ENHANCEMENT OF SIMULATED ANNEALING ALGORITHM FOR SOLVING UNIVERSITY COURSE TIMETABLING PROBLEMS

HASSAN YOUNIS AL-TARAWNEH

THESIS SUBMITTED IN THE FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

FACULTY OF INFORMATION SCIENCE AND TECHNOLOGY
UNIVERSITI KEBANGSAAN MALAYSIA
BANGI

2013

PENAMBAHBAIKAN ALGORITHMA SIMULASI PENYEPUHLINDAPAN UNTUK MENYELESAIKAN MASALAH PENJADUALAN KURSUS UNIVERSITI

HASSAN YOUNIS AL-TARAWNEH

TESIS YANG DIKEMUKAKAN UNTUK MEMPEROLEH IJAZAH
DOKTOR FALSAFAH

FAKULTI TEKNOLOGI DAN SAINS MAKLUMAT
UNIVERSITI KEBANGSAAN MALAYSIA
BANGI

2013

DECLARATION

I hereby declare that the work in this thesis is my own except for quotations and summaries which have been duly acknowledged.

19 July 2013

HASSAN YOUNIS AL-TARAWNEH
P46477

ABSTRACT

Simulated Annealing (SA) is a common metaheuristic algorithm that has been widely used to solve complex optimization problems. This is due to its ease of implementation and its capability to escape from the local optimum. However, it could still get trapped in the local optimum and it takes a longer time to find a good quality solution. Thus, the aim of this thesis is to improve the SA performance and overcome the disadvantages. Two real world university course timetable datasets, which are the ITC2007 Track3 benchmark and the UKM-Faculty of Engineering datasets, were used in this research. The first phase involved conducting a thorough investigation into three SA components: the initial temperature, the cooling schedule and the neighbourhood structure. It was observed that a high initial temperature would cause the SA to accept any solution (wasting more computational time), whilst a lower value would cause it to be quickly trapped in local optimum. Based on the findings from this phase, a technique has been proposed for each component to overcome these limitations. These are: (i) a dynamic initial temperature mechanism that dynamically chooses a suitable initial temperature for each instance problem; (ii) an adaptive cooling schedule that will adjust the temperature value during the search; and (iii) a new neighbourhood structure to improve the search ability by minimizing the random selection. In the second phase, the simulated annealing was hybridized with a memory called (SAM) to enhance the capability of the SA in escaping the local optimum. Moreover, a guided shaking procedure was also proposed for the SAM that can effectively divert the search to another promising region using adaptive soft constraint weights. In the final phase, the SAM was further enhanced by integrating a tabu list memory (SA-TL-AM) onto the SAM to avoid repetition. The experimental results showed that the SA-TL-AM improved the performance of the SAM by increasing its capability to escape from local optimum and reducing the possibility of it being re-trapped in recent local optimum. The new adaptive neighbourhood selection (AD-NS) is another technique that has been designed to select a neighbourhood with the best improvement strength history. The AD-NS enhanced the solution quality by avoiding the disconnected neighbourhood structure. The experimental results showed that the proposed techniques and approaches in all the phases outperformed the SA and were comparable to other approaches in the literature (tested on ITC2007 Track^3 and the UKM-Faculty of Engineering's university course timetabling datasets). Therefore, it can be concluded that the research objectives have been achieved.

ABSTRAK

Simulasi Penyepuhlindapan (SP) adalah algortima meta-heuristik yang digunakan secara meluas dalam menyelesaikan masalah pengoptimuman yang kompleks. Ini disebabkan SP mudah dilaksanakan dan keupayaan SP untuk mengelak dari terperangkap dalam masalah optima setempat. Walau bagaimanapun, SP masih boleh terperangkap dalam optima setempat dan ini menyebabkan SP mengambil masa yang lama untuk mencari penyelesaian yang berkualiti. Oleh yang demikian, matlamat tesis ini adalah untuk meningkatkan prestasi dan mengatasi kelemahan SP. Penyelidikan ini dibahagikan kepada tiga fasa. Dua set data sebenar untuk penjadualan kursus universiti digunakan dalam penyelidikan ini iaitu ITC2007 Track^3 penanda aras dan Fakulti Kejuruteraan UKM set data. Fasa pertama adalah untuk menjalankan siasatan terperinci keatas tiga komponen SP iaitu suhu awal, jadual penyejukan dan struktur kejiranan. Kajian menunjukkan suhu awal yang tinggi akan menyebabkan SA menerima apa sahaja penyelesaian (membuang masa komputeran), manakala nilai yang lebih rendah menyebabkan SP lebih cepat terperangkap dalam optima setempat. Berdasarkan dapatan fasa ini, teknik baru dicadangkan untuk mengatasi batasan yang terdapat pada setiap komponen. Teknik berkenaan adalah: (i) suhu awal dinamik, iaitu mekanisme dinamik untuk memilih suhu awal yang sesuai bagi setiap masalah; (ii) jadual penyejukan adaptif yang akan melaraskan nilai suhu semasa menggelintar; dan (iii) struktur kejiranan baru untuk meningkatkan keupayaan carian dengan meminimumkan pemilihan rawak. Dalam fasa kedua, kami telah menghibrid SP dengan memori (SPM), untuk meningkatkan keupayaan SP bagi mengelak optima setempat. Selain itu, kami juga mencadangkan prosedur ‘goncangan’ untuk SAM yang boleh mengalih gelintaran ke rantau lain yang lebih baik, menggunakan pemberat kekangan lembut yang adaptif. Pada fasa terakhir, kami telah meningkatkan SPM dengan mengintegrasikan senarai memori tabu (SP-TL-SM) dalam SPM untuk mengelakkan pengitaran semula. Keputusan eksperimen menunjukkan bahawa SA-T-AM telah meningkatkan prestasi SAM setelah mengelak dari isu optimum setempat dan mengurangkan kemungkinan terperangkap semula dalam optimum setempat. Pilihan kejiranan dinamik (PKJ) merupakan teknik yang direka untuk memilih kejiranan dengan sejarah kekuatan peningkatan terbaik. PKJ meningkatkan kualiti penyelesaian dan mengurangkan masa pengiraan. Keputusan eksperimen menunjukkan bahawa teknik dan pendekatan yang dicadangkan pada semua fasa telah mengatasi SA malah setanding dengan pendekatan lain yang dilaporkan di dalam liputan kesusasteraan (diuji pada ITC2007 Track^3 dan Fakulti Kejuruteraan UKM). Oleh yang demikian, dapatlah disimpulkan bahawa objektif penyelidikan ini telah tercapai.

