Abstracts of Current Computer Literature

These abstracts and the associated subject and author indexes were prepared on a commercial basis under the direction of Dr. Geoffrey Knight, Jr., Consultant to Cambridge Communications Corp., 1612 K Street, Washington, D. C., who publish Information Processing Journal, Solid State Abstracts Journal, Electronics Abstracts Journal, Cumulative Computer Abstracts, Cumulative Solid State Abstracts, and Cumulative Electronics Abstracts.

CONTENTS

ABSTRACTS.................................................. Pages 511-517

Abstract Numbers

0) GENERAL; PEOPLE AND SOURCES; EDUCATION .......... 5655-5657
1) LOGIC AND SWITCHING THEORY; SEQUENTIAL MACHINES ........................................... —
2) DIGITAL COMPUTERS AND SYSTEMS .................. 5658-5665
3) LOGIC DEVICES AND CIRCUITS (HARDWARE) ....... 5666
4) DIGITAL STORAGE AND INPUT-OUTPUT EQUIPMENT .... 5667-5671
5) PROGRAMMING AND CODING OF DIGITAL MACHINES .. 5672-5685
6) HUMAN COMMUNICATION, DOCUMENTATION, AND HUMANITIES ..................................... 5686-5688
7) BEHAVIORAL SCIENCE, PATTERN RECOGNITION, AND ARTIFICIAL INTELLIGENCE .................. 5689-5690
8) MATHEMATICS ........................................ 5691-5699
9) PROBABILITY, MATHEMATICAL PROGRAMMING, DIGITAL SIMULATION, INFORMATION THEORY, AND COMMUNICATION SYSTEMS ........................................... 5700
10) SCIENCE, ENGINEERING, AND MEDICINE ............ 5701-5702
11) ANALOG AND HYBRID COMPUTERS ..................... —
12) REAL-TIME SYSTEMS AND AUTOMATIC CONTROL; INDUSTRIAL APPLICATIONS ..................... 5703-5704
13) GOVERNMENT, MILITARY, AND TRANSPORTATION APPLICATIONS ................................... —
14) BUSINESS APPLICATIONS OF INFORMATION PROCESSING ........................................... —

SUBJECT INDEX............................................ Pages 517-519
AUTHOR INDEX .............................................. Page 519
0) GENERAL; PEOPLE AND SOURCES; EDUCATION


The author surveys the history of computer development, concentrating on the dramatic progress over the past two decades. The rapid advance of hardware development during this latter period is outlined. Manufacturers of computers have now reached the stage where the major problem is in the area of software development. This "software explosion" is explored. It would appear that the speed frontier in computer technology has been crossed, but that the much more difficult problem of a "complexity barrier" must now be considered.

5656 Human Enhancement: Beyond the Machine Age, W. M. Brodeur (M.I.T., Cambridge) and N. Lindgren (Staff Writer); *IEEE Spectrum*, vol. 5, pp. 79–93, January 1968.

In a previous article the authors proposed a very broad evolving and ecological "think," urging designers to begin considering new types of systems, some to incorporate artificial intelligence, to work for the enhancement of human powers. In this article they describe some bases of that think, and attempt to join researches that, as they evolve, could form the ingredients of evolutions of systems. In the broadest sense, they attempt to interpret aspects of today's science and technology within the evolutionary framework, hoping to stimulate a dialogue among all kinds of system designers.


The report presents a picture of the M.I.T. Computer Center with emphasis upon present performance and its supervision. The actual configuration is assumed. After a presentation of the organizational framework, other batch-processing operations are briefly discussed (as a summary of many direct observations of the general operation of the Center). A detailed analysis of certain variables is made from data on time-sharing operations: system parameters; grade of service; and user's behavior. System is used to get further knowledge of the dynamics of behavior. Three models are presented; they use as input the results of the observations reported in the previous parts. The models represent three different viewpoints: real-time decisions (reaction to overload conditions); day-to-day operations (rules for balancing the batch-processing load and the time-sharing load); long-term study (several management attitudes for the next six-month interim period are studied).

Soviet High-Speed On-The-Fly Line Printer—see 5670.

Soviet Work in the Theory of Programming—see 5672.


A narrative discussion of a distributed control, multiple processor concept is presented. A possible evolution of the system organization is traced beginning with the Holland machine. A previous independent application-oriented derivation of the concept is cited. Comments on the relative significance of the derivations are offered. The discussion concludes that the proposed system organizational concept outweighs all others which are currently known and which are subject to the same constraints.


By an in-and-out conversion is meant that a floating-point number in one base is converted into a floating-point number in another base and then converted back to a floating-point number in the original base. For all combinations of rounding and truncation conversions the question is considered of how many significant digits are needed in the intermediate base to allow such in-and-out conversions to retain the original number (when possible), or at least to cause a difference of no more than a unit in the least significant digit.


