A Combination of Metaheuristic Components based on Harmony Search for The Uncapacitated Examination Timetabling

Mohammed Azmi Al-Betar · Ahamad Tajudin Khader · J. Joshua Thomas

Abstract In this paper, we investigate the effectiveness of combining the key components of the basic metaheuristic methods on the quality of solutions produced for Uncapacitated Examination Timetabling Problem (UETP). These components are *recombination*, *randomness*, and *neighbourhood structures*. The Harmony search Algorithm (HSA) is used to simulate different combinations of these components. It has three main components: Memory Consideration analogous to recombination, Random Consideration analogous to randomness and Pitch Adjustment analogous to neighbourhood structures. The combinations among metaheuristic components are evaluated using 17 different scenarios each of which reflects a combination of one, two or three components. The results show that the system that combines the three components (recombination, randomness, and neighbourhood structures) provides the best results. Furthermore, the best results obtained from the convergence scenarios were compared with 22 other methods that used a *de facto* dataset defined by Carter et al. (1996) for UETP. The results excel those produced by the previous methods in 2 out 12 datasets.

Keywords Examination Timetabling · Harmony Search Algorithm · Metaheuristic-based methods · Exploration · Exploitation

1 Introduction

Problem Background. Examination timetabling is a taxing administrative task that is often repeated in academic institutions every course session. It is the process of assigning a set of exams, each taken by a set of students, to a set of timeslots (and rooms) according to a set of constraints. Two classes of constraints appeared in the literature: *hard* and *soft*. The hard constraints must be satisfied to obtain a *feasible* solution while soft constraints are desired but not essential. Although soft constraints can be violated, the *quality* of solution is often evaluated against soft constraints ful-

Mohammed Azmi Al-Betar (✉) · Ahamad Tajudin Khader · J. Joshua Thomas
School of Computer Sciences, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia
E-mail: mohbetar@cs.usm.my, tajudin@cs.usm.my, joshopever@yahoo.com

fillment. The basic objective is to obtain a feasible solution with the least number of soft constraint violations.

In computing terms, examination timetabling is a hard combinatorial optimisation problem which belongs to an NP-hard class for most of its variations. This problem normally has a huge and rugged search space with considerable local optimal solutions (Ochoa et al. 2009). This makes it hard to lend itself to be tackled using classical methods.

Previous Methods. An examination timetabling problem may be capacitated or uncapacitated (Pillay and Banzhaf 2009). Uncapacitated Examination Timetabling Problem (UETP) is addressed in this paper. Over the last five decades, the Artificial Intelligence and Operational Research communities have been developing a wide variety of *approximation* methods to tackle UETP. An extensive and exhaustive summary of these methods has been provided by Qu et al. (2009b). Earlier developments were based on graph coloring heuristic methods that assigned exams to timeslots, one by one, based on a difficulty level. A backtracking method is often used with these methods as a recovery approach to timetable with unscheduled exams. The main research of UETP was initiated by Carter et al. (1996), who employed several graph coloring heuristic methods to UETP. Other investigations employing graph coloring heuristic methods for UETP include Burke and Newall (2004); Asmuni et al. (2005, 2009).

One of the most notable achievements for solving UETP has been the emergence of metaheuristic-based methods. Metaheuristic-based methods are classified into local search-based and population-based methods (Blum and Roli 2003). Local search-based methods (e.g., Hill climbing, Simulated annealing, Tabu Search) consider one solution at a time. The solution iteratively undergoes changes guided by an objective function until a stationary point near the initial solution is reached. Several local search-based methods that are tailored to UETP had been reported (Di Gaspero and Schaerf 2002; Di Gaspero 2002; Paquete and Stutzle 2003; Burke and Newall 2003; Casey and Thompson 2003; Merlot et al. 2003; Burke et al. 2004; Yang and Petrovic 2005; Burke et al. 2006; Abdullah et al. 2007). On the other hand, population-based methods (e.g., Genetic Algorithm, Ant Colony Optimisation, Artificial Immune System, Harmony Search Algorithm) consider a population of random solutions at a time. The characteristics of the current population are iteratively recombined to generate a new one. Some population-based methods tailored to UETP has been reported (Cote et al. 2005; Eley 2007).

Local search-based methods are able to explore the search space and find a local optimal solution more structurally, precisely and quickly than population-based methods. However, they go through a trajectory without doing a wider scan of the entire search space (Blum and Roli 2003). On the other hand, population-based methods are able to explore several search space regions at the same time. However, they are unable to find a precise local optimal solution to which they can converge (Fesanghary et al. 2008).

Hyper-heuristic methods have also been proposed for UETP. Normally, they have a high-level heuristic to select from a set of low-level heuristics. Several applications of hyper-heuristic for UETP have been reported (Kendall and Hussin 2005; Burke et al. 2007; Qu and Burke 2009; Pillay and Banzhaf 2009; Qu et al. 2009a).

The best solutions obtained for UETP were provided by metaheuristic-based methods, more precisely, by the hybrid metaheuristics (see Qu et al. 2009b). However, in-depth investigation into the main components of these methods that lead to these

Table 1 Examples of intrinsic components of the basic metaheuristic-based methods

Metaheuristic method	Neighborhood structures	Recombination	Randomness
Genetic Algorithm	–	crossover	mutation
Memetic Algorithm	Hill-Climbing optimizer	crossover	mutation
Harmony Search Algorithm	pitch adjustment	memory consideration	random considration
Hill Climbing	move, swap, exchange	–	–
Simulated Annealing	move, swap, kempe-chain	–	cooling schedule
Tabu Search	shake, kicker, s-chain	–	short term memory & aspiration criterion
Large Neighbourhood Structure	move, swap, cyclic-exchange, etc.	–	–

successful outcomes is still lacking. In this paper, a preliminary investigation of the combinations of the basic metaheuristic components,which includes recombination, randomness, and neighbourhood structures are studied.

Metaheuristic Components. The common component among local search-based methods has been the *neighborhood structures* which are able to explore the search space using one or more local changes in the current solution. On the other hand, the common component among population-based methods has been the *recombination*. Recombination exploits the characteristics of the current population in the process of producing a new population. Both local search-based and population-based methods may have a *randomness* component to diversify the search when and if necessary. Some examples of the three metaheuristic components are provided in Table 1. More metaheuristic components are provided by (Blum and Roli 2003).

The key research issue in applying metaheuristic or hybrid metaheuristic method to any combinatorial optimisation problem is to strike a balance between *exploration* and *exploitation* during the search (Qu et al. 2009b). Note that, during the *exploration* stage, the search is encouraged to explore the not-yet-visited search space regions when necessary. While during the *exploitation* stage, the search concentrates on the the already-visited search space regions (Blum and Roli 2003). To put it simply, exploration comes from an unguided search while exploitation comes from a guided search carried out by an objective function of the current solution(s). However, in order to a establish balance between exploration and exploitation, parameter settings (tuning or adaptation) have to be studied. Naturally, the parameter values guide the components of metaheuristics and can be classified into 'exploration consideration' components and 'exploitation consideration' components:

– ***Exploration consideration***. The components that are concerned with exploration rather than exploitation during the search (for example, *mutation* component in Genetic Algorithm).

– ***Exploitation consideration***. The components that are concerned with exploitation rather than exploration during the search (for example, *neighbourhood structures* guided by objective function in Hill climbing and *crossover* guided by the objective functions of the current population in Genetic Algorithm).

To elaborate, we can classify the metaheuristic components based on the source of improvement into the following three types:

1. **Local improvements**: The improvements that result from the *local* changes on the current solution. The main components that are responsible for this type of improvement is *neighbourhood structures*.
2. **Global improvements**: The improvements that *globally* result from recombining the characteristics of the current solutions. The main component responsible for this type of improvement is *recombination*.
3. **Random improvements**: The improvements that *randomly* result from exploring the search space using an unguided strategy. The main component responsible for this type of improvement is *randomness*.

Harmony Search Algorithm. In this study, Harmony Search Algorithm (HSA) is tailored to investigate the effectiveness of combining the metaheuristic components. HSA is a new metaheuristic population-based method inspired by the musical improvisation process (Geem et al. 2001). It has been successfully applied to a wide variety of optimisation problems (Ingram and Zhang 2009) including University Course Timetabling Problem (UCTP) (Al-Betar et al. 2008; Al-Betar and Khader 2009; Al-Betar et al. 2010). HSA is an iterative improvement method initiated with a number of provisional solutions stored in the 'Harmony Memory (HM)'. At each iteration, a new solution called 'new harmony' is generated based on three components: (i) 'Memory Consideration' which makes use of the characteristics of the solutions in HM; (ii) 'Random Consideration' which is used to diversify the new harmony, and (iii) 'Pitch Adjustment', analogous to neighbourhood structures [1]. A new harmony is then evaluated using an objective function and it replaces the worst harmony stored in HM. This process is repeated until a stop criterion is met.

