DEC 6 1967

COO-614-57

ICR QUARTERLY REPORT

Number 14
August 1, 1967

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

memo
nyo.

THE UNIVERSITY OF CHICAGO

THE INSTITUTE FOR COMPUTER RESEARCH

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

The Institute for Computer Research

The University of Chicago

ICR Quarterly Report

Number 14

August 1, 1967

PATENT RIGHTS AND OTHER PROCEDURES
GOVERNING PATENT RIGHTS AND RELEASE ARE ON
FILE IN RECEIVING SECTION.

Prepared under the auspices of AEC Contract No. AT(11-1)-614.

Victor H. Yngve, Principal Investigator

PATENT RIGHTS AND OTHER PROCEDURES
GOVERNING PATENT RIGHTS AND RELEASE ARE ON
FILE IN RECEIVING SECTION.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

Reg per Phil

Staff of the Institute for Computer Research

Naomi Alton

Robert Ashenhurst

Jurgen Bounin

Dolores Brennan

William Brown

Kenneth Burns

Rachel Chakrin

Elaine Chow

Charles Fischer

Etta Franklin

William Georgas

Robert Graves

Richard Herwitt

Leonardo Herzenberg

Paul Kosinski

Hirondo Kuki

George Kreisheimer

Vincent Kruskal

Glenn Manacher

Richard Miller

Lily Monheit

John Munn

Michael Neumann

Daniel Norman

Alex Orden

Walter Petryshyn

Charles Robinson

Clemens Roothaan

Lyle Settle

John Shepherd

Cheng-kiat Teng

Serena Torres

Nancy Trombetta

Michael Williams

Augustine Witt

Victor Yngve

Table of Contents:

Section I

- A. Maniac III Engineering Activity
- B. The Man-Machine Project (C.C.J. Roothaan)
- C. An Investigation of a Graph Simulator Computer (J. Bounin & C. Robinson)

Section II

- A. A Bench Mark Calculation for Retrieval by Confluence of Binary Attributes in a Random-Access Computer (G.K. Manacher)

Section III

- A. Numerical Stability in N-Body Integrations (R.H. Miller)

MANIAC III ENGINEERING ACTIVITY

1. Maniac III

Some improvements have been made to the ventilation system.

2. Ampex 16K Memory

A faulty core has revealed itself in the Ampex Memory Stack.

Fortunately this memory has a 50-bit word, which permitted use of a spare bit to be made.

There have been a number of transistor failures, apparently unrelated, in this equipment. Some have been attributed to low power transistors not being tightly secured to their heat-sinks.

3. Interface Computers

There has been much further discussion of schemes for internal logic structure and instruction set, although no further logical design work was done during this period. It was possible to make some progress on the physical structure.

4. PDP-8 Computer

Programs for automatic testing of Interface Computer circuit cards are being debugged.

5. Pattern Manipulator

Adjustment and testing of circuits associated with the SEC Vidicon tube continues.

6. General

Interface computer specifications are approaching finalization.

The symbolic editor program is completed, and it may be run in

Maniac III or in the PDP-8 which interrupts Maniac as required.

A small-scale model of a graph simulator was built for the purpose of testing some of the features of this special purpose computer (see this Report, Section I-C).

MANIAC III USE

July 14 - October 14

(Programmer and production use; debugging and production times are not separated in this table.)

Systems Work	68.0
Engineering Computer Time	20.5
Language Studies (Robinson)	26.5
Investigations of Projective Planes (Manacher)	12.4
Interface Computer Project	112.5
PERT Studies	5.3
Astronomical Calculations (Miller)	25.2
Student Time	175.7
Miscellaneous	44.2
Central Shop	<u>14.5</u>

Total 504.8

THE MAN-MACHINE PROJECT*

C.C.J. Roothaan

Progress Report No. 10

(for the period 1 April 1967 through 30 June 1967)

System/360 Mathematical Subroutines

Mr. Kuki's work on upgrading the mathematical subroutines is being done in cooperation with members of the SHARE Numerical Analysis Committee. Completion is expected in about two months.

Scientific Subroutine Package for System/360

Mr. Kuki has scrutinized the Exponential Integral, the Gamma Function, the Legendre Polynomials, and the Chebyshev Polynomials.

System/360 Data Management

Messrs. Kosinski, Kruskal, Sevcik and Williams have outlined a "universal" file structure which is similar to OS/360 "indexed sequential" file structure. This structure will serve both as a conventional "sequential" structure for CHAOS or ASP type input/output files, etc., and as a "direct access" structure for on-line maintenance of data and program files. The detailed design of this file structure is now being considered. We are especially intent upon a design which uses storage economically without sacrificing speed.

