- (31) Telex Incorporated; company literature on high-speed printers and mass memory modules.
- (32) "The Atlas Computer," Datamation; May 1961.

SECRET

DISTRIBUTION LIST

<u>Recipient</u>	<u>Copies</u>
*Commander, Rome Air Development Center Griffiss Air Force Base, New York Attention: RAAPT	10
*Commander, Rome Air Development Center Griffiss Air Force Base, New York Attention: RAALD	1
*Commander, Rome Air Development Center Griffiss Air Force Base, New York Attention: ROZMSTT	1
Headquarters, Electronic Systems Division Directorate of Technology Hanscom Air Force Base, Massachusetts Attention: Capt. LeRoy Ross, ESREE	3
Intelligence and Electronic Warfare Directorate Rome Air Development Center Griffiss Air Force Base, New York Attention: Mr. R. Webber, RAWIP	34

Best Available Copy



UNCLASSIFIED
DEPARTMENT OF THE AIR FORCE
AIR FORCE RESEARCH LABORATORY (AFRL)

28 AUG 06

MEMORANDUM FOR: HAF/ICIOD
1000 Air Force Pentagon
Washington DC 20330-1000

FROM: AFRL/IF
26 Electronic Parkway
Rome NY 13441-4514

SUBJECT: Mandatory Declassification Review (MDR) Request, Case 05-MDR-053
(Your Memo, dtd 27 Sep 2005, Same Subject)

1. Per your request, I have reviewed and completed the Mandatory Declassification Review (MDR) of the documents titled Fact Correlation for Intelligence Analysis, Volume 1, Applied Research Plan and Fact Correlation for Intelligence Analysis, Volume 2, Analysis of Technical Problems, each dated 15 Dec 62, RADC-TDR-62-461, Vol. I and RADC-TDR-62-461, Vol. II respectively. Both documents were written by Federal Electronic Corporation of Paramus NJ.

2. Based on my review I concluded the following:

a. A classification change occurred on 31Dec1974 as both documents were downgraded from Secret to Confidential.

b. Major portions of the documents' text are obsolete and describe hardware, systems, and technologies which are over 43 years old.

c. After reviewing the subject documents it was determined that no parts in either document should remain CLASSIFIED. The disclosure of the contents of either document is not expected to cause damage to US national security and there were no reasons why the UNCLASSIFIED portions should not be released.

3. This review was performed in accordance with Executive Order 12958, as amended and both documents were DECLASSIFIED: August 25, 2006. Please contact me immediately if you need further information related to this matter.

DONALD W. HANSON, SES
Director, Information Directorate

2 Atch

1. AD 354604, Vol I (U)
2. AD 354615, Vol II (U)

UNCLASSIFIED



DEPARTMENT OF THE AIR FORCE
AIR FORCE RESEARCH LABORATORY (AFRL)

04 January 2007

MEMORANDUM FOR DTIC-OCQ

ATTN: Larry Downing
Ft. Belvoir, VA 22060-6218

FROM: AFRL/IFOIP

SUBJECT: Distribution Statement Change

1. The following documents (previously unclassified/limited) have been reviewed and have been approved for Public Release; Distribution Unlimited:

AD354604, "Fact Correlation for Intelligence Analysis, Volume 1, Applied Research Plan", RADC-TDR-62-461, Volume 1.

AD354615, "Fact Correlation for Intelligence Analysis, Volume 2, Analysis of Technical Problems", RADC-TDR-62-461, Volume 2.

2. Please contact the undersigned should you have any questions regarding this change notification. Thank you for your time and attention to this matter.


KELLY CAGWIN
STINFO Officer
Information Directorate