Multicriteria Scheduling Strategies in Scalable Computing Systems

Victor Toporkov

Computer Science Department, Moscow Power Engineering Institute,
ul. Krasnokazarmennaya 14, Moscow, 111250 Russia
Phone: +7(495)3627145; Fax: +7(495)3625506
ToporkovVV@mpei.ru

Abstract. An approach to generation and optimization of scheduling and resource allocation strategies in scalable computing systems is proposed. The approach allows the decomposition of the problem of multicriteria strategy synthesis for the totality of parameterized models of programs with the use of partial and vector quality criteria including, for instance, a cost function and load balancing factors.

Keywords: scheduling, resource allocation, strategy, scalability, quality criteria.

1 Introduction

The need for special resource management mechanisms in distributed computing systems arose a long time ago and is well-recognized [1]. In some cases, complex sets of interrelated tasks (jobs) require co-scheduling [2] and resource co-allocation [3] in several processing nodes. Each node may be in an autonomous administrative domain and be represented by a multi-processor unit managed by a local batch system, e.g. CODINE, LL, LSF, NQE, Condor, PBS etc. Analysis of the resource co-allocation problem in distributed systems, including Grid, has shown that efficient management of job processing can be implemented on the basis of strategies that include combinations of different scheduling algorithms and heuristics [4, 5], various factors and critera (management policies, workload etc.) [3, 6]. In a number of papers [3-7], the authors conclude that it is necessary to use multifactor and multicriteria strategies. However, in practice only one of the possible resource allocation algorithms is used, and the set of criteria is convolved into a scalar productivity function [3]. In [6], a method for strategy generation in real-time computer systems is proposed.

In this paper, the method proposed before in [6] is developed and refined as follows. The problem of multicriteria strategy synthesis is considered for different parameterized graphs of programs. In the case of a single program model, it may occur that a schedule does not exist. One possible reason is that there are no free processors because of failures in the system. Therefore, it is impossible to resolve the collisions of parallel tasks [7] that compete for the same processor node. Hence, it is necessary to have strategies for program models with different levels of parallelism and task details.

V. Malyshkin (Ed.): PaCT 2007, LNCS 4671, pp. 313–317, 2007.

2 Assumptions and Statement of the Problem

By T_0^*, we denote the set of program models. Each of these models is associated with some totality of partially ordered tasks $T = \{T_1, T_2, ..., T_n\}$. The relation of the partial order on T is specified by a directed acyclic graph whose set of vertices corresponds to tasks of processing and memory access in subset $P \subseteq T$ and to tasks of data exchange in subset $D \subseteq T$. The set of arcs of the graph represents the informational and logical relations between the tasks. Fig. 1 shows some examples of information graphs in models with different degrees of parallelism and task details.

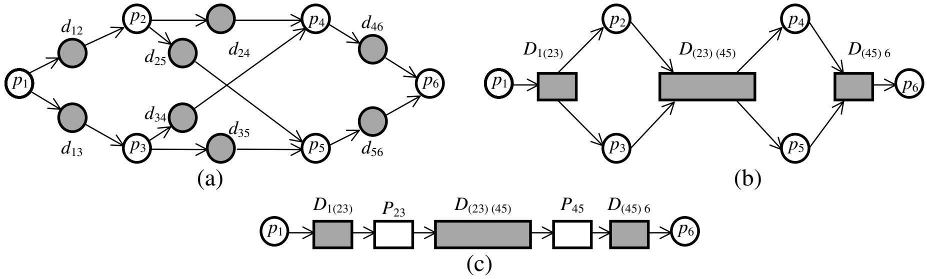

Fig. 1. Information graphs of programs with different degrees of parallelism and task details

The nonshaded vertices correspond to data processing, and the shaded ones correspond to data transmission. The graph of the program is parameterized by a priori estimates, namely, the running time t_{ij}^0 of task $T_i \in T$, $i = 1, ..., n$, on the jth type of processor resource, $j = 1, ..., J$; the amount v_{ij} of computations etc. The parameters of processing tasks are given in Table 1, which corresponds to the graphs shown in Figs. 1a and 1b. When task aggregating, as shown in Fig. 1b and 1c, the values of the corresponding parameters of subtasks are summarized. The duration of all of the data exchanges for the graph in Fig. 1a is equal to one time unit, while data exchanges $D_{1(23)}$ and $D_{(45)6}$ in the graphs in Figs. 1b and 1c need two time units and data exchange $D_{(23)(45)}$ needs four time units. In the resource allocation for tasks in T on a time interval $[0, t^*]$ a resource type is determined by the allocation u_i. We have $u_i = j$ if task $p_i \in P$ is assigned to a so-called basic processor resource whose level is bounded and depends on the parallelizing degree, the cost of the jth type resource, and some other factors [6]. If there occurs a collision of parallel tasks [7] in P, which compete for the same resource of type j, then, taking into account the architecture scalability, we introduce a resource of type $j^\circ \in \{1, ..., J\}$ (whose characteristics are not worse than those of the basic resource) and assign $u_i = j^\circ$. We represent a variant of admissible resource allocation in a quality criterion $w(r)$ by a vector $r = (t_1,...,t_n,u_1,...,u_n)$, and t_i is the running time of task $T_i \in T$. We estimate the efficiency of the resource allocation by the vector $W(r) = (w_1(r),...,w_L(r))$, where $w_l(r)$, $l = 1,...,L$, is a partial criterion.

