The Problem on Computer System Scaling Optimization

D. Usov
Institute of Information Technologies and Applied Mathematics, Omsk, Russia
usov@iitam.omsk.net.ru

V. Shaptsev
Institute of Information Technologies and Applied Mathematics, Omsk, Russia
vash@iitam.omsk.net.ru

Abstract
A problem of computer system modernisation and scaling is discussed. For these propose a mathematical problem of multicriterial optimisation had been set up. Approaches to program realisation of decision making support means in the progress of computer systems modernisation are described and proved.

1. Introduction
This paper is shown the importance of computer systems modernization problem and the optimal solution may to be of very benefit. The need of development of decision support system to solve this problem is proved. The mathematical optimisation problem had been set up as a problem of multicriterial optimisation with implicit goal functions and restrictions. The COM or similar technology usage necessity at program realisation of that decision support system is shown.

2. Computer Systems Modernization and Scaling
Design modularity is an essential property of modern computer systems hardware. Computer component parts are furnished with standard interface connectors easily allowing connecting or disconnecting any electronic module. So, a staff member or even a user equipped with just a screwdriver is quite able to replace a faulty or out-dated module, to set in or out an optional unit, and in the long run to assemble a functional computer with possibly unique configuration from different manufacturers' components.

In the PC world the component replacement takes place both due to the latter getting out of date and to fit higher consumers' demands.

The software manufacturers very quickly respond to the expansive commercial emergence of computers, which poses new potentials, and make programs employing these potentials. In outdated computers such programs are either unable to operate or they function ineffectively. As a consequence, the users can't do other but get a computer compiling with modern requirements. In this situation the software manufacturers take great interest in market promotion of their commercial innovations, advertizing usually greatly overestimating their actual potentials. Both production expenses and current engineering possibilities retard the growth of computer component performances. Therewith the performances of different components grow with different rate. Speaking about PCs based on Intel ideology, the fastest updating involves processors, read/write memory and HDD storage units, while video monitors remain the most slowly enhanced parts. Besides, the perfection of various kinds of components goes on with inconstant rate, which is hard to predict. From the above discussion it might be assumed, that if one purchases a computer suiting modern requirements, then in the nearest future (in a year, on average) some of it's component parts will prove to be outdated and will have to be replaced.

Since a lot of other component parts get out of date not so fast, the substitution of outdated units is financially much more advantageous then buying a new system. But the following questions arise here.

• How many parts should be replaced? Should it be only the processor or motherboard as well?
• What component parts should be used as substitutes? Should those cheap ones but doomed to near replacement be used, or should more expensive and high-quality parts be bought?
• From what dealer to buy? Should he be the one who sells cheaper, or the one with higher price but providing safe guarantee, or the one who gives considerable discounts?
• The parts of what manufacturer to choose, if they offer similar parameters?
• What operational performances should a computer poses for a proper functioning of certain software?
• How to keep within the budget?
The situation gets more complicated if an institution has several computers, which are supposed to perform different functions, and, accordingly, should have different properties. However, the problem here can be partly solved by means of component parts replacement from a more powerful computer into a less powerful one, thus upgrading the latter.

It should be noted, that similar questions arise while purchasing a new system and not only of PCs, but also of servers and commercial computers as well, so we can speak about a complex system synthesis problem.

A system is to be synthesized, meeting certain restrictions (memory size, application requirements compatibility, budget limits) and maximizing several features (speed of operation and response, monitor performance). This is a problem of decision-making support, so a special technique as well as a software have to be developed to assist a person responsible in finding the best possible way of outdated component parts replacement or of constructing a new system from unified modules.

### 3 Setting up the problem

The updating of an existing computer system is reduced to the problem of a new system construction with price parameters of already available components being different. That is, if we decide not to buy a new video monitor keeping the one we have, then we accept the operational parameters of that monitor as similar to that of a new one (wear neglected) but the price will be lower as that of a second-hand item.

Restrictions: a set of hardware component number, function parameter number, and a set of system functions F. A mapping is given:

\[ P: (f, V) \rightarrow \mathbb{R}^n_f, \]

where \( f \in F, V \subseteq X \) vector \( \{ y_0, y_1, ..., y_n_f \} \). The essence of the mapping is \( P \) - the set of parameters, determining the efficiency of fulfillment by the component parts range \( V \) the functions of the system \( f \), where \( y_0 \) equals to 0 if the range \( V \) fails to fulfil the function \( f \), and equals to 1, if it does.

Let the computer system consist of \( k \) computers \( V_j \subset X \), where \( j = 1, k \). Then a polyoptimisation problem in \( q \) criteria is to be solved

\[ P_n (f_i, V_j) \rightarrow \max, \]

where \( i = 1, q \) corresponds to criterion number, \( t = 1, n_t \) - function parameter number, \( j = 1, k \) - number of the system hardware component, and \( m \) restrictions:

\[ P_a (f_i, V_j) \geq a_s, \]

where \( s = 1, m \) - restriction number, \( r = 1, n_t \) - parameter number, \( j = 1, k \) - hardware component number.

Some peculiarities of the polyoptimisation problem under consideration should be underlined. It is essential that the relation \( P \) and hence the restrictions and criteria of the optimisation problem have been set algorithmically (in the form of a program code), but not in analytical form. The reasons of the above are the following.

- While calculating the parameters of computer system functions fulfillment, the data from tables, networks, temporary buffers are to be used. Their explicit analytical form would be too cumbersome to apply.
- Some parameters are to be calculated by means of complex algorithms or simulation techniques.
- The fulfillment of one kind of system functions is calculated recurrently via the other ones, as an explicit setting of the functions by means of component parts range would decelerate the calculations.
- Some parameters can be presented explicitly. However, according to the character of the problem these are going to be mostly nonlinear and discontinuous functions, which doesn't permit to apply the methods of solving multicriteria optimization problems with continuous input data [1,2].

