Application of a Hybrid Multi-Objective Evolutionary Algorithm to the Uncapacitated Exam Proximity Problem

Pascal Côté, Tony Wong, and Robert Sabourin

Department of Automated Manufacturing Engineering
École de technologie supérieure, Université du Québec
1100 Notre-Dame, Montréal (Québec), Canada
{tony.wong, robert.sabourin}@etsmtl.ca

Abstract. A hybrid Multi-Objective Evolutionary Algorithm is used to tackle the uncapacitated exam proximity problem. In this hybridization, local search operators are used instead of the traditional genetic recombination operators. One of the search operators is designed to repair unfeasible timetables produced by the initialization procedure and the mutation operator. The other search operator implements a simplified Variable Neighborhood Descent meta-heuristic and its role is to improve the proximity cost. The resulting non dominated timetables are compared with thouse produced by other optimization methods using 15 public domain datasets. Without special fine-tuning, the hybrid algorithm was able to produce timetables ranking first and second in 9 of the 15 datasets.

1 Introduction

This paper presents a hybrid Multi-Objective Evolutionary Algorithm (MOEA) designed for the uncapacitated exam proximity problem in which a timetable has to offer student maximum free time between exams while satisfying the clashing constraint (exam conflicts) and without regard to the seating capacity. The proposed multi-objective approach also considers timetable length as an optimization objective. It is thus possible to generate a set of alternative solutions without multiple execution of the optimization process. The hybridization is comparable to Radcliffe and Surry's Memetic Algorithm (MA) description [1]. In the basic MA, local search operators are added to the genetic recombination and mutation operators and local optimization is performed following the genetic reproduction phase. To obtain a reasonable computation requirement, the local search operators are usually implemented as greedy hill climbers. It is possible to introduce more sophisticated local search heuristics but the optimization response time will increase as a function of search complexity. A way to incorporate advanced local search heuristics while maintaining acceptable computation time is to remove the genetic recombination operator from the MA. The genetic recombination can be viewed as an exploitation strategy where the search focuses on neighbors of good solutions. A local search heuristic can play the same exploitative role much more efficiently in combinatorial optimization problems.

This paper is organized as follows. Section 2 describes the problem model including the clashing constraint (exam conflicts). Section 3 presents a brief survey of previous methods. This survey is restricted to research carried out on the datasets provided by Carter et al. [2], Burke et al. [3] and Merlot et al. [4]. Section 4 explains the multi-objective approach investigated in this work. Section 5 details the results, and the conclusions follow in section 6.

2 Problem Description

Given a set of exams $\mathcal{E} = \{e_1, e_2, \ldots, e_{|\mathcal{E}|}\}$ and a set of timeslots $\mathcal{T} = \{1, 2, \ldots, |\mathcal{T}|\}$, the goal of examination timetabling is to obtain an assignment where each exam in $\mathcal{E}$ is allocated to a timeslot in $\mathcal{T}$. The result of such an assignment is a timetable represented here by a set h of ordered couples (t, e) where $t \in \mathcal{T}$ and $e \in \mathcal{E}$. A timetable h is called feasible if it satisfies all required constraints. Otherwise, h is identified as unfeasible. A fundamental requirement in exam timetabling is to prohibit clashing, or exam conflicts (a student having to take 2 or more exams in a given timeslot). In this work, clashing is a hard constraint and can be expressed as:

$$\sum_{k=1}^{|\mathcal{T}|} \sum_{i=1}^{|\mathcal{E}|} \sum_{j=1}^{|\mathcal{E}|} \eta_{ij} \varepsilon_{ik} \varepsilon_{jk} = 0. \tag{1}$$

In (1), η_{ij} is the number of students taking exam e_i and exam e_j, $\varepsilon_{ij} \in \{0, 1\}$ is a binary quantity with $\varepsilon_{ij} = 1$ if exam e_i is assigned to timeslot j. Otherwise, $\varepsilon_{ij} = 0$. A timetable is feasible if (1) is satisfied.

The basic examination timetabling problem is to minimize the number of timeslots used in a feasible timetable. This minimization problem is defined as:

$$\begin{aligned} &\text{minimize } u_1 = |\mathcal{T}|\,, \\ &\text{s.t.} \qquad \sum_{k=1}^{|\mathcal{T}|} \sum_{i=1}^{|\mathcal{E}|} \sum_{j=1}^{|\mathcal{E}|} \eta_{ij} \varepsilon_{ik} \varepsilon_{jk} = 0. \end{aligned} \tag{2}$$

Note that (2) is equivalent to the graph-coloring problem. A more elaborate problem is the exam proximity problem (EPP). A practical timetable should allow students to have more free time between exams. Thus, the objective of the EPP is to find a feasible timetable while minimizing the number of students having to take consecutive exams. Equation (3) is a variant of the EPP model where q is the number of timeslots per day, $N \geq 0$ is the number of free timeslots between exams and K is a constant representing the maximum timetable length.

That is,

$$
\begin{aligned}
&\text{minimize } u_2 = \tfrac{1}{2} \sum_{k=1}^{|\mathcal{T}|-(N+1)} \sum_{i=1}^{|\mathcal{E}|} \sum_{j=1}^{|\mathcal{E}|} \eta_{ij}\varepsilon_{ik}\varepsilon_{jk+(N+1)}, \forall k \text{ where } k \bmod q \neq 0; \\
&\text{s.t.} \qquad \sum_{k=1}^{|\mathcal{T}|} \sum_{i=1}^{|\mathcal{E}|} \sum_{j=1}^{|\mathcal{E}|} \eta_{ij}\varepsilon_{ik}\varepsilon_{jk} = 0, \\
&\qquad\quad\; |\mathcal{T}| \leq K.
\end{aligned} \tag{3}
$$

The above model represents an Uncapacitated Exam Proximity problem (UEPP) because it does not take into account classroom seating capacity.

3 Previous Methods

The UEPP has been investigated by many researchers. However, the problem formulation and enrolment data are often defined by the environment and requirements of a particular institution. As a result, many methods and algorithms have been proposed to solve particular instances of the UEPP.

