A Reinforcement Learning – Great-Deluge Hyper-heuristic for Examination Timetabling

Ender Ozcan
University of Nottingham, United Kingdom
Mustafa Mısır
Yeditepe University, Turkey
Gabriela Ochoa
University of Nottingham, United Kingdom
Edmund K. Burke
University of Nottingham, United Kingdom

ABSTRACT

Hyper-heuristics are identified as the methodologies that search the space generated by a finite set of low level heuristics for solving difficult problems. One of the iterative hyper-heuristic frameworks requires a single candidate solution and multiple perturbative low level heuristics. An initially generated complete solution goes through two successive processes; heuristic selection and move acceptance until a set of termination criteria is satisfied. A goal of the hyper-heuristic research is to create automated techniques that are applicable to wide range of problems with different characteristics. Some previous studies show that different combinations of heuristic selection and move acceptance as hyper-heuristic components might yield different performances. This study investigates whether learning heuristic selection can improve the performance of a great deluge based hyper-heuristic using an examination timetabling problem as a case study.

Keywords: hyper-heuristics, reinforcement learning, great deluge, meta-heuristics, exam timetabling

INTRODUCTION

Meta-heuristics have been widely and successfully applied to many different problems. However, significant development effort is often needed to produce fine tuned techniques for the particular problem or even instance at hand. A more recent research trend in search and optimisation, *hyper-heuristics* (Burke et al., 2003a; Ross, 2005; Chakhlevitch et al., 2008; Ozcan et al., 2008; Burke et al. 2009a, 2009b), aims at producing more general problem solving techniques, which can potentially be applied to different problems or instances with little development effort. A hyper-heuristic approach is able to intelligently choose an appropriate low-level heuristic, from a given repository of heuristics, to be applied at any given time. Thus, in hyper-heuristics, we are interested in adaptively finding solution methods, rather than directly producing a solution for the particular problem at hand.

Several hyper-heuristics approaches have been proposed in the literature, which can be categorised into approaches based on *perturbative* low-level heuristics, and those based on *constructive* low-level heuristics. The latter type of hyper-heuristics builds a solution incrementally, starting with a blank solution, and using constructive heuristics to gradually build a complete solution. They have been successfully applied to several combinatorial optimisation problems such as: bin-packing (Ross et al., 2003), timetabling (Terashima-Marin et al., 1999; Asmuni et al., 2005; Burke et al., 2007, Qu et al., 2008a), production scheduling (Vazquez-Rodriguez et al., 2007), and cutting stock (Terashima-Marin et al., 2005). On the other hand, approaches based on perturbative heuristics, find a reasonable initial solution by some straightforward means (either randomly or using a simple constructive heuristic) and then use heuristics, such as shift and swap to perturb solution components with the aim of finding improved solutions. In other words, they start from a complete solution and then search or select among a set of neighbourhoods for better solutions. Perturbative (improvement) hyper-heuristics have been applied to real world problems, such as, personnel scheduling (Cowling et al., 2001; Burke et al., 2003b), timetabling (Burke et al., 2003b), and vehicle routing problems (Pisinger et al., 2007). In a perturbative hyper-heuristic framework, search is mostly performed using a single candidate solution. Such hyper-heuristics, iteratively, attempt to improve a given solution throughout two consecutive phases: *heuristic selection* and *move acceptance* as illustrated in Figure 1. A candidate solution (S_t) at a given time (t) is modified into a new solution (or solutions) using a selected heuristic (or heuristics). Then, a move acceptance method is employed to decide whether to accept or reject a resultant solution. This process is repeated until a predefined stopping condition is met. Only problem independent information flow is allowed between the problem domain and hyper-heuristic layers. A perturbative hyper-heuristic can be denoted as *heuristic selection* − *move acceptance* based on its components.

