# A Construction Approach for Examination Timetabling based on Adaptive Decomposition and Ordering

Syariza Abdul Rahman[1], Edmund Burke[1], Andrzej Bargiela[1], Barry McCollum[2] and Ender Özcan[1]

[1] *University of Nottingham, School of Computer Science, Jubilee Campus, Nottingham NG8 1BB, UK*
{sax, ekb, abb, exo}@cs.nott.ac.uk

[2] *Queen's University Belfast, School of Electronics, Electrical Engineering and Computer Science, University Road Belfast, BT7 1NN, Northern Ireland, UK*
b.mccollum@qub.ac.uk

In this study, we investigate an adaptive decomposition strategy that automatically divides examinations into *difficult* and *easy* sets for constructing an examination timetable. The examinations in the difficult set are considered to be hard to place and hence are listed before the ones in the easy set. Moreover, the examinations within each set are ordered using different strategies based on graph colouring heuristics. Initially, the examinations are placed into the easy set. During the construction process, the examinations that cannot be scheduled are identified as the ones causing infeasibility and are moved forward in the difficult set to ensure earlier assignment than the others for the subsequent attempts. On the other hand, the examinations that can be scheduled remain in the easy set. Within the easy set, a new subset called the boundary set is introduced to accommodate shuffling strategies to change the given ordering of examinations. The proposed approach which incorporates different ordering and shuffling strategies is explored on the Carter benchmark problems. The empirical results show that its performance is promising and comparable to existing constructive approaches.

*Keywords*: timetabling, decomposition, graph colouring, heuristic, grouping.

# 1 Introduction

Timetabling attracts numerous researchers and practitioners due to its challenging nature. Timetabling problems are NP hard real-world problems (Even et al. 1976) that are hard to solve and often require considerable amount of either human or computational time or both. There are many types of timetabling problems e.g. educational timetabling, nurse rostering, etc. The focus of this study is the university examination timetabling problem. Principally, the examination timetabling problem is concerned with the scheduling of a list of examinations into a restricted number of time-slots while satisfying a defined set of constraints. Hard constraints must be satisfied in creating a feasible solution e.g. no student should take two examinations at the same time. Soft constraints on the other hand can be broken but it is desirable to satisfy them as much as possible. The evaluation of the degree these soft constraints are satisfied provides an indication of the overall quality of a given solution. In relation to examination timetabling, evaluating the average cost of student spread in the timetable as an indicator of how 'good' a given solution is was introduced by Carter et al. (1996). More overview information on the examination timetabling problem and associated constraints can be found in (Carter and Laporte 1996; Carter et al. 1996; Petrovic and Burke 2004; Qu et al. 2009).

Considering that the only constraint dealt with is the requirement that no student should sit two examinations at the same time, the formulation of the examination timetabling problem is closely similar to other graph colouring problems. Ülker et al. (2007) discusses a grouping representation for this type of examination timetabling problems. The vertices and edges of a graph denote the examinations and the conflicting examinations that should not be scheduled at the same time, respectively, where the colour of a vertex denotes a time-slot in the timetable. The heuristic ordering methods for graph colouring are considered constructive approaches. These approaches have been used to find an initial solution before getting further to the improvement phase. There are several heuristic ordering methods commonly used in examination timetabling i.e. largest degree, saturation degree, largest weighted degree, largest enrolment and colour degree (Carter 1986; Carter and Laporte 1996, Burke et al. 2004a).

A wide variety of approaches have been applied to examination timetabling. The approaches vary from exact methods, such as, constraint logic programming and constraint satisfaction to meta-heuristic approaches, such as, tabu search, simulated annealing and population based approaches, such as, evolutionary algorithms. Recent applications of search methodologies, such as hyper-heuristics that perform search over the heuristics space (Burke et al. 2003; Özcan et al. 2008) and case-based reasoning approaches to timetabling aim to work at a higher level of generality than meta-heuristics. An overview of methodologies employed for examination timetabling is provided in Table 1.

Recent studies in timetabling have focused on the constructive approaches for obtaining high quality solutions. Graph colouring heuristics have been customized with the adaptive approaches to order the examinations based on their difficulty of timetabling (Burke and Newall 2004). We have utilised the framework of 'squeaky wheel optimisation' (Joslin and Clement 1999), where the difficulty of scheduling an examination is identified based on its feasibility versus infeasibility in a previous iteration. In this work, the difficulty indicator of scheduling an examination was subsequently increased based on a certain parameter to enable it be scheduled earlier in the next iteration. In 2009, Abdul Rahman et al. extended this study by introducing more strategies for choosing an examination to be scheduled and the time-slots. In another adaptive approach, Casey and Thompson (2003) developed a GRASP algorithm for solving the examination timetabling problems. In their approach, the next examination to be scheduled is chosen from the top items in the list (called candidate list) using roulette wheel selection and then assigned to the first available slot.

Table 1. Some representative methodologies for solving examination timetabling problems

| **Methodology** | **Reference(s)** |
|---|---|
| Cluster-based/decomposition | Balakrishnan et al. (1992), Burke and Newall (1999), Qu and Burke (2007) |
| Tabu search | Di Gaspero and Schaerf (2001), White and Xie (2001) |
| Simulated annealing | Thompson and Downsland (1998), Merlot et al. (2003) |
| Great deluge algorithm | Burke et al. (2004b) |
| Variable neighbourhood search | Burke et al. (2010) |
| Large neighbourhood search | Abdullah et al. (2007) |
| Iterated local search | Caramia et al. (2001) |

| | |
|---|---|
| GRASP | Casey and Thompson (2003) |
| Genetic algorithms | Burke et al. (1995), Ülker et al. (2007) |
| Memetic algorithms | Burke and Newall (1999), Ozcan and Ersoy (2005), Ersoy et al. (2007) |
| Ant algorithms | Eley (2007) |
| Exact method | Boizumault et al. (1996), David (1998), Merlot et al. (2003) |
| Multi-objective | Petrovic and Bykov (2003), Ülker et al. (2007) |
| Hyper-heuristic | Bilgin et al. (2007), Ersoy et al. (2007), Pillay and Banzhaf (2009) |
| Case-based reasoning | Burke et al. (2006) |
| Fuzzy approaches | Asmuni et al. (2009) |
| Neural network | Corr et al. (2006) |
| Constructive approaches | Burke and Newall (2004), Abdul Rahman et al. (2009) |

