The International Workshop
on Discrete-Event System Design, DESDes'01,
June 27÷29, 2001; Przytok near Zielona Gora, Poland

ALGORITHM FOR OPTIMIZATION OF PARALLEL COMPUTATIONS ON THE BASIS OF GENETIC ALGORITHMS AND MODEL OF A VIRTUAL NETWORK

Raouf Kh. SADYKHOV*, Aliaksei V.OTWAGIN**

*-Prof. D.Sc., Byelorussian State University of Informatics and Radioelectronics, 6 P.Brovka st., 220600, Minsk, BELARUS

**- post-graduate student, Institute of Engineering Cybernetics, 6 Sourganov st. 220072, Minsk, BELARUS, *forlelik@mail.ru*

***Abstract**: The problem of the program modules assignment onto processors of the multiprocessor computing system is considered. The general statement of a task and initial data for its description is given. The common approach and number of algorithms, used for solution search, is considered and their basic features are analyzed. The algorithm of the solution search, using representation of parallel algorithm as virtual neural network with training, based on the genetic algorithms, is offered. The comparative estimations of clusterization algorithms and algorithm of a virtual network training are proposed.*

***Key Words**: parallel program, virtual neural network, clusterization, genetic algorithm.*

1. INTRODUCTION

The problem of the parallel program modules assignment onto processors of the multiprocessor computing system is related with class of so-called NP-complex tasks. The goal is such distribution of the interconnected modules set for the program onto parallel system processors, at which the most uniform loading of processors by computing is provided, and also the time and cost of information interchange between modules located on different processors is minimized. At the exact solution of such task on condition that module can be assigned on any processor it is necessary to consider N^M variants for an modules allocation on processors, where N is the number of the system processors, M – the number of the program modules. If N and M are great enough, the exact solution can not be found for acceptable time, especially when we have problem of real time control. Therefore, for the

decision of such tasks the certain heuristic algorithms, which generally are not ensuring the optimum decisions, are used.

In given report a number of algorithms to be used for distribution of the parallel program modules on processors of the parallel computing system is considered. The analysis of these algorithms, offered in [1], has allowed to estimate general heuristics and offer a search method of the optimum or suboptimum solution search for finite time.

2. STATEMENT OF A TASK

Let's consider the multiprocessor system, in which each processor has own RAM, the processors are non-uniform under the performance, and there are communication channels between all processors. The synchronization losses for information interchange between processors are not taken into account.

The parallel program, executed in system, is divided on modules, where the subset can be executed only on the certain processor (that have the certain resource), and other subset can be executed on the any processor. The module represents indivisible unit of the information processing (elementary operation). Thus, the module chosen for running on the processor at certain moment, occupies it without interruption on all time of it's execution.

In world practice of the decision of such tasks the methods from the graph theory and the network diagrams are widely applied. Let's number of modules of the program is equal to M, and number of processors in system is N, then parallel program is represented as directed acyclic weighted graph (DAG). The graph can be described as a tuple G = (V, E, C, T), where:

V - set of graph nodes representing separate modules of the program. Each of modules is characterized by execution time $t_{m,n}$ ($m \in M, n \in N$);

E - set of communication edges representing connection between modules on the information or control continuity. The edges are directed from the module - transmitter to the module - receiver of the information. Weight of an edge determines cost of the information transfer on an edge $c_{i,j}$ ($i, j \in M$) ;

C(M×-M) - matrix of cost determining cost of the information transfer between the certain modules of the program. If there is no connection between modules, then zero is put in the appropriate position;

T(M×N) - matrix of task execution times for each of processors in the system. If the module can not be allocated onto processor, zero is put in the appropriate position.

Each task can be executed only after the execution all of its predecessors are completed. After end of a task the data will transmit to all its followers. Thus, the certain similarity of this statement of a task with a task of a minimal cutsection finding for the network flow is observed. The decision of this task in the general case is decision of a task of linear programming.

