Working Paper

OpTiX-II: A Software Environment for MCDM Based on Distributed and Parallel Computing

Harald Boden and Manfred Grauer

WP-94-54
August 1994

IIASA International Institute for Applied Systems Analysis □ A-2361 Laxenburg □ Austria
Telephone: +43 2236 71521 □ Telex: 079 137 iiasa a □ Telefax: +43 2236 71313

OpTiX-II: A Software Environment for MCDM Based on Distributed and Parallel Computing

Harald Boden and Manfred Grauer

WP-94-54
August 1994

International Institute for Applied Systems Analysis □ A-2361 Laxenburg □ Austria
Telephone: +43 2236 71521 □ Telex: 079 137 iiasa a □ Telefax: +43 2236 71313

Foreword

Although processing power of modern computers increases fastly, the need of fast analysis of ever more complex problems increases even faster. Luckily, a lot of progress is being made in enabling communication between computers or between processors. This progress makes it possible to use standard computers (personal computers or workstations) or standard processors in an integrated way to solve complex problems. The only disadvantage to this tendency is that programming of such an integrated system requires a lot of knowledge about the integrated system as well as about the particular problem. Therefore, it is important to develop tools which can help to bridge the gap. The software environment OpTiX-II is meant for bridging the gap for nonlinear optimization problems.

Abstract

The intention of the paper is to give an introduction to the OpTiX-II Software Environment, which supports the parallel and distributed solution of decision problems which can be repre sented as mathematical nonlinear programming tasks. First, a brief summary of nonsequential solution concepts for this class of decision problems on multiprocessor systems will be given. The focus of attention will be put on coarse-grained parallelization and its implementation on multi- computer clusters. The conceptual design objectives for the OpTiX-II Software Environment will be presented as well as the implementation on a workstation cluster, a transputer system and a multiprocessor workstation (shared memory). The OpTiX-II system supports the steps from the formulation of decision problems to their solution on networks of (parallel) computers. In order to demonstrate the use of OpTiX-II, the solution of a decision problem from the field of structural design is discussed and some numerical test results are supplied.

Contents

1 Introduction 1

2 Nonsequential solution approaches in nonlinear optimization 2

2.1 Parallelism inherent to the algorithm 2

2.2 Parallelism inherent to the problem structure 3

3 Conceptual design of OpTiX-II 4

4 Logical design and implementation of OpTiX-II 5

4.1 Problem formulation phase 6

4.2 Problem translation phase 8

4.3 Problem solution phase 8

4.4 Implementation issues 10

5 Some numerical test results 10

6 Summary and conclusion 11

7 References 12

8 Acknowledgement and further information 13

9 Appendices 14

OpTiX-II: A Software Environment for MCDM Based on Distributed and Parallel Computing

Harald Boden* and Manfred Grauer*

1 Introduction

The aim of this paper is to provide an introduction to the OpTiX-II Software Environment. This software system supports nonsequential solution approaches for so-called multidisciplinary decision problems (e.g. [1]). The underlying mathematical programming problem can be formulated as follows: find an n-dimensional vector $x = (x_1, ..., x_n)^T \in R^n$ that minimizes $f(x): R^n \rightarrow R$ subject to the constraints $g_i(x) = 0$, $g_j(x) \leq 0$ and the bounds $x_l \leq x \leq x_u$, where $g_k(x): R^n \rightarrow R$ (i = 1, ..., m_e; j = m_e+1 , ..., m; k = 1, ..., m).

The solution of decision problems based on nonlinear optimization arising from this multidisciplinary field is usually characterized by the fact that either the objective function or the constraints are set up with simulation software systems, based on finite element analysis. The repeated calculation of the simulation model leads to very time consuming solution procedures for the resulting optimization problems. To overcome this problem, the use of computer systems with multiprocessor architecture offers a possibility to reduce the solution time. But then a software environment for formulating, starting and controlling the concurrent tasks of a decision problem and for managing the communication effort during the solution procedure is required. In order to accord with these necessities, the OpTiX-II Software Environment has been designed to support the formulation of these decision problems and their solution on computer networks by applying coarse-grained parallel and distributed approaches. This allows the use of local area networks, which are widely spread in many institutions, as cost efficient alternatives to supercomputers in this specific problem area.

The following section will give a brief overview on the nonsequential solution approaches applicable in nonlinear optimization. They are, at least to some extent, supported by

* *University of Siegen, Faculty of Economics, Information and Decision Sciences Institute, Hölderlinstr. 3, D-57068 Siegen, Germany.*

OpTiX-II. In section 3 the conceptual design objectives of the software environment for distributed and parallel optimization are presented. Section 4 is intended to give a review on the logical design, implementation and use of OpTiX-II. Some numerical results obtained by solving test problems are given in section 5.

2 Nonsequential solution approaches in nonlinear optimization

The nonsequential solution approaches discussed here are divided into two classes. The first class makes use of parallelism that is inherent to a given algorithm. It can be represented by classical optimization algorithms using data parallelism, parallel extensions of hybrid optimization methods, and heuristic solution approaches applied in parallel. The second class utilizes parallelism inherent in the structure of the optimization problems. It can be represented by the use of parallelism inherent to decomposition methods. Because of space limitations, references are only meant to be indicative examples. They are not necessarily the definitive or seminal references of any given topic.

2.1 Parallelism inherent to the algorithm

Parallelism inherent to a given algorithm or a combination of algorithms is procured mainly to speed-up the solution process and, especially in the last few years, to extend the class of solvable problems with respect to the number of available processors in a computing system (scale-up).

