ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY
January 2020, Vol 13(01), 85 – 90
DOI: 10.17485/ijst/2020/v13i01/148732,

Reliability-Based Heuristic-Based Algorithm for Task Execution in Multiprocessor Systems

Dr. V. Anandkumar*, Mr. T.R. Kalaiarasan and Mr. A.M. Ratheeshkumar

Sri Krishna College of Engineering and Technology, India

Article Type: Article

Article Citation: Anandkumar V, Kalaiarasan TR, Ratheeshkumar AM. Reliability-based heuristic-based algorithm for task execution in multiprocessor systems. *Indian Journal of Science and Technology*. 2020; 13 (01), 85-90. DOI: 10.17485/ijst/2020/v013i01/148732

Received date: November 7, 2019

Accepted date: December 13, 2019

***Author for correspondence:**
V. Anandkumar anandkumar@skcet.ac.in Sri Krishna College of Engineering and Technology, India

Abstract

Objective: This study implements a combination of heuristic-based algorithm and finding variable neighborhoods, thereby reducing make-span and improving reliability. **Methods/statistical analysis:** The bi-objective algorithm is proposed for a static planning strategy to achieve high performance in heterogeneous multiprocessor systems. The reliability of a system is based on the probability in which resources of the system execute tasks without any failure. **Findings:** Here, a genetic algorithm integrated using single neighborhood structure Genetic Variable Neighbourhood Search (GVNS) is implemented to improve the efficient search quality. **Applications and improvements:** Simulation is performed to maintain better performance parameters when compared with conventional algorithms.

Keywords: Directed Acyclic Graph, Inter Process Communication, Evolutionary Algorithm.

1. Introduction

Computational requirements today have increased the issues in parallel processing and traditional sequential processing [1–2]. In general, the original large scheduling application is decomposed to smaller tasks. These smaller tasks should obey the topological sorting where the precedence constrain is not violated among the tasks. Hence, the tasks are scheduled in the number of available processors, thus minimizing the duration of the scheduling length. This proposed work considers dual purpose for scheduling application is to enable efficient execution of applications on scattered heterogeneous systems. This minimizes the duration and reliability of available processors by taking failure rates and processor speed into account.

The destination system is based on directed acyclic graph (DAG), composed of vertices and edges with tasks as vertices and interdependency between tasks as edges. This paper assumes the static scheduling algorithm which considers the processing time, data dependencies, and cost for communicating with the tasks before starting the execution.

A high failure rate frequency is predicted depending on reliability, when a processor completes the execution of application task as soon as possible. There exists no single optimum solution, but a variety of number of solutions can be made optimal. The complexity of NP complete problem resulting in an optimal searching solution where the multiprocessor scheduling continues to be an ongoing issue even after intensive studies. Conventional scheduling mechanisms depend on existing heuristic approach to assign priority for each and every task in an ordered list of scheduled tasks. Each process is selected depending on the priorities assigned to them and the largest priority process is detached from the scheduled list assigned to a processor allowing earliest scheduling first time interval. When the processing capability of processes is fair, the homogeneous multiprocessor systems are considered over heterogeneous systems.

2. Scheduling Model

The early or static multiprocessor scheduling algorithm that uses all the inputs given at the time is subdivided into various tasks using priorities on multiprocessor systems and simultaneous task execution [3] of the computed program. Interprocess communication exists exactly between two process can be done through shared memory or message passing. This paper assumes the following characteristics of the multiprocessor systems with regard to task execution.

a) Diversified heterogeneous systems.
b) Task run until completion without interruption.
c) Complete network connection with all tasks completing their execution.
d) Avoid replication of tasks.
e) Performing simultaneous communication and computation of every process that has unique I/O unit.

The application model of the paper depends on related applications represented using DAG, denoted $G=\left(V,\ E,\ \varepsilon,\ \omega\right)$ by, where V is the set of vertices V as *tasks*; E, the interdependency relation between vertices;

$\varepsilon(v)$ is the percentage of work completed by the task in the whole system $v \in V$; and $\omega(u, v)$ is the cost related to the edge $(u, v) \in E$, denoting the time taken for the total data transfer completed from task u to v.

The intermediate task of v is given by $succ\ (v)=\{u \in V\,|\,(v,\ u) \in E\ \}$. The amount of cost incurred on a particular task in the whole system ni∈V on the processor pj is denoted as wi,j. Each edge ei,j ∈ E represents precedence constraints between task ni and nj, which means that the result of task ni has to be transmitted to task nj before task nj starts its execution. Each edge ei,j ∈ E is associated with a nonnegative weight ci,j representing the communication cost between the interdependent tasks ni and nj. Note that the real communication cost is equal to zero when the interdependent pair of tasks are assigned to the same processor. Assume that there is a task graph with T tasks to be assigned to a multiprocessor system with P processors [4].

