ELectronics EXpress

LETTER

IEICE Electronics Express, Vol.11, No.3, 1–9

A heuristic based real time task assignment algorithm for heterogeneous multiprocessors

M Poongothai[a)], **Dr A Rajeswari, and V Kanishkan**

Department of ECE, Coimbatore Institute of Technology, Coimbatore 641014 India

a) *poongothai@cit.edu.in*

Abstract: In this paper a Heuristic Based Improved Linearly Decreasing Weight Particle Swarm Optimization algorithm (ILDW-PSO) is proposed to solve real time task assignment in heterogeneous processors. To achieve better quality solution to the particles, existing LDW-PSO encompasses to ILDW- PSO, which includes Genetic algorithm mutation operator. The objective is to minimize the maximum utilization of the processors with energy aware load balance condition which reduces the energy consumption. Experimental results show that ILDW-PSO outperforms the existing PSO algorithms interms of maximum utilization and normalized energy consumption for both consistent utilization matrix and inconsistent utilization matrix.

Keywords: multiprocessors, task assignment, heterogeneous processors, particle swarm optimization, real time systems

Classification: Integrated circuits

References

[1] D. P. Bunde: J. Scheduling **12** (2009) 489.

[2] J. Kennedy and R. C. Eberhart: Proc. IEEE Int. Conf. Neural Networks (1995) 1942.

[3] R. Mall: *Real-Time Systems Theory and Practice* (Pearson Education, 2007).

[4] P. Visalakshi and S. N. Sivanandam: Int. J. Open Problems Compt. Math **2** (2009) 475.

[5] A. Omidi and A. M. Rahmani: Proc. Int. Conf. Computer Science and Information Technology (ICCSIT) (2009) 369.

[6] S. Baruah:Proc. Int. Conf. Parallel Processing (ICPP) (2004) 467.

[7] T. Braun, et al.: J. Parallel . Distributed Computing **61** (2001) 810.

[8] M. B. Abdelhalim: Proc. Int. Conf. Computer and Electrical Engineering (2008) 23.

[9] H. Chen, A. M. K. Cheng and Y.W Kuo: J. Parallel. Distributed Computing **71** (2011) 132.

[10] Ms. M. Poongothai: Proc. Int. Conf. Process Automation, Control and Computing (ICPAC) (2011) 1.

[11] P.Y Yin, S.S Yu, P.P Wang and Y.T Wang: Computer Standards & Interfaces **28** (2006) 441.

[12] Y. Shi and R. C. Eberhart: Proc. Int. Conf. Evolutionary Computation

DOI: 10.1587/elex.11.20130975
Received December 09, 2013
Accepted December 17, 2013
Publicized January 14, 2014
Copyedited February 10, 2014

(1999) 1945.

1 Introduction

Real-time scheduling plays an important role in design of real-time Embedded systems, because most of the scheduling algorithms are implemented in precarious applications. The heterogeneous multiprocessor platform meets the computational demands for various real-time applications. Real-time embedded systems become a more complicated architecture if it includes many heterogeneous components, because every application requires different execution times up on different processors. So choice of task scheduling algorithm is very important for the accurate functioning of the real-time systems. Assigning the real-time tasks to the heterogeneous computing environment is very difficult, because the complexity increases in such a way that every real time application is in need of different execution times up on heterogeneous processors [10, 1]. Hence, Task assignment in a heterogeneous multiprocessor is a combinatorial problem which is an NP hard problem [9]. It can be solved by heuristics algorithms which are used to get near optimal solution. In this paper, we have proposed Heuristic Based Improved Linearly Decreasing Weight Particle Swarm Optimization algorithm for assigning Real Time Tasks in the Heterogeneous Multiprocessors for finding a solution for task assignment to the heterogeneous processors without exceeding the processors computing capacity and fulfilling the dead line constraints.

2 Related work

Braun et al. [7] proposed a comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing system for solving the heterogeneous multiprocessor task partitioning problem and the objective is to minimize the make span. He has compared eleven static Heuristics for scheduling class of independent tasks onto heterogeneous distributed environment. In [6] Baruah proposed a polynomial time algorithm to determine a feasible mapping solution of a given set of tasks among the processors of multiprocessor platform such that the capacity of any individual processors is not exceeded. The aim in this paper is to determine feasible assignment solution only if the problem instance satisfies certain constraints. In [8] M. B. Abdelhalim proposed off line version of the task assignment problem using Re- Excited Particle Swarm Optimization for assigning and scheduling a set of periodic tasks onto a heterogeneous processors without violating its computing capacity constraint. Energy aware cost function has considered for solving load balance problem. The results were compared with GA, ACO, and PSO heuristics and proved that the Re-excited PSO performs better than the other heuristics. Chen, H., A. M. K. Cheng and Y. W. Kuo [9] proposed assigning real-time tasks to heterogeneous processors by applying ant colony optimization to improve task assignment solution. The results were also compared with GA and LP based approches.

