# Memetic Algorithms and Hyperhill-climbers

Ersan Ersoy
İstanbul Technical University, Informatics Institute, İstanbul, Turkey, eersoy@be.itu.edu.tr
Ender Özcan
Yeditepe University, Computer Engineering Department, İstanbul, Turkey, eozcan@cse.yeditepe.edu.tr
A. Şima Uyar
İstanbul Technical University, Computer Engineering Department, İstanbul, Turkey, etaner@itu.edu.tr

Memetic algorithms (MAs) are meta-heuristics that join genetic algorithms with hill climbing. MAs have recognized success in solving difficult search and optimization problems. Hyperheuristics are proposed as an alternative to meta-heuristics. A hyperheuristic is a mechanism that chooses a heuristic from a set of heuristics, applies it to a candidate solution, and then makes a decision for accepting or rejecting the new solution. In a traditional MA, a single hill climbing method is utilized during the search process. In the presence of multiple hill climbers, the hyperheuristic mechanisms can be adapted for the MAs and employed to exploit the strength of each hill climber better without changing the framework of the MAs. In this study, a set of such mechanisms referred to as *hyperhill-climbers* is investigated for solving exam timetabling problems.

*Keywords: Memetic Algorithms, Hill Climbing, Hyperheuristics, Timetabling*

## 1. Introduction

Practical timetabling problems are NP complete constraint optimization problems [16]. A set of events and resources are scheduled subject to a set of constraints. Due to the immense search space and the constraints, traditional approaches might fail in obtaining an optimal solution for a given problem. Numerous approaches, including meta-heuristics and hyperheuristics are proposed for timetabling problems [9]. Ozcan [27] covers different constraint types in timetabling problems. Course timetabling, exam timetabling, sports timetabling and nurse rostering are the most commonly studied problem types in the literature. This study focuses on exam timetabling problems.

Heuristics are used to make a move from a given point in a landscape to another point during the search. Heuristics can be classified as *mutational heuristics* and *hill climbers*. A hill climber is a local search technique that aims to produce a candidate solution with an improved quality whenever applied, whereas mutational heuristics do not have such a purpose. Cowling, et al. [13] describe hyperheuristics as methods that are used to select a low level heuristic from a set of heuristics during the search. In a *simple* hyperheuristic, a single candidate solution representing a problem at hand is continuously processed in an iterative cycle until some termination criteria are satisfied. At each step, a heuristic is selected based on some problem independent criteria, such as the fitness change. Then it is applied to the candidate solution producing a new one. This new solution is accepted or rejected based on a strategy within the hyperheuristic.

Burke et al. [6] provide a hyperheuristic framework that allows the use of any type of heuristics together. Ozcan et al. [29] propose different hyperheuristic frameworks for utilizing hill climbers together with mutational heuristics. One of the proposed frameworks, in which a hill climber is invoked after the application of a selected mutational heuristic, yields an improved performance. Bilgin et al. [4] analyze the performance of thirty five simple hyperheuristics on a set of benchmark functions and a set of exam timetabling benchmark problems by pairing up five *heuristic selection* and seven *move acceptance* strategies.

Genetic Algorithms (GAs), proposed by Holland [20], are inspired from the Darwinian theory of evolution and population genetics. Memetic Algorithms (MAs) [25] are hybrid approaches that use local search (hill climbing) within GAs. A *meme* denotes a hill climbing method. There is strong empirical evidence that the use of a meme improves the performance of a GA [28]. MAs are already used in solving many hard problems, including timetabling problems. Burke et al. discusses the MAs for timetabling in [8]. In [1], [26] and [30], a successful MA is described for different type of timetabling problems. Each MA uses a problem tailored, violation directed mechanism, named as VDHC that manages a set of constraint based hill climbers. This study is an extension to these previous studies. A set of MAs is investigated using multiple hill climbers for solving a set of benchmark exam timetabling problem instances. The MAs include four different VDHC instances and a set of hyperheuristics which will be referred to as hyperhill-climbers used for dynamically determining the suitable hill climber. Moreover, the problem formulation in [12] is used for exam timetabling which is a different formulation than the one used in [30].

