2013 International Conference on Computer Communication and Informatics (*ICCCI* -2013), Jan. 04– 06, 2013, Coimbatore, INDIA

Meta-Heuristic Hybrid dynamic task scheduling in heterogeneous Computing environment

R.Leena Sri
Assistant Professor,Dept. of Information Technology
Thiagarajar College of Engineering
Madurai
rlsit@tce.edu

Dr.N.Balaji
Prof. & Head, Dept. of Information Technology
K.L.N.College of Engineering
Madurai
baljin@klnce.edu

Abstract--***System State estimation and decision making are the two major components of dynamic task scheduling in a distributed computing system. Heuristic and meta-heuristic approaches seem to be the most effective methods of scheduling in Heterogeneous computing due to their ability of relative fast generation of high quality solutions. Most of the available Meta heuristic algorithms attempt to find an optimal solution with respect to a specific fixed fitness measure. The major challenges when using Genetic Algorithms to solve dynamic optimization problems are: (a) to generate and keep the diversity in the populations, which is crucial for avoiding the premature convergence to the local optima and (b) to evolve robust solutions that are able to track the optima. All of these issues will necessitate the development of intelligent adaptive algorithms that can dynamically adapt to the changes in the large-scale Computing Groups. We propose a Hybrid Genetic and Case based reasoning algorithm*** HGAC ***to improve the make span by predicting the performance of online resources to better converging the local optima and improve decision faster in dynamic environment.***

Keywords: Heuristic and Meta heuristic, Dynamic optimization problems, Hybrid Genetic and Case based reasoning

I. Introduction

Task scheduling is a major challenge in parallel and distributed systems. Task scheduling techniques in distributed systems are usually based on trusting the accuracy of the information about the status of resources (e.g. their load). This information is usually submitted by resource providers to a central database which is accessible to schedulers.

Multiprocessor Scheduling is an NP-hard problem [2, 4, and 5]. The problem of scheduling a set of dependent or independent tasks in a distributed computing system is a well-advanced area. In this paper, the dynamic task allocation methodology is examined in a heterogeneous computing environment. Dynamic allocation techniques can be applied to large sets of real-world applications that can be formulated in a manner that allows for deterministic execution[9],[20]. Some advantages of the dynamic techniques over static techniques are that, static techniques should always have a prior knowledge of all the tasks to be executed but dynamic techniques do not require that. The traditional methods such as branch and bound, divide and conquer, and dynamic programming gives the global optimum, but is often time consuming or does not apply for solving typical real-world problems. The researchers [6] have derived optimal task assignments to minimize the sum of task execution and communication costs with the branch-and bound method and evaluated the computational complexity of this method using simulation techniques.

The resource performance dynamism is the major issues in heterogeneous environment. The dynamic changing of resource performance mainly comes from the autonomy of the sites, and the contention incurred by resource sharing among a potentially large number of users. Usually the performance dynamism is resulted from completion among jobs sharing the same resource. This problem could be reconciled by considering the possibility of conflict when the scheduling decision is made. Current Grid scheduling systems adopt three levels of techniques to relieve the dynamic problem: scheduling based on just-in-time information from Grid information services (e.g., dynamic algorithms for load balancing), performance prediction, and dynamic rescheduling at run time.

We provided a comprehensive analysis of the efficiency of HGAC to predict better performance knowledge prior to the task scheduling whereas the current scheduling algorithms only consider a snapshot value of the prediction when they make the estimate, and assume that value is static during the job execution period.

In this article, we propose a hybrid approach using genetic algorithms (GAs) as an alternative methodology to assign the importance of attributes for case-based retrieving. We use GAs to extract knowledge that can guide effective retrievals of useful cases contains job information, machine information, and job run time . Our specific interest lies in the assignment of importance values to each dimension of case features in the problem of dynamic task scheduling. This article is organized as follows. Section II provides a brief description of GAs. Section III describes the characteristics of indexing and retrieving methods of CBR. Section IV describes our hybrid approach with GAs and CBR. Finally, Section VI discusses the conclusions.