This dissertation presents the results of a study of a class of transformations applicable to the operations of addition and subtraction in parallel digital computers utilizing redundant number representations in the arithmetic unit. The dissertation is organized into three main sections presented in sequence. The first section includes the definition, development, and study of a class of transformations on sets of real numbers. The second section is concerned with the application of the transformations to addition and subtraction. The primary function of the second section is the development, by an example, of a systematic procedure for finding some limited carry/borrow propagation adders subject to specified operand and sum/difference digit sets. The same procedure is applicable whatever the number of input operands may be. The third section presents the results of a study of some statistical properties of a few selected adders; the purpose of the study being to determine probabilistic distributions of sum digits given the distributions of operand digits, and to examine the effect of repeated uses of an adder on such distributions.


A recently developed deterministic procedure is described, and used to determine binary carry-save adder and borrow-save subtractor design. Designs believed to be new as well as those previously known are developed. A particular structure, expressed in terms of normalized digit sets, leads to three practical algebraic structures. Each of these in turn leads to nine (not necessarily unique) logical designs. Examples are given and possible application to variable field length systems are discussed.


The structure of a decision table which imparts self-diagnostic features to digital computers along with a method of implementing it is developed in this letter. Apart from considering the relative frequency of failure in the system, the procedure described includes an on-line assessment of the sequence of execution of the tests.


The digital differential analyzer proposed in the article contains only one integral, which assumes in succession during each integration step the role of all the necessary integrators; however, the information in the digital integrator is processed not sequentially, but in parallel code. Its advantages are claimed to be higher operating speed than possessed by the sequential–sequential differential analyzer, approaching the speed of the parallel–sequential operation (where there are as many integrators as are required for the solution of the problem), but requiring much less equipment. The author discusses the possible block diagram, for such an analyzer, the parallel-type digital integrator employed, the various individual circuit elements, and the scheme whereby the increments of the integrand are generated.
5664

The article describes the development of a special-purpose computer for statistical studies with the following requirements: 1) it must possess high accuracy and great high speed operation, which will allow considering it a standard statistical measuring informational system; 2) it must have a universal input device, allowing the input of data (registered in graphic form on tapes of various width), the execution of direct data input without preliminary registration, and, finally, the execution of recording and subsequent data input on a magnetic tape; 3) it must determine statistical characteristics of certain classes of nonstationary random processes; and 4) it must determine not only auto- and mutual-correlation functions, but also mathematical expectations and spectral densities. The first computer will be of the form of a laboratory computer occupying an area of about 2 m² and, being an experimental model, has four binary digits.

Simulating a Special-Purpose Stored-Program Digital Computer for Manipulating Formal Languages—see 5668.

5665

Oregon State University has designed and constructed a medium-speed serial-digital computer using glass delay lines circulating at 22 MHz as memory. The design objectives as originally conceived in a special seminar were: 1) to be a research project in computer design; 2) to be used as an educational machine; 3) to have easily modifiable hardware for basic research in computer systems design. An unusual arrangement of information within the 22-MHz memory allows a simple interface with the 340-kHz arithmetic unit, which results in an effective zero latency time and provides possibilities for an associative memory. The arithmetic unit has a command structure similar to large parallel machines and uses flip-flop arithmetic and control registers throughout. All hardware development has been aimed toward the concept of easy modification, elaborate console controls for effective man-machine interaction, and low cost.

3) LOGIC DEVICES AND CIRCUITS (HARDWARE)

5666

The report describes the design of six basic saturating logic circuits, including TTL, DCTL, DTL, and TTL for operation at microamperes current levels. The exact worst case design equations are derived and written for each circuit to include component tolerance, device variation with temperature, and immunity to external noise voltage. Mathematical expressions for power dissipation and average propagation delay time are derived from equivalent circuits under nominal operating conditions. With the aid of a digital computer, the range of valid worst case designs is determined. The evaluation of these designs for minimum power dissipation and minimum propagation delay time concludes that DCTL and TTL offer the best overall micropower performance.

4) DIGITAL STORAGE AND INPUT-OUTPUT EQUIPMENT

Scatter Storage Techniques for Implementing Symbol Tables of Assemblers and Compilers—see 5681.

Improved Hash Code for Better Scatter Storage—see 5682.

Congestion Analysis of Magnetic Core Storage Systems—see 5684.

5667

Applications of thin magnetic films in random-access memories of digital computing systems have received considerable attention recently because of their potential speed advantages for using hundreds of thousands, or even millions of these elements in a single computer dictates that careful attention be given to cost. This paper describes the production design package for a thin-film, 20 000-word rod memory array that offers nanosecond switching and reasonable fabricating cost. Rod fabrication, solenoid plane fabrication, solenoid plane assembly, rod insertion, electrical connections, and final assembly are described.