## 2. Preliminaries

Memetic Algorithms (MAs) use local search techniques to improve the exploitation capability of GAs [2],[25]. In a traditional GA, candidate solutions are encoded as *chromosomes* which form the *individuals*. Each individual is made up of *genes*, where each gene receives an *allele* from a set of predetermined values. For example, in a binary encoding an individual is a binary string, where {0,1} is the allele set. In the initial *generation*, a *population* of random individuals is generated. A *fitness function* is used to measure the quality of an individual. In an evolutionary cycle, all individuals go through a set of genetic operations, i.e. *selection*, *crossover* and *mutation*. Depending on the fitness of all the individuals in the population, two of them, termed as *mates*, are randomly selected through a mechanism which favors individuals with better fitness values. *Tournament selection* strategy randomly compares the fitness of a *tournament size* number of individuals and selects the one having the best quality as one of the mates. A *crossover* mechanism is applied to the selected mates, generating new candidate solutions called *offspring*. *One point crossover* randomly determines a crossover location and exchanges the parts of the mates to one side of this point forming two new individuals. The traditional *mutation* mechanism changes each allele to another value from the allele set, usually with a probability of *1/chromosome_length* for each gene in each offspring. Finally, the current population is replaced using individuals from the current generation and from the offspring pool. A *weak elitist* strategy for a *trans-generational* MA (or GA) keeps the best two individuals from the current generation and the rest of the population is filled in from the offspring pool. Evolution terminates whenever some criteria are satisfied.

In a traditional MA, a hill climbing method is applied to each offspring following the mutation step. In this manner, a pool of improved offspring is formed. Notice that in the existence of a set of hill climbers, a mechanism can be used to select one from this set during the improvement stage without changing the original MA framework. For example, Krasnogor formalizes a co-evolutionary framework in [23] as multimeme memetic algorithms and describes a self-adaptive mechanism based on a Lamarckian learning approach. Another possibility is making use of hyperheuristics for deciding which hill climber to apply whenever necessary. In this study, such mechanisms are investigated. Existing hyperheuristics are adapted to select the best hill climber and the best hill climber orderings within an MA. Cowling et al. discuss most of the simple hyperheuristics in [13]. *Simple Random* (SR) heuristic selection mechanism randomly chooses one of the low level heuristics. *Random Permutation Gradient* (RPG) generates a random permutation of the low level heuristics at first. Then, RPG applies the low level heuristics in turn repeatedly without changing the order of heuristics as long as an improved result is produced. The *Greedy* (GR) method allows all heuristics to process a given candidate solution and chooses the one that generates the most improved solution. *Choice Function* (CF) [22] makes a selection based on the performance history of each low level heuristic and the successively applied pair of low level heuristics. The performance is evaluated using a problem independent measure such as the degree of the previous improvement and the execution time. Gaw et al. [19] study the CF based hyper-heuristics in a different context. Bilgin et al. [4] and Ayob et al. [3] utilize *Improving and Equal* (IE) move acceptance mechanism which rejects only worsening moves. Kendall et al. [21] test the *Great Deluge* (GD) acceptance criterion combined with the SR heuristic selection method as a hyper-heuristic on a set of channel assignment problems. In GD, all moves generating a better or equal objective value than a level computed at each step during the search are accepted. The initial level is set to the objective value of the initial candidate solution. At each step, the level is updated at a linear rate towards the expected objective value.

Except the CF, all strategies can be directly used as a hyperhill-climber within an MA to select the best hill climber whenever required. Each individual is allowed to carry a performance measure along with the genetic material as in multimeme memetic algorithms [23]. Updates are performed in the same way as in [22], [14]. Still, a mechanism is needed for transmitting the measure values of the mates to the offspring. In this study, the simplest strategy is chosen and a randomly selected measure is transmitted to each one. GD requires an initial level to be defined. Traditionally, this level is the fitness of the initial candidate solution in a simple hyperheuristic framework. But, MAs are population based techniques, hence the average fitness of the initial population is used as the initial level in the GD.