3. THE GENERAL APPROACH TO THE ALLOCATION TASK DECISION

In [1] the given task is considered as a task of graph division on subgraphs (clustering), when each of subgraphs contains tasks nominated to the certain processor. There is a number of clustering algorithms, however they provide the optimum solutions on the certain subset of tasks, but do not provide the solutions generally. The basic ideas and characteristics of this group of algorithms are considered below in brief.

All clustering algorithms begin functioning from number of clusters, equal to number of graph nodes. In clustering procedure separate clusters are united, if this association promotes achievement of the goal for algorithm.

The clustering algorithms are based on so-called "edge zeroing". Thus the edges of information interchange between tasks on identical processor are considered as removed, and weight or cost of the communications of these edges is accepted equal 0. At definition of performance time of tasks graph these edges do not influence on estimations. On one step the algorithm can reject the certain number of edges depending on realization.

Each algorithm uses certain heuristics at clusterization. These of heuristics determine edges, which should be "vanished", i. e. the edges, on which two subgraphs of tasks are united in one cluster. Heuristics are be of two types:

1) Heuristics of performance - minimization of performance time, minimization of the communications cost, maximization of function of effective cost;

2) Non-performance heuristics - requirement of clusterization linearity (each task has even one edge connecting with one of tasks for given cluster), miss of cycles in cluster, conformity between clusters and data distribution in system. These heuristics can be taken into account with a various priority, thus the results of clusterization can be essentially differed. In a Fig. 1 the examples of graph division are given with account of various heuristics.

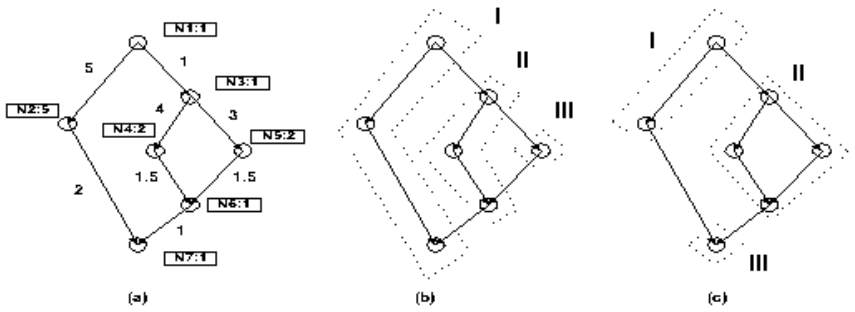

Fig. 1. Clustering examples

The Fig. 1a represents graph for the parallel program. The graph nodes have a designation Ni:c, where i - number of node, c - its performance time. The graph edges have weight indicating time of transition on this edge in a case, when the tasks, appropriate to its beginning and the end, are nominated to different processors. The Fig. 1b represents an example of linear clusterization, 1c – nonlinear clusterization. Performance time(PT) of the parallel program (in relative time units - RTU) for linear clusterization (1b) PT=12.5 RTU, for nonlinear (1c) PT=9 RTU.

Usually heuristics are defined as cost functions of the certain kind allowing its calculation for polynomial time. Thus some special cases of task graphs allowing formulate an estimation function are usually considered.

The clusterization algorithms also assume that technique of "search without return " (non-backtracking) are used.

Each of algorithms is estimated on expenses of time necessary for search of optimum clusterization. The estimation is made by quantity of conditional operations v and e, where v - operation of the analysis for one of nodes, e - operation of the analysis for one of edges. Some algorithms of "edge zeroing" are below listed.

1) Algorithm Kim and Browne's [2]. This algorithm is based on a critical path search in the graph and association of nodes of a critical path in one cluster. Critical path is the path, which has a greatest length or the sum of edges cost.

2) Algorithm Sarkar's [3]. The essence of algorithm in sorting all edges according decrease of their cost and consecutive zeroing of edges of maximal cost, while it does not contradict the algorithm goal. The goal consists in minimization of parallel time, but it is transformed to minimization of communications cost between units.

