Asian Journal of Research in Computer Science

5(3): 21-38, 2020; Article no.AJRCOS.55744
ISSN: 2581-8260

A Mathematical Programming Model and Enhanced Simulated Annealing Algorithm for the School Timetabling Problem

O. A. Odeniyi[1*], E. O. Omidiora[2*], S. O. Olabiyisi[3*] and C. A. Oyeleye[4*]

[1]Department of Computer Science, Osun State College of Technology, Esa-Oke, Nigeria.
[2]Department of Computer Engineering, Ladoke Akintola University of Technology, Ogbomoso, Nigeria.
[3]Department of Computer Science, Ladoke Akintola University of Technology, Ogbomoso, Nigeria.
[4]Department of Information Systems, Ladoke Akintola University of Technology, Ogbomoso, Nigeria.

Authors' contributions

This work was carried out in collaboration among all authors. Author OAO formulated the mathematical programming model and Enhanced Simulated Annealing (ESA) algorithm for solving the school timetabling problem and wrote the first draft of the manuscript. Authors EOO and SOO implemented and validated the formulated mathematical model and ESA algorithm using Matrix Laboratory 8.6 software and a Nigerian high school dataset respectively. Author CAO managed the literature searches, references and does the final manuscript. All authors read and approved the final manuscript.

Article Information

DOI: 10.9734/AJRCOS/2020/v5i330136
Editor(s):
(1) Dr. G. Sudheer, GVP College of Engineering for Women, India.
Reviewers:
(1) Pasupuleti Venkata Siva Kumar, Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering & Technology, India.
(2) Tajini Reda, École Nationale Supérieure des Mines de Rabat, Morocco.
(3) Marco Benvenga, UNIP Universidade Paulista, Brazil.
(4) David Lizcano, Madrid Open University, Spain.
Complete Peer review History: http://www.sdiarticle4.com/review-history/55744

Original Research Article

Received 20 February 2020
Accepted 27 April 2020
Published 02 May 2020

ABSTRACT

Despite significant research efforts for School Timetabling Problem (STP) and other timetabling problems, an effective solution approach (model and algorithm) which provides boundless use and high quality solution has not been developed. Hence, this paper presents a novel solution approach for solving school timetabling problem which characterizes the problem-setting in the timetabling problem of the high school system in Nigeria. We developed a mixed integer linear programming model and meta-heuristic method - Enhanced Simulated Annealing (ESA) algorithm. Our method

**Corresponding authors: E-mail: olufemiodeniyi@gmail.com, eoomidiora@lautech.edu.ng, soolabiyisi@lautech.edu.ng, caoyeleye@lautech.edu.ng;*

incorporates specific features of Simulated Annealing (SA) and Genetic Algorithms (GA) in order to solve the school timetabling problem. Both our solution approach and SA approach were implemented using Matrix Laboratory 8.6 software. In order to validate and demonstrate the performance of the developed solution approach, it was tested with the highly constrained school timetabling datasets provided by a Nigerian high school using constraints violation, simulation time and solution cost as evaluation metrics. Our developed solution approach is able to find optimal solution as it satisfied all the specified hard and soft constraints with average simulation time of 37.91 and 42.16 seconds and solution cost of 17.03 and 18.99, respectively, for JSS and SSS to the problem instance. A comparison with results obtained with SA approach shows that the developed solution approach produced optimal solution in smaller simulation time and solution cost, and has a great potential to solve school timetabling problems with satisfactory results. The developed ESA algorithm can be used for solving other related optimization problems.

Keywords: School timetabling problem; model; mixed integer linear programming; meta-heuristics; genetic algorithm; simulated annealing.

1. INTRODUCTION

The School Timetabling Problem usually denoted as High School Timetabling Problems (HSTP) is concerned with the process of fixing a sequence of meetings between teachers and classes (set of lessons) to a specific number of timeslots within a prefixed time period (typically a week), satisfying a set of constraints of various kinds [1,2,3]. These constraints are usually classified into two types, hard and soft. Hard constraints must be satisfied in order to provide a feasible solution, whereas, soft constraints which express the preferences and the quality of the timetable can be violated (but must be satisfied as far as possible) [4]. The quality of a timetable is measured based on how well the soft constraints have been satisfied. The more soft constraints are satisfied, the better is the quality of the solution considered. Further discussion on school timetabling can be found in [2,5].

Literature on school timetabling is very extensive with several solution approaches (models and algorithms) proposed. In fact, many models including XHSTT format (which is based on the Extensible Markup Language standard) and several NP-complete variants of HSTP have been proposed in the literature, which differ due to the difference in educational system of each country, context of the application, the school and the place where it is located. Representative literature include [6,7,8,9,10,11,12,13,14]. In addition, a variety of heuristic (analytical) and complete methods have been proposed to solve HSTP including hybrid or enhanced which is surveyed by [3].

Nevertheless, developing good models and algorithms for solving HSTP is a challenging task due to its inherent problem characteristics and complexities (a large search space of the problem – large number of events required to be assigned to resources while satisfying a large list of constraints; increasing number of courses/ subjects that are very diverse for the different countries to be scheduled; a varying structure in different high schools even in the same country or educational systems and a wider variety of many different requirements (both objectives and constraints) which are usually institution-specific that changes day by day) [5,15,16,17,18]. Furthermore, [19] confirmed that every model has limited use and many problems are still not solved efficiently or optimally as most of these solution approaches however generated feasible or low quality solutions. Even [20] opined that finding a feasible solution to HSTP is often difficult especially when the resources are tight and usually results to low computational efficiency.

HSTP has been intensively investigated since the1960s [21,22]. Recent years have seen an increased level of research activity in this area. This is evidenced (among other things) by survey studies, for example, [23,24,25], and the emergence of a series of international conferences on the Practice and Theory on Automated Timetabling (PATAT), the Multi-disciplinary International Scheduling Conference: Theory and Application (MISTA) and the establishment of a European Association of Operational Research Societies (EURO) working group on automated timetabling [26]. However, the international conference series on PATAT has contributed largely to the area, for example [10,18,27,28,29].

Graphs, Networks and Integer Programming (IP) have proven to be useful in the mathematical formulation and solution of school timetabling

models [15,30]. Integer Programming (IP) has been used to model problems with more sophisticated requirements including educational timetabling problems since the very early days of Operations Research, for example, [22,31]. However, IP is mainly used for timetabling due to its modeling strength, and not as an actual solution method since only small instances, or simplified variants of the problem, were shown to be solved in reasonable time. In recent times several improvements have been made in IP solvers [32] and several IP based techniques have been introduced for similar HSTPs which were able to provide bounds and good solutions after longer runs. [7,8] surveyed some of these recent contributions.

Over the last few years, meta-heuristics including its hybrids have proven to be very effective in the optimisation literature, and particularly in school timetabling. Souza et al. [33] applied a hybrid approach of GRASP-Tabu search algorithm to solve school timetabling problem in order to improve the timetable's compactness and speeds up the process of obtaining better quality solutions. In this work, while the partially greedy constructive procedure was used to generate good initial solutions and diversifies the search, the Tabu Search procedure was used to improve (refine) the constructed solution (search). Though quality solution was generated, it was suggested that quality of the solution can be improved by considering other requirements and adding component which measures the difference between the current and the desired solution to the objective function.

Soza et al. [34] addressed the solution of timetabling problems using cultural algorithms. The authors proposed the use of domain knowledge, both a priori and extracted during the search to improve the performance of an evolutionary algorithm when solving timetabling problems. The experimental results provided very encouraging results. However, the optimality of the solution was in doubt as soft constraints violations were not addressed. It was suggested that the performance of the developed algorithm can be improved by analysing the mechanisms of the simulated annealing method for incorporation into an evolutionary algorithm or a cultural algorithm.

Santos et al. [35] presented a hybrid SA-ILS approach to solve the high school timetabling problem. The Kingston's High School Engine (KHE) was used to generate an initial solution while Simulated Annealing (SA) and Iterated Local Search (ILS) were used to perform local search around the initial solution. The experimental results produced several best known solutions for the test set. However, it was suggested that the quality of the solution can be improved by the development of an augmented set of (larger) neighbourhoods and a proper experimental study to fine tune parameter selection.