## 3. Memetic Algorithms for Exam Timetabling

Exam timetabling problems (ETPs) require a search for an optimal arrangement of resources for a set of exams based on a set of constraints. The constraints can be divided into two groups; *hard* and *soft* constraints. In a *feasible* solution, all hard constraint must be satisfied. For instance, a student cannot attend two different exams at the same time. The soft constraints are preferences that increase the quality of a solution. For example, increasing the free time between consecutive exams of a student can be considered as a soft constraint. More on exam timetabling can be found in [5], [9], [12], [32]. In this study, the formulation of the exam timetabling problem provided by Carter et al. [12] is used. An optimal schedule for a set of exams is searched for balancing the load of the students subject to the constraint that a student must attend no more than one exam at any time slot (period). An assignment in an ETP is an ordered pair $(x,y)$, where $x \in A$ (set of exams), $y \in B$ (set of periods). The interpretation of this assignment in terms of ETP is: "Exam $x$ starts at time $y$". Given $E$ exams and a timetable of $P$ periods, the search space size is $E^P$, hence non-traditional approaches are preferred for solving exam timetabling. The cost function for evaluating the quality of a given solution $x$ is formulated as in Equation 1.

$$f(x)=\frac{1}{S}\sum_{i=1}^{E-1}\sum_{j=i+1}^{E} c_{ij} w_t \text{ , where } w_t = \begin{cases} S*10^6 & t=0 \\ 2^{5-t} & t=1,2,3,4,5 \\ 0 & t>0 \end{cases} \quad (1)$$

$E$ is the number of exams, $S$ is the number of students, $c_{ij}$ is the number of students taking the exams $i$ and $j$; $i, j \in (1, \ldots, E)$, $w_t$ is the violation weight for $t$ free time slots between exams $i$ and $j$.

In this study, a traditional MA framework is used as described in Section 2. An allele value indicates the period assigned for the corresponding exam. The chromosome length is equal to the number of exams for a given problem instance. Two different initial population generation methods are implemented. The first method, named as *Largest Degree First* (LDF), schedules the exams in descending order of the number of conflicts. The exam that causes the largest number of conflicts with the other exams is scheduled first. During this process, one of the *available* periods is randomly assigned to the exam. An available period is the one that does not cause an overlap with the previously scheduled exams. If there is no such period, then the exam is randomly scheduled. The second method referred to as *Largest Weighted Degree First* (LWD), schedules the exams in a similar manner. However, instead of using the raw the number of conflicts, each exam is weighted by the total number of students involved in the conflicts for that exam. The assignment process is the same as LDF. Both approaches are explained further in [9].

Considering the decomposable cost function in Equation 1, three constraint types can be identified: C1, C2 and C3. C1 is a hard constraint, covering the violations due to conflicting exams which are not supposed to overlap. C2 and C3 represent soft constraints. C2 imposes a condition such that at least three free periods should be between the exams a student should take. Similarly, C3 imposes a condition such that at least six free periods should be between the exams a student should enter. Based on this perspective, three simple hill climbers are implemented to be used within the MA: HC1, HC2 and HC3. Each hill climber attempts to fix the violations due to the corresponding constraint type. Hence, a gradual improvement of individuals is emphasized.

As a data structure, a list for each constraint type is maintained to keep track of the exam pairs which generate a related violation in each individual. A hill climber goes through the relevant list of all exam pairs one at a time. An exam pair from this list is chosen randomly. Then the randomly selected exam is rescheduled to one of the available periods. If there is no such period, then the other exam goes under the same operation. The lists are updated. All hill climbers operate in the same way. The MA requires a mechanism to select the best hill climber after the mutation step. All mechanisms are organized in three groups. Since, there is a small number of hill climbers, all of them can be applied to an individual in some predefined order successively. The first group of mechanisms contains six permutations of three hill climbers for evaluating the significance of the heuristic orderings. The MAs using such a mechanism is denoted by MA_123, MA_132, MA_213, MA_231, MA_312, and MA_321 where the order of the numbers {1,2,3} indicates the application order of the related hill climber, e.g., MA_123 applies HC1, HC2 and HC3 consecutively.