Table 1. Estimates of parameters of tasks

Parameters	Processing tasks					
	p_1	p_2	p_3	p_4	p_5	p_6
Running time on the processor 1	2	3	1	2	1	2
Running time on the processor 2	4	6	2	4	2	4
Running time on the processor 3	6	9	3	6	3	6
Running time on the processor 4	8	12	4	8	4	8
Amount of computations	20	30	10	20	10	20

An example of the efficiency criterion is a cost function of the form

$$\mathrm{CF} = \sum_{i=1}^{n} c_i(t_i, u_i) = \sum_{i=1}^{n} \left\lceil v_{ij} / t_{ij} \right\rceil,\ t_{ij} \geq t_{ij}^0, \tag{1}$$

where t_{ij} is the time of execution of task p_i on a processor of the j th type, n is the number of processing tasks, and $\lceil \cdot \rceil$ denotes the smallest integer not less than a given number.

Suppose that the active (binding) constraints

$$t_g^* - t_g \geq 0,\ t_h^* - \sum_h t_h \geq 0,\ g,\ h \in \{1,...,n\}, \tag{2}$$

are specified for individual tasks and jobs of the program, where t_g, t_h are the execution times for tasks T_g, $T_h \in T_i$ and t_g^*, t_h^* are limiting times of execution of task T_g and a job that includes the task T_h.

Let S be a strategy, i.e., a set of alternatives such that each alternative $r \in S$ corresponds to an admissible resource allocation under the constraints (2). The vector criterion $W(r)$ generates a binary relation F (e.g., the Pareto-relation) for comparison of alternatives on S. We refer to the set of alternatives optimal with respect to F as an F-optimal strategy of resource allocation. It is required to find an F-optimal strategy for all models of the set T_0^*.

3 Strategy Synthesis by the Totality of Criteria and Models

When searching for the F-optimal strategy on the basis of the parallel scheme [7], the synthesis may be decomposed by the totality of basic schemes, each one providing a conditionally optimal strategy by the corresponding partial criterion $w_l(r)$.

Example 1. Suppose conditionally optimal strategies of process allocation should be constructed by the basic scheme [7] for the information graph in Fig. 1a. The limiting time $t^* = 20$ is specified for the execution of all these tasks. Let the vector criterion include the cost function CF (1) and the loading factors $\mathrm{UP}_j, j = 1,...,4$, of the basic processors. The collisions between competing tasks are resolved at the expense of nonallocated basic processors such that their inclusion in the set of resources is accompanied by the minimal value of the penalty cost function on the analogy of (1), where $v_{ij} = v_{ij^\circ}$, $t_{ij} = t^0_{ij^\circ}$, $j^\circ \in \{1,..., J\}$. Strategies are constructed for the upper and lower bounds of the maximal interval of the variation of t_i. Strategies conditionally optimal by criteria CF, UP_1, UP_2, UP_3, and UP_4 are represented in Table 2 by the variants No. 1-3; 4-7; 8 and 9; 10 and 11; and 12-14, respectively. The collisions between tasks p_4 and p_5 in variants 2, 13 are resolved by allocating task p_4 to a processor of type 3 and task p_5 to a processor of type 4.

Table 2. Scheduling strategies for the graph in Fig. 1a

No.	Running time						Allocation						Criterion				
	t_1	t_2	t_3	t_4	t_5	t_6	u_1	u_2	u_3	u_4	u_5	u_6	CF	UP_1	UP_2	UP_3	UP_4
1	2	3	3	2	2	10	1	1	3	1	2	4	41	0,35	0,10	0,15	0,50
2	2	3	3	10	10	2	1	1	3	3	4	1	37	0,35	0	0,65	0,50
3	10	3	3	2	2	2	4	1	3	1	2	1	41	0,35	0,10	0,15	0,50
4	2	3	3	2	2	10	1	1	3	1	2	1	41	0,85	0,10	0,15	0
5	2	3	3	10	10	2	1	1	3	4	1	1	38	0,85	0	0,15	0,50
6	2	11	11	2	2	2	1	4	1	1	2	1	39	0,85	0,10	0	0,55
7	10	3	3	2	2	2	1	1	3	1	2	1	41	0,85	0,10	0,15	0
8	2	11	11	2	2	2	1	4	2	1	2	1	39	0,30	0,65	0	0,55
9	10	3	3	2	2	2	2	1	2	1	2	1	41	0,35	0,75	0	0
10	2	11	11	2	2	2	1	3	4	1	2	1	41	0,30	0,10	0,55	0,55
11	10	3	3	2	2	2	3	1	3	1	2	1	41	0,35	0,10	0,60	0
12	2	3	3	2	2	10	1	1	3	1	2	4	41	0,35	0,10	0,15	0,50
13	2	3	3	10	10	2	1	1	3	3	4	1	39	0,35	0	0,65	0,50
14	10	3	3	2	2	2	4	1	3	1	2	1	41	0,35	0,10	0,15	0,50

