Extended Optimization Procedures for Static List based Task Scheduling Algorithms for HeDCS

K. Vijaya Kumar, E. Laxmi Lydia, P. Amananatha Reddy

Abstract--- No matter how powerful a single system is efficient at processing, there are still reasons to Control the power of multiple computational units. The Distributed computational system performs scheduling tasks advantageously the processors to minimize the execution time in any application. Despite the problem in determining NP-Complete the execution time in Scheduling is minimized. This paper identifies, a specific different algorithm Sorted Nodes in Leveled DAG Division (SNLDD) based on Task-Scheduling. The fundamental principle of this algorithm is to partition the data as a Directed Acyclic Graph (DAG) two stages and categorize each task of every stage in decreasing order depending upon the estimated size. Outcomes of the proposed algorithm are processed using correlative analysis and productive outcome with respect to HEFT with CPOP is implemented among existing algorithms. With respect to the comparative analysis of the outcomes, the performance of the suggested algorithm with SPOP implementations improves execution in the aspect of speedup, effectiveness, complexity, and excellence. Further, a new algorithmic strategy SPOP and CPOP has been developed and executed in the proposed SNLDD in HEFT.

Keywords--- Task Scheduling, Sorted Nodes in Leveled DAG Division, Superior Performance Optimization Procedure, Heterogeneous Earliest Finish Time, Critical Path on Processor.

I. INTRODUCTION

Multiprocessor system in the Distributed Computing system is a system with more processors residing apart. Every Individual processor has its private, confidential memory, allowing security. Data Transmission is transferred through message passing between the processors. The processing elements interact with each and every processor in order to accomplish the common outcome. Distributed Computing systems support the following aspects:

- Providing efficiency in communication delay, balancing of load, distributed algorithms for huge messages.
- Providing flexibility in terms of modularity, scalability, portability, and interoperability.
- Computational Consistency that gives correct outputs irrespective of time at which the computation is finished for the identical input conditions.
- Ensuring robustness in certain circumstances of security violations and also in flexibility failures.

Figure 1: Distributed Computing Systems

Figure 2: Sharing of results with different Processors

Task Scheduling

Task scheduling deals with the process of organizing tasks among processors and minimizing their length in scheduling, i.e. the time is taken to execute every task corresponds to the starting time of the initial task. Queries generated on the outcome is the selection of tasks to processors. Real-time task scheduling originally deals with regulating the order in which the different tasks are to be reserved up for implementation by the operating system. Every operating system depends on more than one task, schedules to prepare the schedule of application of different tasks it process to execute. Each task scheduler is identified by the scheduling algorithm it operated. A massive number of designed algorithms for scheduling, real-time tasks have been refined. Real-time scheduling.
EXTENDED OPTIMIZATION PROCEDURES FOR STATIC LIST BASED TASK SCHEDULING ALGORITHMS FOR HEEDCS

II. LITERATURE SURVEY

Distributed computing systems are the number of an individual collectivity of processors interconnected through a tremendous system of connections that maintains the implementation of parallel operations [1]. The effectiveness of implementing various parallel operations on the systems desperately relies on the adopted approach to perform the tasks of the various parallel operations on existing processors [2]. Every System in DCs, the execution of parallel computations in inter-processor communications liable [3]. This burden exists whenever tasks are positioned to various processors in heart transfer of data. Accordingly, in Heterogeneous distributed Systems generating high capacity task performances in parallel implementation are more critical [4]. Such Conditions, an extensional Arrangement among achieved speedup by automatic parallelization, the communicational inter-mechanisms over systems, HeDCSSs having scheduling algorithms by considering the different processors executing at different times of the same task. Some unreliable Scheduling arrangement in HeDCSSs may restrict the effectiveness of the system when they perform in the very slowprocessors[5]. Mostly, Distributed Computing Systems in Scheduling algorithms have Static scheduling and Dynamic scheduling as two different existing task scheduling.

The primary Static scheduling algorithms, perform operations on the entire existing datafor scheduling, for the framework of parallel operations, processing operations with respect to execution time; the cost among the tasks of communication is acknowledged prior [5]. To evaluate specific information various approaches are used[4, 6]. In the parallel implementations, Static task scheduling concurrently has an activity of running processesat compile time [2,3,7].

Lately, a modernized algorithm known as the Longest Dynamic Critical Path was initiated [6]. Based on the algorithm, an improved attribute has identified the priorities accurately in HeDCSSs of each task. The outcome of the LDPC is measured for comparison of two algorithms HEFT[11] and the DLS [5] algorithms related to performance.

