Improving CBR-LA algorithm to variable size problems

S. Sabamoniri
Islamic Azad University, Soufian Branch, Iran
Saba_Moniry@Hotmail.com

B. Masoumi
Islamic Azad University, Qazvin Branch, Iran
bmasoumi@Qazviniau.ir

M. R. Meybodi
Amirkabir University of Technology, Department of Computer Engineering and IT, Tehran, Iran
mmeybodi@aut.ac.ir

***Abstract*- In this paper an improved approach based on CBR-LA model is proposed for static task assignment in heterogeneous computing systems. The proposed model is composed of case based reasoning (CBR) and learning automata (LA) techniques. The LA is used as an adaptation mechanism that adapts previously experienced cases to the problem which must be solved (new case). The goal of this paper is to expressing some weak points of the CBR-LA and proposing new algorithm called ICBR-LA which has improved performance in terms of Makespan performance metric. The results of experiments have shown that the proposed model performs better than the previous one.**

I. Introduction

Mixed-machine heterogeneous computing (HC) systems are used by a sequence of distributed high performance machines in order to being able to execute application programs that have different computing requirements. Assigning a set of tasks (Metatask) to the machines included in HC and scheduling them for executing on those machines is called *mapping*. The general problem of optimized mapping of tasks to machines in a HC sequence is known as a NP-complete problem [1]. The Metatask is defined as a set of independent tasks and there isn't any dependency among them [2]. The mapping can be done in a static or dynamic way. In the static mapping it is assumed that every machine executes one task at a time in the order which they had assigned to the machine. The goal of this mapping is to minimize the total execution time of the metatask, which is referred to as ***Makespan***. The size of Metatask and number of machines in HC are static and previously known. Some approaches for solving this problem include: graph theory based algorithms, simulated-annealing [3], MET, MCT, Min-Min, Max-Min, A* [4], GA [5,6], and LA [7] which all of them solve the problem without using the previous experiences. Also, a hybrid model so-called CBR-LA (consisting of LA and CBR techniques)_ proposed in [8] that its main goal was to reduce iteration number of LA approach and provide rapid access to the solution was based on using previous experiences. This model is used for recommending a quick mapping of tasks to a set of processors that exist in an HC environment. The goal of this paper is to evaluate the CBR-LA recommendations in terms of Makespan performance metric. So, this study intends to show that the CBR-LA solutions are not as optimal as other approaches such as Max-Min and Min-Min. To achieve this goal, some experiments were done and some modifications to the LA model of CBR-LA were proposed. The new proposed model (called ICBR-LA) is able to generate optimized mappings in contrast with the original CBR-LA.

In section II, the CBR-LA algorithm together with its CBR and LA models are introduced and in section III evaluation results about the CBR-LA are presented. Section IV presents experiments and their results about CBR-LA functionality against dynamic size problems. The proposed LA model which improves the CBR-LA will be described in section V. Section VI, proposes a modification about changing environment response model and its effect on ICBR-LA algorithm. Sections VII and VIII are respectively about the proposed algorithm's performance in comparison with other algorithms and the conclusions of this study.

II. CBR-la Algorithm

Case-based reasoning (CBR) is a knowledge-based problem-solving technique, which is based on reusing of previous experiences. The CBR-LA algorithm is a model for static task assignment in heterogeneous computing systems and proposed in [8]. The proposed model is a combination of the CBR and LA models. The LA model is used as an adaptation mechanism that adapts previous experiences to the new problem. The aim of proposing of the model was to reduce the number of required iterations for finding a semi-optimum solution.

In this section the CBR-LA is briefly introduced and in the next sections it is shown that the solutions presented by the algorithm are quite different from those presented earlier. Then, a new model for improving the results is proposed.