II. Genetic algorithms

GA is an evolutionary technique for large space search. .GAs is particularly suitable for multi-parameter optimization problems with an objective function subject to numerous hard and soft constraints. GAs performs the search process in four stages: initialization, selection, crossover, and mutation (Davis, 1991; Wong and Tan, 1994). Fig. 1 shows the basic steps of GAs.

The general procedure of GA search is as follows [7]:

1. Population generation: A population is a set of *chromosomes* and each represents a possible solution, which is a mapping sequence between tasks and machines. The initial population can be generated by other heuristic algorithms, such as Min-min (called seeding the population with a Min-min chromosome).
2. Chromosome evaluation: Each chromosome is associated with a fitness value, which is the makespan of the task-machine mapping this chromosome represents. The goal of GA search is to find the chromosome with optimal fitness value.
3. Crossover and Mutation operation: Crossover operation selects a random pair of chromosomes and chooses a random point in the first chromosome. For the sections of both chromosomes from that point to the end of each chromosome, crossover exchanges machine assignments between corresponding tasks. Mutation randomly selects a chromosome, then randomly selects a task within the chromosome, and randomly reassigns it to a new machine.
4. Finally, the chromosomes from this modified population are evaluated again. This completes one iteration of the GA. The GA stops when a predefined number of evolutions are reached or all chromosomes converge to the same mapping.

This genetic algorithm randomly selects chromosomes. Crossover is the process of swapping certain sub-sequences in the selected chromosomes.

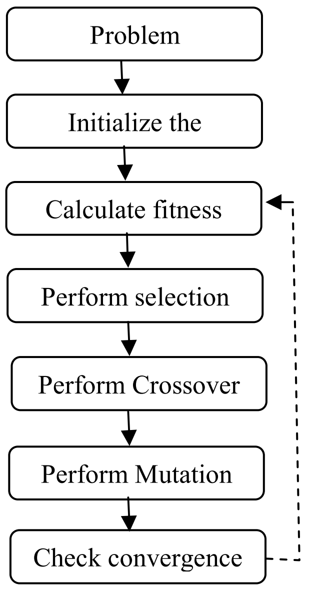

Figure 1.Basic steps of GA

Mutation is the process of replacing certain sub-sequences with some task-mapping choices new to the current population. Both crossover and mutation are done randomly. After crossover and mutation, a new population is generated. Then this new population is evaluated, and the process starts over again until some stopping criteria are met. The stopping criteria can be, for example, 1) no improvement in recent evaluations; 2) all chromosomes converge to the same mapping; 3) a cost bound is met. For its simplicity, GA is the most popular Nature's heuristic used in algorithms for optimization problems. In the realm of Grid scheduling, we can find other examples in [1], [3], [11], [15], [18] [19] and [20].

III. Case Based Reasoning

Case-based reasoning [10] is a problem solving paradigm that in many respects is fundamentally different from other major AI approaches. Instead of relying solely on general knowledge of a problem domain, or making associations along generalized relationships between problem descriptors and conclusions, CBR is able to utilize the *specific* knowledge of previously experienced, concrete problem situations (cases). A new problem is solved by finding a similar past case, and reusing it in the new problem situation. A second important difference is that CBR also is an approach to incremental, sustained learning, since a new experience is retained each time a problem has been solved, making it immediately available for future problems. The CBR field has grown rapidly over the last few years, as seen by its increased share of papers at major conferences, available commercial tools, and successful applications in daily use.

To use the CBR approach, we first need to design a case base, which is a repository of historical data. Here, a case is a record that contains job information, machine information, and job run time. Job and machine information is used to match similar cases, and job run times are used to calculate the estimated run time. The similarity of cases depends on both the similarity of jobs and similarity of machines.

Since a job and a machine may have many characteristics, we must decide which characteristics significantly affect the similarity of jobs and similarity of machines, and thus should be included in the case representation. In addition, as job and machine characteristics are prioritized in order to search similar cases effectively, we need to determine the priority of each characteristic. For the priorities of job characteristics, k-nearest neighbor algorithm is used to check the runtime standard deviation of each characteristic [8].A characteristic with a low standard deviation has a high priority. For the priorities of machine characteristics, change the value of each machine characteristics and keep the values of other characteristics unchanged, and check how the run time is affected. A characteristic with a high runtime change has a high priority [8].