Paper Contributions. The objectives of this study are as follows: (i) tailoring HSA for the UETP, (ii) investigating the effectiveness of combining the meta-heuristic components for producing high quality solutions to UETP.

Results. The HSA results are compared with other results produced by 22 published methods using a *de facto* standard datasets defined by Carter et al. (1996). Although other datasets exist (See *de jure* standard datasets defined in ITC-2007 [2] (McCollum et al. 2009)), the Carter dataset provides a suitable wide range of comparative methods which the proposed method can be evaluated against.

Paper Organization. In order to present a self-explanatory paper, The UETP is described in section. 2 . The way of tailoring HSA toward UETP is proposed in section. 3. A comparative evaluation and empirical study of combining meta-heuristic components are presented in section. 4. The paper concludes with possible research directions described in section. 5.

[1] neighbourhood structures refer to the *move* operators in local-search based methods, such as, move one exam from timeslot to another, swap the timesolts of two exams, etc.

[2] Second International Timetabling Competition (http://www.cs.qub.ac.uk/itc2007/)

2 Problem description

2.1 Problem definition

The UETP variation is concerned with assigning a set of exams, each taken by a set of students, to a set of timeslots with respect to hard (H1) and soft constraint (S1).

- **H1: Exam clash.** No student can sit for two exams at the same time.
- **S1: Exams spread out.** The exams taken by the same student should be spread out across a timetable.

In UETP, the main objective is to minimise the proximity cost function of soft constraint violations in a feasible timetable. The proximity cost function divides the penalty of soft constraint violations by the total number of students. This function will be described formally in the next section.

2.2 Problem formulation

The notation for UETP formulation is given in Table 2. A timetable solution is represented by a vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_N)$ of exams, where x_i is timeslot, $t \in \mathcal{T}$, for exam, $i \in \mathcal{E}$.

The proximity cost function f($\boldsymbol{x}$) for The UETP is formulated in Eq.(1)

$$\min \qquad f(\boldsymbol{x}) = \frac{1}{M} \times \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} c_{i,j} \times a_{i,j} \tag{1}$$

Where $c_{i,j}$ element contains the total number of students sharing exam i and exam j, $a_{i,j}$ element contains the penalty value made based on the distance between exam i and exam j. This provides the *quality* of a solution in terms of how well the exams are spread. Note that the hard constraint H1 must be satisfied in the timetable $\boldsymbol{x}$ such that

$$x_i \neq x_j \qquad \forall x_i, x_j \in \boldsymbol{x} \wedge c_{i,j} \geq 1$$

The value of the proximity cost function $f(x)$ is referred to as the Penalty Value (PV) of a feasible timetable.

3 Proposed Method

The concepts of Harmony Search Algorithm (HSA) are described within the context of creating a pleasing harmony within a musical context (Lee and Geem 2004, 2005; Lee et al. 2005). Table 3 shows the relationship between the UETP terms and optimisation terms in the musical context. In musical improvisation, a group of musicians improvise the pitches of their musical instruments. From repeated practice sessions, a pleasing harmony as decided by their own audio-aesthetic standard is sought. Similarly, in the optimisation context, a set of decision variables is assigned with values. From repeated iterations, an optimal solution as decided by an objective function is sought. In the timetabling process, a set of exams is scheduled with timeslots. From repeated

Table 2 Notations used to formalise the UETP

Symbol	Description
N	The number of exams.
P	The number of timeslots.
M	The number of students.
$\mathcal{E}$	Set of exams, $\mathcal{E} = \{1, 2, \ldots, N\}$.
$\mathcal{S}$	Set of students, $\mathcal{S} = \{1, 2, \ldots, M\}$.
$\mathcal{T}$	Set of timeslots, $\mathcal{T} = \{1, 2, \ldots, P\}$.
$\boldsymbol{x}$	A timetable is represented by $\boldsymbol{x} = (x_1, x_2, \ldots, x_N)$.
x_i	the timeslot of exam i.
$c_{i,j}$	Conflict matrix element: total number of students sharing exam i and exam j. $$c_{i,j} = \sum_{k=1}^{M} u_{k,i} \times u_{k,j} \quad \forall i, j \in \mathcal{E}$$
$a_{i,j}$	Proximity coefficient matrix element: whether the timetable $\boldsymbol{x}$ is penalized based on the distance between timeslot of exam i in timeslot of exam j. $$a_{i,j} = \begin{cases} 2^{5-\lvert x_i - x_j \rvert} & \text{if } 1 \leq \lvert x_i - x_j \rvert \leq 5. \\ 0 & \text{otherwise.} \end{cases}$$
$u_{i,j}$	Student-exam matrix element: whether student s_i is sitting for exam j $$u_{i,j} = \begin{cases} 1 & \text{if student } i \text{ sitting in exam } j \\ 0 & \text{otherwise.} \end{cases}$$

Table 3 The UETP and Optimisation terms in the musical context

Musical		Optimisation		UETP
Improvisation	$\leftrightarrow$	Generation	$\leftrightarrow$	Scheduling
Harmony	$\leftrightarrow$	Solution vector	$\leftrightarrow$	Timetabling solution
Musician	$\leftrightarrow$	Decision variable	$\leftrightarrow$	Exam
Pitch	$\leftrightarrow$	Value	$\leftrightarrow$	Timeslot
Pitch Range	$\leftrightarrow$	Value Range	$\leftrightarrow$	Feasible timeslots
Esthetic standard	$\leftrightarrow$	Objective function	$\leftrightarrow$	Proximity cost function
Practice	$\leftrightarrow$	Iteration	$\leftrightarrow$	Iteration
Pleasing harmony	$\leftrightarrow$	Optimal solution	$\leftrightarrow$	Feasible timetable with the least number of soft constraint violations

iterations, a feasible timetable with the least weight of soft constraint violations as decided by a proximity cost function is sought.

Algorithm 1 shows the pseudo-code of the HSA applied for UETP with five main steps that will be described below:

Step 1. Initialize the problem and HSA parameters.

The solution is represented by a vector, $\boldsymbol{x} = (x_1, x_2, x_3, \ldots, x_N)$, of exams. The value of each exam x_i is a timeslot. The possible range of each exam is the possible feasible timeslots. The proximity cost function is utilized in HSA as formalised in Eq.(1).

Algorithm 1 The basic harmony search algorithm

STEP1 Initialize the problem and HSA parameters

1: Input data instance of the UETP.
2: Utilize UETP pacific knowledge: objective function and solution representation.
3: Set the HSA parameters (HMCR, PAR, NI, HMS).

STEP2 Initialise the harmony memory

1: Construct feasible timetables based on Saturation Degree (SD) stored in harmony memory, $\mathbf{HM} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^{\mathrm{HMS}}\}$
2: Recognise the worst vector in **HM**,
$\boldsymbol{x}^{\mathrm{worst}} \in \{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^{\mathrm{HMS}}\}$ where $f(\boldsymbol{x}^{\mathrm{worst}}) \geq f(\boldsymbol{x}^j) \wedge j \in \{1, \ldots, \mathrm{HMS}\}$

STEP3 Improvise a new harmony

1: $\boldsymbol{x}' = \phi$ // new harmony vector
2: **for** $J = 1, \cdots, N$ **do**
3: $i \leftarrow$ Saturation-Degree ($\boldsymbol{x}'$)
4: **if** $(U(0,1) \leq \mathrm{HMCR})$ **then**
5: $x'_i \in \mathcal{Q}_i$ $\{\mathcal{Q}_i = \{x_i^j | t_{i,x_i^j} = 1 \wedge j \in 1, \ldots, \mathrm{HMS}\}\}$
6: **if** $(\mathcal{Q}_i = \phi)$ **then**
7: $x'_i \in \mathcal{X}_i$ $\{$ $\mathcal{X}_i = \{d | t_{i,d} = 1 \wedge d \in 1, \ldots, P\}$ $\}$
8: **if** $(\mathcal{X}_i = \phi)$ **then**
9: GOTO 1 { Restart }
10: **end if**
11: **end if**
12: $p \leftarrow U(0,1)$ $\{U(0,1)$ Uniform generator number between 0 and 1$\}$
13: **if** $(p \leq \mathrm{PAR1})$ **then**
14: Pitch adjustment Single-move (x'_i)
15: **else if** $(p \leq \mathrm{PAR2})$ **then**
16: Pitch adjustment Swap-timeslot (x'_i)
17: **else if** $(p \leq \mathrm{PAR3})$ **then**
18: Pitch adjustment Kempe-chain (x'_i)
19: **end if**
20: **else**
21: $x'_i \in \mathcal{X}_i$ $\{$ $\mathcal{X}_i = \{d | t_{i,d} = 1 \wedge d \in 1, \ldots, P\}$ $\}$
22: **if** $(\mathcal{X}_i = \phi)$ **then**
23: GOTO 1 { Restart }
24: **end if**
25: **end if**
26: **end for**

STEP4 Update the harmony memory

1: **if** $(f(\boldsymbol{x}') < f(\boldsymbol{x}^{\mathrm{worst}}))$ **then**
2: Include $\boldsymbol{x}'$ to the **HM**.
3: Exclude $\boldsymbol{x}^{\mathrm{worst}}$ from **HM**.
4: **end if**

STEP5 Check the stop criterion

1: **while** (not termination criterion is specified by NI) **do**
2: Repeat **STEP3** and **STEP4**
3: **end while**

The parameters of the HSA required to solve the UETP are also set in this step:

1. The Harmony Memory Consideration Rate (HMCR), used in the improvisation process to determine whether the value of a decision variable is to be selected from the solutions stored in the Harmony Memory (HM).
2. The Harmony Memory Size (HMS) is similar to the population size in Genetic Algorithm.