*Work on this project, supported by a grant from the International Business Machines Corporation with C.C.J. Roothaan as principal investigator, is carried on under the joint auspices of the Institute and the Computation Center.

POOR (Page Oriented Output Routine)

7094 POOR: Mr. Sherrard made several minor changes in the titling and editing parts. The problem of making POOR accessible to FORTRAN users is being studied. Mr. Eicher has finished a first version of the writeup.

360 POOR: Mr. Sevcik's energies have been diverted from POOR to work on the System/360 Data Management project.

CHAP (CHaos Assembly Program)

Miss Monheit has finished coding the loader and has started debugging it. A few bugs related to relocation are being ironed out in the assembler.

Task Scheduler and Setup Processor

Mr. Kubaitis has completed work on the task scheduler and setup processor. The scheduler uses a complex algorithm, based on concepts developed by Mr. Kruskal, which, much like telephone switching systems, maximizes the set of users receiving "reasonable service", rather than rigidly optimizing the service of a subset of users. In selecting a task for processing, the scheduler considers the time required to process the task, the time that the task has been waiting for processing, and the current work load on the system.

The task setup processor is a single re-entrant program capable of servicing all operator communication stations simultaneously. The setup

processor, taking its cues from the task scheduler, relays setup information for a task to the appropriate operator shortly before the time the task would have been processed had it not required setup. The setup processor awaits operator acknowledgement of the setup instructions before allowing the task to be processed.

AN INVESTIGATION OF A GRAPH SIMULATOR COMPUTER

J. Bounin and C. Robinson

Introduction

Many practical problems can be reduced to finding the path through a weighted graph such that this path satisfies certain constraints. In this paper, we shall investigate a new type of hybrid computer which we call a "graph simulator computer" (abbreviated gsc). It is designed to solve problems such as finding the minimum or maximum path between two nodes of a graph. The two fixed nodes at the beginning and ending of a path are called the source and the sink nodes respectively for that path, and the path that satisfies the constraints desired in the problem is called the critical path. Advantages of a gsc, over conventional digital computer analysis for finding the critical path, lie in its increased speed and in its representation of both input and output in a form suitable for either visual or electromechanical processing.

Section 1 of the paper describes the electrical analog of the weighted graph. Sections 2 and 3 then discuss two different applications of a gsc to practical problems: the routing of a long distance telephone call and the analysis of "extended" PERT type networks. Section 4 examines a proposed small scale model of a gsc which we intend to build, and section 5 presents our conclusions.

1. The Electronic Model

A weighted graph will be represented by an equivalent electrical

network. The two kinds of elements of a graph, namely arcs and nodes, will be represented by their corresponding arc and node circuits (Figure 1). The weight of an arc in a graph will be simulated by a time delay in the arc's circuit. The time it takes a signal to travel from the source node to the sink node along some path in the network will be the analogue of the weight attached to that path in the graph.

Let us consider a weighted graph in which we want to determine the minimum path between two nodes. In this case, all nodes in the network must act as OR functions, i.e., the time delays on the outgoing arcs from a node must be started as soon as the first incoming arc to that node satisfies its time delay. When this condition has been met, i.e., a node's outgoing arcs have been started, we shall say that the node has been selected. The diagram in Figure 2 gives a simplified arc-circuit which will work in this case. When initialized, the flip-flop is in the ZERO state, and both the input and output are at ZERO signal level. When the signal level is raised at the input, the time delay ΔT passes before a signal is received at the AND gate. This gate in turn allows the flip-flop to be set to ONE if and only if the signal at the output (node) is not already ONE, i.e., if the output node has not already been selected by another arc. Thus, when the signal finally reaches the sink node, a tree of flip-flops will have been selected, as shown in Figure 3, one branch of which will be the critical path.

Figure 4 demonstrates how the backtracking from sink to source (pruning the tree) can be done electronically in order to establish the

critical path. The rule is that the incoming arcs of a node should be reset if and only if none of the outgoing arcs from that node are set. Figure 4A gives a possible method of doing this using two junctions per graph node. The circuit in Figure 4B accomplishes the same function using only one junction per graph node, at the cost of increasing the arc circuits' complexity. Essentially this is accomplished by logically having two additional gates which give the circuit the direction of signal travel, Solution 4A is obviously cheaper when the graph's topology is fixed, while 4B makes it easier to vary the wiring.