ACKNOWLEDGEMENTS

بسم الله الرحمن الرحيم

In the name of Allah Most Beneficent Most Merciful All praise is for Allah the Exalted and may the peace and blessings of Allah be upon His Messenger Muhammad and his family and companions and all those who follow them and their way until the Day of Resurrection

I am grateful to Allah that I have made this work so far and would like to acknowledge those who have helped me throughout my study. This work would not have been possible without their contributions.

To my beloved parents, brothers (Nedal and Adnan) and sisters (Hiyam, Siham and Sana') thank you for your patience and your unconditional support through all stages of my life and for giving me a confidence and hope to be better.

I would like to express my sincere thanks and appreciation to my supervisor, Assoc. Prof. Dr. Masri Ayob, for her insights, guidance, helps and advice. I sincerely thank her for everything she did to me, I am so proud to be one of her students. Due thanks must also be extended to the members of the examination committee. I am grateful also to people in Fakulti Teknologi dan Sains Maklumat, Universiti Kebangsaan Malaysia that helped during my study.

To all my friends who supported me and showed me the meaning of the real friendship during my PhD journey.

Dedication

To My Beloved Parents

My Brothers and Sisters

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

SA	Simulated Annealing
UKM	Universti Kebangsaan Malaysia
UCTP	University Course Timetabling Problems
TS	Tabu Search
TL	Tabu List
LR-TM	Local Repair Algorithm with Tabu Mechanism
GA	Genetic algorithm
SAGA	Self-Adaptive Genetic Algorithm
AC	Ant Colony
VNS	Variable Neighbourhood Search
VND	Variable Neighbourhood Descent
EMC	Exponential Monte Carlo
GRASP	Greedy Randomized Adaptive Search Procedure
MC	Monte Carlo
LMC	Linear Monte Carlo
EMC	Exponential Monte Carlo
EMCQ	Exponential Monte Carlo with Counter
GCS	Geometric Cooling Schedule
ACS	Adaptive Cooling Schedule
NS	Neighbourhoods Structure
RR	Round Robin
FS	Feasible Solution
SAMED	Simulated Annealing with Evolution-Based Diversification
TSP	Travel Salesman Problem
HC	Hill Climbing
CB-CTT	Curriculum Based Course Timetabling Problem ITC2007 (Track3) International Timetabling Competition
SD	Steepest Decent
D-IT	Dynamic Initial Temperature
SA-GCS	Simulated Annealing with Geometric Cooling Schedule
SA-ACS	Simulated Annealing with Adaptive Cooling Schedule
SA^*	Enhanced Simulated Annealing
SAM	Simulated Annealing with Memory
SA-TL-AM	Simulated Annealing with Tabu List and Allowable memory
ATL	Adaptive Tabu List
CON	Constraint Oriented Neighbourhoods
AD-NS	Adaptive Neighbourhood Structures Selection
Fix-IT	Fixed Initial Temperature
Sol	Current Solution
Sol^*	Best Solution
Sol_m	Solution from Memory
T	Temperature
T_0	Initial Temperature
T_c	Current Temperature
T_{min}	Minimum Temperature
Δ	The Differences between two Solutions Qualities
Mem	Memory

CHAPTER I

INTRODUCTION

1.1 BACKGROUND AND MOTIVATION

Wren (1996) defined timetabling as *“the allocation, subject to constraints, of given resources to objects being placed in space–time, in such a way as to satisfy, as nearly as possible, a set of desirable objectives’’*.

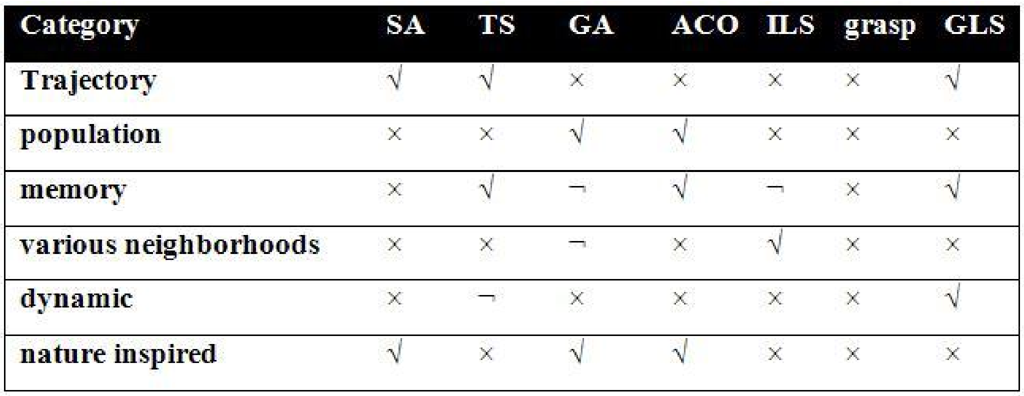
In the educational field, there are three different timetabling problems (school, university course and examination timetabling problems) which have been categorized by Schaerf (1999). The three classes may have the same fundamental characteristics but they are still different. For example, the class sizes for all school courses are usually very similar and the same group of students are associated with a set of courses, whilst the university courses have a different number of student enrolments.

The university coursetimetabling problem (UCTP) is a NP-Hard problem (Lewis 2008), where the UCTP has the task of appointing the events of a university (courses, lecturers) to the rooms and timeslots in such a way as to satisfy the hard constraints and minimize the soft constraints violation. The building of the academic timetable is a typical real world–scheduling problem and it has become a very tedious job in every academic institution annually. The NP-hard problems, such as the course timetabling, are very hard to solve using conventional optimization mechanisms (methods) such as constraint logic programming and backtracking programming, where these techniques might minimise the violation penalties and could result in finding a feasible, but not the optimal, solution(Hooker 2002).

In general, many researchers use metaheuristic methods to solve the timetabling problem. “*Metaheuristics are typically high-level strategies which guide an underlying, more problem-specific heuristic, to increase their performance*” (Blum & Roli,2003). Examples of metaheuristics are: genetic algorithms (Ueda et al.,2004), ant colony optimization (Eley, 2007), tabu search (Alvarez et al., 2002), evolutionary search (Beligiannis et al., 2008) and simulated annealing (Abramson, 1991).Rossi-Doria et al. (2003) compared the state-of-the-art algorithms for solving UCTP which are: Evolutionary Algorithms (EA), Ant Colony Optimization (ACO), Iterated Local Search (ILS), Simulated Annealing (SA), and Tabu Search (TS). Based on the results, they found that in small instances the ILS performed better, followed closely by SA and ACO. The GA performed worse and the TS obtained the worst results. The SA performed better in medium instances. The ILS was still very good. The GA and TS gave similar poor results, and the ACO showed the worst performance. For the large instances, the ILS performed better (with respect to the feasible solution) followed by GA, ACO, SA and TS. However, they concluded that it was very difficult to design a metaheuristic algorithm that could tackle general instances.

