A MULTIOBJECTIVE APPROACH FOR REAL TIME TASK ASSIGNMENT PROBLEM IN HETEROGENEOUS MULTIPROCESSORS

M. Poongothai[1], A .Rajeswari [2], A. Jabar Ali [3]

[1,2,3] Department of Electronics and Communication Engineering
Coimbatore Institute of Technology
Coimbatore 641014, India

E-mail: poongothai@cit.edu.in; rajeswari@cit.edu.in; jabar.dana@gmail.com

DOI: https://doi.org/10.22452/mjcs.vol32no2.3

ABSTRACT

Effective assignment of real-time tasks in heterogeneous multi-processor systems to achieve high performance is said to be an NP-hard problem. This paper addresses the problem of real-time task assignment in heterogeneous multiprocessor systems with the goal of maximizing the number of task assigned and decreasing the energy consumption. A heuristic-based Multi-objective Hybrid Max-Min Ant Colony Optimization algorithm (MO-HMMAS) on the heterogeneous multiprocessor system is proposed to analyze the tradeoffs between resource utilization of all assigned tasks and cumulative energy consumption. Also, we have constructed pareto fronts to illustrate different task allocations, which can cause a heterogeneous multiprocessor system to consume significantly different amounts of energy. The proposed algorithm has been implemented and evaluated using randomly generated problem instances.It was found that the proposed algorithm outperforms the Multi-objective ACO (MO-ACO) in terms of number of the tasks assigned and cumulative energy consumption of all assigned tasks.

Keywords: Real-time Task Assignment, Resource Objective, Energy Objective, Multi-Objective Optimization, Heterogeneous Processors, Ant Colony Optimization

1.0 INTRODUCTION

In general, a heterogeneous multiprocessor platform is based on the different instruction set architectures (ISAs) with configurable and extensible features. This multiprocessor platform meets the computational demands for various applications [1]. Real time embedded systems are more complex as it includes many heterogeneous components. It is very difficult to implement the real time applications on the heterogeneous multiprocessor system. Therefore, implementation of the real-time application on the heterogeneous platform needs additional effort than the homogeneous platform. The complexity increases in such a way that every real time application is in need of different execution times on heterogeneous processors [2,7]. The goal of the task assignment algorithm is to assign each task of the application on the given number of available heterogeneous processors for optimal assignment solutions. Finding such optimal assignment solutions is considered to be NP hard in general [4]. There have been extensive attempts in the literature on the task allocation and scheduling. Metaheuristic algorithms can be used to obtain suboptimal scheduling results for this type of combinatorial optimization problems rather than describing all possible schedules.

In this paper, Multi-Objective Hybrid Max-Min Ant Colony Optimization algorithm (MO-HMMAS) is proposed, which is based on Max- Min Ant System (MMAS) along with a local search as a daemon action finding a solution for real-time task assignment to the heterogeneous processors without exceeding the processors computing capacity and fulfilling the dead line constraints. The proposed MO-HMMAS algorithm considers two objectives for multi objective task assignment algorithm. The first objective is to achieve maximum task assignment (resource objective) in the heterogeneous multiprocessor. The second objective is to minimize the cumulative energy consumption (energy objective) for all assigned tasks. Multi-objective hybrid ant colony optimization (MO-HMMAS) on the heterogeneous multiprocessor system is proposed for handling two conflicting objectives, using the weighted sum approach. Inclusion of local search to such algorithm improves the convergence speed of solutions to the pareto front [3,15].

2.0 RELATED WORKS

There has been extensive focus in the literature on the task assignment and scheduling. In recent days, the research interest for task assignment and scheduling defines the multiple objectives for generating efficient solutions with minimum computation. Multiple objectives such as maximum utilization, the total number of task assigned, communication cost and energy consumption, and various other objectives are considered for optimization. Chen. Cheng and Kuo [6,7] proposed a new algorithm based on ant colony optimization (ACO) metaheuristic for solving the real-time task assignment problem in heterogeneous processors and also included the local search heuristic algorithm to improve task assignment solution. The results are compared with GA and LP based approaches. Prescilla et.al [11] proposed modified binary particle swarm optimization algorithm and novel binary particle swarm optimization to solve the independent real-time task assignment in heterogeneous multiprocessor. The resource objective and energy objective are not considered simultaneously. In particular, several works in [6,7, 9,10,11,20] have applied metaheuristics to task assignment and scheduling for multiprocessor systems. However, those works focused only on a single objective optimization problem and have not constructed pareto optimal solutions.

In [12], Jan Madsen proposed the multi-objective genetic algorithm to solve the problem of mapping a set of task graphs onto a heterogeneous multiprocessor platform with the objectives of minimizing the system cost and power consumption, by constructing the pareto front for both the objectives. MyungryunYoo & Mitsuo Gen [13] proposed a scheduling algorithm for real-time tasks using the multi-objective hybrid genetic algorithm (mohGA) on the heterogeneous multiprocessor environment to minimize the total tardiness and completion time . The convergence of GA was improved by simulated annealing algorithm. However, energy constraints were not considered in [13]. Chitra et.al [5] considered the two conflicting objectives namely, makespan and reliability for task scheduling problem in heterogeneous systems. However, these algorithms are proposed for general tasks without real-time constraints. Multi-objective ACO algorithm has been proposed by Alaya et al [16] for the Multi-Objective Knapsack Problem (MOKP). They have found that the multi-objective ACO variant (m-ACO4 (1, m)) produces globally the best solutions when compared to the existing Evolutionary Algorithms (EA) such as SPEA, FFGA, NSGA, NPGA, HLGA and VEGA for all the tested instances. Pareto ant colony optimization algorithm is developed by Doerner et al [15] for solving the portfolio selection problem with the objectives of computation time and the quality of approximated solution space. The experimental results are compared with pareto simulated annealing, and the Non-dominated sorting genetic algorithm. From the literature, it is observed that the Multi-objective ACO algorithm is better than the existing evolutionary algorithms for searching optimal solution. In this paper, multi-objective hybrid max-min ant colony optimization algorithm (MO-HMMAS) is developed for the heterogeneous processor to obtain pareto set solutions, with the two conflicting objectives such as resource objective, and energy objective simultaneously.

3.0 SYSTEM MODEL AND PROBLEM STATEMENT

In this paper, the heterogeneous multiprocessor environment with m preemptive processors $\{P_1, P_2 \ldots\ldots P_m\}$ based on CMOS technology is considered. 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 [2, 6, 7, 11].The energy consumption is calculated by

$$E_{i,j} = Power_{i,j} \approx \left(C_{ef}.\frac{s_{i,j}^2}{k^2}\right).e_{i,j} = \left(\frac{C_{ef}}{K^2}\right).c_i.s_{i,j}^2 \quad (1)$$

where C_{ef} is the effective switching 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 task T_i and c_i is the number of clock cycles to execute a task T_i. From (1), it is understood that, energy consumption is directly proportional to the $c_i s_{ij}^2$. This equation is significant, because the processors operate at different speeds [6, 7].

A set of N periodic tasks $T= \{T_1, T_2, \ldots \ldots\ldots T_N\}$ is considered. T_i is defined as $T_i = \{ w_{i,j}, p_{i,j}\}$ where $w_{i,j}$ is the worst case execution time and $p_{i,j}$ is the period. The task assignment problem considered here is the off-line version, under the condition that utilization of each processor is less than or equal to 1. In this paper, the partitioned scheme is considered for the task assignment and Earliest Deadline First algorithm (EDF) for scheduling the tasks on each processor. The proposed task assignment problem has two objectives.

- The first objective aims at maximizing the number of tasks assigned (resource objective) in the heterogeneous multiprocessor under the condition that the cumulative utilization of any processor does not exceed the utilization bound of the EDF algorithm, which is considered to be NP-hard problem [4, 7, 8].

- The second objective is to minimize the cumulative energy consumption for the task assignment (Energy objective) made by the proposed algorithm. The proposed algorithm is to analyze the tradeoffs between resource utilization of all assigned tasks and cumulative energy consumption simultaneously.