This section surveys previous solution methods applied to a collection of publicly available datasets. The datasets used in this work are from Carter et al. [2], Burke et al. [3] and Merlot et al. [4]. They contain actual enrolment data taken from several universities and academic institutions. A common proximity metric has also been defined for the datasets which is a weighted version of (3) with $0 < N \leq 4$ (counting the number of students having 0 to 4 free timeslots between exams). This proximity metric can be expressed using (3) as follows:

$$
f = \frac{\sum_{i=0}^{4} w_{i+1}\ u_2|_{q=|\mathcal{T}|,N=i}}{N_s}, \tag{4}
$$

where w_i are are the weighting factors and N_s is the total student enrolment. In (4), the timeslots are numbered contiguously with no overnight gap. The weighting factors were proposed by Carter et al. [2]. They are: $w_1 = 16$, $w_2 = 8$, $w_3 = 4$, $w_4 = 2$, and $w_5 = 1$. Thus, (4) can be viewed as the average proximity cost of a given timetable and the resulting UEPP is:

$$
\begin{aligned}
&\text{minimize } f = \sum_{i=0}^{4} w_{i+1}\ u_2|_{q=|\mathcal{T}|,N=i} \Big/ N_s; \\
&\text{s.t.} \qquad \sum_{k=1}^{|\mathcal{T}|} \sum_{i=1}^{|\mathcal{E}|} \sum_{j=1}^{|\mathcal{E}|} \eta_{ij}\varepsilon_{ik}\varepsilon_{jk} = 0, \\
&\qquad\quad\; |\mathcal{T}| \leq K.
\end{aligned} \tag{5}
$$

Early solution techniques were derived from sequential graph coloring heuristics. These heuristics attempt to assign each exam to a timeslot according to some ordering schemes. Carter et al. [2] successfully applied a backtracking sequential assignment algorithm to produce feasible timetables for the UEPP. The backtracking feature enables the algorithm to undo previous assignments and thus escape from cul-de-sacs. In all, 40 different strategies have been implemented. The results showed that the effectiveness of the sequential assignment algorithm is related to the ordering scheme and the nature of the datasets. Note that the backtracking sequential assignment is a deterministic algorithm. This means that, for a given dataset and ordering scheme, it will always produce the same timetable.

In Burke et al. [3], an initial pool of timetables is generated by grouping together exams with similar sets of conflicting exams. Then timetables are randomly selected from the pool, weighted by their objective value, and mutated by rescheduling randomly chosen exams. Hill climbing is then applied to the mutated timetable to improve its quality. The process continues with the new pool of timetables. Caramia et al. [5] developed a set of heuristics to tackle the UEPP with excellent results. First, a solution is obtained by a greedy assignment procedure. This procedure selects exams based on a priority scheme which gives high priority to exams with high clashing potential. Next, a spreading heuristic is applied to decrease the proximity penalty of the solution without lengthening the timetable. However, if the spreading heuristic failed to provide any penalty decrease then another heuristic is applied to decrease the proximity penalty by adding one extra timeslot to the solution. These heuristics are reapplied until no further improvement can be found. A perturbation technique is also described in which the search process is restarted by resetting the priority and proximity penalty.

The proximity problem was also investigated by Di Gaspero and Schaerf [6]. Their approach starts with a greedy heuristic to assign timeslots to all exams having no common students. The remaining unassigned exams are distributed randomly to different timeslots. The solution obtained is then improved by a Tabu Search algorithm using a short-term Tabu list with random Tabu tenure. The search neighborhood is defined as the set of exams that can be moved from one timeslot to another without violating the constraints. A further reduction of the neighborhood is obtained by using the subset of exams currently in constraint violation. To improve the proximity cost, Di Gaspero and Schaerf also implemented the shifting penalty mechanism from Gendreau et al. [7]. A Tabu Search algorithm (called OTTABU) with a recency-based and a frequency-based Tabu list was implemented by White and Xie [8]. An initial solution is first generated by a bin-packing heuristic ('largest enrolment first'). If the initial solution is unfeasible, then a Tabu Search is executed to remove all constraint violations using the set of clashing exams as neighborhood. Another Tabu Search is used to improve the quality of the feasible solution. This time, the neighborhood is the set of exams that can be moved from one timeslot to another without causing any clashes. White and Xie also devised an estimation technique for the Tabu tenure based on enrolment, the number of exams having the same pool of students and the number of students taking the same exams. More recently, Paquete and Stutzle [9] considered the UEPP by casting the constraints as part of an aggregated objective function. The search process is prioritized and is realized by the use of a Tabu Search algorithm with a short-term Tabu list and random tenure. The 1-opt neighborhood is defined by the subset of exams with constraint violations.

A three-stage approach using constraint programming, simulated annealing and hill climbing was proposed by Merlot et al. [4]. An initial timetable is generated by constraint programming. The resulting timetable is then improved by a simulated annealing algorithm using the Kempe chain neighborhood [10] and a slow cooling schedule. In the last stage, a hill climbing procedure is applied to further improve the final timetable. The GRASP meta-heuristic [11] was also used to solve the UEPP. Casey and Thompson [12] used a probabilistic version of the sequential assignment algorithm from Carter et al. to realize the construction phase of GRASP. In the improvement phase of GRASP, they ordered the exams according to their contribution to the objective value. Then, for each exam, a timeslot is found such that the objective value is decreased. The construction and improvement phases are restarted with a blank timetable a number of times and the best timetable is kept.

Burke and Newall [13] investigated the effectiveness of the local search approach to improve the quality of timetables. In their work, an adaptive technique is used to modify a given heuristic ordering for the sequential construction of an initial solution. They then compared the average and peak improvement obtained by three different search algorithms: Hill Climbing, Simulated Annealing and an implementation of the Great Deluge algorithm [14]. The reported results indicated that the Adaptive Heuristics and Great Deluge combination provided significant enhancement to the initial solution.