3) Algorithm of a dominant sequence [4]. Dominant sequence is a critical path in the graph calculated on each clustering step. Thus, later sequence can include edges from earlier sequence. If edge zeroing does not increase the time of start for a task, this task is added in same cluster, otherwise one concerns to new cluster.

The various statements of the optimization goals impose the features on clustering result. All above mentioned algorithms generally give various results, and are effectively applied only on the limited set of tasks. The algorithm for search of a minimal cut of a network flow [5] takes into account simultaneously execution time for a task and time of information interchanges between tasks. The algorithm uses for the search the modified graph of intermodular connections. In this graph except for edges determining the information transfer, there are edges determining execution time of a task on each of processors. Graph cut unequivocally defines distribution of modules on processors, and weight of a cut is a total cost of the program performance for the given distribution of modules on processors. The opportunity of consecutive application of algorithm for a cuts search for greater, than 2, number of processors, is specified. Thus the following difficulties were specified:

1. Unit nominated on n-th processor in multiprocessor system, will not be always nominated to same unit in dual-processor system.
2. The algorithm cannot be used, if the removal of one of processors of system is made.

4. VIRTUAL NETWORK MODEL

In the given report the algorithm of the minimal cut search of a network flow is considered from the point of view of virtual neural network model, which basic concepts are determined in [6]. In this model set of graph elements is divided on clusters. Each element of cluster is closely connected to other elements in this cluster, i.e. the cost of connections inside cluster should be higher, than cost for external intercluster connections for this cluster. The connection is considered internal for cluster, if the nodes of its beginning and ending belong the given cluster. Thus, the localization of the data exchange inside cluster is achieved. The given rule corresponds to a principle of zeroing most of "expensive" connections. The model of a virtual network is easily scaled on any number of processors and tasks. The given model assumes self-organizing, i.e. alignment of estimations for separate clusters and convergence them to some average size.

The ability of model to self-organizing during its definition or change allows at a correct choice of criteria of an estimation receive splitting the graph of tasks on processors with optimum cost of graph execution on the computing system.

For evaluation function of an estimation of the received model everyone cluster is considered, and the general estimation is defined by the sum of estimations for all clusters.

The goal of the graph clusterization in a virtual network is simultaneous convergence to high reliability for everyone cluster with the minimal loading of the appropriate processor. The loading is determined by the sum of performance times for tasks nominated to the given processor.

The solution search for this task of linear programming is fulfilled by the methods of combinatorics in a combination with methods of the theory of neural networks. Thus the method of construction of a training sequence with the help of genetic algorithms [7] is used. Such method allows simultaneously to consider the whole group of the possible solutions and

fulfill parallel search in various areas of decision space. Therefore probability of reception of the optimum solution is much higher.

At the appropriate coding of the possible solution for the task of chromosome analysis and development of genetic operators of its change (mutation and crossover) it is possible to construct genetic algorithm of search with properties " search without return ". Use of various techniques of search: search in N iterations, search before improvement of the decision within the limits of some $0<\varepsilon<1$, search with an interdiction of separate directions - will allow receive the optimum solution for acceptable time.

5. SEARCH ALGORITHM BASED ON A VIRTUAL NETWORK TECHNIQUE

The essence of algorithm for training of a virtual network with the help of genetic algorithms consists in application of neural networks training principles for training sets generated by genetic algorithm. Thus the virtual network is considered as multi-layered neural network with direct connections.

At training model of a virtual network the Hebb principle [8] have been used : increase of weight of connections promoting reception of the best solution. The weight matrix of a virtual network is initialized by random values. Then via genetic algorithm the most acceptable variant of clusterization is computed according to criterion functions. This variant can be used as a training set for a network.

For experiments the program model, which simulate training of a virtual network for the random graph, has been constructed. The results of modeling allow to make a conclusion about dependence them both from the size the graph, and from its density - number of edges between nodes. The comparison of the received results with results of Sarkar algorithm [3] is given below.