A. The LA Model of CBR-LA

Learning Automata (LA) are adaptive decision-making devices operating on unknown random environments. The LA has a finite set of actions and each action has a certain probability (unknown for the automaton) rewarded by the environment of the automaton. Learning Automata can be classified into two main families: fixed structure and variable structure learning automata (VSLA). A VSLA is a quintuple <α, β, p, T(α,β,p) >, where α, β, p are an action set with s actions, an environment response set and the probability set p containing s probabilities, each being the probability of performing every action in the current internal automaton state, respectively. The function of T is the reinforcement algorithm, which modifies the action probability vector p with respect to the performed action and received β set responses. A general linear schema for updating action probabilities uses reward (a) and penalty (b) parameters which the amount of their values classify automata. That is, when a=b, the automaton is called L_{RP}. If $0<a<<b<1$ the automaton is called $L_{P\varepsilon R}$, and if $0<b<<a<1$, the automaton is called $L_{R\varepsilon P}$ [8]. For more Information about LA the reader may refer to [9,10].

As mentioned earlier, LA is used as the case adaptation mechanism in the CBR-LA. Fig. 1 shows the schematic of the LA model. The model is constructed by associating every task s_i, which $1\le i\le\tau$, in the metatask with a VSLA as ($\alpha(i)$, $\beta(i)$, $A(i)$). Since the tasks can be assigned to any of the μ machines, the action set of all LA are identical. So, there is $\alpha(i)=m_1,m_2,\ldots,m_\mu$ for each task s_i, $1\le i\le\tau$. It is assumed that the environment is a P-model so the input set for each learning automaton A(i) is {0,1}.When $\beta(i)$ equals to 0 it indicates a favorable response, and when equals to 1 it indicates an unfavorable response. The general procedure for LA model is shown in fig. 2.

B. The Case Based Reasoning Model

CBR technique is a knowledge-based approach for solving problems which operates on reusing previous experiences and is emerged from cognitive science research [11, 12]. This approach assumes that similar problems could have similar solutions. Thus, the new problem might be solvable using the experienced solutions in previous similar problems. CBR collects its experiences into a case base (CB). Each experience of the CB is called *case*. A case consists of three parts: problem situation including the initial conditions and the goal, the solution and the performance [8]. The three parts of a case in CBR-LA model are ETC, Mapping and Makespan.

Main activities of problem solving by this approach that are described in CBR cycle have the following four steps: retrieve, reuse, revise, and retain. To solve the new problem, firstly, it must be formally described as a '*new case*'. Secondly, a case that is similar to this new case is retrieved from the CB. Thirdly, the solution in the retrieved case is reused (adapted) to solve the new problem. Finally, a new solution is obtained and presented to the user, who can verify and possibly revise the solution. During the last phase, a new solution is obtained which could be stored as a new experience in CB (this step realizes the learning phase of a CBR application). It should be noted that CBR does not offer a definite solution but proposes hypotheses and ideas for crossing the solution space [8]. Similarity criterion in retrieve phase is defined as similarity between two matrixes: ETC of the new case and ETCs of the stored cases, and is evaluated as the Euclidian distance of the two matrixes [8]. See [11, 13] for more information.

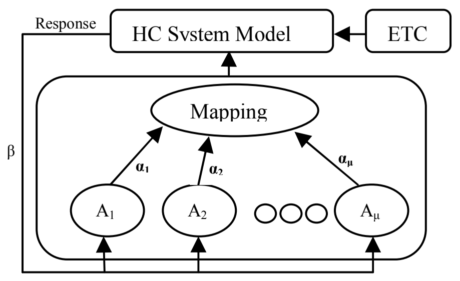

Fig. 1. Learning automata model [8].

```
while(true)
Begin
    LA_i selects action for all 1 ≤ i ≤ τ
    Evaluate the makespan
    If  T_μ (n)<T_μ (n-1) then LA_i.Signal(0) for all 1 ≤ i
        ≤ τ
    Else LA_i.Signal(1) for 1 ≤ i ≤ τ
If no change in makespan occurs for 150 consecutive
iterations or 10000 iterations is over then exit
End
```

Fig. 2. General procedure for learning automata model [8]

