Organizing Local Behavior in Complex Systems

VICTOR VARSHAVSKY
Neural Networks Technologies Ltd.
3, Hashmonaim st. Bnei-Brak, 51264 Israel,
E-mail: victor@nnt-group.com

ILYA LEVIN
Tel Aviv University
School of Education,
Ramat-Aviv 69978, Israel,
E-mail: ilia1@post.tau.ac.i

"Complex systems are systems,
the detailed description of which is:
(a) impossible and
(b) avoidable".

Lev Rozonoer

ABSTRACT

This paper deals with exploring the decentralized control by using a simulation based learning environment from the one hand and by utilizing theoretical results belonging to the *Collective behavior of automata* from the other hand. We focus on one of the important issues of the decentralized control: forming system of laws and rules that provides achieving goals of the system. These laws can be considered as a global control within a decentralized system.

This issue had not been investigated in an educational context. Usually system simulation based educational environments serve just a main idea of the decentralized control – functioning without any global (centralized) algorithm. Meanwhile, real decentralized systems work in a different way. Most of them have some kind of the centralized control being implemented in a form rules or laws relating to the whole environment and providing the tolerance of global and local goals of the system. Introducing of such kind of control into the educational system increase both a plurality of real world oriented learning activities and a theoretical deepness of system phenomena to be discussed on the lesson.

KEY WORDS: distributed artificial intelligence, education, massively parallel computing.

1. INTRODUCTION

The impossibility or, at least, low efficiency and low reliability of direct centralized control among supercomplex systems is a fact that does not require a special proof. However, when the system has general goals and general criteria of the quality of functioning, it is inevitable that a certain central organ intervenes in the behavior of the system components. It can be done in several ways. Social, economic and industrial systems give numerous examples of a control organization which introduce laws and rules of behavior for creating an environment where a motivation to achieve local goals (being limited by the laws and rules) provides the movement toward the general goal. In other words, the aim of the centralized control is forming a system of laws and rules for local (individual) behavior that provide the tolerance of local and global goals, leveling contradictions between them. The simplest example is the control of a national economy via a system of tax privileges.

The explosive rise in complexity of technological systems, such as information and computing ones, makes their structure and behavior similar to social and economic systems in which they are integrating and with which they closely interact. It is natural to assume that principles and laws of functioning should be quite similar to complex systems of different nature. Users, messages, mobile agents, etc. after being initiated, begin to "live" in the system, having a sort of "free will".

Although general evolution laws are equally displayed in the development of both natural and artificial (technological) systems the latter are controlled by the Creator to a greater extent, at least concerning properties of "individuals". So, the aim of the control is not only to create general laws and rules of behavior but also to impart the necessary behavioral properties to the system components, along with the systems of local preferences.

Studying general laws of system's behavior was started in Russia by M. L. Tzetlin and his disciples in 60's in the framework of *Models of Automata Collective Behavior* [2-6]. This direction yielded a lot of useful and interesting results. Unfortunately, they are almost forgotten today. We will follow the basic methodology of this approach, which is personifying the system components, describing the system of local preferences and creating a legend, i.e. a plausible economic analogue.

It has to be clearly realized that systems of high complexity cannot have precisely formulated goals or optimal values of criteria. There are at least three reasons of this phenomenon. The first reason is that the growth of the system's complexity usually associated with the growth of the fuzziness of its goals and criteria. The second reason is that the growth of the system's complexity, usually leads to permanent changes of numerous internal and external factors. And, finally, the third reason is that complex systems usually have multi-criteria behaviors and the optimum is determined either by Poreto's set or by a weighted set of particular criteria provided by "social concord". Therefore, control should provide general permanent tendency to a better functioning quality and a high reactivity to changeable conditions. In other words, our subject should be not optimum but appropriate behavior, though we will use the concept of optimum in our further discussion.

We consider the following two simple examples: a) resource allocation in a massively parallel computing system and b) dispatching tasks in a heterogeneous computing system.