The study by Qu and Burke (2007) describes an adaptive decomposition approach for constructing an examination timetable. This paper draws upon the research on similar adaptive approaches that make use of a decomposition strategy. We propose an approach which divides the problem into two sub-problems. We adopt the same naming convention introduced by Qu and Burke (2007) for these sets as *difficult* and *easy*. In this study, the problem is decomposed into difficult and easy sets at each iteration. A timetable is constructed based on the associated heuristic ordering for each set. We also introduce an additional set of examinations which is located in between the difficult and easy sets, which is referred to as the *boundary* set. This study describes several mechanisms associated with the boundary set in order to vary the search space of solutions. In Section 2, we present the details of our approach based on adaptive decomposition and ordering for examination timetabling. Section 3 describes the experimental data and discusses the results. Finally, the conclusion is provided in Section 4.

## 2 Automated Decomposition and Ordering of Examinations

Most of the timetabling approaches described do not make use of the information obtained from the process of building an infeasible timetable. The examinations causing the infeasibility of a solution provide an indication that those examinations are very difficult to place and should perhaps be treated in different ways. We propose a general constructive framework as presented in Pseudocode 1 for solving the examination timetabling problem based on the automated decomposition of a set of examinations into two sets i.e. difficult and easy.

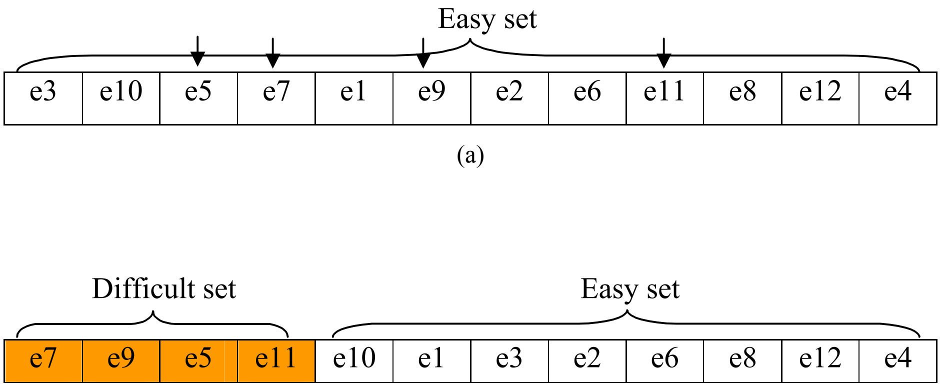
Pseudocode 1: Improvement and construction of a timetable based on automated decomposition and ordering of examinations.

```
E={e1, e2,…., eN}
BoundarySetSize=δ
EasySet=E; DifficultSet=ø; BoundarySet= ø; TempSet=ø
Divide E into subsets
FOR i=0 to MAXIter
  OrderExamsWithinSubsets(DifficultSet, EasySet)
  BoundarySet=CreateBoundarySet(DifficultSet, EasySet)
  WHILE (there are examinations to be scheduled)
    Consider changing the ordering of examinations
     using Shuffling-Strategy
    Employ Selection-Strategy to choose an
      unscheduled exam, e
    IF e can be scheduled THEN
       TempSet=TempSet ∪{e}
       Schedule e to the time-slot with the least penalty
         In the case of the availability of multiple
         time-slots with the same penalty,
         choose one randomly
    ELSE
       Move exam e to DifficultSet
    END-IF
    EasySet=TempSet
   END-WHILE
   Evaluate solution, store if it is the best found so far
END-FOR
```

During each iteration, a new solution is constructed from an ordered list of examinations. The difficult set consists of the examinations that cannot be placed into a time-slot within the timetable due to some conflicts with other examinations from the previous iteration. These examinations need to be associated with a large penalty imposed on the unplaced examinations. On the other hand, the examinations in the easy set cause no violations during the timetabling. In our approach, all the examinations that contribute to the infeasibility in a solution are given priority. They are moved forward in the ordered list of examinations and treated first. Such examinations are detected and included in the difficult set at each iteration and a predefined ordering strategy is employed before their successive assignment to the available timeslots. The remaining examinations that generate no feasibility issues are placed into the easy set and the original ordering of those examinations is maintained. In order to incorporate a stochastic component for the selection of examinations from the generated ordering, some shuffling strategies are utilised. The following subsections discuss these strategies.

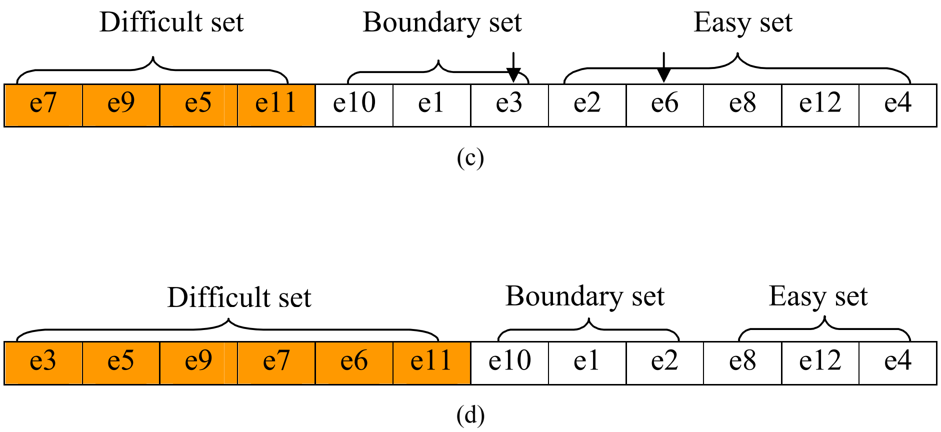
## 2.1 Interaction between Difficult and Easy Sets through a Boundary Set

