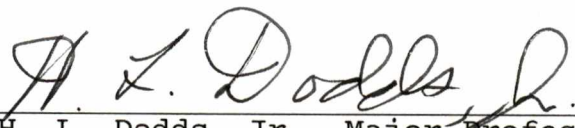
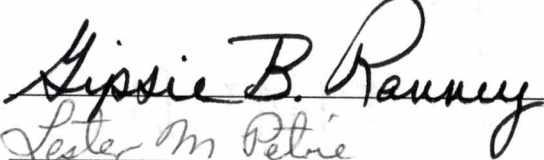
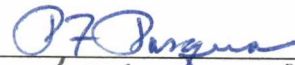


To the Graduate Council:

I am submitting herewith a thesis written by Daniele Ugolini entitled "Monte Carlo Criticality Calculations Using a Parallel Processor Computer." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Nuclear Engineering.

  
H. L. Dodds, Jr., Major Professor

We have read this thesis  
and recommend its acceptance:

  
Lester M. Petrie  
  
L. F. Miller

Accepted for the Council :

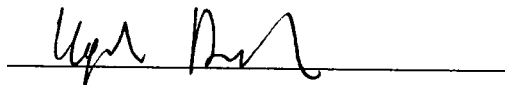
  
Vice Provost  
and Dean of The Graduate School

# STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Master's degree at the University of Tennessee, Knoxville, I agree that the Library shall make it available to borrowers under rules of the Library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgement of the source is made.

Permission for extensive quotation from or reproduction of this thesis may be granted by my major professor, or in his absence, by the Head of the Interlibrary Service when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Signature



Date

July, 1987

MONTE CARLO CRITICALITY CALCULATIONS USING  
A PARALLEL PROCESSOR COMPUTER

A Thesis  
Presented for the  
Master of Science  
Degree  
The University of Tennessee, Knoxville

Daniele Ugolini

August 1987

Ai miei nonni

## ACKNOWLEDGEMENTS

The author wishes to express his gratitude to all who contributed their time and talent for the completion of this thesis, and in particular to:

H. L. Dodds, from UT and O.R.N.L., for his advice, contribution, and recommendation which made this thesis possible;

L. M. Petrie and N. L. Landers, from O.R.N.L., who introduced me to the KENO code and helped me through the development of this thesis;

P. F. Pasqua, head of the Nuclear Engineering Department at UT for making this work possible through a research assistantship;

J. Jones, from O.R.N.L., for his patience and his advice in familiarizing me with the NCUBE computer; and,

My parents, Giancarlo and Carla, my brother Claudio, my friends Carlos March-Leuba and David Auslander, the members of the coffee club, and the graduate fellows of Room 204 of NE for their invaluable help and encouragement.

## ABSTRACT

The purpose of this work is to implement and evaluate the Monte Carlo criticality code KENO V.a on a parallel processing computer, specifically the NCUBE with 64 processors. Because the particle histories are independent of each other, it should be possible to track groups of particles simultaneously via the parallel computer and thus effect a savings in time relative to following the particles sequentially on a large mainframe computer. Several different versions of KENO V.a have been developed for the parallel computer starting with only the tracking subroutine and then progressively adding more subroutines for the parallel portion of the calculation. The results obtained indicate that the NCUBE is a fast computer only if nearly everything is put in parallel with little or no sequential calculations being performed. The banking of the histories into generations, an essential part of the Monte Carlo methodology employed in KENO V.a, must be done sequentially and, hence, slows the calculational process considerably. The problem caused by the banking process can be circumvented to some extent by using smaller banks of histories but this causes the statistical uncertainties to increase.

In summary, the 64 processor NCUBE is not ideally suited for conventional Monte Carlo criticality calculations. The host portion of the NCUBE computer,

which performs the sequential part of the calculation, needs to be much faster. Alternatively, an interface between the NCUBE and a mainframe computer could be created, which is essentially a hybrid computer, to efficiently handle the parallel and sequential parts of the calculation, respectively.

## TABLE OF CONTENTS

| CHAPTER                                           | PAGE |
|---------------------------------------------------|------|
| INTRODUCTION . . . . .                            | 1    |
| I. SIMPLIFIED VERSION OF KENO V.a . . . . .       | 5    |
| Most Significant Subroutines of KENO V.a . .      | 5    |
| CES - A Simplified Version of KENO V.a . . .      | 7    |
| Results of KENO V.a and CES on the                |      |
| I.B.M. Computer . . . . .                         | 11   |
| II. DESCRIPTION OF NCUBE . . . . .                | 14   |
| General Description of Parallel                   |      |
| Processor Computers . . . . .                     | 14   |
| Description of the NCUBE . . . . .                | 15   |
| NCUBE Architecture and Communication System .     | 15   |
| Implementation of a Fortran Program               |      |
| on the NCUBE . . . . .                            | 17   |
| The Speed-Up Factor . . . . .                     | 18   |
| III. IMPLEMENTATION OF CES ON THE NCUBE . . . . . | 20   |
| Use of the Random Number Generator                |      |
| in Parallel . . . . .                             | 20   |
| Parallelization of CES . . . . .                  | 21   |
| Description of ENA.a . . . . .                    | 23   |
| Results of CES with ENA.a . . . . .               | 24   |
| Description of ENA.b . . . . .                    | 28   |
| Results of CES with ENA.b . . . . .               | 29   |
| Description of ENA.c . . . . .                    | 32   |
| Results of CES with ENA.c . . . . .               | 34   |

| CHAPTER                                   | PAGE |
|-------------------------------------------|------|
| III. (Continued)                          |      |
| A Modified Version of ENA.c . . . . .     | 37   |
| Results of CES with ENA.c.mod . . . . .   | 38   |
| IV. CONCLUSIONS AND FUTURE WORK . . . . . | 40   |
| Conclusions . . . . .                     | 40   |
| Comments About the Current Version        |      |
| of the NCUBE . . . . .                    | 41   |
| Future Work . . . . .                     | 42   |
| BIBLIOGRAPHY . . . . .                    | 44   |
| APPENDIX . . . . .                        | 46   |
| VITA . . . . .                            | 53   |

# LIST OF TABLES

| TABLE                                                                                                                                | PAGE |
|--------------------------------------------------------------------------------------------------------------------------------------|------|
| 1. SAMPLE PROBLEM DATA . . . . .                                                                                                     | 11   |
| 2. COMPARISON OF THE AVERAGE K-EFFECTIVE AND<br>THE STANDARD DEVIATION OBTAINED WITH<br>CES AND KENO V.a ON THE I.B.M. MAINFRAME . . | 12   |
| 3. COMPARISON OF THE RESULTS OF CES USING<br>DIFFERENT RANDOM NUMBER GENERATORS<br>ON THE I.B.M. MAINFRAME . . . . .                 | 13   |
| 4. AVERAGE K-EFFECTIVE AND STANDARD DEVIATION<br>OF CES WITH ENA.a USING<br>1, 2, 4, 8, 16 NODES . . . . .                           | 25   |
| 5. COMPUTATIONAL TIME AND SPEED-UP FACTOR<br>OF CES WITH ENA.a USING<br>1, 2, 4, 8, 16 NODES . . . . .                               | 25   |
| 6. AVERAGE K-EFFECTIVE AND STANDARD DEVIATION<br>OF CES WITH ENA.b USING<br>1, 2, 4, 8, 16 NODES . . . . .                           | 29   |
| 7. COMPUTATIONAL TIME AND SPEED-UP FACTOR<br>OF CES WITH ENA.b USING<br>1, 2, 4, 8, 16 NODES . . . . .                               | 30   |
| 8. AVERAGE K-EFFECTIVE AND STANDARD DEVIATION<br>OF CES WITH ENA.c USING<br>1, 2, 4, 8, 16 NODES . . . . .                           | 34   |

| TABLE                                                                                                    | PAGE |
|----------------------------------------------------------------------------------------------------------|------|
| 9. COMPUTATIONAL TIME AND SPEED-UP FACTOR<br>OF CES WITH ENA.c USING<br>1, 2, 4, 8, 16 NODES . . . . .   | 35   |
| 10. AVERAGE K-EFFECTIVE AND STANDARD DEVIATION<br>OF CES WITH ENA.c.mod USING<br>1, 2, 4 NODES . . . . . | 39   |
| 11. COMPUTATIONAL TIME AND SPEED-UP FACTOR<br>OF CES WITH ENA.c.mod USING<br>1, 2, 4 NODES . . . . .     | 39   |
| A.1. KENO V.a (16 GROUP) SAMPLE PROBLEM OUTPUT<br>( K-EFFECTIVE AND STANDARD DEVIATION ) . . .           | 47   |
| A.2. CONFIDENCE INTERVALS ( 67, 95, 99 PER CENT )<br>OF KENO V.a (16 GROUP) . . . . .                    | 50   |

## LIST OF FIGURES

| FIGURE |                                                                                                                                                                                                     | PAGE |
|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|
| 1.     | Flow Chart of the Most Significant<br>Subroutines of KENO V.a . . . . .                                                                                                                             | 6    |
| 2.     | NCUBE Architecture and Communication Network<br>with Two, Four, Eight, and Sixteen Nodes . . .                                                                                                      | 16   |
| 3.     | Diagram of the Principal Subroutines Executed<br>in the Host Program and in the<br>Nodes Program . . . . .                                                                                          | 22   |
| 4.     | Measured Speed-Up Factor (indicated by<br>the symbol X) of CES with ENA.a<br>Using 1, 2, 4, 8, 16 Nodes.<br>Lines (a) and (b) Are the Theoretical<br>and the Real Speed-Up Factor Respectively. . . | 27   |
| 5.     | Measured Speed-Up Factor (indicated by<br>the symbol X) of CES with ENA.b<br>Using 1, 2, 4, 8, 16 Nodes.<br>Lines (a) and (b) Are the Theoretical<br>and the Real Speed-Up Factor Respectively. . . | 31   |
| 6.     | Measured Speed-Up Factor (indicated by<br>the symbol X) of CES with ENA.c<br>Using 1, 2, 4, 8, 16 Nodes.<br>Lines (a) and (b) Are the Theoretical<br>and the Real Speed-Up Factor Respectively. . . | 36   |

## INTRODUCTION

Research in computer technology has always been conducted primarily toward the development of larger and faster mainframe computers. These type of computers are able to handle large codes but, on the other hand, their CPU time (i.e. time spent for the computation) is prohibitively expensive. It is important to note that the initial cost of these computers has confined them to national research laboratories, large corporations, and computing centers of the nation's Universities.