Table 1.1 summarizes some of the common metaheuristic algorithms under different categories which were presented by Birattari et al.(2001), where √ means this algorithm was totally categorized under this category and × means not, whilst ¬ means partly present.

Table 1.1 Summary of the metaheuristic algorithms characteristics



Category	SA	TS	GA	ACO	ILS	grasp	GLS
Trajectory	√	√	×	×	×	×	√
population	×	×	√	√	×	×	×
memory	×	√	¬	√	¬	×	√
various neighborhoods	×	×	¬	×	√	×	×
dynamic	×	¬	×	×	×	×	√
nature inspired	√	×	√	√	×	×	×

Note: SA is simulated annealing; TS is tabu search; GA is genetic algorithm; ACO is ant colony optimization; ILS iterated local search; grasp is greedy randomized adaptive search procedure; GLS is guided local search.

SA is a trajectory-based algorithm or a single point algorithm (Birattari et al., 2001). Single point algorithms consider a single element of the solution space at the iteration by jumping from one position to another in the solution space.

SA uses a special strategy to escape from a local optimum. At first, the SA will explore the search space (diversification) by easily accepting the worse solution based on the probability acceptance criterion. The probability is decreased during the search space process, leading to a greater focus on promising regions (intensification) (Affenzeller et al., 2009). The probability acceptance decreases depending on the temperature, which reduces during the search process to give a balance between the diversification and the intensification. SA is one of the oldest and more popular metaheuristic algorithms to have been successfully used to solve the university course timetabling problem .For example, a combination of SA with hill climbing and the great deluge was ranked first in comparison with other state-of-the-art approaches in the ITC2007 track[3] curriculum-based course timetabling problem competition.

However, SA can be trapped in a local optimum and may take a longer time to find a good solution (Xinchao, 2011). Hence, many researchers have attempted to improve the SA performance by using, for example, the adaptive SA (Ingber, 2000), or by hybridizing it with other metaheuristics, such as genetic algorithms (Ueda et al., 2004).In fact, the advantage of combining different algorithmic techniques is that the best performance can be achieved by combining and uniting the advantages of each individual technique under one hybrid algorithm. Therefore, this thesis is aimedat investigating further into how to improve the SA performance by escaping from local optimum and reducing the computational time.

1.2 PROBLEM STATEMENT

Firstly, this thesis chose to solve the problem of university course timetabling based on two main reasons: (i) the UCTP is still a challenging task to be solved and improved, and (ii) the UCTP is a combinatorial optimization problem (NP-hard problem).

Algorithms based on metaheuristics have proven to be highly effective in solving problems concerning university course timetabling (Burke et al., 2007; Chiarandini et al., 2006).

Glover and Kochenberger (2003)reported the advantages of metaheuristics as follows:-

- Easy to develop.
- Decrease the software development time.
- Easy to adapt to different problems.

SA is one of the oldest and more popular metaheuristic algorithms due to its ease of implementation and its ability to solve many combinatorial optimization problems (Xinchao, 2011). The SA can solve the university course timetabling problem and outperform other approaches. For example, the SA with hill climbing and great deluge algorithms was ranked first in comparison with other state-of-the-art approaches that were obtained in the ITC2007 track[3] curriculum-based course timetabling problem.

SA is still one of the popular metaheuristic algorithms and the challenging task facing researchers is to enhance its performance in solving different kinds of problems. Some recent researches that used the SA or the hybrid SA with metaheuristic algorithms to solve several NP- hard problems are:(Ceschia et al. 2012; Abdullah et al. 2012; Al-Betar & Khader 2012; Bai et al. 2012; Bellio et al. 2012; Brito et al. 2012; Fonseca et al. 2012; Gogos et al. 2012; Nothegger et al. 2012; Özcan et al. 2012; Pal et al. 2012; Chen & Shih 2013; MirHassani & Habibi 2013)

However, SA has two major disadvantages. First, it can get trapped in local optimum, and second it takes a longer time to find good solutions (Xinchao, 2011). Therefore, due to its ability to outperform other metaheuristic algorithms when it is combined with other algorithms, the SA has been selected as the focus of this research with the aim of enhancing its performance without combining it with other metaheuristic algorithms. For example, this study attempted to enhance the SA components to effectively solve the course timetabling by improving the quality of the solution. This thesis investigated the enhancement of the SA performance by eliminating its weaknesses in order to solve the university course timetabling problem.

The initial temperature, cooling schedule and neighbourhood structure are important key factors for the SA performance (Solla et al., 1986; Moscato, 1993; (Lü& Hao, 2010), where a good initial temperature will balance the diversification and intensification properly by avoiding wastage of computational time when the initial temperature is high, or being trapped in a local optimum when it is very low (Varanelli & Cohoon, 1999). Furthermore, adjusting the decrease in the given temperature during the search process is also a very important issue in order to avoid wasting the computational time (Elmohamed et al.,1998). A high temperature with a slow cooling schedule will enable the SA to accept many worse solutions (a zigzag walk), while a low temperature with a fast cooling schedule will enable the SA to be quickly trapped in a local optimum by rejecting most of the unimproved solutions and accepting the improved solutions only (descent method). Furthermore, an effective neighbourhood structure may easily improve the solution quality (Solla et al., 1986; Eglese, 1990; Moscato, 1993). Moreover, choosing the most suitable neighbourhood is an important issue as well in order to avoid the discounted neighbourhood and to save on the computational time (Ogbu & Smith, 1990). On the other hand, it is possible for the SA to be trapped in local optimum especially when the temperature is low.

Two main research questions have been identified (regarding the SA algorithm) to be answered:

a) *How can the SA performance be enhanced by escaping from a local optimum?*

b) *How can the SA computational time be reduced?*

In order to answer the main research questions, five research questions that need to be answered have been identified as follows:-

Q1) How can the initial temperature be initialized dynamically based on the individual problem instance?

Q2) Which effective cooling schedule can adaptively adjust the temperature decrement amounts during the SA search process?

Q3) How can a good neighbourhood structure be proposed that can help the SA to effectively search for a good quality solution and minimize the random swap and move?

Q4) How can a search escape when it gets trapped in a local optimum and the solution be diverted to other good promising regions?

