CHAPTER 7

CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

This concluding chapter recounts the research contributions with a brief discussion on the merits of the developed task scheduling algorithms. It also reveals a few open problems in the focused area of research.

7.1 Conclusions

Efficient task scheduling is vital to maximize the benefits of executing an application represented by a DAG in DHCS. The DAG structure frequently occurs in many regular and irregular applications such as Cholesky factorization, GE, FFT, MDC, Diamond graphs, etc. However, finding an optimal schedule without violating precedence constraints among the tasks is known to be a NP-complete problem. Consequently, various categories of heuristic algorithms such as list-scheduling, clustering and task duplication-based algorithms have been developed in the literature. The list-scheduling algorithm provides better schedule with minimum overhead while clustering and task duplication-based algorithms reduce the communication cost thereby minimizing the schedule length. Further, genetic approach is also applied to task scheduling in order to generate better schedules than the heuristic algorithms.

In this research, heuristic algorithms in list-based, task duplication-based and genetic-based scheduling categories have been developed for DHCS. The developed algorithms were evaluated by conducting simulation experiments using a large set of randomly generated task graphs. The experiments were also conducted for task graphs of some real world applications. The features of the developed algorithms are briefly explained in the subsequent sections.

(1) Two novel list-scheduling algorithms namely, HPS and PETS have been developed for completely connected DHCS. In HPS algorithm task priority is assigned based on the size of data received from the predecessor tasks, the size of
data sent to the successor tasks and the average computation cost. In PETS algorithm task priority is assigned based on the priority of the predecessor task and the size of data sent to the successor task. Both the HPS and PETS algorithms schedule the tasks based on their priority. Insertion-based scheduling policy is used in both the algorithms for computing earliest finish time of the task on each processor. The task is scheduled on a suitable processor which gives minimum completion time.

The experimental results show that both HPS and PETS algorithms outperform the existing HEFT, HCPT, CPOP and DLS algorithms in terms of average SLR, efficiency, speedup and the frequency of best results. The HPS algorithm performs better than the HEFT by 10 percent, HCPT by 17 percent, CPOP by 23 percent and DLS by 34 percent in terms of average SLR. The average SLR value of the PETS algorithm on all generated graphs is better than the HEFT by 8 percent, HCPT by 15 percent, CPOP by 20 percent and DLS by 33 percent. The time complexity of HPS and PETS algorithms are the same and is given as $O(v^2p)$, which is reasonable for practical applications.

(ii) A task duplication-based scheduling algorithm namely. HPDCS has been developed for completely connected DHCS. The HPDCS algorithm uses number of successor tasks and average computation cost of a task to assign the task priority. HPDCS employs insertion-based scheduling in addition to task duplication for computing the earliest finish time of a task and selects the best suitable processor for task execution. The HPDCS algorithm is compared with the existing algorithms such as DCPD, HCNF and LDBS1 algorithms. The experimental results show that HPDCS algorithm surpasses DCPD by 10 percent, HCNF by 19 percent and LDBS1 by 27 percent in terms of average SLR. The time complexity of HPDCS algorithm is $O(v^2p)$ which is very much lesser than the time complexity $O(v^2ep^4)$ of LDBS1 algorithm, and comparable with that of HCNF and DCPD algorithms.

Another task duplication-based scheduling algorithm namely, HCDBTS has been developed for arbitrarily connected DHCS. In HCDBTS algorithm, a task
which communicates highly with its successors is given higher priority. While scheduling the tasks onto the processors, HCDBTS algorithm computes the shortest communication path between the processors. The algorithm uses task duplication mechanism to minimize the earliest finish time of the task. The effectiveness of the algorithm is verified by comparing with MH and MMH algorithms using various evaluation parameters. The comparison shows that HCDBTS algorithm is better than the MH and MMH algorithms. On an average the HCDBTS algorithm is better than the MH by 16 percent and the MMH by 10 percent in terms of average SLR value. The time complexity of HCDBTS algorithm is $O(v^2p^2)$ which is lesser than the time complexity of MH and MMH algorithms.

(iii) Two genetic algorithms namely, GATS and DGATS have been developed for completely connected DHCS. The GATS algorithm combines genetic and list-scheduling approaches to generate schedules. It generates the initial population using the predecessor and successor communication costs of the task. The tasks are selected based on the priority in the chromosome and scheduled to the suitable processor using levelized earliest finish time decoding heuristic. The GATS algorithm generates new chromosomes using crossover and mutation operators for a predetermined number of times. The number of generations is determined using depth and breadth of the task graph. The performance of the GATS algorithm is compared with the existing algorithms such as HEFT, CPOP, PSGA and HPDCS based on various evaluation parameters. The experimental results show that the GATS algorithm provides better performance than HEFT, CPOP and PSGA algorithms, whereas HPDCS is better than GATS algorithm because it uses the task duplication-based mechanism.

The DGATS algorithm is an extension to GATS algorithm which uses the task duplication-based levelized earliest finish time decoding heuristic in order to reduce the finish time of the task. The DGATS algorithm is compared with HPDCS, GATS, PSGA, HEFT and CPOP algorithms. The experimental results show that DGATS algorithm is better than HPDCS by 8 percent, GATS by 13.
percent, PSGA by 17 percent, HEFT by 24 percent and CPOP by 31 percent in
terms of average SLR.

In this research work, the problem of scheduling DAGs in MCS has also
been explored in addition to DHCS environment. Though the MCS has constraints
such as power/energy, availability and mobility, it has the potential to execute larger
applications due to the advancement in computing and communication technologies.
However, execution of larger program in MCS requires both the compile-time and
run-time support. A new task scheduling algorithm namely, HPSM has been
developed to minimize either the schedule length or the total energy consumed, or
both at the compile-time. At the run-time, HPSM algorithm reschedules the tasks
assigned to a node during the compile-time to other node in order to successfully
complete the execution, whenever a node becomes unavailable due to mobility or
energy exhaustion.

The HPSM algorithm uses a novel concept of weight values to represent the
relative importance between energy efficiency and schedule length minimization.
However this concept has not been explored in the MCS. Therefore, the existing
HEFT and PETS algorithms have been modified to suit MCS in order to evaluate the
performance of HPSM algorithm. The modified algorithms are named as HEFTM
and PETSM. Experimental results show that HPSM algorithm surpasses these
algorithms in terms of schedule length and energy consumption. The HPSM
algorithm is better than PETSM algorithm by 4 percent and HEFTM algorithm by 8
percent with respect to the energy consumption.

7.2 Future Research Directions

In this research, an attempt is made to provide solutions to the task
scheduling problem by developing and implementing various categories of task
scheduling algorithms for DHCS. The algorithms developed in this research can be
further extended in the following aspects:
(i) The developed algorithms other than the HCDBTS algorithm assume the much idealized model of the target system where, the processors are fully connected and all communications are performed concurrently without contention. Hence, the developed algorithms can be extended to arbitrarily connected processors by considering the communication contention while scheduling the tasks.

(ii) The algorithm developed for the mobile computing system considers the energy consumption of the computing and communicating components only. The algorithm can be extended to include the energy consumption of other components in the MCS by which energy minimization can still be achieved.

(iii) The algorithms developed in this research can be extended to dynamic scheduling environment to overcome the limitations of static scheduling.

(iv) As the number of processors in the DHCS increases, the likelihood of processor and link failures also increases. Hence, scheduling with fault tolerance becomes an important issue. The developed algorithms may be extended to include fault tolerance feature.

(v) The developed algorithms can be extended to provide an integrated solution for the task scheduling problem by considering various QoS factors such as reliability, energy saving etc., in the future.