IV. Hybrid systems with genetic algorithms and case base reasoning (HGAC)

Integration of domain knowledge into case indexing and retrieving process is important in building a useful case based system. The central idea of combination of GAs and case-based system is that CBR transfers the burden of knowledge assignment of the indexing and retrieving process to the searching and learning capabilities of evolutionary algorithms. In this subsection, we propose a hybrid approach using GAs to case indexing and retrieving process in an attempt to increase the overall effectiveness of the dynamic resource system [14].

A. Case indexing

In developing a CBR system, the choice of indexing and retrieving method is a very important step. As described in Section III, prior studies suggest some guidelines on choosing an indexing technique. Barletta (1991) and Buta (1994) state that inductive indexing is more appropriate when the retrieval goal is well defined while nearest-neighbor is preferred when the retrieval goal is subjective. Grid scheduling systems operate in dynamic environments subject to various unforeseen and unplanned events that can happen at short notice. In this aspect, inductive approaches are suitable for finding out the performance of dynamic schedule, such as breakdown of computers, arrival of new jobs, processing times are subject to stochastic variations, etc.

However, inductive indexing requires a large number of cases, case is a record that contains job information, machine information, and job run time, to form induction trees (Barletta, 1991). In addition, finding and maintaining an optimal induction tree for case based retrieval is an expensive task (Kolodner, 1993).We employ the nearest-neighbor approach for the current study. The nearest-neighbor approach, however, has difficulty in deciding a set of feature weights that could accurately retrieve cases in a given situation (Kolodner, 1993, Aamodt and Plaza, 1994, Leake, 1996).

B. Nearest-neighbor matching

Kolodner (1993) states that the importance of a dimension in judging similarity and degree of match should be considered in building matching functions. Matching and ranking is the process of comparing two jobs with each other and determining their degree of match and ordering resource according to the goodness of match or the usefulness for the application (Kolodner, 1993). A good matching function takes into account which features of a resource are more important and scores resource for the usefulness according to the scheduling. A resource that matches important features but not less important ones needs to be judged as a better match than one that matches less important features but does not match important ones.

One of the most obvious measures of similarity between two jobs is the distance. This study uses a numeric evaluation function measuring the distance taking into account of importance features to compute the degree of match in retrieval. A matching function of the nearest-neighbor method is as follows [14]:

$$DIS_{ab} = \sqrt{\sum_{i=1}^{n} w_i \times (f_{ai} - f_{bi})^2}, \tag{1}$$

where DIS is the matching function using Euclidean distance between resources, *n* the number of jobs, and w_i the importance weighting of job *i*. Basic steps of nearest neighbor retrieval algorithms are quite simple and straightforward. Every feature in the input case is matched to its corresponding feature in the stored case, and the degree of match of each pair is computed using the matching function. Based on the importance assigned to each dimension, an aggregate match score is computed.

C. Assigning importance values to case attributes

Another way to assign importance values is to do a statistical evaluation of known cases to determine which dimensions predict the solutions best. The more significant predictors in the statistical evaluations are considered to be of importance for matching. For example, the magnitude of the correlation coefficient between each input and the output in the reference set can be used to weigh each input when computing the distance measure for a new example.

As an alternative approach, we introduce the notion of machine learning to learn the optimal weights from historical cases using evolutionary search technique. By evaluating the fitness of different weight vectors, we may find good solutions for the system. As described in Section II, GAs apply crossover and mutation to generate a new population of problem solutions and select the best solution for the problem.

D. Hybrid structure of GA–CBR system

The overall structure of hybrid approach is shown in Figure 2.

Phase 1: For the first step, we search an optimal or near optimal weight vector with precedent cases for which the prediction outcome is determined. In setting up the genetic optimization problem [8], we need:

1. The parameters that have to be coded for the problem.
2. An objective or fitness function to evaluate the performance of each job.
3. The ranges for the adjustable parameters.
4. Potential constraints to be met.

The parameters that are coded are the weight vectors for nearest-neighbor matching. These weight vectors assign importance values to each feature and are inserted into the matching formulae. The range of the weights between 0 and 1 and do not apply particular constraints for this search.