Applying a family of parallel schemes, we synthesize strategies conditionally optimal by the corresponding partial criterion $w_l(r)$ for all models from T_0^*.

Example 2. Consider the models which are presented by the graphs in Figs. 1a and 1c. For the graph in Fig. 1a, the initial conditions are the same as in Example 1. For the graph in Fig. 1c, the strategy is constructed on the whole interval of t_i variation. We must construct the F-optimal strategy, where F is the union of G_l, $l = 1,..., L$, and G_l is generated by one of the criteria CF, $\mathrm{UP}_1,...,\mathrm{UP}_4$. The results of the resource allocation for the graph in Fig. 1c are presented in Table 3 by the variants No. 1-6.

The F-optimal strategy coincides with the strategy presented in Tables 2 and 3 up to the equivalence relation.

Table 3. Scheduling strategies for the graph in Fig. 1c

No.	Running time				Allocation				Criterion				
	t_1	t_{23}	t_{45}	t_6	u_1	u_{23}	u_{45}	u_6	CF	UP_1	UP_2	UP_3	UP_4
1	2	8	6	4	1	1	1	1	25	1	0	0	0
2	4	8	3	5	1	1	1	1	24	1	0	0	0
3	6	4	6	4	1	1	1	1	24	1	0	0	0
4	8	4	3	5	1	1	1	1	27	1	0	0	0
5	10	4	3	3	1	1	1	1	29	1	0	0	0
6	11	4	3	2	1	1	1	1	32	1	0	0	0

4 Conclusions

In this paper, we propose the approach for the problem of multicriteria scheduling strategy synthesis in computing systems with a scalable architecture.

First, this approach allows us to obtain a strategy, which is conditionally optimal by a partial criterion. Second, the strategy synthesis may be decomposed by the totality of partial criteria. Finally, the general decomposition allows us to generate scheduling strategies by a vector criterion for different models of the same program.

Acknowledgments. This work was supported by the Russian Foundation for Basic Research, grant no. 06-01-00027.

References

1. Casavant, T.L., Kuhl, J.G.: A Taxonomy of Scheduling in General-Purpose Distributed Computing Systems. IEEE Trans. on Software Eng. 14(2), 141–154 (1988)
2. Ioannidou, M.A., Karatza, H.D.: Multi-site Scheduling with Multiple Job Reservations and Forecasting Methods. In: Guo, M., Yang, L.T., Di Martino, B., Zima, H.P., Dongarra, J., Tang, F. (eds.) ISPA 2006. LNCS, vol. 4330, pp. 894–903. Springer, Heidelberg (2006)
3. Kurowski, K., Nabrzyski, J., Oleksiak, A., et al.: Multicriteria Aspects of Grid Resource Management. In: Nabrzyski, J., Schopf, J.M., Weglarz, J. (eds.) Grid Resource Management. State of the Art and Future Trends, pp. 271–293. Kluwer Acad. Publ., Boston (2003)
4. Zhang, Y., Franke, H., Morreira, J.E., et al.: An Integrated Approach to Parallel Scheduling Using Gang-Scheduling, Backfilling, and Migration. IEEE Trans. on Parallel and Distributed Systems 14(3), 236–247 (2003)
5. Hanzich, M., Gine, F., Hernandez, P., et al.: CISNE: A New Integral Approach for Scheduling Parallel Applications on Non-dedicated Clusters. In: Cunha, J.C., Medeiros, P.D. (eds.) Euro-Par 2005. LNCS, vol. 3648, pp. 220–230. Springer, Heidelberg (2005)
6. Toporkov, V.V.: Optimization of Resource Allocation in Hard-Real-Time Environment. J. of Computer and Systems Sciences Int. 43(1), 383–393 (2004)
7. Toporkov, V.V.: Decomposition Schemes for Synthesis of Scheduling Strategies in Scalable Systems. J. of Computer and Systems Sciences Int. 45(1), 77–88 (2006)