Application of Meta Heuristic Algorithm for Real Time Task Assignment Problem on Heterogeneous Processor

Poongothai M.
Assistant Professor
Dept of ECE
CIT

Rajeswari A.
Professor
Dept of ECE
CIT

UmerFarook K.A.
PG scholar
Dept of ECE
CIT

ABSTRACT
Multiprocessor real-time task assignment algorithm helps in the design and implementation of real time systems. Assigning real time task to heterogeneous multiprocessor system is challenging problem because the performance of each task varies from one processor to another. As the result of this determining solution for assigning task in heterogeneous processor leads to an NP hard problem. In this paper, Hybrid Ant Colony Optimization incorporated with Tabu search algorithm [HACO_TS] is proposed for real time task assignment in the heterogeneous system. The proposed Max-Min Ant System is included with a Tabu search algorithm to improve task assignment solution without exceeding the processors computing capacity and fulfilling the dead line constraints. From the experimental results, the proposed algorithm achieved better utilization compared to random assignment algorithm.

Keywords
Multiprocessor, Heterogeneous, NP hard, HACO_TS, Max-Min Ant system, Tabu search, random assignment

1. INTRODUCTION
It is well known that the fast development in integrated circuit technology and system architecture has led to today's real-time embedded systems more complex. And also it includes many heterogeneous components. Multiprocessor in a real-time system consists of a number of tasks with different deadlines. Assigning these tasks to heterogeneous platform more difficult when compare to the homogenous platform. In general, mapping of task into processor known as NP hard. In this paper a Hybrid Ant Colony Optimization incorporated with Tabu search algorithm [HACO_TS] is proposed. The objectives of the proposed algorithm are resource objective and energy objective. Resource objective is searching for a solution in which each of the tasks is assigned to a specific processor in such a way that the cumulative utilization of any processor does not exceed the utilization bound of the EDF algorithm. Energy objective is minimizing the cumulative energy consumption of all assigned tasks in a solution. The resource objective is given precedence over the energy objective.

2. RELATED WORK
K.Prescilla, A.ImmanuelSelvakumar[12] proposed Modified BPSO[Binary Particle Swarm Optimization], for real time task assignment problem. They were generated Consisted utilization matrix, all the processor having unique speed. Inconsistent utilization matrix each processor runs the each task on different speed. They have proven that the Modified BPSO suitable for inconsistent utilization matrix and ACO suitable for consistent utilization matrix.

T.Vetriselvan, P.Chitra, Dr.P.Venkatesh[3] proposed Ant Colony optimization (ACO) for task scheduling problem. The performance of their algorithm was demonstrated by the time for producing effective schedules for random task graphs.

Hyunjin Kim, Sunghokang[4] proposed an algorithm for off-line communication aware scheduling and voltage selection using Ant Colony Optimization. The main aim of algorithm was to minimize total energy consumption of an application executing on a homogeneous multiprocessor system.

G.Umarani, V.UmaMaheswari[13] proposed a task assignment algorithm considering load balancing across the processor. The objective of this paper isto avoid the situation where some processorsareidle while some others are overloading. The results are compared and it is observed that ACO performs better than FCFS with respect to the wait time.

Qinma Kang, Hong He[5], proposed HBMO for task assignment in Heterogeneous computing system. HBMO is one of new member of the swarm intelligence. It has been used successfullyfor problems from computer science, water resource management partitioning and scheduling in the design of embedded systems. In this model a single queen, single worker, drones and broodsis considered. Unlike ACO each artificial bee has genotype, a set of genes, which can be used to construct complete solution to the problem. The results are compared with Hybrid PSO. The experimental results showed that the HBMO based algorithm runs faster than compared algorithm. But they have not consider the energy consumption of assigned tasksChen, H., A.M.K. Cheng and Y.W. Kuo[2] proposed a new algorithm based on ant colony optimization (ACO) alongwith local search heuristic algorithm to improve task assignment solution. The results are compared with GA and LP based approaches.

