IFAC/IFORS/IIASA/TIMS

The International Federation of Automatic Control
The International Federation of Operational Research Societies
The International Institute for Applied Systems Analysis
The Institute of Management Sciences

SUPPORT SYSTEMS FOR DECISION AND NEGOTIATION PROCESSES

Preprints of the IFAC/IFORS/IIASA/TIMS Workshop
Warsaw, Poland
June 24-26, 1992

Editors:

Roman Kulikowski
Zbigniew Nahorski
Jan W.Owsiński
Andrzej Straszak

Systems Research Institute
Polish Academy of Sciences
Warsaw, Poland

VOLUME 1:

Names of first authors: **A-K**

SYSTEMS RESEARCH INSTITUTE, POLISH ACADEMY OF SCIENCES

SUPPORT SYSTEMS FOR DECISION AND NEGOTIATION PROCESSES
Preprints, IFAC/IFORS/IIASA/TIMS Workshop, June 24-26, 1992, Warsaw, Poland

PARALLEL COMPUTATIONS FOR DECISION SUPPORT SYSTEMS

Ignacy Kaliszewski
02 093 Warszawa, ul. Newelska 6, Systems Research Institute, Polish Academy of Sciences, Poland

Abstract. The problem of effective implementations of computing intensive Decision Support Systems is discussed. Background computations and parallel computations are proposed as means to overcome the computing capacity bottleneck in computer supported interactive decision making. The transputer technology is envisaged as appropriate for multiprocessor implementations of Decision Support Systems.

Key words. Decision support, background computations, parallel computations, Pareto analysis, multiobjective optimization, multiprocessor networks.

1. Introduction

Most recent *Decision Support Systems* (DSS) , both general purpose or specialized, have two futures in common: first - they are *interactive,* second - they are *computer based* (cf eg Lewandowski,Wierzbicki(1989), Dror et al.(1991)). An interactive decision support system is a system designed and implemented to assist *interactive decision making*. An interactive decision making is a decision process split into several stages during which *Decision Maker* (DM) progressively expresses his preferences, analyses trial decisions and learns about the structure of the decision problem. One interaction (iteration) of such a process consists of the decision phase (DM is active, the computer is idle in the sense that it performs no bulky computations but only uses its output devices to present appropriate information in

textual and/or graphic form) and the computing phase (DM is inactive, the computer is active).

Although efficient implementations of DSS has been made possible by making intensive use of computing capacity of computers, computers alone can easily become bottlenecks as the size or complexity of decision problems grows. By obvious reasons in interactive decision making the duration of any computer phase must be kept within reasonable limits. It happens, however, that even for medium size decision problems the time consumed for determining one trial decision is significant and in such situations the practical usefulness of DSS can be questionable. Below we propose two remedies to improve this.

2.Background computations

The first remedy we propose is the idea of *background computations*.

It is a generic feature of all interactive DSS that during the decision phase for the most of the time the computer is idle. Even if DM activates the computer to store, sort, retrieve, compare previously derived decisions, or present related information in various forms, this usually consumes a negligible part of computer capacity. The remaining part of computer capacity can be used to process possible extensions or options of a decision support system, which, to keep the duration of individual computer phases within reasonable limits, have not been implemented. If the assumption that we can use the spare time of the computer at no cost or the cost is low (which is true for personal and dedicated computers) additional processing can be started even if DM may later show no interest in the results. Usually, it takes some time for DM to make a judgment about the current trial decision and set guidelines for the search for the next trial decision. During this time, which is otherwise lost, additional processing can be significantly advanced if not completed.

It is important that all the additional computations should not harass DM in his process of decision making (eg information

presented on the screen should not be affected) and therefore all the related computations must be done in "the background". The idea of background computations is well known in multitasking computer systems, where several tasks (processes) started by one or several users can be processed concurrently. Concurrency means that tasks are processed one by time but interchangeably, where processor after some time spent on processing a task suspends it and starts (or resumes) processing a subsequent task. Usually tasks are structured by some priority rules. It is possible then to interact with one task (which is said to be in the foreground) whereas the remaining tasks run in the background. Tasks are to be managed on the system level of a computer and this is hardly achievable by an ordinary user. It is therefore necessary to organize background computations from the level of a program. This calls for a capability of software to create submodules of a program, called *treads*, which can be processed concurrently. Mechanisms of this type are present in several algorithmic languages as Ada, Modula, and various "parallel" extensions of C, Fortran, and Pascal.

3.Parallel Computations

If it happens that the computer capacity is not sufficient to implement a decision support system or to implement fully its options or potential extensions, then the next possible step is to switch to parallel computations. Parallel computations can be effectuated on multiprocessor computers. In computers of this sort threads can be physically distributed among several processors. This, if done skillfully, results in a speed-up of computations with the theoretical bound on the speed-up equal to the number of processors used. Though some academic and even commercial multiprocessor computers are now available, a limited access to such installations and/or a high cost of their services make them hardly advisable in the decision making context. One must remember that most implementations of DSS have been done with desk-top minicomputers.