In recent years a new type of computer, named parallel processor computer or multi-processor, has been developed. The fundamental concept in the development of this kind of computer is to coordinate several processors to execute different parts of a program simultaneously. Although a complete knowledge of the capability of parallel processor computers is still yet to be ascertained, they are expected to be many times faster than a traditional sequential computer. What has been ascertained is that a parallel processor computer, comprised of small computers, will be much less expensive in CPU time and computing costs, thus making it affordable to a larger audience such as small companies or local research laboratories.

The Engineering Physics & Mathematics Division of the Oak Ridge National Laboratory (O.R.N.L.) has recently

purchased a parallel processor computer called the NCUBE. The NCUBE consists of 64 (expandable to 1024) individual small computers, called nodes, connected together through point-to-point communication channels. The NCUBE is being evaluated by O.R.N.L. for many different types of computing applications. One of the applications currently under consideration is in the area of nuclear criticality safety, specifically, performing k-effective calculations with the Monte Carlo code KENO V.a<sup>1</sup> which is the subject of this thesis. To the best of our knowledge, this is the first time that the performance of Monte Carlo criticality calculations on a parallel processor computer has ever been evaluated.

- In particular, the goals of this research are to
1. develop a simplified version of the KENO V.a code,
  2. implement the simplified version of KENO V.a on the I.B.M. mainframe computer (i.e. the computer for which the code was written) and compare the results obtained with those obtained from the original code, and
  3. implement the simplified version of the code on the NCUBE using a combination of different nodes and compare the results obtained and the computation time required with those obtained on the I.B.M. mainframe.

The variables of importance in this study for evaluating KENO V.a on the NCUBE relative to the I.B.M.

mainframe are accuracy of the results and computational time on each machine. Other variables which could be important to a user, but are not considered in this work, are (1) purchase price of each computer, (2) cost per processor (i.e., node) used on the NCUBE per unit time versus cost per CPU second on the mainframe, and (3) desired turnaround time versus actual turnaround time on each machine.

KENO V.a was developed by the Computing & Telecommunications Division at O.R.N.L.. The code offers a wide array of features but its primary purpose is to compute the k-effective of three dimensional fissile systems. The quantity k-effective can be defined as the ratio of the number of neutrons in the current generation to the number of neutrons in the previous generation. KENO V.a is usually very costly to use and, due to its enormous size, is usually run only on large computers.

A simplified version of the code, called CES, has been developed because the NCUBE could not compile the KENO V.a code. CES contains the heart of the code; namely, those routines which keep track of the particles' trajectories within the system. This is intrinsically the weakest part of every Monte Carlo code because it is here that most of the computation time is spent due to the usually large number of particles being tracked.

The parallelization of CES in the NCUBE has been made possible by the characteristic of the Monte Carlo method which dictates that it is inconsequential as to which particle is being tracked within the batch due to the mutual independence of all particles comprising that batch. In other words it should be possible to track different batches of particles in each node of the NCUBE at the same time.

CES is executed in parallel on the NCUBE in three different ways, each one involving a different part of the program. As will be shown, the results obtained are well below what is expected due to the characteristics of a criticality computation and also due to the actual limits of the NCUBE computer. Nevertheless, it is believed that much can be done to improve the NCUBE and that its application to other Monte Carlo calculations, such as shielding problems, should yield good results.

## CHAPTER I

### SIMPLIFIED VERSION OF KENO V.a

KENO V.a is a large Monte Carlo criticality code. Its size is 1,500,000 bytes and it was originally written for an I.B.M. computer. The code can run as part of the SCALE modular code system or independently on its own. A complete flow-chart together with a user's manual containing all the subroutines of KENO V.a is available through R.S.I.C. (Radiation Shielding Information Center). A short flow chart with the most significant subroutines of the code is shown in Figure 1. These are the subroutines that will be used and sometimes modified in the simplified version of the program.

#### Most Significant Subroutines of KENO V.a

This group of subroutines represents the essence of the code. The following descriptions of the subroutines serve to clarify their role inside the code.

MAIN: This is the starting routine. It is here that all the flags are set, the input data for the problem are read, and the cross sections of the various materials are introduced.

MASTER: This subroutine controls the execution of the program. All the subroutines required by the program are called at this point.

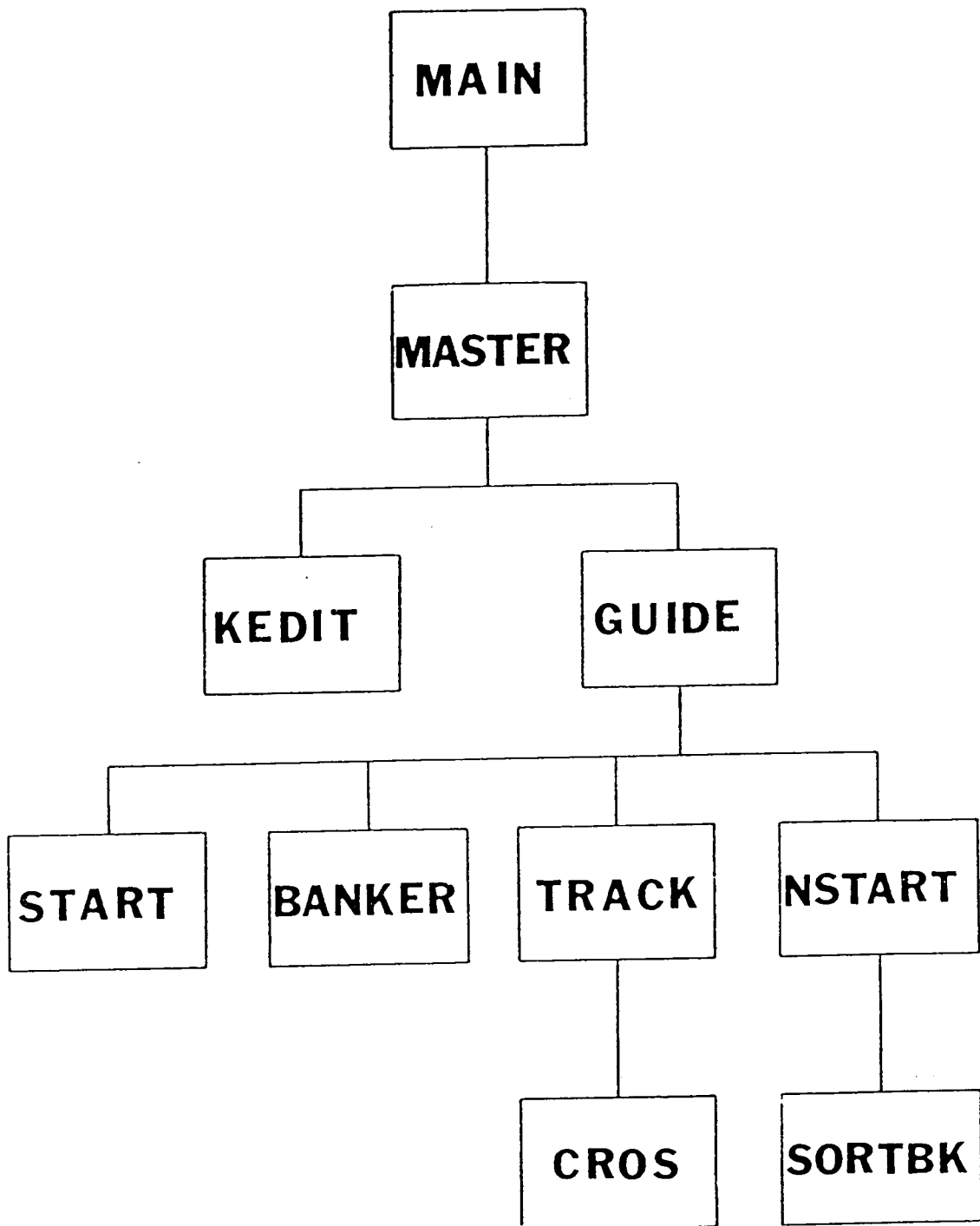


Figure 1.

Flow Chart of the Most Significant Subroutine  
of KENO V.a

KEDIT: This subroutine prints the final results of the program in a predefined format (average k-effective, standard deviation, flux....).

START: This subroutine offers several different choices in position, energy, and direction of the initial source distribution.

GUIDE: This subroutine represents the heart of the code where the loop through the generations is controlled and, during each generation, the following subroutines are called.

BANKER: This subroutine sorts the particles of the current generation at the top of the neutron bank according to their energy group (from the highest energy group to the lowest).

TRACK: This subroutine tracks each particle inside the system until it leaks out or is absorbed. The particles produced by fission are stored into the fission bank.

NSTART: This subroutine moves the information collected inside the fission bank into the neutron bank in order to provide the next source distribution.

#### CES - A Simplified Version of KENO V.a

In order to understand the magnitude of the simplification made to the KENO V.a code, it is adequate to compare the dimension in bytes of the two versions of

the code. As was previously stated, the size of KENO V.a is 1,500,000 bytes, whereas the size of CES, the simplified version, is 35,000 bytes.

The most significant routines contained in CES pertain to the tracking of particles. This is the part of the code that is executed in parallel in a subsequent step.

CES starts with subroutine GUIDE. This subroutine functions as the main program and calls all the other subroutines into execution as they are required which are (in their order of calling) BANKER, TRACK, and NSTART. Subroutine KEDIT, which takes care of the output of the data in the original code, is included in GUIDE.

The approximations and reductions of the subroutines used in CES have been made in two ways:

- a) simplification of the input data and the options available in energy, geometry, and scattering angles,
- b) reduction and adjustment of the utilized subroutines in order to implement the input data of the sample problem.

The input data are implemented in KENO V.a using subroutine MAIN. A new subroutine, INPUT, has been written for CES containing all the values required by the problem including those that are usually taken by default in KENO V.a.

The sample problem used to evaluate CES is a sphere of U-235. Therefore all the other shape options available in KENO V.a (cylinder, cuboid, hemisphere,...) and the related combinations (arrays, array of arrays, holes) were deleted in CES leaving only the sphere option. Also, all the options available for different mixtures and regions were removed from the code.

Several cross section libraries can be used in KENO V.a. The one chosen for this problem is the "16 Group Hansen Roach" library. This library is extremely large and it is unreasonable to copy the whole library into the NCUBE computer. For this reason, an effective "1 Group" cross section for U-235 was created. The data related to the total cross section, non absorption probability, and fission probability were included in the INPUT subroutine. In this manner all the variables in the program are independent of the energy.

