Scheduling English Football Fixtures over the Holiday Period Using Hyper-heuristics

Jonathon Gibbs, Graham Kendall, and Ender Özcan

School of Computer Science, University of Nottingham,
Jubilee Campus, Wollaton Road, Nottingham, NG8 1BB, UK
{jxg16u,gxk,exo}@cs.nott.ac.uk

Abstract. One of the annual issues that has to be addressed in English football is producing a fixture schedule for the holiday periods that reduces the travel distance for the fans and players. This problem can be seen as a minimisation problem which must abide to the constraints set by the Football Association. In this study, the performance of *selection hyper-heuristics* is investigated as a solution methodology. Hyper-heuristics aim to automate the process of selecting and combining simpler heuristics to solve computational search problems. A selection hyper-heuristic stores a single candidate solution in memory and iteratively applies selected low level heuristics to improve it. The results show that the learning hyper-heuristics outperform some previously proposed approaches and solutions published by the Football Association.

Keywords: Hyper-heuristic, Metaheuristic, Local Search, Machine Learning, Sports Scheduling.

1 Introduction

The idea behind hyper-heuristics dates back to the 1960s, although the term was introduced by Dezinger et al. [1]. A hyper-heuristic is a high level problem solving methodology that performs a search over the search space generated by a set of low level heuristics [2]. One of the hyper-heuristic frameworks is concerned with automating the process of selecting and combining several simple heuristics to solve a computational search problem. A *selection hyper-heuristic* is based on this framework which operates on a set of perturbative low level heuristics performing a single point search, storing only one solution, and applying different heuristic strategies to determine which heuristic to apply, and move acceptance strategies, to determine whether the move should be accepted or not [3,4]. Bilgin et al. [5] provide a performance comparison of a variety of selection hyper-heuristics which combine different heuristic selection and move acceptance criteria. More on hyper-heuristics can be found in [6,7,8].

Scheduling, particularly sports scheduling, is a largely studied area [9]. In American sports, it is possible to schedule road trips, where a team travels to different locations, stadiums, without returning home. This is not necessary in

England as the distance between any two teams is relatively small. Therefore minimising the travelling distance over the entire season is not possible. The need for scheduling with distance minimisation in English football however does arise over the holiday period (Boxing Day and New Years Day) as fans do not wish to travel long distances during this time of the year. The travelling salesman problem is the problem of connecting each city together and returning back to the starting city [10]. This problem differs from the travelling salesman problem as it is only necessary for each city to visit only one other city.

Two approaches to the problem have been previously made by Kendall [11,12]. The first approach [11] is a two phase process. Depth first search is used to create fixtures for Boxing Day and New Years Day and then a local search algorithm aims to satisfy the remaining constraints to generate a feasible schedule, with the minimal distance possible. Although Kendall provides an improved fixture schedule from those published by the Football Association it could take up to 30 hours for a feasible solution to be created. Kendall [12] then adopts a different approach to improve the existing run time of the previous solution, a CPLEX and Simulated Annealing approach is adopted, reducing the runtime to approximately 4 minutes. In this paper, a set of hyper-heuristics combining different heuristic selection methods and acceptance criteria are applied to the fixture scheduling problem where they are evaluated and compared based on their ability to produce good quality solutions.

2 Preliminaries

Scheduling fixtures is a real world constraint optimisation problem. Due to the large size of the search space, it is impractical to use an exhaustive method, since the computation time becomes excessive [12]. Therefore, an alternative intelligent search method is needed. In this study, we investigate hyper-heuristics for solving this problem.

Hyper-heuristics can be considered as a set of general search methods that can be applied to computationally hard problems [6]. They are search and optimisation methodologies to *select* or *generate* heuristics. This study focuses on a *selection hyper-heuristic* framework as illustrated in Figure 1. In this framework, a *perturbative* low level heuristic H is selected by one of the *heuristic selection* strategies, such as, simple (uniform) random selection, which is to applied to $S_{current}$ to create a new solution S_{new}. A perturbative heuristic accepts a complete solution, perturbs it, if necessary and returns a new solution. Whether the new solution S_{new} is accepted or rejected is determined by a *move acceptance criteria*, such as, simulated annealing. This process is repeated until some termination criteria is reached. A selection hyper-heuristic will be identified as <heuristic selection method> − <move acceptance criterion> from this point onward. Determining which low level heuristic to apply is done by the heuristic selection methods.