The second group of hyperhill-climbers arranges the order of the hill climbers based on the number of violations due to each constraint type. The *violation ordering hyperhill-climber* (VIOO) uses the raw number of violations for ordering the hill climbers. The violations are counted for each constraint type and they are sorted from the one that causes the highest number of violations towards the one that causes a lower number of violations. Then the corresponding hill climbers are applied in that order. The *cost ordering hyperhill-climber* (CSTO) uses a similar idea. The weighted cost for each constraint type is utilized instead of the raw violations for sorting the hill climbers. The MAs using these mechanisms are referred to as MA_VIOO and MA_CSTO. Their performances are compared to a MA using the RPG hyperhill-climber (MA_RPG) that determines a random order for HC1, HC2 and HC3 and applies it to an individual. Ozcan describes heuristic templates for timetabling in [27] and [28]. Ozcan et al. use an instance of a heuristic template in [30]. A modified version, denoted as VIOD and CSTD are utilized in this study. Note that this formulation of the exam timetabling problem is different than the previous formulation used in [30]. VIOD and CSTD are violation directed heuristics that organize multiple hill climbers where each one improves a corresponding constraint type in a given problem based on the heuristic template as presented in Figure 1. Three iterations, each of which can be considered as a single stage, are performed in the second while loop 2.a. in Figure 1. In the first stage, the area of concern is marked as all exams. In the second stage, a subset of exams is randomly chosen from the individual. In the last stage, the area of concern is lowered further to a random pair of exams. At each stage a selected hill climber is applied through evaluating the violations due to each constraint type. The higher the number of violations is, the higher the chance that the corresponding hill climber will be invoked. VIOD uses the raw number of violations during the evaluation for selecting a hill climber, while CSTD uses the weighted cost for that purpose. MA_VIOD and MA_CSTD utilize these heuristics in an MA framework as the third group of hyperhill-climbers.

The last group of mechanisms consists of the hyperhill-climbers as described in Section 2. Two heuristic selection methods (CF, SR) and two acceptance criteria (IE, GD) are paired up. The MAs utilizing the hyperhill-climbers are labeled as MA_CF_IE, MA_SR_IE, MA_SR_GD and MA_CF_GD. The same settings in [22], [14] for CF are used. For a given problem, there might be many hill climbers in which case GR becomes impractical. A modified version, referred to as mGR is used as a hyperhill-climber. In the MA_mGR, two memes are randomly chosen during the hill climbing stage. They are executed separately and the most improved individual is accepted.

In [17], parameter control techniques are classified based on: what is changed (operator probabilities, hill climbing method etc.), how the change is made (deterministic, adaptive, self-adaptive), the scope of the change (population level, individual level, etc.) and the evidence upon which change occurs (monitoring performance of operators, population diversity, etc.). In the deterministic method of changing the parameters, there is a deterministic rule which is used to modify the parameters without using any feedback from the search. In the adaptive mechanisms, the feedback taken from the ongoing search guides the change in the parameters. In self-adaptation, the parameters are coded into the chromosomes and are allowed to evolve along with the individuals. The hyperhill-climbers used within the MAs during this study can be classified based on this terminology. The first group of hyperhill-climbers is deterministic, while the second group of mechanisms is adaptive mechanisms, except the deterministic one RPG. The other hyperhill-climbers in this group adaptively select an appropriate hill climber by dividing a candidate solution into three subparts based on a decomposable penalty oriented fitness function. Hence, a component level adaptation is employed. In the third group, SR_GD is an adaptive mechanism, operating at individual level, while CF_IE and CF_GD are self-adaptive mechanisms, operating at population level. On the other hand, mGR and SR_IE are deterministic hyperhill-climbers.

```
1. mark the area of concern as all events
2. **while** (some termination criteria are not satisfied) **do**
      a. **while** (there is improvement and some termination criteria$_2$ are not satisfied) **do**
            i. select a constraint type evaluating each constraint type violations for the
               marked events
            ii. apply hill climbing for the selected constraint type to all events within the
               area of concern
      b. **end while**
      c. lower the area of concern and mark the related events
3. **end while**
```

Figure 1. Pseudo-code of a heuristic template for timetabling

## 4. Experimental Results

Six problems from Carter's benchmark [12] are used. The characteristics of each problem instance are presented in Table 1. In the MAs, the population size is fixed as 200, crossover is always applied to selected mates. For each experiment 20 runs are performed. A run is terminated whenever the maximum number of generations is exceeded, namely 10,000. All the results are validated via the tool provided at http://www.cs.nott.ac.uk/~rxq/data.htm. As an evaluation criterion, the best fitness value achieved during the runs is used. Additionally, a ranking considering the ties is performed among the compared approaches based on this offline performance criterion.