3. The Pitch Adjustment Rate (PAR), decides whether the decision variables are to be adjusted to a neighbouring value.
4. The Number of Improvisations (NI) corresponds to the number of iterations.

These parameters will be explained in more detail in the next steps.

Step 2. Initialize the harmony memory.

The harmony memory (HM) is an augmented matrix of size $N \times \text{HMS}$ which contains sets of solution vectors determined by HMS (see Eq.2). In this study, the feasible search space regions is only explored where the HM is initialized with random feasible timetables using Saturation Degree (SD) (Brélaz 1979). The SD was widely used to construct an initial solution for UETP. In SD, the exam that has the least number of valid timeslots in the partial timetable is timetabled first.

$$\text{HM} = \begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_N^1 \\ x_1^2 & x_2^2 & \cdots & x_N^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{\text{HMS}} & x_2^{\text{HMS}} & \cdots & x_N^{\text{HMS}} \end{bmatrix} \tag{2}$$

Algorithm 2 provides a high level schematic pseudo-code of building HM solution using SD. Note that the objective function of each timetable in HM is calculated. The solutions in HM are sorted in ascending order in terms of their objective function values, such as, $f(\boldsymbol{x}^1) \leq f(\boldsymbol{x}^2) \leq \ldots \leq f(\boldsymbol{x}^{\text{HMS}})$.

Algorithm 2 Schematic pseudo-code of building HM solutions

```
for j = 1, ··· , HMS do
    x^j = φ
    k = 1
    while (k < N) do
        i= SaturationDegree(x^i)
        x_i^j = h {h ∈ 1 ... P ∧ h is feasible for x_i^j}
    end while
    calculate f(x^j)
    store x^j in the HM
end for
```

Step 3. Improvise a new harmony.

This is the main step in HSA for iterating towards an optimal solution in which this process called 'improvisation process'. In this step, the HSA will construct (or *improvise*) a *new harmony* vector (timetabling solution) from scratch, $\boldsymbol{x}' = (x'_1, x'_2, \cdots, x'_N)$, based on three operators (or components): (i) memory consideration, (ii) random consideration, and (iii) pitch adjustment.

In the timetabling domain, the process of constructing a feasible timetable (in our case *new harmony*), $\boldsymbol{x}' = (x'_1, x'_2, \cdots, x'_N)$, from scratch often requires an ordering

mechanism (Asmuni et al. 2009). As such, to preserve the feasibility during the improvisation process, the idea of Saturation Degree (SD) has been adopted for ordering the exams as follows: exam i that has the least feasible timeslots to be timetabled in the new harmony is selected first.

Formally let Trajectory matrix ($\mathbf{T}$) of size $N \times P$ contain binary elements, i.e., $(t_{i,j})$, which are assigned as follows:

$$t_{i,j} \leftarrow \begin{cases} 1 & \text{if exam } i \text{ can be feasibly assigned with timeslot } j \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Let $\wp_k = \sum_{j=1}^{P} t_{k,j}$ be the total number of available timeslots for each exam k to be timetabled in $\boldsymbol{x}'$. SD iteratively selects exam i to be assigned in $\boldsymbol{x}'$ where:

$$i = \arg \min_{k=1 \ldots N} \wp_k \tag{4}$$

Definition 1 *Exam i can be feasibly assigned with timeslot j if and only if $x'_k \neq j \quad , \forall x'_k \in \boldsymbol{x}' \wedge c_{i,k} \neq 0 \wedge k \in \mathcal{E}$.*

Memory consideration. Every exam i, selected by SD to be assigned with a timeslot x'_i, selects a feasible timeslot from corresponding timeslots, $x'_i \in \{x_i^1, x_i^2, \ldots, x_i^{HMS}\}$, stored in HM vectors with probability (w.p.) HMCR where $0 \leq \text{HMCR} \leq 1$. The operation of this operator is similar to the recombination operator in other population-based methods and is a good source of exploitation (Yang 2009).

Formally, let exam i be selected by SD to be timetabled with timeslot x'_i, let set $\mathcal{Q}_i = \{x_i^j | t_{i,x_i^j} = 1 \wedge j \in 1, \ldots, \text{HMS}\}$ contain the feasible timeslots available for exam i in HM solutions. The timeslot x'_i of exam i will be randomly selected from $\mathcal{Q}_i$ with probability HMCR. In case the $\mathcal{Q}_i = \phi$ which means no feasible timeslot stored in the HM vectors for x'_i, the '**Exceptional random consideration (ERC)**' will run. In ERC, the random consideration operator described below attempts to assign the exam i with timeslot x'_i, as shown Algorithm 1, STEP 3, Line 7.

Random consideration. Exams that are not assigned with timeslots according to memory consideration are randomly assigned according to their available timeslots by random consideration with a probability of (1-HMCR). Formally, let exam i be selected by SD to be assigned with a timeslot x'_i, let set $\mathcal{X}_i = \{d | t_{i,d} = 1 \wedge d \in 1, \ldots, P\}$ contain all feasible timeslots for exam i. The timeslot x'_i of exam i will be randomly selected from $\mathcal{X}_i$ with probability (1-HMCR).

However, in case $\mathcal{X}_i = \phi$, the improvisation process will restart (Henceforth called **restart process**), see Algorithm 1, STEP 3, Lines 9 and 23. In summary see Eq.(5).

$$x'_i \leftarrow \begin{cases} x'_i \in \mathcal{Q}_i & \text{w.p.} \quad \text{HMCR} \\ x'_i \in \mathcal{X}_i & \text{w.p.} \quad \text{(1 - HMCR)} \end{cases} \tag{5}$$

Random consideration is functionally similar to the mutation operator in Genetic Algorithm which is a source of exploration in HSA (Yang 2009). The HMCR parameter is the probability of assigning one timeslot x'_i of exam i, based on historical timeslots stored in the HM solutions.

Pitch adjustment. Every exam i assigned with a timeslot x'_i in the new harmony vector, $\boldsymbol{x}' = (x'_1, x'_2, x'_3, \ldots, x'_N)$, from memory consideration is pitch adjusted with the probability of PAR ($0 \leq \text{PAR} \leq 1$) as follows:

$$\text{Pitch adjust for } x_i'? \leftarrow \begin{cases} \text{Yes} & \text{w.p.} \quad \text{PAR} \\ \text{No} & \text{w.p.} \quad \text{(1-PAR)} \end{cases} \tag{6}$$

If PAR = 0.10, this means that the HSA modifies the existing timeslot of each exam assigned by memory consideration with a probability of (PAR × HMCR), while the timeslot with probability (HMCR × (1 − PAR)) does not change.

For UETP, the pitch adjustment is a neighbourhood move based on 3 neighborhoods, each of which is selected with equal probability as follows:

$$\text{Adjust } x_i' \leftarrow \begin{cases} \text{Single-move} & 0 \leq p \leq \text{PAR1} \\ \text{Swap-timeslot} & \text{PAR1} < p \leq \text{PAR2} \\ \text{Kempe-chain} & \text{PAR2} < p \leq \text{PAR3} \\ \text{do nothing} & \text{otherwise.} \end{cases} \tag{7}$$

Where PAR1=PAR/3 and PAR1:PAR2:PAR3 is in the ratio of 1:2:3, and $p \leftarrow U(0,1)$ is a uniform distribution function which generates a random number between 0 and 1. For each exam i timetabled with x_i' based on memory consideration operator, x_i' is adjusted as follows:

- Pitch adjustment: **Single-move**. With probability range $[0, \text{PAR1}]$, replace the timeslot x_i' of exam i by another feasible timeslot. Although this process is similar to random consideration, it is guided by the objective function of the new harmony.
- Pitch adjustment: **Swap-timeslot**. With probability range of $(\text{PAR1}, \text{PAR2}]$,exam i and exam j swap their timeslots (x_i', x_j') while the feasibility is preserved.
- Pitch adjustment: **Kempe-chain**. With probability range $(\text{PAR2}, \text{PAR3}]$, x_i' is adjusted as follows: (i) Select the timeslot x_i' of exam i and randomly select another timeslot p'. (ii) All exams that have the same timeslot x_i' and conflict with one or more exams timetabled in p' are entered to Chain $\mathcal{G}$ where $\mathcal{G} = \{j | x_j' = x_i' \wedge t_{j,p'} = 0 \wedge \forall j \in \mathcal{E}\}$ (iii) All exams that have the same timeslot p' and conflict with one or more exams timetabled in x_i' are entered to Chain $\mathcal{G}'$ where $\mathcal{G}' = \{k | x_k' = p' \wedge t_{k,x_i'} = 0 \wedge \forall k \in \mathcal{E}\}$, (iv) simply assign the exams in $\mathcal{G}$ with p' and the exams in $\mathcal{G}'$ with x_i'.