The set of admissible solutions is finite that is we can speak about a discrete optimization task. However, the effective techniques of discrete optimization are not applicable here because of the discontinuous character of the goal function and restrictions. The method of exhaustive variants search can't be applied due to their large number. According to the offers of an average company dealing with computer component parts, about \( 10^5 \) of different functional computers can be assembled.

Since there are several optimality criteria, the goal function is assumed unassigned. For the reduction of a polyoptimisation problem to the one having a single criterion, the approaches discussed in [1, chap.3] have been proposed.

### 4 Solution Approaches

The paper [1, chap.8] suggests that system variants be constructed by means of morphological analysis method. The latter allows similar problems to be solved in assumption, that each system function is performed only by one element. In the given formulation we assume, that the system will perform definite function with a certain range of components available. For example, to perform the function of a read/write memory, the system must comprise definite type read/write memory modules and a motherboard with connectors to plug in this memory type. Apart from that, the paper [1] proposes searching only of Pareto-optimal alternatives. In our problem, however, the choice of the above alternatives is hindered because of the input data discontinuity. If it is taken into account, that the price, reliability and speed will be among the most possible optimization criteria, then the following assumption can be made: the variants of the system with all the mentioned
parameters definitely lower than that of the others are not numerous. Therefore, if one searches through not only Pareto-optimal, but also all admissible alternatives, the search space will grow insignificantly.

Thus, to solve the problem in the formulation given, it is advisable to apply the morphological analysis method, adapted for the functions, fulfilled by the range of component parts.

5 Program formulation matters

We have considered the solution of a polyoptimization problem with the following input data: a range of component parts, their parameters and optimization criteria. While formulating a program, the questions of these data acquisition, input and storage should be considered. Hardware manufacturers keep widening the variety of their products and improving the component parts performance. Meanwhile, such parameters as price and delivery deadlines may vary every week. Therefore the program to develop should be accessible and capable of using operational data. It should allow the manual or automatic input data variation. Operational data are usually stored in programs in the form of relational tables, e.g., the component parts parameters can be stored in the above way. But how can we store and supplement the parameters and criteria with values calculated by using a complex algorithm, or even by the way of simulation technique [3]? This means that additional program modules should be plugged in to the application in the process of the latter's operation. One of the possible ways out might be in the employment of object-oriented data bases (OODB) [4,5].

But this trend is currently facing some fundamental obstacles concerning object allocation over network nodes. Besides, the available versions of OODBs have not yet been developed to the commercial level and are difficult to use. The alternative decision is to apply Component Object Model (COM) for program code storage [6]. COM is a normalized standard, it has been realized at several program platforms, it is being supported by a lot of programming languages and has sufficiently high-speed performance. The disadvantage of the COM when compared to the OOBs is the absence of the object state integrity features on a hard storage medium. But long-term object storage is not required for optimization problems: objects are created in the memory, used in optimization cycle and then removed. When keeping all the data in the memory is not rational, they can be recorded into a relational data base with correspondent references left in the memory.

So, we consider Component Object Model (COM) suitable for a dynamic program modules linking in our problem.

6 User Interface

User interface features are generally considered as technical details. Although commercial success of a lot of program products and, consequently, a real benefit for a user have been gained thanks to well reasoned interface design. Needless to say that an up-to-date interface should be a graphic one. But it is of special importance to treat a user not as an expert in the field of optimization and decision-making support theory, but rather as a person skilled in computer operation and having some idea about the specification of a computer system entrusted to him. So, a user can't be required to describe a problem as a text file in an algorithmic language. Instead, via dialog blocks he is to be asked several questions, which he can either answer or not depending on his competence level. The questions should be formulated in a language familiar to the user, e.g.:

- What applications are supposed to be employed.
- The possibility of videodisks viewing.
- Maximal cost of the system.
- Whether you prefer a large monitor, or a small one but with multimedia.

7 Additional Possibilities

Apart from the nucleus of the program, which provides the solution of the optimization problem, some supplementary environment is necessary for the final success of the product. Above all, there must be a means of optimization process control. After the initial data input is over, the user submits an optimization process (the latter can be interrupted at any moment by pressing a certain button). For the sake of visual proof and in order to develop user's intuition, the optimum search process must be visualized. Moreover, not only the problem solution animation should be demonstrated, but also current configuration movement along the keyline of admissible solutions should be depicted.

Having estimated a system configuration suggested by the computer, the user might change the initial conditions and data of the goal function, and then perform the optimization once again. The procedure is repeated several times until all the user's demands are implemented.

To scroll computer system configurations, as well as for their editing and storing, a special editor should be developed. It will also be used to input information concerning the current computer configurations, whilst they are being modernized. Besides, the editor should be furnished with the means of report generation: configuration pattern, specification, and exchange path.

A system user doesn't have to input parameters data of the system component parts. Instead, a data base should be maintained in a specialized center, being then broadcasted over the network or by means of data media to end users. Technical parameters of component parts should be input once only for all the world's users. As to purchase terms, they depend on regional dealers.

In conclusion it is worth mentioning that at present there is no computer components parameters standard description in electronic form. These parameters can be found currently in manufacturer's specifications, in test results by independent experts, in electronic publications of widely different forms. Presumably, the development of decision -making automatic means will impel hardware manufacturers to employ the
standard and program processable method of required information representation.

References