4 Multi-Objective Approach

As shown in section 3, the UEPP is traditionally treated as a single objective combinatorial optimization problem. The timetable length is chosen a priori and is part of the constraint set. In the context of resource planning, it is often desirable to assess the impact of timetable length on the proximity cost. A timetable length versus proximity cost assessment can also provide the planner with compromise solutions to the timetabling problem. Equation (6) is a bi-objective formulation capable of realizing such an assessment:

$$\begin{aligned} \text{minimize } & f_1 = |\mathcal{T}|\,, \\ & f_2 = \sum_{i=0}^{4} w_{i+1}\, u_2|_{q=|\mathcal{T}|,N=i} \Big/ N_s; \\ \text{s.t.} \quad & \sum_{k=1}^{|\mathcal{T}|} \sum_{i=1}^{|\mathcal{E}|} \sum_{j=1}^{|\mathcal{E}|} \eta_{ij}\varepsilon_{ik}\varepsilon_{jk} = 0. \end{aligned} \tag{6}$$

Now the task is to find a feasible timetable while minimizing timetable length and proximity cost simultaneously.

4.1 Hybrid MOEA Implementation

The proposed hybrid MOEA is a Pareto-based optimization heuristic which uses an auxiliary population (archive) to maintain the best non dominated solutions.

Each potential solution in the population is a timetable (feasible or unfeasible). The timetables are assigned a rank based on the objective functions f_1 (timetable length) and f_2 (proximity cost). A special feature in the proposed hybrid MOEA is the substitution of the recombination operator by two local search operators. Local search algorithms are used here to remove constraint violations and to improve the proximity cost. The following pseudo-code explains the operating principle of the algorithm.

`Procedure HMOEA`
$\mathcal{P}^{(t)}$: `population at iteration` t
$\mathcal{R}^{(t)}$: `intermediate population at iteration` t
$\mathrm{L}_1, \mathrm{L}_2$ `: local search operators`
U `: archive update procedure`
M_{α_m}: `uniform mutation operator with mutation rate` α_m
S: `constrained dominance binary tournament operator`
`OUTPUT`
$\mathcal{Q}^{(t)}$: `archive containing non dominated timetables`

`Initialize` $\mathcal{P}^0$ `and` $\mathcal{Q}^0$ `of size` N `with random timetables`
`For each iteration` $t \leftarrow 0, 1, \ldots, I_{\max}$ `do`
   `// Step 1) Apply local searches to the combined population`
   $\mathcal{R}^{(t)} \leftarrow \mathrm{L}_2(\mathrm{L}_1(\{\mathcal{P}^{(t)} \cup \mathcal{Q}^{(t)}\}))$
   `// Step 2) Compute ranking for the resulting timetables`
   $\mathrm{F}(h), \forall h \in \mathcal{R}^{(t)}$
   `// Step 3) Update archive`
   $\mathcal{Q}^{(t+1)} \leftarrow \mathrm{U}(\mathcal{R}^{(t)})$
   `// Step 4) Create new population by mutation and selection`
   $\mathcal{P}^{(t+1)} \leftarrow \mathrm{S}(\mathrm{M}_{\alpha_m}(\mathcal{R}^{(t)}))$
`End for`

Fig. 1. Working principle of the proposed hybrid MOEA

In Fig. 1, the main population at iteration t is denoted by $\mathcal{P}^{(t)}$ and the archive by $\mathcal{Q}^{(t)}$. Both $\mathcal{P}^{(t)}$ and $\mathcal{Q}^{(t)}$ contain N timetables and their size remains constant during the optimization process. The initial timetables are generated randomly without regard to their feasibility. The first local search operator L_1 is used to remove constraint violations, while the second local search operator L_2 is used to decrease the proximity cost. The timetables produced by L_1 and L_2 form a combined intermediate population $\mathcal{R}^{(t)}$ of $2N$ timetables. Next a ranking value is computed for each timetable in $\mathcal{R}^{(t)}$ using Zitzler's Pareto Strength concept [15]. The non dominated timetables are then inserted into the archive using an archive update rule. Finally, each timetable in the intermediate population is mutated with probability α_m. Since there are $2N$ timetables in $\mathcal{R}^{(t)}$, N timetables are discarded from $\mathcal{R}^{(t)}$ using the constrained tournament selection technique [16].

The remaining N timetables form the new population $\mathcal{P}^{(t+1)}$, and the evolution process continues for $I_{\max}$ iterations.

4.2 Population and Archive Initialization

The same initialization procedure is applied to the main population $\mathcal{P}^{(t)}$ and the archive $\mathcal{Q}^{(t)}$. Both $\mathcal{P}^{(t)}$ and $\mathcal{Q}^{(t)}$ can each contain N timetables and are divided into β slots $l_0 > l_1, \ldots, > l_\beta$ representing different timetable lengths. For each slot i, N/β random timetables with length l_i are generated. The number of slots and the range of the timetable length are determined according to the published results available for the datasets. Note that the initialization procedure will also produce unfeasible timetables. These unfeasible timetables with be repaired with the help of local search operators. This is explained in more detail in the next subsection.

4.3 Search Operators $\mathbf{L_1}$ and $\mathbf{L_2}$

In the hybrid MOEA implementation, local search operators are used instead of the traditional genetic recombination operators. This hybridization scheme enables the evolutionary process focus better on the optimization task. Both local searches L_1 and L_2 are in fact Tabu Search algorithms. The search operator L_1 implements a classic Tabu Search using a simple 1-opt neighborhood. This neighborhood is defined by an ordered triple (e, t_i, t_j), where e is an exam in schedule conflict with at least one other exam, and $t_i \neq t_j$ are two different timeslots such that e can be moved from t_i to t_j without creating a new conflict. The idea is to decrease the number of constraint violations for the timetables currently in the main population $\mathcal{P}^{(t)}$ and in the archive $\mathcal{Q}^{(t)}$.