C. Describing Adaptation Phase of CBR-LA

After retrieving a set of similar cases, adaptation phase starts. Adaptation is the process of transforming the mappings of retrieved cases for finding a mapping for the new case. This process is done by means of the LA model. Let Φ be the set of selected cases. To build up the adaptation model for each case φ_k, $1\le k\le|\Phi|$, every task s_i is associated with a VSLA $A^{\varphi}_k(i)$ and biased to the mapped machine m_j. To do this the automaton action probability has been initialized about *one* for corresponding to *mj* action. For a biased automaton, the rate of penalizing is set to a value much greater than the rate of rewarding. It helps a wrongly biased automaton to correct itself rapidly. LA model for each case φ_k starts iterating as explained in II-A. Mappings derived from Φ are compared with each other and the one with the minimum Makespan is selected as the final solution [8].

D. Conditions of Simulation

For the simulation studies, characteristics of the ETC matrices were varied in an attempt to represent a range of possible HC environments. The ETC matrices used were generated using the method introduced in [8, 14]. To generate different mapping scenarios, the characteristics of the ETC matrix were varied based on several different methods from [2]. The amount of variance among the execution times of tasks in the metatask for a given machine is defined as task heterogeneity. Machine heterogeneity represents the variation that is possible among the execution times for a given task across all the machines. To further vary the characteristics of the ETC matrices different ETC matrix consistencies were used. An ETC matrix is said to be consistent if a machine m_j executes any task s_i faster than machine m_k, then machine m_j executes all tasks faster than machine m_k [8, 14]. In contrast, inconsistent matrices characterize the situation where machine m_j may be faster than machine m_k for some tasks and slower for others. Eight different classes of ETC matrix characteristics are used in this paper: high (*Hi*) or low (*Lo*) task heterogeneity, high or low machine heterogeneity, and one type of consistencies; consistent (*cons.*) or inconsistent (*incons.*). Abbreviations are used for these kinds of heterogeneities such as HiHi-incons standing for high task heterogeneity, high machine heterogeneity and inconsistent environment. The experiments which were done in [8] were considered as a fixed size ETC matrices for all new cases but in this study, dynamic size of them are used to extend the algorithms power in solving different types of problems.

III. Evaluating CBR-LA In Comparision With Max-min

In this section, the CBR-LA's performance from various aspects is evaluated. To achieve this, some equal experimental conditions as input to the CBR-LA algorithm along with Min-Min and Max-Min algorithms are used. In these experiments 160 new cases from 8 classes of heterogeneity were produced. In this section, the results are average of 10 executions. As it was made clear in [14] and our experiments in section VII, Min-Min algorithm acted better than Max-Min, so the results of it were not included in this part.

A. *Comparison with Max-Min*

Just like [8] 512 tasks and 16 machines for ETCs were used and the CB with 100 random cases were produced. And, the numbers of selected similar cases (Φ) were 3. As a result of equality of the ETC dimensions, all of automata will be biased. Due to time restriction, only the results of the HiHi-incons class were compared with max-min approach (shown in table1). As illustrated in table 1 CBR-LA results are about 17 times greater than Max-Min's results in terms of Makespan.

B. *Evaluating CBR-LA with nonrandom CB*

In this experiment we'll evaluate the capability of CBR-LA in using its CB experiences. In order to do this we had changed the CB of CBR-LA, it means that we used 100 nonrandom cases as experiences for the algorithm. To produce the CB we used Min-Min algorithm. As illustrated in table 2, in spite of better experiences, the CBR-LA still have a great difference with results of Max-Min.

C. *Evaluating CBR-LA by changing the number of iterations in LA model*

In this experiment the amount of iteration numbers for LA model were increased from 10000 to 50000, to let the automata search more in response spaces. According to table 3 and fig. 3, the results are the same as the previous experiments.

D. *Evaluating the CBR-LA by changing the size of CB*

In this experiment, the numbers of cases in CB were increased from 100 to 200. But as illustrated in table 4, the results are not so near to optimal.