5668
A Continuous Film Memory Cell for Superconductive Associate Memories, H. G. Kaderet and K. Goer (Siemens, Munich); Proc. IEEE (Letters), vol. 56, pp. 121–122, January 1968.

A storage cell for an associative memory is described which offers a simpler structure and a higher packing density than cryotron storage cells.

5669

The program was devoted to the preparation and investigation of two novel kinds of electron-beam-addressable storage elements of submicron size and densely packed arrays of these elements. One such element, called a micron-cap, consists of an isolated micro-capacitor at the bottom of a hole in a metal/dielectric/metal film sandwich. The other element, called a micron-ring, consists of an isolated metal film ring embedded concentrically with a hole in the dielectric of a multilayer metal/dielectric film sandwich. Storage and read-out data on micron-cap elements were taken at low beam energies (near to the lower unity-crossover energy for secondary emission). System changes were made to allow higher sensitivity and higher speed operation, as well as to facilitate high-beam-energy investigations. The mildly bakeable (400°C) field emitter and molybdenum lens system and the associated commercial ultra-high-vacuum system have been assembled, the manipulators tested and found satisfactory, and beam measurements begun. Work on storage structures is being concentrated on attempting arrays with a commercial electron probe; capability of producing approximately 10 000 elements in a field has been shown. Attempts are underway to make micron-ring elements using the same electron probe to polymerize electron-sensitive resist in the critical step of forming the ring.


The development and test results of the first Soviet on-the-fly printer APM-1 are reviewed. The high-speed line printer uses a continuously rotating print wheel carrying 24 characters; they are selected by a 5-digit binary code. Fast-acting hammers print the characters. The printer is connected to the computer via a control unit which comprises a ferrite-diode logical element and semiconductor amplifiers. Functional and principal circuits of this unit are presented and their operation is briefly explained. During the two-year operation of an APM-1 on-the-fly printer prototype, no failure of a major component occurred.

Graphical Aids to Aerospace Vehicle Mission Analysis—see 5702.

5701

An efficient algorithm is presented for drawing or displaying conic section curve segments, each incremental move being calculated by the computer and the displacement from the intended curve. The inner cycle consists of three additions and one test for each move. Two further additions are required for display devices which
do not accept incremental commands, and two further tests are required to detect possible changes of sector.

Man–Machine Communication in Team Control Systems—see 5683.

5) PROGRAMMING AND CODING OF DIGITAL MACHINES

Software Explosion and the Complexity Barrier in Computer Technology—see 5665.


The report is a critical survey of Soviet efforts on the mathematical theory of computer programming and automatic programming methods (PP or programming programs). The study traces the development of the "operator" theory of A. A. Livapunov and his associates from its starting point in program schemes designed to represent specific problem-solving algorithms to its algebraic formulation in terms of the theory of categories. Other authors have attempted to adapt graph theory and the theory of algorithms to the construction of better programming languages. In contrast to FORTRAN, the practical result of PP has been to raise, rather than lower, the level of technical knowledge required for programming. Current Soviet research is directed toward adaptation and extension of ALGOL 60 rather than further theoretical work. Some of the Russian work, however, may be of practical relevance, particularly Glebov's synthesis of operators from measurable simpler ones.

Decision Table Approach to Self-Diagnostic Computers—see 5662.


Two exploratory experiments have been conducted to compare debugging performance of programmers working under conditions of on-line and off-line access to a computer. These are the first known studies that measure programmers' performance under controlled conditions for standard tasks. Statistically significant results of both experiments indicated faster debugging under on-line conditions, but perhaps the most important practical finding involves the striking individual differences in programmer performance. Methodological problems encountered in designing and conducting these experiments are described; limitations of the findings are pointed out; hypotheses are presented to account for results; and suggestions are made for further research.


This report describes a FORTRAN IV computer program which edits FORTRAN IV source programs by a) renaming any variable in the program and, if desired, by b) cleaning up the program. This consists of 1) renaming all statement numbers so that they increase sequentially, 2) moving all format statements to the end of the program, 3) deleting unnecessary blanks, 4) beginning every statement in column 7 and rewriting control statements so that they all conform to a certain format, 5) labeling every card uniquely. Any FORTRAN program is acceptable to EDIT for variable name changes. If the program is specified to use only those control statements as shown in Appendix B will be processed by EDIT.


In earlier years algebraic languages, list-processing languages, and string-manipulating languages have existed separately from one another. Recently, formula-manipulating languages have evolved, and in addition, there have been efforts to combine various different kinds of processing into one language. The design of Formula ALGOL represents an effort in this direction. Specifically, Formula ALGOL is an extension to ALGOL providing formula-manipulating, list-processing, and limited string capabilities. Thus, Formula ALGOL is a language in which the advantages of these various different kinds of processing are combined. It is anticipated that Formula ALGOL will be particularly well adapted to algebraic symbol-manipulating processes.