In order to improve the proximity cost of the timetables, the search operator L_2 implements a simplified version of the VND (Variable Neighborhood Descent) meta-heuristic [17]. Two neighborhoods, the Kempe chain interchange [10, 18] and 1-move, are used in L_2 with Tabu Search as the search engine. In the simplified VND, local searches are executed in $n > 1$ neighborhoods sequentially. The initial solution of the current search is the best solution obtained from the previous search. Since there are $n > 1$ neighborhoods involved, it is conjectured that VND has better search space coverage than do single neighborhood search techniques [17]. In our implementation, the Kempe chain interchange neighborhood is used first. Similar to the 1-opt neighborhood, a Kempe chain is defined by an ordered triple (e, t_0, t_1), where exam e is assigned to timeslot t_0 and $t_0 \neq t_1$. However, exam e is now selected by sampling the set of exams. Consider a graph G where the vertices are the exams and an edge exists between vertices e_i and e_j if at least one student is taking exams e_i and e_j. Each vertex in G is labeled with the exam's assigned timeslot, and an edge linking two exams indicates a potential clashing situation. A Kempe chain (e, t_0, t_1) is a connected subgraph induced by a subset of linked exams assigned to timeslots t_0 and t_1. The subset of linked exams must also contain the exam e. In other words, it is the subset of exams reachable from e in the digraph D given by

$$\begin{aligned} V(D) &= \{V_{t_0}\} \cup \{V_{t_1}\}, \\ E(D) &= \left\{(u,w) : (u,w) \in E(G), u \in V_{t_i} \wedge w \in V_{t_{(i+1) \bmod 2}}\right\}, \end{aligned} \tag{7}$$

where $E(G)$ represents the set of edges in graph G and V_{t_i} is the subset of exams assigned to timeslot t_i that are reachable from exam e. Thus, a Kempe chain interchange is the relabeling of each chain vertex in timeslot t_0 to timeslot t_1, and vice versa. This relabeling is conflict-free if the original timetable is also conflict-free. It is also applicable to unfeasible timetables.

```
Operator L₂(H)
f(·): proximity cost
h: current best timetable
INPUT
H: set of timetables in the current population and in the archive,
    H ≡ P^(t) ∪ Q^(t).
OUTPUT
R^(t): combined intermediate population
```

```
For each h ∈ H do,
    while true,
      // Apply Tabu Search with Kemp chain interchange
      // Stopping criterion: N_I iterations
      h' ← TS_Kci(h)
      // h'  is better than h ?
      If f(h') > f(h),
        // No. Apply Tabu Search to h using 1-move neighborhood
        h' ← TS_1-move(h)
        // h' is better than h ?
        If f(h') > f(h),
           // No. Exit While loop and process next timetable
           next
        End If
      End If
      // update current best timetable
      h ← h'
    End While
    R^(t) ← {R^(t)} ∪ {h}
End Do
```

Fig. 2. Search operator L_2 implements a simplified VND meta-heuristic

In the Tabu Search implementation, we choose N_k Kempe chains and apply the best chain as the current move. To sample a chain, we choose two linked exams uniformly without replacement from the set of exams and use their timeslots as t_0 and t_1. One disadvantage of the Kempe chain interchange neighborhood is that the number of useful chains decreases as the search progresses toward a local optimum [10]. To avoid this pitfall, we use another neighborhood to explore the search space. After N_I iterations without improvement by the Kempe chain interchange, we start another Tabu Search using the 1-move neighborhood. To select a move from the 1-move neighborhood, we sample N_m legal moves (moving one exam from its assigned timeslot to another timeslot without creating constraint violations) and the one with the best proximity cost is selected. The initial timetable for the 1-move neighborhood search is the current best timetable.

As shown in Fig. 2, $f(\cdot)$ represents the proximity cost, $\mathrm{TS}_{\mathrm{kci}}$ designates a Tabu Search with the Kempe chain interchange neighborhood and $\mathrm{TS}_{1-\mathrm{move}}$ indicates the one using a 1-move neighborhood. For a given timetable, the search terminates when no further improvement can be obtained by $\mathrm{TS}_{\mathrm{kci}}$ and $\mathrm{TS}_{1-\mathrm{move}}$.

4.4 Ranking Computation

The timetables in the combined intermediate population $\mathcal{R}^{(t)}$ are to be ranked in order to determine their quality relative to the current population. The ranking computation process assigns a numerical value to each timetable according to their dominance performance in the current population [19]. An efficient ranking procedure is the one based on the Pareto Strength concept used in the SPEA-II multi-objective evolutionary algorithm [15]. In this procedure, a timetable's Pareto strength $\mathrm{C}(\cdot)$ is the number of timetables it dominates in the combined intermediate population. That is,

$$\mathrm{C}(h_i) = \left| \left\{ h_j : h_j \in \mathcal{R}^{(t)} \wedge h_i \succ h_j \right\} \right| , \tag{8}$$

where the symbol $\succ$ corresponds to the Pareto dominance relation. For a P objective minimization problem with objective functions $f_1, f_2, \ldots, f_p$, a timetable h_1 is said to dominate another timetable h_2, denoted here by $h_1 \succ h_2$, if and only if

$$\begin{array}{ll} 1. & f_i(h_1) \leq f_i(h_2),\ i = 1, 2, \ldots P; \\ 2. & \exists i \text{ such that } f_i(h_1) < f_i(h_2). \end{array} \tag{9}$$

Using the Pareto strength given by (8), the ranking $\mathrm{F}(\cdot)$ of a timetable h_i is determined by the Pareto strength of its dominators,

$$\mathrm{F}(h_i) = \sum_{h_j \succ h_i, h_j \in \mathcal{R}^{(t)}} \mathrm{C}(h_j). \tag{10}$$

Thus, the ranking of a timetable as given in (10) measures the amount of dominance applied to it by other timetables. In this context, a small ranking value indicates a good quality timetable.

4.5 Archive Update

The purpose of an archive is to memorize all current non dominated timetables. To admit a timetable $h_i \in \mathcal{R}^{(t)}$ into the archive $\mathcal{Q}^{(t)}$, no member of $\mathcal{Q}^{(t)}$ should dominate h_i, that is

$$h_j \not\prec h_i, \forall h_j \in \mathcal{Q}^{(t)}. \quad (11)$$

Equation (11) is the archive admission criterion. By contrast, h_i may dominate some members of $\mathcal{Q}^{(t)}$. In this case, all dominated members are removed and h_i is inserted into $\mathcal{Q}^{(t)}$. Another possible situation arises where h_i and the members of $\mathcal{Q}^{(t)}$ do not dominate each other. Then, $h_i \in \mathcal{R}^{(t)}$ replaces $h_j \in \mathcal{Q}^{(t)}$ if and only if the following conditions are met:

$$\begin{array}{ll} 1. & |h_i| = |h_j|, \\ 2. & \mathrm{F}(h_i) < \mathrm{F}(h_j). \end{array} \quad (12)$$

Thus, a timetable replaces another timetable of same length but with a lower rank.

