Some Improved Genetic-Algorithms Based Heuristics for Global Optimization with Innovative Applications

Aderemi Oluyinka ADEWUMI

A thesis submitted to the Faculty of Science, University of the Witwatersrand, Johannesburg in fulfillment of the requirements for the degree of Doctor of Philosophy

Johannesburg, 2010

Declaration

I declare that this thesis is my own work. It is being submitted for the Degree of Doctor of Philosophy in the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination in any other University.

____

(Signature of candidate)

________________________day of _________________________20_________________

Abstract

The research is a study of the efficiency and robustness of genetic algorithm to instances of both discrete and continuous global optimization problems. We developed genetic algorithm based heuristics to find the global minimum to problem instances considered.

In the discrete category, we considered two instances of real-world space allocation problems that arose from an academic environment in a developing country. These are the university timetabling problem and hostel space allocation problem. University timetabling represents a difficult optimization problem and finding a high quality solution is a challenging task. Many approaches, based on instances from developed countries, have been reported in the literature. However, most developing countries are yet to appreciate the deployment of heuristics and metaheuristics in handling the timetabling problem. We therefore worked on an instance from a university in Nigeria to show the feasibility and efficiency of heuristic method to the timetabling problem. We adopt a simplified bottom up approach in which timetable are build around departments. Thus a small portion of real data was used for experimental testing purposes. As with similar baseline studies in literature, we employ genetic algorithm to solve this instance and show that efficient solutions that meet stated constraints can be obtained with the metaheuristics.

This thesis further focuses on an instance of university space allocation problem, namely the hostel space allocation problem. This is a new instance of the space allocation problems that has not been studied by metaheuristic researchers to the best of our knowledge. The problem aims at the allocation of categories of students into available hostel space. This must be done without violating any hard constraints but satisfying as many soft constraints as possible and ensuring optimum space utilization. We identified some issues in the problem that helped to adapt metaheuristic approach to solve it. The problem is multi-stage and highly constrained. We first highlight an initial investigation based on genetic algorithm adapted to find a good solution within the search space of the hostel space allocation problem. Some ideas are introduced to increase the overall performance of initial results based on instance of the problem from our case study. Computational results obtained are reported to demonstrate the effectiveness of the solution approaches employed.

Sensitivity analysis was conducted on the genetic algorithm for the two SAPs considered to determine the best parameter values that consistently give good solutions. We noted that the genetic algorithms perform well specially, when repair strategies are incorporated. This thesis pioneers the application of metaheuristics to solve the hostel space allocation problem. It provides a baseline study of the problem based on genetic algorithms with associated test data sets. We report the best known results for the test instances.

It is a known fact that many real-life problems are formulated as global optimization problems with continuous variables. On the continuous global optimization category therefore, we focus on improving the efficiency and reliability of real coded genetic algorithm for solving unconstrained global optimization, mainly through hybridization with exploratory features. Hybridization has widely been recognized as one of the most attractive approach to solving unconstrained global optimization. Literatures have shown that hybridization helps component heuristics to taking advantage of their individual strengths while avoiding their weaknesses. We therefore derived three modified forms of real coded genetic algorithm by hybridizing the standard real-coded genetic algorithm with pattern search and vector projection. These are combined to form three new algorithms namely, RCGA-PS, RCGA-P, and RCGA-PS-P. The hybridization strategy used and results obtained are reported and compared with the standard real-coded genetic algorithm. Experimental studies show that all the modified algorithms perform better than the original algorithm.

Keywords: Unconstrained global optimization, genetic algorithms, space allocation, hostel space allocation problem, timetabling, pattern search, vector projection, heuristics, metaheuristics, hierarchical heuristics.

Declaration

To Him whose love, care and strength saw me through life journey up till this point of achievement and I still believe will see me through till the end – the **Lord JESUS CHRIST**

Acknowledgements

To the only wise God, immortal, invisible whose eyes runs to and fro to show himself strong on behalf His own, my Lord and Saviour, be all glory, honour, wisdom, power and majesty. Without Him, I could do nothing, without Him, I would have failed, without Him I would be drifting, like a ship without a sail. I love Thee Lord.

To the very centre of My Heart (MH), the One who sacrificed her time, talent and even gave up her lucrative professional job in order to stay by my side and encourage me till our desired dream is achieved, my one and only darling, Olaoluwa. I owe you a lot.

To the olive branch surrounding our table, the arrows in the hand of the Almighty, Jeremiah (Tolu), Joy (Tosin), Julie (Tofunmi) and John (Temi) – I know your sacrifice, months you had to miss dad with the after-effects even on your school performance; your constant prayers and intercession for God to see daddy through in his PhD. I believe your prayers are answered. I love you all, my Jewels.

To my mother, Juliana Monilola Adewumi, I cannot forget those early days, when you determined that come rain come sunshine, the privilege you personally missed, you will not allow your children to miss it. Days of selling what you had to pay my fees and also days of weeping, when you did not have enough to help me. And here we are today. Thank you Mum, I love you.

To my spiritual leader, my Father in the Lord, the Model for our generation, Pastor (Dr) W.F. Kumuyi, you are a great man of God indeed. The Lord used you to release me from “Egypt” at the right time He (God) has prepared a “land flowing with milk and honey for me”. Though, not a single word of prayer as others would expect, but I can never forget your words, “God will go with you, everything will succeed”, “God will see you through”. Lo! They all came to pass. Thank you sir.

To all the brethren and beloved compatriots of the Deeper Christian Life Ministry especially those in DLCF and DLSO, it’s great to be in the family of God - you all rallied round, supported in prayers and encouragement when the battle was tough, we fought together and won the victory together. I specially thank Dr. Johnson Olaleru for his brotherly care and support even when distance separated us, you were always there to help. Thank you ALL.

To the staff and my past students in Department of Computer Science, University of Lagos, I cannot forget the period we spent together. You may not know, but God used everyone to contribute their part. I am particularly grateful to Mr Sawyerr, Mr. Gbenga Bastos and Ms Nnenna Ihemedu for their assistance with data gathering and other aspects of the work. I appreciate you all.

To Professor David Sherwell, all staff and postgraduate students of School of Computational and Applied Matheamtics, I thank you all. The financial supports received from the University of Lagos, School of Computational and Applied Mathematics, University of Witwatersrand and the African Millennium Mathematics Science Initiative (AMMSI) towards the completion of my programme are much appreciated.

Last, but not the least, to my indefatigable, selfless and humble Supervisor, Professor Montaz Ali, you are really an Angel sent by the Lord to achieve this feat. I sincerely express my heartfelt gratitude to you. It was your encouragement, commitment and support that resulted in the production of this thesis. I sincerely appreciate the many hours you spent to make this thesis possible. Your insight, positive criticism and comments have been invaluable. You never saw any useless idea in me, rather you reconstructed them until the job is well done. Thank you Prof.

Adewumi, A.O.