Two different options are available for the scattering angle distribution: isotropic (utilizing subroutines GTISO and AZIRN) and non-isotropic (utilizing a Pn approximation of the Legendre Polynomial). The first one was adopted in CES.

The output of CES included the k-effective for each generation, the average k-effective for all the generations, and the standard deviation for each generation. The output also included the 67%, 95%, and,

99% confidence intervals. The objective here was to see how the confidence interval decreased in relation to the increase in the number of histories being processed. For this reason the subroutine KEDIT was included in GUIDE.

As a result of these simplifications, the subroutines utilized by CES were considerably modified. Subroutine START was included as part of GUIDE because a random source distribution with initial isotropic direction in the center of the sphere was chosen. The principal simplifications in subroutine TRACK were related to the splitting and the inward crossing options. Neither was allowed because there was no importance region and also because there was just one region without a void. The most important data used in the problem are shown in Table 1. The neutron and fission banks contain information on the neutrons. Specifically, the weight, the position and, the direction cosines of each neutron are stored in the neutron bank, while, in the fission bank, only the position of each neutron is stored.

TABLE 1  
SAMPLE PROBLEM DATA

---

|                             |                     |
|-----------------------------|---------------------|
| RADIUS OF SPHERE            | 9 Cm.               |
| DENSITY OF U-235            | 0.05 Atoms/Barn-Cm. |
| NUMBER OF ENERGY GROUPS     | 1                   |
| NUMBER OF SCATTERING ANGLES | 0                   |
| NUMBER OF GENERATIONS       | 103                 |
| NEUTRONS PER GENERATION     | 320                 |
| NEUTRON BANK ARRAY SIZE     | 7,384               |
| FISSION BANK ARRAY SIZE     | 3,320               |

---

Results of KENO V.a and CES on the I.B.M. Computer

The results obtained using the original KENO V.a code and the CES code were compared, using one energy group, to see if there was good agreement among these different versions of the program. The data presented in Table 2 shows the average k-effective and the standard deviation after 103 generations of 320 neutrons per generation. The results of a sixteen group KENO V.a calculation are also presented in Table 2. A complete output for the sixteen group calculation is displayed in the Appendix.

TABLE 2

COMPARISON OF THE AVERAGE K-EFFECTIVE AND THE  
STANDARD DEVIATION OBTAINED WITH CES AND  
KENO V.a ON THE I.B.M. MAINFRAME

| Type of Program | Average<br>K-Effective | Standard<br>Deviation | CPU Time<br>( Sec. ) |
|-----------------|------------------------|-----------------------|----------------------|
| KENO V.a (16 G) | 1.08965                | 5.0 -03               | 17.57                |
| KENO V.a (1 G)  | 1.06691                | 4.3 -03               | 15.63                |
| CES             | 1.06811                | 5.8 -03               | 14.42                |

As shown above, good agreement among the different versions of the program is obtained. The little underestimation of the average k-effective found in the two versions using one energy group is due to the fact that the fission probability is more uniformly distributed in the sixteen energy than it is in the one energy group. In addition, the one energy group cross sections were developed in order to get a response that is the closest possible to the one obtained using the sixteen group energy cross sections of U-235.

A further modification had to be made to the code in the random number generator routine. In KENO V.a, the random numbers are selected using FLTRN, a subroutine available on the I.B.M. computer. This subroutine was written for a 64 bit computer. NCUBE is a 32 bit computer

and, of course, could not execute FLTRN. For this reason, a new random generator called RANGEN was used in CES.

The results of CES using the two different random generators are shown in Table 3.

TABLE 3  
COMPARISON OF THE RESULTS OF CES USING DIFFERENT  
RANDOM NUMBER GENERATORS ON THE I.B.M. MAINFRAME

| Type of program | Average<br>K-Effective | Standard<br>Deviation | CPU Time<br>( Sec. ) |
|-----------------|------------------------|-----------------------|----------------------|
| CES (FLTRN)     | 1.06811                | 5.8 -03               | 14.42                |
| CES (RANGEN)    | 1.06396                | 5.7 -03               | 21.18                |

It is clear that the use of two different random number generators with different order of accuracy (i.e. number of decimal places carried) does not affect the quality of the results for either the average k-effective or the standard deviation values. All the results shown from now to the end of this work were obtained implementing RANGEN as the random number generator.

## CHAPTER II

### DESCRIPTION OF NCUBE

#### General Description of Parallel Processor Computers

Parallel processor computers are usually known as multi-processors. The primary advantage of these types of computers is that a program or part of a program can be executed in more than one processor simultaneously. According to the type of communication used, parallel processor computers can be grouped in one of two categories, either:

- 1) shared-memory computers,
- or
- 2) message-passing computers.

Most of the multi-processor computers available in the market are shared-memory computers. The communication between the individual processors and the storage memory is obtained through a switching network. The switching network also provides the connection between the processors.

The NCUBE is a message-passing computer. The connection between the individual processors is provided by a network of communication channels. Each processor has its own memory. The architecture of the network of communication channels can have various shapes such as a

tree, cube, or integral. The NCUBE uses, as the acronym itself suggests, a cube architecture.

The hardware structure of a message-passing machine such as the NCUBE differs from a shared-storage machine in that it does not employ a switching network between the processors and the storage memory.

### Description of the NCUBE

The NCUBE parallel processor computer is defined in the User's Manual as a "super-computer in a mini-computer form." The system consists of two different and separate parts : the host board and the nodes. The host board runs the sequential part of the program and the operating system and also provides the output.

The nodes implement the parallel portion of the program. Each node has its own independent memory the size of which is approximately one hundred kilobytes. The NCUBE computer available at O.R.N.L. has 64 nodes which is expandable up to 1024.

### NCUBE Architecture and Communication System

The architecture of the node system ranging from one to sixteen nodes is shown in Figure 2.  $N$  is the order of the NCUBE and defines the number of nodes implemented in each particular configuration. An NCUBE of order  $N$  is made with the combination of two  $(N - 1)$  order NCUBES.

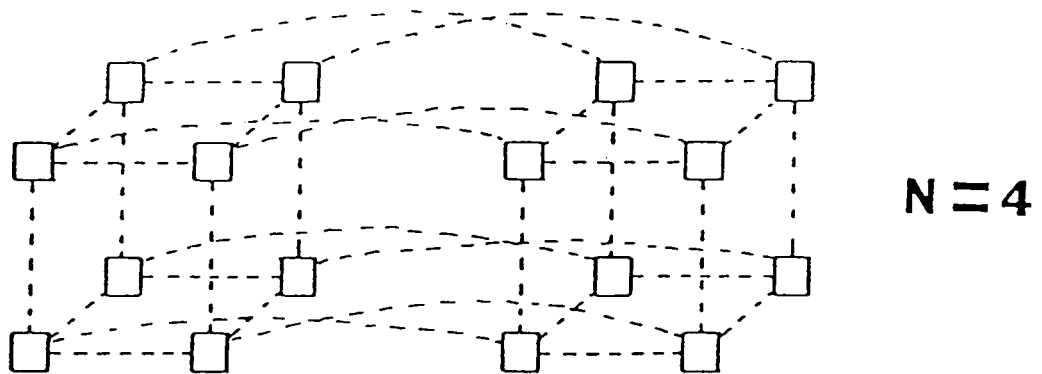
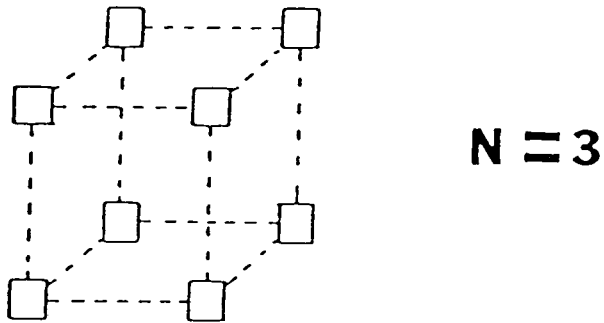
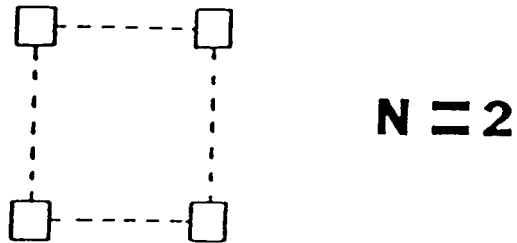


Figure 2.

NCUBE Architecture and Communication Network with  
Two, Four, Eight, and Sixteen Nodes.

The connection between the nodes of two NCUBE's of the same order is accomplished by connecting the nodes that are in the same geometric position relative to each other in each NCUBE. Consequently, only the nodes that are adjacent are connected directly. All the communications between the host board and the nodes must pass through one node which is the node numbered zero. This is the only node that is directly connected with the host. It will be shown later that one of the problems encountered using NCUBE is the lack of a direct connection between each node and the host.

#### Implementation of a Fortran Program on the NCUBE

KENO V.a was written in Fortran 77 and the same language has been adopted in CES since a Fortran 77 compiler is also available on the NCUBE.

In order to write a program that can be performed in parallel on the NCUBE, two different programs must be written: one for the host board and the other for the nodes. These two programs are completely independent of each other and exchange information using the message-passing feature.

Practically all the data that are contained in a common statement or in the arguments of subroutines in a traditional sequential program are transmitted using

message-passing statements from the nodes program to the host program and vice-versa on the NCUBE.

Each message-passing statement contains the types of variables that are transmitted within the message (matrix, array, common statement), the length of the message (in bytes), the destination of the message (node number or host), and the type of the message (a seal of acknowledgement). It follows intuitively that for each statement that sends a message from the host to the nodes, or from a node to another node, or to the host, there is a statement that allows the receiver of the message to read it.