4.6 Mutation and Selection

The uniform mutation operator M_{α_m} is used in this work to provide diversification in the evolution process. Each exam within a timetable h_i has a mutation probability $\alpha_m = 1/|h_i|$. To mutate a timetable, we assign a random timeslot to the selected exams. The resulting effect is a slight perturbation to the scheduling composition of the timetables. However, this is a destructive process because it can introduce constraint violations into feasible timetables. The search operator L_1 will later be used to repair the unfeasible timetables created by the uniform mutation.

A selection procedure S is executed after all timetables have been mutated. The goal is to select N timetables from the combined intermediate population $\mathcal{Q}^{(t)}$ to create the next population $\mathcal{P}^{(t+1)}$. Since the mutation operator can produce both feasible and unfeasible timetables, the selection procedure must be able to discriminate between them. This is accomplished by the use of the constrained dominance binary tournament [16] to select the timetables. A binary tournament involves two randomly selected timetables. The selected timetables are compared and the winner is inserted into the new population $\mathcal{P}^{(t+1)}$. In order to decide which timetable is the winner, the constrained dominance relation is used [19]. Given two timetables h_1 and h_2 with constraint violations c_1 and c_2, timetable h_1 is said to constraint-dominate h_2, denoted here by $h_1 \succ_{\mathrm{C}} h_2$, if one of the following conditions is met:

$$\begin{array}{ll} 1. & c_1 = 0 \,\text{and}\, c_2 > 0, \quad \text{or} \\ 2. & c_1 > 1, c_2 > 1 \text{ and } c_1 < c_2, \quad \text{or} \\ 3. & c_1 = c_2 \text{ and } h_1 \succ h_2. \end{array} \quad (13)$$

The conditions given by (13) always favor timetables with fewer conflict violations. However, when both timetables have identical conflict violations, the constrained dominance relation is reduced to the simple dominance relation.

5 Experimental Results

The hybrid MOEA described in section 4 was tested on 15 datasets. Table 1 shows the number of exams and the number of students for each dataset. Datasets MEL-F-01 and MEL-F-02 were contributed by Merlot [4]. Dataset NOT-F-94 is by Burke [3]. All other datasets are taken from Carter [2].

Table 1. Datasets characteristics

Dataset	Number of Exams	Number of students
CAR-F-92	543	18419
CAR-S-91	682	16925
EAR-F-83	190	1125
HEC-S-92	81	2823
KFU-S-93	461	5349
LSE-F-91	381	2726
MEL-F-01	521	20656
MEL-S-01	562	19816
NOT-F-94	800	7896
RYE-F-92	486	11483
STA-F-83	139	611
TRE-S-92	261	4360
UTA-S-92	622	21266
UTE-S-92	184	2749
YOR-F-83	181	941

Table 2. Hybrid MOEA parameters and environmental setting

Parameter	Value
Number of runs	5 per dataset
Number of iterations	$I_{\max} = 500$
Number of slots in archive	$\beta = 5$
Population and archive size	$\lvert\mathcal{Q}^{(t)}\rvert = \lvert\mathcal{P}^{(t+1)}\rvert = 80$
Neighborhood sample size	$N_k = N_m = 50$
Number of non improvement iterations	$N_I = 100$
Mutation probability	$1/\lvert h_i\rvert$
Computer	Athlon XP 2.2 GHz, 512 MB RAM
OS	Linux 2.4.2-2
Compiler and optimization level	GNU v2.96, -O3

Table 2 gives the algorithmic parameters and environmental settings used in the experiments. For the VND search operator L_2, N_k neighbors in the Kempe chain interchange neighborhood and N_m neighbors in the 1-move neighborhood were selected uniformly to determine the current best move. The number of selected neighbors N_k and N_m are identified as the 'neighborhood sample size' in Table 2. The range of timetable length (objective function f_1) depends on the datasets. Five timetable lengths centered on published values were retained in the archive. The average and best proximity costs of the non dominated timetables were computed using the weighting factors presented in section 3. The results are detailed in Table 3. It is important to note that no fine-tuning of the hybrid MOEA has been performed and that the same parameters are used for all datasets. Although the numerical results (see Table 3) summarize the overall effectiveness of the hybrid MOEA well, it is often interesting to appreciate the dynamics of the search process. Figure 3 and 4 show the progress of the non dominated timetables in the archive for the dataset YOR-F-83. Eight different timetable lengths are used in the figures to help visualize the non dominated front.

