CHAPTER 2

RELATED WORK

Most traditional disk scheduling algorithms, such as FCFS, SCAN, C-SCAN, LOOK, C-LOOK, and SSTF are designed to reduce disk-seek time and increase its throughput (Denning 1967a, Geist and Daniel 1987 a, Chen et al 1991 a). FCFS algorithm performs operations in order the task arrives. However, the performance of this algorithm is poor (Seltzer et al 1990; Hofri 1980). SSTF reduces the total seek time compared to FCFS. The disadvantage of SSTF is starvation that is the R/W head stays in one area of the disk if very busy. These algorithms do not consider real-time constraints of I/O tasks and, therefore, are not suitable to be applied directly on a real-time system (Hofri 1980 a, Chen et al 1992, Worthington et al 1994).

In the SCAN algorithm, the disk arm starts at one end of the disk, and moves toward the spindle, servicing requests as it reaches each cylinder, until it gets to the spindle. From this end, the direction of head movement is reversed and servicing continues. The head continuously scans back and forth across the disk (Chen et al 1992 a, Coffman et al 1972). C-SCAN scheduling is a variant of SCAN designed to provide uniform wait time. Like SCAN, C-SCAN moves the head from one end of the disk to the spindle, servicing the requests along the way. When the head reaches the spindle, it immediately returns to the beginning of the disk, without servicing any requests on return trip (Coffman 1973; Coffman and Hofri 1982). LOOK algorithm is also a variant of SCAN. In LOOK the disk head does not move inward or outward when there is no request in that direction. LOOK performs better than SCAN when load is low but it is equivalent to SCAN when the load is high (Sohn and Kim 1997). A variant of LOOK scheduling is C-LOOK. C-LOOK moves
the head in one direction from its current position, after serving all the requests in current direction, disk head starts to serve the first request in other end without serving the requests in return trip. It provides more uniform wait time for the requests (Sohn and Kim 1997 a).

Earliest Deadline First Scheduling (EDF) serves the request which has the earliest deadline without considering the seek distance. In EDF the disk access time is high because it does not consider the disk arm position and perform no optimization in the seek time (Worthington et al 1994 a).

Group-Sweeping Scheduling (GSS) (Yu et al 1992, Yu et al 1993) is a disk scheduling strategy in which requests are served in cycles with a round-robin manner.

To improve both disk seek-time and miss ratio, SCAN-EDF (Reddy and Wyllie 1994, Reddy et al 2005) algorithm groups the disk requests that have the same deadlines, and then uses SCAN algorithm in each group. The advantage of SCAN-EDF is that it attempts to provide both seek optimization and earliest deadline first service. However, the effectiveness of the SCAN-EDF algorithm depends on how many requests carry the same deadline. The behavior of this algorithm really depends on how deadlines are assigned to I/O requests.

Chang et al (2007) suggests Global Seek-optimizing Real-time (GSR) disk scheduling algorithm that groups the EDF input tasks based on their scan direction. Output schedules of the algorithm are always feasible if the input real-time disk requests are feasible sequences. But with an infeasible input, it is very unlikely to have a feasible output.
Earliest-Deadline-SCAN (D-SCAN) (Abbott and Molina 1990) is a modification of the traditional SCAN algorithm to consider request deadlines. In D-SCAN, the track location of the request with earliest deadline is used to determine the scan direction. In Feasible-Deadline-SCAN (FD-SCAN) algorithm (Abbott and Molina 1990 a), the track of the request with the feasible deadline is used to determine the scan direction. A deadline is feasible if it is estimated that it does not miss. The disk arm moves toward the direction and serves all requests along the way.

Muqaddas et al (2009) introduced a new algorithm, SLOOK. It extends the conventional LOOK algorithm in deciding which way to move the head first in the case the disk head is located in between the two far-end requests.

Zarandioon and Thomasian (2006) proposed a methodology of reducing the computational cost of the SPTF algorithm. An efficient implementation of LOOK-ahead algorithm was also proposed.