3. SYSTEM MODEL
The heterogeneous multiprocessor environment with *m* preemptive processors $\{P_1, P2 \ldots\ldots P_m\}$ based on CMOS technology is considered in our research work [1], [8-10]. The processors in the heterogeneous environment are operated at different speeds and one instruction per cycle is limited to execute in each processor at variable speed. The energy consumption is calculated using equation (1)

$$Energy_{i,j} = Power_{i,j} \times e_{i,j} = (((C_{ef} \times S_{i,j}{}^3)/K^2) \times e_{i,j}) \quad (1)$$
$$= ((C_{ef}/K^2) \times C_i \times S_{i,j}{}^2)$$

Where $C_{ef}$is the effective switch capacitance related to tasks, k is the constant. $e_{i,j}$is the execution time for task T_i on processor P_j , $s_{i,j}$is the speed of P_j for T_i and c_i is the number of clock cycles to execute a task T_i. From the equation (1) it is understood that, energy consumption is directly proportional to the *ci.* s_{ij}^2. This equation is significant, because the processors operate at different speeds [8], [10].

A set of N periodic tasks $T=\{T_1,T_2, \dots \dots \dots T_N\}$ is considered. The tasks are assumed to be mutually independent and there is no inter task communication [9],[10]. T_i is defined as $T_i = \{e_{i,j}, p_{i,j}\}$ where $e_{i,j}$ is the execution time and $p_{i,j}$ is the period. The utilization matrix $u_{i,j}$ is an nxm matrix in which m is the number of heterogeneous processors and n is the number of tasks. The task assignment problem considered here is the off-line version, under the condition that the utilization of each processor is less than or equal to 1. The partitioned scheme for assigning tasks and Earliest Deadline First algorithm (EDF) are considered for scheduling the tasks on each processor [4],[13-15].

4. MAX-MIN ANT COLONY OPTIMIZATION

ACO is a meta heuristic technique. an NP hard combinational optimization problems are determined using ACO.

Marco Dorigo and colleagues introduced the first ACO algorithms in the early 1990's. The algorithm was designed based on the behaviour of ants, which enables them to find the shortest path between nest and foods. Ant System(AS) [16] [17] is the first ant colony algorithm is that the pheromone values are updated by all the ants that have made a tour. There are several variants on AS shown in figure 1.

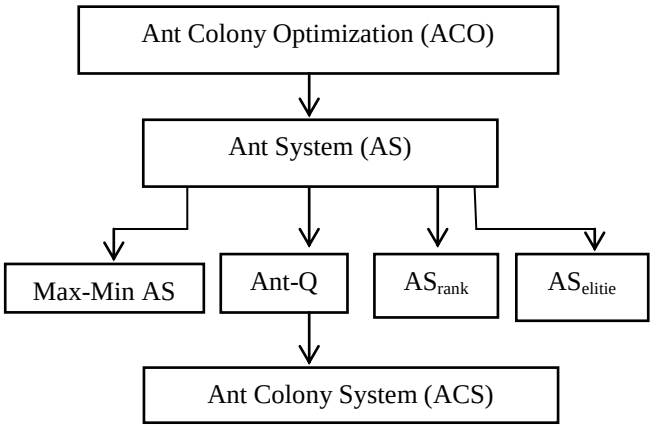

Figure 1: ACO classification

Max-Min AS, Ant-Q, AS_{rank}, AS_{elite}, are the several variants of Ant System.

The key feature of MMAS [18] is that, trails are updated with only one ant. the ant which produced the best solution(global-best ant) or in the current iteration the ant which produced the best solution(iteration best ant) is used for the pheromone trial updating . To avoid gettingstuck at local solutions certain values will be set as the Max & Min values of the pheromones.