An adaptive decomposition approach is experimented with using two graph colouring heuristics for generating the initial ordering of examinations. We have tested the largest degree heuristic that orders the examinations decreasingly with respect to the number of conflicts with each examination and the saturation degree heuristic that dynamically orders the unscheduled examinations based on the number of available time-slots for each during the timetable construction. The reason for testing these two graph colouring heuristics is to compare their achievement in terms of solution quality and the contribution of difficult set size, as they represent static and dynamic ordering heuristics. Initially, all the examinations are considered to be a member of the easy set (as illustrated in Figure 1(a)).



(a)

(b)


Difficult set
Boundary set
Easy set
e7 e9 e5 e11 e10 e1 e3 e2 e6 e8 e12 e4
Difficult set
Boundary set
Easy set
e3 e5 e9 e7 e6 e11 e10 e1 e2 e8 e12 e4

(c)

(d)

Figure 1. (a) All examinations are in a easy set in the first iteration and examinations that cause infeasibility are marked, (b) difficult and easy sets after an iteration resulting with an infeasible solution, (c) boundary set with a prefixed size is added to the difficult set after an iteration and reordering is performed, (d) the step in (a) is repeated and the infeasible examinations are placed in the difficult size, the size of difficult set increased.

During each iteration, the examinations causing infeasibility are identified. As in Figure 1(a), all such examinations are marked as a member of the difficult set to be moved forward towards the top of the list of examinations (Figure 1(b)), while the examinations that caused no violation during the assignment to a time-slot remain in the easy set. In Figure 1(c), the boundary set is created between the difficult and easy set and is merged with the difficult set before a reordering is performed to the difficult set. In the next iteration, more infeasible examinations are detected and included in the difficult set. Consequently, the size of the difficult set is increased from one iteration to another.

## 2.2 Swapping the Examinations Between Difficult and Boundary Sets

This strategy shuffles the difficult set and the boundary set by swapping the examinations in between them randomly. Occasionally, the examination causing infeasibility is not necessarily the one that is very difficult to schedule. The infeasibility may happen due to the previous assignment and ordering. This strategy introduces the opportunity for some of the examinations in the difficult set to be chosen later in the timetable. There is also a possibility that the examinations in the boundary set are swapped back to the original set because this process is done randomly. Figure 2 illustrates how the swapping of examinations between two sets might take place.

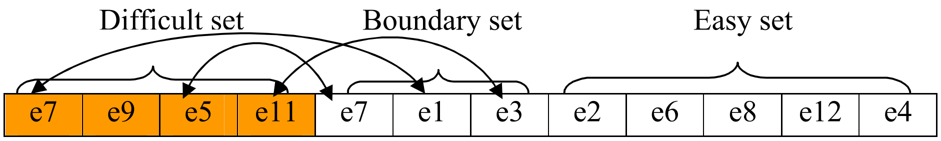


Figure 2: The boundary set is swapped with the difficult set and is reordered before assigning examinations to the time-slots.

### 2.3 Roulette Wheel Selection for Examinations

We utilised a roulette wheel selection strategy that incorporates a stochastic element in choosing examinations before assigning them to the time-slots. If there is no improvement evident for a certain time, a list of examination of size *n* was chosen from the ordered list in the difficult set from which and an examination is chosen based on a probability. The probabilities of an examination being chosen were calculated based on a score, $s_i$ of each examination in the list of size *n*. The new size of the difficult set will be the set which includes the size of boundary whenever there is improvement to the solution quality. The score value, $s_i$ is a dynamic measure that is obtained from the largest and saturation degree values (as in equation 1), where $Num\_clash_i$ is the number of examinations in conflict with the examination *i*, *Max_clash* is the maximum number of conflicts with all examinations, $Sat\_degree_i$ is the saturation degree value for the examination *i* and *Num_slots* is the number of time-slots given to the specified problem. *Sat_degree* value in this problem is initialised as 1.

$$s_i = \frac{Num\_clash_i}{Max\_clash} + \frac{Sat\_\deg ree_i}{Num\_slots} \qquad (1)$$

The probability, $p_i$ of an examination being chosen from *n* list of examinations is,

$$p_i = \frac{s_i}{\sum_{i=0}^{n-1} s_i}, \qquad (2)$$

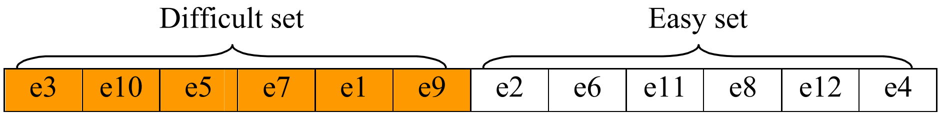
A random number from (0, 1) is obtained in order to choose an examination from a list of examination of size *n*. Those examinations with higher score values will have a greater chance of being chosen.