4.0 APPLICATION MODEL

The application model is defined as *A (T, W),* where:

- The set of *N* periodic tasks $T= \{T_1,T_2, \ldots \ldots\ldots T_N\}$ is considered. The tasks are assumed to be mutually independent and inter task communication is not considered.
- The estimated worst case execution time (WCET) is defined as *W* in which $w_{i,j}$ corresponds to the WCET of task T_i on processor P_j. 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 and is calculated by (2)

$$u_{i,j} = \frac{w_{i,j}}{s_{i,j} \times p_i} \tag{2}$$

where $s_{i,j}$ is the speed of processor P_j for a task T_i and p_i is the period of task T_i

Table 1: Utilization Matrix of the Tasks on Three Processors

$u_{i,j}$	P1	P2	P3
T1	0.14	0.13	0.15
T2	0.13	0.14	0.12
T3	0.15	0.14	0.16
T4	0.11	0.13	0.12
T5	0.18	0.15	0.16
T6	0.16	0.15	0.14

Table 2: Random Assignment of Tasks to Processors

Task	1	2	3	4	5	6
Processor	1	3	2	3	2	2

Table 3: Task assignment solutions on heterogeneous processors

Processor	Tasks
P1	T1
P2	T3,T5,T6
P3	T2,T4

Table 1 represents an example of an application model and represents the utilization matrix ($u_{i,j}$), of the heterogeneous multiprocessor system consisting of processors P_1,P_2,P_3 according to (2). Table 3 shows a solution for the three processor task assignment
The objective function is expressed by

$$Maximize \sum_{i \in T(j,s)} TA(s) \tag{3}$$

subject to

$$\sum_{i=1}^{n} u_{i,j}.\Omega_{i,j}^{s} \leq 1 \tag{4}$$

where *TA(s)* is the total number tasks assigned, *T(j, s)* set of tasks assigned to processor *j* and $u_{i,j}$ is the utilization of task T_i on P_j. The equations (3) and (4) describe the total number of EDF- schedulable tasks under the condition that the sum of utilization of each processor is less than or equal to 1[8].

The cumulative energy consumption of all assigned tasks in solution *s* is given by

$$EC(s) = \sum_{j=1}^{m}\left(\sum_{i=1}^{n} E_{i,j}\right) \tag{5}$$

where $E_{i,j}$ is the energy utilized by task T_i on processor P_j and $EC(s)$ represents the energy consumed by the schedule (s). For solving the task assignment problem, the objectives $TA(s)$ is to be maximized and $EC(s)$ is to be minimized simultaneously.

5.0 MULTI-OBJECTIVE OPTIMIZATION

A multi-objective optimization problem can be formally defined as finding all vectors $x = [x_1, x_2, \ldots, x_n]$ which minimize/maximize the vector function $f(x) = [f_1(x), f_2(x), \ldots, f_m(x)]$. For the proposed problem, we have m= 2, where $f_1(x)$ is the maximum number of the task assigned and $f_2(x)$ is the energy consumption of all assigned tasks. As it is not always possible to find a single solution that maximizes resource objective, and minimizes energy objective simultaneously [12,13,14,16]. The main objective of the proposed algorithm is to determine a static distributed assignment with maximizing the number of tasks assigned (resource objective) and minimizing the cumulative energy consumption of all assigned tasks simultaneously.

5.1 Resource Objective

The maximum number of tasks assigned (resource objective) in the heterogeneous multiprocessor is calculated as

$$f_1 = max\ [\ TA(s)\ ] \tag{6}$$

$TA(s)$ is calculated by using (3).

5.2. Energy Objective

The cumulative energy consumption of assigned tasks is calculated as

$$f_2 = min\ [EC(s)] \tag{7}$$

where $EC(s)$ is calculated using (5).

The energy consumption of each task is proportional to the square of processor speed ($E_{i,j} = c_i\ s^2_{i,j}$), whereas its computing capacity consumption is inversely proportional to the processor speed ($u_{i,j} = c_i/(s_{i,j}p_i)$), which means that the energy objective conflicts with the resource objective, and no single solution can optimize both of them simultaneously. To achieve better compromise between both objectives, pareto-based multi-objective hybrid max-min ant colony optimization algorithm is proposed to compute a set of tradeoff optimal solutions in terms of resource objective, and energy objective. The non-linear multi-objective optimization can be defined as

$$F = [Max\ f_1,\ Min\ f_2] \tag{8}$$

5.3 Weighted-Sum Method

In the weighted-sum approach, the resource objective and energy objective are weighted together to produce the weighted-sum objective function as

$$F = \omega f_1 + (1 - \omega)\, f_2 \tag{9}$$

where f_1 is the resource objective, f_2 is the energy objective and ω is the weighting coefficient in the range 0 and 1. Since the objectives have varying range, they are normalized in the range 0 to 1. When $\omega = 1$, only the resource objective is considered and when $\omega = 0$, only the energy objective is accounted. The trade-off between the resource objective and energy objective can be obtained by varying the values of ω [5,13]. In the proposed algorithm resource objective function 'f_1' has to be maximized and energy objective function 'f_2' has to be minimized. In order to define the overall objective function *'F'* clearly, objective function F_2 can be written as

$$f_2 = 1\text{-}EC(s) \qquad f_2 \text{ has to be maximized} \tag{10}$$

So, the overall objective function 'F' is given by

$$Max\ F = [Max\ f_1\ ,\ Max\ f_2] \tag{11}$$

6.0 MAX–MIN ANT SYSTEM [MMAS]

Stützle and Hoos proposed the Max-Min Ant System [19].The key feature of MMAS is that the pheromone trails are updated with only one ant, this ant could be the iteration-best ant or global-best ant which finds the best solution. Moreover, the maximum and minimum values of the pheromones are limited to certain values to escape getting stuck at local solutions. Additionally, pheromone trails initialize to upper bound τ_{max} to have uniform exploration in the whole search space [19]. MMAS can be distinguished from the Ant System (AS) [17,18] mainly in the following three aspects:

(1) Only one single ant adds pheromone after each iteration
(2) The range of possible pheromone trails on each solution component is limited to an interval $[\tau_{min}, \tau_{max}]$
(3) The initial pheromone trails are set to τ_{max}

MMAS uses either the global or iteration best solution for the pheromone trail update. To avoid stagnation, the amount of pheromone is restricted to the range $[\tau_{min}, \tau_{max}]$ [19]. Initially, ants are put on random places. At each construction step, ant k applies a probabilistic action choice rule to choose a next path to visit until a complete solution has been built. Finally, all of the solutions are evaluated and the pheromone updating rule was applied until all the ants have built a complete solution.

After all ants have constructed a tour, pheromones are updated by applying evaporation as in ant system as given in (12).

$$\tau(i, j) = (1-\rho)\tau(i, j) \; \forall \; (i,j) \in N(s) \tag{12}$$

where $0 < \rho < 1$ is the pheromone evaporation rate and $\tau(i, j)$ is the pheromone trail. Followed by the deposit of new pheromone as given in (13)

$$\tau(i, j) = \tau(i, j) + \Delta\tau(i, j)^{best} \tag{13}$$

where $\Delta\tau(i, j)^{best} = 1/f(s)^{best}$; $f(s)^{best}$ as a solution cost of iteration-best $s_{best} = s_{ib}$

The pheromone limits are calculated by

$$\tau_{max} = f(s^{best}) / \rho$$
$$\tau_{min} = \tau_{max} / (\omega . In(\theta+1)) \tag{14}$$

where θ is the sequential number of the current iteration starting with 1, ω is a constant and $\omega \geq 1$. Here ρ is the evaporation rate of pheromone trails and s_{best} denotes the iteration best solution.