This paper proposes to apply the full power of the Compiler Compiler to an ALGOL-like base language, to provide an extendable language system that is one-pass and produces efficient object code. At its simplest, SNAP permits easily defined "formal macros" which allow conventional routine calls to be coded as open sequences tailor-made to the actual parameters. The paper discusses the problems created by this facility, in particular to ensure efficient code. It finishes with an illustration of the more powerful "informal macros" which use phrase structure notation in the routine headings and Compiler Compiler instructions in the routine bodies.


Simulation programming languages have been going through rapid evolutionary changes. Before 1959 there were no simulation languages — there were only simulation programs. Since 1959, when the first programming languages designed especially for simulation appeared, many different simulation modeling and programming systems have been proposed. At least five unequally different modeling schemes have found widespread acceptance and use. A large amount of modeling and programming experience has been accumulated which simulation language designers are now taking full advantage of. In this paper some theories of simulation modeling and programming are discussed and the design aims and a few of the language facilities of several "second generation" simulation programming languages are described. Finally, the author comments on a probable future for simulation programming.


An algorithm is described which constructs from a suitable BNF grammar an efficient left-right recognizer for sentences of the corresponding language. The type of recognizer, used in a number of compilers, operates with a pushdown stack and with a transition matrix. Two examples illustrate how such recognizers may be used effectively for other purposes besides the usual syntax checking.

5679 Toward a General Processor for Programming Languages, M. I. Halpern (IBM, San Jose); Commun. ACM, vol. 11, pp. 15–25, January 1968.

Many efforts have been made to develop a better way of implementing a higher level programming language than by the construction of a whole new compiler, but so far none has proved generally satisfactory. In this paper, it is contended that a programming language is best described functionally as a body of macro instructions, and that the macro call constitutes a canonical form in terms of which a programming notation may be described. A supporting discussion of the logical and historical role of the macro instruction is presented. Also discussed are the conflict between machine independence and object program efficiency, and the question of where the greatest difficulties lie in compiler construction.

This paper describes those features of the Atlas system which comprise the supervisor/compiler interface, with particular reference to how an Atlas Autocode user sees them.

Compiler for Analytic Algebraic Manipulation—see 5694.

Formula Manipulation Compiler for Symbolic Manipulation—see 5701.


Scatter storage techniques as a method for implementing the symbol tables of assemblers and compilers are reviewed and a number of ways of using them more effectively are presented. Many of the most useful variants of the techniques are documented.


A hash coding method based on fixed-point division rather than multiplication or logical operations is introduced. This new method allows the hash table to have almost any length. Also a new method of handling collisions is discussed. Known as quadratic search, this method is faster than random search and free from the "clusters" that build up with a linear search.


Current clustering programs (i.e., non-hierarchical classificatory programs) are examined, with particular reference to the internal consistency of the methods used for initiation, allocation, and reallocation. It is shown that almost all existing methods are open to serious objection, and that no method fully exploits the potentialities of such systems. The desirable properties of a clustering program are examined de novo, and suggestions are made for optimum lines of further development.


This paper studies the congestion problems of computer core storage units. The principle of storage request interleaving is briefly discussed as a means of increasing throughput. An analytic model is proposed to analyze the core storage system with a two-way interleaving feature. The model, an approximation to the physical system, is an extension of a queuing model with bulk service considered by Bailey.


Currently, both the hardware and software designs of many large computing systems aim at improved system performance through exploitation of parallelism in multiprocessor systems. In studying these systems, mathematical modeling and analysis constitute an important step towards providing design tools that can be used in building such systems. With this view the present paper describes a queuing model of a multiprocessor system operating in a job-shop environment in which arriving jobs consist of a random number of segments (sub-jobs). Two service disciplines are considered: one assumes that the sub-jobs of a given job are capable of parallel operation on different processors while the other assumes that the same sub-jobs must be operated in a strictly serial sequence. The results (in particular, the mean number in the system and waiting time in queue) obtained for these two disciplines are shown to be bounds for more general job structures.

6) HUMAN COMMUNICATION, DOCUMENTATION, AND HUMANITIES

Simulating a Special-Purpose Stored-Program Digital Computer for Manipulating Formal Languages—see 5686.

Sentence Recognizer Operating with a Pushdown Stack and a Transition Matrix—see 5678.


An extension of the Backus notation used in the ALGOL report is described which permits both the syntax and semantics of general languages to be specified readily and compactly. A mechanism based on that of Metcalfe is explained for performing the indicated operations automatically, and a description is given of an ALGOL program which simulates this mechanism on a KDF9 computer.