DOI: 10.1587/elex.11.20130975
Received December 09, 2013
Accepted December 17, 2013
Publicized January 14, 2014
Copyedited February 10, 2014

3 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 system model described by Chen, H., A. M. K. Cheng and Y. W. Kuo [9] is considered in this paper. 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 execution time for each task on processor and speed of processor for a particular task can be expressed as [8, 9]:

$$e_{i,j}=c_i/s_{i,j} \tag{1}$$

where $e_{i,j}$ is the execution time for a task T_i on processor P_j; $s_{i,j}$ is the speed of P_j for a task T_i; c_i is the number of clock cycles to execute a task T_i.

The power consumption of Ti on Pj per period is expressed as

$$\text{Power}_{i,j} = C_{ef}.s_{ij}^3/k^2 \tag{2}$$

The energy consumption is expressed as

$$\text{Energy}_{i,j} = \text{Power}_{i,j} \,.\, e_{i,j} = (C_{ef}.s_{ij}^3/k^2) \,.\, e_{i,j} = (C_{ef}\,/\,k^2) \,.\, c_i.\, s_{ij}^2 \tag{3}$$

where C_{ef} is the effective switch capacitance related to tasks and k is the constant [8, 9]. From the equation (3) 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.

A set of N periodic tasks $T= \{\zeta_1, \zeta_2, \ldots \ldots\ldots \zeta_N\}$ is considered. The tasks are assumed to be mutually independent and inter task communication is not considered [8, 9]. ζ_i is defined as $\zeta_i = \{e_{i,j}, p_{i,j}\}$ where $e_{i,j}$ is its 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 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 value of $u_{i,j}$ is equivalent to $e_{i,j/}\ p_i$. 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. For example the sample utilization matrix is shown in Table I for 5 tasks on 4 processors environment:

Table I. : Utilization matrix with 5 tasks on 4 processors environment

Task _id	P_1	P_2	P_3	P_4
T_1	$U_{1,1}$	$U_{1,2}$	$U_{1,3}$	$U_{1,4}$
T_2	$U_{2,1}$	$U_{2,2}$	$U_{2,3}$	$U_{2,4}$
T_3	$U_{3,1}$	$U_{3,2}$	$U_{3,3}$	$U_{3,4}$
T_4	$U_{4,1}$	$U_{4,2}$	$U_{4,3}$	$U_{4,4}$
T_5	$U_{5,1}$	$U_{5,2}$	$U_{5,3}$	$U_{5,4}$

The above heterogeneous multiprocessor system model is used to find feasible assignment of each task from periodic task set to the specific processor in the heterogeneous multiprocessor environment under the condition of utilization that each processor is less than or equal to 1.

DOI: 10.1587/elex.11.20130975
Received December 09, 2013
Accepted December 17, 2013
Publicized January 14, 2014
Copyedited February 10, 2014

In this paper, we have considered partitioned scheme for task assignment and Earliest Deadline First algorithm for scheduling the tasks on each processor [3]. The partitioning scheme reduces the multiprocessor scheduling problem to a set of uniprocessor ones, which helps to implement uniprocessor scheduling algorithm on each processor. The uniprocessor EDF scheduling algorithm [3] is applied to each processor for scheduling all tasks assigned to it. For a set of periodic real-time tasks $\{T_1, T_2, \ldots, T_n\}$, EDF schedulability criterion can be expressed as:

$$\sum_{\mathrm{i}=1}^{n} \frac{e_i}{\mathrm{P}_i} = \sum_{i=1}^{n} u_i \leq 1 \tag{4}$$

where u_i is average utilization due to the task T_i and n is the total number of tasks in the task set. The necessary and sufficient condition for a set of tasks to be EDF schedulable is given by equation (4). This means that, if a set of tasks is not schedulable under EDF, then no other scheduling algorithm can feasibly schedule this task set. Hence, EDF has been proven to be an optimal uniprocessor scheduling algorithm.