In the original HSA proposed by Lee and Geem (2004), the pitch adjustment is unguided by the objective function which can be considered a good source of exploration (Yang 2009). For the purpose of this paper, this component has been modified to be guided by the objective function as follows: the adjustment performed by any pitch adjustment procedure in Eq (7) is accepted in the new harmony if the objective function is not negatively affected, i.e., $f(\boldsymbol{x}'') \leq f(\boldsymbol{x}')$ where $\boldsymbol{x}'$ is the new harmony before the pitch adjustment while $\boldsymbol{x}''$ is the new harmony after it.

Step 4. Update the harmony memory.

If the new harmony vector, $\boldsymbol{x}' = (x_1', x_2', \cdots, x_N')$, is better than the worst harmony vector in HM, the worst harmony vector is substituted with the new harmony vector .

Table 4 Characteristics of the Cater dataset

Datasets Key	Institution	timeslots	exams	students	Density
CAR-S-91	Carleton University, Ottawa	35	682	16925	0.13
CAR-F-92	Carleton University, Ottawa	32	543	18419	0.14
EAR-F-83	Earl Haig Collegiate Institute, Toronto	24	190	1125	0.27
HEC-S-92	Ecole des Hautes Etudes Commercials, Montral	18	81	2823	0.42
KFU-S-93	King Fahd University, Dharan	20	461	5349	0.06
LSE-F-91	London School of Economics	18	381	2726	0.06
RYE-S-93	Ryeson University, Toronto	23	481	11,483	40.07
STA-F-83	St.Andrew's Junior High School, Toronto	13	139	611	0.14
TRE-S-92	Trent University Peterborough, Ontario	23	261	4360	0.18
UTA-S-92	Faculty of Arts and Sciences, University of Toronto	35	622	21266	0.13
UTE-S-92	Faculty of Engineering, University of Toronto	10	184	2750	0.08
YOR-F-83	York Mills Collegiate Institute, Toronto	21	181	941	0.29

Step 5. Check the stop criterion.

Step 3 and step 4 of HSA are repeated until the termination criterion (maximum number of improvisations) is met. This is specified by the NI parameter.

4 Computational experiments

The proposed method is programmed in Microsoft Visual C++ version 6 under Windows XP. The experiments presented here ran on 16 heterogeneous computers with different CPU and RAM capability over 15 days. Note that the total number of experiments is 2040 (17 scenarios × 12 datasets × 10 runs). Thus the computational time has been neglected. As proposed by Qu et al. (2009b); Abdullah et al. (2007), time is not a major constraint as the timetabling problem does not require realtime solutions.

The proposed method is evaluated against *de facto* dataset released by (Carter et al. 1996) which were freely available[3]. Carter's dataset comprises 12 datasets which varies in size (number of exams, number of timeslots) and complexity . The characteristics of Carter dataset are shown in Table 4. The conflict matrix density in the last column refers to the ratio between the number of elements of value $c_{i,j} > 0$ and the total number of elements in the conflict matrix (Qu et al. 2009b). The main objective is to find a conflict-free timetable with the least number of soft constraint violations. The proximity cost function (see Eq.(1)) is used to calculate the Penalty Value (PV) for each timetable obtained. Note that there are two versions of some of Carter dataset (Qu et al. 2009b) and the annotation 'I' refers to which version is used (Pillay and Banzhaf 2009).

[3] http://www.asap.cs.nott.ac.uk/resources/data.shtml

Table 5 Convergence scenarios designed to simulate combinations among metaheuristic components

HMS	HMCR	PAR	Scenario No.	Source of improvement
50	100%	0%	1	GIM
		3%	2	GIM+LIM
		30%	3	GIM+LIM
	98%	0%	4	GIM+RIM
		3%	5	GIM+LIM+RIM
		30%	6	GIM+LIM+RIM
10	100%	3%	7	GIM+LIM
		30%	8	GIM+LIM
	98%	0%	9	GIM+RIM
		3%	10	GIM+LIM+RIM
		30%	11	GIM+LIM+RIM
1	100%	0%	12	No Improvement
		30%	13	LIM
	98%	0%	14	RIM
		30%	15	LIM+RIM
	75%	3%	16	RIM+LIM
		30%	17	RIM+LIM

4.1 Empirical study in Combination of Meta-heuristic components based on HSA

4.1.1 Experimental design

An empirical study of combining the metaheuristic components was conducted using 17 convergence scenarios, each of which varies in parameter settings as shown in Table 5. Each convergence scenario simulates one case of component combinations. The three components are: (i) *recombination* represented by memory consideration which is the source of Global IMprovement (GIM), (ii) the *randomness* is represented by random consideration which is the source of Random IMprovement (RIM), (iii) the *neighbourhood structures* are represented by pitch adjustment which is the source of Local IMprovement (LIM). Each scenario ran 10 times. Note that the NI= 100,000 is fixed for all experiments.

As shown in Table 5, the first 11 scenarios simulate the behaviour of population-based methods, i.e., HMS $\geq$ 1. The remaining scenarios (i.e., 12-17) simulate the behaviour of local search-based methods.

The value of HMCR determines whether the proposed method used only memory consideration component (i.e., GIM) when HMCR =100% or memory consideration (i.e., GIM) and random consideration (i.e., RIM) components when HMCR $<$ 100%. The value of PAR determines the rate of any gradient decent (LIM). when PAR=0, no LIM is obtained. Larger PAR refers to the rate of using pitch adjustment procedures. Note that PAR determines the values of PAR1, PAR2 and PAR3 (PAR1=PAR/3, PAR2= 2PAR/3, and PAR3=PAR).

The combination of metaheuristic components based on the type of improvements is shown in the last column of Table 5.

Analogies:

– Some scenarios combine GIM + RIM + LIM components (i.e., Scens. (5, 6, 10, 11)) in which the proposed method behaves similar to the Memetic Algorithm (MA) (Establish a good balance between exploration and exploitation).

– Some scenarios combine GIM + RIM components (i.e., Scens. (4, 9)) in which the proposed method behaves similar to Genetic Algorithm (GA) (Establish a balance between exploration and exploitation).
– Some scenarios combine GIM + LIM components (i.e., Scens. (2, 3, 7, 8)) in which the proposed method behaves similar to the MA without the mutation operator. (Easily getting stuck in local optimal solution since there is no RIM component)
– Some scenarios combine LIM + RIM components(i.e.,Scens. (15, 16, 17)) in which the proposed method behaves similar to the Simulated Annealing (SA) (Establish a good balance between exploration and exploitation).
– Scen. (1) has only GIM component in which the proposed method behaves similar to the GA without a mutation operator (This leads to a premature convergence problem).
– Scen. (13) has only LIM component in which the proposed method behaves similar to the Hill Climbing (Easily getting stuck in the local optimal solution).
– Scen. (14) has only RIM component which means the proposed method can be considered in this case a local search-based method (i.e.,HMS=1) but without neighbourhood structures. The proposed method is able to improve the solution by an iterative construction process.

Note that in Scen (12), the proposed method does not have any improvement component which means that the initial solution will remain the same during the search.

4.1.2 Experimental Results

Tables 6, 7 , and 8, summarise the results of the 17 scenarios on the penalty value of the solution by recording the best, average, worst and standard deviation (std.dev.) over 10 runs. Note that the three tables are separated with reference to various HMS. The best solution for each dataset is highlighted in bold. The results show that combining GIM + LIM + RIM components (e.g., Scens. (5, 6, 10, 11)) in the same method in general is promising.

4.1.3 Discussion

A closer look at Tables 6, 7 and 8, each scenario reflects the behaviour of the proposed method when one, two or three components are combined. It has to be noted that in HSA, the memory consideration is the source of GIM, random consideration is the source of RIM, and the three pitch adjustment procedures are the source of LIM.