To find the maximum path between two nodes each node should work as an AND function; i.e., a node should be selected only after all incoming arcs have been set. Since, however, an electrical connection provides a natural OR function, it is cheaper to slightly redesign the arc circuits and use de Morgan's Law that $\overline{a \wedge b} = \overline{a} \vee \overline{b}$. Since we already have a means of providing the OR function, all that remains is to invert the input and output signals as shown in the circuit in Figure 5. An output node remains in the ZERO state until all the incoming arcs to that node have been set. At this point in time the pulse amplifier sends a reset pulse to all the incoming arcs' flip-flops; by appropriate timing, the "reset" pulse will be blocked by the still present "set" pulse on the last arc that was set. This leaves the record of which incoming path took the longest to reach a node. It is further necessary to invert the control signals received by the source node and sent by the sink node.

Maximum and minimum nodes can be mixed within a network by adding some switches as shown in Figure 6A. The rule is that for an AND-maximum node all switches immediately before and after the node must be closed, while for an OR-minimum they must be open. Figure 6B gives such a network showing the positions of the switches.

There remains the possibility of a race condition between arcs terminating in the same node. If it is significant to avoid this ambiguity, a priority reset can be added such that only one arc of an incoming group can remain set (Figure 7).

A complete arc circuit is shown in Figure 8. The input to the arc is received from an AND node, and the arc terminates in an OR node. Method B is used for backtracing.

The simplest way to produce a node is by making the node a solid electrical connection, such as a solder joint. If manual setup of the graph's topology is desired, phone plug connections can be used; however, if the graph's topology changes frequently, one must resort to computer control of the topology. If the most general graph must be provided for, then the switching involved must be able to connect any arc with any node. This can be accomplished by the use of the crossbar system shown in Figure 9. The nodes are represented by the vertical lines in the figure, while the input and outputs of the arcs form two sets of busses perpendicular to these. The system forms a lattice of $2 \times m \times n$ intersection points, where m is the number of arcs and n the number of nodes. At every intersection point there is a switch, and we can now connect any arc to

any node. An example is shown in Figure 10.

Since at most $2n$ switches will be closed at any given time, this approach is rather wasteful. It should be possible to improve on it by finding an algorithm for setting up an m -arc graph such that some of the $2mn$ switches are never needed.

Control circuitry will be needed to supply an input pulse to the source, time the delay of the pulse through the network, read out the results, and finally reset and initialize the system. If the control unit has the further ability to vary the timing delays, according to some probability function, then the gsc could work in conjunction with a digital computer doing Monte Carlo simulation.

2. The Routing of Long Distance Telephone Calls

In a long distance telephone network there are hundreds of switching points (nodes), each of these nodes being connected to other nodes via trunk lines (arcs). Associated with each trunk line there exists a corresponding cost function, dependent on the present load (dynamic cost) and certain fixed costs such as distance and tariff policies. We are thus invited to solve the problem of what route from a sending terminal (source) to a destination (sink) minimizes the total cost of placing the call.

Since part of the cost function is dynamic in nature, we must include a feedback loop as an integral part of the system. Further, if the computation time for finding the optimal path is excessive, the results will be made worthless by the action of the dynamic model, i.e.,

the "optimal" path will no longer be optimal. Of course, the user of the phone system will have become disgusted with the service long before this dynamic failure becomes serious.

If the electronic time delays of the gsc's arcs can be made to correspond to the cost function of using the associated trunk lines, then the problem is soluble. It should be noted that the topology of the graph is constant enough so that the circuits could be wired together. Further, if every sending terminal has its own gsc for computing its optimal outgoing routes, then the source node can also be fixed. Once a gsc has computed the route, the routing information can then be sent, one step at a time, along the trunk lines already established for a specific call.

If a single time shared bus can be provided, which connects all switching centers and thus all gsc's together, then each trunk line could communicate its load factor to the gsc's several hundred times a second. It should be noted that the system works in a true real-time environment, i.e., if data is received by a gsc while it is in the process of computing a route, the new data will be used if the calculation has not proceeded past the trunk line.

3. "Extended" PERT

Another obvious application of the gsc is in the solution of PERT network type problems. The PERT case offers a significantly different challenge to our design in two aspects: the probabilistic nature of the

weights given to the arcs and the human (manager) interface to the system.

In the traditional PERT model, the expected value of the "critical" path was assumed to be the sum of the expected values of the arcs comprising this path. However, since the weights of the arcs are not fixed values but are probability distributions, there is no unique critical path but rather only a probability that any given path will be critical.