Sorensen et al. [36] established a new hybrid approach called matheuristic, combining integer programming and meta-heuristics to solve the high school timetabling problem. The experimental results illustrated that the developed matheuristic provided promising results and the potential of hybridising mathematical programming and meta-heuristics. However, it was suggested that the computational efficiency of the developed matheuristic can be improved by investigating more advanced approaches for adjusting the size of the neighbourhoods and selecting variables in each neighbourhood.

Raghavjee [37] presented genetic algorithms to solve STP in order to test the effectiveness of a genetic algorithm approach in solving more than one type of STP and as well to evaluate the performance of a genetic algorithm that uses an indirect representation (IGA) with that of a genetic algorithm that uses a direct representation (DGA) when solving the STP. Both the DGA and IGA when tested on five STPs were found to produce timetables that were competitive and in some cases superior to that of other methods. However, the IGA outperformed the DGA for all of the tested STPs. The solution did not address the problem of optimality and both software and computational complexities which can be considered for future research. In addition, it was recommended that future research may be geared towards implementing more advanced techniques (more efficient algorithms) to solve the STP so as to provide more bases for comparison.

Kristiansen [38] developed different techniques to solve Multiple Timetabling Problems (High School Timetabling, Student Sectioning and Meeting Planning Problem) at the Danish High Schools. Techniques developed included Adaptive Large Neighbourhood Search (ALNS), Mixed Integer Programming (MIP) Dantzig-Wolfe decomposition in a Branch-and-Bound and Column generation with a Branch-and-Price

(B&P). The experimental results proved that all the developed techniques and algorithms were efficient. However, it was suggested that future research may be geared towards investigating more techniques such as hyper-heuristics and matheuristics to measure and improve the quality of solution and computational efficiency of the developed exact methods.

Odeniyi et al. [2] developed a modified simulated annealing (MSA) algorithm to solve STP. The implementation of the MSA was characterized by cooling schedule modification with the introduction of a non-linear factor to the exponential cooling schedule of classical simulated annealing ($\alpha = (1/\log(1+t))$ to become parabolic (as $\alpha = (1/\log(1+t+t^2))$, in order to decrease the sensitivity to the initial values (parameters) that accounted for the excessive convergence time of classical simulated annealing. MSA when tested on some variants of the STP that originated from Nigerian secondary educational institutions produced conflict-free school timetables which satisfied all hard constraints and optimally minimised all soft constraints violations in less simulation time and less solution cost. However, the solution did not address the problem of software and computational complexities which can be considered for future research. In addition, it was recommended that future research may be geared towards improving the quality of the solution of the developed algorithm by implementing more advanced techniques such as efficient hybridised algorithms.

Dorneles [39] developed techniques that combine mathematical programming and heuristics, so-called matheuristics (fix-and-optimize heuristic combined with a variable neighbourhood descent strategy and column generation approach), to solve efficiently and in a robust way some variants of the High School Timetabling Problem (HSTP) that originated from Brazilian institutions. The performance of the developed matheuristics were evaluated and compared with the state-of-the-art MIP solvers designed for solving the GHSTP, referred as GOAL and SVNS. The experimental results provided very encouraging results which indicated that the state-of-the-art MIP solvers were not efficient for solving instances of the $HSTP^+$. It was suggested that future research may be geared towards reducing the simulation time while improving both the quality of the solution and computational efficiency of the developed matheuristics.

Raghavjee et al. [40] presented a genetic algorithm selection perturbative hyper-heuristic (GASPHH) to solve the STP. A 2-phased approach was taken, with the first phase focusing on hard constraints, and the second on soft constraints. GASPHH used tournament selection to choose parents, to which the mutation and crossover operators were applied. GASPHH was applied to five different school timetabling problems. The performance of GASPHH was evaluated and compared to that of other methods applied to these problems, including a GA that was applied directly to the solution space. GASPHH produced feasible timetables for all problem instances, provided a generalized solution to the STP, and performed better than other methods applied to the same set of problems. However, GASPHH was computational intensive in comparison to SGA based benchmark heuristics. It was suggested that future research may be geared towards investigating approaches to reduce both the simulation time and computational complexity as well to improve the computational efficiency of the developed GASPHH.

Demirovic [6] presented SAT-based approaches for HSTP. The problem of determining whether a propositional logic formula has a solution is called the satisfiability problem (abbreviated as SAT). The thesis explored the relation between propositional logic and HSTP, as well as related approaches. The thesis described the general HSTP (XHSTT), introduced a formal definition, as well two different modelling approaches, SAT- and bitvector based. The author combined local search and large neighbourhood search to solve XHSTT instances which were modelled as maxSAT. Each of the described methods is vastly different from each other and represented distinct ways to tackle XHSTT. The computational results showed that the developed models and solution methods were competitive with the state-of-the-art. However, it was suggested that future research may be geared towards investigating techniques to find and prove optimal solutions in reasonable time for XHSTT in many instances.

A review of the above related works with literature analysis revealed most of these solution approaches however generated feasible or near-optimal solutions with low computational efficiency. Therefore, this research focuses on developing a more efficient solution approach, utilising the strength and minimising the weaknesses of two well-known meta-heuristics,

Simulated Annealing (SA) and Genetic Algorithm (GA). SA is based on an analogy to thermodynamics simulating the cooling of a set of heated atoms. This technique starts its search from any initial solution which can be generated randomly or by using solutions generated by another algorithm [2,41]. The main procedure consists of a loop that randomly generates, at each iteration, one neighbour s′ of the current solution s. Movements are probabilistically selected considering a temperature *T* and the cost variation of the movement Δ [42].

SA is known for its power to avoid local optima and its theoretical guaranty to find the global optimum solution when the initial temperature is high enough and cooling rate is infinitely slow [43,44,45] but converges at excessive time especially when the search space is large [46]. Several attempts have been made to speed up this process, such as treatment or modification of the temperature parameter, which is known as annealing schedule or strategy [2], and hybridization with other techniques [47,48,49,50], among others.

GA is a biologically motivated adaptive meta-heuristic that is based on natural selection and genetic recombination. It utilizes selection, crossover and mutation mechanisms to evolve the population. The usage of GA in obtaining the optimal parameters for a kernel function, its flexibility, coupled with its demonstrated capability of searching or exploring large search spaces [51,52], its demonstrated ability to reach near-optimum solutions to large problems, and its power to discover good solutions rapidly for difficult high-dimensional problems makes the technique an ideal candidate for consideration in solving the timetabling problem.

In this work, we extend previous studies about the HSTP. We propose a mathematical programming model and meta-heuristic method to construct school timetables. We utilise mixed integer linear programming formulation to model the problem to be solved but due to the limitation of a mathematical programming approach to solve large problem instances and slow convergence speed of classical simulated annealing (SA) algorithm, we propose a new meta-heuristic method, called Enhanced Simulated Annealing (ESA) algorithm which incorporates best features of Simulated Annealing (SA) and Genetic Algorithm (GA) to find quality solutions to some variants of the HSTP.

The aim of this work is to develop an effective solution approach (model and algorithm) which provides boundless use and high quality solution to the school timetabling problem. The overall idea is to overcome the problem of low solution quality (slow convergence speed), which is a common problem when using simulated annealing (SA). The developed solution approach was implemented using Matrix Laboratory 8.6 software. In order to validate and demonstrate the performance of the proposed solution approach, it was tested with the highly constrained school timetabling datasets provided by a Nigerian high school, using computation time, solution cost and constraints violation as evaluation metrics.

Developing models and algorithms that automatically generate high quality timetables is of great importance [6] and is still an active field of research as well as important issues in this domain. High quality timetables are relevant for financial and pedagogical reasons. High quality timetables directly influence the quality of teaching and learning, working conditions of teachers as well as the maintenance of the satisfaction of students and staff, among other things, which leads to an overall better use of resources and learning environment [12,53]. Conversely, timetables construction by hand is usually very difficult, time consuming, and error prone. Consequently, assisting the high school planners with efficient algorithms and decision support software that will automatically generate high quality timetables and/or spend less time is of great importance [6,7]

Furthermore, the high schools administration requires a model of the HSTP which is general enough to suit many different requirements, and which is also tractable by computer aided solution methods. This supports the recent trend of developing general models for HSTP, for example [6,9,10,12, 14,16,17,54,55,56,57,58]. In addition, the mathematical programming model formulated in this paper is the first for high school system in Nigeria, and as well as among the most comprehensive models of school timetabling to be found in the literature. Although the developed model is tailored to the Nigeria case but can easily be adapted to other variants from other countries and other related optimization problems. The model is 'complete' in the sense that it contains all relevant practical constraints available in the context of high school timetabling problem in Nigeria.