Pheromone Update Operator

Initially all pheromone values are set to τ_{max} and after each iteration pheromone limits are updated. Pheromone trails are evaporated by (12). The pheromones associated with the best solution are increased by (13).Then the validity of limits is checked by the algorithm shown below

Algorithm 1: Pheromone update operator
1. Procedure Update_Pheromone( )
2. {
3. if $\tau(i,j) < \tau_{min}$
4. { set $\tau(i,j) = \tau_{min}$ }
5. if $\tau(i,j) > \tau_{max}$
6. { set $\tau(i,j) = \tau_{max}$ }
7. }//end procedure

7.0 PROPOSED MULTI OBJECTIVE HYBRID MAX–MIN ANT SYSTEM (MO-HMMAS) FOR REAL TIME TASK ASSIGNMENT PROBLEM

7.1 Construction Graph and Constraints

For given a set of heterogeneous multiprocessor and task set, each task assigned to one processor by the artificial ant stochastically until each of the tasks is assigned to specific processor without exceeding its computing capacity. The sample construction graph is shown in the Table 4 [2,9,11].

Table 4: Utilization Matrix with '*n*' Tasks '*m*' Processors

	P_1	P_2	P_3	.....	P_{m-1}	P_m
T_1	$u_{1,1}$	$u_{1,2}$	$u_{1,3}$	.....	$u_{1,m-1}$	$u_{1,m}$
T_2	$u_{2,1}$	$u_{2,2}$	$u_{2,3}$	.....	$u_{2,m-1}$	$u_{2,m}$
T_3	$u_{3,1}$	$u_{3,2}$	$u_{3,3}$	.....	$u_{3,m-1}$	$u_{3,m}$
:	:	:	:	:	:	:
:	:	:	:	:	:	:
:	:	:	:	:	:	:
T_{n-1}	$u_{n-1,1}$	$u_{n-1,2}$	$u_{n-1,3}$	.....	$u_{n-1,m-1}$	$u_{n-1,m}$
T_n	$u_{n,1}$	$u_{n,2}$	$u_{n,3}$	.....	$u_{n,m-1}$	$u_{n,m}$

In Table 4, T_i $(1 \leq i \leq n)$ represents the task T_i, P_j $(1 \leq j \leq m)$ represents the j^{th} processor, and $u_{i,j}$ represents the utilization of the i^{th} task on the j^{th} processor. 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 row of utilization matrix represents the estimated utilization value for a specified task on each heterogeneous processor. Similarly, the column of utilization matrix represents the estimated utilization value of a specified processor for each task. The utilization matrix U_{n*m} holds the real numbers in (0, 1) and infinity. If $U_{ij} = \infty$ 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: (i) one and only one cell is visited for each of the rows (ii) the accumulative value of the visited cells in the same column is no greater than 1 [2,6,7,11]. An artificial ant constructs the task assignment solution based on the constraints given by (15)

$$\sum_{i=1}^{n} u_{i,j^*} \Omega_{i,j}^{s} \leq 1 \tag{15}$$

$$\Omega_{i,j}^{s} = \begin{cases} 1 & \text{if square } (T_i, P_j) \text{ is visited in solution } s \\ 0 & \text{otherwise} \end{cases}$$

7.2 Solution Construction

An artificial ant increases the pheromone value $\tau(i,j)$ at the edge between T_i and P_j which represents the possibility of assigning the task T_i on processor P_j. The pheromone values of the ant are initialized as same for solution construction. Each ant builds a tour from a starting pair of task and processor [15]. The probability of selecting the next pair of task and processor is given by

$$p(s,i,j)(t) = \begin{cases} \dfrac{\tau(i,j)(t)}{\sum_{(i',j')\in N(s)} \tau(i',j')(t)} & \text{if } (i,j) \in N(s) \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

In (16), $N(s)$ denotes the set of eligible pairs of (Task, Processor) obtained. $\tau(i, j)$ denotes the pheromone trial of (T_i,P_j).

7.2.1 Exclusion of Heuristic Information

From the literature, the existing ACO algorithms for the heterogeneous environment have included heuristic information *(η(i,j))* [6, 7]. The heuristic information calculation degrades the performance of the ant system when the values of utilization matrix becomes small, thereby heuristic calculation approaches close to the worst case scenario making it difficult for choosing the particular cell for including it in the ant's solution [7]. In order to avoid this, the heuristic information has been excluded in the proposed algorithm. The heuristic information excluded formula is given in (16).

7.2.2 Inclusion of Two Local Searches

The proposed MO-HMMAS algorithm included with two local search procedures for compensating the exclusion of heuristic information to improve the task assignment solution after the construction procedure is completed. The proposed local search algorithm starts with an initial task assignment solution, and then searches for better solutions using 1-OPT and 1-DIFF neighborhood structures [6,7].

Local Search 1: Reducing Average Utilization (1-OPT)
Remove a task from the assigned processor, and then assign it to a different processor only if the overall utilization is reduced (1-OPT). The algorithm for 1-OPT local search procedure shown below

Algorithm 2: Local Search 1 (1-OPT)

```
1. Procedure 1-OPT ( )
2. {
3. for each ant k
4. {
5.    Uavg = (Σ^m_j=1 Uj)/m
6.      for  each task i
7.      {
8.        Remove a task from one processor and assign it to  the neighborhood  processor;
9.        Unew = (Σ^m_j=1 Uj)/m
10.         If Unew<Uavg; new  task assignment is updated;
11.         Elseif Unew>Uavg; old  task assignment is retained;
12.       }//  end for
13. }//end for
14. }//end procedure
```

Local Search 2: Reducing Difference in Utilization (1-Diff)

Remove a task from the assigned processor, and then assign it to a different processor, only if the sum of difference between individual utilization and overall utilization is reduced. The algorithm for 1-Diff local search procedure shown below

Algorithm 3: Local Search 2 (1-Diff)

```
1. Procedure 1-DIFF ( )
2. {
3. for each ant k
4. {
5.    Dorg= Σ^m_j=1 (abs(Unew-Uj))
6.      for each task i
7.      {
8.       Remove a task from one processor and assign it to the  neighborhood processor;
9.       Compute Dnew = Σ^m_j=1 (abs(Unew-Uj))
10.        Unew = (Σ^m_j=1 Uj)/m
11.        if Dnew< Dorg; new task assignment is  updated;
12.        Else if Dnew> Dorg ; old task assignment is retained;
13.          }//end for
14. }//end for
15. }//end Procedure
```

Fig. 1 shows the steps involved in the local search algorithm for each ant. Initially, a solution is constructed randomly and then the local search is applied on to this solution to improve the quality of the solution. The initial solution is given as input to the 1-OPT and local search is performed [7]. Fitness value for the resulting solution is calculated and compared with the previous value. To further improve the quality of the solution, another local search (1-DIFF) is proposed. The local search is repeated until no further improving solution is found.

Flow chart of local search

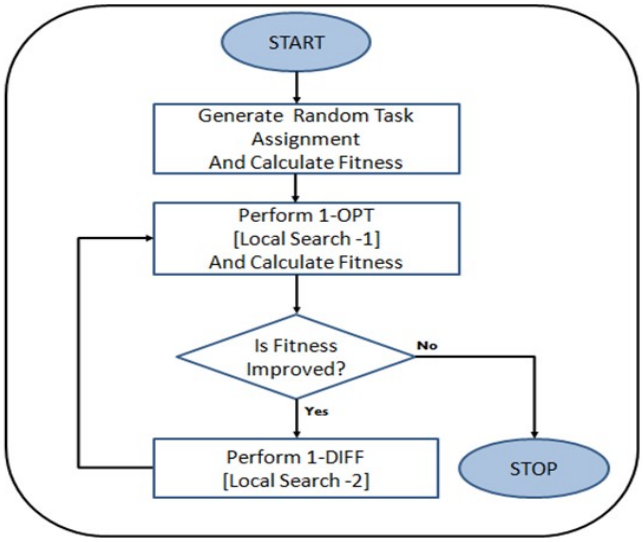

Fig.1 Flowchart of local search

When a solution is constructed, the artificial ants continue to update the pheromone trials by MO-HMMAS according to (17). The pheromone is updated as follows:

$$\tau(i,j) = \begin{cases} (1-\rho)\tau(i,j) + f(s_{best}) & if\ (i,j) \in s_{best} \\ (1-\rho)\tau(i,j) & otherwise \end{cases} \quad (17)$$

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

$$F(s) = (\omega * TA(s)) + ((1-\omega) *(1-(1- EC(s)/MaxEC(s))) \quad (18)$$

$$EC(s) = \sum_{j=1}^{m}\left(\sum_{i=1}^{n} E_{i,j} \times \Omega_{i,j}^{s}\right) \quad (19)$$

$$MaxEC(s) = \sum_{i=1}^{n} c_i \times \max(s_{i,j}^2) \quad (20)$$

where *MaxEC* represents the maximum energy consumed by the schedule and ω is the weighting coefficient.