#### The Speed-Up Factor

The speed-up factor,  $S$ , determines whether or not it is advantageous to use a parallel processor computer. It is defined as

$$S = (\text{time on one node} / \text{time on } n \text{ nodes}).$$

This of course is a theoretical value that tacitly assumes the possibility of performing the entire program in parallel. That is not always the case and, usually, almost all programs have some sections that cannot be performed in parallel. Other factors that can limit the speed-up factor are: the inactive time related to an imperfect balance of the nodes, the waiting time caused by

communication latencies in the nodes and in the message sending, and the processor time used to process, send, or receive the messages. In an actual problem, all the possible latencies explained above must be taken into consideration. Consequently, it is usually better to introduce an efficiency coefficient  $e$  and to define the speed-up factor as,

$$S = e * ( \text{time in one node} / \text{time in } n \text{ nodes} ).^2$$

A good value for the efficiency coefficient is 0.5.<sup>3</sup>

A real speed-up factor can be defined as,

$$S (\text{real}) = 0.5 * S (\text{theoretical}).$$

## CHAPTER III

### IMPLEMENTATION OF CES ON THE NCUBE

#### Use of the Random Number Generator in Parallel

Before embarking on the discourse pertaining to the creation of a parallel form of CES, it is important to explain in what manner the random number generator is used in the NCUBE. In a traditional sequential program a new random number is provided by the random number generator subroutine each time the subroutine is called. The randomness of the generated numbers is guaranteed by the structure of the subroutine. When the program is executed in parallel, a modification must be made in order to avoid the use of the same random number in different nodes. This is achieved in CES by adding a new variable in both the fission and in the neutron bank. This variable is the starting random number or seed for each particle. The initial distribution is obtained from the random number generator by selecting a seed number in node zero and then taking the two hundred fiftieth number of the sequence to be implemented in the second node as its seed and to repeat this procedure as required for the other nodes. The purpose of this procedure is to insure independence between node random number sequences (i.e. to avoid the duplication of a sequence of random numbers between two different nodes). In this way, each fission particle

produced after the first generation will have assigned to it a different random number.

### Parallelization of CES

Figure 3 shows the diagram of the subroutine used in CES. This diagram represents the flow charts of the two programs derived from CES that run on the host and on the nodes respectively. The subroutines illustrated under the nodes section represent the generation loop. Inside this loop the complete analysis of the history of each particle is performed.

The parallelization of CES on the NCUBE was developed in three steps, each one involving a section of the generation loop. Three different programs were written for the nodes called ENA.a, ENA.b, and ENA.c. In the first step, only subroutine TRACK was performed in ENA.a; in the second step, subroutine NSTART was added in ENA.b; and, in the third step, subroutine BANKER was added in ENA.c. It is easy to see that, in the final step, all the subroutines responsible for the generation loop were performed in parallel.

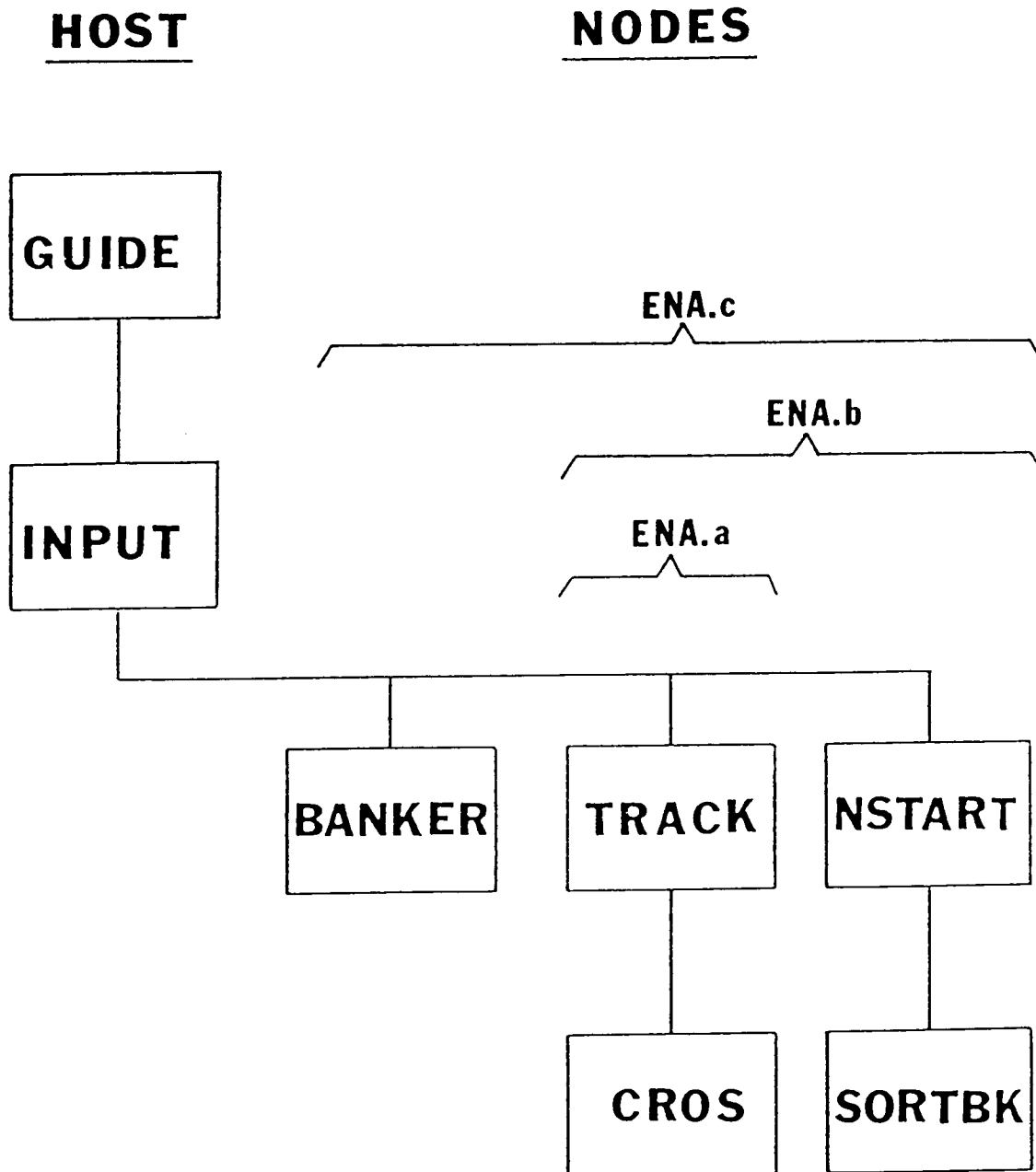


Figure 3.

Diagram of the Principal Subroutines Executed on the Host Program and on the Nodes Program

## Description of ENA.a

ENA.a uses subroutine TRACK as the main program. The procedure of parallelization is implemented by subgrouping the particles of each generation into batches. Each batch is performed in a node. In this way  $(n / \text{number of nodes})$  particles for each generation are run in each node.

An improvement in computation time is expected as the number of nodes utilized is increased. If  $t$  is the time required to run an individual particle,  $(n * t)$  would be the time required to run an entire generation in a traditional sequential program. Whereas the computation time required to run an entire generation would be  $[(n * t) / \text{number of nodes}]$  if a parallel processor computer is used. The gain in computation time is directly proportional to the number of nodes used.

The principal information contained in the messages exchanged between the nodes and the host board (i.e. the parallel portion and the sequential portion of the program) are the fission bank and the neutron bank. In each node, subroutine TRACK selects the particles from the neutron bank and collects the particles produced by fission in the fission bank. When all the particles are computed, each node sends a message containing the fission bank to the host board. The host board (i.e. program CES) reads the messages and rebuilds the fission bank.

In order to avoid the generation of blank columns inside the fission bank, the number of fission particles produced is included in the message. After the fission bank is reconstructed, subroutine NSTART moves the particles from the fission bank to the neutron bank and, subsequently, subroutine BANKER orders the particles according to their energy group. Now CES is ready to send a message back to the nodes containing the neutron bank. The nodes read the message and start tracking the particles of the new generation. This procedure is repeated for 103 generations.

#### Results of CES with ENA.a

CES with ENA.a was executed on the NCUBE using one, two, four, eight and sixteen nodes. After sixteen nodes the computation time was so high that it was considered self-defeating. The results are shown in Tables 4 and 5. Table 4 contains the k-effective and the corresponding standard deviation, and the Table 5 contains the computation time of the nodes, the host, and also the speed-up factor.

TABLE 4

AVERAGE K-EFFECTIVE AND STANDARD DEVIATION OF CES  
WITH ENA.a USING 1, 2, 4, 8, 16 NODES

| Number of<br>Nodes | Average<br>K-Effective | Standard<br>Deviation |
|--------------------|------------------------|-----------------------|
| 1                  | 1.07189                | 5.0 -03               |
| 2                  | 1.06443                | 6.2 -03               |
| 4                  | 1.07121                | 6.4 -03               |
| 8                  | 1.05625                | 6.6 -03               |
| 16                 | 1.04728                | 7.3 -03               |

TABLE 5

COMPUTATIONAL TIME AND SPEED-UP FACTOR OF CES  
WITH ENA.a USING 1, 2, 4, 8, 16 NODES

| Number of<br>Nodes | Host Time<br>( Sec. ) | Node Time<br>( Sec. ) | Speed-up<br>Factor |
|--------------------|-----------------------|-----------------------|--------------------|
| 1                  | 181                   | 393                   | 1                  |
| 2                  | 183                   | 285                   | 1.38               |
| 4                  | 189                   | 262                   | 1.5                |
| 8                  | 204                   | 299                   | 1.31               |
| 16                 | 238                   | 429                   | 0.92               |

As shown in Table 4, the values of k-effective computed with the NCUBE are in reasonably good agreement with the value determined by the standard version of KENO V.a (i.e. the values agree within one or two standard deviations). Also, the magnitudes of the computed standard deviation are at an acceptable level. However, the corresponding computational times are somewhat disappointing. Specifically, the hypothesized reduction in the computation time was not realized. The reason is that the delay introduced in the computation by the message passing procedure after each generation defeated the reduction in computation time. It was confirmed that the computational time in each node was shorter but this benefit was obscured by the waiting time of the nodes after each generation. This effect is clearly shown in TABLE 5 where the Node Time starts increasing when more than eight nodes are used. It is easily seen from the data that the host was quite slow and a lot of time was required in the computation of the subroutines NSTART and BANKER.

The speed-up factor versus the number of nodes used is plotted in Figure 4.