Table 1. The characteristics of Carter's benchmarks used during the experiments

| *Test Case* | *No.of Exams* | *No. of Students* | *Enrollments* | *Density of Conflict* | *No. of Periods* |
|---|---|---|---|---|---|
| Hec-s-92 | 81 | 2823 | 10632 | 42.0% | 18 |
| Kfu-s-93 | 461 | 5349 | 25113 | 5.6% | 20 |
| Lse-f-91 | 381 | 2726 | 10918 | 6.3% | 18 |
| Sta-f-83 | 139 | 611 | 5751 | 14.4% | 13 |
| Ute-s-92 | 184 | 2750 | 11793 | 8.5% | 10 |
| Yor-f-83 | 181 | 941 | 6034 | 28.9% | 21 |

During the first part of the experiments, the initialization methods and the predefined hill climber orderings are tested. Each MA using a different hill climber ordering is compared with both initialization methods. The performance of twelve MAs is ranked from 1 to 12, from the best towards the worst for each benchmark data. The average rank of each method over the benchmarks using a different initialization scheme is provided in Figure 2 in which the x-axis denotes the different methods and the y-axis shows the average ranks. The MAs using LWD for initialization are illustrated as textured bars. MA_213 using LWD as the initial population generation scheme yields the best performance. It is also observed that using HC3 as the first hill climber generates poor results. LWD is used for initial population generation during the further experiments, since it has a slightly better performance than LDF.

In the second set of experiments, MA_RPG and other mechanisms that rely on the violations due to each constraint type for managing the hill climbers are tested. Experiments are divided into two subparts. In the first part of the experiments, the order of the hill climbers to be applied changes in time. As a hill climber ordering strategy, MA_CSTO provides the best performance on average as illustrated in Figure 3(a). MA_VIOO does not seem to be a viable strategy, since it has a slightly poorer performance than MA_RPG. Narrowing down the area of concern as the hill climbers are applied seems to be an effective strategy. MA_VIOD delivers a better performance as compared to MA_CSTD as shown in Figure 3(a).

The results obtained from the last set of experiments are summarized in Figure 3(b). MA_SR_IE and MA_CH_GD provide the best performance. MA_mGR has the worst performance, although it visits more states as compared to the rest of the MAs in this group of experiments. The approach seems to get stuck at a local optimum due to the intensive use of hill climbers in MA_mGR.

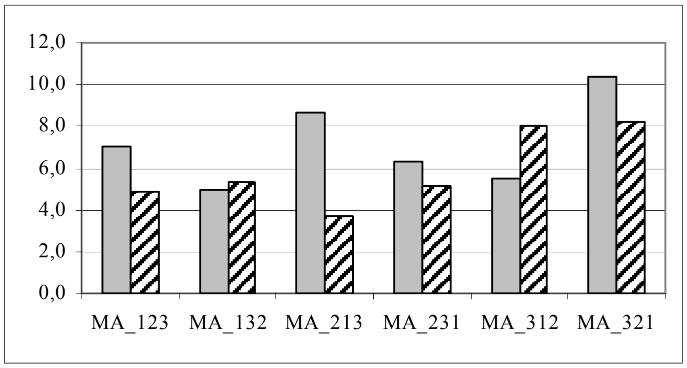
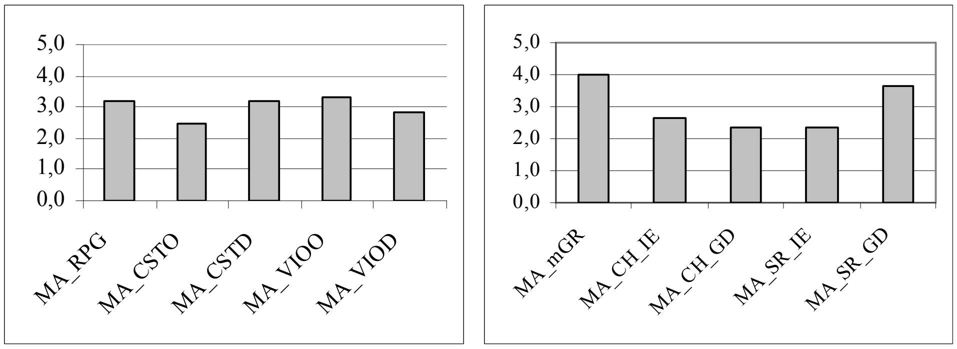