The MO-HMMAS is implemented as shown below:

Algorithm 4 : Proposed MO-HMMAS algorithm

Input: *Random problem instances*
Output*: Task assignment of T (j, s) = {i |s(i) = j} be the set of tasks mapped to processor j or set of task assignment* or set of task assignment
1. Procedure: *Multi-objective H-MMAS (MO-HMMAS)*
2. {
3. Initialize pareto set , ω=1
4. For (ω=1 ; ω>=0 ; ω= ω-0.1)
5. {
6. Set parameters, Initialize pheromone trials
*7. **While** (termination condition not met) **do***
8. {
9. for i = 1,2… number of ants
10. {
11. Construct solution S_i under the condition U<=1;
12. Apply local search algorithms to improve solution S_i
13. {
*14. **While (**Improving solutions) **do***
15. {
*16. **Procedure 1-OPT ( )***
*17. **Procedure 1-DIFF ( )***
18. }//end while
19. }//end local search
20. }//end for each ant
21. Calculate quality for each solution:
*22. F(s) = (ω * TA(s)) + ((1-ω)* (1- EC(s)/MaxEC(s)))using (18)*
23. Choose the ant with the best fitness value of all ants as the g_{best}
24. $\{ Sib = Sj : f(Sj) = max_{j=1}^{no\ of\ Ants} (f(Sj)) \}$
25. if f (s_{ib}) > f (s_{gb}) then f (s_{gb}) = f (s_{ib});
26. Update pheromone trails of only the g_{best} solution using (17)
*27. **Procedure Update_Pheromone()***
28. }//end- while
29. Update pareto set
30. }// end for each value of ω
31. }//end – procedure

The algorithm is terminated when a pre-specified number of iterations are completed. The algorithm starts with storing all non-dominated solutions in pareto set. The steps are repeated until their stopping criterion (Maximum number of iterations) is met, and then the present solutions are the pareto optimal front solutions [12, 14].

8.0 RESULTS AND DISCUSSION

To analyze the performance of the proposed algorithm MO-HMMAS, 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.

8.1 Data Set Description

The utilization matrix U_{n*m} holds the real numbers in (0, 1). For evaluating the performance of the proposed and existing algorithms, the utilization matrix is generated for considering real-time heterogeneous environment situations based on task heterogeneity, processor heterogeneity and consistency. The utilization matrix is generated as in [2,7]. The steps are given below:

1. A *nx1* clock cycle matrix *C* **(cycle vector)** is generated, the number of cycles to execute task T_i is a random number between [100, 1000].

2. A *nx1* task frequency matrix T_B (**task baseline vector**) is generated, the task frequency of T_i is a random number between [1, Φ_T], here Φ_T is task heterogeneity. It is either High Task heterogeneity (HT: [Φ_T=100]) or Low Task heterogeneity (LT; [Φ_T=5]).

3. A *1xm* **speed vector** is generated for each $T_B(i)$, the speed to execute task T_i on P_j that is $S_i(j)$ to a random number between $[\Phi_T, \Phi_T.\Phi_p]$, here Φ_p is processor heterogeneity; It is either High Processor heterogeneity (HP: $[\Phi p$=20]) or Low Processor heterogeneity (LP: $[\Phi_p$ =5]).

4. An *nxm* **utilization matrix** $U_{i,j}$ is generated by $T_B(i)/ S_i(j)$. Consequently, the values of $U_{i,j}$ are controlled by both task heterogeneity and processor heterogeneity : $U_{i,j} \in [1/ (\Phi_T.\Phi_p), 1]$.

5. The utilization matrix is said to be consistent (*C*), if each speed vector values are sorted by descending with processor P_0 which is always the fastest and processor P_{m-1} as the slowest.This implies that a particular processor always runs at same speed for the entire task (i.e., Processor speed doesn't depend on task characteristics). But the inconsistent matrix (*IC*) holds unsorted speed vector values that are random state as they were generated. This implies that a particular processor runs at different speed for different task (i.e., Processor speed depends on task characteristics). The characteristics of the utilization matrix are varied, considering task heterogeneity, processor heterogeneity, and consistency of evaluating the algorithms for different real-time scenarios. Hence, eight combinations of utilization matrix characteristics are used in this research. The combinations are High and Low Task heterogeneity (HT or LT), High and Low Processor heterogeneity (HP or LP) and Consistent and Inconsistent utilization matrix (*C* or *IC*). This test data set contains namely Consistent, High Φ_T, High Φ_P; Consistent, High Φ_T, Low Φ_P; Consistent, Low Φ_T, High Φ_P; Consistent, Low Φ_T, Low Φ_P; Inconsistent, High Φ_T, High Φ_P; Inconsistent, High Φ_T, Low Φ_P; Inconsistent, Low Φ_T, High Φ_P; and Inconsistent, Low Φ_T, Low Φ_P. For example, C_P4_T100_HT_HP represents the Consistent system with 4 processors and 100 tasks with High Task and High Processor heterogeneity and IC_P5_T150_HT_HP represents the Inconsistent system with 5 processors and 150 tasks with High Task and High Processor heterogeneity.

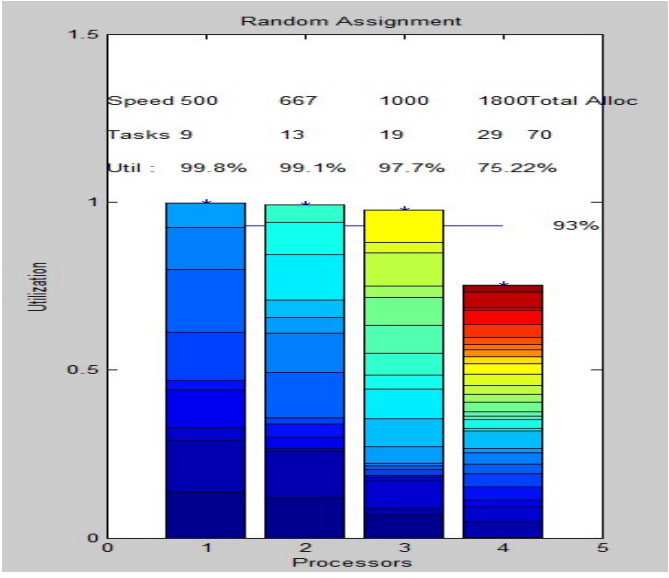

Fig.2: Randomly assigned solution

The task assignment solution in the heterogeneous multiprocessor system begins with the utilization matrix generation as discussed in the implementation section above. For each ant randomly generated the solution which is shown in Fig 2. The local search is performed to find the best solution from among the neighborhood as shown in the flow chart Figure 1. The solution of the ant is shuffled considering one task at a time and checked to reduce the average utilization so as to fill the more number of task as shown in Fig 3.a To further improve the quality of the solution, another local search (1-DIFF) is proposed. In 1-DIFF, the obtained solution is shuffled considering one task at a time and checked to obtain minimum difference for individual processor from the average utilization shown

in Fig 3.b. The remaining un-utilized part of the processors is filled with unassigned tasks if any, again in a random manner, and again, the local search algorithm repeats until no newer tasks can be added. From Table 5, it can be inferred that the number of tasks assigned by 1-DIFF local search is more compared to random assignment and 1-OPT local search.