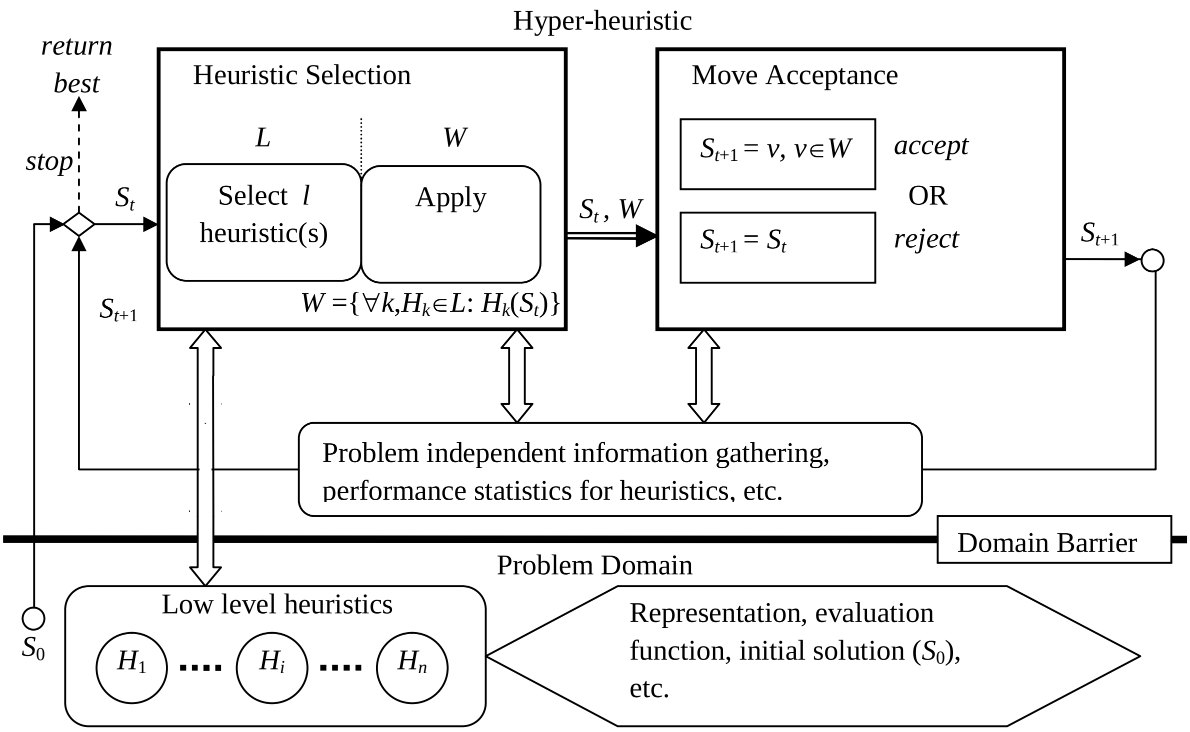

Figure 1. A hyper-heuristic framework based on a single point search

Great deluge is a well-known threshold acceptance criterion (Dueck, 1993). Kendall et al. (2004) employed random choice of low level heuristics (simple random) and great deluge for the first time as hyper-heuristic components for solving the channel assignment problem. Bilgin et al. (2006) experimented with thirty five different hyper-heuristics for solving an examination timetabling. The simple random−great deluge hyper-heuristic ranked the second, yet delivered a similar performance to the best approach; namely, choice function – simulated annealing hyper-heuristic. Obviously, simple random receives no feedback at all during the search to improve upon the heuristic selection process. Hence, in this study, g*reat-deluge* is preferred as the move acceptance component within a perturbative hyper-heuristic framework to investigate the effect of learning heuristic selection on the performance of a perturbative hyper-heuristic for solving the same examination timetabling problem as formulated in Bilgin et al. (2006). The learning mechanisms, inspired by the work in Nareyek (2003), are based on weight adaptation.

BACKGROUND

Hyper-heuristics and Learning

Although hyper-heuristic as a term has been introduced recently (Cowling et al., 2001a), the origins of the idea dates back to early 1960s (Fisher et al., 1961). A hyper-heuristic operates at a high level managing or generating low level heuristics which operate on the problem domain. Meta-heuristics have been commonly used as hyper-heuristics. A hyper-heuristic can conduct a single point or multi-point search. Population based meta-heuristics which perform multi-point search, such as learning classifier systems (Ross et al., 2002; Marín-Blázquez et al., 2005; Tereshima-Marin et al., 2007), evolutionary algorithms (Cowling et al., 2002c; Cowling et al., 2002d; Han et al., 2003a; Han et al., 2003b; Ross et al., 2003, Pillay et al., 2007), genetic programming (Burke et al., 2006; Keller et al., 2007a, Keller et al., 2007b; Burke et al., 2009a), ant colony optimisation (Burke et al., 2005b; Cuesta-Canada et al., 2005; Chen et al., 2007) have been applied to a variety of combinatorial optimisation problems as hyper-heuristics. Distributed computing methods can also be used to perform multi-point search (Rattadilok et al., 2004; Rattadilok et al., 2005; Ouelhadj et al., 2008). Ozcan et al. (2008) presented different hyper-heuristic frameworks showing that a matching performance to memetic algorithms can be achieved. In this study, perturbative hyper-heuristics using the framework as in Figure 1 based a single point search are in focus. The primary components of such hyper-heuristics are heuristic selection and move acceptance.

A major feature of a hyper-heuristic is its applicability to different problem instances having different characteristics as well as different problem domains. In this quest, machine learning techniques are vital for hyper-heuristics to make the right choices during the heuristic selection process. Existing learning hyper-heuristics incorporate *reinforcement learning* (Kaelbling et al., 1996; Sutton et al., 1998). A reinforcement learning system interacts with the environment and changes its state via a selected action in such a way to increase some notion of long term reward. Hence, a learning hyper-heuristic maintains a utility value obtained through predetermined reward and punishment schemes for each low level heuristic. A heuristic is selected based on the utility values of low level heuristics in hand at each step. Remembering and forgetting are the core ingredients of learning. Remembering can be achieved through the rewarding and punishment schemes. Forgetting can be achieved through the use of lower and upper bounds on the utility values. Some reinforcement learning methods use weighted average of learnt utility values. A dynamic weighting scheme can be employed favouring the outcome of the most recent actions or choices. Reward and punishment schemes are allowed to use different adaptation rates in case of an improving and worsening move, respectively. For example, utility value of a selected heuristic can be increased at a constant rate linearly whenever there is an improvement after it is employed, otherwise the utility value can be decreased at a different rate, or even the utility value can be kept constant. Initialisation of the utility values, lower and upper bounds for them along with a memory adjustment scheme (weighting) are the rest of the constituents for a reinforcement learning based hyper-heuristic.

Some previously studied heuristic selection methods are summarised in Table 1. Simple random, random gradient, random permutation gradient, greedy and choice function heuristic selection methods are presented in Cowling et al. (2001a). All these approaches can be considered learning heuristic selection methods, except simple random. In (Cowling et al., 2001b), a parameter-free choice function was presented. As a problem domain, sales summit scheduling was used in both studies. Later, the choice function based hyper-heuristics were applied to nurse rostering (Cowling et al., 2002a) and project presentation scheduling (Cowling et al., 2002b). Cowling & Chakhlevitch (2003) investigated peckish heuristic selection strategies that eliminated the selection and application of all low level heuristics as in greedy heuristic selection.

