10th International Conference of the Practice and Theory of Automated Timetabling
PATAT 2014, 26-29 August 2014, York, United Kingdom

Online Scheduling System for Server Based Personnel Rostering Applications

Přemysl Šůcha · István Módos · Roman Václavík · Jan Smejkal · Zdeněk Hanzálek

Keywords Scheduling · Distributed Computing · Grid Computing · Machine Learning · Personnel Rostering

1 Introduction

In the recent time many software applications have web based character. It means that the computer of the user is used as a terminal while the application is running on a server. Nevertheless, the server may be a bottleneck of the application from the performance point of view. All data manipulations and all computations are processed there and if the server is overloaded it usually causes slow reactions of the application measured by *response time* of the server. Moreover, if the server process non-trivial requests like solving a combinatorial problem, e.g. *personnel rostering* [1], the performance of the server is significantly worse. In this case the incoming requests must be carefully scheduled in order to efficiently exploit available computational resources and to minimize the system response time.

In this paper we consider a server based *personnel rostering application* where many users send their requests at the same time. To solve the requests the server uses a *heterogeneous grid*, i.e. a collection of computer resources such that each one has different speed. A single request of a user is called task which has to be mapped and solved on a computer resource. Tasks are instances of combinatorial optimization like personnel rostering, personnel rerostering etc. The tasks are usually solved by *anytime algorithms*. It means that if the algorithm is interrupted earlier (but not before the minimal execution time) the solution is a valid solution to the problem, however its *quality* in terms of

P. Šůcha (✉) · I. Módos · R. Václavík · J. Smejkal · Z. Hanzálek
Department of Control Engineering, Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, Czech Republic
E-mail: suchap@fel.cvut.cz

the objective function value is lower. This is an important aspect which allows deteriorating quality of solutions in a defined range when the server side is overloaded. The aim of a *scheduling system* that schedules the incoming tasks on a heterogeneous grid is to balance average response time of the server and average quality of solutions.

There are already many papers dealing with task mapping on grids [5]. However, to the best of our knowledge there is no work dealing with heterogeneous grids taking into account properties of anytime algorithms. As a related paper can be considered e.g. [2] where the authors describe an algorithm for task mapping on GPU resources while considering the trade-off between the quality of solutions and execution time. However, their approach requires specially designed anytime algorithms where the scheduler is able to control the execution path of the algorithms. It also requires quite frequent communication between the scheduling algorithm and the computer resources. A related domain is scheduling of imprecise computation tasks [4] where a task consists of two subtasks: a mandatory subtask and an optional subtask. Each task must be completed before its deadline while its mandatory subtask must be completed and the optional part may be left unfinished. The objective is to minimize a penalty proportional to the lengths of unfinished parts of all tasks. In contrast to our work, where duration of the tasks for given quality is not known, their penalty function is given a priori.

As stated above, the existing approaches cannot be efficient for scheduling of anytime algorithms that solve combinatorial problems. The problem is that the execution time of a task for the given solution quality is not known a priori. That means the incorrectly estimated execution time may cause deterioration of response time of the system. Therefore we propose a scheduling system for online scheduling of tasks having character of anytime algorithm executed on a heterogeneous grid. Based on the task features the scheduling algorithm is able to estimate the execution time of tasks. Moreover, a controller in the scheduling system is able to balance the average solution quality in order to achieve reasonable trade-off between the average response time and the average quality of solutions.

2 Scheduling System Design

Our design of the scheduling system is outlined in Figure 1. *External applications*, controlled by users, generate input tasks TI_i (rrt_i is requested response time for task TI_i). These tasks are sent into the *task list* through the *estimator* which defines estimated execution time $p_i(q_t) = \max(a_i e^{b_i q_t}, met_i)$ of TI_i where q_t is demanded quality in time t and met_i is the minimal execution time of task TI_i. The *scheduler* then selects tasks from the task list and allocates them to the resources with respect to $p_i(q_t)$ and speed coefficient s_r of each resource r [3]. The actual task allocation is executed through the *resource manager* which passes the task data (TI_i) and its assigned time for execution ($p_i s_r$) to the particular *resource* r. Resources may be arbitrary com-