4 Proposed Improved Linearly Decreasing Weight Particle Swarm Optimization algorithm (ILDW-PSO)

In this section, an Improved Linearly Decreasing Weight PSO algorithm (ILDW-PSO) to allocate tasks to specific processor in heterogeneous multiprocessor system is presented. The proposed ILDW-PSO task assignment algorithm focuses on minimize the maximum utilization that is, finding for a solution in which each of the tasks assigned to a specific processor under the condition when utilization ≤ 1 (without exceeding the processors capacity). But the problem in that is different solutions produce the same utilization but some of the solutions may not satisfy the load balance between the processors which will affects the energy consumption. Hence we considered energy aware load balance condition between the processors which reduces the energy consumption. In our proposed algorithm, the objective function is to minimize the maximum utilization under the constraints of utilization bound condition for the EDF algorithm and energy aware load balance condition for minimizing the cumulative energy consumption of all assigned tasks in a solution.

The quality of solution is evaluated by the fitness function F(s) and is given by

$$\mathrm{F(s)} = \min\{\max(Ui)\} + \text{penalty} \quad \text{For} \quad i = 1, 2, 3.....m \tag{5}$$

$$\text{penalty} = \sum (Ui > 1) \quad \text{For} \quad i = 1, 2, 3.....m \tag{6}$$

The equation (5) describes the minimization of maximum utilization. If the particle solution exceeds the processing capacity of the processor, penalty is added to the F(s) which is given in equation (6). In our proposed algorithm, the fitness function evaluates the i^{th} particle and provides the solution based on equation (7) and (8) which are described as

$$\mathrm{f_o(i)} = \mathrm{F(s_i)} \tag{7}$$

DOI: 10.1587/elex.11.20130975
Received December 09, 2013
Accepted December 17, 2013
Publicized January 14, 2014
Copyedited February 10, 2014

$$f_{EC}(i) = \frac{\sum_{i=1}^{m} \frac{Ui}{m} + \max(Ui)}{2} \quad (8)$$

In which m is the number of processors and Ui is equal to the i^{th} processor's utilization.

In equation (7), $fo(i)$ is the value of the objective function to minimize the maximum utilization . In equation (8), $f_{EC}(i)$ is the energy aware load balance condition which accounts for energy reduction. The purpose of $f_{EC}(i)$ is that to obtain minimum utilization and at the same time load balance is satisfied. The first one is given precedence over the second one. In equation (8), the first term represents the energy cost while the second term represents the timing cost [8]. Our proposed ILDW-PSO algorithm minimizes both fo and f_{EC}.

Linearly Decreasing Weight Particle Swarm Optimization (LDW-PSO) is introduced by Shi and Eberhart in which a linearly decreasing inertia weight factor $wLDW$ is included into the velocity of the updated equation in the PSO instead of w [12]. Compared to original LDW-PSO, the performance of the proposed ILDW-PSO is effectively improved because the global and local search capabilities of the particle are balanced and includes Genetic Algorithm Mutation Operator used to search better quality solutions. In ILDW-PSO, $wLDW$ is the inertia weight which linearly decreases from 0.9 to 0.1 through the search process [11]. The updated velocity and position value for ILDW-PSO can be calculated as follows:

$$V_{id} = wLDW.V_{id} + C_1.Rand(\,)(pbest_{id} - X_{id}) + C_2.Rand(\,)(gbest_{id} - X_{id}) \quad (9)$$

$$X_{id} = X_{id} + V_{id} \quad (10)$$

Linearly decreasing weight inertia equation is written as:

$$wLDW = wstart + (wstart - wend).\beta \quad (11)$$

$$\beta = \left\{ \frac{1}{1 + itr.\alpha} \right\} \quad (12)$$

$$\frac{}{Total\ itr}$$

itr = current iteration; Total itr = maximum iteration

- $c1,c2$: The balance factors between the effect of self-knowledge and social knowledge in moving the particle towards the target. Usually, the value 2 is suggested for both factors in the (literature) [4, 11]
- Rand() : A random number between 0 and 1, and different at each iteration
- $WLDW$: Linear Decreasing Inertia weight
- $pbest_{id}$: The personal best position of the particle
- $gbest_{id}$: The best position within the swarm
- V_{id} : The current velocity of the particle
- X_{id} : The current position of the particle