The transmittal of engineering language to a computer for document processing is examined, in particular, the organization of a coding dictionary and the selection of code words for such transmittal. These code words or "mnemonics" are of two types, "verbs" and "nouns." Verbs require the computer to perform certain processing actions and nouns transmit standard phrases, sentences, and paragraphs of engineering language to the computer. Rules for the organization of a noun-mnemonic dictionary are presented. Four principal requirements which the mnemonics listed in the dictionary should satisfy are described. These requirements include 1) adherence to the computer sorting order, 2) adherence to the plan of the document being processed, 3) provision of adequate coding range for dictionary expansion or change, and 4) coding of the standard phrases, sentences, and paragraphs in an easily remembered manner. The dictionary organization for a specimen document is examined and specific recommendations are made.


Digital computers are widely used in the mechanical operation of publishing; one application may be to one of the classes of editorial functions which requires no judgment—the detection and correction of misspelled words. This paper describes a computer program written to correct spelling errors through vector representation. Words are uniquely identified through their vectors, and comparison of the vectors of input words with stored vectors permits the detection and correction of four common types of spelling errors. The program is considerably more efficient than programs which use dictionary look-up and letter-by-letter comparison of the stored word with the input word. With appropriate vectors, a similar method could be used to scan sentences and perform other editorial functions.

Clustering Programs for Classificatory Sorting—see 5683.

7) BEHAVIORAL SCIENCE, PATTERN RECOGNITION, AND ARTIFICIAL INTELLIGENCE


It is common in automatic pattern recognition to recognize an unknown pattern as the same as the known pattern to which it is most similar. This paper explores the idea that if appropriate pairs of parts are chosen, then the similarity of a pair of patterns is the sum of the similarities of these pairs of parts. In a simplified recognition problem, it is found that appropriate pairs of parts are chosen automatically when a purely algebraic technique is used for inferring the similarities of unknown to known patterns. This is a critical step in a research program aimed at automatically finding suitable systems of features for recognizing hand-printed and cursively written characters.

In this report, two problems in pattern recognition are considered. 1) When the likelihood ratio criterion results in a quadratic discriminant function, it may be possible to factor the quadratic form into two linear discriminant functions. It is shown that the probabilities of misclassifications are the same for both the linear discriminant functions and that both of them should be used for classification. 2) When the likelihood criterion results in a linear discriminant function, it is shown that the estimate of the "squared distance" has a positive bias and hence the probability of misclassification is underestimated.

Development of Systems to Enhance Human Powers—see 5656.

8) MATHEMATICS

Drawing Ellipses or Hyperbolae with a Digital Plotter—see 5671.


In this paper the significance of normality of a function \( f(x) \) with regard to \( T \) approximation for the numerical treatment of the problem is set forth. Assuming \( f(x) \) to be normal, the Lipschitz-continuous dependence of the Chebyshev approximation \( T[f] \) on \( f(x) \) is shown. Under the same assumptions, a detailed proof of the convergence of the Remez Algorithmus for a sufficiently good initial approximation is provided. The convergence is at least linear. Finally the discrete \( T \) approximation of \( f(x) \) (approximation on a finite point set of the interval) is compared with \( T[f] \). Qualitative results are given.


A general definition of what is meant by "total-step" "single-step," and "successive relaxation iterative method" is given and these concepts are applied to systems of linear equations. In the special case of a matrix with zero diagonal entries the well-known Jacobi, Gauss-Seidel, and relaxation iterative methods are obtained. Theorem 1 gives conditions for the convergence of the relaxation iterative method for general, non-negative matrices. The proof is similar to that given by Stein and Rosenberg for a special case. A corollary gives conditions for the convergence of the relaxation iterative method for non-negative matrices. A theorem about the convergence of the relaxation iterative method with diagonally dominate matrices is also proved.


The problem of generating a given sparse matrix \( A \) into a block diagram form (bdf) and the subsequent transformation of each of the block diagonal matrices into as nearly upper triangular form (uft) as possible, by using only row and column permutations, is discussed. It is shown how some of the results from Graph Theory can be used to transform \( A \) to the bdf. In order to transform the block diagonal matrices into the uft, two methods are described, one of which makes use of linear programming while the other uses approximate probabilistic arguments. The latter method, in relation to the computational effort, yields significant results in practice.


A compiler is described which is capable of performing several analytic algebraic manipulations of generally structured expressions through the transformation of a phrase structure analysis record. The method is described, some example problems are given, and the advantages and limitations of the approach are discussed.

Programming Language for Algebraic Symbol Manipulation—see 5675.

Algebraic Inference of Pattern Similarity—see 5689.