2. BODY OF PAPER

2.1 Resource allocation in a massively parallel computing system

Task [8]: A computing system consists of N processors and N memory blocks placed in knots of an arbitrary graph. The graph arcs determine the possibility of interaction between the processors, in particular, using neighbor's memory. Performance of the processors depends on the capacity of memory (their own memory and a memory of their neighbors) they use; the dependence is described by a certain estimation function $F_j(m_j)$ of the memory m_j available to the certain processor j. Let $M = (m_1, \ldots, m_j, \ldots m_N)$ be a vector of values of the processors' memory capacity. The goal of memory allocation is: $\max_M \sum_N F_j(m_j)$. We have to build an algorithm of the local behavior and interaction rules that would provide permanent movement of the system towards the extreme value of this goal.

Legend: Let every processor has an owner and let the performance of every processor be linked with its profit. The task makes sense only when the resources are limited. In this case, the maximum of performance is attained in the point where the respective Lagrangian function has its maximum (Lagrangian factors work here as resource price). So, the owner must maximize the profit of tasks solving, minus the cost of the memory in use. Now we should introduce a mechanism of forming memory prices.

Assume that at every step of memory reallocation every owner sends requests to all available memory blocks. These requests are certain sums of money $d_{i,j}$ ("sell me $d_{i,j}$ worth of memory"). The memory block allocates whole capacity of its memory M_j proportionally to the received money: $m_{ij} = M_j \; d_{ij} / \sum d_{kj}$. The price of the memory in j-th block is uniquely determined by i-th owner as $\lambda_j = d_{ij}/m_{ij}$. Then the following algorithm provides the simplest variant of the optimizing behavior: $\Delta m_{ij} = k\left(\Delta F_i(m_i)/\Delta m_i - d_{ij}/m_{ij}\right)$.

Global optimum: Balance situation determined by the above algorithm $\left(\Delta F_i(m_i)/\Delta m_i - d_{ij}/m_{ij} = 0\right)$ is the point of Nash balance, not providing the global extreme [7]. The reason is that the capacity of the memory used by the i-th processor affects the capacity of the memory used by its neighbors. Note that this is true only for direct neighbors. The essence of the above statement is the following. If the local estimation function is equal not to the single $F_j(m_j)$ but to the sum of such functions over the closest neighborhood, the global extreme point turns to the point of Nash balance. In terms of the interpretation, it is equal to introducing the law of reallocating profits (tax) inside of an every neighborhood.

2.2. Dispatching tasks in a heterogeneous computing system

Task: A flow of heterogeneous tasks arrives at a heterogeneous computing system. Every task is described by vector $W_j = (w_{j0}, \ldots, w_{jm})$, which is the set of workloads for various types of works. Tasks are divided into *k* types. Inside of the every type the tasks have the same or a similar distribution of workloads. Tasks of every type arrive at the system randomly with flow density λ_j. The system consists of *n* heterogeneous processors. Every processor is characterized by a vector of a performance for different types of works $A_i = (a_{i0}, \ldots, a_{im})$ (a_{ik} is the time that necessary for i-th processor to perform k-th work of the unit workload). Thus, the performance of i-th processor at j-th task is equal to the scalar product $b_{ij} = A_i W_j = \sum a_{ik} w_{jk}$.

The input flow of tasks enters the general queue, being allocated later to the processors as they become free. From the standpoint of the user (source/sources of the tasks), the quality of system functioning is characterized by the average and maximum times of waiting $\bar{T}$ and $\max_j T_j$. The system parameter linked with $\bar{T}$ is an average length of the queue $\bar{Q}$.

Legend: Dispatcher's mission is an optimal allocation of tasks to processors. Optimal dispatching is a complex and time-consuming problem.

Queue control usually requires a well-developed system of priorities. The priorities themselves are formed either on the base of theoretical results or on the base of sense-oriented heuristic consumptions. In our case, it looks fairly reasonable for increasing the system throughput to order the task priorities for every processor according to performance values $\sum w_{ij} / \sum w_{ij} a_{kj}$.

Again, let every processor has an owner and let the client (source of tasks) pay for solving each task a sum equal (or proportional) to the general workload of the task: $v_i = \sum w_{ik}$. Then the profit of the j-th processor's owner in a unit of time, when solving the i-th task, is equal to $\sum w_{ij}/b_{ji}$. It is easy to see that the growth of the performance of the processor when solving tasks of a given type, leads to the growth of its "profitability" for the owner to solve these tasks.