Iyer and Druschel (2001) implemented Anticipatory Scheduler (AS) in which they introduced short, controlled delay period during which disk waits for additional requests to arrive from the process that issued the last request. Corbet (2002) states that the basic idea behind the implementation is to aggressively reorder requests to improve I/O performance while simultaneously ensuring that no I/O requests is being starved. Seelam et al (2005) proposed cooperative anticipatory scheduler (CAS), which eliminated the starvation problem of the anticipatory scheduling. The study showed that the CAS has an order of magnitude improvement in performance in cases where the AS leads to process starvation. Kim et al (2006) says that the deadline I/O scheduler incorporates a per-request expiration-based approach and operates on five I/O queues.
The general real-time disk scheduling with linear seek-cost function is NP-Complete. GA provides a good way to find solution to the NP-complete problems. Hence GA can be employed in solving disk scheduling problem (Hofri 1980). The general real-time disk scheduling with linear seek-cost function is NP-complete (Huang et al 2005, Lu and Yuan 2007). GA proposes a way to find a good but not the best solution of NP-complete problems. Therefore, GA may be employed in solving disk scheduling problem. However, there are some genetic disk scheduling methods in literature; for example, Turton and Arsalan (1995) presented an algorithm that uses the concept of parallel genetic algorithms by employing order-based crossovers. Moilanen and Williams (2005) have attempted to use the evolutionary nature of GA and found the optimized scheduling in a diverse and dynamic environment. In another algorithm presented by Özdem and Karaboga (2006) Ant Colony Optimization (ACO) approach has been used to schedule disk requests. A new disk scheduling method based on GA that considers makespan and number of missed tasks simultaneously was proposed by Bonyadi et al (2010). In the proposed algorithm, a new coding scheme is presented which employs simple GA procedures such as crossover and mutation and a penalty function in fitness.

The concept of (m, k)-firm deadlines was introduced by Hamdaoui and Ramanathan (1995) to model tasks that have to meet m deadlines every k consecutive invocations in the context of a stream of packets of communication networks. Later the same authors developed an analytic model for evaluating the expected probability of dynamic failure for an incoming stream given the other streams present in the system. The (m, k)-firm deadlines approach is a best-effort on-line scheduling algorithm, the priority of a task is raised if it is close to not meeting m deadlines on the last k invocations. Similar approaches have also been proposed Hamdaoui and Ramanathan (1997) and Wedde et al (1999). They are characterized by
providing best-effort scheduling algorithms. However, dynamic failures are possible. Quan et al (2000) presented a guaranteed approach of (m, k)-firm systems that is based on an off-line scheduler of periodic tasks. Its main limitation is that it builds an off-line schedule and therefore it is only applicable to task sets for purely periodic tasks.

Bernat and Cayssials (2001) introduced the notion of weakly-hard real time constraints as a generalization of the concept of (m, k)-firm to cover other types of missed invocation patterns. An initial approach for the joint scheduling of a periodic work with weakly hard schedulers, called enhanced dual priority scheduling, was presented by Bernat and Burns (1997). It is based on the dual priority mechanism (Davis and Wellings 1995) that promoted task instances following a predefined pattern thus making slacker for aperiodic tasks. However, the approach was very rigid as the pattern was fixed beforehand and could not be altered if the tasks did not run for their worst case. The previous approaches have two limitations: One is, some of them only provide best effort guarantees and thus dynamic failures can occur. On the other hand, the approaches are very restrictive requiring process models with Di = Ti or that all m are the same for all tasks.

Bernat and Cayssials (2001 a) al presented an on-line scheduling framework called Bi-Modal Scheduler (BMS) for weakly-hard real-time systems. It is based on a simple and robust mechanism that has two modes of operation. If a task is not able to meet its weakly-hard constraint, then the scheduler enters into panic mode. When there is no such a risk, a generic scheduler can be used.

DWCS is an algorithm which attempts to guarantee no more than x out of a window of y deadlines. If there are more than x deadlines missed for consecutive packets in real-time and multimedia streams, there is a violation
of window-constraint (West and Schwan 1999 a, West et al 2001). West et al (1999) designed DWCS to meet the service constraints on packets from multiple, network-bound media streams. DWCS has the ability to limit the number of late packets over finite numbers of consecutive packets, in loss-tolerant or delay-constrained, heterogeneous traffic streams. However, unlike fair-scheduling algorithms, DWCS can be unfair when necessary.

West et al (2001 a) implemented DWCS as a kernel-loadable module that replaces the default Linux scheduler. It was shown that DWCS is capable of successfully scheduling CPU- and I/O-bound processes in Linux more than 99% of the time, when a feasible schedule is theoretically possible. However, the scheduler cannot be invoked if there are nested kernel control paths.

West et al (2001 b) scheduled CPU and I/O-bound processes in Linux, when a feasible schedule is possible by DWCS. DWCS for real-time packet steam was analyzed and it was shown that DWCS can guarantee real-time service to packets from multiple streams (West and Poellobaeryl 2000). However, it was shown that the DWCS algorithm can fail for arbitrarily low aggregate utilization rates of the packet streams (Mok and Wang 2001). It was also shown that window-constrained scheduling was NP-hard problem.