Table 5: Comparison for random assignment,1-OPT and 1-DIFF local search

	Random Assignment				1-OPT				1-DIFF			
	P1	P2	P3	P4	P1	P2	P3	P4	P1	P2	P3	P4
Utilization Per Processor (%)	99.8	99.1	97.7	75.22	93.8	98.35	94.8	100	99.6	98.35	98.8	99.94
Tasks per Processor	9	13	19	29	9	10	15	42	11	10	25	34
Total Number of Tasks allocated	70				76				80			
Average utilization (%)	93				97				99			

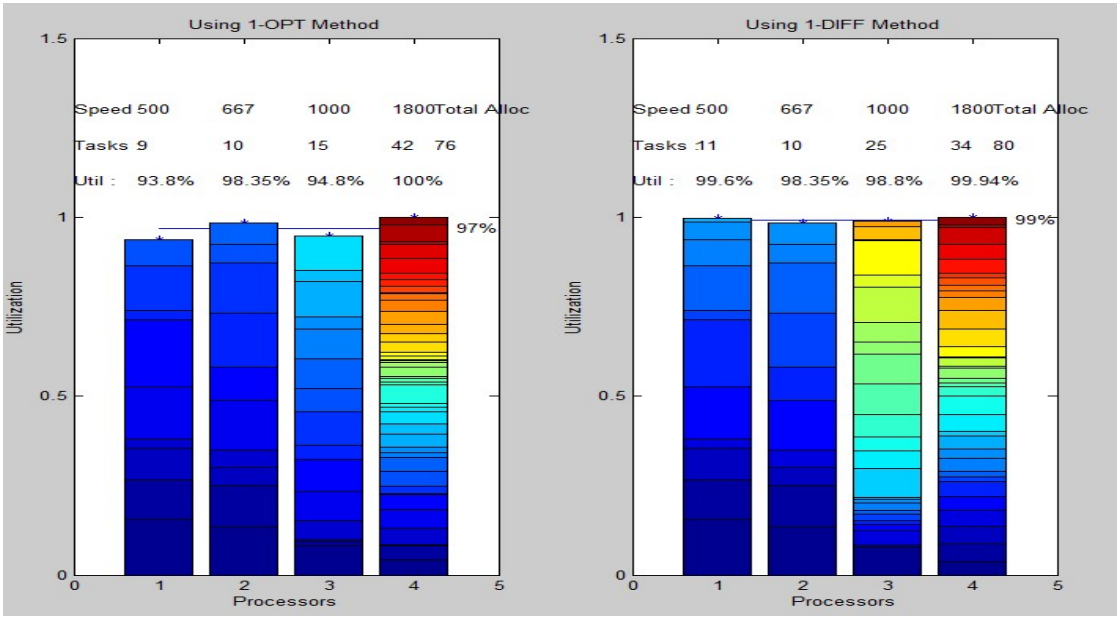

Fig.3: (a) 1-OPT (b) 1-DIFF

8.2 Performance Comparison of the Proposed MO-HMMAS and MO-ACO Algorithm based on Task Heterogeneity, Processor Heterogeneity and Consistency

The utilization matrix is generated for considering real-time heterogeneous environment situations based on task heterogeneity, processor heterogeneity and consistency for evaluating the performance of the proposed MO-HMMAS and MO-ACO algorithm.

8.2.1 Consistency Utilization Matrix

Table 6 gives the results obtained by MO-HMMAS for C_100T_4P_HT_HP instance. Various linearly distributed weights (ω) from 0 to 1 in steps of 0.1 are used. From Table 7, it can be inferred that the best task assigned by MO-HMMAS is 93 whereas by MO-ACO is only 88. Also, the best normalized energy consumption obtained by MO-HMMAS is 0.3328 for 65 tasks, whereas MO-ACO could achieve 0.339 with 61 tasks assigned. The comparison of MO-HMMAS and MO-ACO interms of best resource objective and energy objective for C_100T_4P_HT_HP is given in Table 8. The results show that the MO-HMMAS achieves 6.11% of improvement in the average resource objective and 2.085% of reduction in the average energy objective over the values achieved by MO-ACO for C_100T_4P_HT_HP.

Table 6: Number of task assigned and normalized energy for various values of ω for C_100T_4P_HT_HP of MO-HMMAS

Weight ω	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
Tasks Assigned (Resource objective)	93	90	88	85	82	80	78	73	70	67	65
Normalized Energy (Energy objective)	0.356	0.355	0.353	0.352	0.350	0.349	0.348	0.344	0.339	0.335	0.332

Table 7: Performance comparison of the proposed MO-HMMAS algorithm with the existing MO-ACO algorithm in terms of best resource and energy objective for C_100T_4P_HT_HP problem instance

Objectives	**Best Resource**		**Best energy**	
	MO-HMMAS	**MO-ACO**	**MO-HMMAS**	**MO-ACO**
Resource Objective	93	88	65	61
Energy Objective	0.3564	0.365	0.3328	0.339

Table 8: Comparison of the results obtained by the proposed algorithm with the existing algorithm for C_100T_4P_HT_HP problem instance

Objectives	**Improvement in resource objective by MO-HMMAS(%)**	**Reduction in Energy objective by MO-HMMAS(%)**
Best Resource	5.68%	2.35%
Best energy	6.55 %	1.82%
Average	6.11%	2.085%

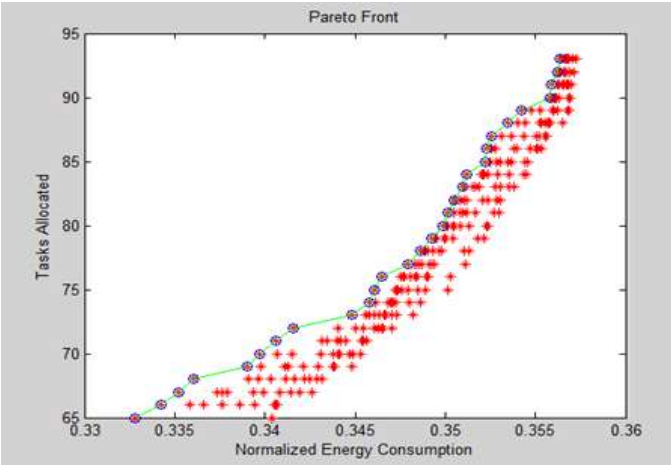

Fig.4 : Search space by MO-HMMAS algorithm for C_100T_4P_HT_HP

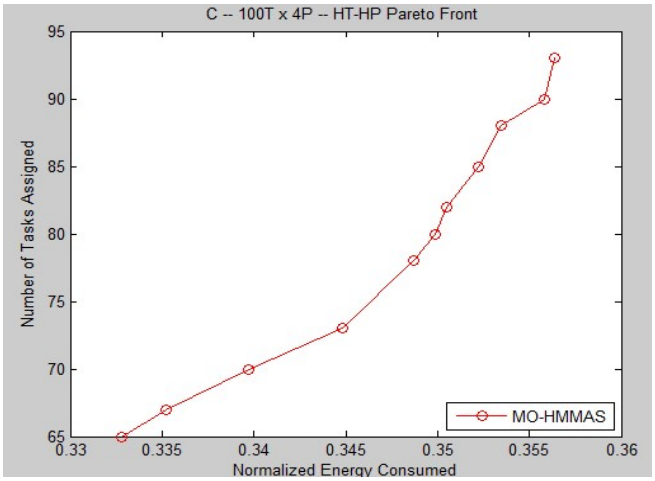

Fig.5 : Pareto front constructed by MO-HMMAS algorithm for C_100T_4P_HT_HP

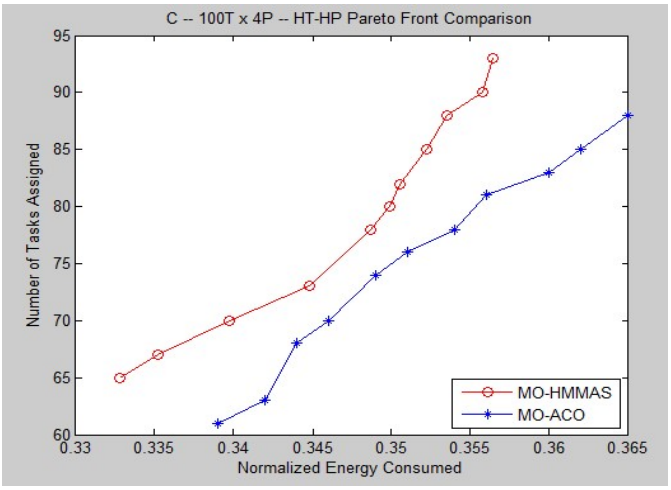

Fig.6: Comparison of pareto tradeoff solutions of MO-HMMAS and MO -ACO for C_100T_4P_HT_HP