It is known from the queuing theory for homogeneous service systems that giving the highest priority to the client with the minimum service time reduces the average queue length and average time of waiting. For heterogeneous systems, the relation between priorities formed on the base of performances and service times is fairly complicated. When forming local preferences, service time can be taken into account by introducing a

constant constituent into the service fee (like the one we pay when take a taxi): $p_{ij} = \left(c + \sum w_{ik}\right) / b_{ji}$. This makes tasks of small workload more “profitable” (c is a parameter of the external control).
We consider a task allocation when the owner forms a limited number of preferences based on how “profitable” tasks of different types are for him. When asking for the next task, the owner informs about two or three of his priorities. If the queue contains a task satisfying his preferences, he gets it. Otherwise, he is given the first task from the queue. The results of computer modeling are given for various ways of forming local priorities.
For implementation of the proposed approach we use StarLogo - the popular learning environment for decentralized system modeling [9]. StarLogo is a programmable modeling environment for exploring the workings of decentralized systems - systems that are organized without an organizer, coordinated without a coordinator. With StarLogo, students model many real-life phenomena, such as bird flocks, traffic jams, ant colonies, and market economies [1].
In decentralized systems, orderly patterns can arise without centralized control. Increasingly, researchers are choosing decentralized models for the organizations and technologies that they construct in the world, and for the theories that they construct about the world. But many people continue to resist these ideas, assuming centralized control where none exists - for example, assuming (incorrectly) that bird flocks have leaders. StarLogo is designed to help students (as well as researchers) develop new ways of thinking about and understanding decentralized systems.
StarLogo is a specialized version of the Logo programming language. With traditional versions of Logo, students can create drawings and animations by giving commands to graphic "turtles" on the computer screen. StarLogo extends this idea by allowing controlling thousands of graphic turtles in parallel. In addition, StarLogo makes the turtles' world computationally active: students can write programs for thousands of "patches" that make up the turtles' environment. StarLogo is particularly well suited for exploring massively parallel phenomena.

3. Conclusion

We have proposed an approach for modeling complex systems’ behavior. The approach is based on forming system of laws and rules that provides achieving goals of the system. These laws can be considered as a global control within a decentralized system. Two examples for illustration of the approach have been presented.
Proposed model of a global control within a decentralized system, being implemented within StarLogo environments, allows enriching the lesson based on system modeling. Distinction between the local and the global goals of the system from the one hand, and the central role of the extreme principle from the other, turn the lesson toward deepness in understanding system dynamic fundamentals.

REFERENCES

[1] M. Resnick, *Turtles, Termites, and Traffic Jams. Explorations in Massively Parallel Microworlds*. (Complex Adaptive Systems series, MIT Press, 1994).
[2] M. L. Tzetlin, *Research in the field of Automata and Biological Systems Modeling*. (Moscow, «Nauka», 1960).
[3] M. L. Tzetlin, V. I. Varshavsky, Automata and Models of Collective Behavior. *Proc. III Congress of IFAC,* paper 350, London, 1966.
[4] V. I. Varshavsky, *Collective Behavior of Automata*, (Nauka, Moscow, 1973), German edition: W. I. Warshawski, *Kollektives Verhalten von Automaten*, (Akademie-Verlag-9/23/019/23/01Berlin, 1978).
[5] V. I. Varshavsky, Collective Behavior and Control Problems. *Proc. Machine Intelligence III*, Edinburgh University Press. 1968.
[6] V. I. Varshavsky, The Organization of Interaction in Collective of Automata. *Machine Intelligence*, (Edinburgh University Press, 1969).
[7] V. I. Varshavsky, M. V. Meleshina, A. I. Semenov, *Organizing collective Behavior in Job Distribution Problem*, (Automatica 7, Pergamon Press, 1971).
[8] V. Varshavsky, T. Kunii, V. Savchenko. Decentralized Distributed Resources Allocation in Massively Parallel Computing Systems, *Proc. of the Second International Conference on Massively Parallel Computing Systems*, (Ischia, Italy, May 6-9, 1996), IEEE Computer Society Press, pp 103 – 109.
[9] U. Wilensky. Connected Mathematics: Making Sense of Complex Phenomena Through Building Object Based Parallel Models http://ccl.northwestern.edu/cm/starlogoT/