2. METHODOLOGY

In this work, mixed integer linear programming technique was used to formulate the mathematical programming model of the school timetabling problem for the high school system in Nigeria (HNSTP). An Enhanced Simulated Annealing (ESA) algorithm for solving the school timetabling problem was formulated, which incorporates specific features of Simulated Annealing (SA) and Genetic Algorithms (GA). The formulated mathematical model with ESA algorithm and SA algorithm were implemented using Matrix Laboratory 8.6 software on an Intel(R) Core(TM) i3-7100 CPU with 2.50GHz speed, 32GB Hard Disk, 4GB Random Access Memory and 32-bit Operating System, with the Window 7 operating system, and validated using a highly constrained Nigerian high school dataset. The implemented algorithms were evaluated using constraints violation, simulation time and solution cost as performance metrics.

2.1 Problem Specification and Model Formulation

The goal of the High School Timetabling Problem in high school system in Nigeria (NHSTP) is to build a weekly timetable. The week is organized as a set of days D, and each day is split into a set of periods P. All Periods are distributed in D week days and H daily periods which occur during the same shift, that is, $P = D * H$. Let C be a set of classes (class-sections), T a set of teachers and S a set of subjects. A class $k \in C$ is a disjoint group of students that follow the same subjects, and no idle time periods during the week, and each subject of a class is taught by only one teacher who is previously determined. A *timeslot* is a pair, composed of a day and a class period (i,j), with $i \in D$ and $j \in$ P wherein all periods have the same duration. Teachers $l \in T$ may be unavailable in some timeslots.

The inputs for the NHSTP are a set of *events* (or meetings) E that must be scheduled, C*T matrix of lesson requirements R = (R_{lk}) where R_{lk} is the number of periods teacher l is required to meet class k for the duration of the timetable (containing a total of P periods in a set P), and teachers and classes unavailability's binary matrices T_{ij} and C_{kj}. Each *event* requires a class $k(e) \in C$, a teacher $l(e) \in T$, and a number of weekly hours $hours(e) \in$ N. Particularly in the Nigerian context, a teacher, a class, and a room are pre-assigned to each event e such that classrooms are not considered in the scheduling. In addition, each event defines how lessons are distributed over a week by requesting an amount of double lessons, restricting the daily limit of lessons, and defining whether lessons taught on the same day are consecutive or not.

A *feasible* timetable has a timeslot assigned to each lesson of events satisfying the hard constraint requirements HC1-HC13 below:

HC1: The number of lessons that each teacher must give to each class must be met (Lesson Requirement Constraint).

HC2: No two classes must be scheduled to the same teacher at the same period (Class Clashes Constraint).

HC3: No two teachers must be scheduled to the same class at the same period (Teacher Clashes Constraint).

HC4: A lesson cannot be scheduled to periods where the teacher is unavailable (Teachers' Unavailability Constraint).

HC5: A lesson cannot be scheduled to periods where the class is unavailable (Classes' Unavailability constraint).

HC6: Each class, for a given set of periods, must be involved in one lesson (One Class One Lesson Constraint).

HC7: Every teacher may be assigned at most one subject and one class (class-section) in a given period with the exception of indicated subjects that require more than one instructor (Uniqueness Constraint).

HC8: All subjects in the curriculum of a class-section should appear in the timetable for the required number of teaching periods (Completeness for Students Constraints).

HC9: All subjects assigned to a given teacher should appear in the timetable for the required number of teaching periods (Completeness for Teachers constraint).

HC10: The teaching periods assigned to a given subject over a whole week should add up to the weekly requirements for the specific subject (Completeness for Subjects Constraint).

HC11: Some pairs of lessons must be scheduled simultaneously (Simultaneity Constraint).

HC12: Certain subjects to be taught in multi-period slots at most once a day for a given class section must be followed (Consecutiveness constraint).

HC13: The timetable of every class-section should not carry empty slots during the week (Compact Student Schedules Constraint).

Besides feasibility regarding hard constraints, as many as possible of the soft requirements SC1-SC8 stated below should be satisfied:

SC1: More lessons to the same class in the same day than the maximum specified for the pair teacher-class must be avoided (Classes' maximum workload per day constraint).

SC2: The specified maximum number of daily lessons of each teacher must be respected (Teachers' maximum workload per day constraint).

SC3: Assignment of teachers to periods in which teachers would prefer not to teach must be avoided (Teachers' Preference or undesired Constraint).

SC4: Avoid teachers' idle periods. As much as possible, minimise the number of *empty or holes* periods between two consecutive periods where a teacher is assigned to a class. Breaks and free-time periods are not considered as idle periods or holes (Teachers' Idle Period Constraint).

SC5: Teachers' request for double lessons (lessons conducted in two consecutive periods) should be granted as often as possible (Double lesson per week constraint).

SC6: Teachers' request for parallelism of subjects (subjects scheduled for the same time periods) should be granted as often as possible (Parallelism of subjects Constraint).

SC7: Lessons for any given subject must be scheduled for at most one teaching period per day of the week, unless it requires more teaching periods than the days of the week or there is a special request for multiple or consecutive hours, in which cases they are scheduled accordingly (Uniform distribution of subjects Constraint).

SC8. Pre-assignments of certain subjects to specific time periods must be respected (Pre assignments of certain subjects Constraint).

The notation used in the problem specification and the formulated model for NHSTP considering all the hard and soft constraints requirements listed above are presented as follows.

Sets

$i \in D$ days of week. $D = \{1,2,...,|D|\}$.

$j \in P$ periods of a day. $P = \{1,...,|P|\}$.

P^1 P without the last two periods of a day. $P^1 = \{1,...,|P| - 2\}$.

$k \in C$ set of classes.

$l \in T$ set of teachers.

$m \in S$ set of subjects.

$e \in E$ set of events.

B set of quadrupes of the form $(l_1, k_1; l_2, k_2)$ with $l_1 \neq l_2$ and $k_1 \neq k_2$ such that all lessons of teacher l_1 to class k_1 must be simultaneous to lessons of teacher l_2 to class k_2

S_{kl} = $\{m \in S: m$ is a subject that teacher l teaches for class-section $k\}$

$S^*_{kl} = \{m \in S: m$ is any regular subject that teacher l teaches for class-section $k\}$, where the term "regular" refers to all subjects that do not require any special type of scheduling.

S_l^{Sim} = $\{(m, k, l^*): m \in S$ is a subject that teacher $l \in T$ teaches for a part of section $k \in K$ simultaneously with another subject taught by the "basic" teacher $l^*\}$

S_l^{Col} = $\{(m, k, l^*): m \in S$ is a subject that teacher $l \in T$ teaches for section $k \in K$ in collaboration with the "basic" teacher $l^*\}$

S_{cons} = {(m, k, h_m): $m \in S$ is a subject of section $k \in K$ that needs to be scheduled in block(s) of h_m consecutive periods}

S_{paral} = {[(m_a, m_b); (k_a, k_b); (l_a, l_b)]: $m_a \in S$ is a subject that teacher l_a teaches for section k_a that needs to be scheduled always in parallel to m_b a subject taught by teacher l_b for section k_b}

S_{excl} = {[(m_a, k_a, l_a; m_b, k_b, l_b; ...;m_w, k_w, l_w) (l_a)]:m_a, m_b, ..., $m_w \in S$ are subjects that teachers l_a,l_b,..., $l_w \in T$ teach to sections k_a, k_b, ..., $k_w \in K$, respectively, however no more than a certain number of them may be scheduled in the same day or time period}

S^{fix} = {(i_a, j_a, m_a, k_a, l_a): $m_a \in S$ is a subject that teacher l_a teaches to section k_a and should be scheduled on day i_a and in period j_a}

D_l = { $i \in D$: i is any day of the week for which teacher l is available for the school}

C_l = { $k \in C$: k is a class-section of the school to which teacher l teaches at least one subject}

Parameters

R_{lk}: The *workload* of an event (l,k), that is, the number of lessons that must be taught by the teacher l for the class k.

