STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVI**, Number 2, 2011

AUTOMATIC SELECTION OF SCHEDULING ALGORITHMS BASED ON CLASSIFICATION MODELS

FLAVIA ZAMFIRACHE AND MARC FRÎNCU

Abstract. Selecting the appropriate scheduling algorithm in distributed heterogeneous systems is a difficult problem. In order to avoid an exhaustive search it is possible to design an automatic selection procedure based on a classification model trained using various characteristics of the tasks to be scheduled. This paper presents a comparative study on the effectiveness of several classification models used to select an effective algorithm for a given scheduling problem. The main contribution of the paper is the hybrid classifier based on non-nested generalized exemplars and an evolutionary selection of attributes and exemplars. The experiments show the ability of the proposed hybrid classifier to identify the appropriate scheduling algorithm when new configurations arrive to the grid scheduler.

1. Introduction

Distributed Heterogeneous Systems require Scheduling Algorithms (SA) in order to efficiently map tasks on existing resources. However due to the unpredictable behaviour of the underlying systems SAs are greatly influenced when trying to optimize the objective cost function (e.g., makespan - time required to complete the schedule, lateness - time delay in executing a task given a specified deadline). The efficiency of the heuristic is both influenced by tasks and system characteristics [1, 7].

So, the problem of designing a SA capable of efficiently dealing with a wide range of scenarios has been given a lot of attention. However most of the work focused on creating improved *switching algorithms* based on existing scheduling heuristics [2, 7, 9], mostly using the Min-Min and Max-Min SAs [7]. The main issue with switching algorithms is that due to the large amount of available SAs and to the tendency to discover new improved versions, creating

Received by the editors: April 4, 2011.

2010 *Mathematics Subject Classification.* 68T20, 68T05.

1998 *CR Categories and Descriptors.* I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search – *Scheduling*.

Key words and phrases. grid scheduler, evolutionary pruning, non-nested generalized exemplars, classification.

a *super-SA* which contains conditional branches to existing heuristics is inappropriate. The reason for this is that the algorithm would require constant re-editing and would eventually become to hard to comprehend.

An alternative to the switching algorithms is a *brute force Best Selection* (BS) strategy in which every existing SA is tested against the existing system configuration. Despite being efficient in identifying the best SA to be applied in a given scenario, this solution has the disadvantage of increasing its runtime when the number of tested algorithms increases. When considering also that the strategy has to be reapplied periodically the time costs can make the approach unsuitable. The periodicity is influenced by tasks completion and arrival events, as they are the only ones that influence the resource load.

Because of the previously mentioned issues an alternative could be to apply BS only in constructing a training set of data. The training data contains several platform characteristics of the scheduling scenario (tasks and resources related) together with the best SA (class label) for that specific configuration, found by BS. This data set could be used to train a classification system. Then, when new configurations occur the classifier generated in the previous step is used to infer the corresponding SA. Different scheduling scenarios need different scheduling algorithms. Extracting the relationship between the characteristics of a scheduling scenario and the corresponding best scheduling algorithm would allow us to design an automatic procedure to select the algorithm suitable (assuring the lowest makespan) to a given scenario. This relationship can be extracted by using either supervised or unsupervised learning. In our experimental analysis we used techniques belonging to both categories.

In this paper we tested several classification strategies in order to find the one that ensures the largest classification accuracy and to identify which characteristics of the scheduling events influence most the choice of an adequate SA. In the experimental analysis we used several classifiers implemented in the WEKA [1] data mining toolkit (MultiLayer Perceptron (MLP) neural network, Radial Basis Function (RBF) network, Non-Nested Generalized Exemplars classifier (NNGE)), a Fuzzy C-Mean unsupervised classifier and a hybrid classifier combining the NNGE algorithm with an evolutionary selection of relevant attributes and exemplars (called Evolutionary Pruning NNGE: EP-NNGE).

2. The EP-NNGE Classifier

The NNGE algorithm is a hybrid instance based learning method which infers from data classification rules represented as non-nested and non-overlapping axes-parallel hyperrectangles [8]. In order to illustrate the NNGE learning process let us consider a set of L training instances (examples), $(E^1, E^2, \ldots, E^L)$,

[1] `http://www.cs.waikato.ac.nz/~ml/weka/`

each one containing the values of N attributes. The aim of the learning process is to construct a set of generalized exemplars (hyperrectangles), $\mathcal{H} = \{H^1, H^2, \ldots, H^K\}$. A hyperrectangle usually covers a set of training instances belonging to the same class. The learning process is incremental, for each example E^j the following three main steps being applied: *classification* (the hyperrectangle H^k which is closest to E^j is identified by using a distance-based criterion), *model adjustment* (the hyperrectangle H^k is split if it covers a conflicting example) and *generalization* (if it is possible, H^k is extended, in order to cover E^j). The classification step is based on the computation of the distance $D(E, H)$ between an example $E = (E_1, E_2, \ldots, E_N)$ and a hyperrectangle $H = (H_1, H_2, \ldots, H_N)$ as given in Eq. (1):

$$D(E,H) = \sqrt{\sum_{i=1}^{N} w_i \frac{d(E_i, H_i)}{E_i^{max} - E_i^{min}}} \tag{1}$$

where $d(E_i, H_i)$ is the distance between the examples attributes and the hyperrectangles sides (Eq. (2) for numerical attributes and Eq. (3) for nominal attributes), and w_i represent the weights corresponding to attributes and are computed based on the mutual information between the attribute and the class.