Pseudo code:

```
Step 1:{Initialization}
       Initialize τ(i,j) and η(i,j).
Step 2. {Construction}
       For each ant k (currently in state i) do
       repeat
                choose in probability the state to move into.
                append the chosen move to the k-th ant's set S.
                until ant k has completed its solution.
       end for
Step 3:{Local search}

Step 4. {Trail update}
       For each ant move
       do compute
       τ(i,j) update the trail matrix.
       end for
Step 5. {Terminating condition}
       If not(end test) go to step 2
```

Step 1 {Initialize Pheromone Values}: At the start of the algorithm the pheromone values are all initialized to a constant value c > 0.

Step 2 {Construct Solution}: The basic ingredient of any ACO algorithm is a constructive heuristic for probabilistically constructing solutions. A constructive heuristic collects solutions as sequences of elements from the finite set of solution components '*C*'. A solution construction starts with an empty partial solution $S = <>$. Then, at each construction step the current partial solution S is extended by adding a feasible solution component from the set $M(S) \subseteq C \setminus \{S\}$. This set is determined at each construction step by the solution construction mechanism in such a way that the problem constraints are met. The choice of a solution component s j i$\epsilon N(S)$ is, at each construction step, done probabilistically with respect to the pheromone model. In most ACO algorithms the probabilities for choosing the next solution component—also called the transition probabilities—are defined as follows

$$p(s,i,j) = \frac{\tau(i,j)^{\alpha}.\eta(i.j)^{\beta}}{\sum_{(i',j')\epsilon N(s)}^{n} \tau(i',j')^{\alpha}.\eta(i'.j')^{\beta}} \text{if(i,j)}\epsilon \text{N(s)} \quad (2)$$

$$= \quad 0 \qquad \text{otherwise}$$

It is influenced by the pheromone trail $\tau(i,j)$ and by locally available heuristic information n=1/di,j that depends on distance di,j between i and j . Ants prefer solution that are close and connected by links with high pheromone trail. α and β are two parameters that determine the relative importance of the pheromone trail and the heuristic information. $N(s)$ is the set of solution ant has not visited yet.

Step 3 {Local search}: Local search procedure for the artificial antsto improve their solutions after they complete the construction procedure. Local search heuristic starts off with an initialassignment solution, and then tries to find better solutions.

Step 4 {Pheromone Update}: Initially all pheromone values are set to τ_{max} and after each iteration pheromone limits are updated. Then validity of limits is checked. If $\tau(i,j)$ is lesser than τ_{min}it will set $\tau(i,j)=\tau_{min}$. If $\tau(i,j)$ is greater than τ_{max}it will set $\tau(i,j)=\tau_{max}$.

If the current best solution has not improved for a certain number of iterations reinitialize the pheromone table by $\tau(i,j) = \tau max$ setting for all i, j.

Step 5 {stopping condition}: The variable $n_{ij}$is initialized. In each round, if the local best solution is not improved than the global one, the value is increased. The algorithm stops when n_{ij} reaches threshold. or, the algorithm stops when at least one feasible solution is found.

5. HYBRID ANT COLONY OPTIMIZATION WITH TABU SEARCH (HACO_TS) FOR TASK ASSIGNMENT PROBLEM

For given set of Heterogeneous Multiprocessor and task set, each task assigned to one processor by the artificial ant without exceeding its computing capacity. The task assignment is associated to the utilization of the task upon the processor.

ui,j represents the utilization of the *ith* task on the *j*-th processor. The utilization matrix *ui,j* is an *nxm* matrix in which *m* is the number of heterogeneous processors and *n* is the number of tasks. The utilization matrix U_{n*m} holds the real numbers in (0, 1) and infinity. If $Uij = \infty$ means, the particular task is not suitable to execute on a specified processor P_j. An artificial ant finds to travel across the construction graph in such a way that *all* of the following constraints are satisfied: (1) one and only one cell is visited for each of the rows; (2) the accumulative value of the visited cells on the same column is no greater than 1. An artificial ant constructs the task assignment solution based on the constraints given by equation (3)

$\sum_{i=1}^{n} u(i,j).\Delta^{s}_{i,j} \leq 1$ j=1,2……..m; s=solution (3)

$\Delta^{s}_{i,j}$ = 1 if square (Ti,Pj) is visited in solution

0 otherwise

5.1 Solution construction

During solution construction thealgorithm will not use any heuristic information. Insteadof heuristic information Tabu search algorithm is included for further improving solutions. Tabu search (TS) is a neighborhood search method which employs "intelligent" search and provides flexible memory technique to avoid being trapped at local optimum.It takes advantage of search history.The historical record is usually maintained for the characteristics of the moves applied.Existing solutions are classified as tabus to restrict the search space and avoid cyclic behavior.