Using DWCS, (West et al 2004) the delay of service to real-time streams is bounded, even when the scheduler is overloaded. Moreover, DWCS is capable of ensuring independent delay bounds on streams. At the same time, it guarantees minimum bandwidth utilizations over tunable and finite windows of time. The algorithm is capable of not only satisfying window-constraints, but can operate as an EDF, static priority or weighted fair scheduler.
Later, Virtual Deadline Scheduling (VDS) algorithm has been developed. It is suitable for applications that can tolerate some degree of delay (Zhang et al 2004 a). A relaxed version of the window-constrained problem was proposed with VDS that provides window-constrained service guarantees to jobs with potentially different request periods. VDS outperformed DWCS and similar algorithms, when servicing jobs with potentially different request periods. Additionally, VDS is able to limit the extent to which a fraction of all job instances are serviced late. VDS can provide better window-constrained service guarantees than other related algorithms.

Guaranteed dynamic priority assignment schemes, GDPA and GDPA-S, for multiple real-time tasks subject to (m, k) firm deadlines were presented by Cho et al (2010). It was analytically confirmed that the two proposed algorithms provide a guarantee of zero probability of dynamic failures when the system is under-loaded.

Based on the energy costs and the contribution to the system performance, a set of jobs were selected and dispatched to CPU. By this, the performance was significantly improved (Tarek and Aydin 2005).

A methodology was proposed for control and weakly-hard real-time scheduling co-design (Ben Gaid et al 2008). The work focused on achieving efficient resource utilization, through control and scheduling co-design according to average resources considerations, while guaranteeing worst-case performance requirements. An abstract model of control tasks execution is introduced, allowing the establishment of a formal relationship linking control performance to deadline misses.
The article (Budin et al 2000) describes the use of GA in real-time systems that employ imprecise computation paradigm. In real-time systems, faults may occur in the computation or the environment that can cause missed deadlines. Hence, the idea of using partial results has been introduced.

The schedule itself is static and fully determined prior to system execution. Further development of the algorithm would include dynamical scheduling of the tasks, which would allow certain parameters, as deadline times or task weights, to change during the process.

Al-Saber et al (2008) proposed the use of concatenation of two or more packets for internal switching to increase the utilization of link bandwidth. This approach effectively increases the bandwidth utilization and also increases the throughput performance of a switch. It was observed that packet-based scheduling outperforms cell-based scheduling for an average packet length larger than a segment size, where a segment is the number of bytes processed internally in a time slot. However, for small packet lengths, with a size close to the segment size, cell-based and packet-based scheduling modes have equivalent performance.

In the literature, several methods for packet loss differentiation have been proposed, based on either an analysis of end-to-end delay characteristics or bit error detection requiring changes to the link or transport layer protocols (Garcia and Brunstrom 2002).

Korhonen and Wang (2005) studied the effect of packet size on loss rate and delay characteristics in a wireless real-time application. An analytical model was derived for the dependency between packet length and delay characteristics. It was shown that the careful design of packetization schemes
in the application layer may significantly improve resource utilization in delay sensitive media streaming under difficult wireless network conditions.

Guo et al (2005) proposed an efficient packet scheduling algorithm Packetized Dynamic Batch CoScheduling (P-DBCS) for a heterogeneous network processor system. P-DBCS is capable of scheduling variable length packets among a group of heterogeneous processors to ensure both load balancing and minimal out-of-order packet delivery. Simulation results verified that fair scheduling among multiple flows is successfully achieved.

Most real-world engineering problems involve simultaneously optimizing multi-objectives where considerations of trade-offs is important. In the last decade, evolutionary approaches have been the primary tools to solve real-world multi-objective problems. During the past decade, a variety of multi-objective Evolutionary Algorithm (MOEA) techniques have been proposed and applied to many scientific and engineering applications.

Konak et al (2006) presented an overview and tutorial describing GA. GA was developed specifically for problems with multiple objectives. It used specialized fitness functions and introducing methods to promote solution diversity.

Van Veldhuizen and Lamont (2000) discussed multi-objective optimization problems and certain related concepts, presented an MOEA classification scheme, and evaluated the variety of contemporary MOEAs.

In this work, five MOEAs were recommended.

- One recommendation is MOGA, implemented by Fonseca and Fleming (1998). In this work, fitness sharing occurs only between solutions evaluating to vectors with identical Pareto rank. The niching distance is measured in phenotypic space; the
distance (over some norm) between two solutions’ evaluated fitness vectors is computed and compared to share (the key sharing parameter). If the distance is less than share, the solution’s associated niche count is then adjusted.

- Van Veldhuizen (1999) implemented Multi-objective messy GA (MOMGA). It was used to explore the relationship between Multi-objective optimization problem (MOP) solution building blocks and their use in MOEA search. It incorporated fitness sharing and Horn et al’s (1994) tournament selection.

- Horn et al (1994 a) defined niching differently in their Niched Pareto Genetic Algorithm (NPGA), which performs selection via binary Pareto domination tournaments. Nondominated Sorting Genetic Algorithm (NSGA) that was implemented by Srinivas and Deb (1994) used to explore bias prevention towards certain regions of the Pareto front. It measured distance (over some norm) genotypic space; the distance between two solutions is compared to share. It shared fitness only between solutions evaluating to vectors with identical Pareto rank.

