An Experimental Study on Hyper-heuristics and Exam Timetabling

Burak Bilgin, Ender Özcan, and Emin Erkan Korkmaz

Artificial Intelligence Laboratory (ARTI), Yeditepe University,
Department of Computer Engineering, 34755 Kadıköy/Istanbul, Turkey
{bbilgin,eozcan,ekorkmaz}@cse.yeditepe.edu.tr

Abstract. Hyper-heuristics are proposed as a higher level of abstraction as compared to the metaheuristics. Hyper-heuristic methods deploy a set of simple heuristics and use only non-problem-specific data, such as fitness change or heuristic execution time. A typical iteration of a hyper-heuristic algorithm consists of two phases: the heuristic selection method and move acceptance. In this paper, heuristic selection mechanisms and move acceptance criteria in hyper-heuristics are analyzed in depth. Seven heuristic selection methods and five acceptance criteria are implemented. The performance of each selection and acceptance mechanism pair is evaluated on 14 well-known benchmark functions and 21 exam timetabling problem instances.

1 Introduction

The term hyper-heuristic refers to a recent approach used as a search methodology [2, 3, 5, 11, 20]. It represents a higher level of abstraction than most of the uses of metaheuristic methods in the literature. Hyper-heuristics involve an iterative strategy that chooses a heuristic to apply to a candidate solution of the problem at hand, at each step. The properties of hyper-heuristics are discussed in [3]. An iteration of a hyper-heuristic can be subdivided into two parts: heuristic selection and move acceptance. In the hyper-heuristic literature, several heuristic selection and acceptance mechanisms are used [2, 3, 5, 11, 20]. However, no comprehensive study exists that compares the performance of these different mechanisms in depth.

Timetabling problems are real-world constraint optimization problems. Due to their NP-complete nature [16], the traditional approaches might fail to generate a solution to a timetabling problem instance. The timetabling problems require assignment of *time-slots* (periods) and possibly some other resources to a set of events, subject to a set of constraints. Numerous researchers deal with different types of timetabling problems based on different types of constraints utilizing a variety of approaches. *Employee timetabling*, *course timetabling* and *examination timetabling* are the research fields that attract the most attention. In this paper, seven heuristic selection methods and five different acceptance criteria are analyzed in depth. Their performance is measured on well-known

E.K. Burke and H. Rudová (Eds.): PATAT 2006, LNCS 3867, pp. 394–412, 2007.

benchmark functions. Moreover, 35 hyper-heuristics, generated by coupling all heuristic selection methods and all acceptance criteria with each other, are evaluated on a set of 21 exam timetabling benchmark problem instances, including the Carter's [10] and Ozcan's [25] benchmarks.

The remainder of this paper is organized as follows. In Section 2 background is provided including the hyper-heuristics, benchmark functions and exam timetabling. Experimental settings and results for benchmarks are given in Section 3. The hyper-heuristic experiments on exam timetabling are presented in Section 4. Finally, the conclusions are discussed in Section 5.

2 Preliminaries

2.1 Hyper-heuristics

Hyper-heuristic methods are described in [3] as an alternative method to techniques that work directly on solution spaces. Most applications of Metaheuristics are 'problem-specific' solution methods, which require knowledge and experience about the problem domain and properties. Metaheuristics are mostly developed for a particular problem and require fine tuning of parameters. Therefore, they can be developed and deployed only by experts who have the sufficient knowledge and experience on the problem domain and the meta-heuristic search method. Hyper-heuristics, on the other hand, are developed to be more general optimization methods, which can be applied to more optimization problems easily. Hyper-heuristics can be considered as black box systems, which take the problem instance and several low-level heuristics as input and which can produce the result independent of the problem characteristics. In this concept, hyper-heuristics use only non-problem-specific data provided by each low-level heuristic in order to select and apply them to the candidate solution [3, 5, 11].

The selection mechanisms in the hyper-heuristic methods were emphasized in the initial phases of the research period. Cowling et al. [11] proposed three types of low-level heuristic selection mechanisms to be used in hyper-heuristics; which are *Simple, Greedy* and *Choice Function*. There are four types of *Simple* heuristic selection mechanisms. *Simple Random* mechanism chooses a low-level heuristic at a time randomly. *Random Descent* mechanism chooses a low-level heuristic randomly and applies it repeatedly as long as it produces improving results. *Random Permutation* mechanism creates an initial permutation of the low-level heuristics and at each iteration applies the next low-level heuristic in the permutation. *Random Permutation Descent* mechanism is the same as *Random Permutation* mechanism, except that it applies the low-level heuristic in turn repeatedly as long as it produces improving results. *Greedy* method calls each low-level heuristic at each iteration and chooses the one that produces the most improving solution. *Choice Function* is the most complex one. It analyzes both the performance of each low-level heuristic and each pair of low-level heuristics. This analysis is based on the improvement and execution time. This mechanism also considers the overall performance. It attempts to focus the search as long as the improvement rate is high and broadens the search if the improvement