5.2 Tabu search

Tabu search algorithm is a metaheuristic that guides a local heuristic search to explore the solution space beyond local optimality. It is based on the use of prohibition techniques and "intelligent" schemes which exploit the past history of the search in order to influence its future steps.

The basis for TS may be described as follows. Let denote S the set of the feasible solutions of a problem. With each s € S, a subset of *S* , called neighborhoodof s and denoted with *N(S)* . The neighborhood of *s* contains all those solutions which can be obtained with a modification of '*s*', called *move.* Given a starting feasible solution '*s*', Tabu search iterates the following steps:

Step 1: compute *N(S)*

Step 2: select within *N(S)* the solution s_N with the best objective function value

Step 3: if such value is better than that of then 's' replace with 's_N'and go to step 1), otherwise stop and return.

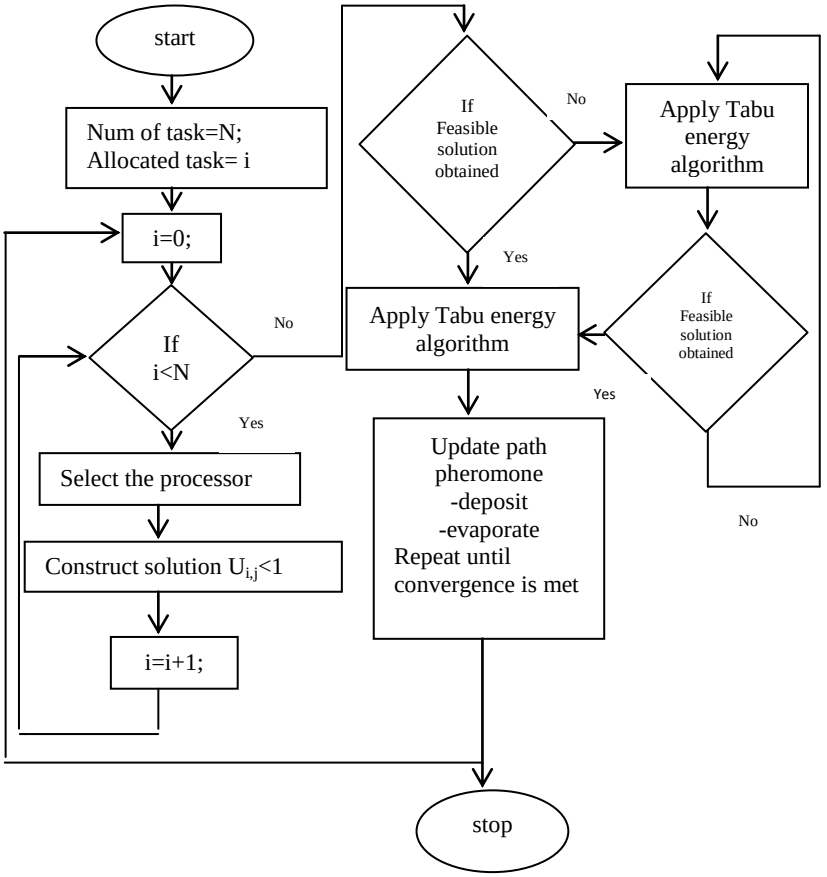

Figure 2: Flow chart for proposed HACO_TS

The flow chart for proposed algorithm is shown in figure 2.

The stepwise procedures of the proposed algorithm are described.

1) Initialization: set the algorithm parameters.