January, 2010

Table of Contents

List of Figures

List of Tables

Abbreviations

Nomenclature	Meaning/Interpretation
SAP	Space Allocation Problem
GA	Genetic Algorithm
RCGA	Real Coded Genetic Algorithm
SRCGA	Standard Real coded Genetic Algorithm
HSAP	Hostel Space Allocation Problem
TTP	Timetabling Problem
UTTP	University Timetabling Problem
LTTP	Lecture Timetabling Problem
COP	Combination Optimization Problems
PS	Pattern Search
Fy	Final year student category as used for HSAP
Sc	Scholar category as used for HSAP
Fr	Fresher category (First year or direct-entry students) as used for HSAP
Ds	Discretionary List category as used for HSAP
Fo	Foreign student category as used for HSAP
Ot	Other student category – in other years aside Fr and Fy as used for HSAP
Ht	Health (Disabled) student category as used for HSAP
Sp	Sport men and woman category as used for HSAP
CaH1	A greedy-like heuristic developed for category allocation in HSAP
CaH2	A percentage ranking based heuristic developed for category allocation in HSAP
HaNH	A new heuristics for hall allocation in HSAP to maximize the distribution spread of students
HaGA	A genetic algorithm metaheuristics for hall allocation in HSAP
FaGA	A genetic algorithm metaheuristics for block/floor allocation in HSAP

List of Included Articles

[A] Adewumi, A.O. and Ali, M.M. A multi-level genetic algorithm for a multi-stage space allocation problem. *Mathematical and Computer Modeling* 51 (2010) 109 -126.

[B] Adewumi A.O. and Ali, M. A hierarchical heuristic strategy for hostel space allocation problem. Submitted to the *Journal of the Operational Research Society*, 2009

[C] Adewumi, A.O., Sawyerr, B.A. & Ali, M.M. A heuristic solution to the university timetabling problem. *Engineering Computations*. 26(8) (2009), 972 - 984.

[D] Sawyerr B.A., Ali, M.M. and Adewumi A.O. A comparative study of some real coded genetic algorithms for unconstrained global optimization. *Optimization Methods and Software*, to appear.

Chapter One

Introduction and Background

"The Journey of a thousand miles begins with a single step"

- Lao Tzu

"The ability to convert ideas to things is the secret of outward success"

- Henry Ward Beecher

"So many fail because they don't get started - they don't go. They don't overcome inertia. They don't begin."

- W. Clement Stone

1.0 Introduction

The optimization technique cuts across many fields of study and is applicable in all areas where a choice among given or possible alternatives is paramount. These include engineering, management science, medicine, computer science, applied mathematics to mention a few. Expectedly, different field of study view optimization from different perspective but the key issue lies in the overall goal of the whole process namely, making an optimum decision. The applicability of optimization in different disciplines makes it difficult to give a single concise definition of the concept. Mathematicians, for instance, aim to find the maxima or minima of a real function within an allowable set of variables [118]. In computing and engineering, the goal is to maximize systems or application performances with minimal runtime and resources possible. Cherkaev [36] remarks and we quote,

> "the desire for optimality (perfection) is inherent for humans. *It seems a natural instinct to search for extremes in all endeavour of life* (personal emphasis). The search for extremes inspires mountaineers, scientists, mathematicians, and the rest of the human race. The mathematical theory of optimization is developed since the sixties when computers become available. The goal of the theory is the creation of reliable methods to catch the extremum of a function by an intelligent arrangement of its evaluations. This theory is vitally important for modern engineering and planning that incorporate optimization at every step of the complicated decision making process."

Generally, an optimization model must have three main components [37,80,115,117] namely, the decision variables, representing components of the model that can be changed to create different possibilities; constraints which represent limitations on the variables; and objective function that assigns a value to different possible values of the variables. The objective function is optimized with respect to the decision variables. Mathematically speaking therefore, optimization is concerned with the study of problems that seek to minimize or maximize a real function by systematically choosing values of real or integer variables from an allowed set [121]. The optimization problem can be presented mathematically as follows:

Minimize	$f(x)$
such that	$x \in S$
where	f is real valued
and	$S = \{ x \in \mathrm{P}^n \mid g_i(x) \leq 0, h_j(x) = 0, i = 1,2,\ldots,m; j = 1,2,..,k; k \leq n \}$.

The elements of S are the candidate or feasible solutions. The function f is the objective or cost function. A feasible solution, $x \in S$, which minimizes (or maximizes, depending on the goal) f is called the optimal solution. Hereafter a feasible solution will be referred to simply as a solution.

Informally then, optimization aims at finding the values of the variables which maximizes or minimizes a given quantity subject possibly to some given restrictions on the variables.

We define some basic terminologies and concepts as follows:

Neighbourhood: If we define a distance measure between two solutions as: $dist: S \times S \rightarrow \mathrm{P}$, then for all $x \in S$, the neighbourhood of x, $N(x)$, is defined as

$$N(x) = \{y \in S \mid dist\,(x,y) \leq \varepsilon \},$$

for real values of $\varepsilon > 0$.

Local optimizer: An element $x \in S$ is a local minimizer if $f(x) \leq f(y)$ for all $y \in N(x)$ and a local maximizer if $f(x) \geq f(y)$ for all $y \in N(x)$. A local optimizer can either be a local minimizer or a local maximizer.

Global optimizer: An element $x \in S$ is a global minimizer if $f(x) \leq f(y)$ for all $y \in S$ and a global maximizer if $f(x) \geq f(y)$ for all $y \in S$. A global optimizer can either be a global minimizer or a global maximizer.