The first group of algorithms includes classical nonlinear mathematical programming methods such as Quasi-Newton-Methods. The design concept of the algorithms as a whole is serial with, for example, direction search followed by line search. Thereby, the speed-up of the internal operations is based on the use of data parallelism (e.g. linear algebra computations [2]). Such algorithms are mostly modified for the use on supercomputer systems with massive parallelism or vector processing units. The algorithms can be included in the solution process by the OpTiX-II Software Environment (to be presented in sections 3 and 4) if they are accessible on dedicated supercomputers within a local area network, but they will not be the focus of attention in this paper. A review on the research in this field of parallel nonlinear optimization algorithms can be found in [3], [4], [5] and [6].

A different approach in parallel nonlinear optimization is to design a parallel algorithm that is scalable and adapts itself to the dimension of the optimization problem and to the number of processors available. For example in [7] a scalable multidirectional search method is introduced where the two parameters (problem dimension, number of processors) determine different

parallel algorithms with different characteristics. This concept is demonstrated in [8] for the class of evolutionary algorithms. For related work see also [4], [6] and [9].

Hybrid optimization methods as described in [10] and [11] represent a further approach which can be extended to parallel computations. They consist of a combination of a globally convergent algorithm with a locally superlinearly convergent algorithm and a test for switching between both methods. For convex optimization problems, the resulting method is globally and superlinearly convergent. This leads in itself to a type of coarse-grained or control parallelism, if both methods are applied in parallel. The necessary exchange of information after a certain number of iterations is asynchronous. A similar heuristic method, which makes use of the same type of parallelism, is to concurrently start a random search method, a method for nondifferential problems, and a local convergent method using derivative information. This helps to locate the global solution and to reduce computing time for the multidisciplinary optimization problems. The controlled exchange of information between these algorithms, operating in parallel, is thereby the basis for a more robust and, in some cases, even faster solution of the optimization problem (see [12]). Another nominee for this type of parallelization is the multi-start heuristic approach which is used for nonconvex optimization problems, since all optimization runs can be executed individually and in parallel.

2.2 Parallelism inherent to the problem structure

The application of decomposition methods for the solution of large scale decision problems leads to the most basic realization of coarse-grained parallelism by implementing the two iterative tasks: multiple subproblem solution (concurrent and asynchronous) and coordination of subproblems´ results. From this viewpoint, however, there is a broad range of alternatives in terms of the complexities referring to subproblems and coordination procedures. Even within this divide-and-conquer parallelism for the global solution strategy, it is possible to apply several forms of parallelism inherent to the solution methods, such as described in the previous section.

The OpTiX-II Software Environment, which will be presented next, is designed to support a distributed solution strategy by allocating dedicated algorithms and processors to subproblems. To our knowledge, such a software environment including algorithms for parallel and/or distributed solution of decision problems has not yet been presented.

3 Conceptual design of OpTiX-II

The predecessor of OpTiX-II ([13]) was developed for solving nonlinear optimization problems involving serial algorithms and one processor. In this section we will present the objectives for the conceptual design of the OpTiX-II Software Environment for the parallel and/or distributed solution of decision problems based on multidisciplinary optimization. The design objectives can be summarized as follows:

(I) The software environment should be an integrated system supporting the formulation of the decision problem as an optimization task, the design of a parallel/distributed solution strategy and the control of the solution process. It should support several hardware architectures being available for the solution. Thereby the efficiency of alternative nonsequential solution strategies for different problem classes and several types of multiprocessor architectures (see Fig. 1, Workstation cluster, Transputer system, Multiprocessor system) can be studied.

(II) The problem formulation language should support the analytical description of decision problems in form of an optimization task as well as the inclusion of external software written in the programming languages C or Fortran. This requirement evolves from the necessity to include existing algorithms or commercial software as part of a problem description. In the case of a problem formulation being given in analytical form, the software environment should provide first and second order symbolic derivatives.

(III) The problem formulation should be independent of the hardware that is subsequently being used for the problem solution. The transformation of the problem formulation and the inclusion of external routines written in C and Fortran should be provided automatically for all supported hardware architectures.

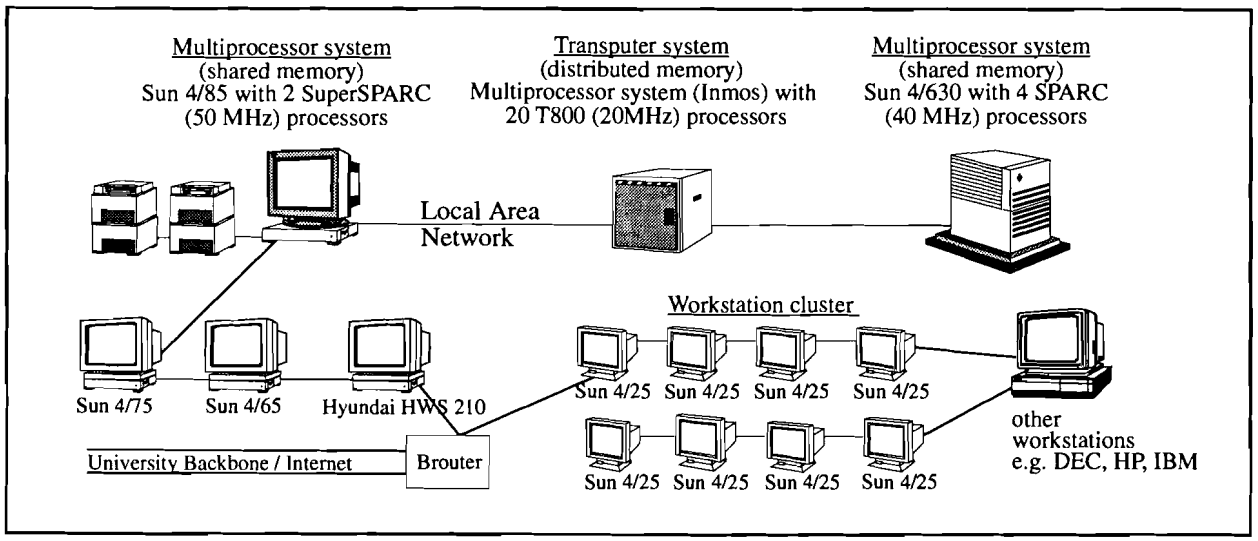

Fig. 1: Multiprocessor testbed for studying different nonsequential solution approaches in optimization (Unix-based workstation cluster and multiprocessor systems with shared memory, transputer system).