2) Generation of random task assignment :each task assigned to one processor by the artificial ant without exceeding its computing capacity

3) Exploitation

i) Intensification search: Remove task i from j^{th}processor assign to the new processor ,

If overall utilization of the processor is improved then task remained in processor

Else

Update tabu list and go to step i

ii) Information sharing: pheromone update by the ant best one searched so far in this cycle.

4) Termination test: If the maximum task assigned , stop; otherwise, empty the memories of ants and then go to (2).

Total number of task is assumed to be *N*. and assigned task is considered as i. Initially i=0. Then each task is assigned to one processor by the artificial ant, such that the processor capacity is not exceeded, after the task gets assigned thevalue ofi is increased by one. This process will continue till total number of task (N) is equal to the number of assigned task (i) .

After that it checks for the feasibility of the solution.

Feasible tour is a condition where the ant stops with all tasks assigned.Infeasible tour isa state where the ant stops with at least one unassigned task.If no stop condition is met, the current tour is called a partial tour

To minimize the energy consumption of all assigned task, Tabu energy algorithm is applied if solution is feasible.

To maximize the number of task assignment, Tabu resource algorithm is applied if solution is infeasible. After that it again check for whether feasible solution is obtained. if feasible solution is obtain Tabu energy algorithm is applied. Else continuously applies Tabu resource algorithm until it will get feasible solution.

For every iteration the pheromone path is updated. It evaporate or deposit pheromone depend upon the solution. This process is repeated until convergence is met.

The pheromone values of ant are initialized as same for solution construction. Each ant builds a tour from a starting pair of task and processor[11]. The probability of selecting next pair of task and processor is given by

$$\tau(i,j) = \frac{\tau(i,j)}{\sum_{(i',j')\epsilon N(s)} \tau(i',j')} \text{ if } (i,j)\epsilon N(s) \quad (4)$$

$$0 \quad \text{otherwise}$$

The equation (4) describes the average of pheromone distribution over the set of eligible pairs. In this equation, *N(s)* denotes the set of eligible pairs of (Task, Processor) obtained; *τ (i, j)* denotes the pheromone trial of (Ti, Pj).

When a solution is constructed, the artificial ants continue to update the pheromone trials by H-MMAS [2], [10] according to the equation (5). *τ(i, j)* will be increased every time task Ti is assigned to processor Pj in the best solution. The pheromone is updated as follows:

$$\tau(i,j) = \begin{cases} (1-\rho)\,\zeta(i,j) + f(s^{best}) & \text{if } (i,j)\epsilon s^{best} \\ (1-\rho)\,\zeta(i,j) & \text{otherwise} \end{cases} \quad (5)$$

f(s) is a quality function which measures quality of the solution and is given by

$$f(s) = \gamma + (1 - \beta) \quad (6)$$

γ =number of assigned task in solution s

$$\beta = \frac{cumlative\ \ energy\ \ consumption\ \ \ of\ all\ assined\ \ task\ \ in\ solut\ ion\ s}{maximum\ \ \ possible\ \ enegy\ \ consumption}$$

γ is maximum number of assigned task function and β is minimums energy consumption of all assigned task function. The final quality function is maximization function.

6. RESULTS AND DISCUSSIONS

To test our proposed algorithm HACO_TS, experiments are performed on an Intel core i3 CPU processor running at 2.27 GHZ with 1.87 GB RAM. The operating system is MS Windows 7, 64 bit running the MATLAB R2011b environment. The utilization matrix Un*m holds the real numbers in (0, 1).

In HACO_TS, Each ant constructs solution in random fashion. Then the Tabu search is applied to the random assignment to improve the solution. The tasks are assigned to the processor only if the addition of utilization of the particular task in that particular processor is below its utilization bound less than or equal to 1, according to the EDF algorithm. The ants construct solution until no more tasks can be added to the solution.