Many authors have pointed out that PERT's simplification of the probabilistic model to the deterministic case always yields an optimistic solution for the expected value of the critical path. [1,2,3] Monte Carlo evaluations of the graph have been proposed to handle this problem; [4] however, the time and cost of producing such runs appears to be excessive, and the use of PERT has recently fallen into disrepute in some quarters. [5]

With a gsc if each of the time delays fluctuated with the random probability distribution desired for that arc, many runs could be made through the graph in a reasonable amount of time with no resetting of the time delay elements. After each run through the graph, the gsc could automatically update expected value counters, provide signals for generating histograms representing the distribution functions over the graph, reset its flip-flops, and recycle the system; this yields an excellent statistical tool for management. For example, the gsc should be fast enough that if, after a critical path is found by a run, a light is pulsed on each arc in this path then the visual intensity of the arc's light will give the manager the desired information on the

criticality of a certain sub-job.

Another advantage of a gsc over the digital computer is that the input and output remain in a form understandable to the manager, and translation mistakes between the directed graph and the PERT coding sheets are eliminated. The manager also has an immediate method of interacting with his problem, changing values of the arcs and adding or deleting arcs at will. This interaction is perhaps the most advantageous use of PERT.[6]

A further aid to management is the fact that the gsc network, once set up, can be used to monitor incoming progress reports, slippages, etc. Thus it can give the most up-to-date estimates possible for the completing of the total job.

Since AND and OR circuits can be mixed in the graph, this "extended" PERT model can take into account the possibility of decision making: the management taking the path through the OR which is most critical. It should be noted, however, that once the decision is made as to which path of an OR possibility is to be taken, the non-selected alternatives should be removed so that the job's total expected value is not low.

4. Proposed Small Scale Model

In order to test the practicality of the proposed graph simulator a small scale model is being built (Figure 11).

The model will perform only the OR functions at the nodes. Time delay adjustment will be done manually, as will the interconnection of the

arcs using phone plugs. The arc circuit will also be an integral part of a single unit that will include the minimal control circuit. The wiring for the signals from the control unit, like reset, selective reset, and read/select will all be permanent.

The model will not include priority reset; the selective reset will be done along the scheme outlined in Figure 4B.

The electronic delay will depend on the time constant of an RC circuit. When the input gate is turned on the capacitor will start charging. After reaching a threshold voltage the capacitor will fire the unijunction transistor. The delay can be varied by changing the resistance of the charging network.

The readout will be done by visual inspection, every set flip-flop having turned on its indicator light. The control unit will provide mainly for manual operation. Pushing the START button will first clear all arc circuits, then start a counter and trigger the source node. As soon as the sink node has been selected the counter will stop and display the elapsed time. The operator may now use the selective reset before reading the set arcs, or reset the graph simulator and start over again.

5. Conclusion

It is our opinion that the gsc offers a new method for analyzing what might be termed "point to point" graph problems, and that the device seems worthy of further investigation. Techniques can be developed to solve such problems as the "next to shortest" or "next

to longest" path through a graph, and the device is capable of answering some connectedness problems of undirected or unweighted graphs. The applications discussed in this paper are hardly the complete set of possible uses, but rather represent two diverse practical usages for such a device. The theoretical simplicity of the model, provided that engineering studies would make it cost-feasible, might well produce a valuable new tool for business and industry to use.

REFERENCES

1. D.G. Malcolm, J.H. Roseboom, C.E. Clark, W. Fazar, Application of a Technique for Research and Development Program Evaluation, Journal of Operations Research, vol. 7, pp. 646-669, (1959).
2. D.R. Fulkerson, Expected Critical Path Lengths in PERT Networks, Journal of Operations Research, vol. 10, pp. 808-817, (1962).
3. K.R. MacCrimmon, C.A. Ryavec, An Analytic Study of PERT Assumptions, Journal of Operations Research, vol. 12, pp. 16-37, (1964).
4. R.M. Van Slyke, Monte Carlo Methods and the PERT Problem, Journal of Operations Research, vol. 11, pp. 839-860 (1963).
5. Kenneth MacCrimmon, private communication, Carnegie Institute of Technology, Pittsburgh, Pa.
6. F.D. Robinson, The Background of the PERT Algorithm, Computer Journal, vol. 5, pp. 197-300 (1962).

