



Development of a Modified Simulated Annealing to School Timetabling Problem

Odeniyi, O. A.

Department of Computer
Science, Osun State College Of
Technogy, Esa-Oke, Nigeria.

Omidiora, E. O.

Dept. of Computer Science and
Engineering, Ladoke Akintola
Univwesity of Technology,
Ogbomoso, Nigeria

Olabiyisi, S. O.

Dept. of Computer Science and
Engineering, Ladoke Akintola
Univwesity of Technology,
Ogbomoso, Nigeria

Aluko, J. O.

Directorate of Information and
Communication Technology, Osun State
College of Education, Ilesa,
Osun State, Nigeria.

ABSTRACT

This work presents a modified simulated annealing applied to the process of solving a typical high school timetabling problem. Preparation of a high school timetable consists basically of fixing a sequence of meetings between teachers and students in a prefixed period of time in such a way that a certain set of constraints of various types is satisfied. The approach presented in the paper has been successfully used to schedule the first time school timetable of Fakunle Comprehensive High School, Osogbo Nigeria during the 2012/2013 session and it was capable of generating timetables for complex problem instances. A task involving 18 Classes, 45 Teachers and 15 Subjects for Junior Secondary School (JSS) with 3 Levels (JSS 1 to JSS 3), and 6 arms each; and 24 Classes, 77 Teachers and 19 Subjects for Senior Secondary School (SSS), with 3 Levels (SSS 1 to SSS 3), and 8 arms (3 for Science Group, 3 for Commercial Group, and 2 for Art Group), for 6 hours, 5 days respectively. The use of the implemented model resulted in significant time saving in the scheduling of the timetables, and a well spread lessons for the teachers. Also none of the teachers and classes was double booked. It was clearly evident that the developed modified simulated annealing reduces the major weakness of slow convergence (convergence at excessive time) associated with the classical simulated annealing.

General Terms

School Timetabling, Meta-heuristics.

Keywords

NP-Complete, Constraint Satisfaction, Combinatorial Optimization, Scheduling, Modified Simulated Annealing.

1. INTRODUCTION

Timetabling problems have been the subject of active research in recent time [1, 2], and [3] reports that timetabling problems have attracted substantial research interests due to its importance in a wide variety of application domains, including Education [4], Employee/Staff Rostering [5], Healthcare Institutions [6], Sports events [7], Interview scheduling [8], and many other constraint satisfaction problems. This diversity and its relevance in practice made timetabling an active research area in operations research [9].

Timetabling problems arising from educational institutions can be classified into three main groups: university examinations timetabling, university course timetabling and school timetabling problems. Most of the research on educational timetabling has concentrated on course timetabling and examination timetabling at the university level [10, 11], and timetabling researchers have obtained very promising and practicable results in these problem areas.

However, school timetabling problem has not been as extensively studied as examination timetabling and course timetabling at the university, and widely usable results are not yet available due to lack of standard benchmarks and data formats [12]. This is also because school timetabling problem has a large number of variants or structure depending on the country (laws), the type of school, and even on the specific school involved (school policies) and other particular variables [13]. In reality, the research in this area is still very active and researchers are not even near to solving all the instances of the school timetabling problem to optimality. In addition, continuous reforms in educational systems all over the world generate new applications to tackle [14].

The school timetabling problem is the process of weekly scheduling of lessons for high schools, and involves fixing a sequence of meetings between teachers and students in a prefixed period of time in such a way that a certain set of constraints of various types is satisfied. A typical educational timetable instance requires several days of work for a manual solution [15]. Generating School timetables manually is time consuming, requires several days or weeks [16], and tedious to do since it often involves numerous rounds of changes before they can be satisfactory. Usually such a process takes several days, and often the quality of the timetables is compromised due to pressure to release the timetables on time [17]. Another major problem in manually generating school timetables is dealing with clashes, and finding clash-free or conflict-free slots, as making a change requires that one has to undo previous lesson allocation and look for a new allocation. This creates a series of backtracks which are difficult to resolve. Also, as the solution obtained may be unsatisfactory in some respect because it is a highly complex task to verify all constraints, considerable attention has been devoted to automated timetabling. Also due to inherent problem complexity (with a large number of events needing to be assigned to resources and a list of constraints needing to be



satisfied), and variability (this problem has a varying structure in different high schools even in the same countries or educational systems), the search space for this problem is very large.

For the above reasons, several techniques or approaches have been developed to automatically solve the problem [18, 19, 20]. A concise overview of some general trends and recent state in automated timetabling can be found in a variety of papers [21, 22, 23]. Some of their concepts and/or methods can be used in real-life timetabling problems.

For the school timetabling field, solutions that are based on the use of *Genetic Algorithms* [24], *Tabu Search* [25], *Constraint Programming* [26], *Simulated Annealing* [27], *Meta-heuristics* [28], are found but they meet neither the requirements nor the structure of school timetabling problem solved in this study.