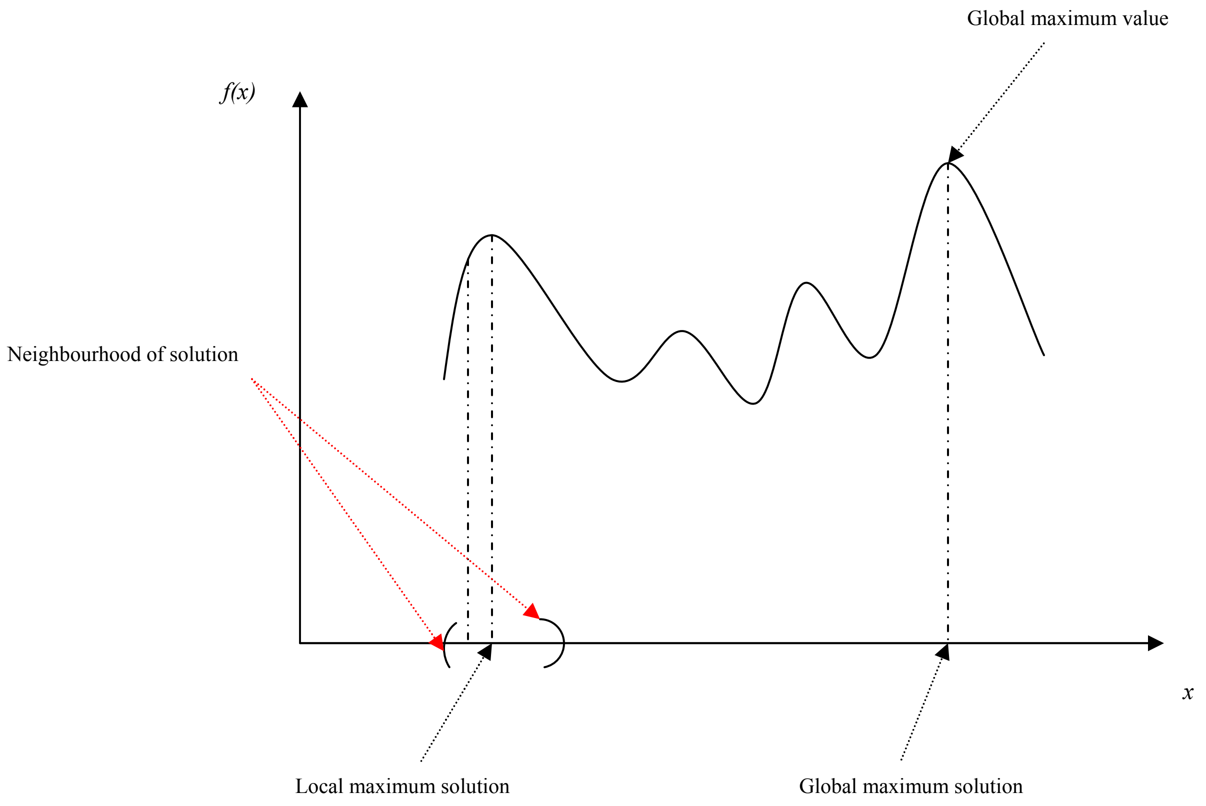

Figure 1.1: Illustration of an optimization problem

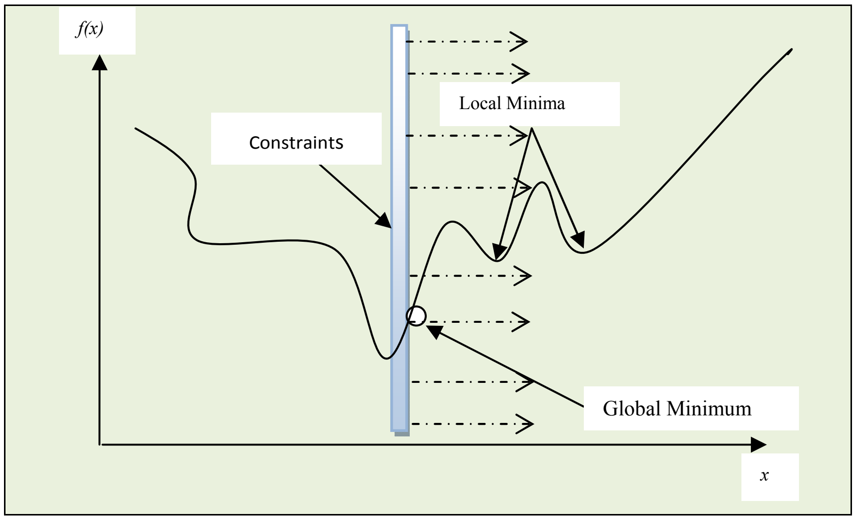

Figure 1.2: Types of minima for constrained optimization problems

Figures 1.1 and 1.2 present graphical illustrations of typical optimal points in a search space. A global optimal solution is such that there is no other feasible solution with a better objective function value within *S* while a local optimal solution has no other feasible solution within its neighbourhood with better objective function value. A lot of efficient algorithms for finding optimal solutions to some classes of optimization problems exist. However, there are still a host of real-world problems where locating the optimal solution is not trivial. A major problem with some algorithms is the inability to differentiate between local and global optimal solutions and thus the possibility of being trapped in a local minimizer. This is where global optimization comes in. The aim of global optimization is to find the best possible global solution within the feasible set, *S*. On the other hand, local optimization techniques aim at finding a good local solution.

Nonlinear optimization models are prominent in many real-world applications such as engineering design, space planning, networking, data analysis, logistic management, financial planning, risk management, and others. Solutions to these problems often require a global search approach. They are generally difficult to solve for many reasons. First, literature have reported that optimization problems arising from these applications are often NP-hard in nature [14,40,52,56,78,90,126]. Secondly, benchmark problems and real-life practical cases present some requirements and constraints which are either hard (in terms of solvability) or conflicting [2,14]. Other reasons include the number of possible solution (e.g. in combinatorial or discrete case), difficulty in formulating or modeling real-world instances, and the computational cost involved in solving these real-world problems. Weise *et al.* [118] attribute the difficulty in solving optimization problems to some fundamental issues encountered during search for solutions. These include premature convergence, ruggedness, causality, deceptiveness, epistasis, robustness, overfitting, over-simplification, and dynamic fitness [118]. The nature of the objective function can also increase the complexity of optimization problems especially for optimization problems with more than one objectives (otherwise known as multi-objectives problems [39,77]). Multi-objective problems are known to be complex due to the conflicting nature of the objectives [39]. It is a general belief that there is no general optimization method that is best or most efficient for all types of problems. The special structure and domain specific characteristics of the problem can often be utilized when choosing a suitable solution method.

At present, no solution method exists that can guarantee global optimal solution of any given problem. Therefore, solution methods are generally referred to as heuristics or meta-heuristics. In this thesis, we consider two practical, real-world problems within the context of a developing country and adapted genetic algorithm meta-heuristic to solve them. The problems are within the domain of space allocation. To the best of our knowledge, one of the studied domains, namely the hostel space allocation problem (HSAP), is new in literature. We further propose some of improve versions of real-coded genetic algorithms (RCGAs) for unconstrained global optimization. These algorithms are tested on a large set of test problems. The rest of this Chapter and Chapter two provide some background review for our study.

1.1 Classification of Optimization Problems

An optimization problem can be thought of as decision problem [114]. As stated earlier, some optimization methods are only appropriate for certain types of problems. It is therefore important to identify the characteristics of a problem in order to apply an appropriate method to it. Brandimarte [23] identified some classification characteristics to include the type of constraints, nature of decision variables, physical structure of the problem, nature of the objective function, permissible value of the decision variables, separability of the functions and number of objective functions. We present a unified classification of optimization problem in Table 1.1. Comprehensive details of the problems can be found in [20,22,42,64,72,79,80,124].

Global optimization problems can be classified based on the properties of the objective function, constraints and the decision variables, the most important being the nature of the objective function. A problem that has no constraint (or bound constraints) is termed an unconstrained global optimization problem. A problem with linear constraints and nonlinear objective function is termed linearly constrained global optimization problem while that with nonlinear constraints and objective function is termed non-linear global optimization problem. The class with only bounded decision variables is known as bound constrained (or unconstrained) global optimization problems. Global optimization problems are also classified based on their inherent nature of the decision variables into either continuous or combinatorial (discrete) global optimization problem. Many real-life problems, for example in applied science and engineering, are formulated as global optimization problems with continuous variables. These problems are often non-smooth, non-convex and often simulation based, making gradient based methods impossible to be used to solve them [70]. They require efficient, reliable and derivative-free global optimization methods.

Table 1.1: Classifications of optimization problems

Characteristics	**Property**	**Classification**
Number of decision variables	One	Univariate, single-objective
	More than one	Multivariate, multi-objective
Number of optima points	One	Unimodal
	More than one	Multimodal
Type of decision variables	Continuous real numbers	Continuous problems
	Integers	Discrete problems
	Both continuous and Integer	Mixed Integer problems
	Integer in permutation	Combinatorial problems
Problem formulation based on existence of constraints	Subject to constraints	Constrained problems
	Not subject to constraint	Unconstrained problems
Objective functions	Linear functions	Linear programming
	Objective function is convex and constraints set form a convex set	Convex programming
	Nonlinear objective or/and constraint functions	Nonlinear/non-convex programming
Nature of the decision variables	Probabilistic	Stochastic problems
Physical structure of the problem	Controlled, dynamic	Optimal control problems

Combinatorial optimization is the process of finding the best solution for problems with discrete set of feasible solutions [96]. Combinatorial optimization problem (COP) is a special class of optimization problems that seeks to find the optimum permutation of decision variables. The solutions are constrained and are usually represented as ordered lists. Combinatorial optimization algorithms solve instances of NP-hard problems by exploring the large solution space of the instances. This problem class finds applications in numerous real-world settings involving operations management and logistics, such as routing, scheduling, packing, inventory and production management, location management, and assignment of scarce resources. According to [96], the economic impact of combinatorial optimization is profound, affecting diverse sectors such as transportation forestry, manufacturing, logistics, aerospace, energy (electrical power, petroleum, and natural gas), telecommunications, biotechnology, financial services, agriculture, and of interest to this thesis, educational sector.