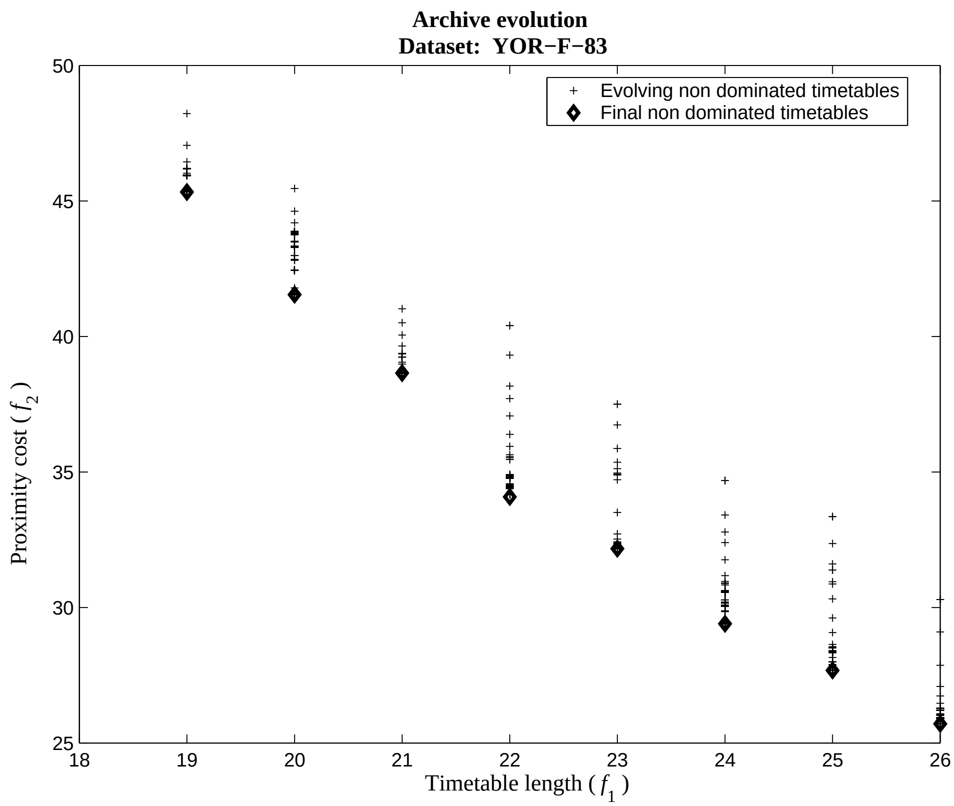

Fig. 3. Evolution of the non dominated timetables in the archive (YOR-F-83)

The effects of the hybrid MOEA can be clearly identified in Figure 4. As the optimization progresses, more and more non dominated timetables of various timetable lengths were admitted to the archive. After 125 iterations, all the

Table 3. Non dominated timetables and their proximity cost (5 runs per dataset)

Dataset		Results					Time (min)
	$\|\mathcal{T}\|$	30	31	32	33	34	
CAR-F-92	Best	4.9	4.5	4.2	4.2	3.9	
	Avg	4.9	4.7	4.3	4.5	4.1	583
	$\|\mathcal{T}\|$	32	33	34	35	36	
CAR-S-91	Best	6.1	5.7	5.4	5.4	5.2	
	Avg	6.2	5.9	5.5	5.5	5.3	816
	$\|\mathcal{T}\|$	23	24	25	26	27	
EAR-F-83	Best	38.0	34.2	31.6	28.8	26.7	
	Avg	39.0	35.6	31.9	29.7	27.5	102
	$\|\mathcal{T}\|$	17	18	19	20	21	
HEC-S-92	Best	12.0	10.4	9.3	8.1	7.3	
	Avg	12.1	10.5	9.3	8.2	7.5	27
	$\|\mathcal{T}\|$	19	20	21	22	23	
KFU-S-93	Best	15.8	14.3	12.1	11.0	10.0	
	Avg	16.2	14.4	12.8	11.6	10.3	165
	$\|\mathcal{T}\|$	17	18	19	20	21	
LSE-F-91	Best	12.3	11.3	9.7	8.5	7.7	
	Avg	12.6	11.5	10.1	9.3	8.1	92
	$\|\mathcal{T}\|$	26	27	28	29	30	
MEL-F-01	Best	3.6	3.2	2.8	2.9	2.4	
	Avg	3.7	3.2	2.9	2.9	2.5	269
	$\|\mathcal{T}\|$	29	30	31	32	33	
MEL-S-01	Best	2.8	2.6	2.4	2.3	2.0	
	Avg	3.0	2.7	2.5	2.3	2.1	281
	$\|\mathcal{T}\|$	22	23	24	25	26	
NOT-F-94	Best	7.8	6.9	6.2	5.7	5.0	
	Avg	8.1	7.2	6.6	5.9	5.1	289
	$\|\mathcal{T}\|$	22	23	24	25	26	
RYE-F-92	Best	9.8	8.8	7.8	7.0	7.0	
	Avg	10.1	9.1	8.1	7.2	7.3	218
	$\|\mathcal{T}\|$	13	14	15	16	17	
STA-F-83	Best	157.0	140.2	125.2	112.7	101.4	
	Avg	157.1	140.4	126.0	113.2	101.6	26
	$\|\mathcal{T}\|$	21	22	23	24	25	
TRE-S-92	Best	10.3	9.4	8.6	7.9	7.2	
	Avg	10.5	9.4	8.8	8.1	7.3	126
	$\|\mathcal{T}\|$	34	35	36	37	38	
UTA-S-92	Best	3.7	3.5	3.3	3.2	3.2	
	Avg	3.9	3.6	3.4	3.2	3.2	265
	$\|\mathcal{T}\|$	10	11	12	13	14	
UTE-S-92	Best	25.3	20.7	16.8	13.9	11.5	
	Avg	25.5	21.2	17.1	14.2	11.6	25
	$\|\mathcal{T}\|$	19	20	21	22	23	
YOR-F-83	Best	44.6	40.6	36.4	33.8	31.6	
	Avg	45.6	41.0	37.6	34.5	31.9	89

empty slots in the archive were occupied with non dominated timetables. Simultaneously, the hybrid MOEA tries to lower their proximity cost. The lowering of the proximity cost can be observed by noticing the vertical displacement of the points in Fig. 4 and by the trace left by the non dominated timetables in Fig. 3.