This is because this problem has a large number of variants depending on the country (laws), on the type of school, and even on the specific school involved (school policies) and other particular variables [13]. The school timetabling problem we consider in this paper comes from the Nigerian high-school system, and particularly interested in all those cases where it is necessary to match fixed periods of time with lessons that are given by certain teachers. Under this context, different constraints have to be considered. Some of them referred as *hard constraints* have to be satisfied for a solution to be considered valid (for example, a teacher cannot teach two lessons at the same time). Others, referred as *soft constraints*, reflect the preferences given by the teachers or by the policies of the school (for example, teachers want to minimize the holes in their schedule, and a school policy could aim at avoiding practicing sports after lunch). Therefore, the accomplishment of hard constraints indicates the feasibility of a solution, whereas the soft constraints permit to quantify its goodness. Also, the school timetabling problem we consider (including all constraints) admits a mathematical representation [20], so that heuristic solution can be obtained by applying well-known techniques in this field. However, school timetabling problem is known to be a NP-complete problem [29]. The objective of this research is to develop a modified simulated annealing algorithm that will efficiently solve school timetabling problems optimally, and that would also enhance the effectiveness of timetable generation in Nigerian high schools.

Research into meta-heuristic methods has provided significant advances in automated timetabling technology [30]. Meta-heuristic methods begin with one or more initial solutions and employ search strategies to find optimal solution, trying to avoid local optimal in the process [31]. Simulated Annealing (SA), Tabu Search (TS), Genetic Algorithm (GA), Memetic Algorithm (MA) and Ant Colony System (ACS) are among the main meta-heuristics for solving challenging problems of intelligent systems [32].

SA is a random-search probabilistic method proposed in [33] and [34] for finding the global minimum of a cost function that may possess several local minima. It works by emulating the physical process whereby a solid is heated, slowly cooled, and eventually “frozen” into a minimum energy crystalline structure, this happens at a minimum energy configuration.

SA is based on the annealing process for metals, and its major advantage over other meta-heuristics is its ability to avoid getting trapped at a local minimum by employing a random search which not only accepts changes (neighbourhood) that increase the objective function value but also some changes that decrease the objective function (assuming a minimization problem), using a probabilistic acceptance criterion [33].

SA is an efficient, robust and a good tool for complex nonlinear optimization problems [35], and is amongst the leading meta-heuristic for solving timetabling problems because of its ability to avoid getting trapped at a local minimum. SA has been widely and successfully used for tackling a variety of different combinatorial optimization problems since been originally developed by [33]. Depending on the problem to which it is applied, SA appears competitive with many of the best heuristics as shown by [36, 37, 38, 39, 40].

However, the major drawback of SA is its very slow convergence (converges at excessive time) as SA is inherently sequential and hence very slow for problems with large search spaces [41]. Several attempts have been made to speed up this process, for example [42] indicated that by carefully controlling the rate of cooling the temperature, SA can find the global optimum exponentially faster since SA algorithm performance depends more on the choice of cooling schedule (cooling rate), and that since slow cooling schedules are generally more effective, the computational cost is generally improved with slower cooling rates.

Hence, SA can be modified by including a mechanism for carefully controlling the rate of cooling the temperature. The inclusion of a mechanism for carefully controlling the rate of cooling the temperature into SA results into the model called Modified Simulated Annealing (MSA) developed in this work that converged at lesser time, thereby reducing the time wasted during manual timetable preparation, resolving the conflict associated with teachers and classes’ clashes, and as well reducing human intervention to solving combinatorial problems to avoid undue favouritism and partiality.

The remainder of this paper is organized as follows: The next section presents the review of SA meta-heuristic technique. Section 3 describes the methodology adopted and especially the technique developed. Section 4 shows the simulation results and the performance obtained. Finally Section 5 concludes the paper.

2. SA META-HEURISTIC

Simulated Annealing (SA) is a particular instance of meta-heuristic algorithms or methods. Meta-heuristic methods (Local search or Population-based search) are sort of heuristic algorithms that are not designed to solve a particular problem, but are rather designed with the aim of being flexible enough to handle as many different problems as possible. In local search an intensive exploration of the solution space is performed by moving at each step from the current solution to another promising solution in neighbourhood while population-based search consists of maintaining a pool of good solution in order to produce hopefully better solutions [43].

Meta-heuristic algorithms constitute a class of computational paradigms useful for function optimization, often inspired by the study of natural processes [44]. They begin with one or more initial solutions, and usually update possible solutions,



one or a whole set at a time, and employ search strategies to find optimal solution, trying to avoid local optimal in the process [31, 45]; in this sense the naming *evolutionary algorithm* is common in the literature [44]. Particular efficient instantiation of evolutionary algorithms are represented by *Simulated Annealing* [33], in which the natural analogy is the annealing process for metals; by *Generic Algorithms* [46], in which the natural analogy is population genetics, and by *Tabu Search* [47, 48].

Simulated Annealing is an effective single-solution, randomized, probabilistic, and local search heuristic, for both discrete and continuous problem, that is, problems whose cost functions are defined on finite set [49], and are also defined on continuous sets [50].

Local search techniques are a family of general-purpose techniques for the solution of optimization problems (Aarts *et al.*, 1997). Consider an optimisation problem, and let S be a possible search space for it. A function N , which depends on the structure of the specific problem, assigns to each feasible solution $s \in S$, its neighbourhood $N(s) \subseteq S$. Each solution $s^1 \in N(s)$ is called a neighbour of s . A local search technique, starting from an initial solution s_0 (which can be obtained with some other technique or generated at random), enters in a loop that navigates part of the search space, stepping iteratively from one solution to one of its neighbours. The connectivity of the search space, with respect to the neighbour relation, is a necessary condition for the technique to work effectively [19]. The modification that transforms a solution into one of its neighbours is called a move. The selection of the move to be performed at each step of the search is based on the *cost function* f , which assesses the quality of the solution. The cost function generally is composed of the so-called *distance to feasibility*, which accounts for the number of constraints that are violated, and the objective function of the problem.