A selection hyper-heuristic approach has been chosen here as, at each state of the problem (the distance), an operation, low level heuristic, can be chosen

```
1 generate an initial candidate solution $S_{current}$
2 while(termination criteria not satisfied){
3   select a heuristic (or subset of heuristics) $H$ from $\{LLH_1, ..., LLH_n\}$
4   generate a new solution $S_{new}$ by applying $H$ to $S_{current}$
5   decide whether to accept or reject $S_{new}$
6   if($S_{new}$ is accepted) then
7     $S_{current} = S_{new}$
8 }
9 return $S_{current}$;
```

Fig. 1. A selection hyper-heuristic framework

which performs well. It allows the combination of hill climbers, to obtain a local optimum, and also mutational heuristics, to explore the search space. [13] and [3] discuss different selection hyper-heuristic frameworks for utilising hill climbers and mutational heuristics efficiently. In this study, a generic framework is used.

Choice function (CF) is a heuristic selection method. It uses a simple learning capability based on a scoring mechanism that evaluates the low level heuristics' most recent performance and the time that has passed since the last invocation. As time progresses, it chooses more relevant low-level heuristics to apply to the solution, increasing the likelihood of finding an optimal or good quality solution. One term of the choice function allows each low level heuristic another opportunity to be called (even if it has a poor recent performance) to see if an improved solution can be generated, thus allowing the opportunity to escape local minima and not punish, or discriminate against, the low-level heuristics for previously poor solutions. The heuristic with the maximum score is selected for invocation at each step. Most of the simple hyper-heuristic components are investigated in [6]. The authors describe hyper-heuristics that combine different heuristic selection methods, including *simple random* (SR) and choice function with two move acceptance methods; *accept all moves* (AAM) and *only improving moves accepted.* The choice function–accept all moves hyper-heuristic is reported to outperform the other methods in solving a scheduling problem. Choice function as a hyper-heuristic component has also been used in [14] and shown to yield good results. Another simple acceptance method is *accept improving and equal moves* (AIEM).

Reinforcement learning (RL) heuristic selection is classified as an online learning hyper-heuristic where learning takes place while the algorithm is solving an instance of the problem [15,16]. A utility value is assigned to each low level heuristic at each iteration. If the selected heuristic improves the current solution then its score is increased by some ratio. Equally if the heuristic decreases the evaluation (assuming a minimisation problem) its score is reduced by some, different, ratio. Low level heuristics are then chosen based upon the highest score for the current state of the problem.

There are numerous reinforcement learning approaches. This paper uses a QV-Learning approach which is an extension of Q-Learning and Sarsa where the state values are taken into consideration. QV-Learning, [17], uses Equation 1 to evaluate each heuristic at each state.

$$Q(s,a) = R(d,a) + \gamma Q(s,a) \tag{1}$$

Equation 1, defines two matrices, consisting of state, where state is the current solution, and actions which relate to the low level heuristics. Q is a matrix that holds an integer value for each heuristic, a, that specifies its quality at each stage of execution. R is another matrix holding rewards or punishments based on the quality of the solution generated by a. Where s is the current state of the solution, the current solution distance, and d is the reduction or increase made to the solution. γ is determined experimentally and typically $0 \leq \gamma \leq 1$. Numerous studies have been made using reinforcement learning with feasible solutions. Burke et al. [16] provides a number of advantages when using the online learning approach, where the learning takes place as the algorithm is solving the problem. Reinforcement learning has been used in a variety of different scheduling problems [18,19,20,21], each of which report to have found optimal solutions.

Bai and Kendall [22] used *simulated annealing* (SA) [23] as a hyper-heuristic move acceptance criteria which performed well for solving a shelf allocation problem. Bilgin et al. [5] reported the success of simulated annealing with a linear cooling schedule as a move acceptance criteria for hyper-heuristics when investigating examination timetabling benchmark instances. Simulated annealing accepts all improving moves, and the worsening moves are accepted with a probability given in Equation 2.

$$e^{-\frac{\Delta f}{\Delta F(1-t/M)}}, \tag{2}$$

where Δf is the fitness change, ΔF is the (estimate of) maximum fitness change, t is the current step and M is the maximum number of steps.