## 2.4 Comparison of Our Approach to a Previous Study

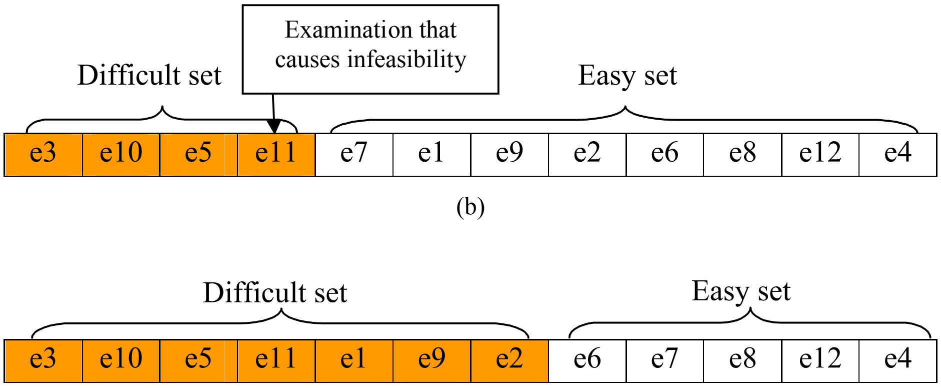
Qu and Burke (2007) previously proposed an adaptive decomposition approach to construct examination timetables. Their approach starts with an initial ordering of examinations using a graph colouring heuristic, namely saturation degree. In the approach, a perturbation is made by randomly swapping two examinations in order to obtain a *better* ordering. Examinations are then decomposed into two sets: difficult and easy.

The initial size of the difficult and easy sets are prefixed as half of the number of examinations in a given problem as shown in Figure 3(a). At each iteration, the size of the difficult set is modified according to the feasibility of the solution. If the solution is infeasible after the adjustment of the ordering of examinations then the first examination that causes infeasibility (e.g. e11) is moved forward for a fixed number of places (e.g., five as illustrated in Figure 3(b)). The size of the difficult set is then re-set to the point where the difficult examination is placed. Otherwise, if feasible solution or an improved solution is obtained, the size of the difficult set is increased (Figure 3(c)).

Our approach initialises with the easy set including all the examinations and the difficult set is formed during each construction phase at each iteration. The size of the difficult set depends on the number of unscheduled examinations that cannot be assigned to any time-slot from all previous iterations. The size of the difficult set never decreases and after a certain number of iteration, the number of examinations in the difficult set might be sustained. On the other hand, in the previous approach, the size of the difficult set is prefixed and increased when the feasible solution or improved solution is obtained statically. The set is also allowed to shrink. Additionally, the previously proposed approach uses an initial ordering and reorders all the examinations without using a heuristic, which is not the case in our approach. Although we have used the same approach for reordering the examinations in difficult and easy sets separately, examinations in different sets can be reordered based on a different heuristic at each iteration.



(a)



(b)

(c)

Figure 3. Difficult and easy sets (a) in the first iteration, (b) after an iteration is over (a) resulting with an infeasible solution, (c) after an iteration is over (a) resulting with a feasible solution.

# 3 Experiments

The experiments were tested on benchmark problems introduced by Carter et al. (1996) and are publicly available at ftp://ftp.mie.utoronto.ca/pub/carter/testprob/. In this study, we used version I of the 12 problems that were adapted from Qu et al. (2009) to differentiate various versions of the problem. During the experiments, five runs are performed and the stopping conditioned has been set as 10 000 iteration as to be equal with the experiment done by Qu and Burke (2007). Two types of heuristic ordering for initialisation are investigated: largest degree (LD) and saturation degree (SD). The difficult set is created using these two initial orders are then reordered with either largest degree or saturation degree. In this study, the same heuristic ordering is used for the examinations in the easy set. The heuristics used in a given approach will be denoted by a triplet as [*heuristic used for the initial ordering – heuristic used for ordering the examinations in the difficult set – heuristic used for ordering the examinations in the easy set*] from this point onwards. The size of the boundary set is fixed as 5.

Table 2 summarises the experimental results obtained applying the proposed approach to the benchmark problem instances. By looking at the best ordering for the difficult sets, we observe that the adding boundary set strategy performed better with largest degree initial ordering where nine out of the twelve problem instances has performed significantly better than saturation degree initial ordering while the swapping strategy has performed better with saturation degree initial ordering with seven out of twelve problem instances are better compare to largest degree initial ordering. The best combination ordering for adding boundary set strategy is [LD-SD-LD] while the swapping boundary set strategy performed the best with [SD-SD-SD]. From the perspective of the strategies, it is clear that by swapping the boundary set with the difficult set produced better solution quality as compared to just combining the boundary set as a part of the difficult set. The swapping strategy has obtained seven better results while the combining strategy produced the better results for only five problem instances.

Table 2. Comparing solution quality for (a) [LD-LD-LD], (b) [SD-LD-SD], (c) [LD-SD-LD], (d) [SD-SD-SD] by adding boundary set into difficult set and swapping examinations between boundary and difficult sets with δ=5. (LD: largest degree; SD: saturation degree) (Bold font indicates the best for different ordering and strategy and italic is the best of all for each problem instance).

| **Problem** | **Add the boundary set (δ=5) into the difficult set** | | | | **Swap examinations in the boundary (δ=5) and difficult sets** | | | |
|---|---|---|---|---|---|---|---|---|
| | **(a)** | **(b)** | **(c)** | **(d)** | **(a)** | **(b)** | **(c)** | **(d)** |
| **car91** | 5.72 | 5.60 | 5.77 | **5.44** | 5.71 | 5.37 | 5.75 | ***5.34*** |
| **car92** | 4.97 | ***4.76*** | 4.90 | 4.85 | 5.03 | **4.87** | 4.95 | 4.91 |
| **ear83 I** | ***41.36*** | 41.42 | 42.53 | 42.51 | **41.90** | 42.48 | 43.38 | 42.78 |
| **hec92 I** | 12.98 | 12.76 | ***12.24*** | 12.45 | 13.32 | 13.15 | 12.72 | **12.52** |
| **kfu93** | 16.68 | 16.57 | **16.35** | 16.40 | ***16.16*** | 16.61 | 16.38 | 16.49 |
| **lse91** | 13.44 | 12.96 | **12.64** | 12.85 | 13.45 | ***12.52*** | 12.93 | 12.95 |
| **rye93** | 11.13 | 10.79 | ***10.23*** | 10.31 | 11.35 | 10.66 | 10.53 | **10.27** |
| **sta83 I** | 163.93 | 162.12 | **159.32** | 159.74 | 161.98 | 159.34 | 159.08 | ***158.99*** |
| **tre92** | 9.77 | 9.72 | **9.54** | 9.69 | 9.81 | 9.50 | 9.66 | ***9.41*** |
| **ute92** | 30.68 | 30.08 | **29.11** | **29.11** | 30.21 | 29.79 | 29.34 | ***28.96*** |
| **uta92 I** | 3.92 | **3.78** | 3.96 | 3.87 | 3.96 | ***3.77*** | 3.89 | 3.82 |
| **yor83 I** | 45.85 | 46.97 | ***44.16*** | 44.75 | 45.84 | 46.28 | **45.30** | 45.39 |