The experiments were spent for random graphs, containing from 10 up to 50 nodes. The average performance time of the parallel program was defined, and its modules were distributed according to clusterization results. Besides the loading of processors actually by calculations which have been not connected to transfer of the data or idle times was defined. For each algorithm 50 experiences have been executed.

The Fig. 2 demonstrates average loading of the processor for two algorithms, where VN - algorithm of a virtual network.

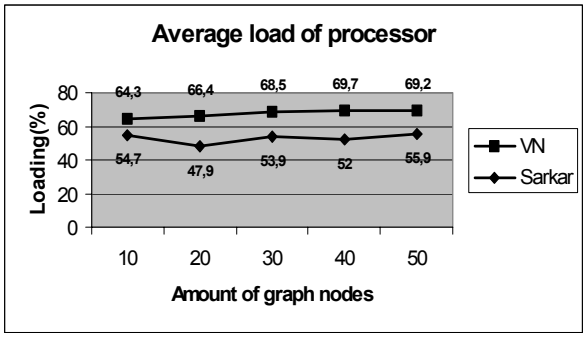

Fig. 2. Loading of system processors at clusterization.

The average loading of processors by calculations is more for algorithm of a virtual network. The uniformity of loading requires allocation of some tasks for processors which are not ensuring generally of the earliest performance a tasks. However, the result of Sarkar's algorithm usually assumes presence in system of the processor which executed considerably large (sometimes in 2-3 times) loading in comparison with others.

The Fig. 3. illustrates process of training for model of a virtual network for 30 tasks. The training was spent for 1000 variants of clusterization. The choice of variant by genetic algorithm was spent in 200 iterations.

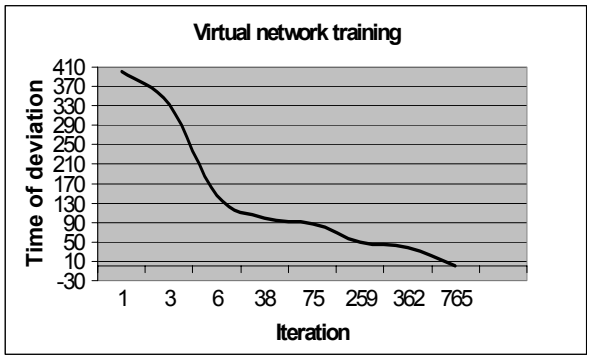

Fig. 3. Process of training of a virtual network

The increase of iterations number for training or choice of a training set increases time of solution search for a virtual network. However, the quality of the received solutions (performance time of the program and uniformity of loading) thus also are improved.

REFERENCES

[1] *A Comparison of Clustering Heuristics for Scheduling DAGs on Multiprocessors*. A.Gerasoulis and T. Yang. Department of Computer Science, Rutgers University, New Brunswick, NJ 08903., 1996

[2] Kim S.J., and Browne J.C. A General Approach to Mapping of Parallel Computation upon Multiprocessor Architectures, *International Conference on Parallel Processing*, vol. 3, 1988, pp. 1-8

[3] Sarkar V. *Partitioning and Scheduling Parallel Programs for Execution on Multiprocessors,* The MIT Press, 1989.

[4] Yang T., Gerasoulis A. A Fast Static Algorithm for DAGs on an Unbounded Number of Processors, *Proc. of Supercomputing ' 91*, *IEEE*, 1991, pp. 633-642.

[5] Trakhtengertz E.A. *The software of parallel processes*. M.: Science, 1987.(In Russian)

[6] Yufik Y. M., Sheridan T. B. Virtual Networks: New framework for operator modeling and interface optimization in complex supervisory control systems, *A Rev. Control*, vol. 20, pp. 179-195.

[7] Michalewicz Z. *Genetic Algorithms + Data Structures = Evolution Programs*. Second, Extended Edition. Springer-Verlag, 1994, 340 pp.

[8] V.A. Golovko. *Neurointelligence: the theory both application. V.1: Organization and training of neural networks with direct communications and feedback*. Brest, 1999.(In Russian).