This paper provides, a modernized beneficial algorithm known as Sorted Nodes in Levelled DAG Division (SNLDD) to support HeDCSS systems with the defined total estimated number of processors for static implementation of task scheduling. The goal or achievement of using this algorithm is to obtain the large capacity scheduling of taskswithnecessary to producea tremendous performance in the Heterogeneous Distributing Computing Systems.

SLNDD is calculated with a parallel correlative study among SLNDD and the HEFT algorithm for execution. Based on these results, the SNLDD algorithmic approach excels the HEFT algorithm in conditions of schedule distance, time, performance, and characteristics of system behavior.

The SNLDD design approach has been changed by recommending a new process known as Superior Performance Optimization Procedure (SPOP) for the reduction of the seek time by applying the idle time concurrently referring tasks to the processors by producing high-quality scheduling of tasks, and to reduces churning length. In addition, both updated algorithms are correlated and altered SNLDD algorithm confirms in-excelling performance than transformed CPOP.

Problem Definition

Parallel operations are expressed by DAG for HeDCSSs, in static task scheduling.

Direct Acyclic Graph is determined by considering tuple (T,E), T defines the collection of ‘n’ tasks and E defines the collection of ‘e’ edges. Every task, t corresponds to an edge in a parallel function, and every edge(starts) is represented to order from the beginning of the constraint and a transfer information among tasks, and L. With the condition edge E(ti, tj), the implementation of ti T is implemented after tj T completes its implementation. The task that is started first related to its parent node of the sink task ta at the same time tj is represented as the parent in the task. Any individual task performing operations with no parents are namedancy task, and any individual task performing operations with no children are named exit task. Most quantities of data that is transmitted from one task to another task (i.e., ti to tj) are related to every edge (ti, tj) [5, 11].

Heterogeneous Distributed Computing Systems is determined by a group of m processors with P that has distinct result performances. Computation cost matrix (W) (i.e., n*m) keeps the implementation costs of all the ‘n’ tasks in m processors. Every component wi,j W produces the time of task processor with predicted performance.
Now, every individual processor for the performance is simulated altogether linked. Information exchange among processors can take place through individual links; by this simultaneous execution of evaluation tasks and interactions among processors are approved [3, 12, 13]. Computational costs of the tasks are acknowledged without changing. Especially, when task (t_i) computation costs at the respective processor (p_j) and processor (p_k). The communication cost between any two processors (p_j) and (p_k) rely on the network initialization at processors p_j and p_k further to the transfer time on the network. Processors (p_j) and (p_k) communication cost rely on processors (p_j) and (p_k) at network initialization and the time needed to communicate. Sender initially starts at some point in time and time at the receiver side is ignored when compared to the residing network time [14]. The data transfer rate inside the network among the two processors predicted as constant and permanent [3].

Implementing SNLDD algorithm grows with higher efficiency compared to the LDCP algorithm. Following are some demonstrations for the algorithm:

- It aims to restore the directed edges, operating time, tasks and communications at every appointment step that is not required to introduce SNLDD algorithm, overheads at execution time are removed in SNLDD algorithm.
- Processors based on their computational size comforting not only adequate scheduling of tasks further accepts to develop the entire system of allocation of tasks basing on many characteristics like its cost for communications, reliance, and execution time at authorized tasks.
- Partitioning every level tasks are based on computing size by managing scheduled tasks with substantial computation size and then tries to reduce the need among tasks.
- Processors performing sleek time are reduced as long as DAG subdivision of levels and allocating to the processors.
- The introduced algorithm is designed to have more effective when compared to other algorithms like LDCP, HEFT, and LCD.
- Several different designs of extremely high currently used algorithms like sorting list [6], tree algorithms [9], and clustering algorithms [2] are confirmed. Therefore, it is confirmed that SNLDD algorithm is a set of selected algorithms.

To overcome the repeated computations on each step of LDCP, SNLDD algorithm, the computation size of every task is executed only once in DAG, and restoring them by estimating on every step.

This takes time in the evaluation of SNLDD algorithms for better calculations. Θ(m × n²) is the time complexity, where m describes a number of processors used in the scheduling algorithm, and n describes the number of tasks considered in the scheduling algorithm.