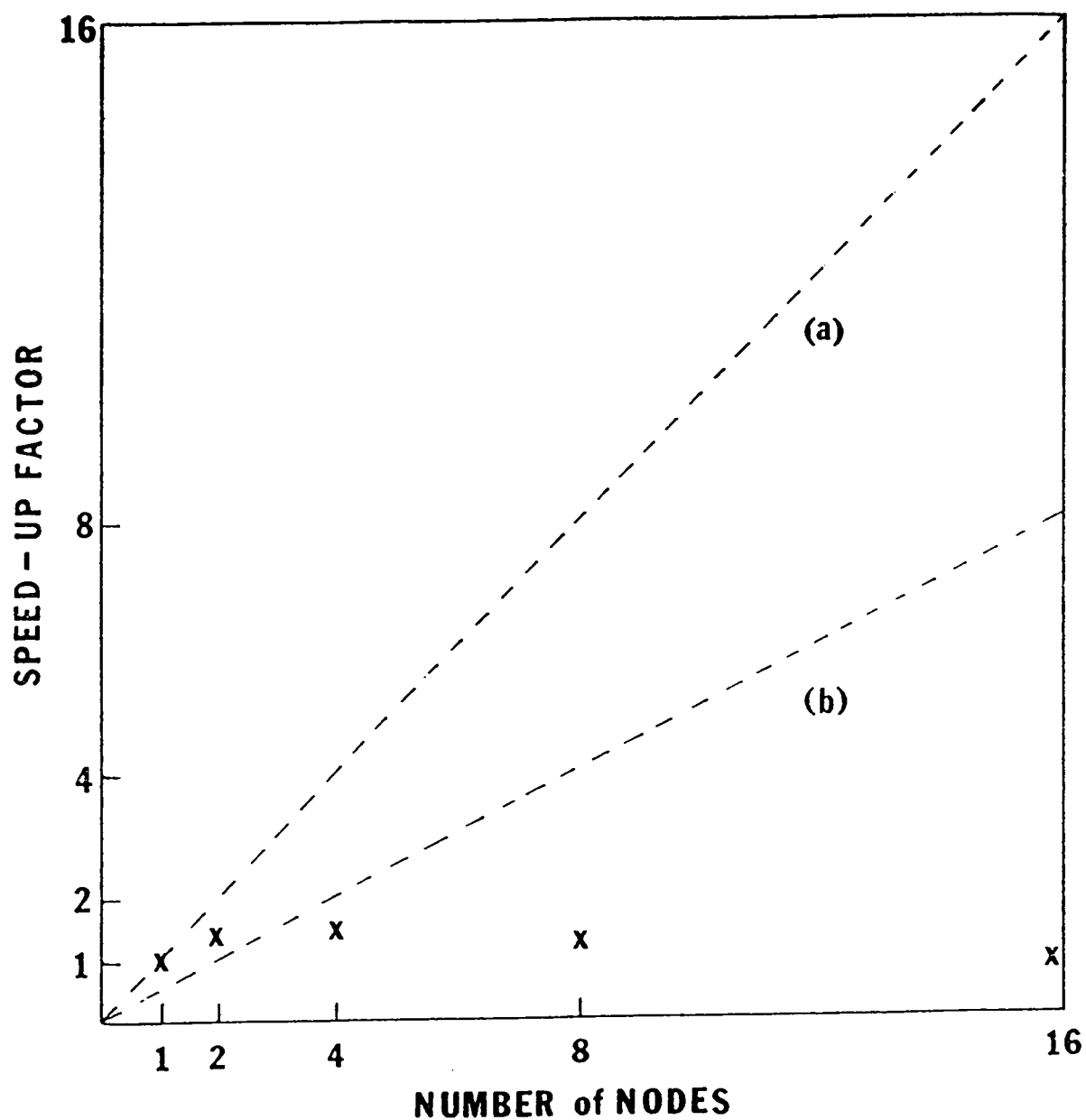


Figure 4.

Measured Speed-Up Factor (indicated by the symbol X)  
of CES with ENA.a Using 1, 2, 4, 8, 16 Nodes.  
Lines (a) and (b) Are the Theoretical and  
the Real Speed-Up Factor Respectively.

## Description of ENA.b

In an attempt to avoid the computational delay encountered in ENA.a, a new program ENA.b was developed for the nodes. The introduction of the subroutine NSTART in the parallel procedure was the main difference between ENA.a and ENA.b. Nothing was changed in the tracking procedure. When all the particles contained in the neutron bank were tracked and the fission particles in the fission bank collected, instead of sending a message containing the fission bank to the host, subroutine NSTART was performed in each node. Then a message containing the neutron bank was sent to the host. The neutron bank was reconstructed in the host and subroutine BANKER performed in order to provide the neutron bank for the next generation. The most important information contained in the messages was in the neutron bank where it was exchanged back and forth 103 times between the nodes and the host at the beginning and at the end of each generation.

The main purpose of ENA.b was to have a configuration of the whole program that was closer to the ideal one (i.e. all of the program performed in parallel). As a consequence, an improvement in the speed up factor together with a reduction of computation time in the host as well as in the nodes was expected.

## Results of CES with ENA.b

CES and ENA.b were performed using one, two, four, eight, and sixteen nodes. As in the previous case, the use of more than sixteen nodes did not bring any improvement in the reduction of computational time.

The values of the k-effective and of the standard deviation after 103 generations are shown in Table 6. The computational time of the nodes and of the host, and the relative speed-up factor after 103 generations are shown in Table 7.

TABLE 6

AVERAGE K-EFFECTIVE AND STANDARD DEVIATION OF CES  
WITH ENA.b USING 1, 2, 4, 8, 16 NODES

| Number of<br>Nodes | Average<br>K-Effective | Standard<br>Deviation |
|--------------------|------------------------|-----------------------|
| 1                  | 1.06754                | 6.2 -03               |
| 2                  | 1.07158                | 5.9 -03               |
| 4                  | 1.04879                | 6.0 -03               |
| 8                  | 1.05843                | 6.7 -03               |
| 16                 | 1.05842                | 7.6 -03               |

TABLE 7  
COMPUTATIONAL TIME AND SPEED-UP FACTOR OF CES WITH  
ENA.b USING 1, 2, 4, 8, 16 NODES

| Number of<br>Nodes | Host Time<br>( Sec. ) | Node Time<br>( Sec. ) | Speed-up<br>Factor |
|--------------------|-----------------------|-----------------------|--------------------|
| 1                  | 56                    | 310                   | 1                  |
| 2                  | 59                    | 189                   | 1.64               |
| 4                  | 65                    | 141                   | 2.2                |
| 8                  | 76                    | 150                   | 2.07               |
| 16                 | 103                   | 277                   | 1.12               |

Again, good agreement between the results obtained using a different number of nodes was achieved. Also, a significant improvement was obtained in the computation time related to the host. The reduction in computational time of the nodes for ENA.b was much better than that of ENA.a but still after eight nodes the computational time began to increase showing that the sixteen nodes computation was almost the same as the one node computation. The predominant cause of this delay could be found in the waiting time between two successive generations.

The speed-up factor for this version of the program is plotted in Figure 5.

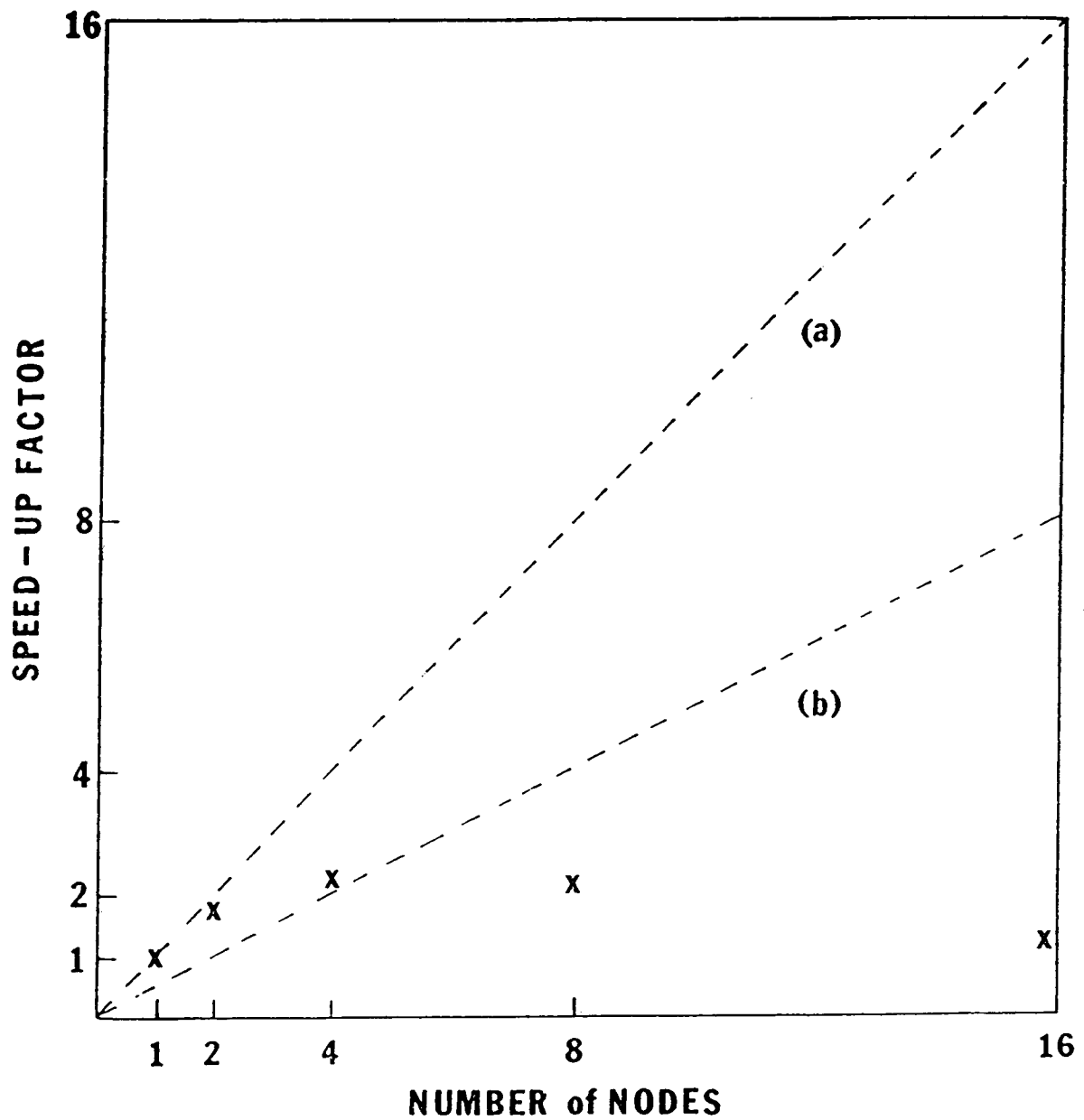


Figure 5.

Measured Speed-Up Factor (indicated by the symbol X)  
of CES with ENA.b Using 1, 2, 4, 8, 16 Nodes.  
Lines (a) and (b) Are the Theoretical and  
the Real Speed-Up Factor Respectively.