Nareyek (2003) investigated reinforcement learning using different reward/penalty schemes and heuristic selection strategies on orc quest problem and logistics domain. Additive/subtractive adaptation rates combined with heuristic selection using the maximal utility generated better results as opposed to a fair random choice (softmax, roulette wheel). All heuristics were assigned to a utility value of 0, initially and raw utility values were maintained. Upper and lower bounds were defined for the utility values. In (Burke et al., 2003b), reinforcement learning was combined with tabu search in a hyper-heuristic and applied to timetabling problems. The aim of this modification was to prevent the selection of some heuristics for a while by inserting them into a variable-length tabu list. A non-tabu heuristic with the highest utility value was chosen at each step.

Table 1. Description of a set of heuristic selection methods used within perturbative hyper-heuristics.

Name	**Description**
Simple Random	Choose a low level heuristic randomly
Random Descent	Choose a low level heuristic randomly and employ the same heuristic as long as the candidate solution in hand is improved
Random Permutation Descent	Generate a random permutation of low level heuristics and form a cyclic list. Starting from the first heuristic, employ it repeatedly until a worsening move is hit, then go to the next heuristic in the list.
Greedy	Apply all low level heuristics to the same candidate solution separately and choose the heuristic that generates the best change in the objective value
Peckish	Apply a subset of all low level heuristics to the same and choose the heuristic that generates the best change in the objective value

Choice Function	Dynamically score each heuristic weighing their individual performance, combined performance with previously invoked heuristic and the time passed since the last call to the heuristic at a given step then a heuristic is chosen based on these scores.
Reinforcement Learning	Each heuristic carries a utility value and heuristic selection is performed based on these values. This value gets updated at each step based on the success of the chosen heuristic. An improving move is rewarded, while a worsening move is punished using a preselected adaptation rate.
Tabu Search	This method employs the same strategy as Reinforcement Learning and uses a tabu list to keep track of the heuristics causing worsening moves. A heuristic is selected which is not in the tabu list.

Some studies concentrate on move acceptance in hyper-heuristics rather than the heuristic selection methods, as accepting a move turns out to be an extremely important decision. In Cowling et al. (2001), heuristic selection methods are combined with either all moves accepted or only improving moves accepted strategy. On the other hand, Ayob & Kendall (2003) proposed three different Monte Carlo move acceptance strategies based on the objective value change due to the move, time (units), number of consecutive non-improving moves. Simple random was used as a heuristic selection within the hyper-heuristic for solving the component placement problem. The best move acceptance turned out to be *exponential Monte Carlo with counter*. One of the well known move acceptance is *simulated annealing* (SA) (Kirkpatrick, 1983). The improving moves or the moves that generate an equal quality solution are accepted, while a worsening move is not rejected immediately. Acceptance of a given candidate solution is based on a probabilistic framework that depends on the objective value change and a temperature that decreases in time (cooling). The difference between exponential Monte Carlo with counter and the simulated annealing is that the latter one uses this *cooling schedule* while the former does not. Bai & Kendall (2003) investigated the performance of simple random – simulated annealing hyper-heuristic on a shelf space allocation problem. Anagnostopoulos et al. (2006) applied a similar hyper-heuristic to a set of travelling tournament problem instances embedding a reheating scheme into the simulated annealing move acceptance. In (Bai et al., 2007), a reinforcement learning scheme is combined with simulated annealing with reheating as a hyper-heuristic and applied to three different problem domains: nurse rostering, course timetabling and 1D bin packing.

In (Dueck, 1993), two move acceptance strategies, namely *great deluge* (GD) and *record-to-record travel* that accept worsening moves based on a dynamic threshold value were presented. Kendall & Mohamad (2004) utilised a simple random – great deluge hyper-heuristic to solve a mobile telecommunication network problem. Great deluge uses a threshold (τ_t) that decreases in time linearly to determine an acceptance range for the solution qualities as presented in Equation (1), where *maxIter* is the maximum number of steps (or total time), t is the number of steps passed, ΔR is an expected range for the maximum fitness change between the initial fitness and f_{opt} which is the final objective value (e.g., lower bound).

$$\tau_t = f_{opt} + \Delta R \left(1 - \frac{t}{maxIter}\right) \quad (1)$$

In case of an improving move, it is accepted, while a worsening move is accepted as well only if the objective value of the resultant candidate solution at step *t* is less than the computed threshold. Kendall & Mohamad (2004) used a step based threshold formula with a maximum number of iterations as a termination criterion aiming a quadratic running time complexity for the overall algorithm.

Bilgin et al. (2007) employed different heuristic selection and move acceptance mechanisms and used their combinations as hyper-heuristics. The results showed that simple random – great deluge hyper-heuristic was the second best after choice function – simulated annealing considering the average performance of all hyper-heuristics over a set of examination timetabling problems. Consequently, a hyper-heuristic without learning delivered a comparable performance to another one with a learning mechanism. Therefore, in this study, reinforcement learning is preferred to be combined with great deluge to observe the effect of learning heuristic selection on the overall performance of the hyper-heuristic for solving the same problem. All the runs during the experiments in (Bilgin et al., 2007) were restricted to 600 seconds; hence, the threshold is computed based on the CPU time within the great deluge move acceptance strategy. If a heuristic takes less time, then the threshold value will be lower as compared to the one that takes longer time. This hyper-heuristic differs from the one that Kendall & Hussin (2005) have investigated, as their hyper-heuristic embeds a tabu list approach to keep the chosen heuristic from getting selected again for a number of steps (tabu duration) into reinforcement learning as a heuristic selection. Moreover, the low level heuristics contained a mixture of thirteen different constructive and perturbative low level heuristics.