λ: The maximum number of permitted lessons per day

μ_m: Is the number of the multi-period slots (double lessons) required for subject m every week.

$\mu_{l,i}$: The maximum number of double lessons requested by teacher l with class k.

$\phi_{l,i}$: The effective number of allocated double lessons.

$\sigma_{k,l,i}$: The total number of lessons allocated for class k with teacher l on day i.

$\lambda_{k,l}$: The maximum number of permitted lessons per day.

$\pi_{l,j,i}$: The total number of lessons allocated for teacher l on period j of day i.

$\eta_{l,i}$: The number of idle times at the agenda of teacher l on day i

TTP: Total Time Periods. *TTP* indicates the total number of time periods to be scheduled in the timetable.

WTL_l: Weekly Teaching Load for teacher l. WTL_l indicates the total number of time periods to be assigned to teacher l each week.

DTL_l: Average Daily Teaching Load for teacher l. DTL_l indicates the daily average number of time periods assigned to teacher l.

WTL_{kl}: Weekly Teaching Load of teacher l for class-section k. WTL_{kl} indicates the total number of time periods to be assigned to teacher l for class-section k each week.

WTL_{klm}: Weekly Teaching Load of teacher l for class-section k and subject m. WTL_{klm} indicates the total number of time periods to be assigned to teacher l for subject m each week.

TTP_k: Total Time Periods for section k. TTP_k indicates the total number of time periods over all subjects of class-section k to be assigned each week.

H_k^{max}: Is a parameter that indicates the maximum number of teaching periods that section k may have during any day of the week. H_k^{max} may be set equal to J, the length of each day, however, in order to create more balanced timetables for the classes, it is preferable to set a different upper limit for each section of the school. Therefore, H_k^{max} equals to [TTP_k/D]. In general, however, it holds that $H_k^{max} \leq P, \forall\ k \in C$.

Variables

$x_{i,j,k,l,m}$: Binary variable that indicates whether subject m, taught by teacher l to the class section k, is scheduled for the j^{th} period of day i.

$\bar{x}_{i,j,k,l,m}$: Complement of the binary variable $x_{i,j,k,l,m}$, that is, $x_{i,j,k,l,m}$ = 0 if $\bar{x}_{i,j,k,l,m}$ = 1 and vice versa.

$y_{i,t_m,k,h_m,m}$: Binary variable that indicates whether subject m, is scheduled for h_m consecutive periods on day i for class-section k, with t_m being the 1st period for this assignment.

T_{ij}: binary variable that indicates whether teacher *l* is available to teach class *k* subject *m* scheduled for j^{th} period of day *i*.

C_{kj}: binary variable that indicates whether class *k* is available to be taught by teacher *l* subject *m* scheduled for j^{th} period of day *i*.

Z_{kj}: binary matrix that indicates whether class *k* must be taught a subject *m* by teacher *l* at j^{th} period of day *i*,

$l_{l,i,j}$: binary variable that indicates whether teacher *l* has been scheduled to teach at an undesired period *j* on day *i*.

X = An arbitrary matrix ($x_{i,j,k,l,m}$) is called a timetable. $X \in \{0, 1\}$

The objective function is to minimise the constraints (soft) violation that is formulated as solution cost function *Cf*(*s*), which associates a *cost* value to a given solution. Such value was used to compare the goodness of different solutions. This function was defined as follows: Let *S* be the search space; $s \in S$, a solution; *n*, the number of type of problem constraints considered, w_i, the penalty weighting associated with each constraint type *i* and $v_i(s)$, represents the number of constraint violations of type *i* in a solution *s*, $v_i(s) = 0$ if constraint type *i* is satisfied and $v_i(s) = 1$ if constraint type *i* is violated.

Therefore, the solution cost function *Cf*(*s*) is given as:

$$. Cf(s) = \sum_{i=1}^{n} w_i v_i(s) \tag{1a}$$

$$\textbf{Minimise} \sum_{i=1}^{n} w_i v_i(s) \tag{1b}$$

Subject to

$$\sum_{j=1}^{P} x_{i,j,k,l,m} = R_{lk} \qquad \forall i \varepsilon D, \forall j \varepsilon P \quad \forall l \varepsilon T, \forall k \varepsilon C \tag{2}$$

$$\sum_{k=1}^{C} x_{i,j,k,l,m} \leq 1 \qquad \forall i \varepsilon D, \forall j \varepsilon P \quad \forall l \varepsilon T, \forall m \varepsilon S \tag{3}$$

$$\sum_{l=1}^{T} x_{i,j,k,l,m} \leq 1 \qquad \forall i \varepsilon D, \forall j \varepsilon P \quad \forall k \varepsilon C, \forall m \varepsilon S \tag{4}$$

$$\sum_{k=1}^{C} x_{i,j,k,l,m} \leq T_{ij} \quad \forall i \varepsilon D, \forall j \varepsilon P \quad \forall l \varepsilon T, \forall m \varepsilon S \tag{5}$$

$$\sum_{l=1}^{T} x_{i,j,k,l,m} \leq C_{kj} \quad \forall i \varepsilon D, \forall j \varepsilon P \quad \forall k \varepsilon C, \forall m \varepsilon S \tag{6}$$

$$\sum_{l=1}^{T} x_{i,j,k,l,m} \geq Z_{kj} \quad \forall i \varepsilon D, \forall j \varepsilon P \quad \forall k \varepsilon C, \forall m \varepsilon S \tag{7}$$

$$\sum_{k \varepsilon C_l} \sum_{m \varepsilon S_{kl}^{*}} x_{i,j,k,l,m} + \sum_{(m,k,l^{*}) \varepsilon S_l^{sim} \cup S_l^{col}} x_{i,j,k,l^{*},m} \leq 1, \forall l \varepsilon T, \forall j \varepsilon P, \forall i \varepsilon D_l \tag{8}$$

$$\sum_{l \varepsilon T_k} \sum_{m \varepsilon S_{kl}} \sum_{i \varepsilon D_l} \sum_{j \varepsilon P} x_{i,j,k,l,m} + \sum_{l \varepsilon T_k} \sum_{(m,k,l^{*}) \varepsilon S_l^{sim} \cup S_l^{col}} \sum_{i \varepsilon D_l} \sum_{j \varepsilon P} x_{i,j,k,l^{*},m} = TTP_k, \forall k \varepsilon C \tag{9}$$

$$\sum_{i\varepsilon D}\sum_{j\varepsilon P}\sum_{m\varepsilon S_{kl}^{*}} x_{i,j,k,l,m} + \sum_{(m,k,l^{*})\varepsilon S_{l}^{sim}\cup S_{l}^{col}}\sum_{i\varepsilon D_{l}}\sum_{j\varepsilon P} x_{i,j,k,l^{*},m} = WLT_{kl}, \forall k\varepsilon C, \forall l\varepsilon T \tag{10}$$

$$\sum_{i\varepsilon D_{l}}\sum_{j\varepsilon P} x_{i,j,k,l,m} = WTL_{klm}, \forall k\varepsilon C, \forall l\varepsilon T_{k}, \forall m\varepsilon S_{kl} \tag{11}$$

$$x_{l_1k_1j}x_{l_2k_2j} + \bar{x}_{l_1k_1j}\bar{x}_{l_2k_2j} = 1 \quad \forall\ ([l_1; k_1; l_2; k_2] \in B;\ j\in P) \tag{12}$$

$$h_{m}^{*} y_{i,t_m,k,h_m,m} \le \sum_{j=t_m}^{t_m+h_m-1} x_{i,j,k,l,m} \le y_{i,t_m,k,h_m,m} + h_m - 1, \forall(m,k,h_m)\varepsilon S_{con,} \forall i\varepsilon D_{l}, t_m \le P - h_m + 1 \tag{13a}$$

$$\sum_{t_m=1}^{p-h_m+1} y_{i,t_m,k,h_m,m} \le 1, \forall(m,k,h_m)\varepsilon S_{con,} i\varepsilon D_{l} \tag{13b}$$