In other words, local search heuristics can be used to search for good solutions as opposed to using a mathematical program to find an optimal solution, which may take too long to find. Local search techniques move from one solution to another by making a “move” to a neighbour of the current solution. Neighbours, as well as moves, may be different for different problem models. Solutions are compared based on an objective function that needs to be minimized or maximized.

Simulated Annealing, originally developed by [33], is a type of local-search heuristic algorithm that avoids getting trapped at a local minimum by employing a random search which not only accepts changes (neighbourhood) that increase the objective function value, but also some changes that decrease the objective function (assuming a minimization problem), using a probabilistic acceptance criterion [32]. It is a probabilistic method for finding the global minimum of a cost function that may possess several local minima.

SA accepts always the randomly generated neighbour if it is better than or equal the current solution, and accepts it if is in an up-going move with a certain probability. Such probability depends on the increase of the objective function and on a factor called temperature reduction parameter (cooling rate), which decrease the number of iteration performed [51, 52].

Simulated Annealing is a random-search technique which exploits an analogy between the way in which a metal cools and freezes into a minimum energy crystalline structure (the

annealing process) and the search for a minimum in a more general system; it forms the basis of an optimization technique for combinatorial and other problems [39, 53, 54]. It works by emulating the physical process whereby a solid is heated, slowly cooled, so that when eventually its structure is “frozen”, it “frozen” into a minimum energy crystalline structure. This happens at a minimum energy configuration [49]. In addition, the search for a minimum in a more general system forms the basis of an optimization technique for solving combinatorial based problems.

The process starts by creating a random initial solution. The main procedure consists of a loop that generates at random at each iteration a neighbour of the current solution. The definition of neighbour depends on the specific structure of the problem. Let's call Δ the difference in the objective function between the new solution and the current one and suppose to deal with a minimization problem. If $\Delta \leq 0$ the new solution is accepted and becomes the current one. If $\Delta \geq 0$ the new solution is accepted with probability $e^{-\Delta/T}$, where T is a parameter, called the temperature. The temperature T is initially set to an appropriately high value T_0 . After a fixed number of iterations, the temperature is decreased by the cooling rate α , such that $T_n = \alpha \times T_{n-1}$, where $0 \leq \alpha \leq 1$. The procedure stops when the temperature reaches a value very closed to 0 and no solution that increases the objective function is accepted anymore, i.e. the system is frozen. The solution obtained when the system is frozen is obviously a local minimum. The control knobs of the procedure are the cooling rate α , the number of iterations at each temperature, and the starting temperature T_0 .

Simulated annealing has been considered a good tool for complex nonlinear optimization problems [35, 55]. The technique has been widely applied to a variety of problems. One of the major drawbacks of the technique is its very slow convergence (converges at excessive time). Simulated Annealing is inherently sequential and hence very slow for problems with large search spaces [41]. Several attempts have been made to speed up this process, such as development of parallel simulated annealing techniques [56] and special purpose computer architectures [57].

[42] indicated that the performance of SA algorithm depends on the cooling rate $\Delta T/L$ than on the individual values of ΔT and L , and that slow cooling schedules are generally more effective, and that the computational cost is in general improves with slower cooling rates. [42] proved further 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.

[58] indicated that the physical analogy on which the SA is based suggests that, to achieve good quality results, the system should be kept close to thermal equilibrium as the temperature is lowered.

[53] reported that to use simulated annealing effectively, it is crucial to use a good cooling schedule and a good method for choosing new trial schedules, in order to efficiently sample the search space. To obtain good results, the various tunable parameters to be used (such as the cooling rate, the update moves, and so on) need to be carefully chosen. Also, [59, 60] reported that the choice of the cooling schedule influences the quality of solution obtained.



3. METHODOLOGY

Solution representation technique appropriate to the problem is defined by mathematical formulation that describes the feasible regions through a solution space, along an objective (cost) function which permits to lead a search process towards an optimal solution. In the developed Modified Simulated Annealing (MSA) algorithm, the temperature reduction parameter for the exponential cooling schedule was modified to become parabolic by adding a non-linear factor which takes care of some of the intrinsic characteristics (good initial schedule, carefully selected update moves, and acceptance distribution) that account for the slow convergence of the SA algorithm. The MSA algorithm was implemented and simulated using MATLAB 7.9 version. Thereafter, the performance of the developed MSA algorithm was compared with the SA algorithm in terms of temperature reduction parameter, simulation time, computational cost, and constraints violation using data collected from Fakunle Comprehensive High School (FCHS) Osogbo, Nigeria.

3.1 Problem Representation

School timetabling is a specific case of the more general educational timetabling problem. In the case of school timetabling, a set of m classes c_1, \dots, c_m , n teachers t_1, \dots, t_n , (q subjects s_1, \dots, s_q) are to be scheduled within a certain number of p periods $1, \dots, p$, such that no class, teacher, or subject is scheduled more than once per period, and subject to a variety of hard and soft constraints (see Table 1 below) [16, 61, 62].