## Description of ENA.c

In a further attempt to reduce the computational time, a new version of the program for the nodes, called ENA.c, was developed.

Since it was known that the waiting time between two generations was the crucial constraint for the flow of the calculation, a final solution was formulated by coordinating the nodes to execute the entire computation without the need of waiting for a message after each generation. At the end of each generation a message containing the k-effective and the standard deviation was sent to the host that subsequently collected all the information and printed the resultant k-effective and standard deviation for the generation.

This scheme was implemented by adding subroutine BANKER to ENA.b, renaming it program ENA.c, and then performing ENA.c in parallel in each node. As a result, the neutron bank as well as the fission bank were divided among the nodes. Each node was independent from the others and the calculation throughout the generations was such that each node was available to get the information for the next generation within itself.

All of the generation loop described in Figure 2 was included in ENA.c. On the other hand, this method presented some problems regarding the statistics associated with a Monte Carlo criticality computation.

Specifically in ENA.c the neutron bank is split among the nodes and never reconstructed after each generation. This procedure, if applied to extreme limits (i.e. splitting the neutron bank in a large number of small parts), will affect the quality of the statistics of the problem and, as a result, the precision of the results obtained. A general rule of thumb<sup>4</sup> was constructed which stated that in order to obtain good statistical results, a minimum of forty ( 40 ) particles per node should be used. The maximum number of nodes that was possible to use was eight. In order to increase the number of nodes implemented and to preserve forty particles per node, the number of particles per generation was doubled and fifty-three ( 53 ) generations, instead of the usual one hundred and three (103), were performed. The total number of histories was preserved ( 32000 ) and sixteen nodes were used in parallel.

This technique could not be expanded further because the nature of the criticality problem prohibited it. Specifically, in a criticality problem, the initial source distribution is guessed. In order to achieve what is usually called the fundamental mode for the eigenvalue, at least thirty ( 30 ) generations must be computed<sup>4</sup>. Therefore, 53 is the minimum number of generations considered in this work.

## Results of CES with ENA.c

ENA.c was executed using one, two, four, eight nodes with 320 particles per generation and 103 generations. For sixteen nodes, 640 particles per generation and 53 generations were used.

The values of the k-effective and the corresponding standard deviations are shown in Table 8. The computational time for the nodes, the host, and the relative speed-up factor are shown in Table 9.

TABLE 8  
AVERAGE K-EFFECTIVE AND STANDARD DEVIATION OF CES  
WITH ENA.c USING 1, 2, 4, 8, 16 NODES

| Number of<br>Nodes | Average<br>K-Effective | Standard<br>Deviation |
|--------------------|------------------------|-----------------------|
| 1                  | 1.06520                | 6.2 -03               |
| 2                  | 1.06859                | 5.8 -03               |
| 4                  | 1.06170                | 6.4 -03               |
| 8                  | 1.06573                | 6.2 -03               |
| 16                 | 1.06679                | 6.1 -03               |

TABLE 9

COMPUTATIONAL TIME AND SPEED-UP FACTOR OF CES  
WITH ENA.c USING 1, 2, 4, 8, 16 NODES

| Number of<br>Nodes | Host Time<br>( Sec. ) | Node Time<br>( Sec. ) | Speed-Up<br>Factor |
|--------------------|-----------------------|-----------------------|--------------------|
| 1                  | 6                     | 242                   | 1                  |
| 2                  | 6                     | 122                   | 1.98               |
| 4                  | 7                     | 63                    | 3.84               |
| 8                  | 7                     | 37                    | 6.54               |
| 16                 | 7                     | 28                    | 8.64               |

The results obtained concurred with the those previously obtained. The computational time for the sixteen node case was comparable to that obtained on the I.B.M. computer. The speed-up factor values were well above the realistic limit. Assuming that it could be possible to extend the calculation to a larger number of nodes ( i.e. not considering the limits imposed by the statistical nature of the problem ), it should be possible to achieve a computational time much better than that possible with the I.B.M. computer.

The speed-up factor versus the number of nodes used is plotted in Figure 6.

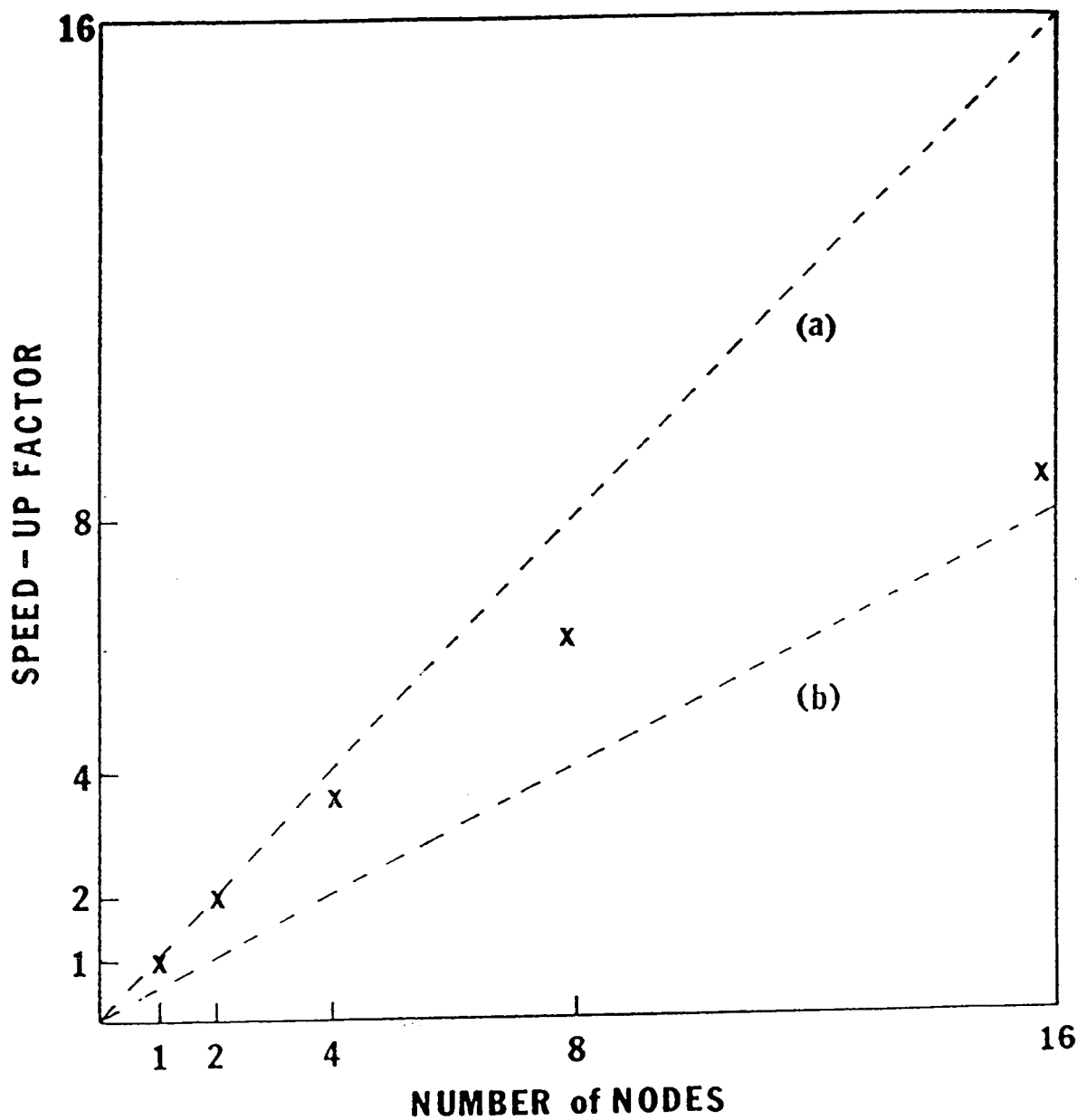


Figure 6.

Measured Speed-Up Factor (indicated by the symbol X)  
of CES with ENA.c Using 1, 2, 4, 8, 16 Nodes.  
Lines (a) and (b) Are the Theoretical and  
the Real Speed-Up Factor Respectively.

## A Modified Version of ENA.c

As previously discussed, the two principal problems encountered during the calculation were the delay between two successive generations and the restrictions imposed by the statistical nature of the problem. In order to avoid both of these, a new version of ENA.c, called ENA.c.mod, was developed. The structure of ENA.c.mod is the same as ENA.c, but the way in which subroutine BANKER is executed is different. In ENA.c, the neutron bank was split among the nodes and the reconstruction was made inside each node by the subroutine BANKER. This procedure introduced problems related to the statistical quality of the results in that the number of neutrons implemented in each node needs to be at least forty. In order to avoid this problem, the reconstruction of the entire neutron bank was done in ENA.c.mod. Instead of performing subroutine BANKER, each node, at the end of each generation, sent the neutron bank to node "zero". Node "zero" then collected all the incoming information, reconstructed the neutron bank, and sent back to each node the new complete neutron bank. At this point, each node was ready to execute subroutine BANKER and subsequently to start the new generation.

Although the introduction of messages between the nodes after each generation and the execution of the whole

neutron bank inside subroutine BANKER increases the total computational time, a better speed-up factor is expected.

#### Results of CES and ENA.c.mod

The results expected were not realized. Again the program showed good values for the k-effective and for the standard deviation respectively. However, the computational time of the host board was essentially the same as the one obtained with ENA.c. On the other hand, the computational time of the nodes was not even comparable. The program was executed using one, two, and four nodes. The results for the k-effective and for the standard deviation are shown in Table 10. The computational time for the host and for the nodes, and the related speed up factor are shown in Table 11. Due to the fact that only four nodes were used in this calculation, the speed-up factor is not presented.

TABLE 10

AVERAGE K-EFFECTIVE AND STANDARD DEVIATION OF CES  
WITH ENA.c.mod USING 1, 2, 4 NODES

| Number of<br>Nodes | Average<br>K-Effective | Standard<br>Deviation |
|--------------------|------------------------|-----------------------|
| 1                  | 1.06754                | 6.2 -03               |
| 2                  | 1.07158                | 5.5 -03               |
| 4                  | 1.06013                | 5.1 -03               |

TABLE 11

COMPUTATIONAL TIME AND SPEED-UP FACTOR OF CES  
WITH ENA.c.mod USING 1, 2, 4 NODES

| Number of<br>Nodes | Host Time<br>( Sec. ) | Node Time<br>( Sec. ) | Speed-Up<br>Factor |
|--------------------|-----------------------|-----------------------|--------------------|
| 1                  | 6                     | 262                   | 1                  |
| 2                  | 6                     | 174                   | 1.51               |
| 4                  | 7                     | 142                   | 1.86               |

## CHAPTER IV

### CONCLUSIONS AND FUTURE WORK

#### Conclusions

Four different versions of the CES program have been implemented and evaluated in the NCUBE: ENA.a, ENA.b, ENA.c, and ENA.c.mod.