Ozcan et al. (2009) combined different heuristic selection methods with late acceptance strategy, a new method that is initially presented as a local search for solving examination timetabling problem. Late acceptance requires a single parameter and it is a memory based approach. A trial solution is compared with a previously visited solution at a fixed distance apart from the current step in contrast to the conventional approaches that usually compare the trial solution with a current one. The trial solution is accepted, if there is an improvement over this previously visited solution. The results showed that reinforcement learning, reinforcement learning with tabu list or choice function heuristic selection methods did not improve the performance of the hyper-heuristic if late acceptance is used. Choosing a heuristic randomly at each step performed the best. More on hyper-heuristics can be found in Cowling et al. (2002b), Burke et al. (2003a), Ross (2005), Ozcan et al. (2008), Burke et al. (2009), Chakhlevitch & Cowling (2008).

The Reinforcement Learning – Great Deluge Hyper-heuristic

Reinforcement Learning (RL) is a general term for a set of widely used approaches that provide a way to learn how to behave when an action comes or “*how to map situations to actions*" (Sutton & Barto, 1998) through *trail-and-error* interactions (Kaelbling et al., 1996). A perturbative hyper-heuristic combining reinforcement learning heuristic selection and great deluge move acceptance is implemented as shown in Figure 2. As suggested in Nareyek (2003), additive adaptation rate that increments the utility value of the low level heuristic is used in case of an improvement as a reward at step 14. This value is tested against three different negative adaptation rates, namely subtractive, divisional and root, denoted as RL_1, RL_2 and RL_3, respectively for the punishment of a heuristic causing a worsening move at step 17:

$$RL_1 : u_i = u_i - 1 \tag{2}$$

$$RL_2 : u_i = u_i / 2 \tag{3}$$

$$RL_3 : u_i = \sqrt{u_i} \tag{4}$$

RL–GD ALGORITHM

Input – *n*: number of heuristics, *u*: array holding utility value for each heuristic, *totalTime*

```
1.  // initialisation
2.  Generate a random complete solution S_current ;
3.  Initialise utility values;
4.  f_current = f_0 = quality( S_current );
5.  startTime = t = time(); level = f_current
6.  // main loop executes until total running time allowed is exceeded
7.  while ( t < totalTime ) {
8.      // heuristic selection
9.      i = selectHeuristic( u ); // select a heuristic using the utility values
10.     S_temp = applyHeuristic( i );
11.     f_temp = quality( S_temp );
12.     t  = time() – startTime;
13.     // move acceptance
14.     if (f_temp < f_current ) then {
15.             u_i = reward( u_i ); // improving move
16.             S_current = S_temp;
17.     } else {
18.             u_i = punish( u_i ); // worsening move
19.             if (  f_temp <  qualityLB + (f_0 – qualityLB)(1 – t/totalTime) ) then
20.                     S_current = S_temp;  // accept the move else reject the move
21.     }
22. }
```

Figure 2. Pseudocode of a Reinforcement Learning – Great Deluge hyper-heuristic

Memory length is implemented not only in terms of adaptation rates, but also using a lower and an upper bound on the utility values. We experimented with four different ranges in [0,*number_of_heuristics*×(5*i*)], i={1,2,3,4}. It is assumed that these bounds are checked during the steps 14 and 17. Optimistic initial utility values are utilised and all utilities are set to $\lfloor 0.75 \times \textit{upper bound} \rfloor$ at step 3 to support exploration. As the environment might change dynamically, bounds on the utility values are essential in order to encourage exploration in further steps. Reinforcement learning is based on the idea that heuristics getting large rewards should be more likely to be selected again, while heuristics getting small rewards should be less likely to be selected again. The reinforcement scheme used returns the same reward for all heuristic choices. Hence, using maximal utility value to select a heuristic is a reasonable choice. Moreover, selecting the heuristic with this strategy that will be denoted as *max* is reported in (Nareyek, 2003) to be the best choice for step 9. If there are multiple low level heuristics under consideration, since their utility values are the same, then a random choice is made. Another approach to decide whether a given total reward is small or large can be achieved by comparing that value to a relative *reference reward*, such as the average of all utility values. Additional to the maximal utility, another heuristic selection scheme that chooses a low level heuristic randomly from the ones that are over (and equal to) the average, denoted as *overAvr* is implemented. The lower bound (*qualityLB*) is set to -1 at step 19 considering the evaluation function (Equation 1) during the experiments.

REFERENCES

Abdullah, S., Ahmadi, S., Burke, E. K., & Dror, M. (2004). Applying Ahuja-Orlin's large neighbourhood for constructing examination eimetabling solution. In E. K. Burke and M. Trick (Ed.), *the 5th International Conference on the Practice and Theory of Automated Timetabling (PATAT'04)*, (pp. 413-419).

Abdullah, S., Ahmadi, S., Burke, E. K., & Dror, M. (2007). Investigating Ahuja-Orlins large neighbourhood search for examination timetabling. *OR Spectrum*, *29*(2), (pp. 351-372).

Alkan, A., & Ozcan, E. (2003). Memetic algorithms for timetabling, *Proceedings of 2003 IEEE Congress on Evolutionary Computation*, (pp. 1796-1802).

Alkan, A., & Ozcan, E. (2003). Memetic algorithms for timetabling, In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'03)*, (pp. 1796-1802).

Anagnostopoulos, A., Michel, L., Hentenryck, P. V., & Vergados, Y. (2006). A simulated annealing approach to the traveling tournament problem. *Journal of Scheduling*, *9*, (pp. 177-193), Kluwer Academic Publishers.

Asmuni, H., Burke, E. K., Garibaldi, J. M., & McCollum, B. (2007). A novel fuzzy approach to evaluate the quality of examination timetabling. In *Proceedings of the 6th International Conference on the Practice and Theory of Automated Timetabling (PATAT'06), LNCS, 3867*, (pp. 327-346), Springer.