Observations:

– The best solutions are obtained from the scenarios that combine GIM + RIM + LIM components. However, some scenarios that combine GIM + LIM components are able to yield few number of the best solutions (see Scen (2, 3) for CAR-S-91-I and UTA-S-92-I). Note that these scenarios might be affected with exceptional random consideration (ERC) which diversify the search (see Sect. 4.2).
– The overall best results are obtained when the HMS = 50, this suggests that larger HMS allow the HSA to explore multiple search space regions simultaneously which may produce better quality solutions.

– The HMCR affects the balance between exploration and exploitation which means that the larger HMCR leads to less exploration and the greater exploitation. For example, the proposed method in Scen (16, 17) is concerned with exploration rather than exploitation and thus the speed of convergence will be slow.
– The PAR is the rate of any gradient descent. Since it had the larger values, it produced the best results. In other words, the larger the PAR values are, the more rigorous is the fine-tuning of the search space region to which the HSA converges. It is also noted from the comparative evaluation as will be shown in Sect. 4.3 that the best cited results for Carter dataset come from local search-based methods. This suggests that the components that are concerned with exploitation are more useful than those concerned with exploration for UETP. In fact, this is one of the reasons why the most timetabling researchers have lately turned to use local search-based methods rather than population-based method in their timetabling problems.
– The HSA method produces the best results for some problem instances (see CAR-S-91-I, EAR-F-83-I, LSE-F-91, STA-F-83-I) when the value of PAR is 3% while the remaining best results are obtained when the value of PAR is 30%. Note that the PAR is the rate of any gradient decent which indicates that some problem instance can be efficiently tackled when the LIM is few in number while others can be efficiently tackled when the LIM is great in number. In general, some search space reigns of the timetabling problems are very rugged and need considerable local changes until the local optimal solution is obtained.
– The results obtained by Scen. (13, 14, 15) worth considering. Scen (13) simulates a local search-based method with only LIM component which similar to Hill Climbing with three neighbourhood structures (move, Swap, and Kempe Chain). In contrast, Scen. (14) simulates a local search-based method with only RIM component. The best results are obtained from Scen. (15) where the HSA combines LIM + RIM components. It is apparent that a local search-based method with an explorative strategy is able to yield better results than those with no explorative strategy.

4.2 Analysis of the Exceptional Random Consideration (ERC) and the restart process

Using the memory consideration, the proposed method might be unable to assign some exams with timeslots based on HM solutions (i.e., There are no feasible timeslots for some exams in the HM solutions). Therefore, the exceptional random consideration (ERC) attempts to assign these exams with timeslots from their available range (see Algorithm 1, STEP 3, Line 7).

However, if the ERC or random consideration were unable to assign any exam with a timeslot from their available range, the improvisation process would *restart* all over again with a different random seed (restart process). (see Algorithm 1, STEP 3, Lines 9 and 23).

Table 9 shows statistical information on the effect of ECR and the restart process on the behaviour of the proposed method with various HMS. Each number in C1 column is to the average number of exams that are assigned with timeslots based on ERC per 100,000 iterations calculated as follows:

$$C1 = \frac{\sum_{i=1}^{NI} \#\text{ exams assigned with timeslot using ERC}}{NI}$$

Table 6 The Penalty Values obtained by the proposed HSA in different convergence scenarios (Note that the HMS = 50)

Dataset		SCEN.1	SCEN.2	SCEN.3	SCEN.4	SCEN.5	SCEN.6
CAR-S-91-I	Best	5.91	**4.99**	5	5.25	5	5.04
	Average	6.06	5.08	5.14	5.36	5.08	5.13
	Worst	6.16	5.47	5.27	5.45	5.16	5.25
	Std.dev.	0.08	0.14	0.1	0.08	0.06	0.07
CAR-F-92-I	Best	5.04	4.35	4.29	4.56	4.31	5.93
	Average	5.22	4.43	4.44	4.76	4.38	6.1
	Worst	5.37	4.48	4.59	4.88	4.43	6.23
	Std.dev.	0.1	0.05	0.08	0.11	0.05	0.09
EAR-F-83-I	Best	38.77	36.61	34.8	35.63	**34.42**	34.91
	Average	39.91	37.36	35.34	36.56	35.51	35.33
	Worst	40.91	38.21	35.76	37.93	36.58	35.94
	Std.dev	0.851	0.584	0.337	0.825	0.674	0.459
HEC-S-92-I	Best	11.8	11.1	11	11.1	10.6	**10.4**
	Average	12.4	11.4	10.9	11.4	11	10.7
	Worst	12.7	11.8	11.1	11.6	11.4	11
	Std.dev.	0.33	0.24	0.14	0.2	0.24	0.19
KFU-S-93	Best	16.48	14.07	**13.5**	15.08	14	13.66
	Average	17.24	14.34	14.02	15.28	14.42	13.78
	Worst	18.13	14.58	14.46	15.68	14.82	13.94
	Std.dev.	0.584	0.216	0.374	0.203	0.269	0.099
LSE-F-91	Best	13.06	10.96	10.6	11.39	**10.48**	10.69
	Average	13.62	11.49	10.71	11.75	10.77	10.88
	Worst	14.03	11.86	10.9	12.11	11.06	11.19
	Std.dev.	0.265	0.32	0.113	0.248	0.214	0.163
RYE-S-93	Best	10.76	8.98	9.04	9.773	8.85	**8.79**
	Average	11.22	9.16	9.16	10.03	9.07	8.94
	Worst	11.51	9.31	9.27	10.2	9.46	9.17
	Std.dev.	0.244	0.12	0.08	0.13	0.18	0.14
STA-F-83-I	Best	157.92	157.16	157.2	157.19	**157.04**	157.12
	Average	158.6	157.35	157.96	157.49	157.16	157.21
	Worst	159.59	157.54	159.26	157.73	157.36	157.27
	Std.dev.	0.5233	0.1446	0.7619	0.1662	0.0916	0.0351
TRE-S-92	Best	9.58	8.36	8.2	8.41	8.26	**8.16**
	Average	9.65	8.52	8.32	8.75	8.35	8.32
	Worst	9.73	8.71	8.42	9.07	8.43	8.46
	Std.dev.	0.07	0.11	0.09	0.21	0.06	0.09
UTA-S-92-I	Best	4.05	3.51	**3.43**	3.59	3.46	3.49
	Average	4.11	3.55	3.51	3.73	3.52	3.55
	Worst	4.16	3.62	3.59	3.83	3.55	3.61
	Std.dev.	0.04	0.04	0.05	0.07	0.03	0.04
UTE-S-92	Best	27.1	25.75	25.6	25.66	25.3	**25.09**
	Average	28.95	26.28	25.95	26.22	25.85	25.45
	Worst	30.03	26.62	26.35	26.9	26.37	25.76
	Std.dev.	0.836	0.298	0.231	0.483	0.329	0.21
YOR-F-83-I	Best	39.79	37.35	36.13	37.73	36.32	**35.86**
	Average	40.85	37.88	37.04	38.4	37.3	36.56
	Worst	41.55	38.71	38.21	38.98	38.79	36.87
	Std.dev.	0.668	0.377	0.671	0.392	0.74	0.31

Table 7 The Penalty Values obtained by the proposed HSA in different convergence scenarios (Note that the HMS = 10)

Data set		SCEN.7	SCEN.8	SCEN.9	SCEN.10	SCEN.11
CAR-S-91-I	Best	5.02	5.09	5.2	5.19	5.17
	Average	5.15	5.65	5.22	5.42	5.26
	Worst	5.23	6.9	5.26	5.57	5.36
	Std.dev.	0.07	0.75	0.03	0.12	0.06
CAR-F-92-I	Best	4.52	5.91	4.69	4.47	4.75
	Average	4.62	6.22	4.81	4.58	5.53
	Worst	4.72	6.36	5.01	4.69	5.9
	Std.dev.	0.08	0.13	0.12	0.09	0.38
EAR-F-83-I	Best	36.1	37.21	35.72	34.91	34.93
	Average	38.11	38.04	38.69	36.26	35.81
	Worst	38.93	39.39	41.42	37.54	36.77
	Std.dev	0.88	0.809	1.69	1.059	0.747
HEC-S-92-I	Best	11.1	11	11.2	10.6	10.5
	Average	11.6	11.3	11.7	11.1	10.7
	Worst	11.8	11.8	12.2	11.5	10.8
	Std.dev.	0.24	0.26	0.33	0.29	0.12
KFU-S-93	Best	14.25	13.77	14.88	13.93	13.56
	Average	14.8	14.23	15.5	14.2	13.74
	Worst	15.68	14.64	16.41	14.56	13.97
	Std.dev.	0.489	0.302	0.423	0.188	0.153
LSE-F-91	Best	11.15	10.58	11.84	10.73	10.59
	Average	11.63	11.16	12.29	11.29	11.14
	Worst	11.98	11.59	12.72	12.02	11.94
	Std.dev.	0.231	0.334	0.325	0.409	0.373
RYE-S-93	Best	9.09	9.01	10.4	8.86	8.84
	Average	9.56	9.17	10.9	9.23	9.04
	Worst	9.78	9.54	11.6	9.57	9.17
	Std.dev.	0.21	0.15	0.37	0.26	0.15
STA-F-83-I	Best	157.1	157.14	157.31	157.07	157.17
	Average	157.23	157.28	157.66	157.21	157.28
	Worst	157.5	157.44	157.97	157.44	157.38
	Std.dev.	0.1449	0.106	0.2392	0.1481	0.0755
TRE-S-92	Best	8.68	8.55	9.09	8.43	8.27
	Average	8.9	8.65	9.32	8.54	8.32
	Worst	9.32	9.02	9.51	8.69	8.36
	Std.dev.	0.19	0.14	0.13	0.09	0.03
UTA-S-92-I	Best	3.89	3.56	3.68	3.62	3.54
	Average	4	3.64	3.71	3.68	3.66
	Worst	4.11	3.72	3.77	3.74	3.78
	Std.dev.	0.07	0.06	0.03	0.04	0.06
UTE-S-92	Best	26.44	26.09	25.91	25.51	25.37
	Average	27	26.49	26.94	25.91	25.76
	Worst	27.92	26.91	28.49	26.55	26.49
	Std.dev.	0.52	0.245	0.754	0.351	0.35
YOR-F-83-I	Best	37.83	37.54	38.25	36.98	36.17
	Average	39	38.01	40.3	37.65	37.3
	Worst	40.03	38.38	41.96	38.46	38.32
	Std.dev.	0.66	0.282	1.328	0.499	0.69