As mentioned in section II, the main advantage of CBR-LA is its reduced number of iterations. But according to the result aspect of the CBR-LA (in terms of Makespan), this algorithm's recommendations are not so near to optimal in comparison with previously proposed algorithms such as Max-Min. Thus, in subsequent sections the LA model of this algorithm is modified to achieve improved results.

TABLE 1
COMPARISON BETWEEN RESULTS OF CBR-LA AND MAX-MIN ALGORITHMS IN HIHI-INCONS CLASS OF PROBLEMS

CBR-LA (with 3 selected Similar Case & random CB)		Max-Min
Makespan mean	31691687	1862188
amount of difference	17 times greater	-

TABLE 2
COMPARISON BETWEEN RESULTS OF CBR-LA (NONRANDOM CB) AND MAX-MIN ALGORITHMS IN HIHI-INCONS CLASS OF PROBLEMS

	CBR-LA (random CB)	**CBR-LA (nonrandom CB)**	**Max-Min**
Makespan mean	31691687	32843660	1862188
amount of difference	17 times greater	17.64 times greater	-

TABLE 3
RESULTS OF COMPARISON OF DIFFERENT NUMBERS OF AUTOMATA ITERATION

CBR-LA (Nonrandom CB)			**Max-Min**
Mean results by 10000 iteration		Mean results by 50000 iteration	Mean results
26132244		25604685	1862188
Amount of difference	14.03 times greater	13.75 times greater	-

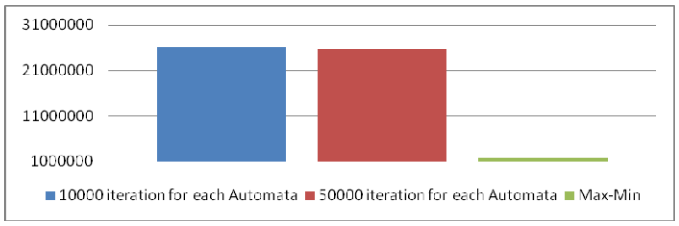

Fig. 3. Comparison of the results of different iteration numbers and max-min heuristic

TABLE 4
RESULTS OF INCREASING THE CB CASES IN CBR-LA

CBR-LA (Nonrandom CB)			**Max-Min**
Mean results by 100 experienced cases		Mean results by 200 experienced cases	Mean results
23483015		25492513	1862188
Amount of difference	12.61 times greater	13.69 times greater	-

IV. CBR-LA AGAINST DYNAMIC SIZE NEW CASES

As we described CBR model of CBR-LA in section II, in all experiments of [8] the ETC matrices had a fixed number of tasks and machines which it caused to biasing all of the automata. But depending on the number of tasks and machines there are in HC, the size of ETC matrix may vary in each new case. So, the algorithm was evaluated against new cases which had *dynamic size ETCs*. For creating the new cases it is assumed that task numbers are vary from 100 to 150 and the machine numbers are vary from 4 to 8 machines for ETC matrices. In the experiments of this section we'll use different reward and penalty values for biased and not biased automata. Also, the results presented in this section are average of 20 executions. Also, in subsection IV-B the improvement process of results by increasing the number of selected similar cases in CBR-LA evaluated.

A. *Reward and penalty values for biased and unbiased automata to support dynamic size new cases*

First of all it is better to examine the CBR-LA with the dynamic size problems. According to the LA model presented in II-A, $L_{P\varepsilon R}$ automata have a great penalty values in contrast with the reward. In order to found a meaningful amount for the penalty and rewards, two experiments were done and the parameters of tables 5 and 6 were selected to do the experiments (the original amounts in [8] were used as reward and penalty for the unbiased automata). Table 5 was used 10 times greater and table 6 was used 30 times greater penalties in contrast with rewards. According to the table7, the results show better Makespan values when the amounts of table 5 were used. From here on, the table 5 values are used.

B. Evaluating the CBR-LA by selecting different numbers of similar cases

In this experiment, improvement process of results was examined by increasing the numbers of selected similar cases in CBR-LA. This feature was claimed in [8] but we will show that this improvement process is not so great and the algorithm still have a great difference by results that other algorithms produces.