4. Multiprocessing on Networks of Transputers

Quite recently a technology has emerged which seems to be perfectly suited to the needs of decision making and solves, at least to some extent, the problem of ensuring appropriate computer capacity for a successful implementation of DSS. It features a family of microprocessors, called transputers (Relofs(1987), Whitby-Stevens,Hodgkins,1990)), each with four links, which can be easily connected via links with other transputers into a network. Moreover, the whole network can be connected via an idle link to any computer turning it into a multiprocessor computer of significant capacity.

A transputer (T800 version) is a 32-bit chip operating with (at most) 30MHz internal clock. It has 4 Kb on-chip memory and has an address space for an external memory up to 4 Gb. A key to the success of transputers is the speed of transmission via links: links are autonomous to transputer's CPU (can operate concurrently with the CPU) which results with a small communication overhead even if all four links are running at the same time. The effective speed of unidirectional transmission is 1.8 Mb/sec. There is no limit on the number of transputers working in a network. Transputer networks can be programmed with parallel extensions of C, Fortran, Pascal (cf eg Parallel C, User Guide(1991)) or assembler-like Occam.

A preliminary application of PC based transputer networks to multiobjective optimization problems (a formal model for many decision problems) has been already successfully completed (Kaliszewski,1990).

5. An Example - A Pilot DSS Implementing Quantitative Pareto Analysis on a Network of Transputers

Quantitative Pareto Analysis (Kaliszewski(1991) is a coherent methodology to provide DM with a variety of information about admissible decisions whenever multiobjective (vector) optimization problems are underlying formal models for decision making.

Quantitative Pareto Analysis offers in one methodological framework methods for:

- partitioning decisions into classes of efficient and nonefficient decisions,
- deriving numerical information on efficient decisions such as values of criteria, distances to a certain ideal (may be fictitious) decision, maximal and minimal values of separate criteria over the admissible set,
- a simple way to impose a certain hierarchical structure over the set of efficient decisions,
- a way for visualizing decision making processes by offering a method for fast approximations of sets of efficient outcomes (Pareto sets),
- bounds on trade-offs,
- approximate sensitivity analysis of efficient decisions with respect to perturbations of utility functions,
- approximate sensitivity analysis of efficient decisions with respect to perturbations of objective functions.

The first two items are standard elements implemented explicitly or implicitly in any DSS. All the remaining items of Quantitative Pareto Analysis result from interpretations of specific numerical characterizations of efficient decisions related to the notions of proper efficiency and substantial efficiency (Kaliszewski(1991)). The analysis is updated each time a new trial decision in the course of interactive decision making is derived.

Quantitative Pareto Analysis can be applied in its full extend in an interactive decision making method to enhance the quality of decisions. At any stage of an interactive decision making process DM is free to select from the whole variety of information provided by Quantitative Pareto Analysis the information he needs. The analysis (especially establishing sharp bounds on trade-offs) is rather demanding in computing capacity and computation time. Therefore, provided the computer used to build a DSS is parallel, all the respective items of the analysis are to be realized as

soon as a trial decision has been derived, even if results of the analysis for this particular decision will not be later used by DM.

A pilot DSS implementing outcomes of Quantitative Pareto Analysis is currently tested in the Mathematical Programming Department of the Systems Research Institute. At present only linear multiple criteria decision making problems can be approached by this methodology. The next step will be to extend the system to linear integer multiple criteria programming problems. A hardware platform for the system is a network of up to six transputers hosted by a PC computer.

6.References

Dror M., Shoval P., Yellin A., (1991), Multiobjective linear programming: another DSS. Decision Support Systems, 7, 221-232.

Kaliszewski I.(1990), Determination of maximal elements in finite sets on a network of transputers. Systems Research Institute Technical Report ZPM2/90, Warszawa.

Kaliszewski I., (1991) , Quantitative Pareto Analysis and the principle of background computations. In: Proceedings of IIASA Workshop on User Oriented Methodology and Techniques of Decision Support, Serock near Warsaw, (to appear).

Lewandowski A., Wierzbicki A.P. (eds), (1989), Aspiration Based Decision Support Systems, Theory, Software, Applications. Lecture Notes in Economics and Mathematical Systems, 331, Springer Verlag. Berlin.

Parallel C, User Guide, (1991). 3L Ltd.

Roelofs B., (1987), The transputer. A microprocessor designed for parallel processing. Micro Cornucopia, 38, 6-8.

Whitby-Stevens C. Hodgkins O. (1990), Transputers - past, present and future. IEEE Micro, 19-19, 76-82.