Fig 4 and 5 show search space explored by the proposed algorithm MO-HMMAS for C_100T_4P_HT_HP and final converging points known as pareto front. The points on the pareto front are non-dominated solutions with respect to each other. From Fig 6, it can be seen that the proposed MO-HMMAS has outperformed the MO-ACO both in terms of the task assignment and normalized energy consumption for C_100T_4P_HT_HP.

8.2.2 Inconsistency Utilization Matrix

Table 9 shows the results obtained by MO-HMMAS for IC_100T_4P_LT_HP instance. The comparison of MO-HMMAS and MO-ACO for the best resource objective and energy objective for IC_100T_4P_LT_HP is given in Table 11. The results show that the MO-HMMAS achieves 5.16% of improvement in the average resource objective and 5.335% of reduction in the average energy objective compared to MO-ACO. The better performance by MO-HMMAS is due to the fact that MO-HMMAS based on max-min ant System along with the presence of two local search techniques which concentrates more on exploring the neighbors of the good solutions and finds the best value in terms of both resource objective and energy objective.

Table 9: Number of task assigned and normalized energy for various values of ω for IC_100T_4P_LT_HP of MO-HMMAS

Weight ω	**1**	**0.9**	**0.8**	**0.7**	**0.6**	**0.5**	**0.4**	**0.3**	**0.2**	**0.1**	**0**
Tasks Assigned (Resource objective)	100	98	95	93	90	88	85	84	82	81	79
Normalized Energy(Energy objective)	0.0309	0.0294	0.0288	0.0283	0.0276	0.0267	0.0264	0.0262	0.0261	0.0254	0.0239

Table 10 : Performance comparison of the proposed MO-HMMAS algorithm with the existing MO-ACO algorithm in terms of best resource and energy objective for IC_100T_4P_LT_HP problem instance

Objectives	**Best Resource**		**Best energy**	
	MO-HMMAS	**MO-ACO**	**MO-HMMAS**	**MO-ACO**
Resource Objective	100	94	79	76
Energy Objective	0.0309	0.0315	0.0239	0.0262

Table 11 : Comparison of the results obtained by the proposed algorithm with the existing algorithm for IC_100T_4P_LT_HP problem instance

Objectives	**Improvement in resource objective by MO-HMMAS(%)**	**Reduction in Energy objective by MO-HMMAS(%)**
Best Resource	6.38%	1.9%
Best energy	3.94 %	8.77%
Average	5.16%	5.335%

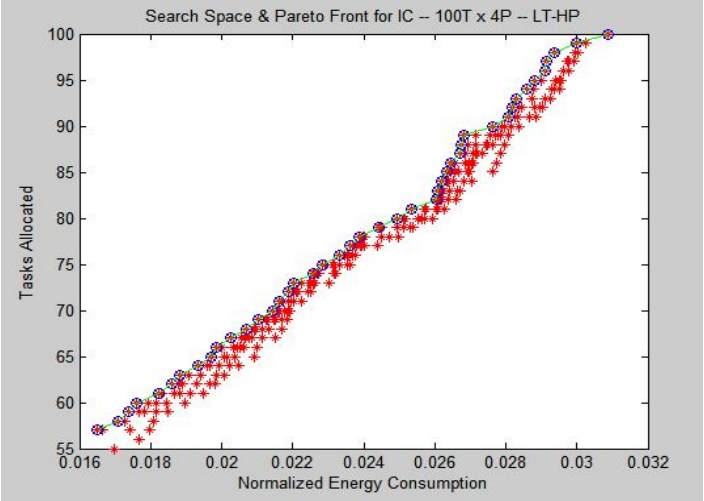

Fig.7: Search space by MO-HMMAS algorithm for IC_100T_4P_LT_HP

.

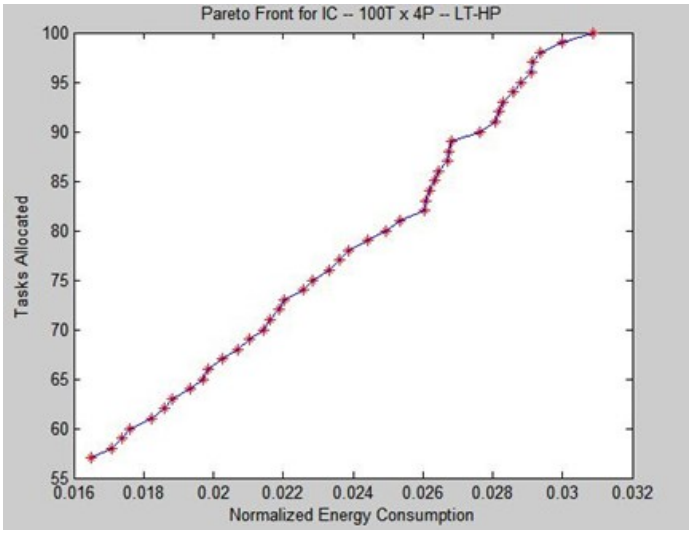

Fig.8: Pareto front constructed by MO-HMMAS algorithm for IC_100T_4P_LT_HP

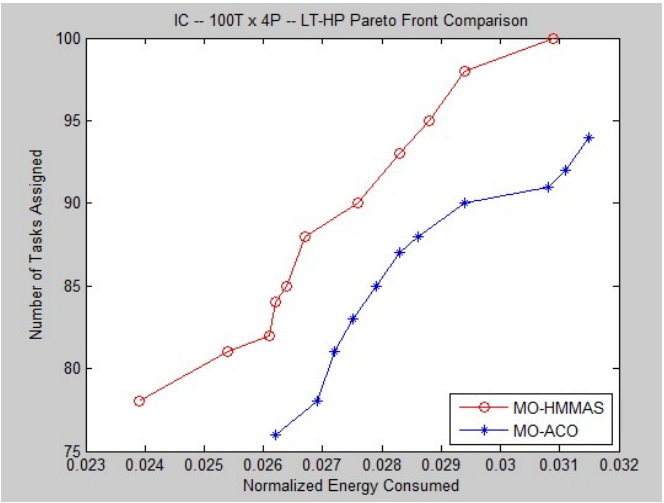

Fig.9: Comparison of pareto tradeoff solutions of MO-HMMAS and MO -ACO for IC_100T_4P_LT_HP

Fig 7 and 8 show the search space explored by the proposed algorithm MO-HMMAS for IC_100T_4P_LT_HP and final converging points known as pareto front. The points on the pareto front are non-dominated solutions with respect to each other. From Fig 9, it can be seen that the proposed MO-HMMAS has outperformed the MO-ACO both in terms of the task assignment and normalized energy consumption for IC_100T_4P_LT_HP.