The following figures 4 and 5 are the sample assumed DAG with computational costs.
Prioritization: HEFT

SPOP is known for a modernized optimization procedure with high-performance, extending itself to specify algorithms. It concentrates on the reliability of the tasks assigned to the processors by reducing its time considering scheduling policy[6]

- The scheduling policy assigned to the tasks by the processors at insertion basedsassumes every feasible idle time on the processor isachieveddata time slot that is similarandhigherlength to the computation/ implementation time. Precedence constraints among tasks are very required in performing.
- The idle time is computed by exploring the ready state time by defining the length and inserted back to the final task that is scheduled on the processors.

**HEFT (Heterogeneous Earliest Finish Time)**

HEFT is one of the straightforward and best approaches in the static task scheduling both environments like heterogeneous and homogeneous even when few processors are considered. HEFT deals with crossing rather evaluating two phases: one is Prioritization phase and another is processor selection stage.

In the first stage of Prioritization: HEFT estimates the importance of the tasks by using the upward ranking(rank_u). An application operation is moving over in upward path and obtains entire list nodes with its rank with the service provided by the mean communication and performance cost. An obtained list is organized in systematized descending order of the rank_u. Every task with upward rank is defined as:

\[
\text{Rank}(n_i) = W_i + \max_{j \in \text{succ}(n_i)} (C_j + \text{rank}(n_j))
\]

\(W_i\) describes the mean computation cost, \(\text{succ}(n_i)\) is described as the immediate child of node \(n_i\), \(C_j\) is described as the mean communication cost of node \((i,j)\). If any two nodes have when two nodes obtain the similar rank value, it will prefer indiscriminately. The attained graph is moved from the end node to start node. The maximum rank value is similar to the end node:

\[
\text{rank}_{\text{exit}}(n_j) = W_{\text{exit}}
\]

**Critical Path On Processor (CPOP)**

CPOP manipulates two methods, namely upward ranking and downward ranking. CPOP calculates the rank by the value of every node by combining the ranking approaches rank + ranked assign them to a queue. CPOP also has two levels of calculation: task prioritization level and task allocation level.

The first phase of task prioritization, every task is prioritized based on the rank value (rank_u + rank_d) by use of transmission and computational costs that id obtained from DAG. Later, they are placed in a queue (in low order). CPOP performs Critical path to obtain the longest path from the start node to last node.

The second phase of task allocation, all the tasks are picked concentrating on maximum rank and chosen the minimum execution time of the task giving appropriate processor.

**IV. EXPERIMENTAL RESULTS AND ANALYSIS**

Result analysis is based on the attributes, namely: schedule length of the tasks, speedup of the processors and efficiency of the entire algorithm.

Comparison metrics of the algorithms SNLDD after modification through the performance of Superior performance optimization Procedure and with the Heterogeneous Earliest Finish Time with respect to Critical Path On Processor.

Schedule length finds overall execution time of a considered Direct Acyclic Graph (DAG), which is also known as makespan. Speedup defines the ratio of processed schedule length with the total number of processors. Efficiency gives the estimated by dividing speed with the total processors each time when they run.

Figures 7 and 6 are the schedule length with 10 tasks and the speedup with processors.

Figure 6 describes the speed of the processors in the distributed computing systems when 20, 40, 60, 80, 100 are considered. It shows that when the number of processors is where the speed is high. The figure thus concludes that the algorithm SNLDD with optimized procedure has given maximum result than the algorithm HEFT.

Figure 7 describes the schedule length of the considered DAG gives overall execution time. Here the comparison is done among the algorithms SNLDD without Optimization Procedure, i.e., only SNLDD, with SPOP, HEFT, and CPOP. Among all the algorithms the execution time of the modified SNLDD gives the fastest evaluation time.
Scheduling Algorithms, mostly List based static algorithms are considered for HDCS. Based on the algorithms, SNLDD, HEFT, CPOP, and implementation of SNLDD with Superior Performance Optimization Procedure are studied. In this paper, the outcome performance of the developed SNLDD algorithm is correlated with current existing algorithms mainly for HeDCSs. The comparative study between the proposed SNLDD algorithm with modified optimization procedure SPOP and HEFT with CPOP are evaluated based on the schedule length, speedup, efficiency of running programs and quality parameters with respect to memory in parallel computer systems has achieved a high speed up and fast execution time by SNLDD.

V. CONCLUSION

REFERENCES

10. Kwok Yu-Kwong. High-performance algorithms for compile-time scheduling of parallel processors. The Hong Kong University of Science and Technology in Partial Fulfillment of the requirements for the degree of Doctor of philosophy in computer science Hong Kong; May 1997.
