NASA Technical Memorandum 89108

DISTRIBUTED COMPUTER SYSTEM ENHANCES PRODUCTIVITY FOR SRB JOINT OPTIMIZATION

James L. Rogers, Jr., Katherine C. Young,
and Jean-Francois M. Barthelemy

February 1987

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

DISTRIBUTED COMPUTER SYSTEM ENHANCES PRODUCTIVITY FOR SRB JOINT OPTIMIZATION

James L. Rogers, Jr.*, Katherine C. Young*, and Jean-Francois M. Barthelemy#
NASA Langley Research Center
Hampton, Virginia 23665

Abstract

Initial calculations of a redesign of the solid rocket booster joint that failed during the shuttle tragedy showed that the design had a weight penalty associated with it. Optimization techniques were to be applied to determine if there was any way to reduce the weight while keeping the joint opening closed and limiting the stresses. To allow engineers to examine as many alternatives as possible, a system was developed consisting of existing software that coupled structural analysis with optimization which would execute on a network of computer workstations. To increase turnaround this system took advantage of the parallelism offered by the finite difference technique of computing gradients to allow several workstations to contribute to the solution of the problem simultaneously. The resulting system reduced the amount of time to complete one optimization cycle from two hours to one-half hour with a potential of reducing it to fifteen minutes. The current distributed system, which contains numerous extensions, requires one hour turnaround per optimization cycle. This would take four hours for the sequential system.

Introduction

After the shuttle tragedy in January 1986, NASA began evaluating changes in the existing design as well as new designs for the solid rocket booster (SRB) joint. NASA Langley's team of researchers designed a candidate which might be used if no method of fixing the current SRB joint proved feasible.[1] Initial calculations determined that this new design would have a significant weight penalty. Optimization techniques were therefore applied to this design to determine the optimal shape of the model while keeping the weight as low as possible, as well as keeping the joint opening closed and limiting the stresses of the structure.[2]

For engineers to analyze the numerous alternatives available through optimization, a software and hardware system had to be developed that would provide fast turnaround with the least amount of development time. In response to this, an existing software system combining structural analysis and optimization was modified to execute on a distributed network of workstations. This paper describes this system in terms of tools and how the system was distributed and tested. Also discussed are problems encountered when distributing the system.

Tools

The major tools used for this project were the hardware (DEC MicroVAX computer workstations), the software (PROSSS, Programming System for Structural Synthesis[3,4,5]), and the networks (DECnet and LaRCNET[6]). This section details each tool and the reasons for their choice.

Hardware

Work was begun on the project in late May, just when many summer professors and students were beginning to arrive. Historically, this influx of summer researchers had caused a noticeable delay in turnaround time of the central computer complex. Other alternatives included a DEC 11/785 minicomputer within the building or a recently installed network of DEC MicroVAX workstations. It was decided that using the workstations as the primary computer, with the minicomputer and central complex as backups, would be the most productive environment.

Originally, the workstations each had 2 million bytes (MB) of main memory with a 71MB hard disk executing with the VMS operating system. The finite element model of the new design for the SRB joint contained over 2200 degrees-of-freedom with approximately 530 elements. Although the workstation was a virtual memory computer, the time penalty in excessive paging between disk and main memory was determined to be too great for the required turnaround time. Therefore, an upgrade to 6MB of main memory was necessary to minimize paging when executing the analysis program for a model this size.

Another feature provided by the workstations that proved to be useful was multiple windows. A text window was used for editing files and word processing; while a graphics window was used for displaying plots. Since multiple windows can be opened at one time, it was possible to view more than a single task at a time. For example, on numerous occasions the engineer would view the output or graphics from one execution on one window while editing an input file for the next execution on another.

Software

PROSSS, a system of computer programs combining structural analysis and optimization,

*. Aerospace Technologist, Interdisciplinary Research Office, Structures Directorate
#. Aerospace Engineer, Interdisciplinary Research Office, Structures Directorate

was chosen as the software tool because the majority of the code in PROSSS was independent of the type of problem being solved. In addition, PROSSS had been converted to run on the workstations earlier in the year and had been verified with the standard test cases. There were five major components of PROSSS: (1) the structural analysis program; (2) the optimization program; (3) the front processor which converted data output from the optimizer into input for the structural analysis program; (4) the end processor which converted data output from the analysis program into input for the optimizer program; and (5) the input runstreams for the analysis program (figure 1).