The task of defining a fitness function is always application specific. In this case, the objective of the system is to retrieve more relevant cases that can lead to the correct solutions. The ability of case-based systems to achieve these objectives can be represented by the fitness function that specifies how well the matching function increases the prediction accuracy. We apply the prediction accuracy rate of the test set to the fitness function for this study. The test set consists of known cases of which the prediction outcome was determined and used to evaluate the fitness of different sets of job weights. Mathematically, this fitness function is expressed as:

$$\max \quad PR = \frac{1}{n}\sum_{i=1}^{n} PA_i \tag{1}$$

$$\text{s.t}\, PA_i = 1 \quad if\ O(T_i) = O\big(S_{j*(i)}\big), \tag{2}$$

$$PA_i = 0 \quad otherwise, \tag{3}$$

$$S_{j*(i)} = \min_{j\in R}\Big(\sqrt{\sum_{k=1}^{l} w_k (T_{ik} - R_{jk})^2}\ \Big), \tag{4}$$

$$for\ a\ given\ i\ (i = 1,2,\ldots\ldots,n),$$

where PR is the Prediction accuracy rate of the test set; PA*i* the prediction accuracy of the *i*th case of the test set denoted by 1 and 0 (‘correct’ =1, ‘incorrect’ =0). For example, the bond rating of the *i*th case of the test set and the closest case in the reference set are ‘A1’ and ‘A1’, respectively, PA*i* is 1, otherwise 0); *O*(*Ti*) the target output of *i*th case of the test set; *O*(*Sj*(i)*) the output of *j*th case of the reference set that has the minimum distance with the *i*th case of the test set; *Sj(i)* the distance between the *i*th case of the test set and the *j*th case in the reference set; T_{ik} the *k*th feature of the *i*th case of the test set (*T*); R_{jk} the *k*th feature of the *j*th case of the reference set (*R*); w_k the importance weight of the *k*th feature of case; *l* the number of features and *n* the number of test cases. For the controlling parameters for experiment, we should also specify:

1. population size;
2. crossover rate;
3. mutation rate;
4. the criteria for stopping the process.

There has been much debate regarding the optimal population size for the tasks and the machines. Generally, the population size is determined according to the size of the tasks and machines (a bigger population for a larger problem) occurrence during runtime.

Phase 2: The second step is to apply the derived weight vector in *phase 1* to the case indexing scheme for the case based retrieval process and evaluate the resulting model with additional validation cases for which the outcome is also known. A weight vector is used in the nearest-neighbor matching function to rank and retrieve useful cases. As the validation cases are not used for parameter optimization, the prediction performance tested by these cases would be the closest to the current or future cases.[8]

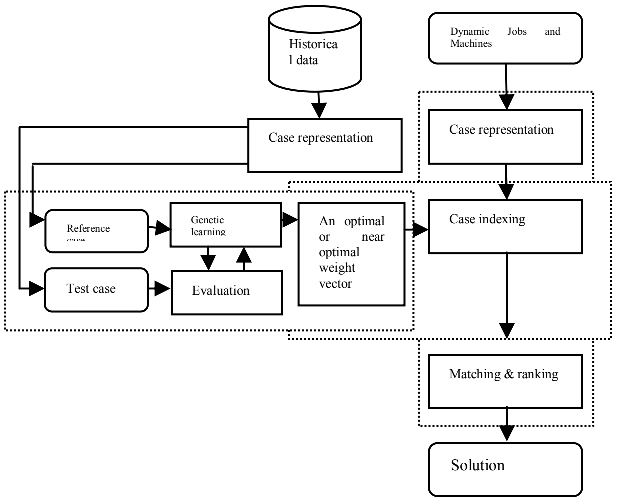

Figure.2. Hybrid structure of GA and CBR

V. Results and Implementation

We propose a pseudo code for our algorithm. We wish to show that using multiple heuristics to generate schedules for the initial population of the GA provides more efficient schedules than using each individual heuristic on its own, or using a purely random initial population. The remainder of the population is generated based on knowledge assignment of best case indexing using random permutations of these heuristics. The use of multiple heuristics in our initial population provides the GA with reasonable starting solutions, compared to starting with a completely randomly generated initial population.

