Parallel Scatter Search Algorithms for Exam Timetabling

Nashat Mansour, Lebanese American University, Lebanon
Ghia Sleiman-Haidar, Lebanese American University, Lebanon

ABSTRACT

University exam timetabling refers to scheduling exams into predefined days, time periods and rooms, given a set of constraints. Exam timetabling is a computationally intractable optimization problem, which requires heuristic techniques for producing adequate solutions within reasonable execution time. For large numbers of exams and students, sequential algorithms are likely to be time consuming. This paper presents parallel scatter search meta-heuristic algorithms for producing good sub-optimal exam timetables in a reasonable time. Scatter search is a population-based approach that generates solutions over a number of iterations and aims to combine diversification and search intensification. The authors propose parallel scatter search algorithms that are based on distributing the population of candidate solutions over a number of processors in a PC cluster environment. The main components of scatter search are computed in parallel and efficient communication techniques are employed. Empirical results show that the proposed parallel scatter search algorithms yield good speed-up. Also, they show that parallel scatter search algorithms improve solution quality because they explore larger parts of the search space within reasonable time, in contrast with the sequential algorithm.

Keywords: Exam Timetabling, Metaheuristics, Parallel Algorithm, Scatter Search

INTRODUCTION

Wren (1999) defined timetabling as “the allocation, subject to constraints, of given resources to objects being placed in space time, in such a way as to satisfy as nearly as possible a set of desirable objectives”. The university timetabling problem is of two types: exam timetabling and course timetabling. In this work, we are concerned with the exam timetabling problem. Exam timetabling consists of allocating periods and rooms to a set of exams, subject to constraints, which can be hard or soft (Qu et al., 2009). An example of hard constraints is room capacity. Examples of soft constraints are number of students having consecutive exams and number of students with multiple exams per day. In our case, we consider a predefined number of periods and a limited number of rooms as resources for allocation. Despite the similarity between exam and course timetabling, they are different; they have different objectives and constraints. For example, in course timetabling, a hard constraint is not to schedule courses taught by the same instructor at the same time, whereas this might be requested by some instructors for exam timetabling and in the same room whose capacity must allow such assign-
ment. In addition, even in exam timetabling alone, different problem versions have been reported and addressed in a variety of ways. Furthermore, it is not possible to schedule exams at the same time as the corresponding course, since courses normally last one hour whereas exams last 2-3 hours. Also, students might take 4 or 5 courses on the same days whereas this is considered unacceptable for exams.

The university timetabling problem is an NP-hard optimization problem (Schaerf, 1999). A number of heuristic algorithms have been developed for finding sub-optimal exam timetabling solutions. Examples of these algorithms and techniques are: case-based reasoning technique in combination with a simple simulated annealing algorithm (Burke et al., 2003); hybridization within a graph based hyperheuristic framework (Qu & Burke, 2009); genetic and evolutionary algorithms (Cheong et al., 2009; Cote et al., 2005); clustering and clique algorithms (Carter & Johnson, 2001; Lotfi & Cerveny, 1991); tabu search (Kendall & Hussin, 2005); simulated annealing approaches (Burke et al., 2004; Mansour et al., 2003; Thompson & Dowsland, 1998); scatter search algorithm (Mansour et al., 2009); hybrid heuristics (Azimi, 2005; Bilgin et al., 2007; Pillay & Banzhaf, 2009); heuristic ordering based method combined with backtracking technique (Carter & Johnson, 2001); great-deluge hyper-heuristic (Ozcan et al., 2010). A recent survey of exam timetabling techniques appears in Qu et al. (2009). Clearly, the exam timetabling literature includes a wide variety of approaches and algorithms and presents a variety of objectives and constraints for the problem (Qu et al., 2009) and, thus, different ways to evaluate solutions. This variety is based on theoretical assumptions as well as diverse requirements found, in reality, among different institutions even within the same country (Burke et al., 1996). Hence, methods that address real-world problems based on real-world data are still needed for bridging the gap between research and practice (McCollum, 2007).

For large numbers of students and exams, sequential algorithms are likely to be slow. In this paper, we present a parallel scatter (PSS) search algorithm for exam timetabling to improve both execution time and solution quality. Not much work has been reported on parallel scatter search algorithms. Adenso-Díaz et al. (2006) have proposed parallelization strategies for scatter search for the 0-1 knapsack problem by dividing the scatter search algorithm into 2 phases. Phase I includes tasks related to initial population generation and reference set creation. Phase II includes all tasks related to Subsets creation and combination, solution improvement reference set update. For each one of these phases, they implemented three different methods of processors communication. The first method is single walk strategy, where every processor works independently from other processors. The second method is multiple walk with independent threads strategy, which is a simple replication on processors. The third method is multiple walk with cooperative threads strategy where processors communicate together and share generated information. For all phases, they used both static and dynamic reference set update methods to update the reference set solutions. Emulation results using threads showed a low parallel efficiency and no clear increase in the solution quality. García-Lopez et al. (2003) have developed a parallel scatter search algorithm to solve the p-median problem. They proposed three different parallel techniques: a low-level synchronous parallel scatter search model by using parallel search instead of local search, a replicated combination scatter search model by distributing multiple subsets on the processors, and a natural replication of parallel scatter search. Bożejko and Wodecki (2008) have presented a parallel algorithm based on scatter search and path re-linking methods to solve a flow shop scheduling problem. In this algorithm, the root processor creates the starting solutions set S of size n, while calculations of path re-linking procedures are executed by all processors on local data. On every processor, n/2 randomly chosen pairs from S undergo path re-linking to produce another set S’ of n/2 solutions. The non-root processors send back the S’ solutions to the root processor and the root