Q5) How can a good neighbourhood structure be selected from among different neighbourhood structures during the search process so as to be able to avoid the disconnected neighbourhood?

In addition, this research needs to answer the following research question:
What is the performance of the approaches in this research in solving the real world university course timetabling problem at the Faculty of Engineering, UKM?

1.3 RESEARCH OBJECTIVES

The aim of this research is to enhance the performance of the SA algorithm by escaping from local optimum and reducing the computational time. To achieve these goals, this research addresses several objectives as follows:

1. To propose a new mechanism that can dynamically set the initial temperature based on the solution space by applying several iterations, and calculate the deviation average of the differences between the quality of the current solutions and the new solutions (neighbours). Then, using the obtained deviation average, the initial temperature can be initialized according to the acceptance criteria ratio (determined earlier).

2. To identify an effective cooling schedule that can adjust the decrement amount of the temperature properly based on the initial temperature value by investigating several cooling schedules and choosing the best among them.

3. To propose a new neighbourhood structure that can enhance the SA performance in searching for a good quality solution by minimizing the random neighbour selection, by calculating the total soft constraints violations for each timeslot in a day and summing them up for all the days in the week, swapping the highest penalty timeslot with a randomly selected timeslot.

4. To enhance the performance of the SA in escaping from local optimum and to divert the search to other promising regions by designing two mechanisms:

   - Propose a new shaking procedure on a non-accepted neighbour solution (saved in memory).
   - Propose a new hybrid of SA with memories. The first short term memory is used to avoid getting caught within cycles (tabu list), whilst the second short term memory (allowable memory) is to memorise several promising non-visited solutions in order to reuse one of these solutions when the search is trapped in a local optimum.

5. To propose a mechanism to choose a suitable neighbourhood structure selection mechanism according to the neighbourhood structure improvement history in the search space by calculating the improvement strength history for each neighbourhood structure during the search process. Then, the neighbourhood with the best improvement history strength will be applied. Namely, one neighbourhood structure is applied per iteration.

In addition, the proposed work is to be applied in a new real world course timetabling problem at the UKM’s Faculty of Engineering (that has additional constraints which do not exist in the current benchmark datasets).

1.4 RESEARCH SCOPE

Two different datasets will be used to test the proposed approaches. The first dataset is the standard benchmark dataset, which is a curriculum-based course timetabling problem of ITC2007-Track 3 (with 32 instances). The second one is a dataset from the Faculty of Engineering, Universiti Kebangsaan Malaysia (Semester 2008/2009). The experimental results for the first dataset are compared with other approaches from the literature that were tested on the ITC2007 track^3, whilst, the second dataset is compared with the current semi-automated timetable in UKM-Faculty of Engineering. This work was focused on improving the SA components and reducing the drawbacks in order to enhance the SA performance.

1.5 THESIS OVERVIEW

This thesis consists of eight chapters. Chapter I discusses the background of the timetabling problem, the research overview, motivations, research questions, objectives and scope.

Chapter II discusses work related to this research in order to identify the strengths and weaknesses that motivated this investigation to overcome the weaknesses and increase the strengths of the SA algorithm.

Chapter III illustrates the research methodology of this study. The research method consists of four phases: the initial, pre-processing, construction and improvement phase.

Chapter IV focuses on improving the SA performance by investigating the effects of the SA components: temperature, cooling schedule and neighbourhood structure on the performance of the SA. The first three research questions (Q1-Q3) will be answered in this chapter.

Chapter V proposes a hybrid SA with a memory (SAM) in order to escape from local optimum. The SAM tries to jump to another promising region by employing non-acceptable solutions that are saved in the memory. This chapter also proposes a new shaking procedure to divert the solution by using adaptive soft constraint weights. In this chapter an attempt is made to answer the fourth question (Q4).

Chapter VI proposes a hybrid SA with two types of memories. The first short term memory is the Tabu List(TL) in order to avoid repetition; whilst, the second short memory is the Allowable Memory, which is meant to save several promising non-visited solutions as a diversification strategy in order to decrease the probability of being re-trapped in the same local optimum. Furthermore, this chapter proposes a new neighbourhood combination by switching among the neighbourhoods according to the improvement strength history for each one during the search process. In this chapter, an attempt is made to answer the research questions (Q4 and Q5).

Chapter VII summarizes and presents an analysis of the results that were obtained by the implementation of the approaches in this research (as presented in Chapters Four to Six) to solve the university course timetabling problem.

Chapter VIII reports the findings and the research contributions of this study and the recommendations for future work.

CHAPTER II

BACKGROUND AND LITERATURE REVIEW

2.1 INTRODUCTION

This chapter will discuss some related work to identify the strengths and weaknesses of previous works in the literature that motivated this research to overcome the weaknesses and increase the strengths of the algorithm. However, the focus will be on the mechanisms/components that affect the performance of the SA algorithm.

Section 2.2 presents an introduction of the heuristic and metaheuristic algorithms. Section 2.3 reviews the use of the A algorithm to solve the university course timetable problem and other problems (i.e. exam and school timetabling problems). Section 2.4 presents a general introduction to timetabling, focussing especially on educational timetabling. Section 2.5 presents revisions on successful metaheuristic algorithms from the literature on the university course timetabling problem and others problems. Finally, Section 2.6 presents a brief summary and conclusion of this chapter.

2.1 HEURISTICS AND METAHEURISTICS

In operations research (OR),any problem is modelled quantitatively, whilst in artificial intelligence (AI), it is modelled as a graph (i.e. tree, network or others). After the problem is modelled, the problem needs to be solved by exploring different solutions in order to choose the best one among them (search algorithm).

Generally, the optimality consists of two types of solutions: global and local optimal solutions. The local optimal solution is the best solution among others within the search space region, but not in all the regions in the search space, whilst the global optimal solution is the best solution among all solutions in the search space in all the regions. In order to find a good local optimal solution/probable global optimum, an effective search mechanism is required to help explore promising regions in the search space. In every search process there are three common phases: (i) the initial solution (initial start or point); (ii) the technique to generate new solutions; and (iii) the termination criteria. There are three types of search mechanisms (Turban, 1990):

i. *Analytical Search*: It uses a mathematical function as a guide to find the optimal solution if possible (exist). By using this search mechanism there is no guarantee that the optimal solution will be reached in every case, but it can reach the local optimum.

ii. *Blind Search*: It is also called the unguided search, where the search space is enumerated to search for the optimal solution exhaustively, and there is no guarantee that the optimal solution will be found.

iii. *Heuristic Search*: It is also called a guided search and, just like the other search types, there is no guarantee that the optimal solution will be found, but it can produce satisfactory solutions/high quality solutions in most cases. This thesis focuses on the heuristic search.