The results in terms of quality and agreement with those of the I.B.M. computer are quite good and independent of the number of nodes used. The process of parallelization of the program inside the NCUBE does not significantly affect the accuracy of the results. The same cannot be said for the computational time and the speed-up factor.

ENA.a is quite slow and the total computational time is approximately twenty times larger than that for the I.B.M.. Most of the computational time of the nodes is spent waiting for a message from the host between two generations. In effect, the host is performing more work than the nodes.

Although an improvement was obtained using ENA.b, the computational time still is not comparable with that obtained with the I.B.M.. Nevertheless, the great reduction in the host board computational time is an indication that the only way to solve the problem was to put the entire generation loop in parallel.

The best results were obtained with ENA.c and, without the restrictions imposed by the statistical nature of the problem, it could have been possible to obtain a computational time much shorter than that obtained with the I.B.M. computer. ENA.c showed that, not only is it important to have the majority of the program execute in parallel, but also that a fine balance exist between the part of the program that runs on the nodes and the part that runs on the host board.

The modified version of ENA.c showed that in order to get a good speed-up factor, it is not sufficient to eliminate the calculations performed on the host between two successive generations. Indeed, it is crucial to coordinate the nodes into executing the program without any kind of delay between the generations.

#### Comments About the Current Version of the NCUBE

The NCUBE parallel processor computer is still in its " infancy " and a lot of work could be done to improve the performance of this machine.

The host board is quite slow in performing calculations with the result that the response of the nodes, especially if they are waiting for a message, is degraded.

It is not possible to send two or more messages in a row to the host or to the same node. If two messages must

be sent at the same time, a control routine with the purpose to check if the message has been received and read should be provided between the two messages.

The nodes and the host are connected only through node " zero ". As a result, the messages coming from the other nodes must follow a path (message channel) inside the cube itself. This situation is one of the principal causes of delay in the computational time and, sometimes, of the failure of the program itself to execute due to the overloading of the message channel.

#### Future Work

It appears that nothing can be done to improve further the calculation of this version of CES on the NCUBE. On the other hand, an improvement might be possible by changing the input data of the problem in order to have most of the computational time required for the calculation spent in tracking the particle. This could be achieved by considering a fictitious material with a small fission probability and a small absorption probability resulting in a particle with a longer life. In this manner, the delays introduced by the execution of the other subroutines would require less time.

Another approach to the problem could be to increase the total number of particles computed in the problem with the clear advantage to be able to use a larger number of

nodes without incurring in the limit imposed by the statistic characteristics of the problem.

The problem considered herein, a criticality calculation, does not preclude other applications of the NCUBE using the Monte Carlo method. For example, a shielding problem where the source is fixed may be more appropriate and feasible for parallel computation.

Another application might be the connection of the NCUBE with a mainframe computer that would serve the function of the host board (i.e. a hybrid computer). In this way the calculation of the sequential part of a program would be much faster and, consequently, would not slow down the overall calculation.

## BIBLIOGRAPHY

## BIBLIOGRAPHY

1. L.M. Petrie and N.F. Landers, "KENO V.a an Improved Monte Carlo Criticality Program with Supergrouping," ORNL/NUREG/CSD-2/V1/R2, December 1984.
2. C.L. Seitz, "The Cosmic Cube," Communications of the ACM, Volume 28, Number 1, January 1985.
3. G.C. Fox and S.W. Otto, "Algorithms for concurrent processors," Physics Today, May 1984.
4. L.M. Petrie, personal communications, January 1986.
5. C.L. Seitz, "Concurrent VLSI Architecture," IEEE Transaction Computers, Vol. C-33, NO. 12, December 1984.
6. W. Fullerton, "Function Rangen," Los Alamos Scientific Laboratory, April 1977.

## APPENDIX

# APPENDIX

## SAMPLE PROBLEM OUTPUT USING KENO V.a (16 GROUP)

TABLE A.1

KENO V.a (16 GROUP) SAMPLE PROBLEM OUTPUT  
( K-EFFECTIVE AND STANDARD DEVIATION )

| Gener. | Generation<br>K-Effective | Average<br>K-Effective | Standard<br>Deviation |
|--------|---------------------------|------------------------|-----------------------|
| 1      | 1.00099 +00               | 1.00000 +00            | 0.0                   |
| 2      | 1.01929 +00               | 1.00000 +00            | 0.0                   |
| 3      | 1.05845 +00               | 1.05845 +00            | 0.0                   |
| 4      | 1.09199 +00               | 1.07522 +00            | 1.67730 -02           |
| 5      | 1.03787 +00               | 1.06277 +00            | 1.57617 -02           |
| 6      | 1.17785 +00               | 1.09154 +00            | 3.08507 -02           |
| 7      | 1.11871 +00               | 1.09697 +00            | 2.45115 -02           |
| 8      | 1.12159 +00               | 1.10108 +00            | 2.04286 -02           |
| 9      | 1.09321 +00               | 1.09995 +00            | 1.73047 -02           |
| 10     | 1.02079 +00               | 1.09006 +00            | 1.79539 -02           |
| 11     | 1.10016 +00               | 1.09118 +00            | 1.58785 -02           |
| 12     | 1.13835 +00               | 1.09590 +00            | 1.49669 -02           |
| 13     | 1.03355 +00               | 1.09023 +00            | 1.46777 -02           |
| 14     | 1.07227 +00               | 1.08873 +00            | 1.34803 -02           |
| 15     | 1.06103 +00               | 1.08660 +00            | 1.25790 -02           |
| 16     | 1.04521 +00               | 1.08365 +00            | 1.20155 -02           |
| 17     | 1.23390 +00               | 1.09366 +00            | 1.50159 -02           |
| 18     | 1.12346 +00               | 1.09552 +00            | 1.41675 -02           |
| 19     | 1.13850 +00               | 1.09805 +00            | 1.35493 -02           |
| 20     | 1.06046 +00               | 1.09596 +00            | 1.29491 -02           |
| 21     | 1.03059 +00               | 1.09252 +00            | 1.27224 -02           |
| 22     | 1.04260 +00               | 1.09003 +00            | 1.23242 -02           |
| 23     | 1.09774 +00               | 1.09039 +00            | 1.17289 -02           |
| 24     | 1.09042 +00               | 1.09039 +00            | 1.11894 -02           |
| 25     | 1.07034 +00               | 1.08952 +00            | 1.07260 -02           |
| 26     | 1.00954 +00               | 1.08619 +00            | 1.07942 -02           |
| 27     | 1.05068 +00               | 1.08477 +00            | 1.04516 -02           |
| 28     | 1.10188 +00               | 1.08543 +00            | 1.00619 -02           |
| 29     | 1.02906 +00               | 1.08334 +00            | 9.90918 -03           |
| 30     | 1.15009 +00               | 1.08572 +00            | 9.83949 -03           |
| 31     | 1.07341 +00               | 1.08530 +00            | 9.50761 -03           |
| 32     | 1.17108 +00               | 1.08816 +00            | 9.62145 -03           |
| 33     | 1.14848 +00               | 1.09010 +00            | 9.50833 -03           |
| 34     | 1.10074 +00               | 1.09043 +00            | 9.21120 -03           |
| 35     | 1.08410 +00               | 1.09024 +00            | 8.93034 -03           |

TABLE A.1 (Continued)

| Gener. | Generation<br>K-Effective | Average<br>K-Effective | Standard<br>Deviation |
|--------|---------------------------|------------------------|-----------------------|
| 36     | 1.01594 +00               | 1.08806 +00            | 8.93418 -03           |
| 37     | 1.06579 +00               | 1.08742 +00            | 8.70086 -03           |
| 38     | 1.11272 +00               | 1.08812 +00            | 8.48462 -03           |
| 39     | 1.07819 +00               | 1.08785 +00            | 8.25598 -03           |
| 40     | 1.12027 +00               | 1.08871 +00            | 8.08010 -03           |
| 41     | 1.08294 +00               | 1.08856 +00            | 7.87011 -03           |
| 42     | 1.20257 +00               | 1.09141 +00            | 8.18695 -03           |
| 43     | 1.15137 +00               | 1.09287 +00            | 8.11487 -03           |
| 44     | 1.01824 +00               | 1.09019 +00            | 8.11767 -03           |
| 45     | 1.08127 +00               | 1.09087 +00            | 7.92934 -03           |
| 46     | 1.09142 +00               | 1.09088 +00            | 7.74693 -03           |
| 47     | 1.05460 +00               | 1.09007 +00            | 7.61724 -03           |
| 48     | 1.04852 +00               | 1.08917 +00            | 7.50253 -03           |
| 49     | 1.04212 +00               | 1.08817 +00            | 7.40935 -03           |
| 50     | 1.02155 +00               | 1.08678 +00            | 7.38526 -03           |
| 51     | 1.13073 +00               | 1.08768 +00            | 7.29023 -03           |
| 52     | 1.14428 +00               | 1.08881 +00            | 7.23126 -03           |
| 53     | 1.13917 +00               | 1.08980 +00            | 7.15894 -03           |
| 54     | 1.09019 +00               | 1.08980 +00            | 7.01805 -03           |
| 55     | 1.10659 +00               | 1.09012 +00            | 6.89025 -03           |
| 56     | 1.17746 +00               | 1.09174 +00            | 6.95467 -03           |
| 57     | 1.09669 +00               | 1.09183 +00            | 6.82947 -03           |
| 58     | 1.14350 +00               | 1.09275 +00            | 6.76839 -03           |
| 59     | 1.07734 +00               | 1.09248 +00            | 6.65543 -03           |
| 60     | 9.92155 -01               | 1.09075 +00            | 6.76459 -03           |
| 61     | 1.07896 +00               | 1.09056 +00            | 6.65186 -03           |
| 62     | 1.03396 +00               | 1.08962 +00            | 6.60749 -03           |
| 63     | 1.07420 +00               | 1.08937 +00            | 6.50227 -03           |
| 64     | 1.12174 +00               | 1.08989 +00            | 6.41960 -03           |
| 65     | 1.02373 +00               | 1.08884 +00            | 6.40375 -03           |
| 66     | 1.01644 +00               | 1.08771 +00            | 6.40257 -03           |
| 67     | 1.09196 +00               | 1.08777 +00            | 6.30408 -03           |
| 68     | 1.12617 +00               | 1.08836 +00            | 6.23543 -03           |
| 69     | 1.17857 +00               | 1.08970 +00            | 6.28647 -03           |
| 70     | 1.10382 +00               | 1.08991 +00            | 6.19818 -03           |
| 71     | 1.06493 +00               | 1.08955 +00            | 6.11700 -03           |
| 72     | 1.08642 +00               | 1.08950 +00            | 6.03026 -03           |
| 73     | 1.01821 +00               | 1.08850 +00            | 6.02785 -03           |
| 74     | 1.14644 +00               | 1.08930 +00            | 5.99982 -03           |
| 75     | 1.04747 +00               | 1.08873 +00            | 5.94465 -03           |
| 76     | 1.07547 +00               | 1.08855 +00            | 5.86829 -03           |
| 77     | 1.17717 +00               | 1.08973 +00            | 5.90865 -03           |
| 78     | 1.00667 +00               | 1.08864 +00            | 5.93163 -03           |
| 79     | 1.20123 +00               | 1.09010 +00            | 6.03451 -03           |