DOI: 10.1587/elex.11.20130975
Received December 09, 2013
Accepted December 17, 2013
Publicized January 14, 2014
Copyedited February 10, 2014

Pseudo Code for the **ILDW-PSO** algorithm:

Initialize the population randomly.
Initialize the velocity for each particle as NULL.
DO
{
For each particle
{
Evaluate fitness of particle swarm.
The average of the fitness value is calculated and swap mutation is applied to the particles whose fitness is greater than the average fitness value.
Compare the fitness value and find best particle swarm.
Find pbest and gbest
Update particle velocity:
$V_{id} = wLDW.V_{id} + C_1.Rand()(pbest_{id} - X_{id}) + C_2.Rand().(gbest_{id} - X_{id})$
Update particle position:
$X_{id} = X_{id} + V_{id}$
}
Choose the particle with the best fitness value of all particles as the gbest.
}
Until termination criterion is met.

5 Experimental results and discussion

5.1 The problem environment

To test our proposed algorithm ILDW-PSO, experiments are performed on an AMD A4-3330MX APU processor running at 2.20 GHZ with 2 GB RAM. The operating system is MS Windows 7, 64 bit running the MATLAB R2012a environment. The performance of ILDW-PSO algorithm is compared with the existing methods, PSO-Fixed and Re-excited PSO, in terms of system utilization, and normalized energy. In Re-excited PSO w starts at 1 and decreases linearly until reaching 0. This algorithm depends on re-exciting new randomized particle velocities at the beginning of each round, while keeping the particle positions obtained so far, it allows another round of domain exploration [8]. We have tested the proposed ILDW-PSO algorithm on a variety of randomly generated problem sets. A problem set is characterized by the utilization matrix U_{n*m}. It holds the real numbers in (0, 1). The optimal solution for task assignment problem and increase in the convergence rate has been achieved in our experiment at the following parameters. Population size =100; Maximum iteration=100; $c1=c_2=1.5$; inertia weight w for PSO-Fixed =0.7; $wLDW$ for ILDW-PSO is dynamically calculated as given in equations (11) and (12) and $\alpha = 4$, $wstart = 0.9$, $wend=0.1$, mutation rate μ_m=0.02. The particle's velocity on each dimension is restricted by a maximum velocity v_{max}, which controls the maximum travel distance during each iteration to avoid the particle flying past good solutions. For initialization of particle velocity, velocity vector v_{ij} is set to the random number from $v_{\min}$ to $v_{\max}$. During a algorithm run, velocity vector is limited to the range $[v_{\min}, v_{\max}]$. So v $_{min}$ is set to -5, and $v_{\max}$ is set to 5.

DOI: 10.1587/elex.11.20130975
Received December 09, 2013
Accepted December 17, 2013
Publicized January 14, 2014
Copyedited February 10, 2014

5.2 Performance evaluation of the ILDW-PSO algorithm based on task heterogeneity, processor heterogeneity and consistency:

The existing Re-excited PSO algorithm [8] has not considered task heterogeneity, processor heterogeneity, and consistency. In this experiment, 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 and existing algorithms. The utilization matrix is generated as in [9]. The steps are given below:

1. An *nx1* clock cycle matrix C is generated, the number of cycles to execute task Ti is a random number between [100, 1000].
2. An *nx1* task frequency matrix T_B is generated, the task frequency of T_i is a random number between $[1,\Phi_T]$, here Φ_T is task heterogeneity. It may be 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 Ti on Pj that is $S_i(j)$ to a random number between $[\Phi_T, \Phi_T. \Phi_p]$, here Φ_p is processor heterogeneity; it may be High Processor heterogeneity (HP) or Low Processor heterogeneity (LP). For High processor Φ_p=20 and for Low Processor Φ_p =5.
4. An *nxm* utilization matrix $U_{i,j}$ is generated by $T_B(i)/\ S_i(j)$ and $U_{i,j}$€ $[1/\ (\Phi_T. \Phi_p),1]$
5. The utilization matrix is said to be consistent if the each speed vector values are sorted by descending. But the inconsistent matrix holds unsorted speed vector values.

The maximum utilization of PSO-Fixed, Re-excited PSO is higher than the proposed ILDW-PSO algorithm for different problem instances. The results show that the maximum utilization of ILDW-PSO is minimized for both consistent and inconsistent matrix compared to the existing approaches. The results of the comparison are shown in Table II.