Table 1: Summary of Constraints Considered

| Label | Definition |
|-------|--|
| HC1 | Requirement Constraint |
| HC2 | Class Clashes Constraint |
| HC3 | Teacher Clashes Constraint |
| HC4 | Teachers Unavailability Constraint |
| HC5 | Classes Unavailability Constraint |
| HC6 | Class-One Lesson Constraint |
| HC7 | Pairs of Lessons Constraint |
| SC1 | Under-use Teaching Period Constraint |
| SC2 | Over-use Teaching Period Constraint |
| SC3 | Undesired Teaching Period Constraint |
| SC4 | Idle Period Constraint |
| SC5 | Class Split Constraint |
| SC6 | Maximum Number of Lessons Per Day Constraint |
| SC7 | Length of the Class Work Constraint |

HC = Hard Constraint

SC = Soft Constraint

3.2 Classical SA Pseudocode

Step 1: Generate initial solution randomly (double move)

Step 2: Initialize Search variables

Step 3: Estimate initial temperature

Step 4: Repeat 'cycle'

Step 5: Set Sao = double random move

Step 6: Initialize $t = 0$

Step 7: Generate "Current Timetable solution"; weights of hard and soft constraints are taken into consideration

Step 8: Repeat 't'

(i) Linear temperature reduction factor is set as

$$\alpha = (1/\log(1+t))$$

(ii) **Set** Final Temperature: (i.e. $T_{t+1} = \alpha T_t$)

(iii) **Set** $Sa1$ = new random move

(iv) Calculate the cost function by Setting $\Delta Sa =$

$$Sa1 - Sao$$

(v) **If** ($\Delta Sa < 0$ or $random < e^{-T_t}$) **Then**

"accept Best timetable solution"

Else

GoTo Step 6

End

Until ($t \leq tmax$)

Step 9: Set Current Timetable = Best Timetable solution

Until ($nocycle \leq cycle$)

Step 10: Stop

3.3 Modified SA Pseudocode

Step 1: Generate initial solution randomly (double move)

Step 2: Initialize Search variables

Step 3: Estimate initial temperature

Step 4: Repeat 'cycle'

Step 5: Set Modified Sao = double random move

Step 6: Initialize $t = 0$

Step 7: Generate "Current Time table solution"; weights of hard and soft constraints are taken into consideration

Step 8: Repeat 't'

(i) Parabolic temperature reduction factor is set as

$$\alpha = (1/\log(1+t+t^2))$$

(ii) **Set** Final Temperature: (i.e. $T_{t+1} = \alpha T_t$)

(iii) **Set Modified** $Sa1$ = new random move

(iv) Calculate the cost function by Setting



Modified $\Delta Sa = Sa1 - Sao$

(v) **If** (Modified $\Delta Sa < 0$ or $random < e^{-T_i}$) **Then**

“accept Best timetable solution”

Else

GoTo Step 6

End

Until ($t \leq tmax$)

Step9: Set Current Timetable = Best Timetable solution

Until ($nocycle \leq cycle$)

Step 10: Stop

Steps 8 (i & ii) of the two algorithms (classical simulated annealing and modified simulated annealing) shows the difference in the cooling schedule. For the classical simulated annealing the reduction parameter α for the exponential cooling schedule was determined using the formula proposed by [33] and, defined as $\alpha = (1/\log(1+t))$. However, for the modified simulated annealing, the reduction parameter α for the exponential cooling schedule was modified to become parabolic by adding a non-linear factor to take care of some of the intrinsic characteristics that could not taken care of in the classical simulated annealing which accounts for the slow convergence of the classical simulated annealing, defined as $\alpha = (1/\log(1+t+t^2))$.

3.4 Data Used for the Work

The following are the set of data used to automatically generate the school timetables:

List of subjects and their grouping

List of subjects’ period per week

List of students classes’, levels, arms and grouping (science, commercial and art)

List of teachers and their grouping (science, commercial and art), with their assigned teaching subjects

Maximum lesson period per day for each class

Maximum lesson period per week for each teacher

Duration of each lesson

(See Table 2, Table 3 and Table 4 for the summary of data used)

Table 2: Summary of Data Used

| | (JSS) | (SSS) |
|----------------|-------|-------|
| No of Classes | 18 | 24 |
| No of Teachers | 45 | 77 |
| No of Subjects | 15 | 19 |

JSS- Junior Secondary School

SSS- Senior Secondary School

Table 3: List of Class’s Level and Class’s Arms (JSS)

| Level | Arm |
|-------|-----------|
| JSS 1 | 6 (a – f) |
| JSS 2 | 6 (a – f) |
| JSS 3 | 6 (a – f) |

Table 4: List of Class’s Level, Arms and Groups (SSS)

| Level | Arm | Groups |
|-------|-----------|--------------------|
| JSS 1 | 8 (a – h) | Science (a - c) |
| JSS 2 | 8 (a – h) | Commercial (d - f) |
| JSS 3 | 8 (a – h) | Art (g – h) |

3.5 Performance Evaluation Metrics

In order to evaluate the coded algorithms, temperature reduction parameter, simulation time, computational cost, and constraints violation were used as performance evaluation metrics. The temperature reduction parameter is the metric which determines how the control parameter, designated by temperature T , is lowered during the annealing. The simulation time is the parameter which measures the time utilized by an algorithm to run until the result is produced. It is otherwise known as execution time. The constraints violation is the metric which determines the feasibility or validity, and the goodness of the solution produced by an algorithm.

As advocated by [63], simulation or execution 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. [64] indicated that one of the most important factors considered before choosing the winner during the second international timetabling competition (ITC-2007) was the computation time

4. RESULT AND DISCUSSIONS

After the implementation of the two algorithms (classical SA and the modified MSA), Table 5 shows the evaluation parameters or metrics and the values obtained or measured. These values were used to evaluate the two algorithms. Table 5 shows that the two algorithms produced feasible solutions, because none violated the constraints considered in this work. Table 1 contains the summary of the hard and soft constraints considered.