In an important contribution Powell has suggested upper bounds for determining the unconstrained minimum of a function of several variables, and determining it without calculating derivatives. This paper studies his approach in some detail. It is first shown by counterexample that his basic method for minimizing a quadratic function in a finite number of iterations contains an error. His modification of his basic method is then simplified, and the simplification is proven to converge for strictly convex functions. Finally, a new method is proposed which not only converges in a finite number of iterations for a quadratic, but also for which theoretical convergence is established in the strictly convex case.


All published error coefficients for estimating quadrature errors for analytic functions were computed on the assumption that the quadrature rule was exact for polynomials up to a given degree. Since these rules use rounded values for the abscissas and weights and since the true values of the intervals of some of the polynomials in question have an infinite binary expansion, the quadrature rule is not exact. Hence these errors must be taken into consideration in computing practical error coefficients.


Gaussian-type formulas are derived for all values of \( N \geq 2 \).


A numerical integration formula is presented which uses unequal sampling intervals. The intervals are equally spaced on a log scale. Such a formulation is useful in Fourier analysis to improve accuracy and ease of usage. A complete set of formulas for numerical Fourier analysis is given.


It is shown that a (multifurcating) ordered rooted tree of \( n \) points has a unique corresponding bifurcating pendant ordered rooted tree of \( n \) end points and vice versa. It is also shown that one can be manipulated into another and vice versa.

9) PROBABILITY, MATHEMATICAL PROGRAMMING, DIGITAL SIMULATION, INFORMATION THEORY, AND COMMUNICATION SYSTEMS


AD 660 953 CFSTI $3.00.

The report discusses the statistical problems that arise in computer simulation experiments. Three problems areas inherent in all stochastic system simulation models are discussed: verification, which determines whether a model actually behaves as an experimenter assumes it does; validation, which tests whether the model reasonably approximates a real system; and problem analysis, which seeks to ensure proper execution of a simulation and proper handling of its results. The study traces the elements of a simulation experiment from initial conception to analysis of final results, defining the statistical problems that arise at each step and relating them to the formal body of statistical theory. Since the aim is to promote awareness of problems, not to solve them, the study offers no general solutions but provides references germane to the statistical problems described.

Special-Purpose Computer for Statistical Studies—see 5664.
ABSTRACTS

Probabilistic Matrix Transformation—see 5693.

Review of Simulation Programming Languages—see 5677.

Simulating a Special-Purpose Stored-Program Digital Computer for Manipulating Formal Languages—see 5686.

Linear Programming Matrix Transformation—see 5693.

Queuing Model of a Multiprocessor System—see 5685.

10) SCIENCE, ENGINEERING, AND MEDICINE


This report applies a new tool that can symbolically perform mathematical operations such as differentiation, substitution, expansion of terms, etc. The tool is an experimental symbolic manipulation system called FORMAC (FORmula MANipulation Compiler) which utilized the IBM 7090/94 computer. Applications of FORMAC computer programs to problems in structural mechanics are presented.


This paper describes an interactive computer graphics system, called Graphic ROCKET, currently being developed as a performance analysis aid for the aerospace vehicle systems analysis process. It is intended to provide a means for rapidly specifying and evaluating the performance of a wide range of aerospace vehicle designs and flight plans. This capability will enable the aerospace mission analyst either to get his job done much faster or to consider a larger number of alternatives for accomplishing his mission objective.

12) REAL-TIME SYSTEMS AND AUTOMATIC CONTROL; INDUSTRIAL APPLICATIONS


This paper deals with some of the problems met within real-time control systems in which the computer is acting as an executive in its own right, albeit in a slightly junior capacity to the man. For the purposes of this paper these systems are defined as team control systems.


Many applications of CPM and PERT techniques involve the laborious procedure of drawing and redrawing “networks” or “flow charts.” The utility of a computer program that would automatically draw these flow charts is apparent. To be useful, PERT diagrams must be easy to read. In this program, events and activities that form a subnetwork are grouped together and the proximity of any two subnetworks reflects their interdependence. Events and activities in one subnetwork are not separated by the elements of another subnetwork. The horizontal axis of the diagram is a critical path along which the events may be time-scaled.

14) BUSINESS APPLICATIONS OF INFORMATION PROCESSING

Construction of PERT Flow Charts—see 5704.