This thesis deals with the application of global optimization methods both in discrete and continuous variable problems.

1.2 Classification of Global Optimization Methods

In this Section, we provide a general overview of some global optimization methods. Figure 1.3 gives a classification of global optimization methods. The classification in Figure 1.3 is inexhaustible. Each class can be further categorized based on other observable characteristics. The list of techniques can also be expanded by appending methods with similar characteristics under appropriate class.

Figure 1.3 divides global optimization methods into the natural division of exact and heuristics (approximate) methods. There are a number of exact methods developed for non-convex global problems with special structures, for example bi-linear and separable problems. An important feature of these methods is that they use convex under-estimator of the non-convex problem. Of these methods, BARON (branch and reduce optimization navigator) [100], αBB (branch and bound) [12] and ECP (extended cutting plane) [119] are widely known. This thesis is concerned with metaheuristic methods in dealing with application of practical interest.

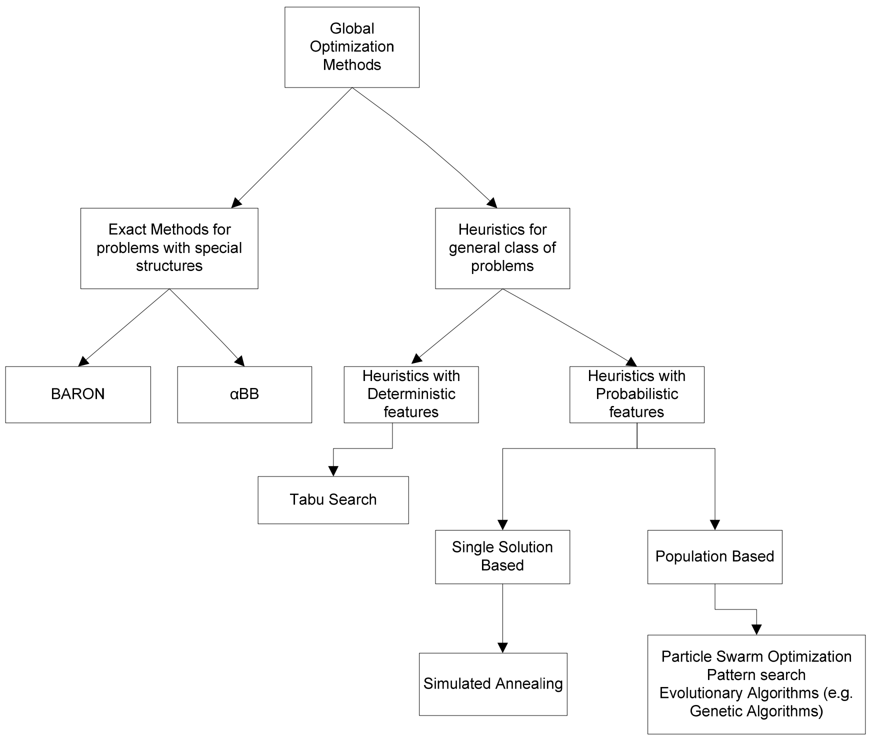
Global Optimization Methods
Exact Methods for problems with special structures
Heuristics for general class of problems
BARON
αBB
Heuristics with Deterministic features
Heuristics with Probabilistic features
Tabu Search
Single Solution Based
Population Based
Simulated Annealing
Particle Swarm Optimization
Pattern search
Evolutionary Algorithms (e.g. Genetic Algorithms)

Figure 1.3: A classification of global optimization methods

1.3 Heuristics and Metaheuristics

A heuristic uses current information gathered during execution to decide the next candidate solution to examine and how it should be processed. The basic concept of heuristic search, as an aid to problem solving, was introduced by Polya [92]. Polya, popularly known as the Father of problem-solving, gave four basic steps that forms the foundation for today's heuristics. These are [92]:

1. understanding the problem (separating the known, unknown and constraints),
2. devising a plan (finding the connection between known and unknown),
3. carrying out the plan (stepwise implementation with correctness proof), and
4. looking back (examining and evaluating solutions obtained).

In algorithmic context, heuristic is a method of performing a minor or a sequence of modifications on a given solution or partial solution in order to obtain a different solution or partial solution [46]. The modification usually involves neighbourhood search. A heuristic therefore helps to create solutions or improve existing solutions by exploring the neighbouring solutions based on certain rules or strategies. A heuristic algorithm iteratively applies one or more heuristics based on given design strategy [46].

Metaheuristic [33,53,59] methods aim to strike a balance between exploration and exploitation during search for optimality. This balance permits the identification of local minima while aiming at the discovery of a globally optimal solution. Exploration ensures a thorough search of the solution space to provide a reliable estimate of the global optimum. Exploitation, on the other hand, concentrates the search effort around the best solution found by searching its neighbourhoods. The exploitation feature helps heuristic methods to obtain the best value for decision variables while the exploration feature makes them well suitable for problems with large search space.

Advocacy for metaheuristics based methods for global optimization problems is recently more pronounced among researchers. While some metaheuristics do not give a guarantee of an exact optimal solution yet the argument is that it is better to have a solution that is little bit inferior to the optimal than one that will require 10^{100} years to be found [117]. This implies a slight compromise in solution quality for computational time and robustness.

Metaheuristics are improvements on heuristics. They are designed to solve more general class of global optimization problems. Metaheuristics include features that may prevent them from pre-mature convergence to local minimizers. They also have search exploratory capabilities. They may include local search procedures for local improvement of solutions. The algorithmic family includes genetic algorithm (GA) [60,67,86], simulated annealing [47,73], tabu search [53,57,58], differential evolution [9,19,94], and pattern search (PS) [45] to mention a few.

This thesis is concerned with the use of GA and some local search heuristics for both discrete and continuous problems of interest. The GA used for our problems were, at some points, augmented by new heuristics we developed. These are reported in papers **[B]-[D].** We also used a modified form of the PS method as an improvement for Real-

Coded Genetic Algorithm (RCGA) for unconstrained global optimization problems. We therefore present a brief overview of GA and PS in this Chapter.

1.4 Genetic Algorithms

GAs belong to the class of population-based metaheuristics that explore a population of individuals randomly sampled over the search space, *S*, based on Darwin's theory of evolution [43] and the principle of survival of the fittest [108]. An objective function, called the fitness function, associates each individual with a fitness value (function value) that reflects its quality. Starting with an initial population, usually generated randomly, GA tries to improve the quality of the individuals by making the population evolve. The evolution is achieved using information exchanges between individuals in order to create new ones or modify the existing ones. The individuals that exchange information are known as 'parents' and the new individuals created (or modified) are referred to as 'children' or 'offspring'. GAs evolve the population using genetic operators such selection, crossover and mutation. GAs therefore are probabilistic algorithms that approximate solutions by maintaining a population of candidate solutions to the problem being solved (Figure 1.4).