$$d(E_i, H_i) = \begin{cases} 0 & \text{if } E_i \in [H_i^{min}, H_i^{max}] \\ E_i - H_i^{max} & \text{if } E_i > H_i^{max} \\ H_i^{min} - E_i & \text{if } E_i < H_i^{min} \end{cases} \tag{2}$$

$$d(E_i, H_i) = \begin{cases} 0 & \text{if } E_i \in H_i \\ 1 & \text{if } E_i \notin H_i \end{cases} \tag{3}$$

Once the set $\mathcal{H}$ of hyperrectangles has been generated by the NNGE algorithm, it can be postprocessed in order to reduce its size and, hopefully, to improve the classification accuracy. Following the idea of the hyperrectangles selection presented in [6] we developed an evolutionary pruning algorithm acting as postprocessor of the results produced by NNGE [10]. The first version of the algorithm, called EP-NNGE (Evolutionary Pruning in NNGE) is based on the idea of evolving a population of M binary strings containing K components. Each element, x, of the population corresponds to a subset of H, e.g., if a component x_k has the value 1 it means that H_k is selected into the model, while if it is 0 it means that H^k is not selected. The quality of each element is quantified using two measures: one related to the accuracy of the classifier based on the selected hyperrectangles and the other one related to the reduction of the model size. Thus the fitness of an element x is given by Eq. (4) where *Acc* denotes the accuracy, $|\mathcal{H}|$ denotes the number of hyperrectangles and $\lambda \in (0, 1)$ is a parameter controlling the compromise between the two quality measures.

$$f(x) = \lambda Acc(\mathcal{H}(x)) + (1-\lambda)((|\mathcal{H}| - |\mathcal{H}(x)|)/|\mathcal{H}|) \tag{4}$$

The general structure of the evolutionary selection strategy is inspired by the adaptive algorithm used in [6]. It uses a population of binary encoded elements that is evolving by applying a one point uniform crossover operator. The selection operator was implemented using a truncation selection in order to preserve the best M elements in the population.

The second approach is that of simultaneously selecting hyperrectangles and attributes. In this case each element in the population has $K + N$ components (K being the initial number of hyperrectangles and N being the total number of attributes). The corresponding algorithm (EPA-NNGE) has the same structure as EP-NNGE and the population elements are evaluated also by using Eq. (4). The main difference between EPA-NNGE and EP-NNGE is related to the computation of the classification accuracy: when computing the distance between a test instance and a hyperrectangle, all non-selected attributes are just ignored in the former case.

3. Tests and Results

The supervised and unsupervised classifiers used for testing and their configuration are presented below.

As supervised classification methods we used two neural networks architectures and a NNGE classifier. For NNGE and RBF classifiers we used the default parameters values from WEKA toolkit. For MLP classifier we used 7 output neurons (one for each SA), a learning rate of 0.3 and 8 hidden neurons.

Fuzzy C-Means is an unsupervised classification technique which identifies clusters in data based on some membership values which quantify the degree of similarity between a data and a cluster. It computes the membership values in an iterative way using as input only the data and the number of clusters to be identified. The number of clusters used in tests is equal with 7.

For EP-NNGE and EPA-NNGE classifiers the population dimension is $M = 50$ and the stopping criterion is a combination between a maximal number of generations (100) and a maximal number of generations without progress (50). The value used for λ is 0.995. This value have been chosen in order to increase the classification accuracy and based on a study developed on the datasets from UCI Machine Learning Repository.

Each instance in the training set contains values corresponding to the following attributes: the *time when the schedule was completed*, the *mean task Estimated Execution Time (EET)* (in seconds); the *mean standard deviation of the EET*; the *mean task Estimated Completion Time (ECT)* (in seconds); the *mean standard deviation of the ECT*; the *mean task size* (in bytes); the *mean standard deviation of the task size*; the *total number of tasks*; and the *number of long tasks* used in the experiment. Besides this information for each configuration, the best SA found by BS was added in order to characterize the class. The training set was derived synthetically generated using the models described in [3]. In addition to them the platform heterogeneity factor $h = s_{max}/s_{min} - 1$ (s_{max} is the fastest CPU and s_{min} is the slowest CPU in flops/s) was also considered and used to build two training sets for homogeneous ($h = 0$) and heterogeneous ($h = 42$) environments. Seven SAs were used for determining the best policy: Max-Min [7], Min-Min [7], Suffrage [1], MinQL [4], MinQL-Plain [4], DMECT [5] and DMECT2 [5].