Atsuta, M., Nonobe, K., & Ibaraki, T. (2007). ITC2007 Track 1: An Approach using general CSP solver. http://www.cs.qub.ac.uk/itc2007/.

Ayob, M., & Kendall, G. (2003). A monte carlo hyper-Heuristic to optimise component placement sequencing for multi head placement machine. In *Proceedings of the International Conference on Intelligent Technologies (InTech'03)* (pp. 132-141).

Azimi, Z. N. (2005). Hybrid heuristics for examination timetabling problem. *Applied Mathematics and Computation 163*(2):705-733.

Bai, R., Blazewicz, J., Burke, E., Kendall, G., & McCollum, B. (2007). *A simulated annealing hyper-heuristic methodology for flexible decision support*. Technical Report, Univeristy of Nottingham.

Bai, R., Burke, E. K., Kendall, G. & McCollum, B. (2007) Memory length in hyper-heuristics: an empirical study. In *Proceeding of 2007 IEEE Symposium on Computational Intelligence in Scheduling (CISched2007)*, (pp.173-178).

Bai, R., & Kendall, G. (2003). An investigation of automated planograms using a simulated annealing based hyper-heuristics. In Ibaraki, T.; Meta-heuristics: Progress as Real Problem Solvers, Nonobe, K. & Yagiura, M. (Ed.), selected papers from *the 5th Metaheuristics International Conference (MIC'03)* (pp. 87-108), Springer.

Bilgin, B., Ozcan, E., & Korkmaz, E. E. (2007). An experimental study on hyper-heuristics and exam scheduling. In *Proceedings of the 6th International Conference on the Practice and Theory of Automated Timetabling (PATAT'06), LNCS 3867*, (pp. 394-412), Springer.

Broder, S. (1964). Final examination scheduling. *Communications of the ACM 7*(8), 494-498.

Burke, E. K., Bykov, Y., Newall, J. P., & Petrovic, S. (2004). A time-predefined local search approach to exam timetabling problems. *IIE Transactions on Operations Engineering, 36*(6), 509-528.

Burke, E. K., Dror, M., Petrovic, S., & Qu, R. (2005a). Hybrid graph heuristics within a hyper-heuristic approach to exam timetabling problems. In B.L. Golden, S. Raghavan and E. A. Wasil (Ed.), *The Next Wave in Computing, Optimization, and Decision Technologies, Conference Volume of the 9th informs Computing Society Conference*, (pp. 79–91), Springer.

Burke, E., Hyde, M., & Kendall, G. (2006). Evolving bin packing heuristics with genetic programming. In Runarsson, T.; H-G.B.; Burke, E.; Merelo-Guervos, J.; Whitley, L. & Yao, X. (Ed.), *Proceedings of the 9th Parallel Problem Solving from Nature (PPSN'06), 4193*, (pp. 860-869), Springer-Verlag.

Burke, E., Hyde, M., Kendall, G., Ochoa, G., Ozcan, E., & Woodward, J. R. (2009a). Exploring hyper-heuristic methodologies with genetic programming. In C. Mumford & L. Jain (Ed.), *Computational Intelligence: Collaboration, Fusion and Emergence*, to appear.

Burke, E., Hyde, M., Kendall, G., Ochoa, G., Ozcan, E., & Rong, Q. (2009b). *Survey of hyper-heuristics*. Technical Report, Univeristy of Nottingham.

Burke, E., Kendall, G., Newall, J., Hart, E., Ross, P., & Schulenburg, S. (2003a). Hyper-heuristics: an emerging direction in modern search technology. In Glover and Kochenberger (Ed.), *Handbook of Metaheuristics, 57*, (pp. 457-474), Kluwer International Publishing.

Burke, E., Kendall, G., Silva, D., O'Brien, R., & Soubeiga, E. (2005b). An ant algorithm hyperheuristic for the project presentation scheduling problem. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'05), 3*, (pp. 2263-2270).

Burke, E., Kendall, G., & Soubeiga, E. (2003b). A tabu-search hyper-heuristic for timetabling and rostering. *Journal of Heuristics, 9*, 451-470, Kluwer Academic Publishers.

Burke, E. K., McCollum, B., Meisels, A., Petrovic, S., & Qu, R. (2007) A graph-based hyper-heuristic for educational timetabling problems. *European Journal of Operational Research 176*(1),177-192.

Burke, E. K., & Newall, J. P. (2004). Solving examination timetabling problems through adaption of heuristic orderings: models and algorithms for planning and scheduling problems. *Annals of Operations Research, 129*, 107–134.

Burke, E. K., & Newall, J. P. (1999). A multi-stage evolutionary algorithm for the timetable problem. *IEEE Transactions on Evolutionary Computation, 3*(1), 63-74.

Burke, E. K., Newall, J. P., & Weare, R. F. (1995). A memetic algorithm for university exam timetabling. In *Proceedings of the 1st International Conference on the Practice and Theory of Automated Timetabling (PATAT'95),* (pp. 241-250).

Caramia, M., DellOlmo, P., & Italiano, G. F. (2001). New algorithms for examination timetabling. In S. Naher & D. Wagner (Ed.), *Algorithm Engineering 4th International Workshop (WAE'00), LNCS, 1982*, (pp. 230-241), Springer.

Caramia, M., & Dell'Olmo, P. (2007). Coupling stochastic and deterministic local search in examination timetabling. *Operations Research*, *55*(2).

Carter, M. W., Laporte, G., & Lee, S. (1996). Examination timetabling: algorithmic strategies and applications. *Journal of the Operational Research Society, 47*(3), 373-383.