Table 8 The Penalty Values obtained by the proposed HSA in different convergence scenarios (Note that the HMS = 1)

Data set		SCEN.12	SCEN.13	SCEN.14	SCEN.15	SCEN.16	SCEN.17
CAR-S-91-I	Best	8.28	5.52	6.19	5.49	7.58	7.64
	Average	8.73	5.67	6.45	5.75	7.66	7.75
	Worst	9.05	5.86	6.72	5.98	7.88	7.86
	Std.dev.	0.24	0.11	0.17	0.17	0.1	0.08
CAR-F-92-I	Best	7.42	4.65	5.2	4.45	6.22	6.3
	Average	7.77	4.77	5.46	4.6	6.42	6.41
	Worst	8.05	4.88	5.63	4.74	6.6	6.49
	Std.dev.	0.21	0.08	0.14	0.09	0.12	0.08
EAR-F-83-I	Best	53.92	38.32	39.99	35.27	44.08	42.71
	Average	56.91	40.12	41.5	37.5	44.95	43.95
	Worst	58.89	41.48	43.83	38.93	46.01	46.08
	Std.dev	1.363	0.99	1.132	1.263	0.644	1.054
HEC-S-92-I	Best	17	11.2	11.2	10.7	11.9	11.5
	Average	19.1	11.6	11.8	11	12.4	11.9
	Worst	22.3	11.9	12.5	11.4	12.8	12.2
	Std.dev.	1.68	0.21	0.46	0.28	0.32	0.23
KFU-S-93	Best	23.99	14.49	15.05	14.24	19.09	16.66
	Average	27.13	14.89	15.91	14.47	19.87	17.03
	Worst	30.08	15.39	16.65	14.84	20.43	17.39
	Std.dev.	1.793	0.265	0.547	0.193	0.37	0.247
LSE-F-91	Best	19.29	11.36	11.55	11.29	14.78	15.35
	Average	21.27	11.83	12.01	11.7	15.47	15.87
	Worst	22.35	12.4	12.38	12.4	15.85	16.48
	Std.dev.	0.904	0.316	0.244	0.346	0.354	0.436
RYE-S-93	Best	19.4	9.51	11.1	10.2	14.7	14.4
	Average	20.9	9.99	11.4	10.5	15	14.8
	Worst	22.4	10.2	11.9	10.8	15.1	15.3
	Std.dev.	0.98	0.22	0.34	0.24	0.13	0.32
STA-F-83-I	Best	168.99	157.41	157.68	157.21	158.65	158.52
	Average	177.73	158.13	157.9	157.35	159.64	159.06
	Worst	185.87	158.39	158.3	157.44	160.59	159.77
	Std.dev.	5.0913	0.3182	0.235	0.0831	0.6015	0.4025
TRE-S-92	Best	12.3	9	9.48	8.63	10.4	10.4
	Average	13.3	9.26	9.75	8.97	10.7	10.7
	Worst	14	9.53	10.1	9.13	11	11
	Std.dev.	0.55	0.17	0.17	0.14	0.21	0.21
UTA-S-92-I	Best	5.71	3.71	4.13	3.93	4.92	4.82
	Average	6.16	3.8	4.32	3.99	5.02	4.91
	Worst	6.6	3.91	4.57	4.06	5.14	5.05
	Std.dev.	0.24	0.07	0.13	0.05	0.07	0.07
UTE-S-92	Best	38.88	27.54	26.47	25.7	29.55	29.91
	Average	41.94	28.51	27.16	26.1	30.68	30.38
	Worst	44.31	30.11	27.87	26.93	31.83	31.87
	Std.dev.	1.48	0.741	0.425	0.389	0.724	0.569
YOR-F-83-I	Best	50.42	39.46	40.47	38.59	43.71	43.96
	Average	53.08	40.84	42.86	39.36	45.03	45.07
	Worst	54.66	42.59	46.54	40.03	47.11	46.47
	Std.dev.	1.306	1.103	1.695	0.412	1.051	0.812

Table 9 The numbers of using ERC and restart process on Carter dataset per 100,000 iterations

Data set	HMS=1		HMS=10		HMS=50	
	C1	C2	C1	C2	C1	C2
CAR-S-91-I	141.89	4	48.77	4	14.54	7
CAR-F-92-I	77.83	81	34.1	14	11.15	160
EAR-F-83-I	13.16	256	12.95	70	4.59	295
HEC-S-92-I	4.11	2602	4.03	2867	3.11	5188
KFU-S-93	21.36	1170	16.77	1568	4.16	2663
LSE-F-91	18.09	637	12.58	636	2.51	861
RYE-S-93	27.64	958	13.66	319	3.57	1175
STA-F-83-I	4.32	0	4.01	0	2.09	0
TRE-S-92	20.73	2	30.11	186	4.46	31
UTA-S-92-I	95.55	25	48.87	13	9.57	38
UTE-S-92	4.79	985	3.88	489	1.84	2581
YOR-F-83-I	22.78	1202	18.71	377	4.55	1588

Each number in C2 column is the total number of iterations skipped per 100,000 (restart process).

The results in C1 suggest that increasing the HMS value in general reduces the number of exams assigned with timeslots using ERC per iteration. Although the results in C2 are not related to the HMS values, larger size and complexity of Carter dataset lead to a larger number of skipped iterations (restart process).

4.3 Comparative Evaluation and Analysis

The proposed Harmony Search Algorithm (HSA) is compared with some published methods using the Carter datasets, known and available for the authors. This includes a total of 22 comparator methods which comprise Local Search-based MetaHeuristic Methods (LSMHM), Population-based MetaHeuristic Methods (PMHM), Heuristic Methods (HM) and Hyper-Heuristic Methods (HHM) (See Table 10).

The results of the proposed method were compared with 22 comparative methods as shown in Tables 11, 12 and 13. The numbers in these tables refer to the Penalty Value (PV) calculated by Eq.(1). The indicator '-' shows where the method did not guarantee a feasible timetable (e.g, a hard constraint was not met) or the method did not test the corresponding dataset. The numbers in bold show the best solution obtained for that Carter dataset (lowest is best). The numbers in italic fonts indicate that a different dataset version was used. Note that the results obtained by the proposed method were collected from Tables 6, 7 and 8.

In the proposed method, the results outperformed those produced by heuristic and hyper-heuristic methods in 8 out of 12 Carter datasets shown in Table 11. Clearly, the heuristic and hyper-heuristic methods have not as yet measured up to the results obtained by the metaheuristic-based methods in terms of solution quality.