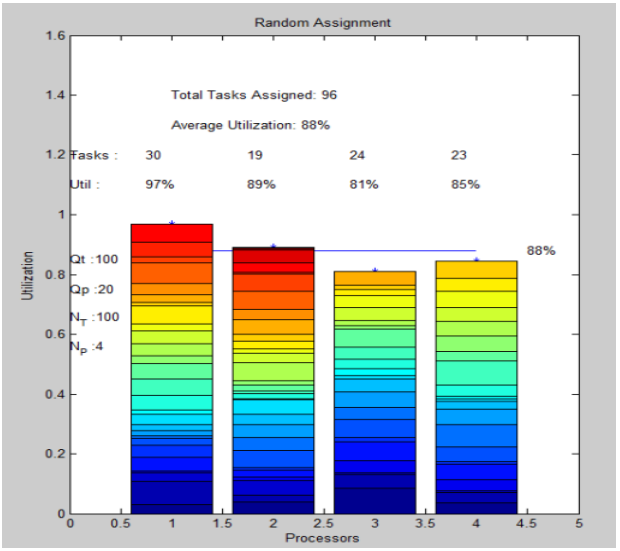

Figure 3: Randomly Assigned Solution for U4*100 C_HT_HP

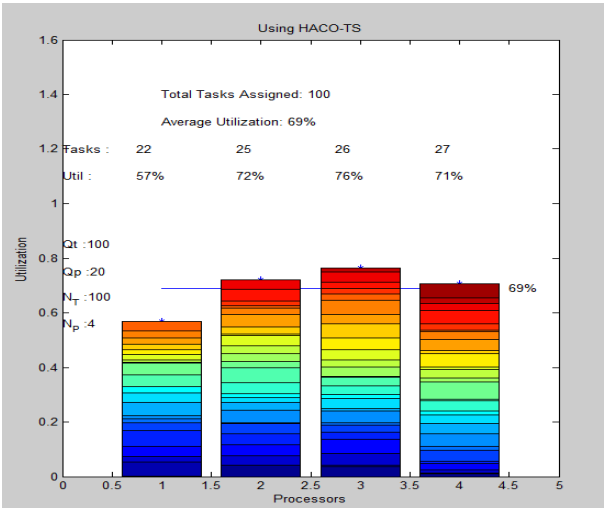

Figure 4: HACO_TS Task assignment solution forU4*100 C_HT_HP

Table 1: Comparison Table for Random Assignment and HACO_TS for U4*100 C_HT_HP

	Random Assignment				HACO_TS			
	P1	P2	P3	P4	P1	P2	P3	P4
Utilization Per Processor (%)	97	89	24	23	85	55	72	72
Tasks per Processor	30	19	25	23	30	21	22	27

Total Number of Tasks allocated	96	100
Average utilization (%)	88	69

For each ant's randomly generated solution shown in figure 3. HACO_TS is performed to find the best solution from the neighborhood. In HACO_TS, the solution of ant is shuffled considering one task at a time and reduce the average utilization shown in figure 4. Table. 1 Shows the comparison of various parameters for Random Assignment and HACO_TS.

To evaluate the algorithms for different realistic scenarios, thecharacteristics of the utilization matrix were varied based on taskheterogeneity, processor heterogeneity, and consistency. Hence,eight combinations of utilization matrix characteristics are usedin our experiments: high or low task heterogeneity (HT or LT),high or low processor heterogeneity (HP or LP), and consistent or inconsistent utilization matrix (C or IC). High task heterogeneity isrepresented by $\Phi T = 100$ and low task heterogeneity by $\Phi T = 5$.High processor heterogeneity values are generated using $\Phi P = 20$. and low processor heterogeneity values using$\Phi P = 5$. Consisted utilization matrix, all the processor having unique speed. Inconsistent utilization matrix each processor runs the each task on different speed.

For each ant's randomly generated solution for 8*150problem instance shown in figure 5. For same problem instance HACO_TS shown in figure 6.

Table 2 shows the comparison between random assignment and HACO_TS for nine different problem instances.