(IV) The user must be able to formulate an overall (parallel) optimization strategy, utilizing the nonsequential solution approaches described in section 2. Therefore it must be made possible to allocate the resulting individual optimization problems to computers within the available testbed and to choose algorithms with default parameters for the solution of the individual optimization problems from a proposed list of methods. This list should be extendable for further solution methods. The parameters of each algorithm must be adjustable to the problem under research.

(V) The user should not be confronted with synchronisation and communication issues that arise from the use of heterogeneous computer networks. These problems should be solved on the basis of standards for Workstations and Transputers and should be hidden from the user.

(VI) A graphical user interface which is based on standards set in the workstation market should be provided. Thus the user will be confronted only with a minimum of new syntax and semantics for the solution of the optimization problems.

In the following we will describe the logical design of the OpTiX-II Software Environment from a user's view and its implementation by demonstrating the solution of an optimization problem from the field of mechanical engineering.

4 Logical design and implementation of OpTiX-II

The objectives (I) to (IV) for the conceptual design are realized in OpTiX-II on the basis of dividing the entire solution process into three phases (see Fig. 2). During the problem formulation phase, the user defines the analytical formulation (including the use of external codes) of the current decision problem (see section 4.1). In the second phase this formulation is then translated into a machine code representation, which is suitable for parallel processing in heterogeneous networks (see section 4.2). The third phase (see section 4.3) is designated for the control of the optimization problem solution. Within the latter phase the user has to define an optimization strategy by choosing the optimization algorithms and by allocating computing resources from the network to the subsystems (individual optimization tasks). The optimization process can then be started by the user. Fig. 2 shows the flow of data and control within the OpTiX-II Software Environment. The realization of the design objective (V) and implementation issues are discussed in section 4.4. In the following the features and the application of OpTiX-II will be demonstrated in designing a gear reducer (see Appendix A). The nonlinear optimization problem consists of minimizing a weight function under the given constraints arising from structural mechanics. It is a very common test example in mechanical engineering and may be solved by the application of decomposition methods.

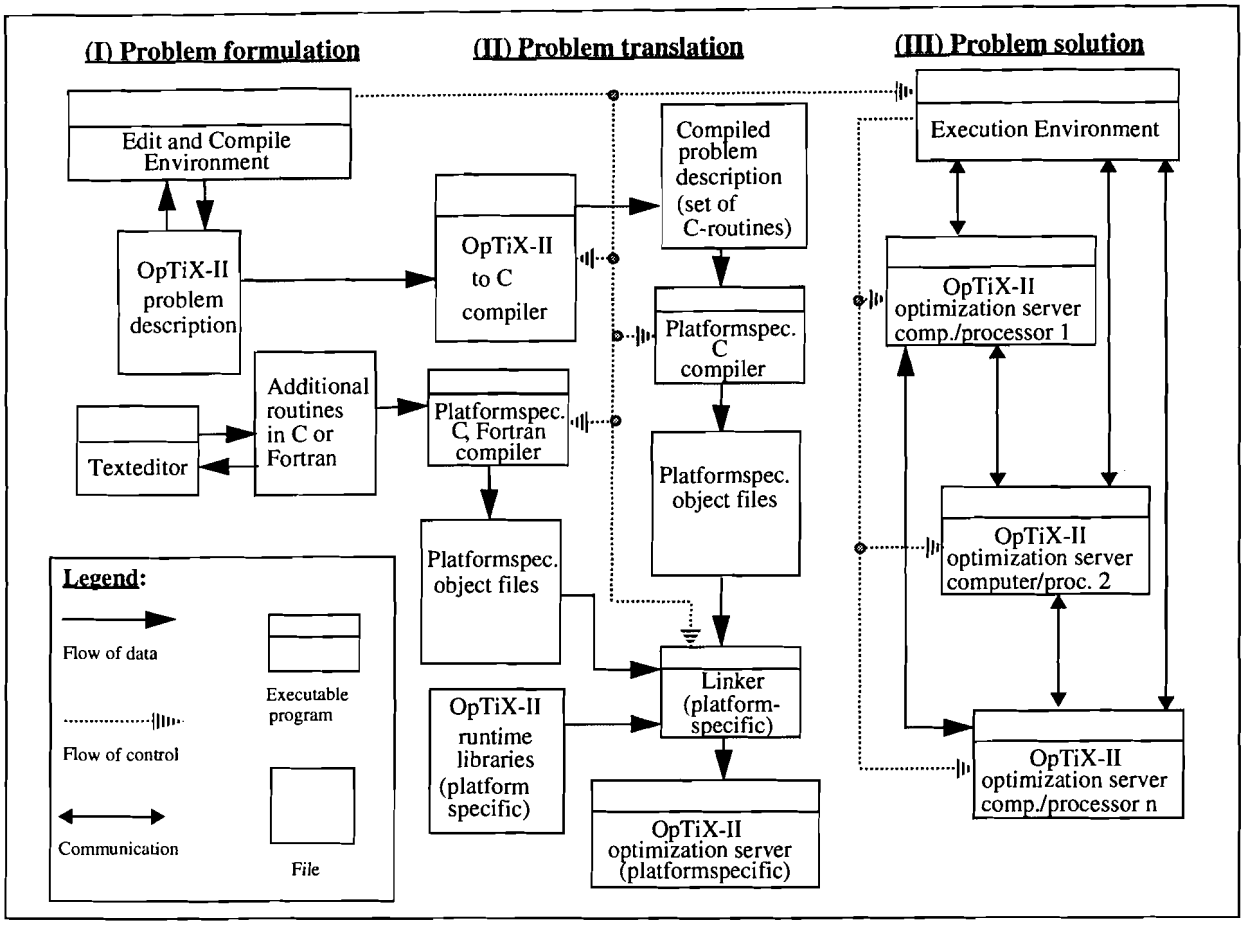

Fig. 2: Flow of data and control in the OpTiX-II software environment (logical design).

