International Journal of Innovative Computing, Information and Control
Volume **5**, Number **9**, September **2009**

ICIC International ©2009 ISSN 1349-4198
pp. 2531–2540

OBSERVATIONS ON USING PROBLEM-SPECIFIC GENETIC ALGORITHM FOR MULTIPROCESSOR REAL-TIME TASK SCHEDULING

Yajun Li[1], Yuhang Yang[1], Liang Zhou[1] and Rongbo Zhu[2]

[1]Department of Electronic Engineering
Shanghai Jiaotong University
800 DongChuan Road, Shanghai, 200240, P. R. China
yajunli@sjtu.edu.cn

[2]College of Computer
Science South-Central University for Nationalities
Wuhan 430074, P. R. China

Received March 2008; revised September 2008

Abstract. *Task scheduling is crucial to the performance improvement of multiprocessor systems. Genetic algorithms are extensively used to deal with task scheduling due to the computational intractability of such issues. However, since the genetic algorithm aims to be a generic solution to a variety of problem types, it hardly exploits problem-specific search techniques which might help speed up the search or lead to a better solution. That may compromise the potential power of the genetic algorithms a lot. To overcome this, a problem-specific genetic algorithm is proposed to handle multiprocessor real-time task scheduling in this paper. Rather than only employing limited problem-specific information, our proposal makes the most of such information throughout the evolution of the genetic algorithm. The simulation results show that the performance of the genetic algorithm is greatly improved with the help of certain problem-specific knowledge.*

Keywords: Scheduling, Genetic algorithm, Real-time, Multiprocessor

1. **Introduction.** Real-time task scheduling is of primary significance to multiprocessor systems. The problem of real-time task scheduling in multiprocessor systems is determining when and on which processor a given task executes [1,2]. This can be done either statically or dynamically. Nevertheless, neither static nor dynamic scheduling algorithm is computationally tractable even under simplified assumptions [2]. As a result, many polynomial-time heuristics are reported to tackle the problem under more pragmatic situations [3-5]. The rationale of these heuristics is to sacrifice optimality for the purpose of reduced time complexity.

Recently, Genetic Algorithms (hereafter referred to as GAs) [6,7] have attracted much attention in multiprocessor scheduling [8-14]. The fundamentals of GAs are that they mimic the principle of nature to guide the search in the problem space so as to find a "better" solution. The most interesting quality of GAs is that they can avoid the problem of being trapped in local optima which is often seen in greedy search algorithms.

A central goal of the research efforts in genetic algorithms is to find a generic algorithm that is robust and performs well across a variety of problem types [7]. In their basic form, GAs are problem-independent procedures. They exploit only the coding and the objective function value to determine plausible trails in the next generation. GAs scarcely use any problem-specific techniques, which might speed up the search or may lead to a better solution. This puts the genetic algorithms at a competitive disadvantage with methods that make use of problem-specific information. Therefore, it may be advantageous to