Table 5: Summary of Data Used Obtained

| | JSS | | SSS | |
|---------------------------------|------|------|------|------|
| | SA | MSA | SA | MSA |
| Parameters | | | | |
| Temperature Reduction Parameter | 0.95 | 0.67 | 0.95 | 0.67 |



| | | | | |
|------------------------------------|-------|-------|-------|-------|
| Simulation Time (Seconds) | 62.85 | 15.70 | 85.63 | 37.91 |
| Computational Cost | 43.76 | 11.59 | 63.87 | 22.79 |
| Number of Hard Constraint Violated | 0 | 0 | 0 | 0 |
| Number of Soft Constraint Violated | 0 | 0 | 0 | 0 |

4.1 Temperature Reduction Parameter

The rate at which the control parameter designated by temperature T , is lowered during the annealing is known as the temperature reduction parameter. The temperature reduction parameter value for the exponential cooling schedule used in classical SA is 0.95, while the temperature reduction parameter value for the parabolic exponential cooling schedule used in the developed MSA is 0.67. This caused the developed MSA algorithm to utilize less simulation time (and converges faster), and less computation cost than the classical SA algorithm.

This indicates that the performance of SA algorithm is in general depends on the cooling rate, and also 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.

4.2 Simulation Time

The time utilized by an algorithm to run until the result is produced is usually called execution time or simulation time. Table 5 shows the measured or obtained values of the simulation time of the two algorithms (classical SA and the developed MSA) for Junior Secondary School (JSS) and Senior Secondary School (SSS) respectively. The simulation time of the classical Simulated Annealing (SA) and the developed Modified Simulated Annealing (MSA) are 62.85 seconds and 15.70 seconds respectively to return a feasible school timetable for JSS, and 85.63 seconds and 37.91 seconds respectively to return a feasible school timetable for SSS. This is clear evidence that the developed Modified Simulated Annealing (MSA) algorithm utilized less time and converges faster than the developed classical Simulated Annealing algorithm.

This indicates that by carefully controlling the rate of cooling the temperature, simulated annealing can find the global optimum exponential faster since slow cooling schedules are generally more effective. This further indicates that it takes more time to compute the timetables for SSS classes than JSS classes because of the different subject groups (arms) that exists such as science group, commercial group and art group in the SSS.

4.3 Computational Cost

Table 5 shows the measured values of the computational cost of the two algorithms (classical SA and the developed MSA) for Junior Secondary School (JSS) and Senior Secondary School (SSS) respectively. The computational cost of classical SA and the developed MSA are 43.76 and 11.59 respectively to return a feasible school timetable for JSS; and 63.87 and 22.79 respectively to return a feasible school timetable for SSS. This is clear evidence that the developed MSA algorithm

used less computation cost and converges faster than the classical SA. This is as a result of the lower reduction parameter value (slower cooling rate) used.

This indicates that computational cost is in general improves with slower cooling rates. This further indicates that it takes higher computational cost to compute the timetable for SSS classes than JSS classes because of the different subject groups that exists such as science group, commercial group and art group in the SSS.

4.4 Constraints Violation

The two algorithms (classical SA and the developed MSA) did not violate any of the constraints (hard and soft) considered in this work. This is clearly evident in the summary of data (result) obtained and presented in Table 5.

This indicates that the two algorithms (classical SA and the developed MSA) produced feasible solutions.

4.5 Summary of Result and Discussion

The analysis of the implemented results shows that:

- i. The two considered algorithms (SA and the developed MSA) generated feasible solutions to the problem instances, as all the pre-specified hard and soft constraints were not violated.
- ii. The results generated generally indicate optimal solutions with high optimality by the developed MSA algorithm.
- iii. The developed MSA algorithm utilized less simulation time (converges faster) than the classical SA algorithm.
- iv. The developed MSA algorithm utilized less computational cost, and lower temperature reduction parameter value (slower cooling rate) than the classical SA algorithm.
- v. The developed MSA algorithm produced feasible high school timetables at the most reasonable time of 15.70 seconds for JSS and 37.91 seconds for SSS.
- vi. The developed MSA algorithm produced feasible high school timetables at the most reasonable computational cost of 11.59 for JSS and 22.79 for SSS.

Furthermore, the analysis of the implemented results confirms that:

- i. Simulated Annealing is a good tool for solving complex nonlinear optimization problems
- ii. To obtain good results the various tunable parameters to be used such as (the cooling rate, the update moves) need to be carefully chosen.
- iii. The choice of the cooling schedule or cooling rate (temperature reduction parameter) influences the quality of solution obtained.
- iv. The performance of Simulated Annealing algorithm depends on the cooling rate, as slow cooling schedules are generally more effective, and that the computational cost is in general improves with slower cooling rates.



- v. By carefully controlling the rate of cooling the temperature, Simulated Annealing can find the global optimum exponential faster since slow cooling schedules are generally more effective.

5. CONCLUSIONS

The school timetabling problem is very challenging for public schools in Nigeria. Several days of work are normally employed to manually solve these problems. In this work, a Modified Simulated Annealing (MSA) was developed as an approach for solving timetabling problems compared to classical Simulated Annealing (SA) approach. The developed algorithm (MSA) presents some modifications to the standard dynamics of temperature decrease function of the classical Simulated Annealing by the use of parabolic exponential temperature decrease function instead of linear exponential temperature decrease function which allows better solutions to be obtained. This modification enables the Modified Simulated Annealing Algorithm to work in short bursts of gradient descent, followed by random but controlled perturbations to escape local minima.