In the next set of experiments, the effect of incorporating the roulette wheel into the examination selection process is tested with *n* = 3. As we can see from the results in Table 3, the adding boundary set strategy with roulette wheel selection has performed better by providing eight better solutions as compared to the swapping strategy with roulette wheel selection. From the results, the adding boundary set and selection strategy performed the best with combination of [LD-SD-LD] while the best combination ordering for swapping with selection strategy is [SD-LD-SD]. Comparing the best results obtained from the strategies without roulette wheel selection in Table 2 and the strategies with roulette wheel selection in Table 3, it shows that when incorporating the selection strategy improves the performance of the approach.

Table 3. Comparing solution quality for (a) [LD-LD-LD], (b) [SD-LD-SD], (c) [LD-SD-LD], (d) [SD-SD-SD] with shuffling strategies of adding the boundary set into the difficult set and swapping examinations between the boundary and difficult sets with δ=5 and includes roulette wheel selection for examinations with *n*=3. (LD: largest degree; SD: saturation degree) (Bold font indicates the best for different ordering and strategy and italic is the best of all for each problem instance).

| **Problem** | **Add the boundary set (δ=5) into the difficult set + Roulette wheel selection (*n*=3)** | | | | **Swap examinations in the boundary (δ=5) and difficult sets + Roulette wheel selection (*n*=3)** | | | |
|---|---|---|---|---|---|---|---|---|
| | **(a)** | **(b)** | **(c)** | **(d)** | **(a)** | **(b)** | **(c)** | **(d)** |
| **car91** | 5.67 | ***5.28*** | 5.64 | 5.43 | 5.67 | 5.57 | 5.77 | **5.48** |
| **car92** | 4.98 | 4.91 | 4.81 | ***4.76*** | 4.90 | 4.95 | **4.86** | 4.89 |
| **ear83 I** | 41.29 | ***40.60*** | 41.39 | 42.74 | **41.67** | 42.24 | 42.14 | 42.51 |
| **hec92 I** | ***12.09*** | 12.36 | 12.45 | 12.70 | **12.20** | 12.97 | 12.38 | 12.48 |
| **kfu93** | 16.25 | 16.22 | ***16.04*** | 16.43 | 16.43 | 16.07 | 16.20 | **16.05** |
| **lse91** | 12.70 | ***12.03*** | 12.76 | 12.67 | 13.06 | 12.75 | **12.14** | 12.82 |
| **rye93** | 10.52 | **10.25** | 10.40 | 10.41 | 10.45 | 10.45 | ***10.20*** | 10.54 |
| **sta83 I** | 160.20 | 162.26 | **158.68** | 160.00 | 158.60 | 160.43 | ***158.39*** | 161.75 |
| **tre92** | 9.31 | 9.72 | ***9.08*** | 9.55 | 9.43 | 9.79 | **9.21** | 9.63 |
| **ute92** | **27.81** | 27.93 | 28.57 | 27.90 | 28.01 | 27.84 | ***27.30*** | 27.55 |
| **uta92 I** | 3.95 | ***3.73*** | 3.83 | 3.81 | 3.88 | 3.92 | 3.89 | **3.74** |
| **yor83 I** | 45.48 | 45.24 | 45.76 | **44.36** | 545.04 | ***44.33*** | 44.51 | 44.81 |

Table 4 compares our best results obtained from the strategy of roulette wheel selection to the other previous results based on constructive approaches. Given by Qu and Burke (2007) is the closest comparison to our approach as they have also implemented a decomposition strategy. Comparing the solutions across all problem instances, it is observed that our approach does not yield the best results. However, it provides one better result when compared to the approach proposed by Qu and Burke (2007) for car91. Moreover, we have obtained better results than the approach by Asmuni et al. (2009) for four problems (car91, lse91, rye93 and ute92), Carter et al. (1996) for two problems (car91, car92), respectively. However, Burke and Newall (2004) and Qu and Burke (2007) do not provide the result for rye93.

Table 4. Comparison of different constructive approaches (LD: largest degree; SD: saturation degree; RWS: roulette wheel selection) (The bold entries indicate the best results for constructive approaches only, while the italic ones indicate the best results for the decomposition approach).

| **Problem** | **Burke & Newall (2004)** | **Carter et al. (1996)** | **Asmuni et al. (2009)** | **Abdul Rahman et al. (2009)** | **Qu & Burke (2007)** | **SD-LD-SD(RWS)** |
|---|---|---|---|---|---|---|
| **car91** | **4.97** | 7.10 | 5.29 | 5.08 | 5.45 | *5.28* |
| **car92** | **4.32** | 6.20 | 4.54 | 4.38 | *4.5* | 4.91 |
| **ear83 I** | 36.16 | 36.40 | 37.02 | 38.44 | *36.15* | 40.60 |
| **hec92 I** | 11.61 | **10.80** | 11.78 | 11.61 | *11.38* | 12.36 |
| **kfu93** | 15.02 | **14.00** | 15.80 | 14.67 | *14.74* | 16.22 |
| **lse91** | 10.96 | **10.50** | 12.09 | 11.69 | *10.85* | 12.03 |
| **rye93** | - | **7.30** | 10.38 | 9.49 | - | 10.25 |
| **sta83 I** | 161.90 | 161.50 | 160.40 | 157.72 | ***157.21*** | 162.26 |
| **tre92** | **8.38** | 9.60 | 8.67 | 8.78 | *8.79* | 9.72 |
| **ute92** | 27.41 | **25.80** | 28.07 | 26.63 | *26.68* | 27.93 |
| **uta92 I** | **3.36** | 3.50 | 3.57 | 3.55 | *3.55* | 3.73 |
| **yor83 I** | 40.88 | 41.70 | **39.80** | 40.45 | 42.2 | 45.24 |