Crossover combines elements of solutions in the current generation to create individuals for the successive generations. It consists of exchanging genetic material between two selected single chromosomes. Mutation, on the other hand, systematically changes elements of a solution in the current generation in order to introduce variety into the next generation. Mutation mainly consists of flipping the bit at a randomly chosen point of the chromosome representation of the solution. While the selection operator helps with the exploitation of search space, crossover and mutation accomplish exploration of the search space by creating diversity in the members of the next generation [86]. Common selection operators used in GAs include roulette wheel, stochastic universal sampling, Boltzmann, rank and tournament selection. Crossover operators include the single-point crossover, double-point crossover, preference preserving crossover, and shuffle crossover. Mutation commonly reported includes flipping, interchanging, reversing, replacement, and random replacement. Details of these specific operators can be found in [60,107].

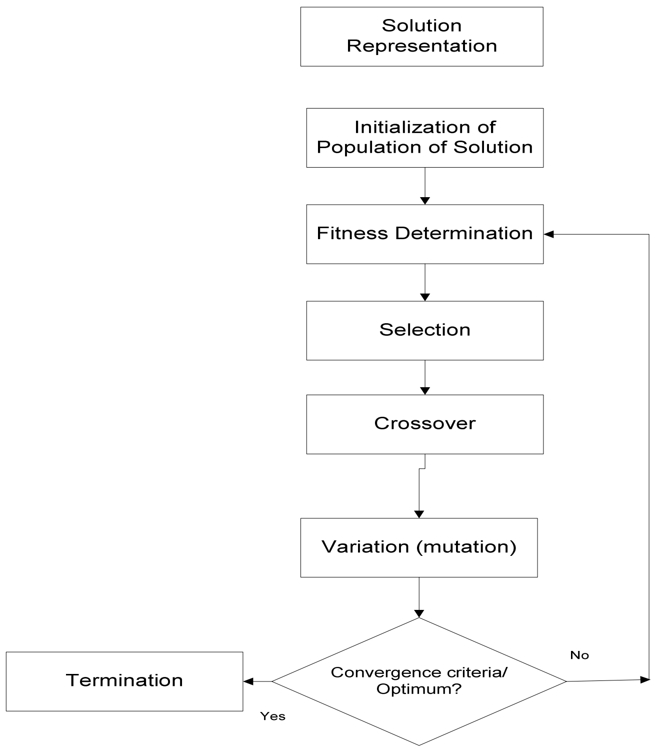

Figure 1.4: Graphical illustration of genetic algorithm steps

GAs are the best known and most successful among the evolutionary algorithms [60,86,107]. This is possibly due to the inherent and unique characteristics that are regarded as the strengths of GAs. These include parallelism, derivative-free nature, ability to explore large solution space, ability to handle complex fitness landscape and deal with multi-objective problems, ability to handle noisy function and escape from local optima and best of all, ability to handle large but poorly understood search space (problem domain) with ease [83,107]. The effectiveness of GAs for hard and complex global optimization problems including real-world instances have been reported in literature. Instances include adaptation to resource allocation problems requiring large scale high performance computing resources [123], complex robotics [11], gene expression and protein folding problems [63,109], transportation, production, logistic planning and routing [127], supply chain scheduling [89], flight scheduling [17], unconstrained global optimization problems [8,71], pattern recognition and data mining [116], to mention a few.

GAs are not without some challenges for users. A great challenge in the application of GAs is the mapping of a problem domain onto the representational structure (chromosome) that will allow for mathematical and computational transformation of the various GA operators on the problem at hand. The choice of the structure depends largely on the nature and complexity of the problem. Binary strings consisting of 0s and 1s is the most commonly used structure. Other possible structures include list, real values, and arrays (of integer or real). Crossover and mutation are performed to keep solutions within the data element boundaries of the structure used while seeking for better solutions. For most data types, specific GA operators can be designed. Different chromosomal data structures seem to work better or worse for different specific problem domains. Another challenge is the determination of the fitness function especially for problems with no known mathematical model or previous domain knowledge. Once the mapping structure and the fitness function are determined, the next challenge is to determine the nature and application strategy of the GA operators that may guarantee 1) convergence to global optimum, 2) escape from local optima, and 3) efficiency of the algorithm in terms of memory space usage and time complexity.

The structure of a typical standard GA is presented below.

Algorithm 1.1: The Standard GA procedure

1. **Initialization.** Generate initial population P_0. Set the crossover and mutation probabilities $p_c \in (0, 1)$ and $p_m \in (0, 1)$, respectively. Set generation counter $k := 1$.
2. **Evaluation**. Evaluate the fitness function f at all chromosomes in P_k
3. **Selection.** Select an intermediate population P_k' from the current population P_k.
4. **Crossover.** Associate a random number from (0, 1) with each chromosome in P_k' and add this chromosome to the parents pool set SP_k if the associated number is less than p_c. Repeat the following Steps 4a and 4b until all parents in SP_k are mated:
    a. Choose two parents p_1 and p_2 from SP_k . Mate p_1 and p_2 to reproduce children c_1 and c_2.

b. Update the children pool set SC_k through $SC_k := SC_k \cup \{c_1, c_2\}$ and update SP_k through $SP_k := SP_k - \{p_1, p_2\}$.

5. **Mutation.** Associate a random number from (0, 1) with each gene in each chromosome in P_k', mutate this gene if the associated number is less than p_m, and add the mutated chromosome only to the children pool set SC_k.
6. **Stopping Conditions.** If stopping conditions are satisfied, then terminate. Otherwise, select the next generation P_{k+1} from $P_k \cup SC_k$. Set SC_k to be empty, set $k := k + 1$, and go to Step 2.

Each iteration of this process is called a generation while the entire set of generations is called a run. It is expected that each run produce one or more highly fit chromosomes in the population.

A lot of refinements such as enforcing diversity [38], self-adaptation of control parameters [48,99], and probabilistic adaptation [71] have been used to extend the applicability of GAs to a large domain of optimization problems. There are significant empirical evidence in literature that GAs converge over time and consistently find good approximate solutions to hard and complex problems [24,62,81,98,104].

1.5 Pattern Search

Pattern search (PS) [45] is a direct search method for local optimization that was initially proposed by Box [21] in the 1950s and later by Hookes and Jeeves [68] in the early 1960s. As a result of more recent strong mathematical proof of its efficiency and convergence [35,112,120], PS is gaining interest among researchers working on optimization problems [49,74,93,110]. PS has been used for parameter estimation in a wide variety of applications and it is popular among optimization researchers because of its simplicity, ease of understanding, ease of implementation and robustness [113]. Furthermore, PS is a derivative-free method that is very useful for optimization problems with either unknown or unreliable function derivatives, or where the function is computed to low accuracy. It serves well as a local optimization algorithm for problems with many known local minima due to its ability to search in multiple directions. This motivates our incorporating PS method as a local search technique into RCGA method to solve benchmark unconstrained global optimization problems (see Chapter 6).

PS is a variation of the coordinate search method [74]. Torczon [112] reported that it belongs to the general class of the direct search methods. It is essentially the characteristics of Hooke and Jeeves method [68] with the basic coordinate search method [74] and the multi-directional search method [111] hence the term, generalized pattern search (GPS) method [35]. Since our interest is to explore a modified form of PS as local optimization heuristic to improve RCGAs for solving unconstrained global optimization problems. We present a brief of PS method.

Basically, PS works by generating a sequence of iterates $\{x(k)\}$ based on the objective function values (without using any information of the derivatives, gradient or second-order derivative). During successive iteration, the objective function is evaluated at a finite number of trial points, taking note of one that yields a lower function value than the current iterate. The point found is set as the new iterate and the iteration is termed successful otherwise the trial points are updated (size of the pattern reduced and function is re-sampled about the current "best" point) and iteration tagged unsuccessful.