4.1 Problem formulation phase

In this phase the optimization problems are formulated, the generation of optimization servers for different platforms are controlled, and the execution environment is started. The problem description is entered into a graphically controlled problem editor using the OpTiX-II problem description language. The problem description consists of a declaration section, in which variables, constants and starting points are declared, and a problem definition section, in which the user defines one or more subsystems, each defining an individual optimization problem or a computational task. These subsystem tasks will be distributed to the resources of the computer infrastructure during the problem solution phase.

The statements made within the problem definition section correspond with the mathematical notation for nonlinear optimization problems. This is demonstrated by formulation of a gear reducer design problem in Fig. 3 (see also Appendix A). For more complex optimization problems, which cannot be described by simple algebraic expressions, OpTiX-II allows the inclusion of external functions written in the C or FORTRAN programming languages. Calls to these

functions may become a subexpression of the objective function or a constraint. Thereby it is also possible to define subsystem tasks that include commercially available simulation packages for the solution of complex mathematical models (see [14]).

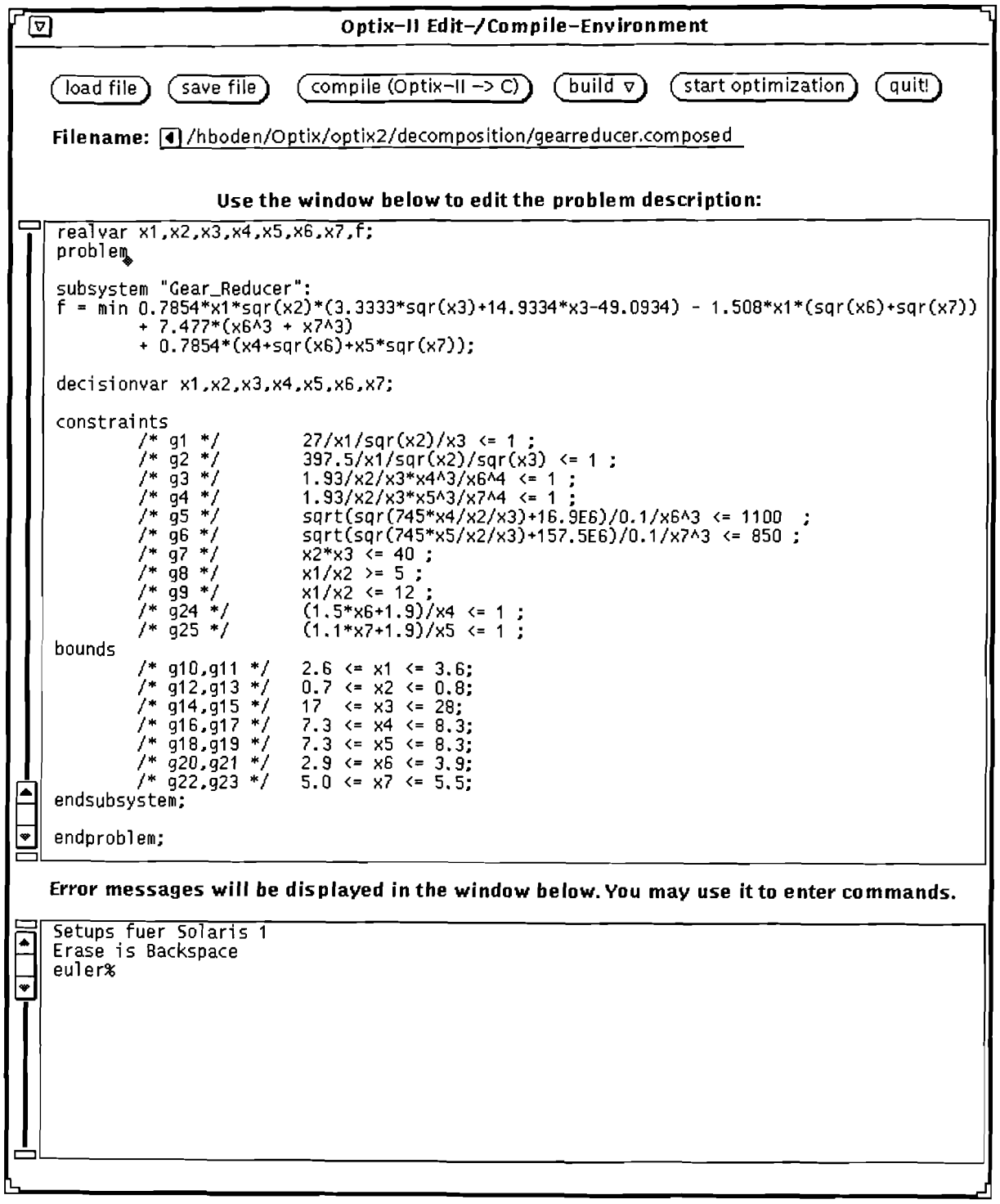

Fig. 3: OpTiX-II problem formulation for a gear reducer design optimization problem (see Appendix A)