The overall results once again highlight the importance of the methodology used to change the ordering of difficult examinations, particularly the ones causing infeasibility. In our approach, the ordering of the examinations within the difficult set with respect to the others appears to be vital combined with the assignment strategy. As shown in Figure 4, for the experiments adding and swapping boundary set and difficult set without roulette wheel selection, the average number of the examinations in the difficult set varies with different ordering strategies. The approach using the largest degree ordering generates infeasibility more often for a given solution during the time-slot assignments as compared to the one using the saturation degree ordering. On the other hand, saturation degree ordering might easily create a feasible solution for some problem instances (e.g. car91 and uta92 I). However, using the saturation degree alone does not for guarantee a good solution quality.

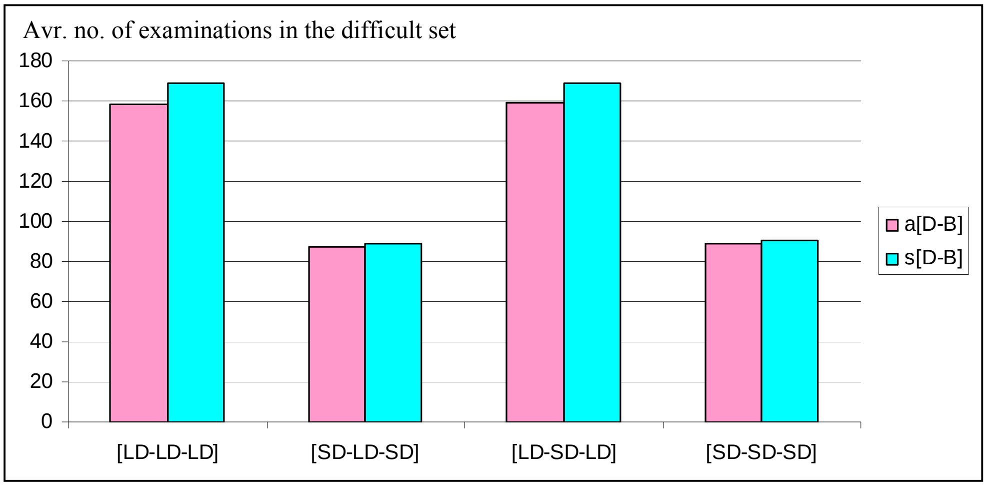
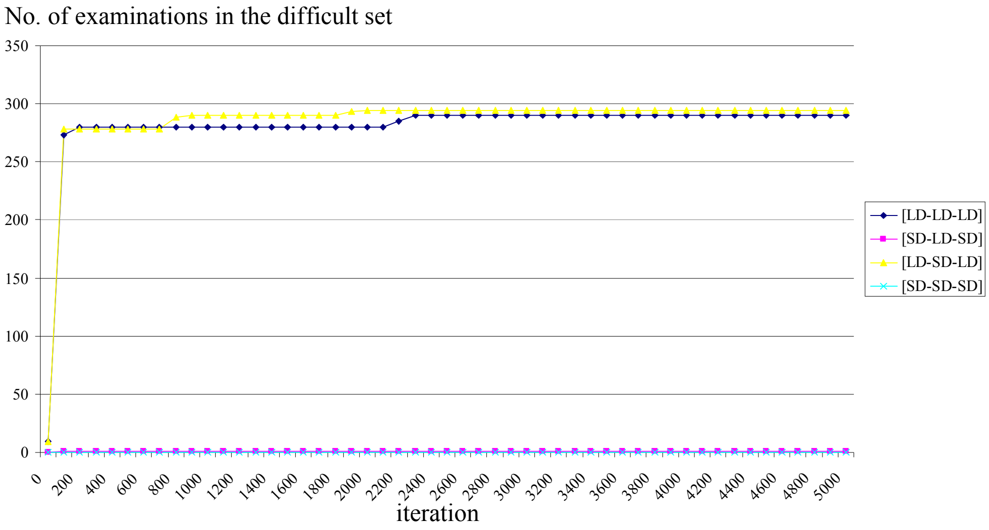


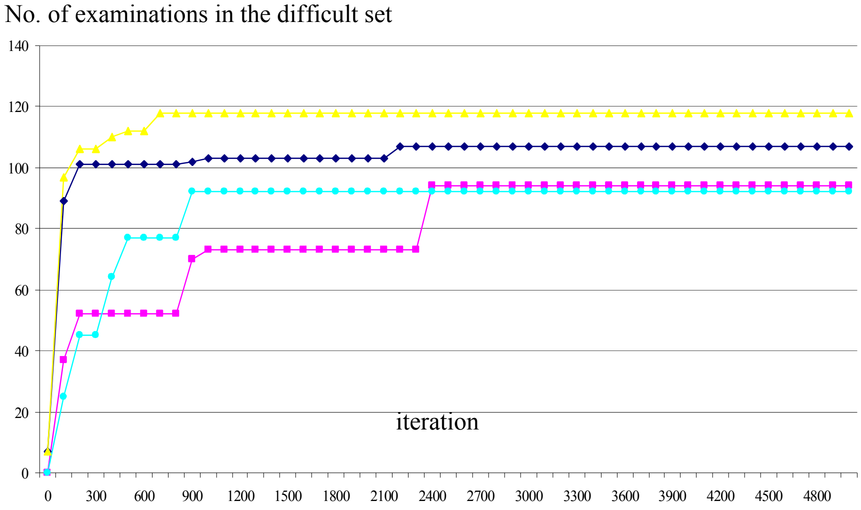
Figure 4. Average number of examinations in the difficult set (its size) over all problems considering all shuffling strategies using different initialisation and reordering heuristics. (LD: largest degree, SD: saturation degree, B: boundary set, D: difficult set, a: add, s: swap).