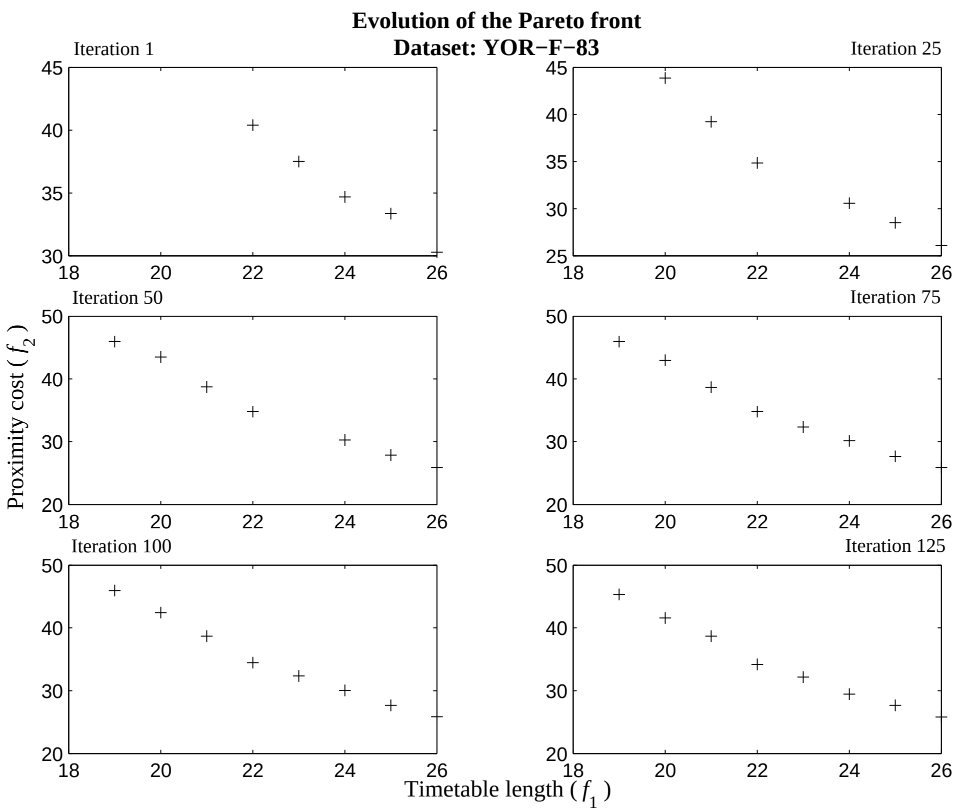

Fig. 4. Non dominated timetables at different iterations (YOR-F-83)

A comparison with other published results was also conducted in order to asses the effectiveness of the hybrid MOEA against other optimization methods. Since most published results for the UEPP are based on the single-objective approach with a fixed timetable length, the performance of the hybrid MOEA will also be shown for that particular timetable length.

From the results given in Table 4, the hybrid MOEA obtained the best score in three datasets. It is worth mentioning that the hybrid MOEA also achieved a second-best position in 6 of the 15 datasets. In summary, the proposed multi-objective evolutionary algorithm was able to produce high-quality timetables in comparison to other optimization methods.

Table 4. Comparison with other methods

Dataset		hMOEA	Car	Whi	Di1	Cara	Bur	Mer	Di2	Paq	Cas
CAR-F-92	Best	4.2	6.2	-	5.2	6.0	**4.0**	4.3	-	-	4.4
32 timeslots	Avg	4.4	7.0	4.7	5.6	-	**4.1**	4.4	-	-	4.7
CAR-s-91	Best	5.4	7.1	-	6.2	6.6	**4.6**	5.1	-	-	5.4
35 timeslots	Avg	5.5	8.4	-	6.5	-	**4.7**	5.2	-	-	5.6
EAR-F-83	Best	34.2	36.4	-	45.7	**29.3**	36.1	35.1	39.4	40.5	34.8
24 timeslots	Avg	35.6	40.9	-	46.7	-	37.1	**35.4**	43.9	45.8	35.0
HEC-S-92	Best	10.4	10.6	-	12.4	**9.2**	11.3	10.6	10.9	10.8	10.8
18 timeslots	Avg	**10.5**	15.0	-	12.6	-	11.5	10.7	11.0	12.0	10.9
KFU-S-93	Best	14.3	14.0	-	18.0	13.8	13.7	**13.5**	-	16.5	14.1
20 timeslots	Avg	14.4	18.8	-	19.5	-	**13.9**	14.0	-	18.3	14.3
LSE-F-91	Best	11.3	10.5	-	15.5	**9.6**	10.6	10.5	12.6	13.2	14.7
18 timeslots	Avg	11.5	12.4	-	15.9	-	**10.8**	11.0	13.0	15.5	15.0
MEL-F-01	Best	**2.8**	-	-	-	-	-	2.9	-	-	-
28 timeslots	Avg	**2.9**	-	-	-	-	-	3.0	-	-	-
MEL-S-01	Best	**2.4**	-	-	-	-	-	2.5	-	-	-
31 timeslots	Avg	**2.5**	-	-	-	-	-	2.5	-	-	-
NOT-F-94	Best	**6.9**	-	-	-	-	-	7.0	-	-	-
23 timeslots	Avg	7.2	-	-	-	-	-	**7.1**	-	-	-
RYE-F-92	Best	8.8	7.3	-	-	**6.8**	-	8.4	-	-	-
23 timeslots	Avg	9.1	**8.7**	-	-	-	-	**8.7**	-	-	-
STA-F-83	Best	157.0	161.5	-	160.8	158.2	168.3	157.3	157.4	158.1	**134.9**
13 timeslots	Avg	157.1	167.1	-	166.8	-	168.7	157.4	157.7	159.3	**135.1**
TRE-S-92	Best	8.6	9.6	-	10.0	9.4	**8.2**	8.4	-	9.3	8.7
23 timeslots	Avg	8.8	10.8	-	10.5	-	**8.4**	8.6	-	10.2	8.8
UTA-S-92	Best	3.5	3.5	-	4.2	3.5	**3.2**	3.5	-	-	-
35 timeslots	Avg	3.6	4.8	4.0	4.5	-	**3.2**	3.6	-	-	-
UTE-S-92	Best	25.3	25.8	-	29.0	**24.4**	25.5	25.1	-	27.8	25.4
10 timeslots	Avg	25.5	30.8	-	31.3	-	25.8	**25.2**	-	29.4	25.5
YOR-F-83	Best	36.4	36.4	-	41.0	**36.2**	36.8	37.4	39.7	38.9	37.5
21 timeslots	Avg	37.5	45.6	-	42.1	-	**37.3**	37.9	41.7	41.7	38.1

Car: Carter et al. [2], Whi: White and Xie [8], Di1: Di Gaspero and Shaerf [6]
Cara: Caramia et al. [5], Bur: Burke and Newall [13], Mer: Merlot et al. [4],
Di2: Di Gaspero [20], Paq: Paquete and Stutzle [9], Cas: Casey and Thompson [12]

6 Conclusions

The hybrid MOEA performed well in comparison to 9 other methods. Most published results for the UEPP using these publicly available datasets are based on the single-objective approach. A systematic comparison of the non dominated sets was not possible. In spite of this, the hybrid MOEA demonstrated its effectiveness by producing timetables ranking first and second in 9 of the 15 datasets without special fine-tuning. Moreover, the MOEA approach was able to generate non dominated timetables for a range of timetable lengths as alternative solutions. A contribution of this work is the use of a single framework to cover all the necessary timetabling steps:

- Initialization: Random initialization of the population and the archive;
- Search (exploitation): Variable Neighborhood Search operator and the ranking of the timetable by Pareto Strength;
- Search (exploration): Destructive uniform mutation with repair operator to obtain feasible timetables;
- Solution selection: Archive admission and non dominated timetable replacement criteria.

