Metadata-driven task scheduling in computer clusters

Ivan, Golubev
Electrotechnical University
Saint-Petersburg, Russia
e-mail: IAGolubev@mail.eltech.ru

Mikhail, Kupryianov
Electrotechnical University
Saint-Petersburg, Russia
e-mail: mikhail.kupriyanov@gmail.com

ABSTRACT

Efficient resource utilization has always been a challenging task in distributed data processing applications. This article is an attempt to formalize the task of job scheduling in heterogeneous computer clusters. An observation of existing correlation between job characteristics and resource requirements is used to estimate resource utilization metrics and assign jobs to cluster nodes. A task to node assignment method is outlined.

Keywords

Computer grid, assignment problem, task scheduling.

1. INTRODUCTION

Centralized resource control is usually carried out by so called *resource allocation (resource planning) system* [1] or *job scheduler (job planning system)* [2], which is usually referred as a middleware: it controls the low level resources, and takes the high level applications' requirements into account.

To do this the system needs to monitor the current resource load in order to assign idle resources to a newly submitted job [1].

Thus, the resource planning system contains the following components [2]:

1. *job scheduler*, performing a task-resource matchmaking
2. *information service*, giving an info about a job execution progress and resources' state.

To handle resources and jobs the scheduler uses *resource allocation (planning, assignment) methods* based on data, provided by the information service.

The problem of jobs scheduling is comprised of resource allocation to jobs and defining an execution order [3].

The following aspects should be taken into account in a job assignment policy:

1. cluster nodes heterogeneity
2. jobs requirements [1] and characteristics
3. job execution history.

Considerable amount of research done in forecasting resource utilization [4], is based on statistical modeling. This work uses the execution history to build prediction models, that is a quite common approach in the field of data processing with computer clusters and cloud systems.

The next section gives a formal description of the tasks scheduling problem in computer clusters.

2. TASK FORMALIZATION

Let $T = \{t_1, t_2, \ldots, t_n\}$, $n \in N$, be a set of jobs (tasks), where each task is characterized by certain metadata (attributes set): $A = \{a_1, a_2, \ldots, a_s\}, s \in N$. Each attribute a_i is a discrete value:

$$a_i \in V_i = \{v_1, v_2, \ldots, v_r\}, r \in N.$$

Let K be a set of processing nodes (to simplify let the amount of nodes be equal to tasks amount n), where each node has certain resource characteristics - ex. threads and memory amounts:

$$\exists K = \{k_1, k_2, \ldots, k_n\}, n \in N : k_i = \{threads, memory\}.$$

$threads$ and $memory$ are also discrete values. For example, they might take the following values:

$$threads \in \{1, 2, 4, 8\}, memory \in \{128, 256, 512, 1024\}.$$

During the data processing the task scheduler assigns tasks T to cluster nodes K:

$$F : T \mapsto K,$$

where F maps a task $t_i \in T$ to a processing node $k_j \in K$.

Each assignment (t_i, k_j) has some cost $c_{i,j}$. The entire mapping of tasks set T to nodes set K is described by a cost matrix:

$$C = \begin{pmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,n} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n,1} & c_{n,2} & \cdots & c_{n,n} \end{pmatrix}.$$

The task scheduler assigns one node to a given job and this job occupies the selected node completely, while different jobs are processed concurrently. Hence, the mapping F, defined by the resource planning system can be described by the assignment matrix:

$$W = \begin{pmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,n} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n,1} & w_{n,2} & \cdots & w_{n,n} \end{pmatrix}.$$

$$w_{i,j} = \begin{cases} 0 & \text{if task } i \text{ is not assigned to node } j \\ 1 & \text{if task } i \text{ IS assigned to node } j. \end{cases}$$

The following constraints hold for the matrix:

$$\sum_{i=1}^{n} w_{i,j} = 1, \sum_{j=1}^{n} w_{i,j} = 1, i, j \in N.$$

- a bijection between tasks and nodes is defined.

The main objective of the resource planning system is to pick coefficients $w_{i,j} \in \{0, 1\}$ of assignment matrix such, that the total cost is minimized:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_{i,j} \cdot w_{i,j} = sum \longrightarrow min.$$

The optimal assignment matrix can be found from the cost matrix using existing algorithms of solving the assignment problem (i.e. bipartile graph matching), most notably the Hopcroft-Karp algorithm [5] in $O(n^3)$ time.

The main problem here is that values $c_{i,j}$ are unknown, and should be estimated. The next sections describe such an estimation.

3. COST CALCULATION ON THE BASIS OF EXECUTION METRICS

This section clarifies, what is the cost value $c_{i,j}$ of running selected job i on processing node j, and how this value can be calculated from the execution metrics.

Each job execution in computing cluster gives certain metrics $M = \{m_1, m_2, \dots, m_k\}, k \in N$. The most typical examples are:

- average processor load,
- average memory usage,
- elapsed time.

These metrics characterize time and resource expenses associated with tasks execution and can be used to assess the execution cost: execution cost is a function, that depends on metrics and maps on interval $[0, d]$:

$$c_{i,j} = F(M) = F(m_1, m_2, \dots, m_k) \longrightarrow [0, d], d \in R.$$

Any function might be used as F, for example:

$$F(m_1, m_2, \dots, m_k) = \\ a_1 \cdot m_1 + a_2 \cdot m_2 + \dots + a_k \cdot m_k = \sum_{i=0}^{k} a_i \cdot m_i,$$

where $a_i \in [0, 1]$ - weight coefficient, that reflects the contribution of each metric to the final cost.