$$\sum_{i\varepsilon D_{l}}\sum_{t_m=1}^{p-h_m+1} y_{i,t_m,k,h_m,m} = \mu_m, \forall(m,k,h_m)\varepsilon S_{con} \tag{13c}$$

$$\sum_{l\varepsilon T_k}\sum_{m\varepsilon S_{kl}^{*}} x_{i,j,k,l,m} + \sum_{(m,k,l^{*})\varepsilon S_{l}^{sim}\cup S_{l}^{col}} x_{i,j,k,l^{*},m} = 1\forall k\varepsilon C, i\varepsilon D,\ j = 1,...., \left(H_{k}^{\max} - 1\right) \tag{14a}$$

$$\sum_{l\varepsilon T_k}\sum_{m\varepsilon S_{kl}^{*}} x_{i}, H_{k}^{\max}{}_{,k,l,m} + \sum_{(m,k,l^{*})\varepsilon S_{l}^{sim}\cup S_{l}^{col}} x_{i}, H_{k}^{\max}{}_{,k,l^{*},m} \le 1, \forall i\varepsilon D, \forall k\varepsilon C \tag{14b}$$

$$\sum_{l\varepsilon T_k}\sum_{m\varepsilon S_{kl}^{*}}\sum_{j\varepsilon P} x_{i,j,k,l,m} + \sum_{(m,k,l^{*})\varepsilon S_{l}^{sim}\cup S_{l}^{col}}\sum_{j\varepsilon P} x_{i,j,k,l^{*},m} \le H_{k}^{\max}, \forall i\varepsilon D, \forall k\varepsilon C \tag{14c}$$

$$\sum_{k\varepsilon C_{l}}\sum_{l\varepsilon T_{l}}\sum_{i\varepsilon D} x_{i,j,k,l,m} \le \sigma_{k,l,i}\forall k\varepsilon C, \forall l\varepsilon T, \forall i\varepsilon D \tag{15}$$

$$\sum_{l\varepsilon T_{l}}\sum_{i\varepsilon D_{l}}\sum_{j\varepsilon P} x_{i,j,k,l,m} \le \pi_{l,j,i}\forall l\varepsilon T, \forall j\varepsilon P, \forall i\varepsilon D \tag{16}$$

$$\sum_{l\varepsilon T_{l}}\sum_{i\varepsilon D_{l}}\sum_{j\varepsilon P} x_{i,j,k,l,m} \le l_{l,i,j}\forall l\varepsilon T, \forall i\varepsilon D_{k}, \forall j\varepsilon P \tag{17}$$

$$\sum_{l\varepsilon T}\sum_{i\varepsilon D}\eta_{l,i} \le 1 \quad \forall l\varepsilon T, \forall i\varepsilon D \tag{18}$$

$$\phi_{l,i} \le \sum_{l\varepsilon T}\sum_{i\varepsilon D}\mu_{l,i}, \forall l\varepsilon T, \forall i\varepsilon D, \forall\left(\mu_{l,i} > \phi_{l,i}\right) \tag{19}$$

$$x_{i,j,k_1,l_1,m_1} - x_{,j,k_2,l_2,m_2} \le 0, \forall\left[(m_1,m_2);(k_1,k_2);(l_1,l_2)\right]\varepsilon S_{paral}, \forall i\varepsilon D, \forall i\varepsilon P \tag{20}$$

$$\sum_{j\varepsilon P} x_{i,j,k,l,m} \le 1, \forall i\varepsilon D, \forall k\varepsilon C, \forall l\varepsilon T_{k}, \forall m\varepsilon S_{kl} \tag{21}$$

$$x_{i,j,k,l,m} = 1, \forall(i,j,k,l,m)\varepsilon S_{fix} \tag{22}$$

$$x_{i,j,k,l,m} = 0 \text{ or } 1 \quad \forall i\varepsilon D, \forall j\varepsilon P, \forall k\varepsilon C\ \forall l\varepsilon T, \forall m\varepsilon S \tag{23}$$

Constraint set (2) ensures that the number of lessons that each teacher must give to each class is fully scheduled. Constraint set (3) ensures that no two classes are scheduled to the same teacher at the same period. Constraint set (4) ensures that no two teachers are scheduled to the same class at the same period. Constraint set (5) ensures that a lesson cannot be scheduled to periods where the teacher is unavailable. Constraint set (6) ensures that a lesson cannot be scheduled to periods where the class is unavailable. Constraint set (7) ensures that each class, for a given set of periods, are scheduled to only one lesson at a time. Constraint set (8) ensures that every teacher is assigned at most one subject and one class (class-section) in a given period with the exception of indicated subjects that require more than one instructor. Constraint set (9) ensures that all subjects in the curriculum of a class-section appeared in the timetable for the required number of teaching periods. Constraint set (10) ensures that all subjects assigned to a given teacher appeared in the timetable for the required number of teaching periods.

Constraint set (11) ensures that the teaching periods assigned to a given subject over a whole week added up to the weekly requirements for the specific subject. Constraint set (12) ensures that some pairs of lessons are scheduled simultaneously. Constraint set (13) ensures that certain subjects to be taught in multi-period slots at most once a day for a given class section are followed. Constraint (13a) forces h_m basic variables $x_{i,j,k,l,m}$ that refer to consecutive time periods to take the value of 1, while constraint (13b) ensures that only one block of consecutive periods may be assigned in any given day and constraint (13c) indicates that there should be exactly μ_m of these blocks for the whole week. Constraint set (14) ensures that the timetable of every class-section did not carry empty slots during the week. Basically Constraint set (14a) and (14b) ensure that for each class-section there is exactly one subject scheduled for any given period (except may be the last one) of any day, while Constraint (14c) on the other hand, checks whether all subjects of class-section *k* are scheduled within the maximum stretch allowed for the class-section.

Constraint set (15) ensures that the limit set on the maximum number of lessons a class may have per day is met. Constraint set (16) ensures that the specified maximum number of daily lessons of each teacher is respected. Constraint set (17) ensures the assignment of teachers to periods in which teachers would prefer not to teach is avoided. Constraint set (18) determines the number of teachers' idle periods in a solution. Constraint set (19) ensures that teachers' request for double lessons is granted. Constraint set (20) ensures that teachers' request for parallelism of subjects is granted. Constraint set (21) ensures that uniform distribution of subjects is met. Constraint set (22) ensures pre-assignments of certain subjects to specific time periods are respected while Constraint set (23) is required to ensure the integrality of the solution.

In this work each hard constraint was assigned a weight of 20 to stipulate their higher priority than the soft constraints and to allow the proposed solution leads the search process towards valid solutions in accordance to the literature [59]. The weight assigned to each of the *soft constraints (preferences)* varies to indicate the relative importance of each preferences compared to others such that a weight of 6 was assigned to *SC1*, *SC2*, *SC5* and *SC6*; a weight of 4 was assigned to *SC7* and *SC8*; a weight of 3 was assigned to *SC3* and finally a weight of 1 was assigned to *SC4*.

These weight set was informed by the literature which stated that weighted penalty based evaluation function should be used for timetabling problems where an abundance of different constraint combinations is encountered [60,61,62] and thus allows for some constraints to have a higher priority than others. It must be noted that (i) the lower the value of *Cf*(*s*) for a given solution *s*, the more the quality of *s*, (ii) both the distance to feasibility and the goodness of the solution was measured.

2.2 Formulation of Enhanced Simulated Annealing (ESA) Algorithm

The simulated annealing (SA) algorithm was enhanced to form Enhanced Simulated Annealing (ESA) Algorithm through the following three stages:

(1) Modification of Simulated Annealing (SA) in terms of the temperature reduction parameter α in order to improve its efficiency in terms of convergence speed and solution cost, by introducing a parabolic reduction parameter α ($\alpha = (1/log(1+t+t^2))$ as suggested by [2]. This is due to the fact that the efficiency of SA

often depends on cooling schedule and by carefully controlling the rate of cooling the temperature; SA can find the global optimum exponential faster [63,64].