In this experiment we performed the CBR-LA by 3, 5, 7 and 10 selected similar cases. The improvement process in results was shown in table 8 and fig. 4

As fig. 4, when the algorithm uses 10 experiences from CB, the results have about 8% improvements rather than when it uses 3 experiences. Then as expressed in [8] the CBR-LA produces better results if the number of experiences increases. But in case of using 10 experiences, the CBR-LA has 73% difference with Max-Min from results aspect.

V. The Proposed LA Model In Oreder To Improve CBR-LA

By evaluating the LA model we found out that in spite of better experiences in CB and biasing automata to these experiences, the CBR-LA can't benefit the experiences. In order to have improvements in results, it is decided to do some changes in the LA model.

TABLE 5
Penalty Is 10 Times Greater Than Rewards For Biased Automata

	Reward	Penalty
Biased Automata	0.01	0.1
Unbiased Automata	0.5	0.1

TABLE 6
Penalty Is 30 Times Greater Than Rewards For Biased Automata

	Reward	Penalty
Biased Automata	0.01	0.3
Unbiased Automata	0.5	0.1

TABLE 7
Results Achived By Different Reward And Penalties Used In Biased Automata

	CBR-LA (CB with 100 Random cases)		**Max-Min's Makespan**
	Penalty is 10 times greater than rewards for biased automata	Penalty is 30 times greater than rewards for biased automata	
Mean Makespan	18863346	21155754.1	4659276
Percentage of difference with Max-Min	75.29	77.98	-

TABLE 8
Results Achived By Increasing The Number Of Selected Similar cases

CBR-LA (Random CB with 100 cases)			**Max-Min**
Number of selected similar cases	Mean Makespans	percentage of difference with Max-Min	
3 cases	18863346	75.30	4659276
5 cases	18066344	74.21	
7 cases	17840050	73.88	
10 cases	17385551	73.20	

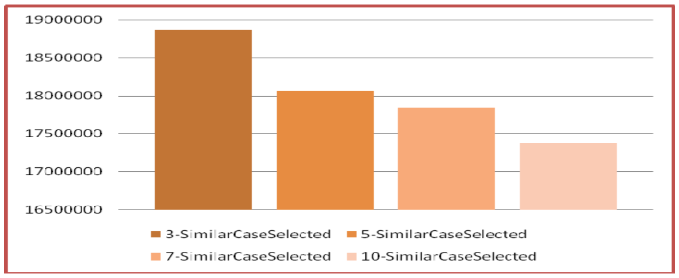

Fig.4. Comparing the CBR-LA results when it uses different number of experiences.

As introduced in section II-A, the LA model in CBR-LA assumed that the environment was a P-model so the input set for each LA was {0, 1}. As mentioned in fig. 2, if the Makespan value at iteration *n* was less than the value at iteration *n-1*, then the input for each automaton was favorable; otherwise it was unfavorable. It was found that this procedure has a major weak point. And as a result, the weak point prevents the CBR-LA to find a near to optimum results. Using the procedure automata always considers two recent results and rewards or penalties to LA actions were given according the results. By so doing, it causes to inefficiency of CBR-LA. In order to have a clear understanding about the weak point, assume that the results achieved in 3 recent iterations are as $T(n-1) << T(n) > T(n+1)$ which shows that in iteration *n-1* the automata has found a good result rather than in iterations *n* and *n+1*. But the procedure first evaluates the results of iterations *n-1* and *n*, then penalize automata and then compares the iterations *n* and *n+1* and in spite of an unfavorable result (in contrast with the *n-1* iteration's result) it rewards. E.g. assume that the Makespan value in iteration *n-1* is 1000 and this value in iterations *n* and *n+1* is 40000 and 15000 respectively. As you see, the value achieved in iteration *n+1* is better than 40000 but it is not favorable in contrast with 1000. This example illustrates that if the procedure of the LA model has a good memory, then it will lead to a better responses.