Simon (1960) said: "*We now have the elements of a theory of heuristic (as contrasted with algorithmic) problem-solving; and we can use this theory both to understand human heuristic processes and to simulate such processes with digital computers.*" This means that a heuristic search does not guarantee that the optimal solution will be found but it has a procedure to work with. Then Glover (1986) argued that by combining the basic heuristic search methods in high level frameworks the search space can be explored more effectively than the basic heuristic method. This is called a metaheuristic. Osman and Laporte (1996) defined a metaheuristic as: "*an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space, where learning strategies are used to structure information in order to find efficiently near-optimal* *solutions.*" Blum and Roli, (2003) said: "*Metaheuristics are typically high-level strategies which guide an underlying, more problem-specific heuristic, to increase their performance.*"

Similar definitions can be found in (Glover 1986; Glover & Laguna 1993, 1997; Osman & Kelly 1997). A metaheuristic starts with one solution or more initial solutions and employs search mechanisms/strategies to search for a better quality solution. There are different ways to categorize metaheuristics algorithms. Birattari et al. (2001) and Blum and Roli (2003) categorized metaheuristic algorithms as follows: Trajectory methods *vs.* discontinuous methods; one neighbourhood *vs.* various neighbourhood structures; population based *vs.* single point search; nature inspired *vs.* non-nature inspiration; dynamic *vs.* static objective function and memory usage *vs.* memoryless methods .

Table 2.1 Metaheuristic categories

Category	**Method**	**Example**
Trajectory methods or Single point	Consider only one solution and jump from one spot to the other in the search space in order to reach a good, promising region	SA (Abramson, 1991); TS (Alvarez et al.,2002)
Discontinuous methods or Population based	Consider a set of points in the search space	GA (Socha et al., 2003; Eley, 2007); evolutionary search (Beligiannis et al., 2008)
One neighbourhood	Using one neighbourhood structure	TS
Various neighbourhood structures	Using set of neighbourhood structures	VNS
Nature inspired	Inspired by nature	ACO
Non-nature inspiration	Not inspired by nature	TS and ILS
Memory usage	Using memory (search history during the search process)	TS

Memoryless methods	Without memory	SA

The next section gives a review of the implementation of the SA algorithm to solve UCTP and others, where the objective of this thesis is to enhance the SA performance.

2.3 REVIEW OF SIMULATED ANNEALING ALGORTHIM

SA is a stochastic optimization technique inspired by the Metropolis algorithm for statistical mechanics (Metropolis et al., 1953). Annealing is the process of cooling heated materials or molten crystalline solids. The target of this cooling process is the alignment of atoms in the most regular possible crystalline structure. Kirkpatrick (1983) and Cerny (1985) showed that the annealing model for solids, which was proposed by Metropolis et al. (1953), could be used for optimization problems where the (minimizing or maximizing) of the objective function corresponds to the status energy of the metal. Figure 2.1 shows a pseudo-code for SA (Hoos & Stützle 2005).

```
Determine initial candidate solution Sol;
Set the initial temperature T;
While (termination criterion not met);
        choose neighbour Sol' of Sol using probabilistic acceptance criteria;
        If Sol' is accepted using probabilistic acceptance criterion (depending on T):
Sol = Sol' ;
        Update T  (using annealing schedule);
Return best solution found
```

Figure 2.1 The basic SA pseudo code

SA starts once the initial solution, *Sol*, is generated (randomly or by using constructive techniques). At each iteration,the SA generates a neighbour solution,*Sol'* of *Sol*. The *Sol'* is accepted if it improves the cost value *f(Sol)* of *Sol* (where $\Delta = f(Sol') - f(sol) \leq 0$).Otherwise,*Sol* is accepted if$p = e^{-\Delta/T}$> generated random number between 0 and 1. *T*is the current temperature. At the beginning of the search, *T*is high and it is decreased during the search following the use of a cooling scheme. The SA stops when the *T* value is close to zero (frozen temperature), where no more worsening moves can be accepted.

2.3.1 SA components

Generally, when applying the SA algorithm, there are certain common components that need to be defined in order to obtain an effective SA algorithm. These are: (i) initial temperature; (ii) cooling schedule; and (iii) neighbourhood structure.

I. Initial temperature

The initial temperature causes the SA to walk randomly over the landscape. Thus, a high or infinite initial temperature would be the best choice (Triki et al., 2005). On the other hand, a high or infinite initial temperature will cause the SA to take a longer time to cool down (Triki et al., 2005).

Kirkpatrick et al. (1983) computed the initial temperature by starting with a high initial temperature. Then, they applied several transitions (moves) using this temperature. The accepted transition ratio was then compared with χ_0, (where χ_0 is a value given between 0 and 0.99). If the accepted transition ratio was less than χ_0, then the initial temperature was multiplied. The procedure was repeated until the acceptance ratio exceededχ_0.

Johnson et al. (1989 &1991) proposed a technique/procedure to obtain the initial temperature using Equation 2.1, where Δ^+ is an increased cost (positive transition).

$$T_0 = -\Delta^{+}{}_{/In(\chi_0)} \tag{2.1}$$

where

$$\chi_0 = \exp(-\Delta^+ / T_0) \tag{2.2}$$

The initial temperature is computed based on the bad transitions (worst quality solution ratio), where Δ is the difference between the current solution, *Sol,* and the new neighbour solution, *Sol'*, $\Delta = f(Sol') - f(Sol)$.

Srichander (1995) examined the SA computational time performance by using spectral decomposition of matrices. He showed that starting the SA with high initial temperatures produced an ineffective search in the sense that the number of the evaluation functions that had to be applied to obtain the global minima was very large. Unfortunately, he discovered that the SA consumed more computational time. On the other hand, other researchers, such as (Grover, 1986; Johnson, 1989; Varanelli, 1996), applied the SA with a low initial temperature so that it would consume less computational time.

Rayward-Smith (1996) started the initial temperature with a very high value, and then decreased it until the acceptance probability was equal to 60%. Aarts et al. (1997) suggested another technique to compute the initial temperature, $\boldsymbol{T_0} = \propto_{\infty}^{\mathbf{2}}$, where $\propto_{\infty}^{\mathbf{2}}$is the solution quality when the temperature is ∞.