FIGURE 1 - GRAPH REPRESENTATION

FIGURE 2-"OR" ARC CIRCUIT

— — — ARC
———— ATTEMPTED PATH
oooooo CRITICAL PATH

FIGURE 3 - TREE OF ATTEMPTED PATHS

FIGURE 4 - ARC CIRCUIT WITH AUTOMATIC BACKTRACE (OR)

ONE SHOT
PULSE AMPLIFIER

FIGURE 5 – "AND" ARC CIRCUIT

A

B

△ SWITCH OPEN-INVERTER IN THE CIRCUIT

■ SWITCH CLOSED-INVERTER NOT IN THE CIRCUIT

WEIGHT OF CRITICAL PATH = $\min (X_1 + X_4, \max (X_1 + X_3, X_2) + X_5)$

FIGURE 6 – MIXED MAX AND MIN NETWORK

FIGURE 7- PRIORITY RESET

1 SELECTIVE RESET
2 RESET
3 PRIORITY RESET

FIGURE 8 — GENERAL ARC CIRCUIT

FIGURE 9 - CROSSBAR SYSTEM

FIGURE 10 – CROSSBAR EXAMPLE

FIGURE 11 - MODEL OF A GSC ARC CIRCUIT

A BENCH MARK CALCULATION FOR RETRIEVAL BY CONFLUENCE
OF BINARY ATTRIBUTES IN A RANDOM-ACCESS COMPUTER

G.K. Manacher

The following problem has never been solved (and probably cannot be solved) efficiently on a random-access computer: Let an item be specified by a set of attributes, which are either present or absent. The set of attributes for any item is a subset of the set of possible attributes. Let a query of order q be similarly defined as a subset of q attributes chosen from the set of possible attributes. Assuming that the items are stored in core memory, the problem is to retrieve efficiently the items having all the attributes specified in the query.

Various schemes for managing specific cases have been devised. A common one consists of storing inverted lists. In this scheme every item having a particular attribute is linked together. A table is kept of the number of items in each list. A query is broken into component attributes; the smallest list belonging to a component attribute is scanned linearly for a match of all attributes.

Various other methods exist. When the number of bits q in the query is nearly equal to the number of bits in each item, schemes based on finite algebras may be employed [1]. When the number of attributes equals the number of query attributes, hash coding schemes may be employed.

The general problem has been attacked successfully only by proposing new hardware schemes, particularly associative memories [2].

The purpose of this note is to analyze the scheme believed by the author to be one of the simplest. While by no means practical, it performs relatively well in relation to other schemes, uses storage quite sparsely, and is amenable to an approximate analysis. This analysis lends itself to an understanding of the dependence of scan time on the various parameters.

Let the "attribute vector" be an ordered set of binary numbers, with "1" standing for "present" and "0" for "absent." Each bit position represents a possible attribute.

Let the attribute vectors be indexed in an arbitrary order: For simplicity, assume that there are 2^u of them, with u some integer. The data structure we shall use to aid in the search is a table of identical size to the one containing the attribute vectors. Adjacent pairs of attribute vectors are "or'd" together, and the resultant vector stored sequentially in a table half the size of the original table. Then adjacent pairs of these are computed and stored, and so on until one obtains the complete binary merge of all the attribute vectors. Thus, in Figure 1, if col. (a) represents the original data, then col. (b) is the data merged once and col. (c), the data merged twice. In Figure 1, vectors 5 and 6 are said to be component vectors below vector 7, and vector 7 is directly above vectors 5 and 6.

Figure 1.

The form of storage for the table of or-merged vectors presents no problem; it may be stored in the fashion of an ordinary array [3]. What is clear is that the amount of storage required for the whole scheme is just twice the amount required for the attribute vectors.

Let M represent the number of possible attributes. Suppose attributes occur in any item with probability p . Then on the average the number of attributes per item will be Mp . Let the number of attributes in a query q , and let the total number of items be N . We shall characterize the average number of lookups sufficient to retrieve all the items satisfying the query as $A(p, q)$. Clearly, $A(p, q) \propto N$, so it makes sense to define $F(p, q) = A(p, q) / N$.

An attribute vector will be called a "vector at the lowest level." The result of merging two of them will be called "a vector at the next lowest level"; in general, the or-merge of two vectors at level m will be a vector at level $m-1$.

The scan is as follows. The highest (0-level) vector is consulted to see if it satisfies the query. If it does, both of its component subvectors are examined, and this process continues recursively, stopping at each

m-level vector failing to match the query, else proceeding downward by attempting to match the two $m+1$ level component vectors "below" the m-level vector. The process of course "finds" a match at the lowest (u-) level, at which the attribute vectors themselves are stored. Note that a failure to match at the highest level implies immediately that the desired attribute vector is not in the set.

Evidently this scheme is impractical. Its value lies in the possibility of analysis, and in the fact that other (hopefully, better!) schemes can use $F(p, q)$ as a benchmark comparison.

