MODIFICATIONS AND ADDITIONS TO NASTRAN AT MARSHALL SPACE FLIGHT CENTER

1969 - 1972

By Archie J. Jordan, Jr., and William G. Ward
Computer Sciences Corporation

SUMMARY

Modifications made to NASTRAN at Marshall Space Flight Center, Huntsville, Alabama, can be grouped into two general classes:

1. Those modifications to NASTRAN code which result from the location, identification, and correction of errors or operational problem areas.

2. Addition of new capability which is structural, executive system oriented, or utility in nature.

In addition to these modifications to NASTRAN, several important studies have been conducted at MSFC relative to NASTRAN efficiencies and inefficiencies.

INTRODUCTION

A brief description on some of the major modifications to NASTRAN will be discussed. These are modifications which have been made to various levels of NASTRAN and which are currently incorporated into MSFC's version of Level 12. Because of the significance of two studies conducted relative to NASTRAN efficiencies and inefficiencies, more time will be allotted to discussion of these studies and their results.

Modifications and Additions

(1) A post-processor was written to stack SC-4020 plot commands generated by NASTRAN onto drum, preceded by the MSFC standard identification frame and followed by the MSFC standard end job frame. This was written in the form of a utility subroutine, but performs as a post-processor in that it is executed after all other NASTRAN activities have been completed. This modification

was made at the request of MSFC Computation Laboratory management in order to solve SC-4020 tape labeling problems on NASTRAN runs coming in from remote terminals.

(2) The method of obtaining accumulated CPU time for triangular decomposition "time-to-go" checks and other time checks in NASTRAN was changed due to a change in the 1108 Executive System's PCT (Program Control Table). Accumulated CPU time is now found by interrogating a locally maintained system routine since the format of the PCT appears to be highly evolutionary.

(3) Additions and modifications to the NASTRAN Executive System were made to allow the capability of inputting NASTRAN Executive, Case Control, and Bulk Data decks from magnetic tape instead of cards. The user may also require the flexibility of inputting the Executive and Case Control decks from cards and the Bulk Data decks from a non-UMF formatted tape. This case often occurs when a user has generated a large amount of bulk data in a small FORTRAN data conversion program and the number of punched cards is prohibitively large. Both of these options were made available.

(4) A NASTRAN DMAP functional module, OUTPUT 2, was developed to write on magnetic tape the non-zero terms of the lower triangular portion of a NASTRAN symmetric matrix. Options have been recently included in this module which allow for writing on tape or punching on cards non-zero terms of a general matrix in a user specified FORTRAN format.

(5) A NASTRAN DMAP functional module was developed to punch out NASTRAN matrices in the NASTRAN Bulk Data DMI card format. A module similar to this will be delivered in Level 15 by NSMO. However, the module developed here will punch out single and double precision values instead of just single precision values and will also punch complex matrices, such as complex eigenvectors.

(6) Modifications to appropriate subroutines in NASTRAN's Executive System have been made which will generate a forced checkpoint if the run time on the run card is greater than or equal to 30 minutes. The modified code will detect the run time, assign a checkpoint tape, generate the card image "CHKPNT YES" in the Executive Control deck and allow for the punching of the restart dictionary. This modification was developed at the request of the MSFC NASTRAN Technical Monitor who felt some users were not using the checkpoint feature at all and, as a result, were needlessly using CPU time in subsequent runs. This inhibits all users from getting machine time. With the automatic checkpoint feature, it is not necessary for the user to include the "CHKPNT YES" option in his run. A close kin to this capability is the development of the NASTRAN Executive System capability of forcing "undeformed structural plots" for NASTRAN runs which appear to be large (greater

than 30 minutes). Again, it was felt that too many users were creating large models and making many long runs without ever checking their model by looking at the undeformed plots. This capability is now being developed.