TABLE 1. Classification accuracy

Training set	Inst.	Cls.	Fuzzy C-Means	MLP	RBF	NNGE	EP-NNGE	EPA-NNGE
h=0	303	6	63.93	80.85	67.98	68.42 $\pm$7.67	87.31 $\pm$ 4.66	87.51 $\pm$ 10.00
h=24	366	6	74.81	81.42	50.54	65.41 $\pm$6.82	85.34 $\pm$ 5.00	85.91 $\pm$10.00
mixed	669	7	68.2	65.32	66.24	64.32 $\pm$4.44	83.70 $\pm$ 4.33	83.16 $\pm$10.00

The average runtime of each (un)supervised technique was below 2.5s (training step + classification), while the BS strategy in the case of the 7 SAs requires around 6 seconds to complete one schedule event. The high classification percentages as well as the low runtimes make the learning techniques suitable for determining the best SAs without requiring a BS or switching policy.

Table 1 presents the accuracy of the classification techniques. The behaviour of EP(A)-NNGE classifiers is similar for the three data sets even if the number of selected attributes involved in classification process varies from 100% rate for the first approach to 47% rate for the second one. But in the case of less attributes selection a larger variance is noticed. By analysing the mutual information of each attribute it follows that the biggest amount of information is offered by the number of tasks involved in the scheduling event ($weight = 0.53$) followed by the task duration information (number of long tasks - $weight = 0.26$, % of long task - $weight = 0.16$). These two parameters are also efficient in determining certain SAs. For instance the *total number of tasks* is an important parameter for selecting DMECT while the *number of long tasks* is essential in classifying Max-Min (direct consequence of the study performed in [7]). The rest of the attributes have low weight values (< 0.1).

4. Conclusions

The EP-NNGE heuristic variants perform better than the other analysed classifiers, the most significant difference being observed in case of mixed data (homogeneous data combined heterogeneous data). The task set characteristics that influence the mostly the scheduling heuristic selection are the tasks' number and size. Since the test data sets are unbalanced, containing two dominant classes DMECT and MaxMin, future work will address a hybrid approach between the EP-NNGE algorithm and some specific techniques applied in case of unbalanced datasets.

References

[1] H. Casanova, A. Legrand, D. Zagorodnov, and F. Berman. Heuristics for scheduling parameter sweep applications in grid environments. In *Procs. 9th Heterogeneous Computing Workshop (HCW)*, pages 349–363, Cancun, Mexico, May 2000.
[2] K. Etminani and M. Naghibzadeh. A min-min max-min selective algorihtm for grid task scheduling. In *ICI '07: Proceedings of the 3rd IEEE/IFIP International Conference in Central Asia on Internet*, pages 1–7. IEEE Computer Society, 2007.
[3] D. G. Feitelson. Workload modeling for computer systems performance evaluation, September 2010.
[4] M. Frîncu, G Macariu, and A. Cârstea. Dynamic and adaptive workflow execution platform for symbolic computations. *Pollack Periodica*, 4(1):145–156, 2009.
[5] M. E. Frîncu. Dynamic scheduling algorithm for heterogeneous environments with regular task input from multiple requests. In *Procs. of the 4th Int. Conf. in Grid and Pervasive Computing GPC'09*, volume 5529 of *Lecture Notes in Computer Science*, pages 199–210. Springer-Verlag, 2009.
[6] S. García, J. Derrac, J. Luengo, C. J. Carmona, and F. Herrera. Evolutionary selection of hyperrectangles in nested generalized exemplar learning. *Appl. Soft Comput.*, 11:3032–3045, April 2011.
[7] M. Maheswaran, S. Ali, H. J. Siegel, D. Hensgen, and R. F. Freund. Dynamic mapping of a class of independent tasks onto heterogeneous computing systems. *Journal of Parallel and Distributed Computing*, 59:107–131, 1999.
[8] B. Martin. Instance-based learning: Nearest neighbour with generalisation. In *Working Paper Series 95/18 Computer Science*, page 90, Hamilton,University of Waikato.
[9] M. Singh and P. K. Suri. A qos based predictive max-min, min-min switcher algorithm for job scheduling in a grid. *Information Technology Journal*, 7(8):1176–1181, 2008.
[10] D. Zaharie, L. Perian, V. Negru, and F. Zamfirache. Evolutionary pruning of non-nested generalized exemplars. *Proc. of 6th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI 2011) to be held on May 19-21, 2011 in Timisora, Romania*, pages 57–62, 2011.

Department of Computer Science, West University of Timişoara, bv. V. Pârvan, no. 4, 300223, Timişoara

E-mail address: zflavia@info.uvt.ro, mfrincu@info.uvt.ro