Varanelli and Cohoon (1999) proposed a technique to choose the initial temperature for a two-staged SA. They concluded that a low initial temperature from the beginning will cause the SA to be trapped in a local optimum very fast, whilst if the initial temperature is very high, the SA will consume more computational time by accepting many worst solutions. Furthermore, in their two-staged SA, they started with a high initial temperature; then, they replaced it with a lower temperature in order to improve the solution quality (more convergence).

Poupaert and Deville (2000) proposed a rule for choosing the initial temperature based on the initial acceptance ratio, χ_0, which is defined as the number of bad transitions that are accepted divided by the number of attempted bad transitions, and the average increase in the objective function value (Equation2.3).

$$\chi_0 = \chi\ (\{\delta_1,\ \delta_n,\ \delta_{n+1},\ \ldots,\ \delta_m\},\ T_0)$$

$$\chi_0 = \frac{1}{m}\sum_{i=1}^{n} exp\ (-\delta_i/\ T_0) + \frac{m-n}{m} \qquad (2.3)$$

where $\chi_0$is the starting acceptance probability between 60% and 80%, T_0 is the starting temperature; $\delta_i = f(s_i) - f(s_0)$,$s_0$ is the initial solution,s_i is the new neighbour of s_0, f(s_i)

is the objective function for s_i, and m is the neighbour's solution space size. They repeated the above procedure until the acceptance ratio exceeded χ_0 .e $\chi_0 > 0.8$).

However, Poupaert and Deville (2000) estimated the temperature during the search process. Thus, the temperature was no longer the control parameter. Instead the acceptance probability became the control parameter. The value of the initial temperature was selected according to the acceptance probability criteria during the search process.

Zhang et al. (2010) determined the initial temperature by starting with a very high initial temperature, and then tried to derive the real start temperature by using the functional dependence between the starting acceptance probability χ_0 (e.g. 0.95 or 0.9) and the initial temperature T_0. They used the same equation proposed by Poupaert and Deville (2000) (see Equation 2.3). The proposal byZhang et al. (2010) can still initialize high temperatures because it creates the initial temperature from the first iteration using the first Δ. Aycan and Ayav (2009) used the same equation (Equation 2.3) but they ran several iterations before the SA algorithm started in order to determine the initial temperature. Table 2.2 summarizes the presented initial temperatures that were reviewed in this section.

Table 2.2 Summary of the presented initial temperatures from the literature

Author	*Mechanism*	*Disadvantages*
Kirkpatrick et al. (1983)	Starting with a high initial temperature value, and then estimate it by comparing the accepted transitions ratio, apply for several bad transitions with a given value	The temperature is not a control parameter anymore and the SA can trap in local optimum very fast
Johnson et al. (1989, 1991)	Compute the initial temperature using the starting part of the SA	The initial temperature is still high especially after the first part of the process,

		means more computational time
Rayward-Smith, 1996	Start the initial temperature with a very high value, and then decrease it until the acceptance probability is equal to 60%	Still can consume more computational time at the starting part of the process
Poupaert and Deville (2000)	Estimate the temperature during the search process	The temperature is not the control parameter anymore but the acceptance probability
Aycan and Ayav (2009	Select the initial temperature before the algorithm starts, comparing the bad transition ratio for the first current solutions with a given value	The temperature is still high after the first part of the search process
Zhang et al. (2010)	Initialize the initial temperature at the first stage of the algorithm	The temperature is still high, especially after the first part of the process, which consumes more computational time

According to the above literature, many researchers have attempted to select the initial temperature in order to avoid consuming more computational time and to avoid being quickly trapped in local optimum. Therefore, this thesis tried to dynamically initialise the initial temperature properly in order to select a suitable initial temperature for each problem independently.

II. *Cooling schedule*

The temperature value in the SA isdecreased during the search process using a procedure called cooling schedule. Romeo and Vincentelli (1991) presented the following procedure in order to design the cooling schedule:

a) *Start the procedure at an initial temperature of T_0, for which a satisfactory approximation of the steady distribution, D_{T0}, is quickly reached.*
b) *Reduce T_0 by a small amount, $\alpha(T)$, such that D_{T0} is a good starting point in order to approximate D_{T0} - $\alpha(T)$.*

The above process is then repeated until there are no more improvements. The cooling schedule can be grouped into two categories:

i. *The static schedule*, which must be specified at the beginning (before the algorithm starts). An example of a static cooling schedule is theGeometric Cooling Schedule (GCS) proposed by Kirkpatrick et al. (1983), which is the most popular cooling schedule in terms of the simplicity in obtaining it. In GCS, the temperature is reduced, as in Equation 2.4.

$$T_{i+1}=\alpha \,.\, T_i \qquad (2.4)$$

where ti+1 is the temperature after i+1 iterations; α ($0< \alpha<1$) denotes the cooling rate or factor.

ii. *The adaptive schedule* adjusts the temperature decrement during the algorithm process according to the information obtained (e.g. use the objective values for each level to decide the decrement amount),namely, the temperature decrement amount in the next iteration is obtained based on the run history. An example of an adaptive cooling schedule was presentedby Laarhoven et al. (1988), which proved that this cooling schedule terminates in polynomial time, as in Equation 2.5:

$$T_{i+1} = T_i \cdot \frac{1}{1 + \frac{In(1+\delta)}{3\sigma(T_i)} T_i} \tag{2.5}$$

Where δ is a real number (small). The purpose of this adaptive cooling schedule is to maintain the search space spots (solutions) close to each other. Another adaptive cooling schedule was presented by Lewis et al. (2007) (Equations 2.6 – 2.8),

$$\lambda_0 = 1 - \beta \tag{2.6}$$

$$\lambda_{i+1} = \lambda_i + \frac{\beta + \beta}{M} \tag{2.7}$$

$$T_{i+1} = T_i - \lambda_{i+1}(\frac{T_0}{M}) \tag{2.8}$$

where β represents a parameter that at each step helps to determine a value for λ. λ is used to influence the amount of concavity or convexity present in the cooling schedule; M represents the number of steps taken by the temperature T_0 to decrease it to a value close to zero T_{min}.Lewis et al. (2007) used this cooling schedule to get the initial feasible solution, where set M is equal 100, T0 = 10; β = - 0.99.

However, Triki et al. (2005) presented two contradictory goals that adaptive cooling schedules are trying to reach:

i. The annealing temperature must remain as close as possible to balance the intensification and diversification.
ii. The annealing temperature procedure should be as short as possible.