By employing nearest neighbor matching, the GA will always produce a solution which is equal to the best heuristic solution in the initial population, because the best/fittest solution is always brought forward to the next generation.

```
Input: Set of tasks and processors
Output: Efficient Mapping of tasks to processors
foreach heuristic do
        generate schedule ;
end
while population not full
        copy and mutate min-min heuristic
        schedules ;
end
repeat
        cycle crossover ;
        random mutations ;
        rebalance ;
        roulette wheel selection ;
        Case indexing
        Nearest neighbor matching
        save best schedule (elitism) ;
        update mutation rate ;
until stopping conditions met;
Assigning case attributes for future reference
end
return schedule with shortest makespan
```

Algorithm 1: Pseudocode for HGAC

The implementation of HGAC pseudo code in a java based distributed system is in progress. A general purpose programmable Java distributed system, which utilizes the free resources of a heterogeneous set of computers linked together by a network, has been developed [12].The system has been successfully deployed on over 500 computers, which were distributed over a number of locations, and has been successfully used to process bioinformatics [13], biomedical engineering [16], and cryptography applications.

The distributed system consists of 3 Java archive files, a client, a server and a remote interface. A problem can be created for the system simply by extending 2 classes, called Algorithm and DataManager. The Algorithm class is run on the client and specifies the actual computation to be performed. The DataManager class is run on the server and specifies how the problem is broken up into tasks and how the processed results are recombined.The distributed system provides a simple scheduling interface,which allows the administrator of the system to select a scheduling algorithm using the remote interface. To create a new scheduler, a programmer only needs to extend the SchedulerCommon API and implement a single method called generateSchedule. This method simply takes in a list of tasks and maps them to processors. The system defaults to the simplest scheduler, round robin.The main objective of our algorithm is to improve makespan with 0% inefficiency.

VI. Conclusion

Both CBR and GAs are artificial intelligent approaches that have received significant attention in recent years. In this article, we have proposed a hybrid approach using CBR and GAs to the problem of dynamic task scheduling. In this approach, the genetic search technique is used to assign relative importance of feature weights for case indexing and retrieving. We have inferred that this approach supports effective retrieval of cases and increases overall prediction accuracy rate significantly.

We conclude with the two findings. First, the knowledge acquired by performance of jobs supports the retrieval of usefully similar cases to solve the problem. As the task of GAs to define a fitness function is always applications specific, this hybrid approach utilizes prediction accuracy and robust optimal task scheduling and case specific knowledge simultaneously. Second, GAs are an effective method of knowledge extraction for case based retrieval. Using GAs, we can obtain the near optimal weights representing the importance of each feature.

REFERENCES

[1] M. Aggarwal, R.D. Kent and A. Ngom, *Genetic Algorithm Based Scheduler forComputational Grids*, in Proc. of the 19th Annual International Symposium on High Performance Computing Systems and Applications (HPCS'05), pp.209-215, Guelph, Ontario Canada, May 2005

[2] W. Allcock, J. Bresnahan, R. Kettimuthu, M. Link, C. Dumitrescu, I. Raicu, I. Foster,*The Globus Striped GridFTP Framework and Server*, in Proc. of the 2005 ACM/IEEE conference on Supercomputing, pp.54-64, Seattle, Washington USA, November 2005.

[3] Andrew J. Page,Thomas M. Keane, Thomas J. Naughton,Multi-heuristic dynamic task allocation using genetic algorithms in a heterogeneous distributed system, J. Parallel Distrib. Comput. 70 (2010) 758_766

[4] F. Berman, *High-Performance Schedulers*, chapter in The Grid: Blueprint for a Future Computing Infrastructure, edited by I. Foster and C. Kesselman, Morgan Kaufmann Publishers, 1998.

[5] F. Berman, R. Wolski, H. Casanova, W. Cirne, H. Dail, M. Faerman, S. Figueira, J. Hayes, G. Obertelli, J. Schopf, G. Shao, S. Smallen, N. Spring, A. Su and D. Zagorodnov, *Adaptive Computing on the Grid Using AppLeS*, in IEEE Trans. On Parallel and Distributed Systems (TPDS), Vol.14, No.4, pp.369--382, 2003.