Figure 2. The average rank of each MA using a different hill climber ordering, where the textured bars indicate the MA using LWD and the other using LDF for initialization



(a) (b)

Figure 3. The average rank of each MA with a different hyperhill-climber for (a) the second and (b) third group of experiments.

The best MAs from the experiments and some previous approaches from the literature are compared. The approaches are divided into three groups: deterministic heuristics, stochastic methods, and the best MAs obtained during this study. Due to the parameter variances among the stochastic approaches and different termination criteria, the comparison between the algorithms can be considered as an indirect comparison as presented in Table 2. The hyperhill-climbers within the MAs provide a promising performance. MA_SR_IE turns out to be the best among the MAs, even though due to the termination criteria the maximum number of states it visits is fewer as compared to MA_312, MA_CSTO and MA_VIOD. MA_SR_IE and MA_CH_GD perform better than the LD, SD, LWD and LE heuristics.

Table 2. The performance comparison of the MAs with previously used approaches based on the best fitness values.

| *Approaches,* [*Source*] | *Hec-s-92* | *Kfu-s-93* | *Lse-f-91* | *Sta-f-83* | *Ute-s-92* | *Yor-f-83* | *Avr. ranks* |
|---|---|---|---|---|---|---|---|
| **LD, [12]** | 10.8 | 14.0 | 12.0 | 162.9 | 38.3 | 49.9 | 10,3 |
| **SD, [12]** | 12.7 | 15.9 | 12.9 | 165.7 | 31.5 | 44.8 | 12,6 |
| **LWD, [12]** | 15.8 | 22.1 | 13.1 | 161.5 | 26.7 | 41.7 | 12,5 |
| **LE, [12]** | 15.9 | 20.8 | 10.5 | 161.5 | 25.8 | 45.1 | 11,6 |

|  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|
| **Wal, [33]** | 12.9 | 17.1 | 14.7 | 158.0 | 29.0 | 42.3 | 13,2 |
| **GS, [15]** | 12.4 | 18.0 | 15.5 | 161.0 | 29.9 | 41.0 | 13,5 |
| **Cal, [10]** | 9.2 | 13.8 | 9.6 | 158.2 | 24.4 | 36.2 | 2,6 |
| **BN, [7]** | 11.3 | 13.7 | 10.6 | 168.3 | 25.5 | 36.8 | 5,9 |
| **Mal, [24]** | 10.6 | 13.5 | 10.5 | 157.3 | 25.1 | 37.4 | 2,2 |
| **PS, [31]** | 10.8 | 16.5 | 13.2 | 158.1 | 27.8 | 38.9 | 8,6 |
| **CT, [11]** | 10.8 | 14.1 | 14.7 | 134.9 | 25.4 | 37.5 | 5,4 |
| **MMAS, [18]** | 11.3 | 15.0 | 12.1 | 157.2 | 27.7 | 39.6 | 6,4 |
| **MA_312** | 11.7 | 16.0 | 14.0 | 157.8 | 26.3 | 41.8 | 9,5 |
| **MA_CSTO** | 11.7 | 16.1 | 13.5 | 158.3 | 27.2 | 41.3 | 10,8 |
| **MA_VIOD** | 11.6 | 16.5 | 13.2 | 158.4 | 26.7 | 41.5 | 10,3 |
| **MA_CH_GD** | 11.8 | 16.1 | 13.4 | 157.7 | 26.3 | 40.9 | 8,8 |
| **MA_SR_IE** | 11.7 | 15.8 | 13.3 | 157.9 | 26.7 | 40.7 | 8,1 |