TABLE A.1 (Continued)

| Gener. | Generation<br>K-Effective | Average<br>K-Effective | Standard<br>Deviation |
|--------|---------------------------|------------------------|-----------------------|
| 80     | 1.15024 +00               | 1.09087 +00            | 6.00449 -03           |
| 81     | 1.07009 +00               | 1.09061 +00            | 5.93505 -03           |
| 82     | 1.07123 +00               | 1.09037 +00            | 5.86452 -03           |
| 83     | 1.13578 +00               | 1.09093 +00            | 5.82059 -03           |
| 84     | 9.85745 -01               | 1.08964 +00            | 5.89044 -03           |
| 85     | 1.11781 +00               | 1.08998 +00            | 5.82853 -03           |
| 86     | 1.04231 +00               | 1.08942 +00            | 5.78636 -03           |
| 87     | 1.08690 +00               | 1.08939 +00            | 5.71816 -03           |
| 88     | 1.12986 +00               | 1.08986 +00            | 5.67158 -03           |
| 89     | 1.06341 +00               | 1.08955 +00            | 5.61387 -03           |
| 90     | 1.13620 +00               | 1.09008 +00            | 5.57562 -03           |
| 91     | 1.14653 +00               | 1.09072 +00            | 5.54874 -03           |
| 92     | 1.10857 +00               | 1.09091 +00            | 5.49120 -03           |
| 93     | 1.04251 +00               | 1.09038 +00            | 5.45672 -03           |
| 94     | 1.12311 +00               | 1.09074 +00            | 5.40828 -03           |
| 95     | 1.04338 +00               | 1.09023 +00            | 5.37399 -03           |
| 96     | 1.11648 +00               | 1.09051 +00            | 5.32291 -03           |
| 97     | 1.07189 +00               | 1.09031 +00            | 5.27148 -03           |
| 98     | 1.07331 +00               | 1.09014 +00            | 5.21775 -03           |
| 99     | 1.11850 +00               | 1.09043 +00            | 5.17325 -03           |
| 100    | 1.10443 +00               | 1.09057 +00            | 5.12161 -03           |
| 101    | 1.00210 +00               | 1.08968 +00            | 5.14767 -03           |
| 102    | 1.09455 +00               | 1.08973 +00            | 5.09615 -03           |
| 103    | 1.08187 +00               | 1.08965 +00            | 5.04610 -03           |

TABLE A.2

CONFIDENCE INTERVALS ( 67, 95, 99 PER CENT)  
OF KENO V.a (16 GROUP)

| No. Initial<br>Gener.<br>Skipped   | Average<br>K-Effective | Deviation                            | 67 Per Cent<br>Confidence Interval |
|------------------------------------|------------------------|--------------------------------------|------------------------------------|
| 95 Per Cent<br>Confidence Interval |                        | 99 Per Cent<br>Confidence Interval   | Number of<br>Histories             |
| 3<br>1.07980 to 1.10012            | 1.08996                | + or - 0.00508<br>1.07472 to 1.10520 | 1.08488 to 1.09504<br>30000        |
| 4<br>1.07967 to 1.10020            | 1.08994                | + or - 0.00513<br>1.07454 to 1.10533 | 1.08481 to 1.09507<br>27900        |
| 5<br>1.08016 to 1.10078            | 1.09047                | + or - 0.00516<br>1.07500 to 1.10594 | 1.08531 to 1.09563<br>29400        |
| 6<br>1.07931 to 1.09983            | 1.08957                | + or - 0.00513<br>1.07418 to 1.10496 | 1.08444 to 1.09470<br>29100        |
| 7<br>1.07892 to 1.09962            | 1.08927                | + or - 0.00517<br>1.07374 to 1.10479 | 1.08409 to 1.09444<br>28800        |
| 8<br>1.07849 to 1.09936            | 1.08893                | + or - 0.00522<br>1.07327 to 1.10458 | 1.08371 to 1.09414<br>28500        |
| 9<br>1.07833 to 1.09943            | 1.08888                | + or - 0.00527<br>1.07306 to 1.10470 | 1.08361 to 1.09415<br>28200        |
| 10<br>1.07905 to 1.10017           | 1.08961                | + or - 0.00528<br>1.07378 to 1.10545 | 1.08433 to 1.09489<br>27900        |
| 11<br>1.07883 to 1.10017           | 1.08950                | + or - 0.00534<br>1.07349 to 1.10550 | 1.08416 to 1.09483<br>27600        |
| 12<br>1.07823 to 1.09970           | 1.08896                | + or - 0.00537<br>1.07286 to 1.10506 | 1.08359 to 1.09433<br>27300        |
| 17<br>1.07826 to 1.09964           | 1.08895                | + or - 0.00535<br>1.07291 to 1.10498 | 1.08360 to 1.09429<br>25800        |
| 22<br>1.07848 to 1.10063           | 1.08956                | + or - 0.00554<br>1.07294 to 1.10617 | 1.08402 to 1.09509<br>24300        |

TABLE A.2 (Continued)

| No. Initial<br>Gener.<br>Skipped | Average<br>K-Effective             | Deviation          | 67 Per Cent<br>Confidence Interval |                        |
|----------------------------------|------------------------------------|--------------------|------------------------------------|------------------------|
|                                  | 95 Per Cent<br>Confidence Interval |                    | 99 Per Cent<br>Confidence Interval | Number of<br>Histories |
| 27                               | 1.09125                            | + or - 0.00578     | 1.08547 to 1.09703                 |                        |
| 1.07970 to 1.10281               |                                    | 1.07392 to 1.10859 |                                    | 22800                  |
| 32                               | 1.09028                            | + or - 0.00595     | 1.08433 to 1.09623                 |                        |
| 1.07837 to 1.10218               |                                    | 1.07242 to 1.10813 |                                    | 21300                  |
| 37                               | 1.09083                            | + or - 0.00623     | 1.08460 to 1.09706                 |                        |
| 1.07837 to 1.10328               |                                    | 1.07215 to 1.10951 |                                    | 19800                  |
| 42                               | 1.08849                            | + or - 0.00644     | 1.08205 to 1.09494                 |                        |
| 1.07560 to 1.10138               |                                    | 1.06916 to 1.10783 |                                    | 18300                  |
| 47                               | 1.08931                            | + or - 0.00679     | 1.08252 to 1.09610                 |                        |
| 1.07574 to 1.10288               |                                    | 1.06895 to 1.10967 |                                    | 16800                  |
| 52                               | 1.09047                            | + or - 0.00710     | 1.08337 to 1.09757                 |                        |
| 1.07627 to 1.10467               |                                    | 1.06917 to 1.11178 |                                    | 15300                  |
| 57                               | 1.08704                            | + or - 0.00754     | 1.07950 to 1.09459                 |                        |
| 1.07196 to 1.10213               |                                    | 1.06441 to 1.10967 |                                    | 13800                  |
| 62                               | 1.08969                            | + or - 0.00790     | 1.08179 to 1.09759                 |                        |
| 1.07389 to 1.10549               |                                    | 1.06599 to 1.11339 |                                    | 12300                  |
| 67                               | 1.09303                            | + or - 0.00850     | 1.08454 to 1.10153                 |                        |
| 1.07604 to 1.11003               |                                    | 1.06754 to 1.11852 |                                    | 10800                  |
| 72                               | 1.08998                            | + or - 0.00935     | 1.08063 to 1.09933                 |                        |
| 1.07128 to 1.10868               |                                    | 1.06193 to 1.11803 |                                    | 9300                   |
| 77                               | 1.08941                            | + or - 0.00987     | 1.07954 to 1.09928                 |                        |
| 1.06966 to 1.10915               |                                    | 1.05979 to 1.11902 |                                    | 7800                   |
| 82                               | 1.08691                            | + or - 0.00969     | 1.07722 to 1.09661                 |                        |
| 1.06753 to 1.10630               |                                    | 1.05783 to 1.11599 |                                    | 6300                   |
| 87                               | 1.09104                            | + or - 0.00992     | 1.08112 to 1.10096                 |                        |
| 1.07120 to 1.11088               |                                    | 1.06128 to 1.12080 |                                    | 4800                   |

TABLE A.2 (Continued)

| No. Initial<br>Gener.<br>Skipped   | Average<br>K-Effective             | Deviation                            | 67 Per Cent<br>Confidence interval |
|------------------------------------|------------------------------------|--------------------------------------|------------------------------------|
| 95 Per Cent<br>Confidence Interval | 99 Per Cent<br>Confidence Interval |                                      | Number of<br>Histories             |
| 92<br>1.05640 to 1.10217           | 1.07928                            | + or - 0.01144<br>1.04495 to 1.11361 | 1.06784 to 1.09073<br>3300         |
| 97<br>1.04565 to 1.11260           | 1.07912                            | + or - 0.01674<br>1.02891 to 1.12933 | 1.06239 to 1.09586<br>1800         |

## VITA

Daniele Ugolini was born in Cesena, Italy, the 24th of July in 1956. He attended the Università degli Studi di Bologna, where he obtained the degree of Ingegnere Nucleare in 1981 and the Abilitazione alla Professione di Ingegnere in 1982. Since 1985, he has been enrolled in the Nuclear Engineering Department of the University of Tennessee, and has been a graduate research assistant at the Oak Ridge National Laboratory, in the Division of Computer and Telecommunications.