Table 12 shows the results of the proposed method compared with local-search based methods. The proposed method produces better overall results in 5 out of 11 comparative methods in all Carter dataset. Also the proposed method outperformed the local-search based methods in 2 out 12 Carter datasets. It has to be noted that these methods produced the best solutions cited above. Table 13 lists the results of the proposed method in comparison with the population-based methods. Once again in 9 out of 12 Carter dataset, the proposed method achieved better results

Table 10 Key to the comparator methods

#	Method	Class	Reference
0	Harmony Search Algorithm	PMHM	< *proposed method*>
1	Graph Coloring Heuristic Methods	HM	Carter et al. (1996)
2	Tabu Search Algorithm	LSMHM	Di Gaspero and Schaerf (2002)
3	Tabu Search Algorithm	LSMHM	Di Gaspero (2002)
4	Tabu Search Algorithm	LSMHM	Paquete and Stutzle (2003)
5	Local Search-based Methods	LSMHM	Burke and Newall (2003)
6	GRASP local Search-based Method	LSMHM	Casey and Thompson (2003)
7	Simulated Annealing and Hill Climbing	LSMHM	Merlot et al. (2003)
8	Time-Predefined Great Deluge	LSMHM	Burke et al. (2004)
9	Adaption of Heuristic Orderings	HM	Burke and Newall (2004)
10	Similarity Measure for Heuristic Selection	HM	Yang and Petrovic (2005)
11	Fuzzy Multiple Heuristic Orderings	HM	Asmuni et al. (2005)
12	Tabu Search Hyper-Heuristic Approach	HHM	Kendall and Hussin (2005)
13	Multi-Objective Evolutionary Algorithm	PMHM	Cote et al. (2005)
14	Hybrid Variable Neighbourhood	LSMHM	Burke et al. (2006)
15	Ant Colony Algorithm	PMHM	Eley (2007)
16	Graph-Based Hyper-Heuristic	HHM	Burke et al. (2007)
17	Ahuja-Orlin's Method	LSMHM	Abdullah et al. (2007)
18	Graph-Based Hyper-Heuristic	HHM	Qu and Burke (2009)
19	Local Search-based Methods	LSMHM	Caramia et al. (2008)
20	Graph-Based Hyper-Heuristic	HHM	Qu et al. (2009a)
21	Graph-Based Hyper-Heuristic	HHM	Pillay and Banzhaf (2009)
22	Fuzzy Multiple Heuristic Orderings	HM	Asmuni et al. (2009)

Finally, Table 14 shows the results of the proposed method in comparison with the best overall results obtained by the 22 comparative methods in Table 10. The differences between the results in the 3rd column indicate that the proposed method is able to produce respectable solutions that are very much near the best solutions. The last column in the Table 14 shows the ranking position of each result in a total of 22 comparative methods. For example, 5th position means that the result obtained by HSA ranks 5th out of 23 methods (including the proposed method).

Table 11: Comparison with heuristic and hyper-heuristic approaches

Data set	< *proposed method*>	Carter et al. (1996)	Burke and Newall (2004)	Asmuni et al. (2005)	Kendall and Hussin (2005)	Burke et al. (2007)	Qu et al. (2009a)	Qu and Burke (2009)	Pillay and Banzhaf (2009)	Asmuni et al. (2009)
CAR-S-91-I	4.99	7.1	5	5.19	5.37	5.36	5.11	5.16	**4.97**	5.29
CAR-F-92-I	4.29	6.2	4.3	4.51	4.67	4.93	4.32	**4.16**	4.84	4.54
EAR-F-83-I	**34.42**	36.4	36.2	36.64	40.18	37.92	35.56	35.86	36.86	37.02
HEC-S-92-I	**10.4**	10.8	11.6	11.6	11.86	12.25	11.62	11.94	11.85	11.78
KFU-S-93	**13.5**	14	15	15.34	15.84	15.2	15.18	14.79	14.62	15.8
LSE-F-91	**10.48**	10.5	11	11.35	-	11.33	11.32	11.15	11.14	12.09
RYE-S-93	8.79	**7.3**	-	10.05	-	-	-	-	9.65	10.38
STA-F-83-I	**157.04**	161.5	161.9	160.79	157.38	158.19	158.88	159	158.33	160.42
TRE-S-92	**8.16**	9.6	8.4	8.47	8.39	8.92	8.52	8.6	8.48	8.67
UTA-S-92-I	3.43	3.5	3.4	3.52	-	3.88	**3.21**	3.59	3.4	3.57
UTE-S-92	**25.09**	25.8	27.4	27.55	27.6	28.01	28	28.3	28.88	28.07
YOR-F-83-I	**35.86**	41.7	40.8	39.79	-	41.37	40.71	41.81	40.74	39.8

Table 12: Comparison with local based metaheuristic-based search approaches

Data set	< *proposed method*>	Caramia et al. (2008)	(Di Gaspero and Schaerf 2002)	Di Gaspero (2002)	Paquete and Stutzle (2003)	Burke and Newall (2003)	Casey and Thompson (2003)	Merlot et al. (2003)	Burke et al. (2004)	Yang and Petrovic (2005)	Burke et al. (2006)	Abdullah et al. (2007)
CAR-S-91-I	4.99	6.6	6.2	5.7	-	4.65	5.4	5.1	4.8	**4.5**	4.6	5.2
CAR-F-92-I	4.29	6	5.2	-	-	4.1	4.4	4.3	4.2	**3.93**	4	4.4
EAR-F-83-I	34.42	**29.3**	45.7	39.4	38.9	37.05	34.8	35.1	35.4	33.7	32.8	34.9
HEC-S-92-I	10.4	**9.2**	12.4	10.9	11.2	11.54	10.8	10.6	10.8	10.83	10	10.3
KFU-S-93	13.5	13.8	18	-	16.5	13.9	14.1	13.5	13.7	13.82	**13**	13.5
LSE-F-91	10.48	**9.6**	15.5	12.6	13.2	10.82	14.7	10.5	10.4	10.35	10	10.2
RYE-S-93	8.79	**6.8**	-	-	-	-	-	-	8.9	8.53	-	8.7
STA-F-83-I	**157.04**	158.2	160.8	157.4	168.3	168.73	*134.9*	157.3	159.1	158.35	159.9	159.2
TRE-S-92	8.16	9.4	10	-	9.3	8.35	8.7	8.4	8.3	7.92	**7.9**	8.4
UTA-S-92-I	3.43	3.5	4.2	4.1	-	3.2	-	3.5	3.4	**3.14**	3.2	3.6
UTE-S-92	25.09	**24.4**	27.8	-	29	25.83	25.4	25.1	25.7	25.39	24.8	26
YOR-F-83-I	**35.86**	36.2	41	39.7	38.9	37.28	37.5	37.4	36.7	36.35	37.28	36.2

Table 13 Comparison with population based metaheuristic approaches

Data set	< *proposed method*>	Cote et al. (2005)	Eley (2007)
CAR-S-91-I	**4.99**	5.4	5.2
CAR-F-92-I	4.29	**4.2**	4.3
EAR-F-83-I	34.42	**34.2**	36.8
HEC-S-92-I	**10.40**	**10.4**	11.1
KFU-S-93	**13.5**	14.3	14.5
LSE-F-91	**10.48**	11.3	11.3
RYE-S-93	**8.79**	8.8	9.8
STA-F-83-I	**157.04**	158.03	157.3
TRE-S-92	**8.16**	8.6	8.6
UTA-S-92-I	**3.43**	3.5	3.5
UTE-S-92	**25.09**	25.3	26.4
YOR-F-83-I	**35.86**	36.4	39.4

5 Conclusion and future work

In this paper, we have conducted a preliminary investigation of combining the key metaheuristic components that lead to the best solutions for Uncapacitated Examination Timetabling Problem (UETP). Harmony Search Algorithm (HSA), tailored for the purpose of this study, is iterated toward an optimal solution using three components: Memory Consideration, Random Consideration, and Pitch Adjustment. The proposed method has been evaluated against 12 datasets defined by Carter et al. (1996) and has been able to obtain two best overall results when compared with those obtained by 22 comparative methods available to the researcher.

Three metaheuristic components have been investigated: recombination, randomness and neighborhood structures, that have been classified according to the types of improvement provided: global improvement (GIM), random improvement (RIM), and local improvement (LIM) respectively. we experimented with 17 convergence scenarios each of which simulating a combination among those components. The results show that the method combining GIM + RIM + LIM components can obtain high quality solutions for for most test timetabling instances. This suggests that hybridization be-

Table 14 The comparison between the proposed method and the best cited results obtained from approaches in Table 7, 8 and 9 of Carter benchmarks

Data set	< *proposed method*>	Best result cited	differences	position – out of 23 methods
CAR-S-91-I	4.99	**4.5**	0.49	5th
CAR-F-92-I	4.29	**3.93**	0.36	7th
EAR-F-83-I	34.42	**29.3**	5.12	5th
HEC-S-92-I	10.4	**9.2**	1.2	4th
KFU-S-93	13.5	**13**	0.5	2nd
LSE-F-91	10.48	**9.6**	0.88	6th
RYE-S-93	8.79	**6.8**	1.99	5th
STA-F-83-I	**157.04**	157.2	-0.16	1st
TRE-S-92	8.16	**7.9**	0.26	3rd
UTA-S-92-I	3.43	**3.14**	0.29	8th
UTE-S-92	25.09	**24.4**	0.69	3rd
YOR-F-83-I	**35.86**	36.2	-0.34	1st
Total	316.45	305.17	11.28	

tween the key components of local search-based methods and those of population-based methods, is a promising research area which can produce good solutions for the most difficult UETP instances.