For this project EAL, Engineering Analysis Language[7], was chosen for the structural analysis. An EAL input runstream already existed for the finite element model, and the EAL data libraries were useful for passing data between EAL and the optimizer. Precision was a concern when working with a 32 bit computer. EAL allowed for double precision in the processor for assembling the system elastic stiffness matrices and the processor for factoring this assembled system stiffness matrix. The factored system stiffness matrix, however, was truncated to single precision.

The optimizer, CONMIN[8], computed a new set of design variables based on the values of the objective function, constraints, and their gradients. In PROSSS, CONMIN was treated as a "black-box". A problem-independent driver program read data from EAL libraries and called the optimizer as well as the front and end processors. Finite differencing was selected from the options available in PROSSS for computing gradients. The design variables were perturbed one at a time by an input stepsize in a subroutine also called by the driver program. Several tests were made to determine that a step size of 20% was required to minimize the effect of roundoff errors due to the 32-bit precision.[2] After each perturbation an analysis was executed and the gradients of the constraints and objective functions saved. After all of the design variables had been perturbed and all gradients calculated, the data were passed to CONMIN for optimization. This optimization was in a loop with a subroutine for first-order approximation to the initial problem. Results from the optimization and approximation were stored in EAL data sets for the front processor.

The front processor was a very important piece of software developed specifically for this project to provide an automatic finite element model generating capability.[2] It was programmed as a set of functions that generate the coordinates of the joints based on changes in the design variables input as parameters to the equations. Output from the front processor consisted of EAL runstreams reflecting the changed model.

The end processor was also developed specifically for this project. The input to this processor consisted of the design variables, stresses, displacements, reactions, and objective function. Fourteen design constraints were calculated: ten maximum stresses at critical parts of the joint, one to keep the gap between the booster segments closed at the O-ring, one to maintain the integrity of the finite element mesh, one to allow sufficient room to insert the stud, and one to allow sufficient room to insert the nut. The constraints and their gradients were computed based on allowable values and stored into arrays which were used in constructing the first-order approximations.

There were two EAL runstreams used as input, a non-repeatable runstream and a repeatable runstream. The non-repeatable runstream was executed only one time to initialize the system with respect to initial model joint locations, the element connectivities, the loading conditions, the constraints, and data tables for use in the optimizer and front and end processors. The repeatable runstream was executed for each analysis (baseline and perturbed design variables) to compute the objective function, stresses, displacements, and reactions; and to store these data into the EAL data base.

Networks

The workstations and a DEC VAX 11/785 were connected by two networks, DECnet and LaRCNET (figure 2). Normally, the networks were seldom used while the system was executing on a single workstation, however they were critical for the distributed system. In several instances, having two networks available was a necessity because of problems occurring on one of them.

DECnet, the name for the DEC software and hardware products that allow different Digital operating systems to operate as network, connected the workstations through an Ethernet circuit. Although the maximum data transmission rate on an Ethernet circuit was 10 million bits per second (Mbps), DECnet's data throughput rate was much less.

LaRCNET was the local area network developed specifically for NASA's Langley Research Center to provide a centerwide capability for transferring data files among multi-vendor distributed computer systems. This packet-switched network was based on the 10Mbps Ethernet (intrabuilding) and Pronet fiberoptic token-passing ring (interbuilding) technologies. It allowed any connected device to access any other connected device at speeds generally limited by the end devices, normally up to 0.5Mbps effective throughput.

In theory the transmission rate for both DECnet and LaRCNET should have been the same because they were using the same hardware circuit. However, because of software differences in accessing the network, LaRCNET required less data transmission time and became the main network tool for this project.

Distributing the System

Initially, PROSSS executed sequentially on a single workstation. A single analysis of the finite element model required fifteen minutes for execution. Seven design variables (figure 3) were chosen for the optimization process.[2] Computing the gradients using finite differences required a baseline analysis and an analysis for each perturbed design variable before the optimization could begin. When the design

variables were perturbed sequentially, seven design variables (plus a baseline analysis) at fifteen minutes per analysis, would take two hours to complete a single optimization cycle . Much of the initial optimization work needed only a single cycle to give the engineers a feel for what would work, what would not work, and what was needed to optimize the model. However, five to seven cycles were needed to obtain refined optimization results, thus the sequential system would require from ten to fourteen hours to complete execution. A tight schedule to produce results required faster turnaround, so it was decided to distribute the system among several of the workstations.