This developed model was used for generating all school timetables of the Fakunle Comprehensive High School (FCHS) Osogbo, Nigeria, and similar solution quality is obtained for classical SA. The analysis of the results showed that both the classical SA and developed MSA yielded better solutions when compared to manual allocation procedures but the improvement in the cooling schedule computation of the developed MSA makes it the preferred approach because it get out of local minima and converges in less time and utilizes less computational cost.

The results of this work confirmed previous research reports that the performance of Simulated Annealing algorithm depends on the cooling rate, and that by carefully controlling the rate of cooling the temperature, Simulated Annealing can find the global optimum exponential faster as slow cooling schedules (cooling schedule with lower temperature parameter) are generally more effective, and that the computational cost is in general improves with slower cooling rates since the higher the value of temperature reduction parameter, the longer it will take to decrement the temperature to the stopping criterion.

The developed MSA model thus make provisions for a robust school timetabling system that will ease administrators of the stress usually associated with manual timetabling and the time spent in its preparation would be drastically minimized. This work will also assist future researchers especially in scheduling or optimization techniques to be able to take more informed decisions when faced with the problem of selecting algorithm(s) that can solve this class of problem and related ones.

This work therefore recommends the use of the developed MSA in solving school timetabling problems, because it reduces the weaknesses of slow convergence (convergence at excessive time), and of high computational cost associated with classical SA.

6. FUTURE WORK

The outcomes of this work have indicated several future research directions as stated below:

- i. Application of other meta-heuristic methods, such as Ant Colony Optimization and Case-Based Reasoning to solve school timetabling problem.
- ii. Since each country or high school has different characteristics and requirements, the mathematical programming model can be extended or modified in order to adapt to these characteristics and requirements.
- iii. Further improvements to the solution quality of school timetabling problem of this research could be found by hybridizing two-stage method. The stage of the hybrid approach may be integrated more fully, to yield a more powerful and robust algorithm.

7. REFERENCES

- [1] Burke, E. K. and De Causmaecker, P. 2003. The Practice and Theory of Automated Timetabling: Selected Papers from the 4th International Conference. Lecture Note in Computer Science, Berlin: Springer-Verlag, Vol. 2740.
- [2] Burke, E. K. and Trick, M. 2005. The Practice and Theory of Automated Timetabling V: Revised Papers from the 5th International Conference, Pittsburgh 2004. Lecture Note in Computer Science, Berlin: Springer-Verlag, vol. 3616.
- [3] Gunawan, A. and Ng, K. M. 2011. Solving the Teacher Assignment Problem by Two Metaheuristics. International Journal of Information and Management Sciences, Vol. 22, pp. 73-86.
- [4] Kwan, R. S. K. 2004. Bus and train driver scheduling. In: J. Leung (Ed.), Handbook of Scheduling: Algorithms, Models and Performance. CRC Press, Boca Raton, Chapter 51.
- [5] Burke, E. K., De Causmaecker, P. and Berghe, V. G. 2004. Novel metaheuristic approaches to nurse rostering problems in Belgian hospitals. In: J. Leung (Ed.), Handbook of Scheduling: Algorithms, Models, and Performance Analysis. Chapman Hall/CRC Press, Vol.44, pp. 1-18.
- [6] Gunawan, A. and Lau, H. C. 2009. Master physician scheduling problem. In Proceedings of the 4th Multidisciplinary International Scheduling Conference, 10-12 August, Dublin, Ireland.
- [7] Schonberger, J., Mattfeld, D. C. and Kopfer, H. 2004. Memetic algorithm timetabling for noncommercial sport leagues. European Journal of Operational Research, Vol. 153, No. 1, pp. 102-116.
- [8] Tsuchiya, K. and Takefuji, Y. 1997. A neural network parallel algorithm for meeting scheduling problems. Applied Intelligence, Vol. 7, No. 3, pp. 205-213.
- [9] Burke, E. K. and Rudova, H. 2007. The Practice and Theory of Automated Timetabling: Selected Papers from the Sixth International Conference. Lecture Note in Computer Science, Berlin: Springer-Verlag.
- [10] Burke, E. K., MacCarthy, B. L., Petrovic, S. and Qu, R. 2006. Multiple retrieval case-based reasoning for course timetabling problems. Journal of Operations Research Society (JORS), Vol. 57, No. 2, pp. 148-162.