## 5. Conclusions

Hyperheuristics are becoming a central research area beside metaheuristics in search and optimization. This study shows that they can be used as a support mechanism for managing multiple hill climbers within metaheuristics. Memetic algorithms (MAs) as metaheuristics are successfully utilized for solving many difficult problems. In the case of multiple hill climbers, hyperheuristics can be embedded into the MAs as hyperhill-climbers for selecting the best hill climber to apply or deciding the best ordering for successive application of hill climbers. In this study, a set of deterministic, adaptive and self-adaptive hyperhill-climbers are investigated on a benchmark of exam timetabling problem instances, each requiring a satisfactory schedule subject to some constraints. Three hill climbers, each aiming to reduce the violations due to a specific constraint type are utilized. The self-adaptive hyperhill-climbers perform better as compared to the adaptive ones. The CH_GD as a self-adaptive mechanism delivers a good performance. Ordering the hill climbers with respect to the weighted violations for consecutive application seems to be a better choice than ordering them with respect to the raw number of violations. Yet, a deterministic hyperhill-climber, namely, SR_IE that selects a single hill climber at a time turns out to be the best one among all. The experimental results show that the hyperhill-climbers are viable strategies to manage a set of low level hill climbers within an MA. The affect of the number of hill climbers that a hyperhill-climber manages will be investigated further as a future work.

## References

[1] Alkan, A., Ozcan, E. (2003), Memetic Algorithms for Timetabling. Proc. of IEEE Congress on Evolutionary Computation , 1796 – 1802.

[2] Areibi S:, Moussa M., Abdullah, H. (2001), A Comparison of Genetic/Memetic Algorithms and Other Heuristic Search Techniques, Int. Conf. on Artificial Intelligence, 660 – 666.

[3] Ayob, M. Kendall, G. (2003), A Monte Carlo Hyper-Heuristic to Optimise Component Placement Sequencing for Multi Head Placement Machine, Proc. of the Int. Conf. on Intelligent Technologies, 132 – 141.

[4] Bilgin B., Özcan E., Korkmaz E. E. (2006), An Experimental Study on HyperHeuristics and Final Exam Scheduling, Proc. of the 6th Int. Conf. on the PATAT, 123 – 140.

[5] Burke, E., Elliman, D., Ford, P., Weare, B.(1996), Examination Timetabling in British Universities- A Survey, *Lecture Notes in Computer Science*, Springer-Verlag, vol. 1153:76 – 90.

[6] Burke, E. K., Kendall, G., Newall, J., Hart E., Ross, P., Schulenburg, S. (2003), Hyperheuristics: An Emerging Direction in Modern Search Technology, *Handbook of Metaheuristics. International Series in OR & Management Science*, Kluwer Academic Publishers, Boston Dordrecht London, vol. 57, 457 – 474.

[7] Burke, E.K. and Newall, J. (2003), Enhancing timetable solutions with local search methods. *Lecture Notes in Computer Science*, Springer-Verlag, 2740:195 – 206.