SUBJECT INDEX
Formula Manipulation Compiler for Symbolic Manipulation 5701
Graphical Aids to Aerospace Vehicle Mission Analysis 5702
Construction of PERT Flow Charts 5704
—see also Control, Education, Mathematics, Patterns

Computer Systems
Extended Holland Machine System 5658
Distributed Control Multiple Processor System Organization 5658
Supervisor/Compiler Interface of the Atlas System 5680
—see also Digital Computers

Computers
History of Computer Development 5655
Software Explosion and the Complexity Barrier in Computer Technology 5655
Operational Analysis of a Computer Center 5657
—see also Computer Systems, Digital Computers

Control
Distributed Control Multiple-Processor System Organization 5658
Man–Machine Communication in Team Control Systems 5703

Conversors; Conversion
In-and-Out Conversions in Floating-Point Arithmetic 5659

Cryotrons; Cryogenic
—see Superconductivity

Data Processing
—see Computer Systems, Digital Computers, Education, Multiprocessing, Processing

Data Transmission
—see Codes

Debugging; Checkout
Experimental Comparison of On-line and Off-line Debugging Performance 5673

Decision Tables
Decision Table Approach to Self-Diagnostic Computers 5662

Delay Lines
Nebula Computer Using Glass Delay Lines 5665

Design
—see Logic Design

Diagnosis
Decision Table Approach to Self-Diagnostic Computers 5662
—see also Error

Derivatives
Minimizing Functions without Calculating Derivatives 5695

Digital Computers
Serial–Parallel Digital-Differential Analyzer 5663
Special Purpose Computer for Statistical Studies 5664
Nebula Educational Computer 5665
Simulating a Special-Purpose Stored-Program Digital Computer for Manipulating Formal Languages 5686
—see also Computer Systems, Processing

Editing
Editing FORTRAN IV Source Programs 5674

Education
Nebula Educational Computer 5665

Electron Beams
Arrays of Electron Beam Addressable Storage Elements 5669

Error
Error Coefficients for Estimating Quadrature Errors for Analytic Functions 5696

Failures; Faults
—see Diagnosis

Files
—see Magnetic Films

Flow Charts (Diagrams, Graphs, Tables)
Construction of PERT Flow Charts 5704

Graphics
Graphical Aids to Aerospace Vehicle Mission Analysis 5702

Graphs; Graph Theory
Graph Theory Matrix Transformation 5693
Correspondence Between Two Types of Trees 5699
—see also Flow Charts

Information
—see Codes

Integration; Quadrature
Error Coefficients for Estimating Quadrature Errors for Analytic Functions 5696
Gaussian-Type Numerical Integration Formulas 5697
Numerical Integration Formula Useful in Fourier Analysis 5698

Interaction
—see Man–Machine Communications

Languages
Simulating a Special-Purpose Stored-Program Digital Computer for Manipulating Formal Languages 5686
Mnemonic Dictionary for Computer-Aided Engineering Writing 5687
—see also Programming Languages

Learning
—see Artificial Intelligence

Linear Programming
Linear Programming Matrix Transformation 5693

Logic
Performance of Micropower Transistor Logic Circuits 5666

Logic Design
Logical Design of Carry-Save Adders and Borrow-Save Subtractors 5661

Macro
SNAP Language with Formal Macro Facilities 5676
Macro Instructions and Compiler Construction 5679

Magnetic Cores
Congestion Analysis of Magnetic Core Storage Systems 5684

Magnetic Films
Thin Magnetic Film Rod Memory Array Production 5667

Man–Machine Communications (Systems)
Man–Machine Communication in Team Control Systems 5703

Mathematical Programming
—see Linear Programming

Mathematics
Drawing Ellipses or Hyperbolas with a Digital Plotter 5671
Numerical Integration Formula Useful in Fourier Analysis 5698

Matrices
Using Transition Matrices in Compiling 5678
Convergence of Relaxation Methods of Solving Linear Equations and their Matrices 5692
Row–Column Permutations of Sparse Matrices 5693

Memory
—see Storage

Minimization
Minimizing Functions without Calculating Derivatives 5695

Models
Queueing Model of a Multiprocessor System 5683
Stochastic Simulation Models 5700

Multiprocessing
Queueing Model of a Multiprocessor System 5685

Numbers
Addition and Subtraction in Parallel Digital Computers Using Redundant Number Representation in the Arithmetic Unit 5660

Numerical Analysis
—see Integration, Matrices

Optimal
—see Minimization

Parallel
Addition and Subtraction in Parallel Digital Computers Using Redundant Number Representation in the Arithmetic Unit 5660
Bounds on Parallel Processing of Queues with Multiple-Phase Jobs 5685

Patterns
Algebraic Inference of Pattern Similarity 5689
Linear and Quadratic Discriminators for Pattern Recognition 5690

Plotters
Drawing Ellipses or Hyperbolas with a Digital Plotter 5671

Printers
Soviet High-Speed On-the-Fly Line Printer 5670

Probability
Probabilistic Matrix Transformation 5693
—see also Statistics

Processing; Processors
Distributed Control Multiple Processor System Organization 5658
General Processor for Programming Languages 5679
Bounds on Parallel Processing of Queues with Multiple-Phase Jobs 5685
—see also Associative, Computers, Macro, Multiprocessing

Programmers
Experimental Comparison of On-Line and Off-Line Debugging Performance 5673
ABSTRACTS OF CURRENT COMPUTER LITERATURE