Furthermore, Romeo and Vincentelli (1991) discussed several cooling schedules in terms of the initial temperature, number of iterations, inner loops to decrease the current temperature value, the decrement amount of the current temperature (i.e., cooling rate), and the SA stopping criteria. They concluded that the theoretical results obtained so far have not been able to explain why theSA is so successful even when a diverse collection of static cooling schedule heuristics is used. However, through the enormous number of cooling schedules presented in the literature, the effectiveness of these schedules can only be compared through experimentation. Romeo and Vincentelli (1991) conjectured that the neighbourhood and the corresponding topology of the objective function are also responsible for the behaviour of the SA. Srichander (1995) suggested that the cooling schedule is not necessary for the SA to converge a final solution to the global optimum.

Thompson and Dowsland (1996) presented and investigated the cooling schedules for a wide range of examination timetabling problems on universities, and suggested that a very slow cooling schedule should be used. Elmohamed et al. (1998) investigated three types of cooling schedules: geometric, adaptive and adaptive with reheating. The experimental results showed that the SA with an adaptive cooling schedule and reheating function is able to generate competitive results compared to the state-of-the-art techniques, and outperform other cooling schedules.
Abramson et al. (1999) used the SA to solve the problem of school timetabling. Furthermore, they applied six different cooling schedules: (i) a basic geometric cooling schedule, (ii) geometric reheating, (iii) a cooling scheme that used multiple cooling rates(different cooling rates), (iv) a non-monotonic cooling schedule, and two cooling schedules with reheating functions, (v) cost as a reheating parameter, and (vi) enhanced geometric reheating. For the purpose of comparison, they used the basic geometric cooling schedule presented by Laarhoven and Aarts (1988). They found that applying multiple cooling rates yielded a better result according to the obtained experimental results, and outperformed the single cooling schedule in terms of consuming less computational time.

Fielding (2000) presented an empirical study of several static cooling schedules for solving the problem of travelling salesmen. They concluded that a static cooling schedule can lead to optimal and near optimal solutions. Azizi and Zolfahgari (2004) presented a new adaptive cooling schedule that updates or manages the temperature dynamically based on the search path profile. The temperature can be changed in any direction: cooling or reheating and based on the number of consecutive improving/non-improving moves.

Chainate et al. (2008) presented a modified SA (MSA) to solve the problem of university course timetabling,where the MSA included a repair process that repairedanyunfeasible solutions that were found. They discovered that the solutions generated in minimum violations were obtained by using the regeneration as a neighbourhood search and the cooling schedule presented by Lundy and Mees (1986) (similar to that by Laarhoven et al. (1988))in Equation2.9,

$$T_{i+1} = T_i \cdot \frac{1}{1 + \frac{\lambda}{\upsilon} T_i} \tag{2.9}$$

where λ is a small number and υ is the difference between the current solution and the new solution, υ = *f(Sol) – f(Sol')*. More cooling schedules are described in Otten (1984), and Huang et al. (1986). Table2.3 summarizes the presented cooling schedules and cooling schedule investigations that were revised in this review.

Table 2.3 Summary of the presented cooling schedules and investigations from the literature

Author	***Mechanism***	***Advantages***	***Disadvantages***
Kirkpatrick et al.1983)	Proposed the geometric cooling schedule,$T_{i+1}=\alpha.\ T_i$, where α is the cooling rate between zero and one, and T_k is the current temperature value	Most popular cooling schedule in terms of simplicity in application	Static cooling schedule, must be specified before the search starts and it needs to be carefully tuned
Laarhoven et al. (1988)	Proposed a new cooling schedule that terminates in polynomial time	Maintains the search space spots (solutions) to be close with each other	------
Elmohamed et	Investigated several	They found that	-----

al. (1998)	cooling schedules (geometric, adaptive and adaptive with reheating)	the adaptive cooling schedule with reheating performed better than others	
Triki et al (2004)	Discussed and investigated different cooling schedules. Furthermore, they formulated a new practical annealing schedule for a given objective function	They showed that the presented logarithmic cooling schedule was better than any of the others to ensure the SA convergence	Need to compare the obtained cooling schedules with other adaptive schedules, in order to make a fair comparison
Azizi and Zolfahgari (2004)	Adaptive cooling schedule that update or manage the temperature dynamically based on a number of consecutive improving moves	Adjust the temperature based on the search path profile	It need to tune the consecutive improving iterations carefully
Lewis et al. (2007)	They presented formulas to adjust the temperature decrement amount during the search process, in order to generate the feasible solution	It adjust the temperature decrement amount during the search process properly	It take to much computational time if the value of the initial temperature is high, therefore, the initial temperature need to be selected carefully (not high)
Chainate et al.	They investigated a two	They observed	The initial and final

(2008)	schedules,(geometric and Lundy & Mees (1986) cooling schedules) in order to study the significance performance for each one	that using the cooling schedule that presented by Lundy and Mees (1986) is obtained better results	temperatures are not statistically significant. Thus, the initial and final temperatures should be carefully tuned

Based on Table 2.3, many cooling schedules have been presented in the literature with different conclusions, where the strength of any cooling schedule is determined by comparing it with others. Therefore, to overcome the problem of choosing a suitable cooling schedule, this study investigated both static and adaptive cooling schedules.

III. Neighbourhood structures

The solution neighbourhood is defined as: *"a neighbourhood structure which is a function N: $S \rightarrow 2^S$ from the solution space, S, to the power set of the solution space that assigns a set of neighbours, N(s), to every solution"* Blum et al. (2005).

One of the important features of a local search algorithm is the definition of its neighbourhood (Lü and Hao 2010). The effective and good neighbourhood structure positively influences the SA performance (Moscato 1993).

In fact, choosing the best neighbourhoodstructure or the size of each neighbourhood are critical issues in terms of improving the solution quality and the SA efficiency (Solla et al., 1986; Eglese, 1990; Moscato, 1993 and Fleischer & Jacobson, 1999). Moreover, some researchers have reported that a small neighbourhood structure size is the best (Cheh et al., 1991),whilst others have proven that a large size is better in order to improve the SA performance (Ogbu & Smith, 1990). Logically, a large sized neighbourhood structure leads the SA to behave like a descent method, whilst a small sized neighbourhood structure causes difficulty in finding a reasonably good quality solution within a reasonable time, where the Markov chain is unable to move around the search space fast enough (Goldstein &

Waterman, 1988). However, the neighbourhood structure can strongly affect the total uncertainty associated with the SA or the information rate (Fleischer, 1994; Fleischer & Jacobson, 1999).Eglese (1990) stated that: "*A 'smooth' topology with shallow local minima which is imposed by a neighbourhood structure is preferred, rather than a 'bumpy' topology with many deeply local minima*".