[8] Burke, E. K., Landa Silva, J. D. (2004), The Design of Memetic Algorithms for Scheduling and Timetbaling Problems. Krasnogor N., Hart W., Smith J. (eds.), *Recent Advances in Memetic Algorithms,* Studies in Fuzziness and Soft Computing, vol. 166, Springer, 289 – 312.
[9] Burke, E. K. S. Petrovic (2002), Recent Research Directions in Automated Timetabling, *European Journal of Operational Research*, 140(2), 266 – 280.
[10] Caramia, M. P., Dell'Olmo, and Italiano, G.F. (2001), New algorithms for examination timetabling. *Lecture Notes in Computer Science*, Springer-Verlag, 982:230 – 241.
[11] Casey, S. and Thompson, J. (2003), Grasping the examination scheduling problem. *Lecture Notes in Computer Science*, Springer-Verlag, 2740:233 – 244.
[12] Carter, M. W, Laporte, G., and Lee, S.T. (1996), Examination Timetabling: Algorithmic Strategies and Applications. *Journal of the Operational Research Society*, 47, 373 – 383.
[13] Cowling P., Kendall G., Soubeiga E. (2000), A Hyperheuristic Approach to Scheduling a Sales Summit, *LNCS 2079*, Proc. of the 3th Int. Conf. on the PATAT, 176 – 190.
[14] Cowling P., Kendall G., Soubeiga. E.(2001), A Parameter-free Hyperheuristic for Scheduling a Sales Summit. *Proc. of the 4th Metaheuristic International Conference*, MIC, 127 – 131.
[15] Di Gaspero, L. and Schaerf, A. (2001), Tabu search techniques for examination timetabling. *Lecture Notes in Computer Science*, Springer-Verlag, 2079:104–117.
[16] Even, S., Itai, A., Shamir, A. (1976), On the Complexity of Timetable and Multicommodity Flow Problems, *SIAM J. Computing*, 5(4), 691 – 703.
[17] Eiben A. E., Hinterding R., Michalewicz Z. (1999), Parameter Control in Evolutionary Algorithms, *IEEE Trans. Evolutionary Computation*, 3(2), 124 – 141.
[18] Eley M. (2006), Ant Algorithms for the Exam Timetabling Problem, Proc. of the 6th Int. Conf. on the PATAT, 167 – 180.
[19] Gaw A., Rattadilok P., Kwan R. S. K. (2004). Distributed Choice Function Hyperheuristics for Timetabling and Scheduling, Proc. of the 5th Int. Conf. on the PATAT, 495 – 498.
[20] Holland, J. H. (1975), *Adaptation in Natural and Artificial Systems*, Univ. Mich. Press
[21] Kendall G. Mohamad M. (2004), Channel Assignment in Cellular Communication Using a Great Deluge Hyperheuristic, *2004 IEEE International Conference on Network*, 769 – 773.
[22] Kendall G., Cowling P. Soubeiga E. (2002), Choice Function and Random HyperHeuristics, SEAL'02*,* 667 – 671.
[23] Krasnogor, N. (2002), *Studies on the Theory and Design Space of Memetic Algorithms*, PhD Thesis, University of the West of England, Bristol, UK.
[24] Merlot, L.T.G., Boland, N., Hughes, B.D. and Stuckey, P.J. New benchmarks for examination timetabling. Testproblem Database, http://www.or.ms.unimelb.edu.au/timetabling.html.
[25] Moscato, P., Norman, M. G. (1992), A Memetic Approach for the Traveling Salesman Problem Implementation of a Computational Ecology for Combinatorial Optimization on Message– Passing Systems, *Parallel Computing and Transputer Applications*, 177 – 186.
[26] Ozcan, E. (2005) Memetic Algorithms for Nurse Rostering, P. Yolum (Eds.): *Lecture Notes in Computer Science 3733*, Springer-Verlag, The 20th ISCIS, 482 – 492.
[27] Ozcan, E. (2005), Towards an XML based standard for Timetabling Problems, TTML, *Multidisciplinary Scheduling, Theory and Applications*, Springer Verlag, 163 – 185.
[28] Ozcan, E. (2006), An Empirical Investigation on Memes, Self-generation and Nurse Rostering, Proc. of the 6th Int. Conf. on the PATAT, 246 – 263.
[29] Ozcan, E., Bilgin B., Korkmaz E. E. (2006). Hill Climbers and Mutational Heuristic, *Lecture Notes in Computer Science 4193*, Springer-Verlag, PPSN IX, 202 – 211.
[30] Ozcan, E., Ersoy, E. (2005), Final Exam Scheduler-FES, Proc. of 2005 IEEE Congress on Evolutionary Computation, vol. 2, 1356 – 1363.
[31] Paquete, L. and Stuetzle , T. (2002), Empirical analysis of tabu search for the lexicographic optimization of the examination timetabling problem. Proc. of the 4th PATAT, 413 – 420.
[32] Qu R., Burke E.K., McCollum B., Merlot L.T.G. Lee S.Y. (2006), *A Survey of Search Methodologies and Automated Approaches for Examination Timetabling*, Technical Report NOTTCS-TR-2006-4, School of CSiT, University of Nottingham.
[33] White, G.M., Xie, B.S. and Zonjic, S. (2004) Using tabu search with long-term memory and relaxation to create examination timetables. *European Journal of Op. Research*, 153:80 – 91.