Casey, S., & Thompson, J. (2003). GRASPing the examination scheduling problem. In E. K. Burke & P. De Causmaecker (Ed.)*, Practice and Theory of Automated Timetabling: Selected Papers from the 4th International Conference*. *LNCS, 2740*, (pp. 232-244), Springer.

Chakhlevitch, K., & Cowling, P. I. (2008). Hyperheuristics: recent developments. *Adaptive and Multilevel Metaheuristics*, (pp. 3-29).

Chen, P.-C., Kendall, G., & Berghe, G. V. (2007). An ant based hyper-heuristic for the travelling tournament problem. *Proceedings of IEEE Symposium of Computational Intelligence in Scheduling (CISched'07),* (pp. 19-26).

Cheong, C. Y., Tan, K. C., & Veeravalli, B. (2007). Solving the exam timetabling problem via a multi-objective evolutionary algorithm – a more general approach. In *Proceedings of the IEEE Symposium on Computational Intelligence in Scheduling (CI-Sched'07),* (pp. 165-172).

Cole, A. J. (1964). The preparation of examination timetables using a small-store computer. *Computer Journal, 7,* (pp. 117-121).

Corne, D., Ross, P., & Fang, H. L. (1994). Fast practical evolutionary timetabling. *AISB Workshop on Evolutionary Computing*, (pp. 250-263).

Corr, P. H., McCollum, B., McGreevy, M. A. J., & McMullan, P. (2006). A new neural network based construction heuristic for the examination timetabling problem. T. P. Runarsson et al. (eds.), *PPSN IX*, *LNCS* 4193, (pp. 392-401).

Cowling, P., Kendall, G., & Han., L. (2002d). An adaptive length chromosome hyperheuristic genetic algorithm for a trainer scheduling problem. *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution And Learning (SEAL'02),* (pp. 267-271), November 18-22, Orchid Country Club, Singapore.

Cowling, P., Kendall, G., & Han, L. (2002c). An investigation of a hyperheuristic genetic algorithm applied to a trainer scheduling problem. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'02),* (pp. 1185-1190).

Cowling P., Kendall, G., & Hussin N. M. (2002). A survey and case study of practical examination timetabling problems. In *Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling (PATAT'02),* (pp. 258-261).

Cowling, P., Kendall, G., & Soubeiga, E. (2001a). A hyperheuristic approach to scheduling a sales summit. In *Proceedings of the 3rd International Conference on Practice and Theory of Automated Timetabling (PATAT'00)*, (pp. 176-190), Springer-Verlag.

Cowling, P., Kendall, G., & Soubeiga, E. (2001b). A parameter-free hyperheuristic for scheduling a sales summit. In *Proceedings of 4th Metaheuristics International Conference (MIC'01)*, (pp. 127-131).

Cowling, P., Kendall, G., & Soubeiga, E. (2002a). Hyperheuristics: a robust optimisation method applied to nurse scheduling. In *Proceedings of the 7th International Conference on Parallel Problem Solving from Nature (PPSN'02)*, (pp. 851-860), Springer-Verlag.

Cowling, P., Kendall, G., & Soubeiga, E. (2002b). Hyperheuristics: a tool for rapid prototyping in scheduling and optimisation. In *Proceedings of the Applications of Evolutionary Computing on EvoWorkshops*, (pp. 1-10), Springer-Verlag.

Cowling, P., & Chakhlevitch, K. (2003). Hyperheuristics for managing a large collection of low level heuristics to schedule personnel. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'03)*, (pp. 1214-1221), IEEE Computer Society Press, Canberra, Australia.

Cuesta-Canada, A., Garrido, L., & Terashima-Marin, H. (2005). Building hyper-heuristics through ant colony optimization for the 2d bin packing problem. *Proceedings of the 9th International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES'05), LNCS, 3684*, (pp. 654–660).

De Smet, G. (2008). ITC2007 - Examination track, *Practice and Theory of Automated Timetabling (PATAT'08)*, Montreal, 19-22, August 2008.

Di Gaspero, L., & Schaerf, A. (2001). Tabu search techniques for examination timetabling. In Burke E.K. and Erben, W. (Ed.), selected papers from *the 3rd International Conference on the Practice and Theory of Automated Timetabling (PATAT'00), LNCS, 2079.* (pp. 104-117).

Dowsland, K., & Thompson, J. (2005). Ant colony optimization for the examination scheduling problem. *Journal of the Operational Research Society*, *56*(4), (pp. 426-438).

Dueck, G. (1993). New optimization heuristics: the great deluge algorithm and the record-to-record travel. *Journal of Computational Physics, 104*, (pp. 86-92), Academic Press Professional, Inc.

Eley, M. (2006). Ant algorithms for the exam timetabling problem. In *Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling (PATAT'04)*, (pp. 364-382).

Erben, W. (2001). A grouping genetic algorithm for graph colouring and exam timetabling. In Burke, E. K. and Erben, W., (Ed.), *the 3rd International Conference on the Practice and Theory of Automated Timetabling (PATAT'00), LNCS, 2079*, (pp. 132-156), Springer, Berlin.

Ergul, A. (1996). GA-based examination scheduling experience at Middle East Technical University. *Practice and Theory of Automated Timetabling, Springer, LNCS, vol 1153*, (pp. 212-226).

Ersoy, E, Ozcan, E., & Uyar, S. (2007). Memetic algorithms and hyperhill-climbers. In *Proceedings of the 3rd Multidisciplinary International Scheduling Conference: Theory and Applications (MISTA'07)*, (pp. 156-166).

Even, S., Itai, A., & Shamir, A. (1976). On the complexity of timetable and multicommodity Flow problems. *SIAM Journal on Computing* 5(4):691-703.