Great deluge (GD) algorithm is an optimisation heuristic proposed by Dueck [24]. This method uses a threshold that decreases in time at a given rate (e.g., linearly) representing an expected solution quality. Improving moves are accepted, while worsening moves may also be accepted if it is better than the threshold. This acceptance criteria is used as a hyper-heuristic component with simple random in [25] for solving a mobile telecommunication network problem. The same hyper-heuristic is reported to perform well over a set of benchmark functions in [5]. More on selection hyper-heuristic components can be found in [6,7,3,16].

3 Hyper-Heuristics for Sports Scheduling

The English Football League is made up of four leagues known, at the time of writing, as The Barclays Premier League, Coca-Cola Championship, Coca-Cola League One and Coca-Cola League Two. Each league consists of 24 teams except for the Barclays Premier League which has 20 teams. Therefore there will be a total of 46 fixtures on both Boxing Day and New Years Day (that is 92 teams each having to play). Each league can be treated as an individual search space with the exception of teams that are classified as paired teams, these are teams that are geographically close together and have limits on how many paired teams may play in the same location on the same day. Our goal is to generate a set of fixtures which are more efficient in terms of travelling distance (i.e. lower travelling distances) than the fixtures that are released by the Football Association each June/July whilst ensuring that all constraints are respected. This section describes the problem and defines the constraints that have been put in place by the Football Association, the constraints presented here are based on [11], which can be referred to for a more complete discussion.

C1. The first constraint, home and away, requires that if a team plays at home on Boxing Day, then it must play away on New Year's Day. Equally, if a team plays at home on New Year's Day, it must play away on Boxing Day.

C2. This constraint, playing twice, requires the same teams not to play each other on both New Year's Day and Boxing Day.

C3. The third constraint, known as paired teams, requires paired teams not to play each other over the holiday period. Paired teams are teams that are, typically, geographically close to each other. This constraint has been put in place by the Football Association due to the policing requirements. We treat this as a hard constraint, however the Football Association do occasionally violate it.

C4. A pair clash occurs when two or more paired teams play at home on the same day. This constraint, pair clashes, restricts the total number of paired teams playing at home, which cannot exceed the limits specified in the Football Associations fixtures during the holidays.

C5. This constraint, London and Manchester based, restricts the number of London-based clubs that can play at home which must not exceed the limits used by the Football Association during the holidays. Similarly, there is a limit on the London based Premier League teams and Greater Manchester clubs that can play at home on the same day.

The main objective is to minimise the total travelling distance that each team is required to undertake during the holiday period; see Equation 3.

$$\min\{\sum_{x=0}^{n}\sum_{y=0}^{n} D_{x,y}X_{x,y}\} \tag{3}$$

where $D_{x,y}$ is the distance, in miles, between team x and team y and $X_{x,y}$ is 1 if team x is playing team y or 0 otherwise.

The selection hyper-heuristic framework shown in Figure 1 is used during the experiments. The performance of twelve hyper-heuristics that combine the following heuristic selection and move acceptance criteria combinations are investigated: {simple random, choice function and reinforcement learning} versus {simulated annealing, great deluge, all moves accepted, accept improving and equal moves}. Six low level heuristics are implemented which do not allow the violation of any of the five hard constraints, explained in the problem definition, throughout execution with the exception of when the initial Boxing Day or New Years Day fixtures are generated. Each time a new solution is generated a check is made using another heuristic to ensure that the swaps made do not violate constraints, if it does the move is not made. In Figure 1, low level heuristics are selected using heuristic selection at line 3 before applying the heuristic at line 4. The low level heuristics are described as follows.

LLH_1: A hill climbing heuristic that aims to improve the distance for Boxing Day. LLH_1 selects a random league and two random teams and performs a swap, if the solution yields a worse fitness value, the original solution is returned. The swap is only made for home teams.

LLH_2: A mutational heuristic that allows the solution to move away from local optima. A random league and two random teams are selected and swapped. Providing there is no violation of constraints the move is accepted regardless of its fitness value.

LLH_3: A mutational heuristic which randomly selects a league and a fixture. The home and away teams of the chosen fixture are swapped having only a minor effect on the distance but allowing a new range of home or away swaps to occur. The main purpose of LLH_3 is to conform to C4, specified in the problem definition.