In this section the LA model was changed in a way that each automaton would compare its result with the best result achieved until yet. This new procedure is illustrated in fig. 5 as ICBR-LA that was abbreviated for Improved CBR-LA. By this algorithm, the automata can maintain its best result during its searches in response space of the problem.

A. Evaluating the functionality of ICBR-LA algorithm in contrast with the CBR-LA

In this part, the CBR-LA and ICBR-LA algorithms were evaluated to show improvements achieved by the new LA model in the results. Both of the algorithms were selected 3 and 5 similar cases to solve each problem in the experiments (results are shown in table11 and are mean values of 20 executions). According to the table 11, in both CBR-LA and ICBR-LA algorithms the mean Makespan were reduced by increasing the numbers of selected similar cases, but the ICBR-LA gives more optimized results (24 percent in the case that the algorithm uses 5 selected similar cases).

```
While (true)
Begin
  LA_i selects its action  for all 1≤ i ≤τ ;
  Evaluate the Makespan ;
  If  Current-Makespan < The-Best-Makespan Then
      Set Reward for all LA_i  that  1≤ i ≤τ ;
  Else
      Set Penalty for all LA_i  that 1≤ i ≤τ ;
  EndIf
  If (no change occurs  in Makespan for 150 consecutive iteration)   OR
(10000 iteration  is over)Then
      Exit ();
  EndIf;
End
```

Fig.5. The general procedure for ICBR-LA.

VI. Changing The Type Of Environment Responses

As described in II-A, the VSLA modifies its actions probability values as a result of analyzing the environment's responses. Also, the CBR-LA algorithm assumes the environment as a P-model. In order to the results that achieved in previous sections, it's likely that the environment model couldn't describe how the actions were favorable, so the responses were not useful for the automata. Then it is decided to use S-model environment. The ICBR-LA algorithm was examined by this new environment model and the results showed improvements in terms of Makespan.

In spite of P-model that uses only two values 0 and 1 as a response, an S-model environment generate responses in a continuous range (the responses lie in the interval [0, 1], that the smaller responses illustrate the more favorable ones). This feature will cause to describe the effect of automata actions better, and consequently it will be beneficial in making decisions about giving reward or penalty for automata actions depending on how well the action was. Parts of this section are dedicated to describe details of generating the S-model environment responses and the results achieved from experiments.

A. The algorithm for generating S-Model responses

The procedure of S-Model response generator is shown in pseudo code at fig.6. According to the procedure the difference of new actions Makespan and the best Makespan until yet is calculated first. In order to do this, after performing actions by LA, the Makespan related to current actions is calculated and the difference of new Makespan with the best one is calculated next. The result of recent calculation will be a value between 0 and 100. This value is a measure which presents the desirability of the actions was done by the automata.

Because of the effect that penalizing has on the probability set of the LA, for the situations which the Makespan is less than or equal to the best Makespan, the response of environment is considered as zero (the most favorable). Otherwise, depending on the amount of difference with the best Makespan, a number between 0 and 1 is generated as an S-model response. To do this, the amount of difference percentage with the best Makespan must be calculated (the result will be in range [0,100]), then the result is divided by 100. By so doing, the response of environment will always be in the continuous range of [0, 1].

B. The results achieved by using S-Model environment

In this section the effect of changing environment responses from P-Model to S-Model is presented. In order to have a full investigation on the results of ICBR-LA, the algorithm was examined over all classes of problems (section II-D). The results are shown in table 10 that each value is average of 20 executions of the algorithm. Also, the number of selected similar cases by the algorithm was 3 cases. According to the table 10, S-Model responses of the environment were improved the solutions of the ICBR-LA algorithm especially in case of the consistent classes of the problems.