Fisher, H., & Thompson, G. L. (1961) Probabilistic learning combinations of local job-shop scheduling rules. In *Factory Scheduling Conference*, Carnegie Institue of Technology.

Gogos, C., Alefragis, P., & Housos, E. (2008). A multi-staged algorithmic process for the solution of the examination timetabling problem. In *Proceedings of the 7th International Conference on Practice and Theory of Automated Timetabling (PATAT'08)*, Montreal.

Han, L., & Kendall, G. (2003a). An investigation of a tabu assisted hyper-heuristic genetic algorithm. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'03), 3*, (pp. 2230-2237).

Han, L., & Kendall, G. (2003b). Guided operators for a hyper-heuristic genetic algorithm. In T. D. Gedeon and L.C.C. Fung, (Ed.), *Proceedings of AI-2003: Advances in Artificial Intelligence. The 16th Australian Conference on Artificial Intelligence (AI'03)*, *2903*, (pp. 807-820).

Kaelbling, L. P., Littman, M., & Moore, A. (1996). Reinforcement learning: a survey. *Journal of Artificial Intelligence Research, 4*, (pp. 237-285).

Keller, R. E., & Poli, R. (2007a). Cost-benefit investigation of a genetic-programming hyper-heuristic. In *Proceedings of the 8th International Conference on Artificial Evolution (EA'07),* Tours, France, (pp. 13-24).

Keller, R. E., & Poli, R. (2007b). Linear genetic programming of parsimonious metaheuristics. In: *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'07),* Singapore, (pp. 4508-4515).

Keller, R. E., & Poli, R. (2007b). Linear genetic programming of parsimonious metaheuristics. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'07),* Singapore, (pp. 4508-4515).

Kendall, G., & Hussin, N. M. (2005). Tabu search hyper-heuristic approach to the examination timetabling problem at university of technology MARA. In E. K. Burke and M. Trick (Ed.), *Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling (PATAT'04), LNCS, 3616*, (pp. 270-293), Springer.

Kendall, G., & Mohamad, M. (2004). Channel assignment in cellular communication using a great deluge hyper-heuristic. In *Proceedings of the 12th IEEE International Conference on Network (ICON'04),* (pp. 769-773).

Kirkpatrick, S., Gelatt, C., & Vecchi, M. (1983). Optimization by simulated annealing. *Science, 220*, (pp. 671-680).

Marin, H. T. (1998). *Combinations of GAs and CSP strategies for solving examination timetabling problems*. Ph.D. Thesis, Instituto Tecnologico y de Estudios Superiores de Monterrey.

Marín-Blázquez, J., & Schulenburg, S. (2005). A hyper-heuristic framework with XCS: learning to create novel problem-solving algorithms constructed from simpler algorithmic ingredients. In Kovacs, T.; Llorà, X.; Takadama, K.; Lanzi, P.; Stolzmann, W. & Wilson, S. (Ed.), the *8th*

International Workshop on Learning Classifier Systems (IWLCS'05), 4399, (pp. 193-218), Springer.

Mccollum, B. (2006). University timetabling: Bridging the gap between research and practice. In: *Proc. of the 5th International Conference on the Practice and Theory of Automated Timetabling*, (pp. 15-35), Springer.

McCollum, B, McMullan, P, Burke, E. K., Parkes, A. J., & Qu, R. (2009). A new model for automated examination timetabling, *Annals of OR*, to appear.

Merlot, L. T. G., Boland, N., Hughes, B. D., & Stuckey, P. J. (2002). A hybrid algorithm for the examination timetabling problem. In: E.K. Burke and P. De Causmaecker (Ed.), *the 4th International Conference on the Practice and Theory of Automated Timetabling (PATAT'02), LNCS, 1153*, (pp. 207-231).

Müller, T. (2008). ITC2007 Solver description: a hybrid approach. *Practice and Theory of Automated Timetabling (PATAT'08)*, Montreal, 19-22, August 2008.

Nareyek, A. (2003). Choosing search heuristics by non-stationary reinforcement learning. *Metaheuristics: Computer Decision-Making*, 523-544, Kluwer Academic Publishers.

Ouelhadj, D., & Petrovic, S. (2008). A cooperative distributed hyper-heuristic framework for scheduling. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC'08),* (pp. 2560-2565).

Ozcan, E., Bilgin, B., & Korkmaz, E. E. (2008). A comprehensive analysis of hyper-heuristics. *Intelligent Data Analysis, 12*, 3-23, IOS Press.

Ozcan, E., & Ersoy, E. (2005). Final exam scheduler – FES. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'05), 2,* (pp. 1356-1363).

Ozcan, E., Bykov, Y., Birben, M., & Burke, K. E. (2009). Examination timetabling using late acceptance hyper-heuristics. In *Proceedings of Genetic and Evolutionary Computation Conference (GECCO'09).*

Paquete, L. F., & Fonseca, C. M. (2001). A study of examination timetabling with multiobjective evolutionary algorithms. In *Proceedings of the 4th Metaheuristics International Conference (MIC'01)*, (pp. 149-154).

Petrovic, S., Patel, V., & Yang, Y. (2005). Examination timetabling with fuzzy constraints. In Burke, E. K. and M. Trick, (Ed.), *the 5th International Conference on the Practice and Theory of Automated Timetabling (PATAT'05), LNCS, 3616*, (pp. 313-333), Springer-Verlag.

Petrovic, S., Yang, Y., & Dror, M. (2003). Case-based initialisation for examination time-tabling. In *Proceedings of 1st Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA'03),* (pp. 137–154).

Petrovic, S., Yang, Y., & Dror, M. (2007). Case-based selection of initialisation heuristics for metaheuristic examination timetabling. *Expert Systems with Applications: An International Journal, 33*(3), (pp. 772-785).