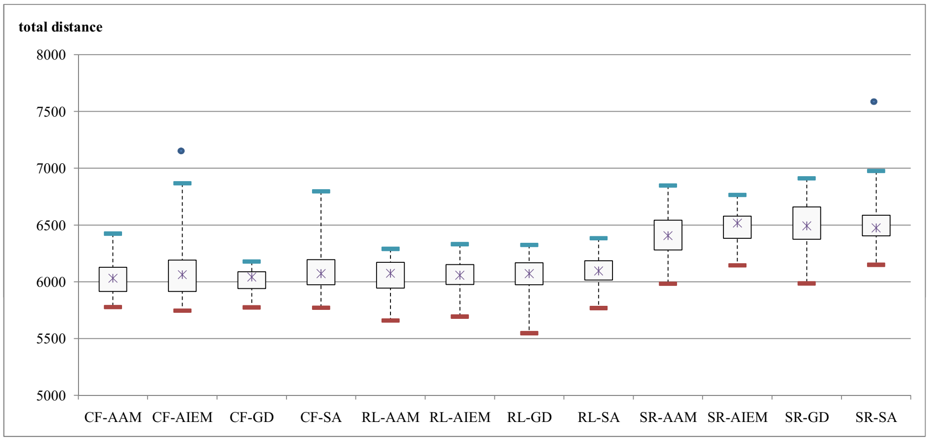
LLH_4: A hill climbing heuristic that selects a random league and two random teams and performs a swap. It uses delta evaluation, evaluating only the changed items in the solution, to evaluate the swaps and if either of them have a distance greater than 120miles (we use 120miles as this represents about 2 hours travelling time) the original solution is returned.

LLH_5: A next gradient hill climbing heuristic. A random league is selected and each home team is swapped with the one below, when ordered as a list of fixtures. Each time a swap is made, if the fitness value is improved the move is accepted. This happens for the entire league each time the heuristic is used.

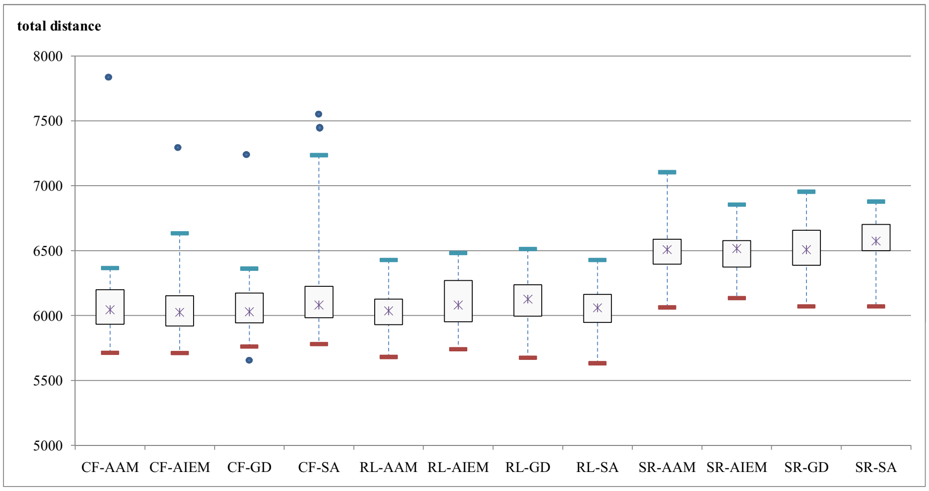
LLH_6: A hill climbing heuristic that relates to LLH_1. A random league and two random teams are selected and swapped. If the move results in a worse move the original solution is returned. The only difference between LLH_1 and LLH_6 is that LLH_6 also swaps away teams. This has been added as a separate heuristic as the performance of the two heuristics does differ.

4 Experimental Results

Two datasets are used during the experiments, the 2009-2010 season fixture set along with the 2005-2006 season from the top four divisions in England; Premier League, Coca Cola Championship and Division One and Two. The 2005-2006 season was selected to be able to compare our results to the previously proposed approaches. Season 2009-2010 was used so that the results generated here could be compared to the latest season. Distances were collected based on the



(a) 2005-2006



(b) 2009-2020

Fig. 2. Box plot of total distances found in all runs by each hyper-heuristic (displaying maximum, upper quartile, median, lower quartile and mininimum) for (a) 2005-2006 and (b) 2009-2010 problems.

postcodes of the football stadiums from greenflag (http://www.greenflag.co.uk - last accessed 4th April 2010).

An Intel Core 2 Duo, 2GHz laptop with 2.00GB memory was used to conduct the experiments. All of the selection and acceptance method combinations were executed fifty times, each of which generated a random initial fixture list; there were no pre-defined fixture lists. They were allowed to run up to the maximum number of 100,000 iterations or until 6,000 non-improving moves were made.