Table 12: Comparison of the best quality of solution for proposed MO-HMMAS and MO-ACO and their ranks

Problem instance	Size	Proposed MO-HMMAS		MO-ACO		Improvement in resource objective by MO-HMMAS(%)	Reduction in Energy objective by MO-HMMAS(%)	Rank of the solution	
		Resource objective	Energy objective	Resource objective	Energy objective			Proposed MO-HMMAS	MO-ACO
C_HT_HP	U4*100	93	0.3564	88	0.365	5.68	2.35	1	2
C_HT_LP	U8*60	56	0.386	48	0.535	16.66	27.85	1	2
C_LT_HP	U4*80	78	0.232	70	0.345	11.42	32.75	1	2
C_LT_LP	U8*50	47	0.365	40	0.49	17.5	25.51	1	2
IC_HT_HP	U5*150	150	0.023	150	0.0654	0	64.831	1	2
IC_HT_LP	U8*60	60	0.135	60	0.169	0	20.11	1	2
IC_LT_HP	U4*100	100	0.0309	94	0.0315	6.38	1.90	1	2
IC_LT_LP	U8*60	60	0.1228	57	0.1289	5.26	4.73	1	2
IC_LT_LP	U5*20	20	0.125	20	0.2488	0	49.75	1	2
Average						6.98	25.53	1	2

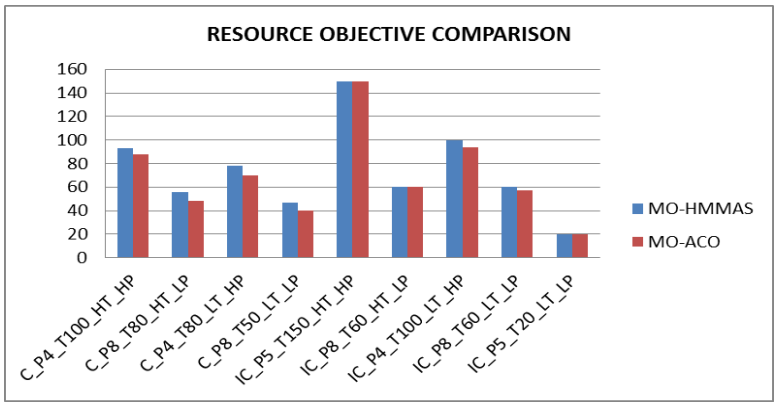

Fig.10: Comparison of the total number of task assigned by the MO-HMMAS and MO-ACO algorithm of consistency and inconsistency matrix

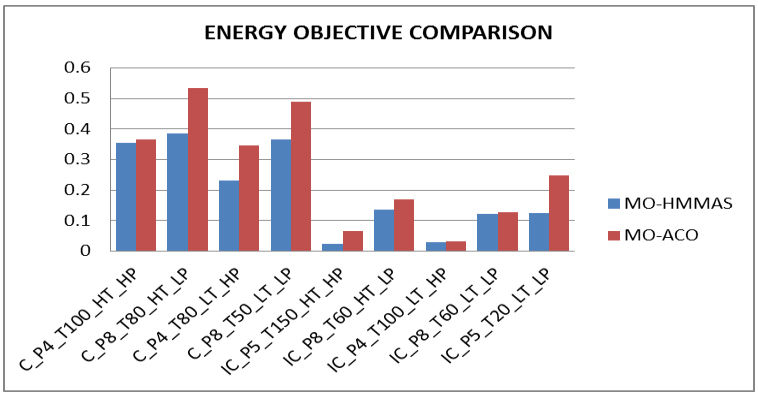

Fig.11: Comparison of normalized energy consumption by the MO-HMMAS and MO-ACO algorithm of consistency and inconsistency matrix

Table 12 shows the best quality of the solution for MO-HMMAS and MO-ACO and their ranks. As it can be seen that the quality of the solution of MO-HMMAS is higher than MO-ACO for all problem instances. In a case of consistency matrix, the proposed algorithm outperforms MO-ACO due to the searching behavior of the ants enriched with local search algorithm. In case of inconsistency matrix, the maximum number of tasks are assigned

similar to MO-ACO. The inclusion of local search has its influence in energy part of the solution. MO-HMMAS achieves an improvement in resource objective and reduction in energy objective by 6.98% and 25.53% respectively over the values achieved by MO-ACO. From the experiment, the proposed MO-HMMAS has proved to be the best algorithm for the task assignment optimization problem in the heterogeneous multiprocessor system. Fig 10 and 11 show comparison of MO-HMMAS and MO-ACO algorithm of consistency and inconsistency matrix in terms of tasks assigned and normalized energy consumption.

8.3 Performance Comparison of H-MMAS and MO-HMMAS

A single objective Hybrid Max-Min Ant System H-MMAS [21] is also implemented that considers both the objectives separately. Table 13 shows the values of resource objective and energy objective for the best solution obtained by the H-MMAS and the proposed MO-HMMAS. The results show that MO-HMMAS produces the best quality of solutions compared to H-MMAS [21]. MO-HMMAS achieves an improvement in resource objective and a reduction in energy objective by 1.85% and 29.75% respectively over the values achieved by H-MMAS.

Table 13: Comparison of H-MMAS and MO-HMMAS algorithm

Problem instance	Size	Proposed MO-HMMAS		H-MMAS [21]		Improvement in resource objective by MO-HMMAS(%)	Reduction Energy objective by MO-HMMAS(%)
		Resource objective	Energy objective	Resource objective	Energy objective		
C_HT_HP	U4*100	93	0.3564	90	0.394	0	9.54
C_HT_LP	U8*60	56	0.386	54	0.4937	3.70	21.81
C_LT_HP	U4*80	78	0.232	77	0.3561	1.29	34.84
C_LT_LP	U8*50	47	0.365	45	0.471	4.44	22.50
IC_HT_HP	U5*150	150	0.023	150	0.0248	0	7.25
IC_HT_LP	U8*60	60	0.135	60	0.2413	0	44.05
IC_LT_HP	U4*100	100	0.0309	98	0.0678	2.04	54.42
IC_LT_LP	U8*60	60	0.1228	57	0.1291	5.26	4.87
IC_LT_LP	U5*20	20	0.125	20	0.397	0	68.51
Average						1.85	29.75

8.4 Performance Comparison of ACO and MO-ACO

A single objective ACO [7] is also implemented and considers both the objectives separately. Table 14 shows the values of resource objective and energy objective for the best solution obtained by the single objective ACO and the MO-ACO [16]. The results show that MO-ACO produces the best quality of solutions. MO-ACO achieves an improvement in resource objective and a reduction in energy objective by 1.79% and 20.096 % respectively over the values achieved by ACO.

Table 14: Comparison of MO-ACO and ACO

Problem instance	Size	MO-ACO		ACO		Improvement in resource objective by MO-ACO(%)	Reduction Energy objective by MO-ACO(%)
		Resource objective	Energy objective	Resource objective	Energy objective		
C_HT_HP	U4*100	88	0.365	75	0.5418	17.33	32.631
C_HT_LP	U8*60	48	0.535	48	0.561	0	4.63
C_LT_HP	U4*80	70	0.345	68	0.4553	2.94	24.22
C_LT_LP	U8*50	40	0.49	40	0.53	0	7.54
IC_HT_HP	U5*150	150	0.0654	150	0.0749	0	12.68
IC_HT_LP	U8*60	60	0.169	60	0.1955	0	13.55
IC_LT_HP	U4*100	94	0.0315	98	0.0493	-4.08	36.105
IC_LT_LP	U8*60	57	0.1289	57	0.1311	0	1.67
IC_LT_LP	U5*20	20	0.2488	20	0.4841	0	48.60
Average						1.79	20.096

8.5 Performance Comparison of the Proposed MO-HMMAS & H-MMAS with the Existing MO-ACO & ACO Algorithms

Fig 12 and 13 show the comparison of proposed MO-HMMAS algorithm with the existing algorithms in terms of resource objective and energy objective for both consistent and inconsistent systems. The better performance by MO-HMMAS is because MO-HMMAS concentrates on exploring the neighbors of the good solutions using two local search techniques and finds the best value in terms of both resource objective and energy objective.