All these steps are fully integrated into the hybrid MOEA presented in this paper.

Acknowledgements. The authors would like to thank the referees for their constructive and helpful advise.

References

1. Radcliffe, N.J., Surry, P.D.: Formal memetic algorithms. In: Proceedings of the AISB Workshop on Evolutionary Computing, Apr 11-13 1994. Lecture Notes in Computer Science 865, Leeds, UK, Springer-Verlag (1994) 1–16
2. Carter, M.W., Laporte, G., Yan Lee, S.: Examination timetabling: algorithmic strategies and applications. Journal of the Operational Research Society **47** (1996) 373–383
3. Burke, E., Newall, J., Weare, R.: Memetic algorithm for university exam timetabling. In: 1st International Conference on Practice and Theory of Automated Timetabling, Aug 29-Sep 1 1995. PATAT 1995. Lecture Notes in Computer Science 1153, Springer-Verlag (1996) 241–250
4. Merlot, L., Boland, N., Hughes, B., Stuckey, P.: A hybrid algorithm for the examination timetabling problem. In: 4th International Conference on the Practice and Theory of Automated Timetabling, 21-23 Aug. 2002. PATAT 2002. Selected Papers. Lecture Notes in Computer Science 2740, Gent, Belgium, Springer-Verlag (2003) 207–31
5. Caramia, M., Dell'Olmo, P., Italiano, G.: New algorithms for examination timetabling. In: 4th International Workshop on Algorithm Engineering, 5-8 Sept. 2000. WAE 2004. Lecture Notes in Computer Science 1982, Saarbrucken, Germany, Springer-Verlag (2001) 230–41

6. Di Gaspero, L., Schaerf, A.: Tabu search techniques for examination timetabling. In: 3rd International Conference on the Practice and Theory of Automated Timetabling, 16-18 Aug. 2000. PATAT 2000. Selected Papers. Lecture Notes in Computer Science 2079), Konstanz, Germany, Springer-Verlag (2001) 104–17
7. Gendreau, M., Hertz, A., Laporte, G.: Tabu search heuristic for the vehicle routing problem. Management Science **40** (1994) 1276–1290
8. White, G., Xie, B.: Examination timetables and tabu search with longer-term memory. In: 3rd International Conference on the Practice and Theory of Automated Timetabling, 16-18 Aug. 2000. PATAT 2000. Selected Papers. Lecture Notes in Computer Science 2079), Konstanz, Germany, Springer-Verlag (2000) 85–103
9. Paquete, L., Stutzle, T.: Empirical analysis of tabu search for the lexicographic optimization of the examination timetabling problem. In: 4th International Conference on the Practice and Theory of Automated Timetabling, 21-23 Aug. 2002. PATAT 2002. Selected Papers. Lecture Notes in Computer Science 2740, Gent, Belgium, Springer-Verlag (2003) 413–420
10. Morgenstern, C., Shapiro, H.: Coloration neighborhood structures for general graph coloring. In: Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms, San Francisco, California, USA, Society for Industrial and Applied Mathematics (1990) 226–235
11. Feo, T.A., Resende, M.G.: Greedy randomized adaptive search procedures. Journal of Global Optimization **6** (1995) 109–133
12. Casey, S., Thompson, J.: Grasping the examination scheduling problem. In: 4th International Conference on the Practice and Theory of Automated Timetabling, 21-23 Aug. 2002. PATAT 2002. Selected Revised Papers. Lecture Notes in Computer Science 2740, Gent, Belgium, Springer-Verlag (2003) 232–44
13. Burke, E., Newall, J.: Enhancing timetable solutions with local search methods. In: 4th International Conference on the Practice and Theory of Automated Timetabling, 21-23 Aug 2002. PATAT 2002. Selected Papers. Lecture Notes in Computer Science 2740, Gent, Belgium, Springer-Verlag (2003) 195–206
14. Dueck, G.: New optimization heuristics: The great deluge algorithm and the record-to-record travel. Journal of Computational Physics **104** (1993) 86–92
15. Zitzler, E., Laumanns, M., Thiele, L.: Spea2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In: Evolutionary Methods for Design Optimisation and Control, Barcelona, Spain, CIMNE (2002) 95–100
16. Osyczka, A., Krenich, S.: New constraint tournament selection method for multicriteria optimization using genetic algorithm. In: Proceedings of the IEEE Congress on Evolutionary Computation, Jul 16-Jul 19 2000. Volume 1 of CEC2000., California, USA, IEEE, Piscataway, NJ (2000) 501–508
17. Mladenovic, N., Hansen, P.: Variable neighborhood search. Computers and Operations Research **24** (1997) 1097–1100
18. Thompson, J.M., Dowsland, K.A.: Robust simulated annealing based examination timetabling system. Computers and Operations Research **25** (1998) 637–648
19. Deb, K.A.: Multi-Objective Optimization Using Evolutionary Algorithms. John Wiley and Sons, Hoboken (2001)
20. Di Gaspero, L.: Recolor, shake and kick: A recipe for the examination timetabling problem. In: 4th International Conference on the Practice and Theory of Automated Timetabling, 21-23 Aug. 2002. PATAT 2002. Selected Revised Papers. Lecture Notes in Computer Science 2740, Gent, Belgium, Springer-Verlag (2003) 404–407