VII. Comparing Proposed ICBR-LA With CBR-LA And Two Other Algorithms

In this section, the results achieved by the proposed ICBR-LA and its original form (CBR-LA) beside Max-Min and Min-Min algorithms were compared. Therefore, like the section VI-B we examined the algorithms over all classes of problems. The average results of running these algorithms for 20 problems of each class of heterogeneity are shown in tables 11, 12 and 13. Table 11 represents the amount of optimization that gained according to the proposed modifications in the CBR-LA. It is illustrated that at the worst case the proposed algorithm optimized the results of CBR-LA up to 25.20% and at the best case this optimization reaches to 39.95%, which is a great reduction. But as illustrated in tables 12 and 13, in spite of significant improvements that the proposed algorithm gained, there is still a great difference with other algorithms from results aspect, that is, at the best case ICBR-LA has 23.42% difference with Max-Min and 46.97% difference with Min-Min algorithm. Therefore we'll try to find other solutions to much more improving the proposed algorithm during our next researches.

VIII. Conclusion

In this paper, functionality of the hybrid CBR-LA model, which has proposed in [8] for task assignment in HC systems, evaluated and improved significantly. The main idea in this improvement was modifying the LA model (by changing the conditions which inspect the effect of automata actions on HC environment) and the type of responses which the environment generates, from P-model to S-model. Using computer simulations it was shown that by applying the both changes in the algorithm, the results improves significantly in terms of Makespan performance metric to generating near to optimal solutions (at the best case, 39.95% reduction). Finally we had presented some experiment results in order to representing comparison between the proposed ICBR-LA algorithm in contrast with Min-Min and Max-Min algorithms. The results showed that in spite of great improvements still the proposed algorithm has a great difference to reach other algorithms performance.

```
Calculate current-Makespan's difference percentage with best- Makespan;
If difference <=0 Then
    β=0;
Else
  Set β=difference percentage/100;
EndIf
```

Fig. 6. The general procedure for generating S-Model responses.

TABLE 9
RESULTS ACHIVED BY COMPARING THE ICBR-LA AND CBR-LA ALGORITHMS FOR 20 PROBLEMS

CBR-LA (with 100 Random Cases in its CB)			**ICBR-LA**(with 100 Random Cases in its CB)	
Number of selected similar cases	3 Cases	5 Cases	3 Cases	5 Cases
mean Makespan	18863346	18066344	13951096	13714116
Percentage of difference with CBR-LA	-	-	26.04124	24.09025

TABLE 10
THE RESULTS ACHIEVED BY EXECUTING ICBR-LA ALGORITHM IN P-MODEL AND S-MODEL ENVIRONMENTS

Consistency	**Task Heterogeneity**	**Machine Heterogeneity**	**ICBR-LA (P-Model)**	**ICBR-LA (S-Model)**	**Reduction in Makespan values**
Incons.	High	High	13951096	13747204.5	1.46
	High	Low	157053.3	154768.9	1.45
	Low	High	481184.3	473516.9	1.59
	Low	Low	5861.6	5777.35	1.44
Cons.	High	High	16967742	16147229	4.83
	High	Low	166106.2	159174.4	4.17
	Low	High	515709.2	510468.3	1.01
	Low	Low	5725.35	5492.9	4.06

TABLE 11
COMPARISION BETWEEN CBR-LA AND PROPOSED ALGORITHM (ICBR-LA)

Consistency	**Task Heterogeneity**	**Machine Heterogeneity**	CBR-LA	**ICBR-LA (S-Model)**	Reduction in Makespan values
Incons.	High	High	19316288	13747204.5	28.83
	High	Low	211800.6	154768.9	26.92
	Low	High	662140.3	473516.9	28.49
	Low	Low	7723.55	5777.35	**25.20**
Cons.	High	High	26888098	16147229	**39.95**
	High	Low	249500.8	159174.4	36.20
	Low	High	753277.2	510468.3	32.23
	Low	Low	8452.1	5492.9	35.01

TABLE 12
COMPARISION BETWEEN RESULTS ACHIEVED BY ICBR-LA AND MAXMIN ALGORITHMS.