4.2 Problem translation phase

In the translation phase the problem description is transformed into a machine code suitable for the execution in a heterogeneous network of (parallel) computers. It is started through the selection of menus in the Edit and Compile Environment (see Fig. 3). A compiler translates the problem description into a set of statements in the programming language C. Thereby, the compiler produces symbolic expressions for first and second order derivatives of the objective function and the constraints. In the case of external function calls, numeric gradient approximations are inserted into the generated code. The set of C functions is then compiled and linked into optimization servers (see Fig. 2) for different computing systems (e.g. Transputer systems under the Helios Operating System, Unix-based workstations with SPARC or MIPS processors). These optimization servers are called from within the Execution Environment. After this generation of platform specific optimization servers it is possible to utilize a heterogeneous computer network for parallel optimization. The OpTiX-II Software Environment, in its current stage of development, supports the following hardware/software architectures: (i) Unix workstations from various manufacturers, (ii) Unix-based Multiple Instruction stream Multiple Data stream (MIMD) type computers with shared memory (e.g. Sun SPARCstation 10, SPARCserver 600 and SPARCcenter 1000/2000 series) and (iii) Helios based Transputer systems (MIMD-type computer with distributed memory).

4.3 Problem solution phase

The OpTiX-II Execution Environment allows the formulation of parallel and/or distributed solution strategies as well as the recording of the problem solution processes and the display of the results. The user interacts with the control module of the Execution Environment (see Fig. 4). The nonsequential optimization strategy is formulated by writing a strategy script, which consists of a sequence of solution steps. In each step the user may solve several individual optimization subproblems in parallel. Thus, the user must choose algorithms for the solution of the individual optimizations (subsystems) and allocate computing resources within the network to the individual optimization tasks (subsystems in Fig. 4). The nonsequential solution approaches discussed in section 2 can be realized in this manner.

The gear reducer design optimization problem described in Fig. 3 and Appendix A can be decomposed into two subsystems, each consisting of a shaft and two bearings, coupled by the variables x_1, x_2 and x_3. For the solution of such an optimization problem within OpTiX-II, the user has to define three optimization problems in the problem definition phase, one for each subsystem and for a 2nd level coordination problem. During the problem solution phase three individual optimization tasks have to be processed iteratively: the two subproblems in parallel and

subsequently the coordination problem. For each individual optimization task, a host (computing facility) and an algorithm must be chosen. The subwindow on the bottom left (Fig. 4) gives the history of the solution and the subwindow on the bottom right displays the current best solution.

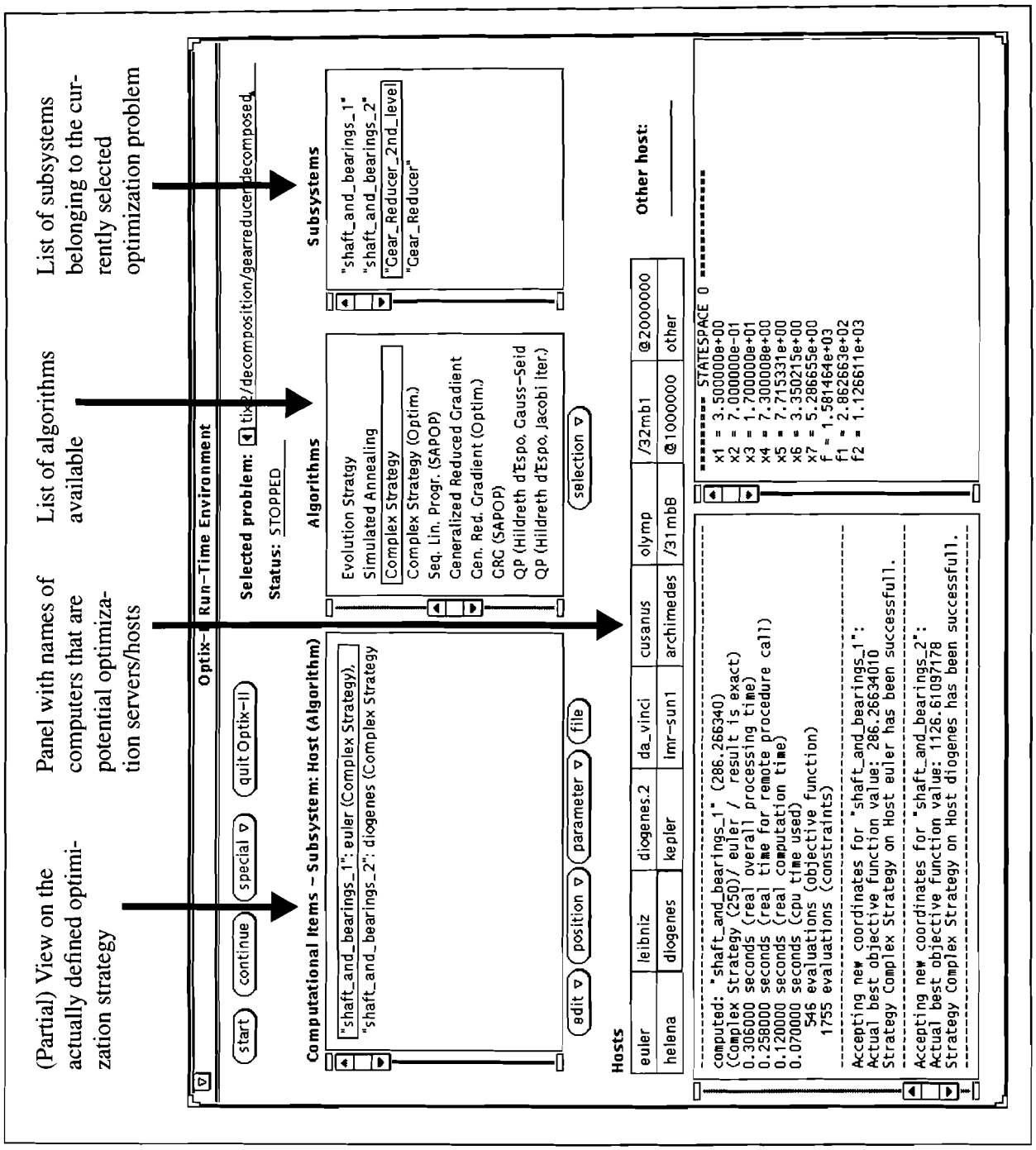

Fig. 4: The OpTiX-II Execution Environment (screen hardcopy), with a partial view on a strategy script for the solution of the decomposed gear reducer design optimization problem (see Appendix A)