The obvious advantage, and the whole purpose behind distributing the system, was to decrease turnaround time. With the four workstation system, time was decreased by approximately a factor of four. (Note: the current distributed system, which contains numerous extensions, now requires one hour turnaround per optimization cycle. This would take four hours per optimization cycle for the sequential system, and 20-28 hours for a complete optimization.) The reduction of the time required for a complete optimization of 5-7 cycles yielded results once or twice per day rather than overnight. By reducing the time for testing concepts in a single cycle, many different ideas were tested in a much shorter time.

Distribution Process

Distributing the system required minor modifications to PROSSS. One workstation was used as the controlling system. The front processor was removed from the optimization driver and run as a stand-alone program. The non-repeatable analysis and the front processor were run on the controlling workstation. The front processor was modified to loop through the design variables, perturbing them one at a time and creating a separate file for each design variable with the changed shape in the form of updated joint locations. These files, along with the EAL libraries from the non-repeatable analysis, were then distributed to separate workstations for analysis. The baseline analysis, the one with no change in the shape, was also executed on the controlling workstation. All workstations were sent a command file with checks to prevent them from executing until all of the required data were available. Once the analysis of the model with a perturbed shape completed execution, an EAL library file containing the objective function, stresses, and reactions was sent to the controlling workstation. The controlling workstation executed with a command file that prevented optimization from beginning until all of the required data had been returned from the other workstations. The end processor, called by the optimization driver program, was modified to read the objective function, stresses, and reactions from the various EAL libraries returned from the other workstations and compute the constraints and gradients before beginning the optimization. If the model was optimized, the system stopped, otherwise it looped back to the front processor to begin a new cycle with a new shape determined by the change in the design variables.

The majority of the changes involved creating new command files to execute the system. The command files used to execute the system were written in the VAX/VMS DCL command language. There were five of these command files, one interactive procedure and four batch procedures. The interactive procedure queried the user as to which network to use, the number of design variables, the number of optimization cycles, and the names of the workstations to use. In addition, it executed the non-repeatable analysis and the front processor to create the initial EAL libraries. These libraries, along with the batch procedures, were sent to each designated workstation in the network. Finally, the interactive procedure submitted the batch procedure files to the designated workstation batch queues for execution. Each batch procedure had a built-in loop that required all data to be available before starting the analysis. There were two methods for stopping the batch jobs. One was the receipt of a specified file from another workstation and the other was a built in time check to shut down if data were not received in a specified amount of time.

Testing the System

A simple finite element model of a beam composed of solid brick elements with three design variables was developed to test the sequential system. The first test of the distributed system was made using this model with four workstations. This phase of testing was completed when the distributed results compared exactly with the sequential results. Work with the sequential system on the SRB design continued while the distributed system was being tested. The software for the sequential system, along with the SRB model was frozen at a point to be used in testing the distributed system, and the results from the sequential system were saved for comparison.

Prior to distributed production with the SRB model with seven design variables, a decision had to be made as to how many workstations were to be used for the distributed system. Eight workstations (including the VAX 11/785) were available, which meant one analysis could run on each workstation and one optimization cycle would be completed in fifteen minutes instead of two hours. However, since only eight workstations were available and all eight would be needed to execute the system with seven design variables, any one workstation not being available would prevent the use of the distributed system. Therefore it was decided to include the option to use either four or eight workstations, but the plan was that the majority of the work would be done on four with the others to be available as backups. Figure 3 shows the breakdown of the analyses based on the perturbed design variables across the four workstations. The choice of relying on only four workstations at a time proved to be wise when, during the course of the project, several of the workstations were down at different times with hardware problems.

In the distributed system with four workstations, one workstation was used as a controller and sequentially executed the baseline analysis and one analysis with a

perturbed design variable. The other three workstations executed the analysis two times in sequence, each with a different perturbed design variable. This phase of testing was completed when the answers compared exactly with those of the saved results from the sequential system. Changes made to the sequential system after the "freeze" point were added to the distributed system to bring both to the same level. After identical results were obtained from both systems, the distributed system was put into production.

Problems Encountered When Distributing the System

Several problems had to be solved when distributing the system. First it was determined that the configuration of the workstations would allow only one EAL execution at a time. When two EAL jobs were competing for the CPU and disk space, there was virtually no throughput. Because several engineers not working on this project executed EAL on their workstations during the day, the competition for the CPU had to be resolved. The solution was to create a "lock" file whenever EAL was in use so that only one EAL job would run at one time on one workstation. An option was added to the command file to allow the user to choose the workstations for execution of the system. A manual check of each workstation was made for the existence of the EAL "lock" file before the distributed system began execution, and those workstations were avoided. Limiting EAL to one execution at a time on a workstation was another reason behind the choice of using four instead of eight workstations as well as the sequential execution of the two analyses per workstation.