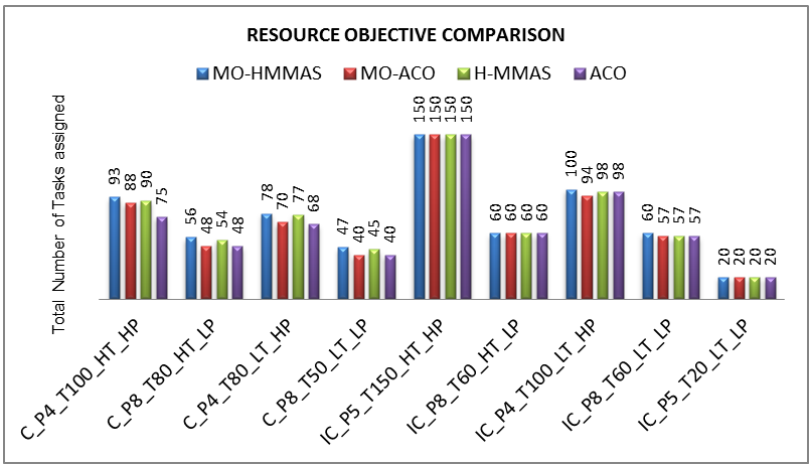

Fig. 12: Comparison of the total number of task assigned by the proposed algorithm with the existing algorithms for consistency and inconsistency matrix

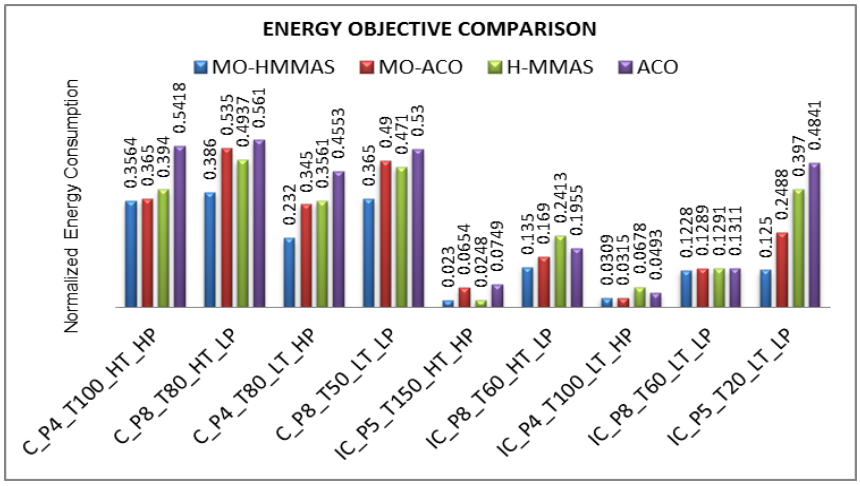

Fig. 13: Comparison of normalized energy consumption of assigned tasks by the proposed algorithm with the existing algorithms for consistency and inconsistency matrix

9. 0 CONCLUSION

In this paper, the Multi objective Hybrid Max-Min Ant Colony Optimization Algorithm (MO-HMMAS) is proposed for solving the real-time task assignment problem in the heterogeneous multiprocessors. This paper considers the two conflicting objectives of maximizing the number of tasks assigned (resource objective) and to minimize the cumulative energy consumption for the task assignment (energy objective) simultaneously. The best values for resource objective and energy objective obtained using proposed MO-HMMAS algorithm for randomly generated problem instances are reported, and compared with MO-ACO. The proposed algorithm outperforms MO-ACO due to the searching behavior of the ants enriched with hybridization of local search algorithm. Results showed that MO-HMMAS algorithm is well-suited for obtaining a good pareto optimal solutions for the task assignment problem in the heterogeneous multiprocessors by considering the above two objectives. The performance of MO-HMMAS is improved with the hybridization of two local search techniques with Max–Min Ant System. In our future work, we will investigate the possibility of MO-HMMAS for the heterogeneous multiprocessor platform, explore inter-task communication on the task sets and explore other heuristic algorithms such as NSGA- II and NPSO to achieve better results.

REFERENCES

[1] M.Poongothai, “ARM Embedded Web Server Based on DACS System”, *IEEE International Conference on Process Automation, Control and Computing (ICPAC11),* 2011, pp.50-51.

[2] M. Poongothai, A.Rajeswari and V.Kanishkan, “A heuristic based real time task assignment algorithm for the heterogeneous multiprocessors”, *IEICE Electronic Express*, Vol 11, No.3, 2014, pp.1-9.

[3] M.R.Yoo and M.Gen,“Scheduling algorithm for real-time tasks in heterogeneous multiprocessors system using multiobjective hybrid genetic algorithm”, *Journal of Computers and Operations Research,* Vol.34, No.10, 2007, pp.3084–3098.

[4] M.R. Garey and D.S. Johnson, *Computers and Intractability*: *A Guide to the Theory of NP-Completeness*, W.H. Freeman and Company, New York, 1979.

[5] P. Chitra, R. Rajaram and P. Venkatesh, “Application and comparison of hybrid evolutionary multiobjective optimization algorithms for solving task scheduling problem on heterogeneous systems”, *Applied Soft Computing*, Vol.11, 2011, pp.2725–2734.

[6] H. Chen and A.M.K. Cheng, "Applying ant colony optimization to the partitioned scheduling problem for heterogeneous multiprocessors", *WIP session*, IEEERTAS, 2005.

[7] H.Chen, A.M.K. Cheng and Y.W. Kuo, "Assigning real-time tasks to heterogeneous processors by applying ant colony optimization", *Journal of Parallel Distributed Computing*, Vol.71,2011 pp.132-142.

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

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

[10] Umarani and G. Srikanth, "Tasks Scheduling using Ant Colony Optimization", *Journal of Computer Science*, Vol.8, No.8,2012, pp.1314-1320.

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

[12] Jan Madsen, Thomas K. Stidsen, Peter Kjaerulf and Shankar Mahadevan," Multi-Objective Design Space Exploration of Embedded System Platforms From Model-Driven Design to Resource Management for Distributed Embedded Systems", *International Federation for Information Processing,* Vol.225, 2006, pp 185-194.

[13] MyungryunYoo and Mitsuo Gen, "Scheduling algorithm for real-time tasks using multiobjective hybrid genetic algorithm in heterogeneous multiprocessors system", *Computers & Operations Research,* Vol.34, 2007, pp.3084 – 3098.

[14] A. Qazi, R. G. Raj, M. Tahir, M. Waheed, S. U. R. Khan, and A. Abraham, "A Preliminary Investigation of User Perception and Behavioral Intention for Different Review Types: Customers and Designers Perspective," *The Scientific World Journal*, Vol. 2014, Article ID 872929, 8 pages, 2014. doi:10.1155/2014/872929.

[15] K. F. Doerner, W. J. Gutjahr, R. F. Hartl, C. Strauss, and C. Stummer, "Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection," *Annals of Operations Research*, Vol. 131, 2004, pp.79–99.

[16] I. Alaya, C. Solnon, and K. Ghedira, "Ant colony optimization for multi-objective optimization problems," *19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007)*, *Los Alamitos, CA: IEEE Computer Society Press*, Vol.1, 2007, pp.450–457.

[17] M. Dorigo and T. Stützle, *Ant Colony Optimization*, *MIT Press*, Cambridge, 2004. doi:10.1007/b99492

[18] M.Dorigo, V.Maniezzo and A., Colorni, "Ant System: Optimization by a colony of cooperating agents", *IEEE* transactions on Systems, Man, and Cybernetics, part B26, Vol.1, 1996, pp.29–41.

[19] T. Stützle and H. H. Hoos, The MAX–MIN Ant System and local search for the traveling salesman problem" *IEEE International Conference on Evolutionary Computation (ICEC'97),* IEEE Press, Piscataway, NJ, 1997. pp.309–314.

[20] 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, 2001, pp.810-837.

[21] M.Poongothai & A. Rajeswari, "Application of hybrid meta-heuristic algorithm for assigning real-time tasks to heterogeneous processors". 6th IEEE International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2015, pp. 1-7.