PROGRAMMING

Experiment Comparison of On-Line and Off-Line Programming Performance 5673
—see also Linear Programming

PROGRAMMING LANGUAGES

Editing FORTRAN IV Source Programs 5674
Preliminary Sketch of Formula ALGOL 5675
Programming Language for Algebraic Symbol Manipulation 5673
SNAP Language with Formal Macro Facilities 5676
Review of Simulation Programming Languages 5677
Sentence Recognizer Operating with a Pushdown Stack and a Transition Matrix 5678
General Processor for Programming Languages 5679

PROGRAMS

Editing FORTRAN IV Source Programs 5674
Experimental Comparison of On-Line and Off-Line Programming Performance 5673
Use of FORMAC Programs in Symbolic Formulation and Solution of Engineering and Scientific Problems 5701
Simulating a Special-Purpose Stored-Program Digital Computer for Manipulating Formal Languages 5686
Construction of PERT Flow Charts 5704

QUEUES

Queuing Model of Magnetic Core Storage Systems 5684
Bounds on Parallel Processing of Queues with Multiple-Phase Jobs 5685

RANDOM PROCESSES

—see Stochastic Processes

REAL-TIME

—see On-Line

AUTHOR INDEX

A
Apostolatos, N. 5692
B
Behrendt, D. L. 5667
Boehm, B. W. 5702
Boles, J. A. 5665
Brittan, J. N. G. 5703
Brodey, W. M. 5656

C
Chang, W. 5684
Cheeves, V. L. 5665
Coffman, Jr., E. G. 5685

D
Di Paola, R. A. 5672
Domaratzki, A. N. 5664
Dunn, R. M. 5658

E
 Erikson, W. J. 5673

F
 Ferreira, S. K. 5701
 Fishman, G. S. 5700

G
 Giangardella, J. J. 5688
 Goser K. 5668
 Grant, E. E. 5673
 Gries, D. 5678

H
 Haek, J. N. 5656
 Halpern, M. I. 5679
 Hoselton, G. A. 5665
 Howden, W. E. 5704
 Hudson, J. F. 5688

I
 Iturriaga, R. 5675

J
 Kadereit, H. G. 5668
 Kalyaev, A. V. 5663
 Kelly, J. 5669
 Khodakov, V. E. 5670
 Kiviat, P. J. 5677, 5700
 Korshikov, R. V. 5663
 Kulisch, U. 5692

L
 Lance, G. N. 5683

M
 Matula, D. W. 5659
 Maurer, W. D. 5682
 Mccains, F. 5674
 Meindl, J. D. 5666
 Meisel, W. S. 5698
 Morris, J. E. 5687
 Morris, R. 5681

N
 Napper, R. B. E. 5676
 Nievergelt, J. 5655

O
 Obruca, A. K. 5699

P
 Perls, A. J. 5675
 Phillips, G. M. 5697
 Pitteway, M. L. V. 5671

R
 Rabinowitz, P. 5696
 Rao, P. S. R. S. 5690
 Raynaird, T. G. 5657
 Reeves, C. M. 5686
 Rieber, J. E. 5702
 Robertson, J. E. 5661
 Rogers, K. T. 5669
 Rogoff, B. L. 5665
 Rohatinski, F. A. 5660
 Rohleder, J. S. 5680
 Roper, R. S. 5688
 Rosenthal, L. E. 5694
 Scatter Storage Techniques for Implementing Symbol Tables of Assemblers and Compilers 5681
 Improved Hash Code for Better Scatter Storage 5682
 Congestion Analysis of Magnetic Core Storage Systems 5684
 Simulating a Special-Purpose Stored-Program Digital Computer for Manipulating Formal Languages 5686
 —see also Arrays, Associative

SUPERCONDUCTIVITY

Continuous Film Memory Cell for Superconductive Associative Memories 5668

SUPERVISORS

Supervisor/Compiler Interface of the Atlas System 5680

SWITCHING FUNCTIONS

—see Logic

SYMBOLIC; SYMBOLS

Programming Language for Algebraic Symbol Manipulation 5675
Scatter Storage Techniques for Implementing Symbol Tables of Assemblers and Compilers 5681
Simulating a Special-Purpose Stored-Program Digital Computer for Manipulating Formal Languages 5686
Compiler for Symbolic Algebraic Manipulation 5694
Use of FORMAC Programs in Symbolic Formulation and Solution of Engineering and Scientific Problems 5701
Formulam Manipulation Compiler for Symbolic Manipulation 5701

SYSTEMS

Development of Systems to Enhance Human Powers 5656
—see also Computer Systems

T

TRANSISTORS

Performance of Micropower Transistor Logic Circuits 5666

TRANSLATORS

Syntax-Directed Translator 5686