Pillay, A. (2007). Developmental Approach to the Examination timetabling Problem. http://www.cs.qub.ac.uk/itc2007/.

Pillay, N., & Banzhaf, W. (2007). A genetic programming approach to the generation of hyper-heuristics for the uncapacitated examination timetabling problem. In Neves J, Santos MF, Machado J (Ed.), *Progress in Artificial Intelligence, 13th Portuguese Conference on Artificial Intelligence*, *LNCS*, *4874*, (pp. 223–234), Springer.

Pillay, N., & Banzhaf, W. (2008). A study of heuristic combinations for hyper-heuristic systems for the uncapacitated examination timetabling problem. *European Journal of Operational Research* (2008), doi:10.1016/j.ejor.2008.07.023.

Qu, R., Burke, E. K., Mccollum, B., Merlot, L. T., & Lee, S. Y (2009). A survey of search methodologies and automated system development for examination timetabling. *Journal of Scheduling*, *12*(1), (pp. 55-89).

Qu, R., & Burke, E. K. (2005). Hybrid variable neighbourhood hyper-heuristics for exam timetabling problems. In *Proceedings of the 6th Metaheuristic International Conference (MIC'05)*, (pp. 22-26).

Qu, R., & Burke, E. K. (2007). Adaptive decomposition and construction for examination timetabling problems.In *Proceedings of the 3rd Multidisciplinary International Scheduling Conference: Theory and Applications (MISTA'07)*, (pp. 418-425).

Qu, R., Burke, E. K., & McCollum, B. (2008a). Adaptive automated construction of hybrid heuristics for exam timetabling and graph colouring problems. *European Journal of Operational Research*, *198*(2), 392-404.

Qu, R., Burke, E. K., McCollum, B., Merlot, L., & Lee, S. (2008b). A survey of search methodologies and automated system development for examination timetabling. *Journal of Scheduling, DOI: 10.1007/s10951-008-0077-5*.

Rattadilok, P., Gaw, A., & Kwan, R. (2004). *A Distributed Hyper-heuristic for Scheduling*. Technical Report, University of Leeds, School of Computing.

Rattadilok, P., Gaw, A., & Kwan, R. (2005). Distributed choice function hyper-heuristics for timetabling and scheduling. *Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling (PATAT'2004)*, (pp. 51-67), Springer-Verlag.

Ross, P. (2005). Hyper-heuristics. In: Burke, E.K., Kendall, G. (eds), *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, chap 17, (pp. 529-556), Springer.

Ross, P., Corne, D., & Fang, H. L. (1994). Improving evolutionary timetabling with delta evaluation and directed mutation. *Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel Problem Solving from Nature*, (pp. 556-565), Springer-Verlag.

Ross, P., Hart, E., Marin-Blazquez, J., & Schulenberg, S. (2003). Learning a procedure that can solve hard bin-packing problems: a new GA-based approach to hyperheuristics. *Proceedings of Genetic and Evolutionary Computation Conference (GECCO'2003)*, (pp. 1295-1306).

Ross, P., Schulenburg, S., Marn-Blázquez, J., & Hart, (2002). Hyper-heuristics: learning to combine simple heuristics in bin-packing problems. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'02)*, (pp. 942-948), Morgan Kaufmann Publishers.

Schaerf, A., & Di Gaspero, L. (2006). Measurability and reproducibility in timetabling research: State-of-the-art and discussion (invited paper). In E. K. Burke & H. Rudova (Ed.), *the 6th International. Conference on the Practice and Theory of Automated Timetabling (PATAT'06),* (pp. 53-62).

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: an introduction*. The MIT Press.

Tereshima-Marin, H. T., Zarate, C. J. F., Ross, P., & Valenzuela-Rendon, M. (2007). Comparing two models to generate hyper-heuristics for the 2d-regular bin-packing problem. In *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation (GECCO'07),* (pp. 2182-2189).

Terashima-Marin, H., Ross, P., & Valenzuela-Rendon, M. (1999). Evolution of constraint satisfaction strategies in examination timetabling. In *Proceedings of the Genetic and Evolutionary Computation Conference* (GECCO'99), (pp 635-642).

Terashima-Marin, H., Moran-Saavedra, A., & Ross, P. (2005). Forming hyper-heuristics with GAs when solving 2D-regular cutting stock problems. In *Proceedings of the 2005 IEEE Congress on Evolutionary Computation*, *2*, (pp. 1104-1110), IEEE Press.

Thompson, J. M., & Dowsland, K. A. (1996). Variants of simulated annealing for the examination timetabling problem. *Annals of Operations Research, 63*, 105–128.

Thompson, J. M., & Dowsland, K. A. (1998). A robust simulated annealing based examination timetabling system. *Computers & Operations Research, 25,* 637-648.

Vazquez-Rodriguez, J.A., Petrovic, S. & Salhi, A. (2007). A combined meta-heuristic with hyper-heuristic approach to the scheduling of the hybrid flow shop with sequence dependent setup times and uniform machines. In P. Baptiste, G. Kendall, A. Munier-Kordon & F. Sourd (Ed.), *the 3rd Multi-disciplinary International Scheduling Conference: Theory and Applications (MISTA'07)*, (pp. 506-513), Paris, France.

Wong, T., Cote, P., & Gely, P. (2002). Final exam timetabling: a practical approach. In *Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering*, *2*, (pp 726-731).

Yang, Y., & Petrovic, S. (2005). A Novel similarity measure for heuristic selection in examination timetabling. In E. K. Burke & M. Trick (Ed.), *Practice and Theory of Automated Timetabling: Selected Papers from the 5th International Conference*, *LNCS, 3616*, (pp. 377-396), Springer.