# Chapter 6
# ACO Based Dynamic Scheduling Algorithm for Real-Time Multiprocessor Systems

**Apurva Shah**
*G H Patel College of Engg & Tech, India*

**Ketan Kotecha**
*Nirma University, India*

## ABSTRACT

*The Ant Colony Optimization (ACO) algorithms are computational models inspired by the collective foraging behavior of ants. The ACO algorithms provide inherent parallelism, which is very useful in multiprocessor environments. They provide balance between exploration and exploitation along with robustness and simplicity of individual agent. In this paper, ACO based dynamic scheduling algorithm for homogeneous multiprocessor real-time systems is proposed. The results obtained during simulation are measured in terms of Success Ratio (SR) and Effective CPU Utilization (ECU) and compared with the results of Earliest Deadline First (EDF) algorithm in the same environment. It has been observed that the proposed algorithm is very efficient in underloaded conditions and it performs very well during overloaded conditions also. Moreover, the proposed algorithm can schedule some typical instances successfully which are not possible to schedule using EDF algorithm.*

## INTRODUCTION

Real-time systems are defined as those systems in which the correctness of the system depends not only on the logical results of computations but also on the time at which the results are produced (Ramamritham & Stankovik, 1994). The objective of real-time system is to guarantee the deadline of tasks in the system as much as possible when we consider soft real-time system and for achieving this goal vast research on real-time task scheduling has been conducted.

Real-time scheduling techniques can be broadly divided into two categories - Static and

DOI: 10.4018/978-1-4666-2065-0.ch006

Dynamic. Static algorithms assign all priorities at design time and it remains constant for the lifetime of the task. Dynamic algorithms assign priority at runtime, based on execution parameters of tasks.

The ACO algorithms are computational models inspired by the collective foraging behavior of ants. Each ant is an autonomous agent that constructs a path. There might be one or more ants concurrently active at the same time. Ants do not need synchronization. Forward ant moves to the good-looking neighbor from the current node, probabilistically. Probabilistic choice is biased by Pheromone trails previously deposited and heuristic function. Without heuristics information, algorithm tends to converge towards initial random solution. In backward mode, ants lay down the pheromone. In ACO algorithm, pheromone is added only to arcs belonging to the global best solution (Dorigo & Gambardella, 1997). Pheromone intensity of all the paths decreases with time, called pheromone evaporation. It helps in unlearning poor quality solution (Dorigo & Stutzle, 2004).

ACO based scheduling algorithm has been presented for single processor real-time system (Shah & Kotecha, 2010). Moreover, ACO has been applied for heterogeneous multiprocessor systems (Turneo et al., 2008; Chang, Wu, Shann, & Chung, 2008) and reconfigurable parallel processing system (Saad, Adawy, & Habashy, 2006). In this paper, ACO based dynamic scheduling algorithm for multiprocessor real-time operating system has been proposed.

## RELATED WORK

A multiprocessor system is tightly coupled so that global status and workload information on all processors can be kept current at a low cost. The system has shared memory and generally uses centralized scheduler. If system uses separate scheduler for each processor, the decisions and actions of the schedulers of all the processors are coherent. Multiprocessor systems are divided in two basic types, homogeneous and heterogeneous. In homogeneous system, processors can be used interchangeably and in contrast, heterogeneous processors cannot be used interchangeably. Homogeneous processors can be subdivided in two types: identical and uniform. In identical processors, it is assumed that all processors are equally powerful whereas uniform multiprocessor machine is characterized by a speed (Baruah, Funk, & Goossens, 2003).

There are two main strategies when dealing with the problem of multiprocessor scheduling: partitioning strategy and global strategy (Oh & Son, 1995). In a partitioning strategy, once a task is allocated to a processor, all of its instances are executed exclusively on that processor. In a global strategy, any instance of a task can be executed on any processor, or even be preempted and moved to a different processor before it is completed (Lopez, Diaz, & Garcia. 2004).

EDF (Earliest Deadline First) and LLF (Least Laxity First) algorithms are proved optimal under the condition that tasks are preemptive, there is only one processor and it is not overloaded (Dertouzos & Ogata, 1974; Mok, 1983). EDF is appropriate algorithm to use for on-line scheduling on uniform multiprocessors (Funk, Goossens, & Baruah, 2001). However, many practical instances of multiprocessor real-time system are NP-complete, i.e. it is believed that there is no optimal polynomial-time algorithm for them (Ramamritham, Stankovik, & Shiah, 1990; Ullman, 1973).

The scheduling is considered as on-line, if scheduler makes scheduling decision without knowledge about the task that will be released in the future. On-line scheduling algorithms cannot work efficiently in overloaded conditions. Researchers have proved that, for single processor system, the competitive factor of an on-line scheduling algorithm is at most equal to 0.25 when the system is highly overloaded and its value can't be more than 0.385 when the system is slightly over-loaded. They have further derived the up-