Table 1. The comparison of the total distances for both days against the Football Association (FA), local search approach [11], CPLEX and simulated annealing approach [12], and the best performing hyper-heuristics.

Instance	FA	Kendall [11]	Kendall [12]	Hyper-heuristics
2005-2006	10631	6917	6020	5547 (RL−GD)
2009-2010	8621	-	-	5633 (RL−SA)

Figure 2 summarises the results obtained by using different hyper-heuristics for the problem. The best results are obtained using reinforcement learning−great deluge and reinforcement learning−simulated annealing for 2005-2006 and 2009-2010 fixtures, respectively. Hyper-heuristics produce improvements over the published fixtures shown in Table 1. Indeed both approaches by Kendall in [11,12] improve the distances generated by the Football Association. However, we do not need to explicitly compare against these results due to the differences in collecting the distance data. A rough comparison does show that the hyper-heuristic approach is superior. When the average performance of hyper-heuristics are compared, choice function−great deluge and reinforcement learning−accept all moves are better than the others for 2005-2006 and 2009-2010 fixtures, respectively. Wilcoxon test shows that both of these hyper-heuristics perform significantly better than the simple random heuristic selection based hyper-heuristics within a confidence interval of 95%. Almost the rest of the learning hyper-heuristics deliver a similar performance.

Simple random requires a much shorter computational time (4 seconds on average) than choice function (12 seconds on average) and reinforcement learning (52 seconds on average) for solving a given problem instance. However, the learning approaches, choice function and reinforcement learning, do produce improved results. It can be observed that, on average, as the computation time increases, typically with the time spent for learning, the results improve. The results illustrate that there is a trade off between time and quality of solutions. Simple random as a heuristic selection runs quicker than any of the other heuristics yet produces the worst results. Choice function improves the results of simple random, by approximately 5%, yet increases the run time by approximately three times as much, on average. Finally, reinforcement learning as a heuristic selection method provides the highest ranked results for each of the criteria. It dramatically improves the solutions generated by simple random but the computation time is much higher.

5 Conclusion

Each combination of heuristic selection and move acceptance within a selection hyper-heuristic framework improved the results and provided a good quality solution to the sports scheduling problem in reasonable time whilst conforming to the constraints specified. While applying the 2005-2006 dataset, the reinforcement learning−great deluge hyper-heuristic made savings to the Boxing Day and New Years Day fixtures of 44.19% and 50.28%, respectively, whilst even comparing results from the current season, dataset 2009-2010, it was possible to improve the Boxing Day fixtures by 41.42% and New Years Day fixtures by 28.29% using the reinforcement learning−simulated annealing hyper-heuristic.

A trade off must be made between computation time and quality of solutions. For each of the datasets, reinforcement learning found the best solutions but had a runtime of at least two times than that of choice function and over ten times the amount compared to simple random. Reinforcement learning produced the most consistent set of solutions on average where each move acceptance criterion generated similar results and, with the exception of great deluge, all executed in similar times. Therefore we conclude, that reinforcement learning is the best heuristic selection component to be used within hyper-heuristics for solving this sports scheduling problem.

References

1. Denzinger, J., Fuchs, M., Fuchs, M.: High performance atp systems by combining several ai methods. In: Proceedings of the 4th Asia-Pacific Conference on SEAL, IJCAI, pp. 102–107 (1997)
2. Özcan, E., Bykov, Y., Birben, M., Burke, E.K.: Timetabling using late acceptance hyper-heuristics. In: Proc. of the IEEE Congress on Evolutionary Computation, pp. 997–1004. IEEE Press, Los Alamitos (2009)
3. Özcan, E., Bilgin, B., Korkmaz, E.: A comprehensive analysis of hyper-heuristics. In: Intelligent Data Analysis, pp. 3–23 (2008)
4. Özcan, E., Mısır, M., Ochoa, G., Burke, E.K.: A reinforcement learning - great-deluge hyper-heuristic for examination timetabling. International Journal of Applied Metaheuristic Computing 1(1), 39–59 (2010)
5. Bilgin, B., Özcan, E., Korkmaz, E.E.: An experimental study on hyper-heuristics and exam timetabling. In: Proceedings of the 6th Practice and Theory of Automated Timetabling (PATAT 2006). LNCS, vol. 3867, pp. 394–412. Springer, Heidelberg (2006)
6. Cowling, P., Kendall, G., Soubeiga, E.: A hyperheuristic approach to scheduling a sales summit. In: Burke, E., Erben, W. (eds.) PATAT 2000. LNCS, vol. 2079, pp. 176–190. Springer, Heidelberg (2001)
7. Burke, E.K., Hart, E., Kendall, G., Newall, J., Ross, P., Schulenburg, S.: Hyper-heuristics: An emerging direction in modern search technology. In: Glover, F., Kochenberger, G. (eds.) Handbook of Metaheuristics, pp. 457–474. Kluwer, Dordrecht (2003)
8. Ross, P.: Hyper-heuristics. In: Burke, E.K., Kendall, G. (eds.) Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques, pp. 529–556. Springer, Heidelberg (2005)
9. Kendall, G., Knust, S., Ribeiro, C., Urrutia, S.: Scheduling in sports: An annotated bibliography. Computers & Operations Research 37, 1–19 (2010)
10. Applegate, D.L., Bixby, R.E., Chvatal, V., Cook, W.J.: The Traveling Salesman Problem: A Computational Study. Princeton Series in Applied Mathematics. Princeton University Press, Princeton (2007)