- [11] Mccollum, B. 2006. University timetabling: bridging the gap between research and practice. In: E. H. R. Burke (Ed.), Proceedings of the 6th International Conference on Practice and Theory of Automated Timetabling, pp. 15-35.
- [12] Nurmi, K. and Kyngas, J. 2007. A framework for school timetabling problem. In Proceedings of 3rd Multidisciplinary International Scheduling Conference: Theory and Applications, Paris, pp. 386-393.
- [13] Schaerf, A. 1995. A survey of automated timetabling, Technical Report CS-R9567. In Centrum Voor Wiskunde en Informatica Amsterdam, Netherlands, Vol. 115, pp. 33, ISSN: 0169-118X..
- [14] Post, G., Ahmadi, S., Daskalaki, S., Kingston, J. H., Kyngas, J., Nurmi, C., Ranson, D. and Ruizenaar, H. (2010). An XML format for benchmarks in high school timetabling. Retrieved from [http://www.home.utwente.nl-postgf/BenchmarkSchoolTimetabling/..](http://www.home.utwente.nl-postgf/BenchmarkSchoolTimetabling/)
- [15] De Werra, D. (1985). An introduction timetabling. European Journal of Operational Research, Vol. 19, No. 2, pp. 151-162.
- [16] Abdennadher, S. and Marte, M. 2000. University course timetabling using constraint handling rules. Journal of Applied Artificial Intelligence, Vol. 14, No. 4, pp. 311-326.
- [17] Mahar, K. 2006. Automatic generation of University timetables: an evolutionary approach. IADIS, ISBN: 972-8924-09-7.
- [18] De Werra, D. 1996. Some combinatorial models for course scheduling. In: Lecture Note in Computer Science, Berlin: Springer-Verlag, Vol. 1153, pp. 296-308.
- [19] Burke, E. and Carter, M. W. 1998. The Practice and Theory of Automated Timetabling: Selected Papers from the Second International Conference. Lecture Notes in Computer Science, Berlin: Springer-Verlag, Vol. 1408.
- [20] Schaerf, A. 1999. A survey of automated timetabling. Artificial Intelligence Review, Vol. 13, No. 2, pp. 87-127.
- [21] Petrovic, S. and Burke, E. K. 2004. University timetabling. In: J. Leung (Ed.), Handbook of Scheduling: Algorithms, Models, and Performance Analysis. Chapter 45, Chapman Hall/CRC Press.
- [22] Qu, R., Burke, E. K., Mccollum B., Merlot, L.T.G. and Lee, S.Y. 2006. A Survey of search methodologies and automated approaches for examination timetabling, Technical Report NOTTCS-TR-2006-4, School of CSiT, University of Nottingham, UK.
- [23] Lewis, R. 2007. A survey of meta-heuristic based techniques for University timetabling problems. Operation Research (OR) Spectrum. Retrieved from <http://www.w3schools.com/xml/> or <http://en.wikipedia.org/wiki/xml>.
- [24] Wilke, P., Grobner, M. and Oster, N. 2002. A hybrid genetic algorithm for school timetabling. In: B. McKay and J. Slaney (Eds.), AI2002: Advances in Artificial Intelligence, 15th Australian Joint Conference on Artificial Intelligence, Canberra, Australia. Lecture Notes in Computer Science, Berlin: Springer-Verlag, Vol. 2557, pp. 455-464.
- [25] Santos, H. G., Ochi, L. S. and Souza, M. J. F. 2004. A tabu search heuristic with efficient diversification strategies for the class/teacher timetabling problem. In: Proceedings of the 5th International Conference on Practice and Theory of Automated Timetabling, Brazil, pp. 343-358
- [26] Valouxis, C. and Housos, E. 2003. Constraint programming approach for school timetabling. Computers and Operations Research, Vol. 30, pp. 1555-1572.
- [27] Abramson, D., Krishnamoorthy, M. and Dang, M. H. 1996. Simulated annealing cooling schedules for the school timetabling problem. Asia-Pacific Journal of Operational Research, Vol. 16, pp. 1-22.
- [28] Gunawan, A. and Ng, K. M. 2011. Solving the teacher assignment problem by two metaheuristics. International Journal of Information and Management Sciences, Vol. 22, pp. 73-86.
- [29] Willemen, R. J. 2002. School timetable construction: Algorithm and complexity, Doctoral Thesis, Technical University Eindhoven, Netherlands.
- [30] Burke, E. K., MacCarthy, B. L., Petrovic, S. and Qu, R. 2006. Multiple retrieval case-based reasoning for course timetabling problems. Journal of Operations Research Society (JORS), Vol. 57, No. 2, pp. 148-162.
- [31] Myszkowski, P. and Norberciak, M. 2003. Evolutionary algorithms for timetable problems, Annales UMCS, Sectio Informatica, Lublin, Vol. I.
- [32] Zahra, N. A. 2005. Hybrid heuristics for examination timetabling problem. Applied Mathematics and Computation, Vol. 163, No. 2, pp. 705-733.
- [33] Kirkpatrick, S., Gellatt, C. D. and Vecchi, M. P. 1983. Optimization by Simulated Annealing. Science, Vol. 220, No. 5, pp. 671-680.
- [34] Cerny, V. 1985. A thermodynamic approach to the travelling salesman problem: An efficient simulation algorithm. Journal of Optimization Theory and Application, Vol. 45, pp. 41-55.
- [35] Eglese, R.W. 1990. Simulated annealing: A tool for operation research. European Journal of Operations Research, Vol. 46, pp. 271-281.
- [36] Johnson, D. and C., Mcgeoch, L. 1992. The travelling salesman problem: a case study in local optimization. In: E. H. Aarts and J. K. Lenstra (Eds.), Local Search in Combinatorial Optimization. John Wiley and Sons, Chichester, England.
- [37] Olabisiyi, S. O., Omidiora, E. O. and Oyeleye, C. A. 2005. Asymptotic time complexity analysis for two dimensional object inspection using string matching. Science Focus, Nigeria, Vol. 10, No. 3, pp. 83-89.
- [38] Olabiyisi, S. O., Akanmu T. A., Oyeleye, C. A., Sabayo, O. D. and Adelana, J. B. 2007. Complexity analysis of a