Triki et al. (2005) presented an empirical study on the neighbourhood structure effect on the performance of the SA algorithm. The experimental results showed that a good neighbourhood is crucial for the SA performance in finding good quality solutions. Bouffard and Ferland (2007) proposed a technique to enhance the SA algorithm by combining it with a variable neighbourhood search in order to solve the problem of resource constrained scheduling. They compared their technique numerically with other techniques (neighbourhood search techniques), which are the TS and threshold accepting methods. The obtained results demonstrated that the VNS enhanced the SA performance by improving the search ability and solution quality.

Abdullah et al. (2010) applied a Round Robin (RR) algorithm, which is used to control the selection of neighbourhood structures, instead of an improvement algorithm called a dual-sequence SA (DSA) on post enrolment course timetabling problems. They showed that the performance of the proposed approach was able to generate competitive results compared with other state-of-the-art techniques.

Zhang et al. (2010) presented an SA with a new neighbourhood structure for high school timetabling problems in order to search for the best neighbour by performing a sequence of swaps between pairs of timeslots instead of the standard swap between two assignments in a standard SA.

In conclusion, it was found in this study that a neighbourhood structure will influence the performance of the SA. Therefore, in this thesis a good neighbourhood structure is suggested that may enhance the quality of the solution. Furthermore, a new neighbourhood structure combination (operator) is also proposed in order to select the best suitable neighbourhood during the search process so as to avoid the disconnected neighbourhood and save on the computational time.

2.3.2 Hybridization of SA with others

As in many metaheuristic techniques, the SA has a chance to revisit the same solution (repetition), getting trapped in a local optimum and taking a longer time to find areasonably good solution. This problem will lead to extra computational time without any improvement. Therefore, the use of a memory in SA is an effective way to overcome the problem of repetition (Osman, 1993). For example, (Osman, 1993; Zolfaghari & Liang, 1999; Swarnkar & Tiwari, 2004; Jeon & Kim, 2004) combinedSA and TS.

Patrick (2012) compared the performance of the SA and HC in solving the university course timetabling problem. It was found that both algorithms were good choices for selection. Both algorithms could perform better if applied with other alternative methods, such as GA, TS, ILS and VNS. The experimental results showed that the SA performed better than the HC.

Kalender et al., (2012) presented a combination of the SA with an improvement-oriented heuristic selection mechanism to solve a curriculum-based course timetabling problem at Yeditepe University, where a hyper-heuristic was developed to handle different problem domains with different properties. The aim of this hyper-heuristic development was to build a general method for variant problems. However, they applied it for six other problem domains and it proved to be comparable with various other hyper-heuristics. The results showed that the proposed method was sufficiently general and powerful.

Stephen et al. (2012) presented a hybrid approach that combined the SA and GA to solve the examination timetabling problem. They exploited useful features from each algorithm to obtain a good quality solution. The experimental results showed that the presented hybrid approach could produce a good quality solution within a reasonable computation time.

Gunawan et al., (2012) presented a combination of SA and constraint programming techniques to solve master course timetabling problems. The search started by generating the initial solution using a Largrangian relaxation technique and the given solution was improved by using the simulated SA as a second stage. Zhang et al., (2010) presented an SA with a new neighbourhood structure for high school timetabling problems in order to search for the best neighbour by performing a sequence of swaps between pairs of timeslots instead of the standard swap between two assignments in a standard SA.

Azizi et al. (2009) proposed a hybrid SA with an evolution-based diversification approach called SAMED to solve job shop scheduling problems. SAMED includes SA, three types of memories and a GA-based crossover component. The first two types of memories are short term memories (tabu list and seed memory list), while the third type is long term memory. SAMED uses the short term memory (tabu list) to temporarily save the accepted solutions to avoid repetition; whilst the second short term memory (seed memory) is to keep track of the best solutions with the lowest objective functions for further improvement.

Frausto-Solís et al. (2008) presented a hybrid SA with a TS named Fralonso. Fralonso is applied for the Post Enrolment Course Timetabling ITC track2. The hybrid approach starts with an SA which adds an extra timeslot in the hopeof reaching a feasible solution. If a feasible solution is not found, the SA continues without any extra timeslots. Otherwise, if a feasible solution is still not found, the SA optimization is performed to find a feasible solution and to improve the solution quality (with a new temperature). Finally, the TS is employed when the stop criterion of the SA is met (frozen temperature or the time limit).

Abdulla and Hamdan (2008) presented a hybrid approach consisting of three stages. The first stage was to generate the initial solution using a constructive heuristic, whilst the second stage tried to improve the quality of the solution by employing a randomized iterative algorithm with an SA and a composite neighbourhood structure. Finally, they applied hill climbing in the third phase to further improve on the quality of the solution. However, they found that the second phase improved the solution quality better than the hill climbing phase.

Suman et al. (2005) combined the SA with a fractional factorial analysis called orthogonal SA in order to enhance the solution convergence and accuracy. The work involved several factorial experiments to determine the best combination of factors. They evaluated the orthogonal SA by solving several global optimization problems. The orthogonal SA experimental results showed that the orthogonal SA method outperformed the SA in solving global optimization problems with a linear or nonlinear objective function.

Gao et al. (2006) presented a hybrid meta-heuristic algorithm by combining the characteristics of the SA, GA and Chaos strategy to solve the TSP problem. The experimental results showed that the hybrid metaheuristic algorithm (CASAGA) was quite flexible with satisfactory results. Hint: The Chaos strategy is a well-known phenomenon that exists in the nonlinear system. It has a better capacity for hill climbing by using intrinsic stochastic properties to find the optimum solution.

Duong and Lam (2004) combined constraint programming and the SA to solve examination timetabling problems on real world datasets. They concluded that the SA can solve any difficult scheduling problems. On other hand, it is very important to choose the SA components carefully according to the problem instances. They also applied a Kempe chain as the neighbourhood structure and a special mechanism for determining the initial temperature. Furthermore, the constraint programming phase was used to generate the feasible initial solution. The experimental results showed that a very slow cooling schedule should be used, and it proved that the hybrid approach was a promising approach among several approaches implemented for the examination timetabling problem.

Zolfaghari and Liang (2002) presented a comparative study of SA, GA, and TS in order to test the performance of each algorithm. They implemented their algorithms to solve the varying types of binary machine grouping problems, which involved machine/part types, sizes, machine capacities and processing times. The comparisons were made in terms of solution quality, pre-search effort and the behaviour of the search convergence. It was indicated that the SA outperformed both the GA and TS mostly for large problems.