(2) Integration with Genetic Algorithm (GA) in order to: (i) further improve the performance of SA in terms of speed and quality of solution as suggested by [65]; (ii) find a good balance between the exploitation of found-so-far elements and the exploration of the search space in order to find global optimal solution as suggested by [66] and (iii) improve its local search ability as suggested by [67]. By this enhancement (integration), the genetic operators of GA were applied to observe the behaviour of Simulated Annealing which resulted into less parameter to control. In tune with the principle of the integration, new individuals were produced with Genetic Algorithm (GA) after which these individuals were processed with SA while the corresponding results were further used as the new individuals of the next generation.

(3) Reordering the sequence of evolutionary operations to become (mutation, selection and crossover) instead of (selection, crossover and mutation) in order to further reduce the convergence time and as well as to generally improve its computational efficiency as reported by [68,69].

The above sequence of processes resulted into Enhanced Simulated Annealing (ESA) algorithm. The algorithmic structure of ESA algorithm that was coded using the MATLAB Laboratory 8.6 software is as presented as follows:

Step 1: Initialize the temperature parameter T, that is, set $T = T_0$, in which T_0 is a large positive number, set the exponential temperature reduction factor as $\alpha = (1/log(1+t+t^2))$, and final temperature as $T_{t+1} = \alpha T_t$

Step 2: Produce the initial population made up of n individuals.

Step 3: Compute the fitness of each individual in the initial population.

Step 4: Repeat:

Step 4.1: Carry out evolutionary operations, that is, mutation, selection and crossover, for individuals in the current solution population.

Step 4.2: Let C_j be the child individual produced by a parent individual P_j, $j = 1, \ldots, n$.

Step 4.3: Compute the fitness $E(C_j)$ of new individual C_j, $j = 1, \ldots, n$.

Step 4.4: For $j = 1, \ldots, n$, compute $\Delta E = E(C_j) - E(P_j)$, where $E(P_j)$ is the fitness of parent individual P_j. If $\Delta E \geq 0$, then produce a number r with uniform distribution in interval [0,1]. If exp $(-\Delta E/T) > r$, then replace individual P_j by child individual C_j. If $\Delta E < 0$, then discard the child individual C_j.

Step 4.5: If the termination condition is satisfied, then the whole procedure is stopped, else the value of temperature T is decreased.

The algorithmic structure of SA algorithm that was coded using the MATLAB Laboratory 8.6 software is as presented as follows:

Step 1: Generate an initial schedule S.

Step 2: Set the initial best schedule $S^* = S$.

Step 3: Compute cost of S: C(S).

Step 4: Compute initial temperature T_0.

Step 5: Set the Initial Temperature $T = T_0$, set Parabolic Temperature reduction factor as $\alpha = (1/log(1+t))$, and Final Temperature as $T_{t+1} = \alpha T_t$

Step 6: While stop criterion is not satisfied do.

(a) Repeat Markov Chain Length (M) times:

i. Select a random neighbor S^1 to the current schedule, ($S^1 \in N_S$).
ii. Set Δ (C) = C(S^1) – C(S).
iii. If (Δ (C) ≤ 0 {downhill move}).
   - Set $S = S^1$
   - If C(S) < C(S^*) then set $S^* = S$
iv. If (Δ (C) > 0 {uphill move}).
   - Choose a random number r uniformly from [0, 1]
   - If $r < e^{-\Delta(C)/T}$ then set $S = S^1$

(b) Reduce (or update) temperature T.

Step 7: Return the Schedule S^*

3. RESULTS AND DISCUSSION

The Simulated Annealing (SA) algorithm and the developed Enhanced Simulated Annealing (ESA) algorithm were tested with the highly constrained school timetabling dataset provided by a Nigerian high school. The algorithms were tested under several optimization runs. The results of

performance evaluation of both algorithms (SA and ESA) are as shown in Table 1.

As depicted in Table 1, both the average simulation time and solution cost to generate a high quality timetable in SA algorithm is higher than that to generate an optimal timetable in the developed ESA algorithm. The higher simulation time and solution cost of SA algorithm can be attributed to the slow convergence speed and exponential cooling schedule inherent in SA algorithm. The modified annealing schedule (slow cooling schedule - parabolic), improved local search ability and the reordered evolutionary operation sequence in the developed ESA algorithm helped to improve the convergence speed which in turns reduced the average simulation time and solution cost to generate an optimal timetable in the developed ESA algorithm.

This result confirmed the previous literatures that indicated that: high quality solutions can only be obtained if SA's parameters (cooling schedule, update moves, initial solution, among others) are well tuned and that (i) the efficiency of SA algorithm depend on cooling schedule and by carefully controlling the rate of cooling the temperature SA can find the global optimum exponential faster [63,64]; (ii) the performance of SA in terms of speed, quality of solution (global optimal solution) and local search ability can be improved by integrating it with GA [65,66,67]; and (iii) the convergence time and computational efficiency of GA-based algorithm can be improved by reordering the sequence of evolutionary operations to become (mutation, selection and crossover) instead of (selection, crossover and mutation) [68,69].

The following section gives the detailed result of the performance evaluation of the two solution approaches (SA and the developed ESA):

(i) Constraints violation: Constraints violation is the metric that measures the feasibility and optimality of the solution produced by an algorithm. An algorithm that satisfies the problem's all *hard constraints* is said to produce a feasible solution. The optimality (goodness/quality) of an algorithm's solution is indicated by how much soft constraints an algorithm satisfied. As depicted in Table 1, SA algorithm produced high quality timetable as a result of violation of one of the soft constraints while the developed ESA algorithm produced optimal solution as it satisfied all the specified constraints (hard and soft).

(ii) Simulation Time: Simulation time is the parameter which measures the time utilized by is an algorithm to run until the result is produced. It otherwise known as computation, execution or run time. As advocated by [73], simulation time should be considered first when dealing with the performance evaluation of optimization algorithms for combinatorial problems. It should be a key element of any such evaluation [74]. Indicated that one of the most important factors considered before choosing the winner during the second international timetabling competition (ITC-2007) was the simulation time.

Table 1 showed the obtained values of the simulation time of both SA algorithm and the developed ESA algorithm for JSS and SSS respectively. The simulation time of the SA algorithm and the developed ESA algorithm are 40.90 and 37.91 seconds respectively for JSS and 45.82 and 42.16 seconds respectively for SSS. This is clear evidence that the developed ESA algorithm utilized less time and converges faster than the SA algorithm to produce an optimal timetable as a result of its enhanced features (slow cooling schedule - parabolic, improved local search ability and the reordered evolutionary operation sequence).

Table 1. Constraints violation, simulation time and solution cost evaluation result

Parameters	SA	ESA
Junior Secondary School		
Number of Hard Constraint Violated	0	0
Number of Soft Constraint Violated	1	0
Average Simulation time (Seconds)	40.90	37.91
Average Solution Cost	20.88	17.03
Senior Secondary School		
Number of Hard Constraint Violated	0	0
Number of Soft Constraint Violated	1	0
Average Simulation time (Seconds)	45.82	42.16
Average Solution Cost	23.43	18.99

It was observed that the developed ESA yielded high convergence speed which resulted into the lower average simulation time in producing optimal solution, but utilised more time to compute the timetables for SSS classes than JSS classes because of the different subject groups that exist in SSS, such as Science group, Commercial group and Art group.. This result confirmed the literatures that by carefully controlling the rate of cooling the temperature, SA can find the global optimum exponential faster since slow cooling schedules are generally more effective [63] and that the performance of SA in terms of speed and quality of solution can be improved by integrating it with GA [65,67].

(iii) Solution cost: The quality of a timetable is defined by a solution cost, otherwise known as fitness value or solution quality function. The fitness function calculates the number of constraint breaches (usually with a weighted value) for different constraints. This value is used as measurement for quality of the timetable generated by the algorithms. It is used to compare the goodness of different solutions as better timetables are produced the better fitness values emerges. It is implicitly defined through the school timetabling problem specification and constraints as given in Equation 1a. The lower the value of solution cost for a given solution the more the quality of the solution [70].

Table 1 showed the measured values of the solution cost of the two algorithms (SA and the developed ESA) for Junior Secondary School (JSS) and Senior Secondary School (SSS) respectively. The average solution cost values of SA algorithm and the developed ESA algorithm were 20.88 and 17.03 respectively for JSS, and 23.43 and 18.99 respectively for SSS. This is clear evidence that the developed ESA returned the best optimal solution (with lower solution cost value), but with higher value for SSS because of the somewhat more complicated structure of student groups and the demand for compact scheduling in SSS than JSS. This is attributed to the slow temperature reduction component (parabolic cooling rate of the developed ESA algorithm) since the solution cost is generally improved with slow cooling rate. This result confirmed the literatures that the choice of the cooling schedule influences the quality of solution obtained with SA [71,72] and that that slow cooling schedules are generally more effective, and also that the solution cost generally improves with slower cooling rates [63].