Table II. comparison of Maximum Utilization for the three algorithms of consistent and consistent matrix

Problem set	Size	PSO-Fixed	Re-excited PSO	ILDW-PSO
IC_LT_HP	U4*100	0.9320	0.9822	0.9209
IC_LT_LP	U8*60	0.8393	0.8472	0.8333
IC_HT_LP	U8*60	0.8298	0.8230	0.7818
IC_HT_HP	U5*150	0.7990	0.7854	0.7471
C_LT_LP	U8*50	0.8917	0.9111	0.8611
C_LT_HP	U4*80	0.9011	0.9429	0.8848
C_HT_HP	U4*100	0.8891	0.9829	0.8585
C_HT_LP	U8*60	0.8636	0.9524	0.8333

The test results from our experiments for normalized energy consumption for the three algorithms of consistent and inconsistent matrix are shown in Table III. As it can be seen that the ILDW-PSO performs better

DOI: 10.1587/elex.11.20130975
Received December 09, 2013
Accepted December 17, 2013
Publicized January 14, 2014
Copyedited February 10, 2014

for all problem instances interms of energy consumption. But, the execution of the proposed ILDW-PSO algorithm takes slightly more time to solve each problem instances compared to PSO-Fixed and Re-excited PSO. The results of the comparison are shown in Table IV. The reason is that the proposed algorithm required more number of computations to obtain optimal solution. However this paper is considered as the off line version of task assignment problem, so the increase in CPU execution time can be tolerable.

Table III. Comparison of normalized energy consumption for the three algorithms of consistent and inconsistent matrix

Problem set	Size	PSO-Fixed	Re-excited PSO	ILDW-PSO
IC_LT_HP	U4*100	0.9253	0.8938	0.8187
IC_LT_LP	U8*60	0.8082	0.7994	0.8115
IC_HT_LP	U8*60	0.7990	0.7854	0.7471
IC_HT_HP	U5*150	0.8677	0.8636	0.8455
C_LT_LP	U8*50	0.8434	0.8525	0.8505
C_LT_HP	U4*80	0.8929	0.8897	0.8642
C_HT_HP	U4*100	0.8518	0.8784	0.8358
C_HT_LP	U8*60	0.8118	0.8370	0.7982

Table IV. Comparison of average CPU time (secs) by the three algorithms to execute eight different problem instances

Problem set	Size	PSO-Fixed	Re-excited PSO	ILDW-PSO
IC_LT_HP	U4*100	9.1197	9.2197	10.2649
IC_LT_LP	U8*60	8.5489	8.7985	9.8437
IC_HT_LP	U8*60	8.6269	8.7517	9.8749
IC_HT_HP	U5*150	8.5489	9.0325	10.3585
C_LT_LP	U8*50	9.1105	9.1885	9.8125
C_LT_HP	U4*80	9.0013	9.3757	9.8905
C_HT_HP	U4*100	8.6737	9.2665	10.1245
C_HT_LP	U8*60	10.9981	11.1853	11.6533

6 Conclusion

In this paper, we have proposed Heuristic Based Improved Linearly Decreasing Weight Particle Swarm Optimization algorithm (ILDW-PSO) for solving task assignment problem in heterogeneous multiprocessor platform. and its performance is compared with PSO-Fixed and Re-excited PSO algorithms. Our proposed algorithm is proven feasible task assignment with the objective of minimizes the maximum utilization and distribute the tasks to the processor equally, at the same time energy minimization is also improved. The experimental results show that the proposed ILDW-PSO outperforms PSO-Fixed and Re-excited PSO algorithms interms of maximum utilization and normalized energy. The reason

DOI: 10.1587/elex.11.20130975
Received December 09, 2013
Accepted December 17, 2013
Publicized January 14, 2014
Copyedited February 10, 2014

is that ILDW-PSO includes Genetic algorithm mutation operator for finding better quality solutions to the particles, and the global and local search capabilities of the particle are balanced by inertia weight $wLDW$. But the execution time of proposed ILDW-PSO algorithm is slightly increased for solving each problem instances compared to PSO-Fixed and Re-excited PSO.

In our future work, we will work on reducing execution time and we will investigate the possibility of ILDW-PSO for heterogeneous multiprocessor platform, explore task precedence, and inter-task communication on the task sets and explore other heuristic algorithms to achieve better results.

DOI: 10.1587/elex.11.20130975
Received December 09, 2013
Accepted December 17, 2013
Publicized January 14, 2014
Copyedited February 10, 2014