Disk space had to managed carefully. The 71MB hard disk on each workstation filled up rapidly during the iterative optimization process, especially with the large EAL libraries and output files. The EAL runstreams were modified to remove all but the most recent data from the libraries during each pass. The command files were modified so that only the two most recent outputs from the analyses with perturbed design variables were retained. All baseline analyses output and all optimization output were retained. Important files were backed up on tape cartridges after the process completed.

The effect on the distributed system when one of the workstations failed had to be considered. System failure resulted from disk (both capacity and hardware), network, and analysis problems. Initially, each workstation continued checking for required files even though one or more of the workstations had stopped processing data. This led to enormous log files that would eventually fill the disk. To solve this problem, code was added to the command files to send an error file to each workstation if a problem occurred. The command files were also modified to check for this error file as well as the analysis data file. In addition, the command file was modified to check the time, and if no files were found after a certain amount of time then the system would stop on that workstation.

The networks also caused a problem. Initially all data were passed among workstations using LaRCNET because it was much faster than DECnet. However, because of the newness of LaRCNET some problems, such as transmission errors or the network being down, still existed. To solve this problem an option was added to allow the user to choose between LaRCNET and DECnet.

Despite these problems, the system served its purpose well and allowed the technical task to proceed to completion without major delays. This can be primarily attributed to the redundancy of available networks and workstations, and the flexibility of PROSSS.

Concluding Remarks

A computer software system coupling structural analysis and optimization has been successfully distributed over a network of workstations. Because the finite difference technique of computing optimization gradients was well suited for parallel execution, several workstations contributed to the solution of the problem simultaneously. By distributing the workload over four workstations instead of just one, the turnaround time for an optimization cycle improved from two hours to one-half hour. The system was applied to a finite element model of an alternate design of the faulty SRB joint. This application was to aid engineers in determining an optimal shape with minimum weight, while keeping the gap between the SRB joints closed and limiting the stress of the structure. Because of better turnaround time achieved with the distributed system, engineers were able to test more alternatives in a shorter time. The key features were the effective use of hardware redundancies (more workstations available than used in the system and two networks) and flexible software which permitted the optimization to proceed with minimal delay and decreased turnaround time.

References

1. "Conceptual Design of Solid Rocket Booster In-line Bolted Joint", NASA TM 89046, June 1986.

2. Barthelemy, J. F. M.; Chang, K.; and Rogers, J. L. Jr.: "Shuttle Solid Rocket Booster Bolted Field Joint Optimization", AIAA Paper 87-0702-CP, Presented at the AIAA/ASME/ASCE/AHS 28th Structures, Structural Dynamics and Materials Conference, Monterey, CA, April 6-8, 1987.

3. Sobieszczanski-Sobieski, J.; and Bhat, R. B.: "Adaptable Structural Synthesis Using Advanced Analysis and Optimization Coupled By a Computer Operating System." A Collection of Technical Papers on Structures - AIAA/ASME/ASCE/AHS 20th SDM Conference, April 1979, pp. 20-71, AIAA Paper No. 79-0723.

4. Rogers, J. L., Jr; Sobieszczanski-Sobieski, J.; and Bhat, R. B.: "An Implementation of the Programming Structural Synthesis System (PROSSS)", NASA TM 83180, December 1981.

5. Rogers, J. L., Jr.; Dovi, A. R.; and Riley, K. M.: "Distributing Structural Optimization Software Between a Mainframe and a

Minicomputer", Engineering Software II, Hobbs The Printers, Southampton, England, 1981, pp. 400-415.

6. Riddle, E. P.: "NASA LaRC Distributed Computing Network", VIM 44 Conference Proceedings, April 1986, pp. 2-44 through 2-49.

7. Whetstone, W. D.: "EISI-EAL: Engineering Analysis Language", Proceedings of the Second Conference on Computing in Civil Engineering, ASCE, 1980, pp. 276-285.

8. Vanderplaats, G. N.: "CONMIN - A FORTRAN Program for Constrained Function Minimization. User's Manual. NASA TM X-62282, 1973.