Definition 2.1

Let D be the set of positive spanning directions. A positive combination of the set of vectors $D = \{d_i\}_{i=1}^{r}$ is a linear combination $\sum_{i=1}^{r} \lambda_i d_i$, where $\lambda_i \geq 0$, $i = 1, 2, \cdots, r$.

Definition 2.2

A finite set of vectors $D = \{d_i\}_{i=1}^{r}$, $1 \leq r \leq 2n$, forms a positive spanning set for P^n if any $\nu \in P^n$ can be expressed as a positive combination of vectors in D. The set of vectors D is said to positively span P^n. The set D is said to be a positive basis for P^n if no proper subset of D spans P^n. The simplest search directions used by PS method consist of $r = 2n$ vectors and given by the set

$$D = \{e_1,\cdots, e_n,-e_1, \cdots,-e_n\} = \{ d_1, d_2, \cdots ,d_{2n}\}, \tag{1.1}$$

where e_i is the i^{th} unit coordinate vector in P^n. The set D in equation (1.1) is a sample set with maximum positive spanning directions.

The two key components of the PS method are the generating matrix and the exploratory moves algorithms [35,49]. Operations on these two components gives the PS the basic two steps namely, the SEARCH step and the POLL step. The generating matrix represents the set of points that can be sampled at any given iteration k, thus it defines the pattern from which the function is sampled. The exploratory moves algorithm specifies how the sampling should be done. PS method generates a sequence of iterates $\{x^{(1)}, x^{(2)}, \cdots x^{(k)}, \cdots \}$ with non-increasing objective function values. Each iteration k, goes through the two steps of SEARCH and POLL respectively. We now give a more formal description of PS method with an assumption that $r = 2n$.

In the SEARCH step, the objective function is evaluated at a finite number of points (say a maximum of V points) on a mesh (a discrete subset of P^n) so as to improve the current iterate. The mesh at the current iterate, $x^{(k)}$, is given by

$$M_k = \{m \in R^n \mid m = x^{(k)} + \Delta_k Dq : q \in Z_+^r \}, \tag{1.2}$$

where m is a mesh trial point, $\Delta_k > 0$ is a mesh size parameter (or step size control parameter) which depends on the iteration k, and Z_+ is the set of nonnegative integers. The generation of the trial points for SEARCH step in the current mesh is largely user-depended and can be done using some heuristic rules. This step finds a feasible trial point, $m \in M_k$, (where m is one of the V points) with a smaller objective function value than the value at $x^{(k)}$, that is, $f(m) < f(x^{(k)})$. If m is found, it is updated as the new iterate and the step size Δ_k is increased in order to choose the next trial points on the now larger mesh. If m is not found, then the SEARCH step is unsuccessful for the current iterate, the POLL step is then executed around $x^{(k)}$ to decrease the objective function value. This step must be done before terminating the iteration.

The POLL step samples the function about the current iterate $x^{(k)}$ in a deterministic fashion to generate trial points which produce a new and better iterate (one that minimizes the objective function). This produce a poll set, P_k, with trial points that are positioned a step Δ_k away from the current iterate $x^{(k)}$ in the direction designated by the columns of D. P_k, can thus be represented as:

$$P_k = \{ \mathrm{p}_i \in R^n \mid p_i = x^{(k)} + \Delta_k d_i : d_i \in D, i = 1, \ldots, r \}, \tag{1.3}$$

where p_i is a trial point in the POLL step. Note that the order of evaluation of P_k does not matter nor affect the convergence of the algorithm. With the two steps defined, we present the complete PS algorithm as given in [6].

Algorithm 2.1: Standard PS algorithm (based on the SEARCH and POLL steps)

1. **Initialization.** Choose an initial point $x^{(0)} \in \Omega$ and an initial mesh size $\Delta_0 > 0$ Set the iteration counter $k := 0$,
2. **SEARCH**. Evaluate the fitness function f at a finite number of points in the mesh P_k as defined in equation (1.2). Then,
    a. **If** $f(m) < f(x^{(k)})$ for some $m \in M_k$, **then** set $x^{(k+1)} = m$, tag_{SEARCH} = **SUCCESSFUL**; go to step **4**.
    b. **Otherwise (**i.e. $f(m) \geq f(x^{(k)})$ for all V points), to step **3**.
3. **POLL.** Follow the steps
    a. **If** $f(p_i) < f(x^{(k)})$ for some $p_i \in P_k$ defined by equation (1.3), **then** set $x^{(k+1)} = p_i$; go to step 4; tag_{POLL} = **SUCCESSFUL**
    b. **Otherwise (**i.e $f(x^{(k)}) \leq f(p_i)$ for all $p_i \in P_k$ defined by equation (1.3) ,**then** set $x^{(k+1)} = x^{(k)}$ and go to step 5 tag_{POLL} = **UNSUCCESSFUL**
4. **Mesh Expansion.** Let $\Delta_{k+1} = \phi_k \Delta_k$, with $\phi_k > 1$. Set $k = k+1$; Go to step 2

5. **Mesh Reduction.** Let $\Delta_{k+1} = \phi_k \Delta_k$, with $0 < \phi_k < 1$. Set $k = k+1$; Go to step 2

Algorithm 2.1 represents a typical procedure for PS method consisting of both the SEARCH and the POLL steps. Implementation steps may however differ depending on the problems to solve and objective to achieve. Literature has reported implementation that uses only the POLL step [6,74]. Further details on some modified and improved PS methods with applications to optimization problems can be found in [6,49,93,111].

1.6 Problem Statements

In this thesis, we consider problems from the two broad classes of global optimization, that is, combinatorial optimization and unconstrained global optimization. We selected two practical real-world problem instances from the class of space allocation problems (SAPs) which have recently attracted attention among metaheuristics researchers. Much work has been done with regards to instances of SAPs some areas such as office space allocation, timetabling and shelf space allocation. However, most works even in these problem instances used cases from developed countries. In addition, there has not been any reported work on metaheuristics for HSAP, which is fast becoming a major administrative concern for management in tertiary institutions especially in developing countries. This, alongside obvious needs that arose in our case study, motivated the research into metaheuristic application to the two real-world instances of COPs, namely the university timetabling problem (UTTP) and HSAP. Since HSAP is new in literature to the best of our knowledge, we designed some basic heuristics and GA metaheuristic to solve the problem at different stages. One of the major objectives of this thesis is to show the applicability of heuristics and metaheuristics to the new domain of HSAP especially within the context of the case study considered.

Furthermore, we developed some modified RCGAs for finding the global minimum of some unconstrained global optimization problems. We ran simulation experiments based on standard RCGA (SRGA) and variants of modified RCGA. The results of the modified RCGAs based on PS and vector projection methods are compared with that of SRCGA.

The numerical efficiency and robustness of the methods were tested with fifty seven (57) bounded global optimization test problems (see paper **[D]** in Chapter 6).

The thesis is a further attempt to show the robustness and efficiency of GAs in handling real-life global optimization problems.