(7) A modification was made to the real eigenvalue module to print out the eigenvalue iteration information as computed and to force exit from the algorithm as soon as the first root is obtained. This capability is often desirable in buckling problems where only the first mode is desired and run times are long.

(8) Several functional modules have been developed and incorporated into MSFC's NASTRAN version which allow for a complex eigenvalue solution for a spinning flexible body such as NASA's SKYLAB. The user DMAP sequence for this capability is available, and a detailed user paper is being presented in this colloquium by Dr. Jayant S. Patel of Teledyne-Brown Engineering.

(9) Two DMAP functional modules have been developed which will allow for the capability of substructuring in NASTRAN as defined by a segment of the MSFC structural analysis NASTRAN user community. Basically, one module unpacks and writes NASTRAN matrices on a magnetic tape; the other reads matrices from magnetic tape, packs them in NASTRAN matrix format and generates corresponding NASTRAN output data blocks. These data blocks (matrices) are then utilized in a DMAP sequence or rigid format to accomplish the substructuring task. This approach can be used to model a structure with up to eight substructures.

Studies on NASTRAN Efficiencies and Inefficiencies

Two significant studies will be presented relative to NASTRAN efficiencies and inefficiencies:

1. NASTRAN CPU Time Versus Available Core for Triangular Decomposition.

2. Investigation of NTRAN I/O Usage in NASTRAN.

The first study was initiated in order to discover some quantitative relationship between "core" solutions and "spill logic" solutions for NASTRAN problems requiring triangular decomposition on MSFC's UNIVAC 1108 EXEC VIII. For large NASTRAN problems, a substantial amount of the total computing time is associated with triangular decomposition. NASTRAN performs this decomposition task utilizing all available core. It dynamically allocates its large arrays at execute time by employing an "open core" concept. The use of fixed dimensions for large arrays is avoided since this automatically restricts the size of the problems that can be solved. Instead, FORTRAN routines are programmed to allocate space as required and to use all of core available to user programs. When all of core has

been allocated, NASTRAN then uses "spill logic" to transfer data to scratch files and back to core again. NASTRAN calculates the amount of available core ("open core") for allocation purposes by determining the number of words available between the origin of "open core" and the end of core.

To demonstrate the relationship between CPU time and available core for triangular decomposition, it was decided to develop a problem which would yield an "all core" decomposition for 65K on the UNIVAC 1108 and then decrease the amount of available core to NASTRAN in subsequent runs. This decrease in the amount of available core would cause the "all core" solution to become a "spill logic" solution. Comparisons of CPU run times would give some insight into the excessive run times being experienced at this center for decomposition problems.

Using Version 12 of NASTRAN, the amount of "open core" can be reduced with relative ease. By choosing appropriate options on the UNIVAC 1108 Executive Control XQT card, a user can limit the amount of core available for his own NASTRAN problem. Four runs were submitted for various size models with available core figures of:

1. 65K
2. 55K
3. 50K
4. 42K

The structural problem developed to use as a test case for this study was a rectangular panel made up of quadrilateral plate elements with a bandwidth of 169 and 480 degrees of freedom. A buckling analysis was performed on the panel, and only one eigenvalue was obtained. The CPU run times for the four test runs were evaluated by three different methods:

1. Total CPU run time versus available core.

2. CPU time spent in the NASTRAN functional module, READ, versus available core. This module performs the matrix decomposition and eigenvalue solution.

3. CPU time spent only in the decomposition portion of READ versus available core. It is this portion of the READ module which contains the "spill logic".

These run times are given in Attachment A, and the results are plotted in Figures 1, 2, and 3 to graphically illustrate the increase in CPU time in all program areas as the amount of available core is steadily decreased. Since the aim of this study was to determine the increase in CPU time due to the use of "spill logic" in triangular decomposition, it is important to note that the CPU time for

the entire program increased 2.4 times (240%) when core was decreased by 1/3 (33 1/3%). The total decomposition CPU time increased 6.5 times (650%) due to the same decrease in available core. It is also important to notice that the increase in CPU time in the three areas mentioned is not linear; i.e., the ratio of the change in CPU time to the change in available core is not constant.