4. CONCLUSION

The high school timetabling is a classical combinatorial optimization problem that takes a large number of variables and constraints into account. Due to its combinatorial nature, solving medium and large instances to optimality is a challenging task. Specifically, school timetabling problem is perhaps the most difficult problem which high schools face in Nigeria. In this paper, we presented a novel solution approach for solving a highly constrained school timetabling problem which characterizes the problem-setting in the timetabling problem of the high school system in Nigeria which has not been completely addressed in the literature. We presented a new mathematical programming model of timetabling for high schools in Nigeria using mixed integer linear programming formulation for which the current timetable is considerably improved. In addition, we presented a meta-heuristic method, an Enhanced Simulated Annealing (ESA) algorithm that incorporates specific features of Simulated Annealing and Genetic Algorithms for solving the problem. Results show that our approach provides high quality solutions (optimal timetables) in smaller computational time and solution cost when compared with results obtained with SA approach.

The results of this work confirmed previous research reports that high quality solutions can be obtained if SA's parameters are well tuned as the analysis of the results showed that though both the SA algorithm and developed ESA algorithm yielded better quality solutions when compares to manual allocation procedures but the developed ESA algorithm yielded higher quality solutions. The observed high quality solutions provided by the developed ESA algorithm resulted from its tuned parameters - modified annealing schedule (slow cooling schedule - parabolic), improved local search ability and the reordered evolutionary operation sequence. Furthermore, the result shows that the developed solution method (ESA algorithm) is very promising to solve the school timetabling problem, motivating its use to variants of this problem, as well as to other general combinatorial optimization problems.

A possible future research area is to develop other solution methods that might solve the problem more efficiently. For example, the efficiency of the developed solution method can be improved by incorporating it within an evolutionary algorithm or a cultural algorithm..

Another future research area is to incorporate additional requirements that might be required by other high schools and then solve the resulting problem using the developed solution method. It is also possible to consider extending the developed solution method to solve other types of timetabling problems.

COMPETING INTERESTS

Authors have declared that no competing interests exist.

REFERENCES

1. Melício F, Caldeira P, Rosa A. THOR: A tool for school timetabling. In: Burke EK, Rudova H, editors, PATAT. 2006;6:532–535.
2. Odeniyi OA, Omidiora EO, Olabiyisi SO, Aluko JO. Development of a modified simulated annealing to school timetabling problem. International Journal of Applied Information Systems. 2015;8(2):16-24.
3. Odeniyi OA, Oyeleye CA, Olabiyisi SO, Omidiora EO, Makinde BO, Aluko JO, et al. School timetabling: Solution methodologies and applications. International European Extended Enablement in Science, Engineering & Management. 2020;8(3):35-62.
4. Tassopoulos IX, Beligiannis GN. Solving effectively the school timetabling problem using particle swarm optimization. Expert Systems with Applications. 2012;39(5): 6029-6040.
5. Pillay N. A survey of school timetabling research. Annals of Operations Research. 2014;218(1):261-293.
6. Demirovic E. SAT-based approaches for the general high school timetabling problem. Unpublished PhD dissertation, Vienna PhD School of Informatics, TU Wien; 2017.
7. Sørensen M, Stidsen TR. Comparing solution approaches for a complete model of high school timetabling. Technical report, DTU Management Engineering; 2013a.
8. Sorensen M, Dahms FHW. A two-stage decomposition of high school timetabling applied to cases in Denmark. Computers & Operations Research. 2014;43:36-49.
9. Demirovic E, Musliu N. Modeling high school timetabling as partial weighted maxsat. LaSh 2014: The 4th Workshop on Logic and Search (a SAT/ICLP workshop at FLoC 2014); 2014a.
10. Demirovic E, Musliu N. Solving high school timetabling with satisfiability modulo theories. In: Ozcan E, Burke EK, McCollum B, editors. Proceedings of the 10th International Conference of the Practice and Theory of Automated Timetabling. 2014b;142-166.
11. Kristiansen S, Sørensen M, Stidsen TR. Integer programming for the generalized high school timetabling problem. Journal of Scheduling. 2015;18(4):377–392
12. Demirovic E, Musliu N. Modeling high school timetabling with bitvectors. Ann Oper Res; 2016.
DOI: 10.1007/S10479-016-2220-6
13. Al-Yakoob SM, Sherali HD. A mixed-integer programming approach to a class timetabling problem: A case study with gender policies and traffic considerations. European Journal of Operational Research. 2007;180(3):1028-1044.
14. Al-Yakooba SM, Sheralib HD. Mathematical models and algorithms for a high school timetabling problem. Comput. Op. Res. 2015;61:56–68.
15. Schaerf A. A survey of automated timetabling. Artificial Intelligence Review. 1999;13(2):87–127.
16. Post G, Kingston J, Ahmadi S, Daskalaki S, Gogos C, Kyngas J, et al. XHSTT: an XML archive for high school timetabling problems in different countries. Ann Oper Res. 2011;1–7.
17. Post G, Ahmadi S, Daskalaki S, Kingston J, Kyngas J, Nurmi C, Ranson D. An xml format for benchmarks in high school timetabling. Annals of Operations Research. 2012a;194:385-397.
18. Post G, Gaspero LD, Kingston JH, McCollum B, Schaerf A. The third international timetabling competition. Proceedings of the Ninth International Conference on the Practice and Theory of Automated Timetabling (PATAT 2012), Son, Norway; 2012b.
19. Birbas T, Daskalaki S, Housos E. School timetabling for quality student and teacher schedules. Journal of Scheduling. 2009; 12(2):177-197.
20. Dorneles AP, de Araújo OCB, Buriol LS. The impact of compactness requirements on the resolution of high school timetabling problem. Pre-anais: XVI CLAIO - XLIV SBPO - LIASGT, Rio de Janeiro. 2012; 3336–3347.