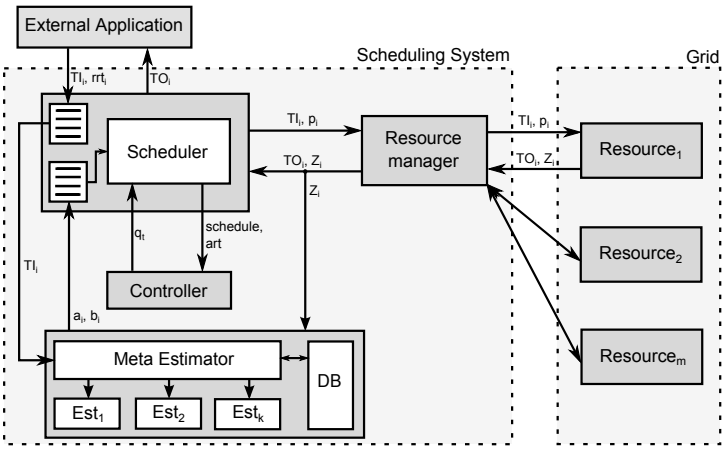

Fig. 1 Scheduling System Design

puters/CPU cores which executes the particular rostering algorithm. When the resource finishes processing the allocated task, it sends the solution (TO_i) together with computing progress (Z_i - progress of the objective function value over time) to the scheduling system. The solution is then sent to the corresponding external application and Z_i is stored in the estimator database for the online re-learning purposes.

The importance of the *controller* is to dynamically control the average response time (art) of the system through the q_t. This quality is dependent on the number of tasks to be processed and the amount of available resources.

3 Preliminary Results

For the preliminary experiments we consider STF+MCT (shortest task first + minimal completion time) scheduler, bisection controller [3] and six computer resources.

The behavior of the scheduling system with bisection controller and without controller on the same data is illustrated in Figure 2 and 3 respectively. The first subgraph in those figures shows the number of waiting tasks over time, the second one displays the average lateness, i.e. how many times the due date of tasks is exceeded, over time and finally the third subgraph indicates the quality set by the controller over time.

A comparison of Figure 2 and 3 shows significant improvement of the average response time of the system with the controller and the estimator. More experimental results and comparisons will be shown at the conference.

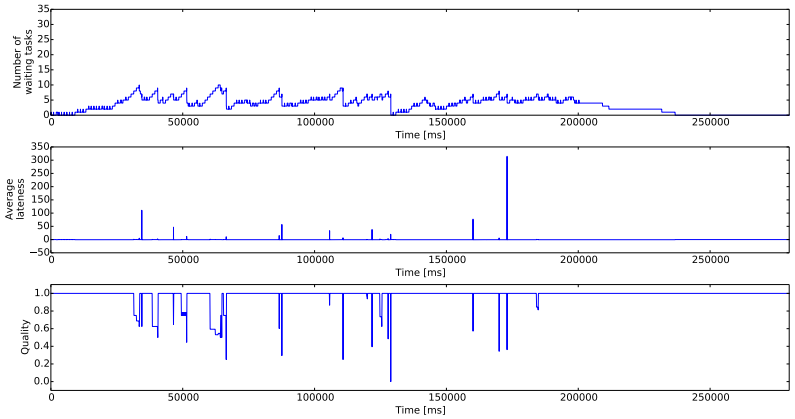

Fig. 2 The behavior of the scheduling system with bisection controller

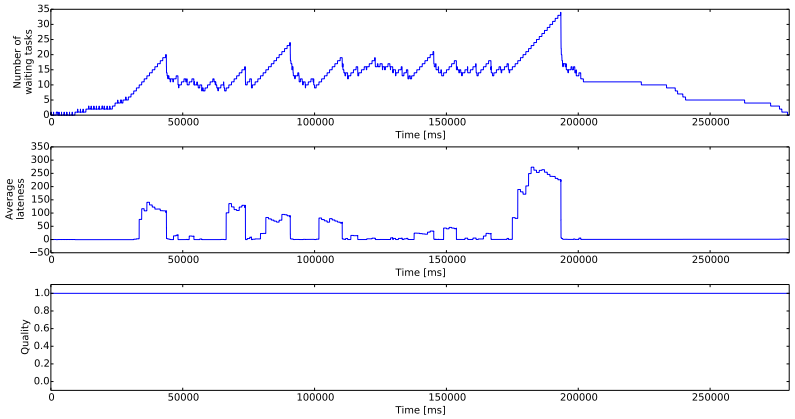

Fig. 3 The behavior of the scheduling system without controller (=constant quality)

References

1. E. K. Burke, P. De Causmaecker, G. Vanden Berghe and H. Van Landeghem, The State of the Art of Nurse Rostering, Journal of Scheduling, 7, 441-499 (2004)
2. R. Mangharam and A. A. Saba, Anytime Algorithms for GPU Architectures, IEEE 32nd Real-Time Systems Symposium, 47-56 (2011)
3. I. Módos, P. Šůcha, R. Václavík, J. Smejkal and Z. Hanzálek, Server Based Rostering Application, Czech Technical University in Prague, Internal technical report (2014)
4. G. Wan and J. Y.-T. Leung and M. L. Pinedo, Scheduling imprecise computation tasks on uniform processors, Information Processing Letters, 104, 45-52 (2007)
5. F. Xhafa and A. Abraham, Computational models and heuristic methods for Grid scheduling problems, Future Generation Computer Systems, 26, 608-621 (2010)