In some cases, using the saturation degree ordering may easily create a feasible solution when adding or swapping with the boundary set, the infeasible examinations can be obtained in this approach since this approach gives priority of ordering the difficult set. Consequently, adding or swapping the boundary set with the difficult set might have increased the number of examinations in the difficult set.

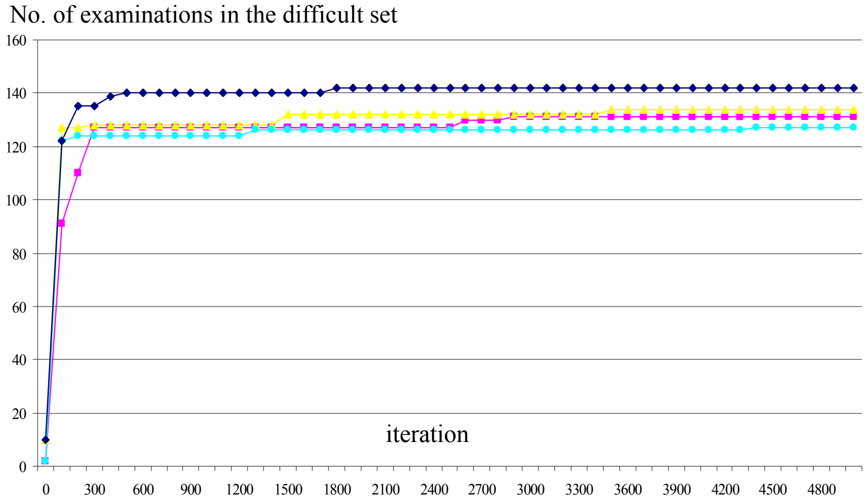
Figure 5((a), (b), (c)) illustrate the number of infeasible at each 100 iteration for different combination of initial ordering and reordering heuristics for the difficult set for car91, kfu93 and yor83 I, respectively. It shows that using largest degree causes increasing number of examinations to generate infeasible solution when compared to the saturation degree. car91 has an obvious difference in the number of infeasible examinations when comparing with the other two types of ordering i.e. [LD-LD-LD] and [LD-SD-LD]. In the other problem, kfu93 and yor83 I the number of infeasible examinations for different ordering is approximately the same but still using [SD-LD-SD] and [SD-SD-SD] are slightly advantageous. In all problems, the number of infeasible examinations is converged to a steady state after some point.



(a) car91



(b) kfu93



(c) yor83 I

Figure 5. The change in the size of the difficult set and the solution quality at every 100 iteration during the sample runs for (a) car91, (b) kfu93, (c) yor83 I. (LD: largest degree, SD: saturation degree).

# 4 Conclusion

This study discusses a novel approach based on adaptive strategies that decomposes the examinations in a given problem into two sets: a set of difficult to schedule and a set of easy to schedule examinations. This decomposition is performed automatically at each iteration, and is augmented with suitable ordering of examinations within each set. In this study, it is observed that by merging or swapping the boundary set with the difficult set could improve the solution quality. A stochastic component based on roulette wheel selection is embedded into the approach in order to shuffle the order of examinations. This mechanism gives a higher chance to an examination with a higher score to be selected for timetabling. It is observed that using saturation degree could decrease the possibility of creating infeasible solution and that dynamic ordering gives better ordering of examinations in the list. This preliminary study shows that the proposed approach is simple to implement, yet it is competitive to the other previous constructive approaches. In this study, the same ordering heuristics are used for reordering the examinations in the difficult and easy sets. In fact, the proposed framework allows the use of different strategies. As a future work, different strategies will be investigated for reordering of examinations and choosing the examinations from the difficult set.

# 5 References

Abdul Rahman, S., Bargiela, A., Burke, E. K., McCollum, B., & Ozcan, E. (2009). Construction of examination timetables based on ordering heuristics. ISCIS 2009 (pp. 727-732) (In proceeding of 24th international symposium of computer and information sciences, Cyprus).

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

Asmuni, H., Burke, E. K., Garibaldi, J. M., McCollum, B., & Parkes, A. J. (2009). An investigation of fuzzy multiple heuristic orderings in the construction of university examination timetables. *Computers and Operations Research*, 36, 981-1001.

Balakrishnan, N., Lucena A., & Wong, R. T. (1992). Scheduling examinations to reduce second order conflicts. *Computers and Operations Research*, 19, 353-361.

Bilgin, B., Özcan, E., & Korkmaz, E. E. (2007). An experimental study on hyper-heuristics and exam scheduling, *Practice and theory of automated timetabling* 2006, Springer-Verlag, selected papers, *Lecture notes in computer science*, vol. 3867, 394-412.

Boizumault, P., Delon, Y., & Peridy, L. (1996). Constraint logic programming for examination timetabling. *Journal of Logic Programming*, 26(2), 217-233.

Burke, E. K., Petrovic, S., & Qu, R. (2006). Case based heuristic selection for timetabling problems. *Journal of Scheduling*, 9, 115-132.

Burke, E. K., & Newall, J. P. (2004). Solving examination timetabling problems through adaptation of heuristics orderings. *Annals of Operational 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., Bykov, Y., Newall, J. P., & Petrovic, S. (2004b). A time-predefined local search approach to exam timetabling problem. *IIE Transactions*, 36(6), 509-528.

Burke, E. K., Eckersley, A., McCollum, B., Petrovic, S., & Qu, R. (2010). Hybrid Variable Neighbourhood Approaches to University Exam Timetabling. *European Journal of Operational Research*, to appear 2010.