At the highest level, let the probability for an attribute be p_0 . Consider only the case of $q \ll N$. The probability for the confluence of q particular attributes is approximately p_0^q . The probability for no query match at level 0 is therefore $1 - p_0^q$.

Consider (either) one of the component vectors "under" the 0-level vector, at level 1. Clearly this vector has the property that whenever it is 1, the vector "above" it is 1, but not vice versa. Suppose the probability for an attribute at level 1 is p_1 . The probability for a bit at level 1 given that this bit is present directly "above" it at level 0 is clearly p_1/p_0 [observe that necessarily $p_1 < p_0$]. Hence the probability for a query matching a single component vector at level 1 given a match "above" at level 0 is $(p_1/p_0)^q$. The probability for a match at level 0 and no match at a particular component vector at level 1 is $p_0^q(1 - (p_1/p_0)^q) = p_0^q - p_1^q$.

Continuing this analysis by pushing forward an arbitrary number of steps, we find that the probability for a match at the $(n-1)^{\text{th}}$ level but no match of an n^{th} level vector directly below is

$$(1) \quad p_0^q (p_1/p_1)^q (p_2/p_1)^q = (p_{n-1}/p_{n-2}) (1 - (p_n/p_{n-1})^q) = p_{n-1}^q - p_n^q.$$

We shall call (1) Π_n^q .

It is easy to compute p_n . Remembering the merging scheme, we conclude that the probability for a bit not being present at the n^{th} level is just $(1-p)^r$, with $r = 2^{(u-n)}$ and $2^u =$ number of attribute vectors in the table.

Hence,

$$(2) \quad p_n = 1 - (1-p)^{2^{(u-n)}}$$

and

$$(3) \quad \Pi_n^q = \left\{ 1 - (1-p)^{2^{(u-n+1)}} \right\}^q - \left\{ 1 - (1-p)^{2^{(u-n)}} \right\}^q.$$

Now consider the entire binary tree in the fashion of the fragment shown in Figure 1. About 2^n nodes are consulted in connection with all scans terminating at the n^{th} level. The probability for a termination at this level is just Π_n^q . Hence the mean number of consultations to extract all the attribute vectors satisfying the query is just

$$(4) \quad \bar{D} = \sum_{n=0}^u 2^n \pi_n^q$$

Replacing (4) with an integral and putting $2^n = e^{n \ln 2}$ [$\ln = \log$ base e]

we obtain

$$(5) \quad \bar{D} \approx \int_0^u e^{n \cdot \ln 2} \left\{ \left[1 - t^{e^{(u+n-1) \ln 2}} \right]^q - \left[1 - t^{e^{(u-n) \ln 2}} \right]^q \right\} dn;$$

where

$$(6) \quad t = 1 - p.$$

By making the substitution $z = 2^{(u-n) \ln t}$ one can reduce (4) to

$$(7) \quad \bar{D} \approx \frac{N \ln t}{\ln 2} \int_{\ln t}^{N \ln t} \left\{ (1 - e^{2z})^q - (1 - e^z)^q \right\} \frac{dz}{z}.$$

Observing that $\ln t$ is negative, that the two terms in the integrand can be combined, and assuming p small, one can make a further approximation, obtaining

$$(8) \quad \bar{D} \approx - \frac{N \ln t}{\ln 2} I(q)$$

where $I(q) = \int_0^\infty (1 - e^{-z})^q \frac{dz}{z}.$

In the classic definite integral table of Bierens de Haan [4], one may find $I(q)$ expressed as a finite sum involving binomial coefficients. For q fairly large (≥ 4) one obtains an excellent approximation (not worse than one part in 100 for $q \geq 4$ and not worse than about 20% for all integral q):

$$(9)^* \quad I(q) \xrightarrow{q \rightarrow \infty} \frac{\pi \ln 2}{2 \ln q}$$

Combining (8) and (9), and observing that for small p , $-\ln t \approx p$,

$$(10) \quad \bar{D} \approx \frac{\pi N p}{2 \ln q}$$

Hence

$$(11) \quad F(p, q) \approx \frac{\pi}{2} \frac{p}{\ln q}$$

It is thus seen that this simple scheme is about as efficient as an inverted list scheme, and even improves (though not very much!) with increasing q .

On the average, the number of items present that satisfy the query should be $N \cdot p^q$. Consequently (11), like every other result relating to this difficult problem, shows that the scan is very inefficient.

*The Maniac III computer was used to obtain a series which aided in obtaining (9).