- new edge-adaptive zooming algorithm for digital image. *Journal of Res. Phys. Sci.*, Vol. 3, No. 2, pp. 67-71.
- [39] Omidiora, E. O., Olabiyisi, S. O., Arulogun, O. T., Oyeleye, C. A. and Adegbola. A. 2009. A prototype of an access control system for a computer laboratory scheduling. *AICTTRA 2009 Proceedings*, Obafemi Awolowo University, Ile Ife, Nigeria, pp. 114-120.
- [40] Oyeleye, C. A., Olabiyisi, S. O., Omidiora, E. O., and Fagbola, T. 2014. Hybrid metaheuristic of simulated annealing and genetic algorithm for solving examination timetabling problem. *International Journal of Computer Science and Engineering (IJCSE)*, ISSN(P):2278-9960, ISSN(E): 2278-9979, Vol. 3, Issue 5, (Sep. 2014), pp. 7-22.
- [41] JanakiRam, D., Screenivas, T. H. and Ganapathy S. K. 1996. Parallel simulated annealing algorithm. *Journal of Parallel and Distributed Computing*, Vol. 37, pp. 207-212.
- [42] Thompson, J. M. and Dowsland, K. A. 1998. A robust simulated annealing based examination timetabling system. *Computers and Operational Research*, Vol. 25, No. 7/8, pp. 637-648.
- [43] Hertz, A. and Widmer, M. 2003. Guidelines for the use of metaheuristics in combinatorial optimization. *European Journal of Operational Research*, Vol. 151, pp. 247-252
- [44] Maniezzo, V., Colomi, A. and Dorigo, M. 1995. Algodesk: an experimental comparison of eight evolutionary heuristics applied to the Quadratic Assignment Problem. *European Journal of Operational Research*, Vol. 81, No. 1, pp. 188-205.
- [45] Newall, J. P. 1999. Hybrid methods for automated timetabling, Doctoral Thesis, Department of Computer Science, University of Nottingham, UK.
- [46] Holland, J. H. 1975. *Adaption in natural and artificial systems*. University of Michigan Press, Ann Arbor, Michigan. (Reprinted by MIT Press, 1992).
- [47] Glover, F. 1989. Tabu Search-Part I. *ORSA Journal on Computing*, Vol. 1, No. 3, pp. 190-206.
- [48] Glover, F. 1990. Tabu Search-Part II. *ORSA Journal on Computing*, Vol. 2, No. 1, pp. 4-32.
- [49] Bertsimas, D. and Tsitsiklis, J. 1993. Simulated annealing. *Statistical Science*, Vol. 8, No. 1, pp. 10-15.
- [50] Yang, R. L. 2000. Convergence of the simulated annealing algorithm for continuous global optimization. *Journal of Optimization Theory and Applications*, Vol. 104, No. 3, pp. 691-716.
- [51] Van Laarhoven, P. J. M. and Aarts, E. H. L. 1987. *Simulated annealing: theory and applications*. Dordrecht: D. Reidel Publishing Company, Kluwer Academic Publishers Group.
- [52] Aarts, E. H. L. and Korst, J. 1989. *Simulated annealing and boltzmann machines*. John Wiley and Sons, New York.
- [53] Elmohamed, S., Coddington, P. and Fox, G, 1998. A comparison of annealing techniques for academic course scheduling. In: E. Burke and M. Carter (Eds.), *The Practice and Theory of Automated Timetabling: Selected Papers from the Second International Conference. Lecture Notes in Computer Science*. Berlin: Springer-Verlag, Vol. 1408, pp. 92-112.
- [54] Dowsland, K.A. 1996. Simulated annealing solutions for multi-objective scheduling and timetabling. In: *Modern Heuristic Search Methods*. John Wiley and Sons, Chichester, England, pp. 155-166.
- [55] Van Laarhoven, P. J. M., Aarts, E. H. L., and Lenstra, J. K. 1992. Job shop scheduling by simulated annealing. In *Operation Research*, Vol. 40, pp. 113-125.
- [56] Allwright, James, R. A. and Carpenter, D. B. 1989. A distributed implementation of simulated annealing for the traveling salesman problem. *Parallel Comput.* Vol. 10, pp. 335-338.
- [57] Abramson, D. 1992. A very high speed architecture for simulated annealing. *IEEE Computer*, pp. 27-36.
- [58] Lam, J. and Delosme, J. M. 1988. An efficient simulated annealing schedule: implementation and evaluation, Report 8817, Department of Computer Science and Department of Electrical Engineering, Yale University.
- [59] Johnson, D., Aragon, C., Mcgeoch, L. and Schevon, C. 1990. Optimization by simulated annealing: an experimental evaluation, Part I: Graph Partitioning. *Operational Research*, Vol. 37, pp. 865-892.
- [60] Johnson, D., Aragon, C., Mcgeoch L. and Schevon, C. 1991. Optimization by simulated annealing: an experimental evaluation, Part II: Graph Colouring and Number Partitioning. *Operational Research*, Vol. 39, pp. 378-406.
- [61] Odeniyi, O. A. 2014. A modified simulated annealing approach for school timetabling. An unpublished M. Tech. Thesis, Ladoke Akintola University of Technology, Ogbomoso, Nigeria.
- [62] Smith, K. A., Abramson, D. and Duke, D. 2003. Hopfield neural networks for timetabling: Formulation, methods, and comparative results. *Computers and Industrial Engineering*, Vol. 44, No. 2, pp. 283-305.
- [63] Kohonen J. 1999. 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.
- [64] McCollum, B. 2006. University timetabling: bridging the gap between research and practice. In: E. H. R. Burke (Ed.), *Proceedings of the 6th International Conference on Practice and Theory of Automated Timetabling*, pp. 15-35.