Investigation of NTRAN I/O Usage in NASTRAN

The second study was initiated in order to determine if NASTRAN's NTRAN I/O package was hurting NASTRAN performance at MSFC since it was known that the NTRAN I/O was excessively inefficient. A series of test runs was made in which 249 word records (FORTRAN buffer size) were written from core to drum by MSFC's NTRAN, NASTRAN's NTRAN, and FORTRAN binary I/O operations. In addition, two approaches to using NTRAN were tested and were designated NTRAN 1 and NTRAN 2 as follows:

NTRAN 1 - A method of NTRAN usage in which a wait in NTRAN occurs until all previous operations, for the specified logical unit, are complete before stacking any further operations or returning to the user's program. This approach is the method employed by NASTRAN.

NTRAN 2 - A method of NTRAN usage in which the transmission status word is tested in a FORTRAN loop until an error occurs or the transmission is successfully completed.

The results from one such series of tests follow. In these tests, each I/O method was allowed to write as many records as possible (up to 4000) in two minutes (CPU).

249 WORD RECORDS (MSFC NTRAN)

METHOD	DEVICE	RECORDS TRANSMITTED	CPU TIME
NTRAN 1	DRUM	577	2:00 Min.
NTRAN 2	DRUM	284	2:00 Min.
FORTRAN	DRUM	4000	1:17(Min.Sec.)

249 WORD RECORDS (NASTRAN NTRAN)

METHOD	DEVICE	RECORDS TRANSMITTED	CPU TIME
NTRAN 1	DRUM	4000	25 Sec.
NTRAN 2	DRUM	1523	2:00 Min.
FORTRAN	DRUM	4000	1:11 (Min.Sec.)

It was determined that NTRAN 1 or the wait and unstack approach is vastly superior to the NTRAN 2 programming approach and substantially better than FORTRAN binary I/O. It was also determined that NASTRAN's NTRAN is substantially more efficient than MSFC's NTRAN and does not inhibit NASTRAN's I/O performance.

CONCLUDING REMARKS

Modifications and additions to the local MSFC standard version of NASTRAN are only one area of NASTRAN maintenance which occurs at MSFC; however, it is probably the most significant since it hopefully plays a part in keeping NASTRAN a meaningful, useful, and evolutionary tool for the structural analysts at MSFC instead of the obsolete monster it could become if modification and addition could not be performed on this remarkable system.

ATTACHMENT A

A. TOTAL CPU RUN TIME

UPPER CORE LIMIT	CPU TIME
65K	20.1 MIN.
55K	25.5 MIN.
50K	31.9 MIN.
42K	48.2 MIN.

B. CPU TIME IN READ

UPPER CORE LIMIT	CPU TIME
65K	13.5 MIN.
55K	19.3 MIN.
50K	25.2 MIN.
42K	41.3 MIN.

C. CPU TIME IN DECOMPOSITION

UPPER CORE LIMIT	CPU TIME
65K	4.7 MIN.
55K	9.7 MIN.
50K	15.7 MIN.
42K	30.5 MIN.