1.7 Structure of the Thesis

The thesis consists of two parts: I) introduction and background study; and II) reports on scientific research. Part I covers Chapters 1 and 2. Chapter 1 gives a general overview of the background area of global optimization problems and methods. An overview on global optimization methods with emphasis on GAs and local search PS is presented. Chapter 2 presents some backgrounds on SAPs which form a main focus of this thesis. In Part II we concentrate on the work done in the papers **[A]** to **[D]**. Chapter 3 presents paper **[A],** which describes a multi-level GA that forms the baseline study for the HSAP. New heuristics were developed for the first two identified stages of the HSAPs and results of the various implementation options compared. These are discussed in paper **[B]** as presented in Chapter 4. Chapter 5 discusses a GA-based metaheuristic solution to the UTTP as presented in paper **[C]**. Chapter 6 presents paper **[D]** on some modified RCGAs for unconstrained global optimization problems. A summary of the entire work with some drawn conclusions and further research directions are highlighted in Chapter 7. Some statements of the contributions of this thesis are provided in Chapter 8.

Chapter Two

Space Allocation Problems:

Introduction and Related Works

“If we can really understand the problem, the answer will come out of it, because the answer is not separate from the problem”

- Krishnamurti

2.0 Introduction

Space planning, distribution and optimization are important managerial responsibilities that have great effects on institutions and organizations. Mismanagement, over-utilization or under-utilization of space can affect the overall ability of an institution to meet its target goals and objectives. For instance, shelf space in the supermarket continually filled with items that are out of demands at the expense of much demanded goods will negatively affect the profitability and functionality of the supermarket. Space is therefore an important asset that must be well managed in order to achieve stated goals and objectives. In real-world instances, a common reality is the limited availability of space compared with the competitive space demanding entities. We refer to this as *scarcity* of space. An obvious example is the scarcity of housing (dwelling space) in comparison with the rate of population growth in most developed communities. Expansion in business, increase in demands, staff strengths, goods and services as well as competition among service providers without corresponding increase in space provision (office, shelf, accommodation, etc.) make space planning and optimization a challenging problem for researchers. The functionality of some institutions therefore depends on the ability to efficiently manage and distribute available but limited space.

SAPs are those in which the capacity of limited space available has to be distributed among a set of items while observing some specific requirements and constraints. The requirements and constraints are sometimes complex and conflicting. SAPs have some close similarity to scheduling problems and are NP-Hard in nature [14,125]. Wren [122] defined scheduling as arrangement of objects into a pattern of time or space in such a way that the goals are achieved or nearly achieved, and the constraints of the objects are satisfied or nearly satisfied. This is the goal of SAP as an optimization problem. A good space distribution must ensure that all demanding entities are given the minimal required space as much as possible and space utilization efficiently meets stated domain-specific goals, objectives and constraints. Space overuse by any entity must be prevented while space wastage is reduced to the barest minimum possible.

Space allocation in academic institution is a complex, difficult and time consuming task, often carried out manually or semi-automatically by the officers involved. If we consider SAP as a capacity allocation problem, then it has similarities with the classical knapsack and bin-packing problems [77,84]. The application of heuristics to tackle domain-specific SAP was recently suggested and studied [25]. Subsequently, a lot of studies were done on the application of heuristics to instances of SAP. Among these are office SAP [77], lecture room allocation (otherwise referred to as timetabling problems) [26,41], and shelf SAP [14,15,125]. A domain that has not been studied is the hostel space allocation problem (HSAP) especially with reference to the recent increased demand for on-campus accommodation in tertiary institutions as a result of increase in admissions. We consider an instance of this new case (HSAP) in this thesis. In physics, time is considered as the fourth dimension of space. Similarly, if we consider time factor in lecture room allocation to courses, then timetabling problem becomes essentially a space allocation (distribution) problem. Lecture timetabling problem can therefore be treated as a course SAP [34]. This motivated part of our study on an instance of the university lecture timetabling problem in the context of a developing country where metaheuristics have not been explored to solve the problem before.

2.1 Space Allocation Problem – An Overview

Space planning hinges on the efficiency of resource usage and its impacts on institutions such as companies, organizations, housing, and education. Practical problems involving space allocation include disk storage space allocation in computer science, room allocation among staff, lecture room allocation to courses (lectures), and so on. The dynamic nature of these institutions makes space planning process a regular and repeated one that requires efficient techniques for carrying it out. The limited availability of space makes it necessary to evolve an efficient distribution strategy for efficiency which can only be guaranteed when all demanding entities are given the minimum required space while observing, to a large extent, given constraints and/or requirements.

SAP, in a higher institution context, can be defined as the allocation of various entities (for example, staff, students, laboratory, lectures) to areas of space (for example, rooms, bed space) in such a way that satisfies stated requirements and constraints. Generally, allocating rooms in the university environment is a multi-stage process [77]. This class of problem is highly constrained with multiple objectives that vary among institutions, and requires frequent modifications to accommodate the addition or removal of entities and/or rooms [77]. Other characteristics of SAPs are huge search space that increases with the size of the problem instance, difficulty in finding a suitable representation that can capture the complete system constraints; and the determination and computation of an adequate fitness function for the problem instance [29,34]. The automated scheduling, optimization and planning research group of the University of Nottingham, UK, listed and grouped possible constraints and requirements for SAP into about eleven some of which are presented in [77]. However, constraints and requirements generally depend on domain specific problem under consideration and the environment.

The increasing demand for university graduates with the attendant increase in admission rate and the trend towards electronic-based learning environment bring about the need for more flexibility not only in learning but also in management and organization structures in higher education institutions of the twenty-first century. Shabha [102,103] submitted that this trend will impact on space management as there will be a shift emphasis towards a more time-flexible, space-flexible and location-flexible space planning in higher institution. The insufficiency of existing campus buildings and inadequacy of their accommodation units to cater for the increased students intake particularly in government-owned universities have been pointed out in [82,103]. This problem is compounded by the financial constraint and complex organizational framework experienced by most institutions especially in the third world countries [106]. Shabha [103] further submitted that the relationship between space and service distribution is the most significant factor which contributes to sustainable functionality in most specialized building such as hospitals. This explains why space management must be well-planned and structured in order to cater for the peculiarity of different categories of entities requiring space. For example, it will not be an efficient distribution to locate people with disability at the topmost floor in a high rise building far away from where they can have easy access to health care and attention. SAP therefore has direct impact on the functionality of institutions especially tertiary institutions which is the focus of this thesis.

Optimization of space allocation is a complex, multivariate problem [51]. The complexity is introduced by the nature of some requirements and constraints which on a broad sense might include technical space requirements, operational costs of available space, resource requirements such as utilities and networking, compliance with space guidelines and requirements, and so on. Despite this complexity, the task of space allocation is done manually in most cases especially in developing countries with some form of reliance on database or spreadsheet driven applications for record maintenance [28]. The need to incorporate good algorithms to determine an optimal allocation of spaces is therefore inevitable [2,28].

SAPs have been classified into either reorganization of the existing allocations or construction of completely new solutions [51]. The main differences lie in the objectives and requirements of the problem. Reorganization of the existing allocation is the re-arrangement of a current space distribution among various entities in order to improve the existing solution under existing conditions or modify the allocation because of changes in requirements or constraints. Construction of a complete allocation is the generation of a new solution from scratch to distribute available space among all eligible entities based on given requirements and constraints. HSAP, as a new instance of SAP, falls into the second category as we seek to construct complete distribution of university hostel space among eligible students while observing given requirements and constraints. The main objectives for a re-organization process might include minimizing the cost of relocation of entities and the distance between related entities.

The SAP can therefore be viewed as a problem of distributing the available space among the demanding entities in such a way that the space utilization is optimized [77**]**. An important condition that applies to most SAPs such as bin packing problem, knapsack problem, academic resource allocation and others is that the available space and events are fixed and are not subject to modification. Knapsack problem for example has a number of items of given sizes and a number of knapsacks of given capacities. Each item has associated profit and weight assigned. The objective is to fill each knapsack with a subset of the items without exceeding the capacity of the knapsack and at the same time maximizing the total profit [84]. We give a brief overview of the two instances of SAPs considered in the next subsections.