[6] F. Berman, *High-Performance Schedulers*, chapter in The Grid: Blueprint for a FutureComputing Infrastructure, Morgan Kaufmann Publishers, 1998.

[7] R. Braun, H. Siegel, N. Beck, L. Boloni, M. Maheswaran, A. Reuther, J. Robertson,M. Theys, B. Yao, D. Hensgen and R. Freund, *A Comparison of Eleven StaticHeuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems*, in J. of Parallel and Distributed Computing, vol.61, No. 6, pp.810-837, 2001.

[8] Edward Xia,Igor Jurisica,Julie Waterhouse, and Valerie Sloan,Runtime Estimation Using the Case-Based Reasoning Approach for Scheduling in a Grid Environment, IEEE Transaction on Parallel and Distributed Systems (TPDS), 2010

[9] Hamid Mohammadi Fard, Radu Prodan and Thomas Fahringer,A Truthful Dynamic Workflow Scheduling Mechanism for Commercial Multi-Cloud Environments, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS Digital Object Indentifier 10.1109/TPDS.2012.257,June 20,2012

[10] Janet L. Kolodner**,** An Introduction to Case-Based Reasoning, *Artificial Intelligence Review* 6, 3--34, 1992

[11] S. Kim and J. B. Weissman, *A Genetic Algorithm Based Approach for Scheduling Decomposable Data Grid Applications*, in Proc. of the 2004 International Conference on Parallel Processing (ICPP'04), pp. 406-413, Montreal, Quebec Canada, August 2004.

[12] T. Keane, R. Allen, T.J. Naughton, J. McInerney, J. Waldron, Distributed Java platform with programmable MIMD capabilities, in: N. Guelfi, E. Astesiano, G. Reggio (Eds.), Scientific Engineering for Distributed Java Applications,in: Springer Lecture Notes in Computer Science, vol. 2604, 2003, pp. 122–131.

[13] T.M. Keane, T.J. Naughton, S.A.A. Travers, J.O. McInerney, G.P. McCormack, DPRml: distributed phylogeny reconstruction by maximum likelihood,Bioinformatics 21 (7) (2005) 969–974.

[14] Kyung-shik Shin, Ingoo Han ,Case-based reasoning supported by genetic algorithms for corporate bond rating, *Expert Systems with Applications* Elsevier *16 (1999),* pp.85–95, PII: S0957-4174(98)00063-3

[15] Lei Wang a, Dun-bing Tang b, An improved adaptive genetic algorithm based on hormone modulation mechanism for job-shop scheduling problem, Expert Systems with Applications 38 (2011)

[16] A.J. Page, S. Coyle, T.M. Keane, T.J. Naughton, C. Markham, T. Ward, Distributed monte carlo simulation of light transportation in tissue, in: Proceedings of the 20th IEEE International Parallel and Distributed Processing Symposium, IEEE Computer Society, Rhodes, Greece, 2006, pp. 1–4.

[17] Sasmita Kumari Nayak, Sasmita Kumari Padhy, Siba Prasada Panigrahi, A novel algorithm for dynamic task scheduling, Future Generation Computer Systems 28,Elsevier (2012) 709–717

[18] D.P. Spooner, J. Cao, J.D. Turner, H. N. L. C. Keung, S.A. Jarvis and G.R. Nudd, *Localised Workload Management using Performance Prediction and QoS Contracts*,in the Proc. of the 18th Annual UK Performance Engineering Workshop (UKPEW' 2002), University of Glasgow, UK, July 2002.

[19] S. Song, Y. Kwok, and K. Hwang, *Security-Driven Heuristics and A Fast Genetic Algorithm for Trusted Grid Job Scheduling*, in Proc. of 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05), pp.65-74, Denver,Colorado USA, April 2005

[20] M. Wieczorek, R. Prodan and T. Fahringer, *Scheduling of Scientific Workflows in the ASKALON Grid Environment*, in ACM SIGMOD Record, Vol.34, No.3, pp.56-62, September 2005.