- Strength Pareto Evolutionary Algorithm (SPEA) was used by Zitzler and Thiele (1999 a). SPEA stores the current solutions in a secondary population, immediately culling solutions whose evaluated vectors are dominated. It employed the Pareto ranking scheme and incorporated fitness sharing.

Zitzler and Thiele (1999 a), MOEA from Tan et al (1999), Incrementing Multi-Objective Evolutionary Algorithm (IMOEA) from Tan et al (2000) and Exploratory Multi-Objective Evolutionary Algorithm (EMOEA) from Tan et al (2001 a) have been surveyed. The simulation results showed that none of the methods is the most superior considering all aspects of the performance measures.

Fonseca and Fleming (1993 b) described a rank-based fitness assignment method for MOGA. Conventional niche formation methods were extended to the class of multimodal problems and theory for setting the niche size was presented. The work was done for a Pegasus gas Turbine engine.

Yoo and Gen (2001) proposed a new tasks scheduling algorithm, Proportion-based Genetic Algorithm (pp-GA) that used GA. It was designed for continuous media in soft real-time systems. pp-GA was to determine execution sequence of tasks with the objective of minimizing the variance of deadline missing and the total number of context switching among tasks. It was shown that the results produced by pp-GA were better than that of EDF and mPS scheduling algorithms. pp-GA produced all Pareto optimal solutions.

Lee and Chen (2003) proposed a Partitioned Genetic Algorithm (PGA) to schedule parallel program on multiprocessor system. PGA integrates the concept of Divide-and-Conquer mechanism to partition the entire problem into subgroups and solve them individually. PGA performs similar to GA and spends less time in scheduling.

Gen and Lin (2005) solved the Bicriteria Network Design Problem (BNDP) with the two conflicting objectives of minimizing cost and maximizing flow with a new Multi-objective HGA (MO-HGA) approach. It
was shown that the performance of MOGA can be improved by hybridization with Fuzzy Logic Control (FLC) and Local Search (LS). It was concluded that the convergence speed to the Pareto front was improved by hybridization.

Zdansky and Pozivil (2002) have proposed an algorithm of the combination of GA and Tabu search to optimize schedules for a hybrid flowshop by minimizing the makespan. It was tested on sets of randomly (within defined parameters) generated input data matrices. The performance of the algorithm was better than that of both Tabu search and GA.

Yoo and Gen (2004) proposed a new scheduling algorithm for multiprocessor soft real-time system using period-based Genetic Algorithm (pd-GA). The work focused on scheduling for continuous tasks that are periodic and nonpreemptive, with the objective of minimizing the sum of all tasks’ deadline missing time.

Rahoual and Saad (2006) designed algorithm that generates timetables for any type of academic institution by hybridizing genetic algorithm and tabu search. The algorithm was able to process more than 11 constraints from among the most common ones, with the possible extension to others. The algorithm provides good convergence rate and the rate of success.

Walid and Alaa Sheta (2008) proposed the idea of using SA based local search technique to provide an efficient power load distribution for distributed generation network. The goal was to optimize the exact amount of electrical power to be injected in a distributed generation system. The results show that the SA was capable of optimizing the electrical power and power loss.
Teresa et al (2006) illustrate the application of an evolutionary algorithm to the design of diesel engine in order to reduce emissions and improve fuel consumption. It was concluded that design of the optimized combustion chamber for a small bore direct injection diesel could be carried out in a completely automated fashion.

The first to report an implementation of a micro-GA was Krishnakumar (1989). Micro-GA was compared against a simple GA. Micro-GA showed faster and better results on two stationary functions and a real-world engineering control problem (a wind-shear controller task). After him, several other researchers have developed applications of micro-GAs (Charles 1991, Dozier et al 1994, Johnson and Abushagar 1995, Xiao and Yabe 1998). Knowles and Corne (2000) used multi-membered versions of Pareto Archived Evolution Strategy (PAES) which may be seen as a form of micro-GA. But, it was concluded that the addition of a population did not improve the performance of PAES and increased the computational overhead.

Jaszkiewicz (1998) proposed an approach in which a small population initialized from a large external memory and utilized it for a short period of time. However, to the best of our knowledge, this approach has been used only for multi-objective combinatorial optimization.

A multi-objective optimization approach based on a micro-GA was proposed with a population size of four and a reinitialization process (Coello and Pulido 2001). Three forms of elitism and a memory to generate the initial population of the micro-GA were used. The results obtained showed that this approach produced an important portion of the Pareto front at a very low computational cost. The convergence was also very fast.