21. Appleby JS, Blake DV, Newman EA. Techniques for producing school timetables on a computer and their application to other scheduling problems. The Computer Journal. 1961;3(4):237-245.
22. Gotlieb C. The construction of class-teacher timetables. In: Popplewell CM, editor, Proceedings of IFIP Congress (Munchen 62), North Holland, Amsterdam. 1962;73-77.
23. Pillay N. An overview of school timetabling research. Proceedings of the International Conference on the Theory and Practice of Automated Timetabling, Belfast, United Kingdom. 2010;321-335.
24. Schmidt G, Ströhlein T. Timetable construction – an annotated bibliography. Computer Journal. 1979;23(4):307-316.
25. Bardadym V. Computer-aided school and university timetabling: The new wave. In: Burke E, Ross P, editors, Practice and Theory of Automated Timetabling, volume 1153 of Lecture Notes in Computer Science. Springer: Berlin / Heidelberg. 1996;22-45.
26. Qu R, Burke EK, McCollum B, Merlot LTG, Lee SY. A survey of search methodologies and automated system development for examination timetabling. Journal of Scheduling. 2009;12(1):55-89.
27. Gendreau M, Burke E, editors. PATAT2008: Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling; 2008.
28. McCollum B, Burke E, White G. editors. PATAT: Proceedings of the 8th International Conference on the Practice and Theory of Automated Timetabling; 2010.
29. Kjenstad D, Riise A, Nordlander TE, McCollum B, Burke E. editors. PATAT 2012: Proceedings of the 9th International Conference on the Practice and Theory of Automated Timetabling; 2012.
30. de Werra D. An introduction to timetabling. European Journal of Operational Research. 1985;19(2):151-162.
31. Lawrie NL. An integer linear programming model of a school timetabling problem. The Computer Journal.1969;12(4):307–316.
32. Lodi A. Mixed integer programming computation. In: 50 years of integer programming 1958–2008. Berlin, Germany: Springer. 2010;619-645.
33. Souza M, Ochi L, Maculan N. A GRASP-Tabu search algorithm for solving school timetabling problems. In Metaheuristics: Computer Decision Making, Kluwer Academic Publishers, Boston. 2003;659–672.
34. Soza C, Becerra RL, Riff M C, Coello, CAC. A cultural algorithm with operator parameters control for solving timetabling problems. Lecture Notes in Computer Science. 2007;4529:810–819.
35. Santos HG, Toffolo TAM, Brito SS, Souza MJF, Fonseca GHG. A SA-ILS approach for the high school timetabling problem. Proceedings of the Ninth International Conference on the Practice and Theory of Automated Timetabling (PATAT 2012), Son, Norway; 2012
36. Sorensen M, Stidsen TR. Hybridising integer programming and meta-heuristics for solving high school timetabling. In comparing solution approaches for a complete model of high school timetabling. DTU Management Engineering. DTU Management Engineering Report, No. 5; 2013b.
37. Raghavjee R. Study of genetic algorithms for solving the school timetabling problem; Unpublished M.Sc. Dissertation, University of Kwazulu-Natal, Pietermaritzburg; 2013.
38. Kristiansen S. Solving Multiple Timetabling Problems at Danish High Schools. Unpublished PhD Thesis, Department of Management Engineering, Technical University of Denmark; 2014.
39. Dorneles AP. A matheuristic approach for solving the high school timetabling problem. Unpublished Ph.D. Thesis, Universidade Federal do Rio Grande do Sul, Porto Alegre; 2015.
40. Raghavjee R, Pillay N. A genetic algorithm selection perturbative hyperheuristic for solving the school timetabling problem. Orion. 2015;31(1):39-60.
41. Bogdanov D. A comparative evaluation of meta-heuristic approaches to the problem of curriculum-based course timetabling. Unpublished Bachelor's Thesis, KTH Royal Institute of Technology, School of Computer Science and Communication; 2015.
42. Fonseca G, Brito S, Santos H. A simulated annealing based approach to the high school timetabling problem. Lecture Notes in Computer Science. 2012a;7435:540-549.

43. Chen DJ, Lee CY, Park CH, Mendes P. Parallelizing simulated annealing algorithms based on high-performance computer. Journal of Global Optimization. 2007;39(2):261-289.
44. Smith KI, Everson EM, Fieldsend JE. Dominance based multi-objective simulated annealing. IEEE Transactions on Evolutionary Computation. 2008;12(3): 323-342.
45. Omidiora EO, Olabiyisi SO, Arulogun OT, Oyeleye CA, Adegbola A. A prototype of an access control system for a computer laboratory scheduling. Proceedings of AICTTRA 2009, Obafemi Awolowo University, Ile Ife. 2009;114-120.
46. Janaki Ram D, Screenivas TH, Ganapathy SK. Parallel simulated annealing algorithm. Journal of Parallel and Distributed Computing. 1996;37:207-212.
47. Olabiyisi SO, Fagbola TM, Omidiora EO, Oyeleye AC. Hybrid meta-heuristic feature extraction technique for solving timetabling problem. International Journal of Scientific & Engineering Research. 2012;3(8):1-6.
48. Oyeleye CA, Olabiyisi SO, Omidiora EO, Fagbola T. Hybrid metaheuristic of simulated annealing and genetic algorithm for solving examination timetabling problem. International Journal of Computer Science and Engineering. 2014;3(5):7-22.
49. Delport V. Parallel simulated annealing and evolutionary selection for combinatorial optimisation. Electronic Letters. 1998;34:758-759.
50. Kanoh H, Nakamura T. Knowledge based genetic algorithm for dynamic route selection. Proceedings of Knowledge-Based Intelligent Engineering Systems and Allied Technologies. 2000;2:616-619.
51. Zhou J, Bai T, Tian J, Zhang A. The study of SVM optimized by culture genetic algorithm on predicting financial distress. Proceedings of IEEE International Conference on CSIT. 2008;524-528.
52. Liogys M, Žilinskas A. On multi-objective optimization heuristics for nurse rostering problem. Baltic Journal of Modern Computing. 2014;2(1):32-44.
53. Dorneles AP, de Araújo OCB, Buriol LS. A fix-and-optimize heuristic for the high school timetabling problem. Computers & Operations Research. 2014;52:29-38.
54. Burke E, Kingston J, Pepper P. A standard data format for timetabling instances. In: Burke E, Carter M, editors, Practice and Theory of Automated Timetabling II, volume 1408 of Lecture Notes in Computer Science,. Springer: Berlin /Heidelberg. 1998;213-222.
55. Asratian AS, de Werra D. European Journal of Operational Research. 2002; 143(3):531-542.
56. Özcan E. Towards an XML-based standard for timetabling problems: Ttml. In Kendall G, Burke EK, Petrovic S, Gendreau M, editors, Multidisciplinary Scheduling: Theory and Applications Springer: US. 2005;163-185.
57. Causmaecker PD, Berghe G. Towards a reference model for timetabling and rostering. Annals of Operations Research. 2010;1-10
58. Bonutti A, De Cesco F, Gaspero LD, Schaerf A. Benchmarking curriculum-based course timetabling: formulations, data formats, instances, validation, visualization and results. Annals of Operations Research. 2010;1-12.
59. Cedeira-Pena A, Carpente L, Farina A, Seco D. New approaches for the school timetabling problem. Proceedings of the 7th Mexican conference on artificial intelligence (MICAI 2008). 2008;261-267.
60. Schaerf A. Tabu search techniques for large high school timetabling problems. Proceedings of 13th National Conference on Artificial Intelligence (AAAI-96), Portland, USA. 1996;363-368.
61. Wright M. School timetabling using heuristic search. Journal of the Operational Research Society. 1996; 47(3):347-357.
62. Lewis R. A survey of metaheuristic-based techniques for university timetabling problems. OR Spectrum. 2008;30(1):167-190.
63. Thompson JM, Dowsland KA. A robust simulated annealing based examination timetabling system. Computers and Operational Research. 1998;25(7/8):637-648.
64. Aarts E, Kort J, Michiels W. Simulated Annealing. In Burke E, Kendall G (editors), Search Methodologies: Introductory Tutorials in Optimization and Decision Support and Search Techniques. Springer. 2005;7:187-211.
65. Delport V. Parallel simulated annealing and evolutionary selection for combinatorial optimisation. Electronic Letters. 1998;34:758-759.
66. Tan KC, Chiam SC, Mamun AA, Goh CK. Balancing Exploration and Exploitation

with Adaptive Variation for Evolutionary Multi-Objective Optimization. European Journal of Operational Research. 2009; 197(1):701-713.

67. Kanoh H, Nakamura T. Knowledge based genetic algorithm for dynamic route selection. Proceedings of Knowledge-Based Intelligent Engineering Systems and Allied Technologies. 2000;2:616-619.
68. Angelova M, Pencheva T. Tuning genetic algorithm parameters to improve convergence time. International Journal of Chemical Engineering, ID 646917; 2011. Available:http://www.hindawi.com/journals/ijce/2011/646917
69. Angelova M, Atanassov K, Pencheva T. Multi-population genetic algorithm: Quality assessment implementing intuitionistic fuzzy logic. Proceedings of Federated Conference on Computer Science and Information Systems. 2012;365-370.
70. Kohonen J. A brief comparison of simulated annealing and genetic algorithm approaches. Term Paper for the Three Concepts Utility Course, Department of Computer Science, University of Helsinki; 1999.
71. McCollum B. University timetabling: bridging the gap between research and practice. Proceedings of the 6th International Conference on Practice and Theory of Automated Timetabling. 2006; 15-35.
72. Reis LP, Oliveira E. A language for specifying complete timetabling problems. Lecture Notes in Computer Science. 2001; 2079:322-341.
73. Johnson D, Aragon C, Mcgeoch L, Schevon C. Optimization by simulated annealing: An experimental evaluation, Part 1: Graph Partitioning. Operational Research. 1990;37:865-892.
74. Kohonen J. A brief comparison of simulated annealing and genetic algorithm approaches. Term Paper for the Three Concepts Utility Course, Department of Computer Science, University of Helsinki; 1999.

Peer-review history:
The peer review history for this paper can be accessed here:
http://www.sdiarticle4.com/review-history/55744