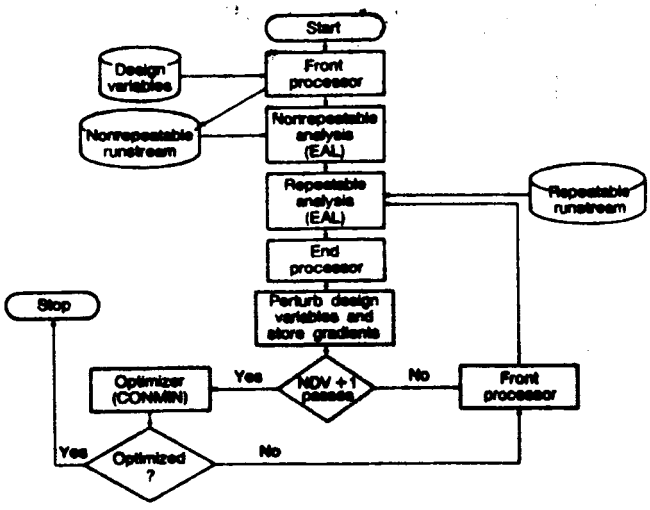

Figure 1. Flowchart of PROSSS with finite difference option.

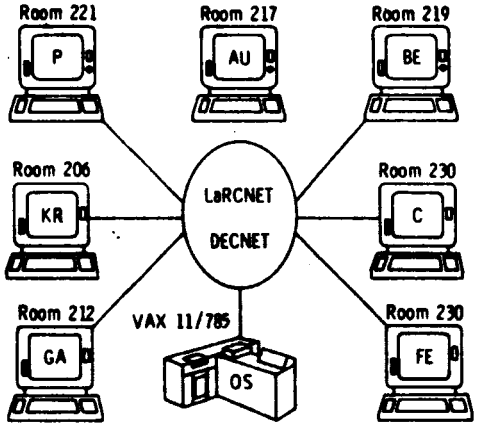

Figure 2. Workstation network.

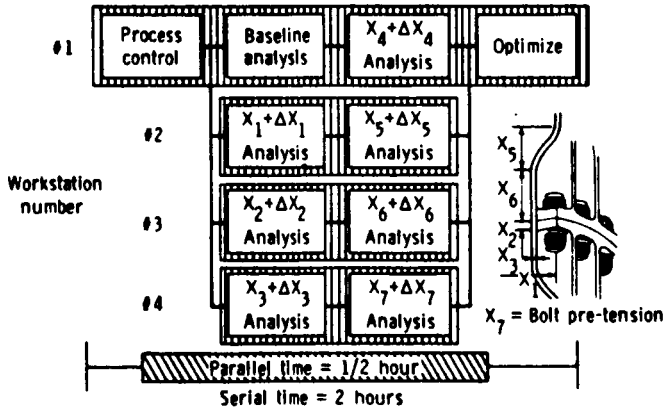

Figure 3. Distributed computing system for bolt model with seven design variables.

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TM-89108		

4. Title and Subtitle	5. Report Date
Distributed Computer System Enhances Productivity for SRB Joint Optimization	February 1987
	6. Performing Organization Code: 505-63-11-01

7. Author(s)	8. Performing Organization Report No.
James L. Rogers, Jr., Katherine C. Young, and Jean-Francois M. Barthelemy	
	10. Work Unit No.
9. Performing Organization Name and Address	
NASA Langley Research Center Hampton, VA 23665	11. Contract or Grant No.
	13. Type of Report and Period Covered: Technical Memorandum
12. Sponsoring Agency Name and Address	
National Aeronautics and Space Administration Washington, DC 20546	14. Sponsoring Agency Code

15. Supplementary Notes

To be presented at 28th AIAA/ASME/ASCE/AHS Structures, Structural Dynamics and Materials Conference, Monterey, CA, April 6-8, 1987

16. Abstract

Initial calculations of a redesign of the solid rocket booster joint that failed during the shuttle tragedy showed that the design had a weight penalty associated with it. Optimization techniques were to be applied to determine if there was any way to reduce the weight while keeping the joint opening closed and limiting the stresses. To allow engineers to examine as many alternatives as possible, a system was developed consisting of existing software that coupled structural analysis with optimization which would execute on a network of computer workstations. To increase turnaround, this system took advantage of the parallelism offered by the finite difference technique of computing gradients to allow several workstations to contribute to the solution of the problem simultaneously. The resulting system reduced the amount of time to complete one optimization cycle from two hours to one-half hour with a potential of reducing it to 15 minutes. The current distributed system, which contains numerous extensions, requires one hour turnaround per optimization cycle. This would take four hours for the sequential system.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Optimization, distributed computer system Workstations Structural analysis	Unclassified-Unlimited Subject category - 62

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	6	A02