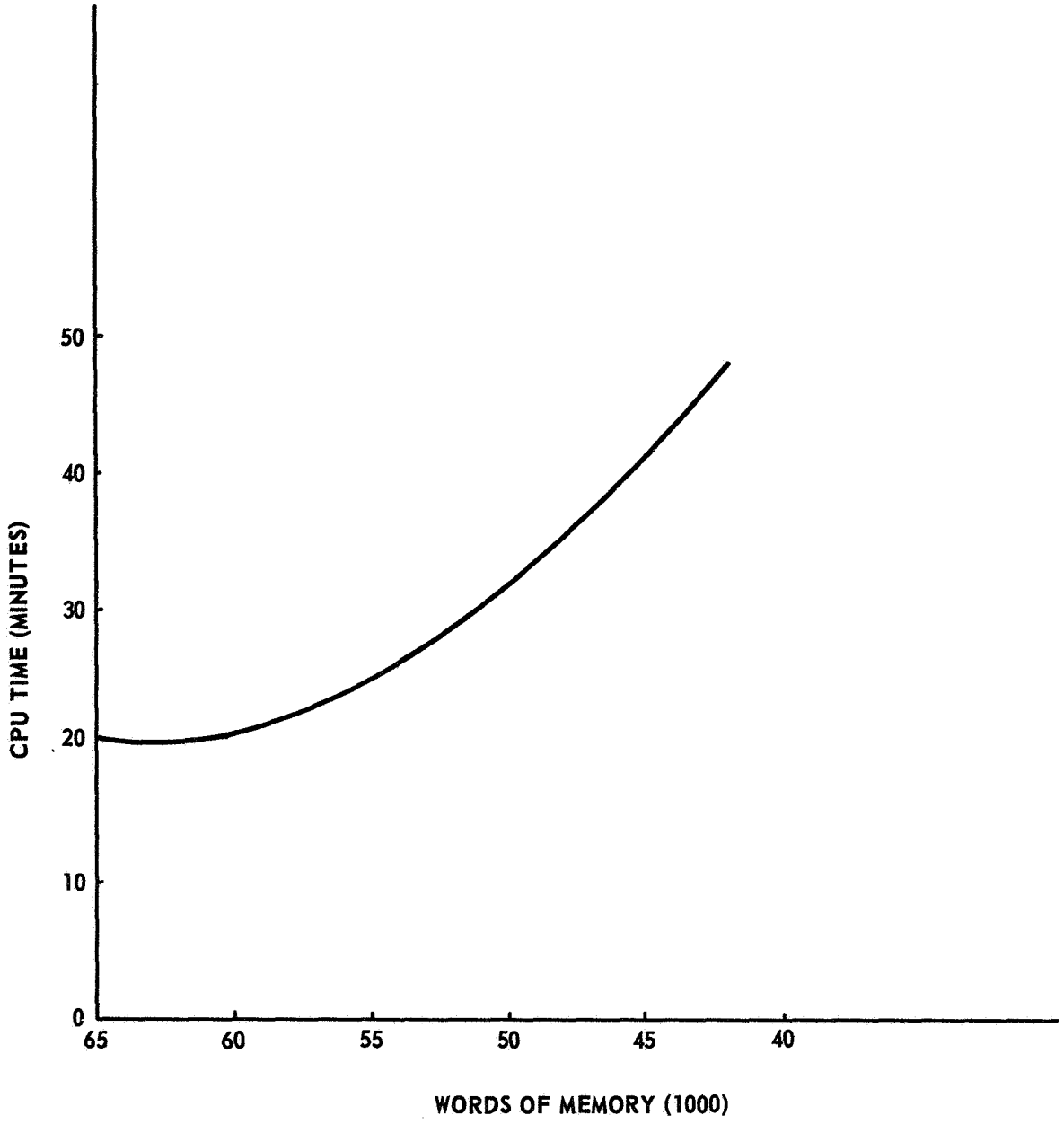

FIGURE 1.