Consistency	**Task Heterogeneity**	**Machine Heterogeneity**	Max-Min	ICBR-LA	Percentage of difference
Incons.	High	High	9135809	13951096	34.51
	High	Low	109420.9	157053.3	30.33
	Low	High	299151	481184.3	37.83
	Low	Low	4335.154	5861.6	26.04
Cons.	High	High	9078309	16967742	46.50
	High	Low	127206.1	166106.2	23.42
	Low	High	295614.4	515709.2	42.68
	Low	Low	4576.571	5725.35	20.06

TABLE 13
COMPARISION BETWEEN RESULTS ACHIEVED BY ICBR-LA AND MIN-MIN ALGORITHMS.

Consistency	**Task Heterogeneity**	**Machine Heterogeneity**	Min-Min	ICBR-LA	Percentage of difference
Incons.	High	High	5460156	13951096	60.86
	High	Low	63562.51	157053.3	59.53
	Low	High	169390.7	481184.3	64.80
	Low	Low	2494.503	5861.6	57.44
Cons.	High	High	5967740	16967742	64.83
	High	Low	88076.4	166106.2	46.97
	Low	High	191751.7	515709.2	62.82
	Low	Low	3035.603	5725.35	46.98

REFERENCES

[1] D. Fernandez-Baca, "Allocating Modules to Processors in a Distributed System," *IEEE Transaction on Software Engineering*, Vol.15, pp.1427-1436, 1989.

[2] T. D. Braun, H. J. Siegel, and N. Beck, "A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems," *Journal of Parallel and Distributed Computing*, Vol.61, pp. 810-837, 2001.

[3] M. Coli, and P. Palazzari, "Real Time Pipelined System Design through Simulated Annealing," *Journal of Systems Architecture*, Vol.42, pp.465-475, 1996.

[4] K. Chow, and B. Liu, "On Mapping Signal Processing Algorithms to a Heterogeneous Multiprocessor System," *International Conference on Acoustics, Speech, and SignalProcessing*, Vol.3, pp.1585-1588, 1991.

[5] H. Singh, and A. Youssef, "Mapping and Scheduling Heterogeneous Task Graphs Using Genetic Algorithms," *5th IEEE Heterogeneous Computing Workshop*, pp. 86-97, 1996.

[6] L. Wang, H. J. Siegel, V. P. Roychowdhury, and A. A. Maciejewski, "Task Matching and Scheduling in Heterogeneous Computing Environments Using a Genetic-Algorithm-based Approach," *Journal of Parallel Distributed Computing,* Vol. 47, pp.1-15, 1997.

[7] R. D. Venkataramana, and N. Ranganathan, "Multiple Cost Optimization for Task Assignment in Heterogeneous Computing Systems Using Learning Automata," *IEEE 8th Heterogeneous Computing Workshop*, pp.137, 1999.

[8] S. Ghanbari, M. R. Meybodi, and K. Badie, "A Case-Based Recommender for Task Assignment in Heterogeneous Computing Systems", *Proceedings of the Fourth IEEE International Conference on Hybrid Intelligent Systems,* pp. 110-115, 2004.

[9] K. Narendra, and M. A. L. Thathachar, *Learning Automata: An Introduction*, Prentice Hall, Englewood Cliffs, New Jersey, 1989.

[10] K. Najim, and A. S. Poznyak, *Learning Automata: Theory and Application*, Tarrytown, NY: Elsevier Science Ltd., Pergamon Press, Cambridge, England, 1994.

[11] S. K. Pal, and S. C. K. Shiu, *Fundations of Soft Case-Based Reasoning*, Wiley Series on Intelligent Systems, A John Wiley & Sons INC. Publication, 2004.

[12] R. Bergman, "Engineering Applications of Case-Based Reasoning," *Journal of Engineering Applications of Artificial Intelligence*, Vol. 12, pp.805, 1999.

[13] A. Aamodt, and E. Plaza, "Case-Based Reasoning: Foundational Issues, " *Methodological Variations and System Approaches AI Communications*, IOS Press, Vol. 7: 1, pp. 39-59, 1994.

[14] 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 Distributed Computing*, Vol.59, pp.107-121, 1999.