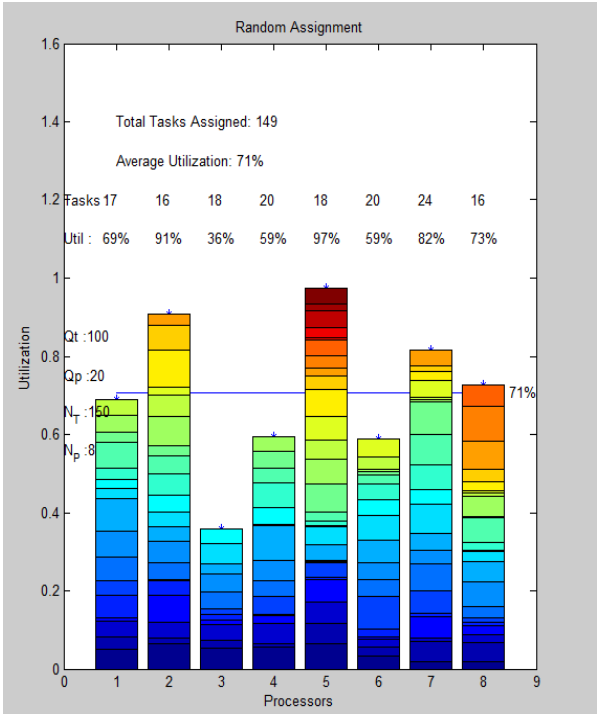

Figure 5: Randomly Assigned Solutionfor U8*60 C_HT_HP

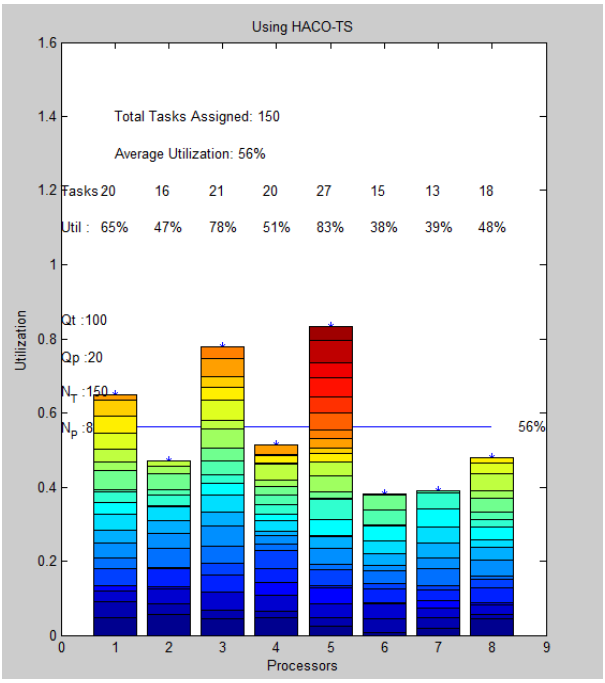

Figure 6: HACO_TS Task assignment solution forU8*60 C_HT_HP

Table 2: Comparison Table for Random Assignment and HACO_TS for nine problem instance

Problem set	**Size**	Random task assignment		HACO_TS	
		Total Number of Tasks allocated	Average utilization (%)	Total Number of Tasks allocated	Average utilization (%)
C_HT_HP	U4*100	96	88	100	69
C_HT_LP	U8*60	60	27	60	19
C_HT_HP	U8*150	149	71	150	56
C_LT_HP	U4*80	78	81	80	69
C_LT_LP	U8*50	50	26	50	19
IC_HT_LP	U8*60	60	30	60	20
IC_LT_HP	U4*100	99	87	100	70
IC_LT_LP	U8*60	59	29	60	20
IC_LT_LP	U5*20	20	20	20	12

From the table 2, it can be observed that the proposed HACO_TS algorithm assigns more number of tasks than random task assignment and also average utilization is minimized compare to random task assignment

7. CONCLUSION

The proposed HACO_TS algorithm performance is compared with Modified BPSO and ACO algorithms in terms of number of task assigned, and average utilization. The proposed Max-Min Ant System is included with a Tabu search algorithm. . The objectives of the proposed algorithm are resource objective and energy objective.From the experimental results, the proposed HACO_TS algorithmperforms better than random assignment algorithm.