CPU RUN TIME VERSUS AVAILABLE
CORE (TOTAL PROBLEM RUN TIME)

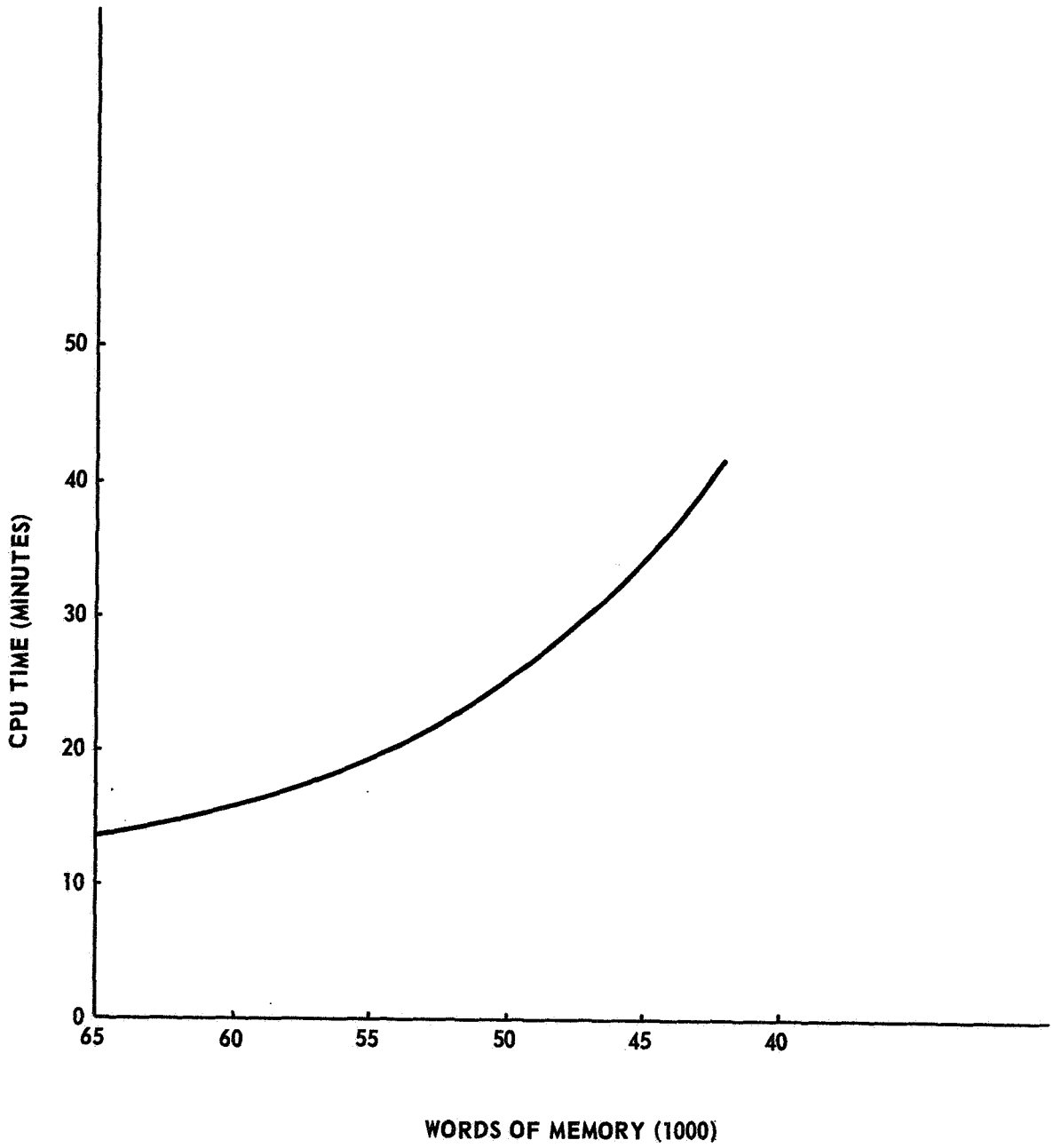

FIGURE 2.