4.4 Implementation issues

The main parts of the OpTiX-II Software Environment have been developed using the programming language C. The graphical user interface has been designed with the Sun Microsystems GUIDE program. This tool was selected because it supports different user interfaces (e.g. OPENLOOK and MOTIF). Software developed for Unix-based computers conforms to the X/Open portability guide and the POSIX standards. The transputer-cluster operates under the Helios operating system. It offers POSIX compatible libraries and supports the Network File System (NFS). Communication within the workstation network is based on Sun Microsystems's Remote Procedure Call concept and makes use of TCP/IP based services (e.g. remote copy, remote login, NFS). The decision for the tools mentioned above was made at the beginning of the project. Meanwhile various tools for parallel programming in heterogeneous computer networks are available (see [15],[16]). Many of these tools also make use of the Remote Procedure Call concept, which has become an industry-standard in workstation computing.

5 Some numerical test results

In the following some test results are provided to demonstrate the ability of OpTiX-II to serve as a testbed for parallel and distributed solution strategies and algorithms in decision problems based on nonlinear optimization on different multiprocessor systems. The optimization problems, with which OpTiX-II was tested, are the 100-dimensional Rosenbrock Problem (see Appendix B, RO-100) and the generalized Rastrigin Problem (see Appendix C, RA-30). Both problems were solved for comparative purposes with a sequential and two nonsequential approaches. Of all nonsequential solution approaches, a parallel extension to a hybrid method and, since both problems are separable, a decomposition approach (distributed computing) were applied. The three solution approaches are applied on three different architectures of multiprocessor systems (see Table below). The results for the multimodal function RA-30 are the average of 10 runs. Of course, the implications of these results (in the Table) are very restricted, but it is nevertheless not difficult to recognize that current transputer systems are not a competitive hardware for parallel optimization. On the other hand, workstation clusters and shared memory systems give promising results.

In addition to these results, OpTiX-II has been used to realize a parallel solution strategy for a structural optimization problem, thereby optimizing a complex mechanical structure (automotive wheel) in parallel on a network of workstations [17].

Type of Multi-processor-system \ Solution strategies and testproblems	Test-problem	Sequential approach*	Nonsequential approaches	
			Parallel hybrid**	Distributed computing***
Multiprocessor system (shared memory, see Sun 4/630 in Fig. 1)	RO-100	46.00	29.52	13.61
	RA-30	341.03	90.18	7.10
Workstation Cluster (see Sun 4/25 in Fig. 1)	RO-100	137.30	78.87	45.08
	RA-30	574.00	160.00	10.97
Transputersystem (see Fig. 1)	RO-100	829.00	531.16	157.63
	RA-30	3374.63	2552.76	68.87

The following algorithms and solutions strategies are used:
* RO-100: Quasi-Newton-Method; RA-30: Adaptive-Random-Search.
** RO-100: Quasi-Newton-Method, Steepest-Descent;
RA-30: Steepest-Descent, Nelder-Mead, Adaptive-Random-Search;
communication after 100 iterations.
*** RO-100: 3 subsystems and coordination by Quasi-Newton;
RA-30: 3 subsystems by Adaptive-Random-Search.

Table: Computation times (in seconds) for the solution of the Rastrigin and Rosenbrock test functions.

6 Summary and conclusion

The OpTiX-II Software Environment which supports the steps from the formulation of a decision problem based on nonlinear optimization problems to the solution on networks of parallel computers was presented. It enables the user to define the problem and a nonsequential solution strategy without necessarily taking into account the architecture of the hardware that will subsequently be used to solve this problem. Numerical tests have shown that workstation clusters and multiprocessor systems with shared memory give very promising results for the use of parallel and distributed solution strategies in the field of decision support.

The use of parallelism inherent to problems and algorithms, as indicated above, can be extended to other problem classes and corresponding algorithms (e.g. production scheduling and branch-and-bound methods).

7 References

[1] J.E. Dennis, R.M. Lewis, E.J. Cramer, P.M. Frank, G.R. Shubin, Multidisciplinary Optimization, paper at „Symposium on Parallel Optimization", Madison, 1993.

[2] P. Chaudhuri, Parallel Algorithms - Design and Analysis, Prentice Hall, 1992.

[3] F.A. Lootsma and K.M. Ragsdell, State-of-the-Art in Parallel Nonlinear Optimization, Parallel Computing 6 (1988) p. 133.

[4] S.A. Zenios, Parallel Numerical Optimization: Current Status and Annotated Bibliography, ORSA Journal of Computing 1 (1989) p. 20.

[5] T.L. Freeman and C. Phillips, Parallel Numerical Algorithms, Prentice Hall, 1992

[6] J. Eckstein, Large-Scale Parallel Computing, Optimization, and Operations Research: A Survey, ORSA CSTS Newsletter, Fall 1993, Vol. 14, No.2

[7] J.E. Dennis and V. Torczon, Direct Search Methods on Parallel Machines, in: SIAM J. Optimization, Vol. 1, No. 4, 1991, pp. 448-474.

[8] F. Hoffmeister, Scalable Parallelism by Evolutionary Algorithms, in: *Parallel Computing and Mathematical Optimization*, ed. M. Grauer and D.B. Pressmar (Springer-Verlag, Berlin, 1991) p. 175.