Given a partial schedule solution, considering scheduling the highest-priority ready task ni on the processor pj, its earliest start time TEST(ni, pj) can be defined as

TEST(ni, pj) = max$\{T_{avail}(P_j), T_{ready}\,(n_i,p_j)\}$, where Tavail(pj) is the time when processor pj is available to the execution of the task ni. It can be defined as

$$T_{avail}(P_j) = \max_{n€\, exe(pj)} \{T_{AFT}(n_k)\}$$

where exe (pj) denotes the set consisting of all the tasks that have already been scheduled on the processor pj. TAFT (nk) represents the actual finish time when the task nk actually finishes its execution. Tready (ni,pj) denotes the time when all the data needed for the execution of task ni have been transmitted to the processor pj, which can be defined as

$$T_{ready}(n_i,P_j) = \max_{n€\, pred(ni)} \{T_{AFT}(n_k)\} + C_{k,i}$$

pred(ni) denotes the set consisting of all the immediate predecessors of the task ni.

The reliability of a system is based on the probability in which resources of the system execute tasks without any failure [5]. In this work, only processor failures are considered, which are statistically independent and follow the Poisson Law with a failure rate of FR(pj), $\forall$ pj $\in P^2$. This rate represents the number of processor failures per unit of time that can occur. It is assumed here that the communication links between processors are reliable, based on the assumption that communication protocols are able to tolerate their failures [6]. The task reliability cost associated with the execution of v in pj is then RC(v, pj) = FR(pj) $\times$ eft(v, pj), which should be minimized.

3. Evolutionary Algorithm

The optimization process begins with the initialization of the GA population based on a random procedure [7]. After the initialization, each scheduling algorithm represented by each person is subjected to a task priority calculation based on a heuristic method [8]. According to the hierarchy of tasks, the population is evaluated and classified with make-span being the aptitude of each individual. A set of genetic operators, including selection of binary tournaments, one-point random crossover, and reversal mutation, is then performed on the evaluated population. The population is then verified to select a subset of individuals for VNS initialization [9]. The local VNS search procedure is applied to each individual of the selected subset. Finally, the combined results of GA and VNS and alternative strategies are introduced to update the population. The evolution process is repeated until the criterion reaches a threshold value.

4. Encoding of Solutions

The population in GVNS consists of a group of individuals with each representing a potential solution to the given scheduling. Each solution is encrypted as a string of integers that denotes the index of the processor to the assigned task. Assume that the multiprocessor scheduling problem consists of a multiprocessor system with P processor available and a parallel program with T tasks. In addition, it is assumed that each task has a unique identifier ranging from 1 to T, while each processor has a unique index ranging from 1 to P.

4.1. Heuristic-Based Task Priority Calculation

In the bottom level b level(v) is the priority assigned to the tasks in V, which is the rank of tasks defined in [3]. Studies have shown that in the list scheduling framework, the use of b level(v) leads to efficient schedules on a heterogeneous processor system [6]. During the ordering phase, a list of tasks V_{ord} is constructed in accordance with the non-decreasing order of b level(v), defined as

$$blevel(v) = \{^{\overline{et(v)}} \max_{M \in SMCC(v)} \{\overline{et(v)} + \overline{ct(v,u)} + blevel(u)^{\substack{ifsucc(v)=\varphi, \\ otherwise,}}$$

where $\overline{et(v)}$ is the average earliest finish time of v considering all the processors and $ct(v,u)$ is the average communication time, considering all the links. During the scheduling phase, the algorithm seeks for the processor that optimizes the cost function. Let v be the next task to be scheduled with the highest *b level*(v).

4.2. Performance Parameters

This paper has incorporated the binary tournament selection without taking roulette wheel for selection. Substrings are exchanged between two individuals during crossover exchanges that provide new solutions to the already existing problems. In this paper, a crossover point is deputed in some random order that uses one-point operator, and the mutation rate is calculated based on the probability in which the integer in a string is changed which is directly proportional to allocating the unfinished task to another waiting processor. This mutation operator is a very essential issue that prevents GA from converging initially [10].

5. Results and Discussion

It is obvious that from the above Table 1, the analysis of our proposed system outperforms the conventional mechanisms with all sample cases with respect to mutation rate and crossover rate given in Figures 1 and 2. The proposed GVNS achieves a consistent performance for the various parameter sensitive analyses in terms of scheduling priority and achieving a marginal increase in CCR value outperforming the conventional mechanisms.

TABLE 1. Comparison of performance parameters

Parameter	GVNS	IGA	VNS	AIS
Sample size	500	500	–	500
No of generations	4000	4000	4000	4000
Operator selected	Binary tournament	Roulette wheel	–	Binary tournament
Crossover operator	Random one-point	–	–	–
Mutation rate	0.04	0.2	–	0.003
Sampling rate	0.2	–	0.1	–
Crossover rate	1.0		–	0.9