11. Kendall, G.: Scheduling english football fixtures over holiday periods. Journal of the Operational Research Society 59(6), 743–755 (2008)
12. Kendall, G.: Hybridising cplex with simulated annealing to minimise travel distances for english football fixtures (2009) (in review)
13. Özcan, E., Bilgin, B., Korkmaz, E.E.: Hill climbers and mutational heuristics in hyperheuristics. In: Runarsson, T.P., Beyer, H.-G., Burke, E.K., Merelo-Guervós, J.J., Whitley, L.D., Yao, X. (eds.) PPSN 2006. LNCS, vol. 4193, pp. 202–211. Springer, Heidelberg (2006)
14. Kendall, G., Cowling, P., Soubeiga, E.: Choice function and random hyperheuristics. In: Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution And Learning, SEAL, pp. 667–671 (2002)
15. Nareyek, A.: Choosing search heuristics by non-stationary reinforcement learning. In: Resende, M.G.C., de Sousa, J.P. (eds.) Metaheuristics: Computer Decision-Making, pp. 523–544. Kluwer, Dordrecht (2003)
16. Burke, E., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., Woodward, J.: A classification of hyper-heuristic approaches. In: Handbook of Metaheuristics. Springer, Heidelberg (to appear, 2010)
17. Wiering, M.: Qv(lambda)-learning: A new on-policy reinforcement learning algorithm. In: Proceedings of the 7th European Workshop on Reinforcement Learning (2005)
18. Aydin, M., Öztemel, E.: Dynamic job-shop scheduling using reinforcement learning agents. In: Robotics and Autonomous Systems, vol. 33, pp. 39–59. Elsevier, Amsterdam (2000)
19. Luiz, A., Ribeiro, C., Costa, A., Bianchi, R.: Heuristic reinforcement learning applied to robocup simulation agents. In: Visser, U., Ribeiro, F., Ohashi, T., Dellaert, F. (eds.) RoboCup 2007: Robot Soccer World Cup XI. LNCS (LNAI), vol. 5001, pp. 220–227. Springer, Heidelberg (2008)
20. Wang, Y., Usher, J.: Application of reinforcement learning for agent-based production scheduling. In: Engineering Applications of Artificial Intelligence, vol. 18, pp. 73–82 (2005)
21. Zhang, W., Dietterich, T.: A reinforcement learning approach to job-shop scheduling. In: Proceedings of the 14th international joint conference on Artificial intelligence, vol. 1, pp. 1114–1120 (1995)
22. Bai, R., Kendall, G.: An investigation of automated planograms using a simulated annealing based hyper-heuristics. In: Ibaraki, T., Nonobe, K., Yagiura, M. (eds.) Metaheuristics: Progress as Real Problem Solver. Operations Research/Computer Science Interface Serices, vol. 32, pp. 87–108. Springer, Heidelberg (2005)
23. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science 220, 671–680 (1983)
24. Dueck, G.: New optimization heuristics: The great deluge algorithm and the record-to record travel. Journal of Computational Physics 104, 86–92 (1993)
25. Kendall, G., Mohamad, M.: Channel assignment optimisation using a hyperheuristic. In: Proceedings of the 2004 IEEE Conference on Cybernetic and Intelligent Systems (CIS 2004), Singapore, December 1-3, pp. 790–795 (2004)