One more point should be made. Mooers [5] has discussed the situation in which objects are retrieved from a file by e.g. punch-coding cards. One inserts rods into a deck of cards at given punch-code positions. Upon lifting the deck, all cards having open slots at all the punch card positions are left behind, whereas any with a closed hole at at least one position are extracted. In this situation, Mooers has emphasized the economies in space utilization to be had by overlapping the codes, with the result that an occasional wrong item is retrieved. He has pointed out the utility of random functions to encode the attributes. However, retrieval of this type is essentially associative processing, and analyses of it do not add to an understanding of the peculiar problems of random-access memories.

REFERENCES

1. S.P. Ghosh and C.T. Abraham, "Application of Finite Geometry in File Organization for Records with Multiple Valued Attributes," IBM Research Center, Yorktown Heights, N.Y., no serial number, 1964.
2. J.A. Feldman, "Aspects of Associative Processing," M.I.T. Lincoln Labs, Technical note 1965-13, April, 1965.
3. R. W. Floyd, "Treesort 3", Comm. ACM, 7, Dec., 1964, p. 701.
4. Bierens de Haan, "Nouvelles Tables d'Intégrales Définies", P. Engels, Leiden, 1867.
5. C.N. Mooers, "Choice and Coding in Information Retrieval Systems," Trans. IRE Prof. Group on Information Theory PGIT-4, 112-118, 1954.

NUMERICAL STABILITY IN N-BODY INTEGRATIONS

R.H. Miller

(Lectures given at the Seventh Annual Summer Institute in Dynamical Astronomy held at Purdue University June 19-July 14, 1967. The Institute was devoted to the field of Stability in Celestial Mechanics.)

An n-body integration--or any computer calculation--is an experimental approach carrying with it an obligation for the experimenter to demonstrate the validity and applicability of his results. Computer experiments are a class of activity somewhere between theory and observation (or experiment), combining some of the features of each of these more traditional approaches to scientific problems. A well-designed experiment that makes effective use of a computer may be quite different from a simple translation of a desk-calculator operation. It probably will involve completely different ways of representing the system and correspondingly different ways of carrying out the required manipulations. In the case of physical or astronomical problems, this means going back to the underlying physics and determining the fundamental physical process that is to be represented. This fundamental property should be represented exactly, if that is possible, perhaps even at the expense of treating some other features less precisely.

A large n-body calculation was constructed along these lines. The physical process mimicked exactly is the incompressible reversible flow

III A-2

in phase space described by the collision-free Boltzmann equation.

Because a detailed report of this work will be published shortly

(with K.H. Prendergast, in the Astrophysical Journal), this note will

emphasize those features that have not been published elsewhere or

that are not included in the report. The new calculation is based on

a discrete representation of the phase space (technically, the μ -space)

in the computer memory. Each lattice point is represented by a memory

core, and occupancy of that lattice point is represented by the "number"

in the core. Bits are moved around the computer memory to represent

the evolution of the system. In fundamental properties, the new repre-

sentation is equivalent to the old; it gains exact integrals of the approximation

at the expense of dealing with a shorter effective word length.

All stellar dynamical n -body calculations incorporate, at some

stage, an integration according to a finite-difference scheme.

Phenomenologically, such systems do not seem to time-reverse in

microscopic detail, an effect that was reported some time ago (Miller,

1964, 1966a). The experimental techniques developed to explore this

effect appear to be usable for a wide class of initial value problems and

a study of that problem will be published shortly (Miller, 1967).

The finite-difference scheme used in the new n -body integration

can easily be studied by the formal methods usually applied to finite-

difference approximations to initial value problems (Lax and Richtmyer, 1956;

or see, for example, Wendorff, 1966, Sec. 3.1). The equations of the

new n -body integration are

$$u^{(n+\frac{1}{2})} = u^{(n-\frac{1}{2})} + f^{(n)},$$

$$x^{(n+1)} = x^{(n)} + u^{(n+\frac{1}{2})},$$

where $f^{(n)}$ is to be determined from the configuration of $x^{(n)}$. If the x and u belonging to a single stage of the process are comprehended into a "state vector", this system can be written in matrix notation as

$$\xi^{(n+1)} = A \xi^{(n)} + \varphi^{(n)} = \begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix} \xi^{(n)} + \varphi^{(n)},$$

where $\xi^{(n)} = \begin{pmatrix} x^{(n)} \\ u^{(n-\frac{1}{2})} \end{pmatrix}$ is a "vector" with two elements for each degree of freedom and Δt is the value of the time-step. The vector

$\varphi^{(n)}$ is $f^{(n)} \begin{pmatrix} \Delta t^2 \\ \Delta t \end{pmatrix}$ with the sequence of steps as defined here