The proposed work can be extending by considering task precedence and inter-task communication in heterogeneous computing environment. From a practical point of view, it will be interesting to test proposed HACO_TS algorithm in a variety of real computer clusters under variable working loads.

8. REFERENCES

[1] T.Braun, et al., "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing system", Journal of Parallel and Distributed computing Vol. 61, pp810-837,2001.

[2] Chen, H., A.M.K. Cheng and Y.W. Kuo, "Assigning real-time tasks to heterogeneous processors by applying ant colony optimization", J.Parallel Distributed Computing, 71: 132-142, 2011.

[3] T.Vetiselvan, P.Chitra, Dr.P.Venkatesh, "parallel implementation of task scheduling using Ant Colony Optimization",International Journal of Recent Trends in Engineering,Vol. 1. No. 1.may 2009.

[4] HyunJin Kim, Sungho Kang, "Communication-aware task scheduling and voltage selection for total energy minimization in a multiprocessor system using Ant Colony Optimization", information sciences 181, pp 3995-4008, 2011.

[5]Qinma Kang, Hong He, "Honeybee Mating Optimization algorithm for Task assignment in heterogeneous ccomputing systems", Intelligent Automation & soft Computing, 12 July 2013.

[6] M.B.Abdelhalim, "Task assignment for heterogeneous multiprocessors using Re-Excited Particle Swarm Optimization", International Conference on computer and Electrical Engineering, pp 23-27, 2008.

[7] Albert M. K., Cheng, Real-Time Systems: Scheduling, Analysis, and Verification, University of Houston, John Wiley & Sons, 2002.

[8] Peng-Yeng Yin, Shiuh-Sheng Yu, Pei-Pei Wang, Yi-Te Wang, "A hybrid particle swarm optimization algorithm for optimal task assignment in distributed systems", Computer Standards & Interfaces, Vol.28, pp. 441-450, 2006.

[9] H. Chen, A.M.K. Cheng, Applying ant colony optimization to the partitionedscheduling problem for heterogeneous multiprocessors, WIP session, in: IEEERTAS, 2005.

[10] Marco Dorigo and Thomas Stützle ., Ant ColonyOptimization**,** MIT PressCambridge, MassachusettsLondon, England 2004.

[11] Ms.M.Poongothai, "ARM Embedded Web Server Based on DACS System", IEEE proceedings, International Conference on Process Automation, Control and Computing(ICPAC11), July 20-22, 2011.

[12] K. Prescilla, A. Immanuel Selvakumar, "Modified Binary Particle Swarm optimization algorithm application to real-time task assignment in heterogeneous multiprocessor", Microprocessors and Microsystems Vol.37, 583–589, 2013.

[13] Umarani, G**.** Srikanth, "Tasks Scheduling using Ant Colony Optimization**",** Journal of Computer Science Vol.8 (8): 1314-1320, 2012.

[14] Hong Jin, Hui Wang, Hongan Wang, Guozhong Dai, "An ACO-Based Approach for Task Assignment and Scheduling of Multiprocessor Control Systems"Springer Berlin Heidelberg Proceedings, Third International Conference on Theory and Applications of Models of Computation(TAMC 2006), Beijing, China, May 15-20, pp 138-147, 2006.

[15] Jian Wu, Xinxue Liu, JianshengShu, Yaxiong Li, Kaifeng Liu, "Independent Task Assignment of Space Warfare Based onMASand ACO", Journal of Information & Computational Science 10:12 3861- 3867, 2013.

[16] M.,Dorigo, V., Maniezzo and A., Colorni, "Ant System: Optimization by a colony of cooperating agents", IEEE. T. Syst. Man. Cyb. PartB*26*, no.1, 29–41, 1996.

[17] M., Dorigo and L. M., Gambardella, "Ant Colony System: A cooperative learning approach to the traveling salesman problem", IEEE. T. Evolut. Comput.*1*, no. 1, 53–66, 1997.