The simplest case is $F = \sum_{i=0}^{k} m_i$ - where each metric is treated identically.

It is often useful to normalize all the values $c_{i,j}$ to map all the cost estimations to standard interval $[0, 1]$, because the only important to the task scheduling is the relative cost of running task i on cluster node j, not the absolute value. This can be done with a simple formula:

$$c_{i,j}^{norm} = \frac{c_{i,j} - c_{min}}{c_{max}}.$$

In the next section under the execution cost $c_{i,j}$ the $c_{i,j}^{norm}$ is implied.

4. COST ESTIMATION ON THE BASIS OF METADATA

The previous section covered how execution cost can be calculated when execution metrics are available. While this approach is perfect when dealing with a training sample of jobs, it makes no sense for real tasks since metrics are only available as a result of execution. On the other hand, cost matrix calculated for the train data is not completely useless, it might be used to estimate cost for the new jobs.

So the main question here is how execution cost for the new (test) jobs can be estimated on the basis of cost matrix of training sample of jobs.

In order to do this we have to establish the relationship between a new test task t_{new} and old training task t_{old}, where the latter has cost value $c_{t_{old}}$ and the same node k is considered for both tasks. In other words, it is required to find similar task(s) and use their associated cost values to estimate cost for a new job c_{new}.

Tasks can be compared using metadata, - the attributes, associated with each task, as it was defined in section 2: $A = \{a_1, a_2, \dots, a_s\}, s \in N$.

Attributes together with some distance function allow to find similarity between tasks. For example, the Hamming distance function might be used:

$$D(t_a, t_b) = \sum_{i=1}^{s} (|a_i| - |b_i|),$$

where a_i and b_i - attributes for jobs t_a and t_b, correspondingly. Efficient data structures such as kd-trees might be used on this step to find similar tasks in linearithmic time.

In a broader sense, such tasks comparison is a kind of data classification, where tasks' metadata are used as the classification attributes. However, it is usually impractical to build a global approximation function, - a local approximation by the nearest neighbours will usually suit [6].

When distance to all tasks for a given task has been calculated, a subset T of the most similar tasks is chosen. In the simplest case one task with the least distance is selected.

Then the execution cost for a new task t_{new} is estimated by an average of cost values for the found set T:

$$c_{t_{new}} = \frac{\sum_{i=1}^{n} c_i}{n}, c_i \in T.$$

In the end, after all the above actions each new task has the associated cost vector, showing the relative cost of running this task on all the available cluster nodes. These vectors together form a cost matrix, which in turn is used to distribute tasks between nodes in an optimal way using the Hopcroft-Karp assignment algorithm.

5. RESULTS

The above described approach to planning jobs in computer clusters can be summarized in the following way. To build a model:

- select a train set of jobs,

- execute these jobs on all types of cluster nodes to gather metrics,
- map metrics to normalized cost values.

To assign new jobs to cluster nodes:

- for each task find similar tasks (nearest neighbours) in the train set,
- estimate a cost matrix by means of cost values of similar tasks,
- retrieve assignment matrix by Hopcroft-Karp algorithm,
- launch tasks on cluster nodes according to the assignment matrix.

This method allows resource planning and task scheduling that takes tasks metadata and nodes characteristics into account and might give performance benefits for certain applications where such information is available.

REFERENCES

[1] P. Endo, A. D. A. Palhares, N. Pereira, G. Goncalves, D. Sadok, J. Kelner, B. Melander, and J.-E. Mangs, "Resource allocation for distributed cloud: concepts and research challenges," *Ieee Network*, vol. 25, no. 4, pp. 42–46, 2011.

[2] M. Chtepen, F. Claeys, B. Dhoedt, F. De Turck, P. A. Vanrolleghem, and P. Demeester, "Performance evaluation and optimization of an adaptive scheduling approach for dependent grid jobs with unknown execution time," in *18th World IMACS Congress and MODSIM09 International Congress on Modelling and Simulation, Proceedings*, pp. 1003–1009, 2009.

[3] J. M. G. Barbosa and B. D. R. Moreira, "Dynamic job scheduling on heterogeneous clusters," in *Proceedings of the 2009 Eighth International Symposium on Parallel and Distributed Computing*, ISPDC '09, (Washington, DC, USA), pp. 3–10, IEEE Computer Society, 2009.

[4] M. T. Imam, S. F. Miskhat, R. M. Rahman, and M. A. Amin, "Neural network and regression based processor load prediction for efficient scaling of grid and cloud resources," in *14th International Conference on Computer and Information Technology (ICCIT)*, IEEE, 2011.

[5] J. Hopcroft and R. Karp, "An n. 5/2 algorithm for maximum matchings in bipartite graphs," *SIAM Journal on Computing*, vol. 2, pp. 225–231, 1973.

[6] P. K. Janert, *Data Analysis with Open Source Tools - a Hands-on Guide for Programmers and Data Scientists*. O Reilly, 2011.