Burke, E. K., Elliman, D. G., & Weare, R. (1995). A hybrid genetic algorithm for highly constrained timetabling problems. Paper presented at the 6th International Conference on Genetic Algorithms (ICGA'95), San Francisco, CA, USA, Pittsburgh, USA.

Burke, E. K., Kendall, G., Newall, J., Hart, E., Ross, P., & Schulenburg, S. (2003). Hyperheuristics: an emerging direction in modern search technology. In F. Glover & G. A. Kochenberger (Eds.), *Handbook of Metaheuristics*. 457-474. Netherlands: Kluwer.

Burke, E. K., Kingston, J., & de Werra, D. (2004a). Applications to timetabling. In J. Gross & J. Yellen (Eds.), *Handbook of graph theory* (pp. 445-474). Chapman Hall/CRC Press.

Caramia, M., DellOlmo, P., & Italiano, G. F. (2001). New algorithms for examination timetabling. In S. Naher & D. Wagner (Eds.), *Lecture notes in computer science*. *Algorithm engineering 4th international workshop, proceeding WAE 2000* (vol. 1982, pp. 230-241). Berlin: Springer.

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

Carter, M. W., & Laporte, G. (1996). Recent developments in practical examination timetabling. In E. K. Burke & P. Ross (Eds.), *Lecture notes in computer science*. *Practice and theory of automated timetabling I: selected papers from the 1st international conference* (vol. 1153, pp. 3-21). Berlin: Springer.

Carter, M. W. (1986). A survey of practical applications of examination timetabling algorithms. *Operational Research*, 34, 193-202.

Casey, S., & Thompson, J. (2003). GRASPing the examination scheduling problem. In E. K. Burke & P. De Causmaecker (Eds.), *Lecture notes in computer science*. *Practice and theory of automated timetabling IV: selected papers from the 4th international conference* (vol. 2740, pp. 234-244). Berlin: Springer.

Corr, P. H., McCollum, B., McGreevy, M. A., & McMullan, P. (2006). A new neural network based construction heuristic for the examination timetabling problem. In the international conference on parallel problem solving from nature (PPSN 2006) (pp. 392-401), Reykjavik, Iceland, September 2006.

David, P. (1998). A constraint-based approach for examination timetabling using local repair techniques. In E. K. Burke & M. W. Carter (Eds.), *Lecture notes in computer science. Practice and theory of automated timetabling II: Selected papers from the 2nd international conference* (vol. 1408, pp. 169-186). Berlin: Springer.

Di Gaspero, L., & Schaerf, A. (2001). Tabu search techniques for examination timetabling. In E. K. Burke & W. Erben (Eds.), *Lecture notes in computer science. Practice and theory of automated timetabling III: selected papers from the 3rd international conference* (vol. 2079, pp. 104-117). Berlin: Springer.

Eley, M. (2007). Ant algorithms for the exam timetabling problem. In E. K. Burke & H. Rudova (Eds.), *Lecture notes in computer science. Practice and theory of automated timetabling VI: selected papers from the 6th international conference* (vol. 3867, pp. 364–382). Berlin: Springer.

Ersoy, E., Özcan, E., & Uyar, Ş. (2007). Memetic algorithms and hyperhill-climbers. In P. Baptiste, G. Kendall, A. Munier-Kordon, & F. Sourd (Eds.), *Proceedings of the 3rd multidisciplinary international conference on scheduling: theory and applications* (pp. 159-166). Paris, France.

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

Joslin, D. E., & Clements, D. P. (1999). “Squeaky wheel” optimization. *Journal of Artificial Intelligence Research*, 10, 353-373.

Merlot, L. T. G., Boland, N., Hughes, B. D., & Stuckey, P. J. (2003). A hybrid algorithm for the examination timetabling problem. In E. K. Burke & P. De Causmaecker (Eds.), *Lecture notes in computer science. Practice and theory of automated timetabling IV: selected papers from the 4th international conference* (vol. 2740, pp. 207-231). Berlin: Springer.

Özcan, E., Bilgin, B., & Korkmaz, E. E. (2008). A comprehensive analysis of hyper-heuristics, *Intelligent Data Analysis*, 12(1), 3-23.

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

Petrovic, S., & Burke, E. K. (2004). Chapter 45: University timetabling. In J. Leung (Ed.), *Handbook of scheduling: Algorithms, models, and performance analysis*. CRC Press.

Petrovic, S., & Bykov, Y. (2003). A multiobjective optimisation technique for exam timetabling based on trajectories. In E. K. Burke & P. De Causmaecker (Eds.), *Lecture notes in computer science. Practice and theory of automated timetabling IV: selected papers from the 4th international conference* (vol. 2740, pp. 179-192). Berlin: Springer.

Pillay, N., & Banzhaf, W. (2009). A study of heuristic combinations for hyper-heuristic systems for the uncapacitated examination timetabling problem. *European Journal of Operational Research*, 197(2), 482-491.

Qu, R., & Burke, E. K. (2007). Adaptive decomposition and construction for examination timetabling problems. In Multidisciplinary international scheduling: theory and applications (MISTA'07) (pp. 418-425). Paris, France.

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

Thompson, J. M., & Dowsland, K. A. (1998). A robust simulated annealing based examination timetabling system. *Computers and Operations Research*, 25(7/8), 637-648.

Ülker, Ö., Özcan, E., & Korkmaz, E. E. (2007). Linear linkage encoding in grouping problems: applications on graph coloring and timetabling, *Practice and theory of automated timetabling* 2006, Springer-Verlag, selected papers, *Lecture notes in computer science*, vol. 3867, 347-363.

White, G. M., & Xie, B. S. (2001). Examination timetables and tabu search with longer-term memory. In E. K., Burke & W. Erben (Eds.), *Lecture notes in computer science. Practice and theory of automated timetabling III: selected papers from the 3rd international conference* (vol. 2079, pp. 85-103). Berlin: Springer.