[9] D.P. Bertsekas and J. Tsitsiklis: Parallel and Distributed Computation. Prentice-Hall, 1988.

[10] O. Burdakov and C. Richter, Parallel Hybrid Optimization Methods, in: *Optimization, Parallel Processing and Applications,* ed. A. Kurzhanski, A., K. Neumann and D. Pallaschke (Springer-Verlag, Berlin, 1988).

[11] H. Kleinmichel, C. Richter and K. Schönefeld, On a Class of Hybrid Methods for Smooth Constrained Optimization, Journal of Optimization Theory and Applications, Vol. 73, No. 3, June 1992, p. 465.

[12] H. Boden, R. Gehne and M. Grauer, Parallel Nonlinear Optimization on a Multiprocessor System with Distributed Memory, in: *Parallel Computing and Mathematical Optimization*, ed. M. Grauer and D.B. Pressmar (Springer-Verlag, Berlin, 1991).

[13] M. Grauer, St. Albers and M. Frommberger, Concept and First Experiences with an Object-Oriented Interface for Mathematical Programming, S. 474-483, in: *Impacts of Recent Computer Advances on Operations Research*, ed. R. Sharda et al (North Holland, New York, 1989)

[14] H. Boden, OpTiX-II: Ein Softwaresystem zur parallelen Lösung von nichtlinearen Optimierungsproblemen - Benutzerhandbuch und Tutorial zu Version 2.5, Report of the Research Centre for Multidisciplinary Analysis and Applied Structural Optimization, University of Siegen, 1993. Download per anonymous ftp from magellan.fb5.uni-siegen.de is possible.

[15] I. Glendinning, S. Hellberg and P. Shallow, Tools for Parallel High Performance Systems - Comparative Evaluation, Technical Report SNARC 92-01, University of Southampton, Department of Electronics and Computer Science, 1992.

[16] H. Bal, Programming Distributed Systems (Prentice Hall / Silicon Press, 1990).

[17] H. Eschenauer and M. Weinert, Approximation Concepts for the Decomposition-Based Optimization of Complex Mechanical Structures on Parallel Computers, in: *Advances in Design Automation 1993, Volume 2*, ed. B.J. Gilmore, D.A. Hoeltzel, S. Azarm and H.A. Eschenauer (American Society of Mechanical Engineering, DE-Vol. 65-2, 1993).

[18] S. Azarm and W.C. Li, Optimality and Constrained Derivatives in Two-Level Design Optimization, Journal of Mechanical Design, Vol. 112, Dec. 1990.

[19] G. Rudolph, Global Optimization with Parallel Evolution Strategies, Diploma thesis, University of Dortmund, Department of Computer Science, 1990 (in German).

8 Acknowledgement and further information

Part of the work presented here was supported by a research grant by the Ministry of Science and Education in Düsseldorf (Nordrhein-Westfalen, Germany) under the topic "Decomposition methods and parallel computing".

A restricted (in user time) version of OpTiX-II is available to the public and can be downloaded from the Internet. Details can be obtained under the electronic mail address: optix@fb5.uni-siegen.de.

9 Appendices

Appendix A:

The nonlinear optimization problem considered here, consists in the design of a gear reducer with minimal weight under a set of given constraints (see [18] and Fig. A.1). The analytical formulation of the gear reducer design optimization problem is the following:

$$\text{Minimize } f(x) = 0{,}7854x_1x_2^2(3{,}3333x_3^2 + 14{,}9334x_3 - 43{,}0934) - 1{,}508x_1(x_6^2 + x_7^2) + 7{,}477(x_6^3 + x_7^3) + 0{,}7854(x_4x_6^2 + x_5x_7^2)$$

subject to:

$$g1: \ 27x_1^{-1}x_2^{-2}x_3^{-1} \le 1 \quad g2: \ 397{,}5x_1^{-1}x_2^{-2}x_3^{-2} \le 1 \quad g3: \ 1{,}93x_2^{-2}x_3^{-1}x_4^3x_6^{-4} \le 1 \quad g4: \ 1{,}93x_2^{-1}x_3^{-1}x_5^3x_7^4 \le 1$$

$$g5: \ A_1/B_1 \le 1100 \quad g6: \ A_2/B_2 \le 850 \quad B_1 = 0{,}1x_6^3 \quad B_2 = 0{,}1x_7^3$$

$$A_1 = \sqrt{\left[\left(\frac{745x_4}{x_2x_3}\right)^2 + 16{,}9\times10^6\right]} \qquad A_2 = \sqrt{\left[\left(\frac{745x_5}{x_2x_3}\right)^2 + 157{,}5\times10^6\right]}$$

$$g7: \ x_2x_3 \le 40 \quad g8,g9: \ 5 \le x_1/x_2 \le 12 \quad g10,g11: \ 2{,}6 \le x_1 \le 3{,}6 \quad g12,g13: \ 0{,}7 \le x_2 \le 0{,}8$$

$$g14,g15: \ 17 \le x_3 \le 28 \quad g16,g17: \ 7{,}3 \le x_4 \le 8{,}3 \quad g18,g19: \ 7{,}3 \le x_5 \le 8{,}3 \quad g20,g21: \ 2{,}9 \le x_6 \le 3{,}9$$

$$g22,g23: \ 5{,}0 \le x_7 \le 5{,}5 \quad g24: \ (1{,}56x_6 + 1{,}9)x_4^{-1} \le 1 \quad g25: \ (1{,}56x_7 + 1{,}9)x_5^{-1} \le 1$$