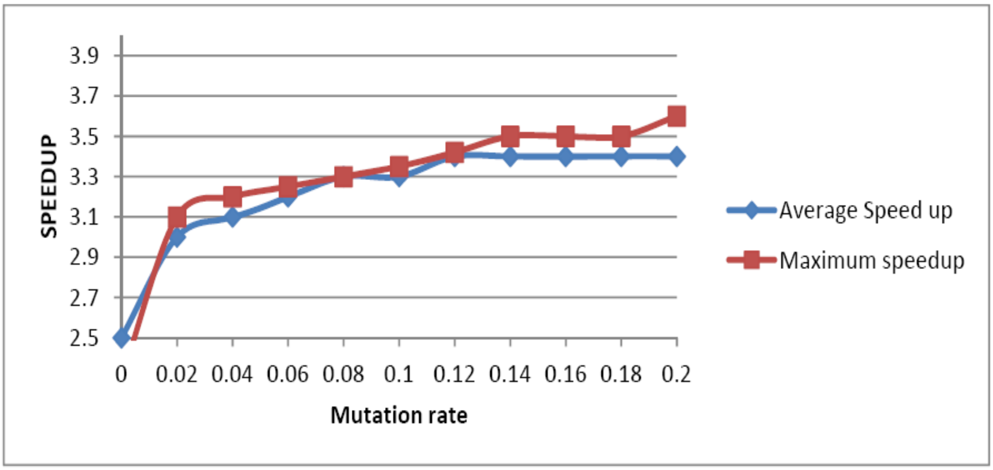

FIGURE 1. Variation of mutation rate from 0.0 to 0.03.

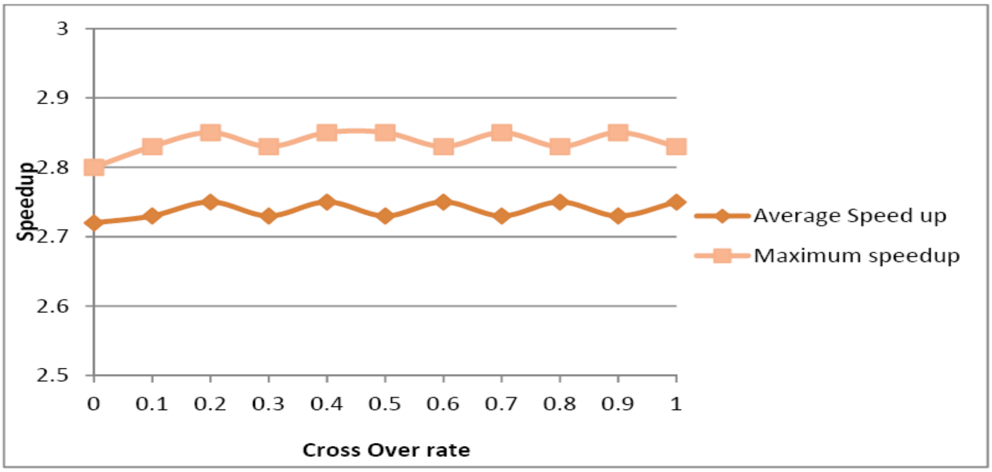

FIGURE 2. Variation of crossover rate from 0.0 to 1.0.

6. Conclusion and Future Study

This paper introduces the novel heuristic approach of processing the static scheduling algorithm approach in the heterogeneous multiprocessor system by using GA, using VNS and conventional scheduling techniques. This approach extends the conventional GA by using searching ability to improve the mutation and crossover rates over the proposed GVNS algorithm for computing the priority of the tasks for completing the system. The performance can be further improved by increasing the quality of the schedule with high CCR value to improve the efficiency of the scheduling algorithm.

References

1. Parallel computer architecture: a hardware/software approach. https://www.amazon.in/Parallel-Computer-Architecture-Hardware-Software-ebook/dp/B004JF5QRO. Date accessed: 29/09/1998.
2. Advanced computer architecture: parallelism, scalability, programmability. https://www.amazon.in/Advanced-Computer-Architecture-Parallelism-Programmability/dp/007053070X. Date accessed: 2003.
3. Anandkumar V, Kalaiarasan TR, Ratheesh Kumar AM. Impact of finding selfish nodes in MANET, *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*. 2019; 8(10), 1–12.
4. Wen Y, Xu H, Yang J. A heuristic-based hybrid genetic-variable neighborhood search algorithm for task scheduling in heterogeneous multiprocessor system. *Journal of Information Science*. 2011; 181(3), 567–581.
5. Topcuouglu H, Hariri S, Wu M. Performance-effective and low-complexity task scheduling for heterogeneous systems. *IEEE Transactions on Parallel and Distributed Systems*. 2014; 13(3), 260–274.
6. Boeres C, Milian I, Drummond LMA. An efficient weighted bi-objective scheduling algorithm for heterogeneous systems. *Journal of Parallel Computing*. 2011; 37(8), 349–364.
7. Kalaiarasan TR, Anandkumar V, Ratheesh Kumar AM. Sales forecasting using RNN. *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*. 2019; 8(9), 2748–2751.
8. Hansen P, Mladenovi c´ N. Variable neighborhood search. Search methodologies. 2018, 211–238.
9. Carrizosa E, Hansen P, Moreno-Perez JA. Variable neighborhood search. *Journal of Global Optimization*. 2015; 63(3), 427–629.
10. Ullman J. NP-complete scheduling problems. *Journal of Computer and System Sciences*. 2014; 10, 384–393.