The convergence scenarios can also be seen as a parameter sensitivity analysis for the proposed HSA. Their results suggest that increasing HMS leads to increasing the ability of the proposed method to explore multiple search space regions simultaneously. Furthermore, the more the HCMR is, the less exploration and greater exploitation will be. In the timetabling domain, exploitation is more useful than exploration due to the structure of the search space. As such HMCR should be large enough to avoid the random search. PAR is the rate of any gradient descent. Larger PAR leads to rigorous fine-tuning in the search space region and more exploitation. In conclusion, larger HMCR, PAR and HMS, lead to better results.

The combination of metaheuristics components using HSA as an optimisation framework can inspire future researchers with food for thought. For example, the acceptance rule of Simulated Annealing can be combined with STEP 4 of the HSA.

References

Abdullah S, Ahmadi S, Burke EK, Dror M (2007) Investigating ahuja-orlin's large neighbourhood search approach for examination timetabling. OR Spectrum 29(2),351–372

Al-Betar MA, Khader AT (2009) A hybrid harmony search for university course timetabling. In: Blazewicz J, Drozdowski M, Kendall G, McCollum B (eds) Proceedings of the 4nd Multidisciplinary Conference on Scheduling: Theory and Applications (MISTA 2009), Dublin, Ireland, pp 157–179

Al-Betar MA, Khader AT, Gani TA (2008) A harmony search algorithm for university course timetabling. In: 7th International Conference on the Practice and Theory of Automated Timetabling (PATAT 2008), Montreal, Canada

Al-Betar MA, Khader AT, Liao IY (2010) A harmony search algorithm with multi-pitch adjusting rate for university course timetabling. In: Geem Z (ed) Recent Advances In Harmony Search Algorithm, Studies in Computational Intelligence (SCI), vol 270, Springer-Verlag, Berlin, Heidelberg, pp 147–162

Asmuni H, Burke EK, Garibaldi JM, McCollum B (2005) Fuzzy multiple heuristic orderings for examination timetabling. In: Proceedings of the 5th International Conference onPractice and Theory of Automated Timetabling (PATAT2004), LNCS, vol 3616, Berlin: Springer-Verlag, Pittsburgh, PA, USA

Asmuni H, Burke EK, Garibaldi JM, McCollum B, Parkes AJ (2009) An investigation of fuzzy multiple heuristic orderings in the construction of university examination timetables. Computers & Operations Research 36(4),981–1001

Blum C, Roli A (2003) Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Comput Surv 35(3),268–308

Brélaz D (1979) New methods to color the vertices of a graph. Commun ACM 22(4),251–256

Burke E, Newall J (2004) Solving examination timetabling problems through adaption of heuristic orderings. Annals of Operations Research 129(1),107–134

Burke EK, Newall JP (2003) Enhancing timetable solutions with local search methods. In: in Proceedings of the 4th International Conference on Practice and Theory of

Automated Timetabling (PATAT2002), LNCS, vol 2740, Berlin: Springer-Verlag, KaHo St.-Lieven, Gent, Belgium, pp 195–206

Burke EK, Bykov Y, Newall J, Petrovic S (2004) A time-predefined local search approach to exam timetabling problems. IIE Transactions 36(6),509–528

Burke EK, Eckersley AJ, McCollum B, Petrovic S, Qu R (2006) Hybrid variable neighbourhood approaches to university exam timetabling. Tech. Rep. Technical Report NOTTCS-TR-2006-2, School of Computer Science, University of Nottingham, UK

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

Caramia M, Dell'Olmo P, Italiano G (2008) Novel local-search-based approaches to university examination timetabling. Informs Journal on Computing 20(1),86–99

Carter MW, Laporte G, Lee SY (1996) Examination timetabling: Algorithmic strategies and applications. Journal of the Operational Research Society 74,373–383

Casey S, Thompson J (2003) Grasping the examination scheduling problem. In: in Proceedings of the 4th International Conference on Practice and Theory of Automated Timetabling (PATAT2002), LNCS, vol 2740, Berlin: Springer-Verlag, KaHo St.-Lieven, Gent, Belgium, pp 232–244

Cote P, Wong T, Sabouri R (2005) Application of a hybrid multi-objective evolutionary algorithm to the uncapacitated exam proximity problem. In: Proceedings of the 5th International Conference on Practice and Theory of Automated Timetabling (PATAT2001), LNCS, vol 3616, Berlin: Springer-Verlag, pp 151–168

Di Gaspero L (2002) Recolour, shake and kick: A recipe for the examination timetabling problem. In: in Proceedings of the 4th International Conference on Practice and Theory of Automated Timetabling (PATAT2002), KaHo St.-Lieven, Gent, Belgium

Di Gaspero L, Schaerf A (2002) Tabu search techniques for examination timetabling. In: Proceedings of the 3rd International Conference on Practice and Theory of Automated Timetabling (PATAT2001), LNCS, vol 3616, Berlin: Springer-Verlag

Eley M (2007) Ant algorithms for the exam timetabling problem. In: Proceedings of the 5th International Conference on Practice and Theory of Automated Timetabling (PATAT2001), LNCS, vol 3616, Berlin: Springer-Verlag, pp 364–382

Fesanghary M, Mahdavi M, Minary-Jolandan M, Alizadeh Y (2008) Hybridizing harmony search algorithm with sequential quadratic programming for engineering optimization problems. Computer Methods in Applied Mechanics and Engineering 197(33-40),3080–3091

Geem ZW, Kim JH, Loganathan GV (2001) A New Heuristic Optimization Algorithm: Harmony Search. Simulation 76(2),60–68

Ingram G, Zhang T (2009) Overview of applications and developments in the harmony search algorithm. In: Geem ZW (ed) Music-Inspired Harmony Search Algorithm, Springer, Berlin, Heidelberg, pp 15–37

Kendall G, Hussin N (2005) A tabu search hyper-heuristic approach to the examination timetabling problem at the mara university of technology. In: Proceedings of the 5th International Conference on Practice and Theory of Automated Timetabling (PATAT2001), LNCS, vol 3616, Berlin: Springer-Verlag, pp 270–293

Lee K, Geem ZW, Lee Sh, Bae Kw (2005) The harmony search heuristic algorithm for discrete structural optimization. Engineering Optimization 37(7),663–684

Lee KS, Geem ZW (2004) A new structural optimization method based on the harmony search algorithm. Computers and Structures 82(9-10),781 – 798

Lee KS, Geem ZW (2005) A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. Computer Methods in Applied Mechanics and Engineering 194(36-38),3902–3933

McCollum B, Schaerf A, Paechter B, McMullan P, Lewis R, Parkes A, Di Gaspero L, Qu R, Burke EK (2009) Setting the Research Agenda in Automated Timetabling: The Second International Timetabling Competition. INFORMS JOURNAL ON COMPUTING DOI 10.1287/ijoc.1090.0320

Merlot LT, Boland N, Hughes BD, Stuckey PJ (2003) A hybrid algorithm for the examination timetabling problem. In: in Proceedings of the 4th International Conference on Practice and Theory of Automated Timetabling (PATAT2002), LNCS, vol 2740, Berlin: Springer-Verlag, KaHo St.-Lieven, Gent, Belgium, pp 207–231

Ochoa G, Qu R, Burke EK (2009) Analyzing the landscape of a graph based hyper-heuristic for timetabling problems. In: GECCO '09: Proceedings of the 11th Annual conference on Genetic and evolutionary computation, ACM, New York, NY, USA, pp 341–348

Paquete L, Stutzle T (2003) Empirical analysis of tabu search for the lexicographic optimization of the examination timetabling problem. In: in Proceedings of the 4th International Conference on Practice and Theory of Automated Timetabling (PATAT2002), LNCS, vol 2740, KaHo St.-Lieven, Gent, Belgium, pp 413–420

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 (2009) Hybridizations within a graph-based hyper-heuristic framework for university timetabling problems. Journal of the Operational Research Society 60,1273–1285

Qu R, Burke EK, , McCollum B (2009a) Adaptive automated construction of hybrid heuristics for exam timetabling and graph colouring problems. European Journal of Operational Research 198(2),392–404

Qu R, Burke EK, McCollum B, Merlot LTG, Lee SY (2009b) A survey of search methodologies and automated system development for examination timetabling. Journal of Scheduling 12(1),55–89

Yang XS (2009) Harmony search as a metaheuristic algorithm. In: Geem ZW (ed) Music-Inspired Harmony Search Algorithm, Springer, Berlin, Heidelberg, pp 1–14

Yang Y, Petrovic S (2005) A novel similarity measure for heuristic selection in examination timetabling. In: Proceedings of the 5th International Conference on Practice and Theory of Automated Timetabling (PATAT2001), LNCS, vol 3616, Berlin: Springer-Verlag, pp 247–269