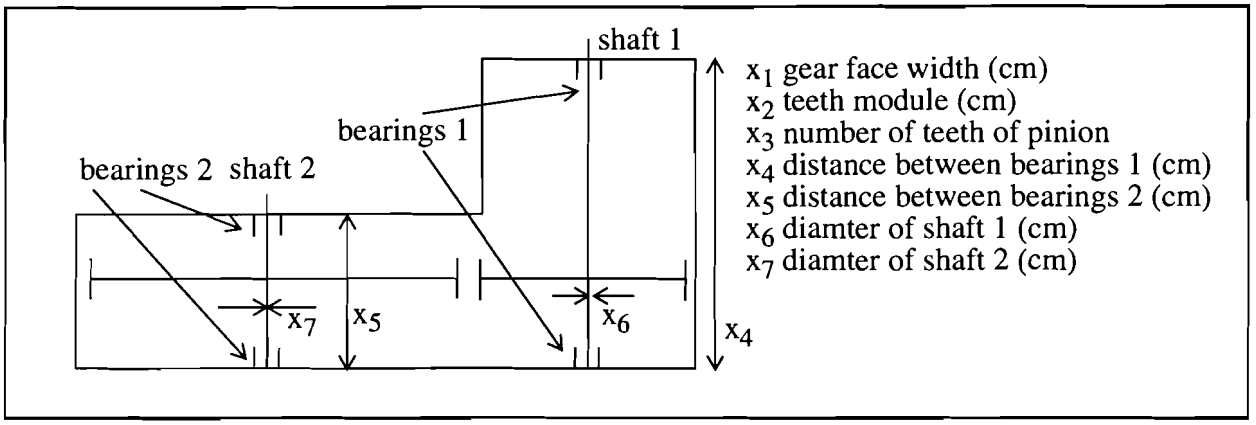

Fig. A.1: Gear Reducer under Consideration (see [18]).

The gear reducer design optimization problem can be decomposed into two subsystems, each consisting of a shaft and two bearings, coupled by the variables x_1, x_2 and x_3. For the solution of such an optimization problem within OpTiX-II, the user has to define three optimization problems in the problem definition phase, one for each subsystem and one for a 2nd level coordination problem. The problem formulations are given in Fig. A.2 to A.4.

```
subsystem "shaft_and_bearings_1":
    f1 = min -1.508*x1*sqr(x6) + 7.477*x6^3 + 0.7854*x4*sqr(x6);
decisionvar x4,x6;
constraints
    /* g3 */ 1.93/x2/x3*x4^3/x6^4 <= 1;
    /* g5 */ sqrt(sqr(745*x4/x2/x3)+16.9E6)/0.1/x6^3 <= 1100;
    /* g24 */ (1.5*x6+1.9)/x4 <= 1;
bounds
    /* g16,g17 */ 7.3 <= x4 <= 8.3;    /* g20,g21 */ 2.9 <= x6 <= 3.9;
endsubsystem;
```

Fig. A.2: OpTiX-II problem description for subsystem 1 of the decomposed gear reducer.

```
subsystem "shaft_and_bearings_2":
    f2 = min -1.508*x1*sqr(x7) + 7.477*x7^3 + 0.7854*x5*x7^2;
decisionvar x5,x7;
constraints
    /* g4 */ 1.93/x2/x3*x5^3/x7^4 <= 1;
    /* g6 */ sqrt(sqr(745*x5/x2/x3)+157.5E6)/0.1/x7^3 <= 850;
    /* g25 */ (1.1*x7+1.9)/x5 <= 1;
bounds
    /* g18,g19 */ 7.3 <= x5 <= 8.3;    /* g22,g23 */ 5.0 <= x7 <= 5.5;
endsubsystem;
```

Fig. A.3: OpTiX-II problem description for subsystem 2 of the decomposed gear reducer.

```
subsystem "coordination_problem":
f = min -1.508*x1*sqr(x6) + 7.477*x6^3 + 0.7854*x4*sqr(x6)
        -1.508*x1*sqr(x7) + 7.477*x7^3 + 0.7854*x5*x7^2
        + 0.7854*x1*sqr(x2)*(3.3333*sqr(x3)+14.9334*x3-43.0934);
decisionvar x1,x2,x3;
constraints
    /* g1 */ 27/x1/sqr(x2)/x3 <= 1;
    /* g2 */ 397.5/x1/sqr(x2)/sqr(x3) <= 1;
    /* g7 */ x2*x3 <= 40;
    /* g8 */ x1/x2 >= 5;
    /* g9 */ x1/x2 <= 12;
bounds
    /* g10,g11 */ 2.6 <= x1 <= 3.6;
    /* g12,g13 */ 0.7 <= x2 <= 0.8;
    /* g14,g15 */ 17 <= x3 <= 28;
endsubsystem;
```

Fig. A.4: OpTiX-II problem description for coordination problem.

Appendix B:

The 100-dimensional Rosenbrock Banana Problem is used as follows:

$$min\ f(x) = \sum_{i=1}^{99} [100 \times (x_{i+1} - x_i^2)^2 + (1 - x_i)^2]. \qquad \text{(RO-100)}$$

The starting point is $x_{start} = (1.2, -1.2, \ldots, 1.2, -1.2)^T$, and the bounds are $x_l = (-3.0, \ldots, -3.0)^T$ and $x_u = (3.0, \ldots, 3.0)^T$. The optimal solution is $x_{opt} = (1.0, \ldots, 1.0)^T$ with $f(x_{opt}) = 0$.

Appendix C:

The 30-dimensional generalized Rastrigin Problem is used as follows (see [19]):

$$min\ f(x) = nA + \sum_{i=1}^{n} (x_i^2 - A\cos(\omega x_i)) \text{ with n=30, A=50 and } \omega = 2\pi. \qquad \text{(RA-30)}$$

The starting point is randomly generated from within the bounds $x_l = (-5.0, \ldots, -5.0)^T$ and $x_u = (5.0, \ldots, 5.0)^T$. The optimal solution is $x_{opt} = (0.0, \ldots, 0.0)^T$ with $f(x_{opt}) = 0$.