(with a different definition, $\xi^{(n)} = \begin{pmatrix} x^{(n)} \\ u^{(n+\frac{1}{2})} \end{pmatrix}$, the corresponding

$\varphi^{(n)} = f^{(n)} \begin{pmatrix} 0 \\ \Delta t \end{pmatrix}$). All time-steps are equal, so the result of integration steps is just

$$\xi^{(n+1)} = A^S \xi^{(n-S+1)} + \varphi^{(n)} + A \varphi^{(n-1)} + \dots + A^{S-1} \varphi^{(n-S+1)},$$

with an obvious generalization to unequal time-steps. Anything untoward

will happen in the A^S term. The usual arguments require defining the

norm $\|A\| = \sup \frac{\|A\eta\|}{\|\eta\|}$ for all non-zero vectors, η . In the present

case, $\|A\| = 1 + \frac{\Delta t}{2} + O(\Delta t^2)$. Then, if a total elapsed time, $T = n \Delta t$,

and $n \rightarrow \infty$ while $\Delta t \rightarrow 0$ so that T is constant,

$$\|\xi^{(n)}\| \leq \|A^{(n)}\| \cdot \|\xi^{(0)}\| \leq \lim_{n \rightarrow \infty} \left(1 + \frac{T}{2n}\right)^n \|\xi^{(0)}\| = e^{\frac{T}{2}} \|\xi^{(0)}\|.$$

This establishes a bound on the amount by which the approximation may depart from an exact solution. Lax and Richtmyer (1956) show, under certain conditions which this problem meets, that $\|A\| = 1 + O(\Delta t)$ is a sufficient condition for stability of the approximation.

Similar arguments might be presented for some of the more complicated integration schemes used in other n-body integrations.

The disturbing point about this argument is that the bound placed on the departure of the approximate solution from the exact solution grows exponentially with the elapsed time.

Earlier work on the n-body problem (Miller, 1964) demonstrated that two numerical solutions also diverged exponentially in the elapsed time. It was argued that this is of physical origin, or (equivalently) that it is a property of the operator of the problem being approximated. This operator may be defined as follows: If $x(t)$ be the state vector of the system at time t , defined in the usual way, then a time-evolution operator $E(t)$ can be defined such that $x(t) = E(t)x(0)$, in terms of the initial conditions. The operator $E(t)$ should satisfy certain continuity and boundedness conditions for finite time. In stellar dynamics, the phase space is unbounded. Infinite time is required to go to infinite distance along the configuration directions, but a collision of two point-particles would lead

to infinite momenta that could be reached in a finite time. It is physically reasonable to limit the force at short range, thus providing an operator $E(t)$ that is bounded for finite times. Because the departure of the approximation from the exact solution has an exponential character, the exponential growth of the separation of two similar systems is not sufficient to guarantee that the trouble lies in $E(t)$ rather than in the approximation. The growth rate may be enough to distinguish, but a stronger argument is desirable.

The divergence of similar trajectories under $E(t)$ can be investigated without recourse to the same kind of n -body integrations that disclosed the trouble in the first place (Miller, 1966b), but these investigations are not yet in satisfactory form. The stellar dynamical equilibrium represents a delicate balance of forces such that the balance for a nearby orbit (in the sense of a perturbation) is enough different that the two systems diverge exponentially.

The question whether computer n -body investigations adequately represent physical systems can be honestly faced. Some progress has been made, but definitive answers are not yet available. A properly designed experimental approach may prove adequate, but a re-designed calculation that conserves some of the physical integrals exactly offers a valuable alternative.

The work on the new n -body calculation was carried out in collaboration with Professor Kevin H. Prendergast of the Astronomy Department at Columbia University. Part of the work was done with

the aid of a National Research Council Senior Postdoctoral Resident Research Associateship supported by the National Aeronautics and Space Administration. The hospitality of Dr. Robert Jastrow and of the Goddard Institute for Space Studies is gratefully acknowledged.

REFERENCES

- Lax, P.D. and Richtmyer, R.D., 1956, Comm. Pure and Applied Math. 9, 267-293.
- Miller, R. H. 1964 Astrophys. J. 140, 250-256.
- _____ 1966a I.A.U. Symposium No. 25, "Theory of Orbits in the Solar System and in Stellar Systems." G. Contopoulos, Editor. (New York: Academic Press).
- _____ 1966b Astrophys. J. 146, 831-37.
- _____ 1967 Journal of Computational Physics (to be published).
- Wendorff, B., 1966. Theoretical Numerical Analysis, (New York: Academic Press).