CPU RUN TIME VERSUS AVAILABLE CORE
(DECOMPOSITION PLUS EIGENVALUE
EXTRACTION CPU TIME)

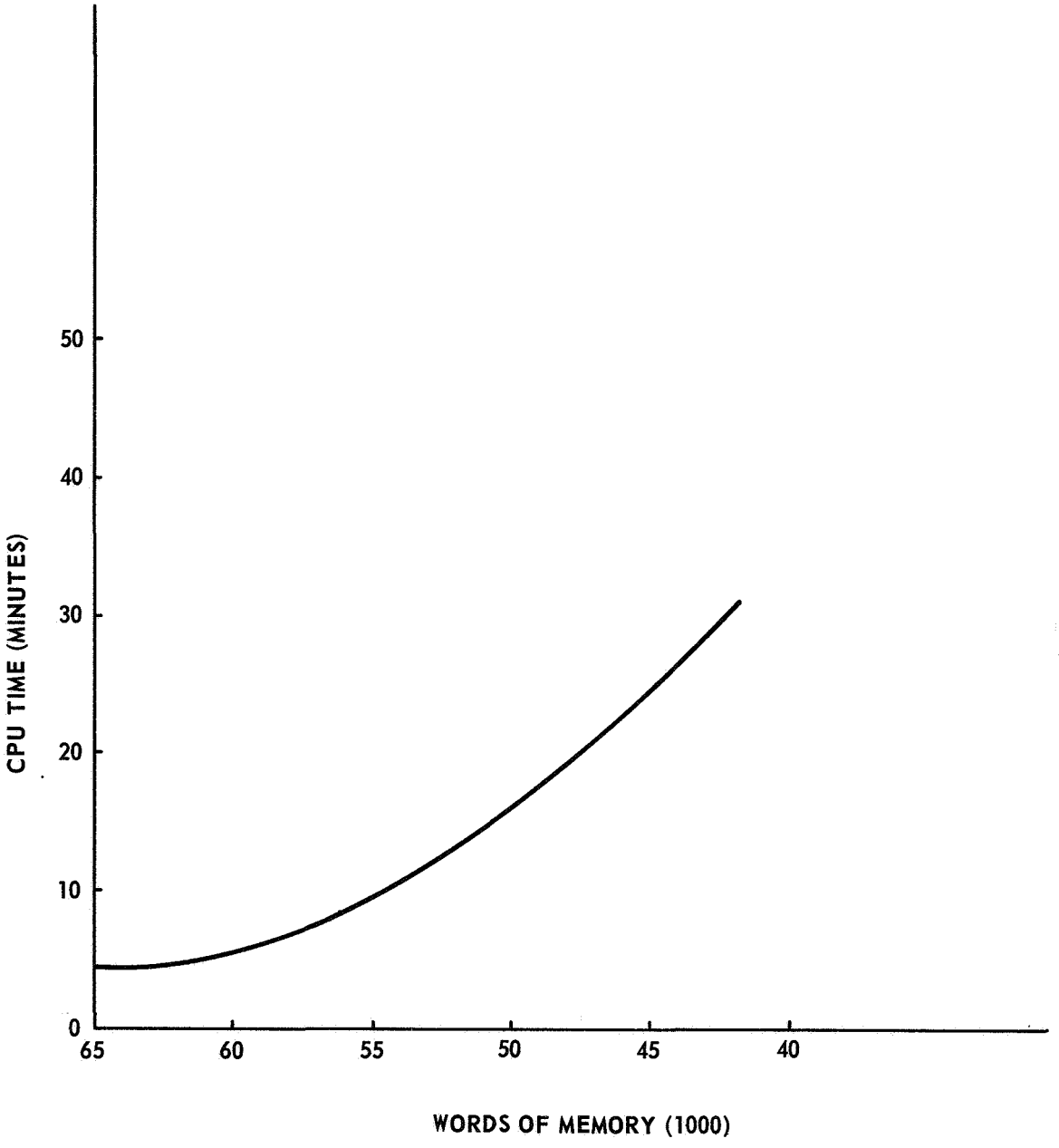

FIGURE 3.

CPU RUN TIME VERSUS AVAILABLE CORE

(DECOMPOSITION CPU TIME ONLY)