2.1.1 University Timetabling Problem

The timetabling problem (TTP) is a special class of NP-Hard problem that abounds in many real life situations especially in educational institutions. It takes a lot of man-hour effort to generate an acceptable timetable manually and yet the search for optimal solution to the problem is still on. Most manually generated timetables are often subjects to regular revision as they do not meet all domain-specific requirements. A change in the requirements or preconditions renders the whole process unusable and a new process has to be restarted. Even when the problem is reduced through relaxation of some requirements, it is still extremely complex to find the optimal solution. This accounts for the trend in heuristics or metaheuristics application to solve TTP. Part of this thesis is a pioneer work to advocate the use of metaheuristics for UTTP in Nigerian universities. GAs has proved very useful in search of solution to similar problem within other domain instances (see [5,7,27,54,61,75,91,101]). Hence, we experimented with the same metaheuristic[1] in our study.

TTP in education institutions naturally divide into two namely, the lecture (course) timetabling problems (LTTP) and the examination timetabling problems. An essential difference lies in the rigidity of the constraints and requirements of the problems. Since the focus of our study is on the LTTP, all further reference to university timetabling problem (UTTP) will be taken to imply LTTP.

[1] While some authors refer to GA as a heuristic, we considered it a metaheuristic and employed adapted version of the same to solve our problems in this thesis. We however designed other heuristics which are incorporated to improve the quality of solutions obtained in some of our problems.

UTTP is an NP-Hard problem [122] with level of difficulties varying from institution to institution depending on space availability and specified constraints and/or requirements. In our study, lecture timetabling is considered in a bottom-up fashion, starting from the faculty/department level to the university level. Courses are designated as 1) departmental course – offered majorly (almost exclusively) by registered students in the department, 2) faculty courses, - offered to students across certain discipline within a faculty, and 3) university courses – general courses offered by students across more than one faculty simultaneously. There is a university central timetabling committee in charge of timetabling at the topmost level. Classrooms of various capacities are built around each faculty with few dedicated classes for some departments. To cater for the university-wide courses, there are large lecture halls that are controlled by the central committee. This arrangement makes it easy to adopt a bottom-up approach to timetable generation where each department/faculty can allocate lectures to classrooms they control exclusively.

Constraints that affect timetable schedule can be classified into hard and soft constraints. Hard constraints are conditions that cannot be violated if feasible solutions are to be ascertained. Soft constraints, on the other hands, might be slacked with some penalty if the system cannot fully satisfy them. Two major constraints that influence classroom allocation to courses are the classroom capacity and the class size. A major characteristic of a good lecture timetabling heuristic is the ability to resolve conflicts that arise naturally during timetabling generation. This is a sort of sharing restrictions that prevents two lectures being assigned to the same room simultaneously. Other similar restrictions include allocation of two or more compulsory courses offered by the same students within the same timeslot and allocation of two or more courses taking by the same lecturer within the same timeslot. Therefore, no entity (students or courses) must be allocated to more than one location at any given time. Moreover, for each period or timeslot, there should be sufficient resources available for all scheduled events. Depending on environment, promixity - ensuring that lectures are allocated close to the department (or students) offering it - can be considered a soft constraint. Other might include preferential treatment – allocating some lectures to desired classes or period, and reduction of space wastage and overuse.

LTTP can therefore be defined as a SAP involving the distribution of available classroom space with different capacities and specifications, among sets of timetable events having different requirements and sizes, without violating any mandatory condition but satisfying as many other requirements and/or constraints as possible in order to ensure optimum space utilization. Chapter 5, where paper **[C]** is presented, gives more overview of problem instance and our work in this area.

2.1.2 Hostel Space Allocation Problems

Hostel space allocation is becoming a big concern for universities administration in developing countries where hostel facilities are provided for students. This concern stems from many, and often conflicting, factors and objectives to be achieved. A major issue is the increased rate of admission and the attendant requests for campus residence. One great concern is the decrease in capital fund allocation to tertiary education especially in developing country which makes it difficult to consider capital project expansion including hostel facility for students. Some institutions have to depend on possible donations from external bodies which is either highly uncertain and grossly insufficient to meet their growing needs. Demands for increased funding have led to many strike actions by university staff in recent years which subsequently disrupted academic activities and plans [4,95]. The increasing population of students thus poses a challenge of finding an optimal design strategy for accommodating changes especially with regards to space requirements and provision. While it might seem easy to predict the short-term space requirements based on past admission statistics, it is becoming difficult to predict the long-term space requirements due to uncertainty and future admission rate. The pressure is much on the few available tertiary institutions in developing countries to admit the ever-growing population of admission seekers. There is therefore the need to efficiently manage existing hostel facilities among eligible students while not compromising the set goals, objectives and standards of the institutions. Ideas relating to effective utilization and better deployment of existing hostel facilities are therefore of prime importance to university authorities especially in developing countries.

We define HSAP as the problem involving the allocation of *scarce* bed space resources within hostels among many competitive ‘customers’ (eligible students) under given hard and soft constraints. The application of well-known heuristics to this instance of SAP has not been reported in literature. To the best of our knowledge, our work is the first attempt at employing heuristics to handle instance of HSAP as defined in our context especially within the ambit of our case study. Our work thus forms the baseline for studies into HSAP for students in tertiary institutions.

2.1.2.1 Problem definition

HSAP refers to the distribution of the available bed spaces in halls of residence (hostels) among a number of categories of students with different sizes and conditions so as to ensure the optimal space utilization and the satisfaction of additional requirements and/or constraints.

Our work is based on instances as obtained at the University of Lagos in Nigeria. The university currently has a present combined student population of over 39,000 with halls of residence in the main campus built to accommodate both undergraduate and postgraduate students. Our concentration is on the undergraduate students who form the majority of the student population. As at the time of study, hostels in the main campus consist of twelve undergraduate halls, six for males and six for females. The halls are built and grouped into zones based on their physical location (see papers **[A]** and **[B]**). Hostel space allocation is done just before the beginning of each session by the Students’ Affair Office assisted by appointed hall managers. The stages involve include:

1. **Application and Submission –** interested students collect, fill and submit application forms.
2. **Data entry –** The accommodation office at the Students’ Affairs Office enters necessary data from received applications into the system.

3. **Validation –** Applications are validated (by manual cross-checking) to determine eligibility of students. Applicants are then categorized into different category as shown below.
4. **Allocation generation –** Hall lists are generated, released and distributed for further allocation to the hall managers.

Like other university SAP, HSAP is thus a multi-stage process. We summarized the above processes into the follow:

1. Compilation of applicants' list by the Students' Affair Office.
2. Categorization of students into various categories. Determination of number of students to allocate under each category.
3. Allocation of part or all students in each category into various hall based on certain requirements and priorities.
4. Allocation of students in each category to various floors/blocks within the each hall of residence.

In our work, we decomposed the whole process into three stages namely, category allocation, hall allocation, and block/floor allocation (see Figure 1 of paper **[B]**). We identified the requirements and constraints for each of these stages. GA-based heuristics are designed to handle each of these stages of allocation. The main objective of our work is to investigate the viability of heuristic application the case instances considered with the aim of helping to improve the efficiency and utilization of the limited physical space resources.

There are eight categories used for allocation purpose namely, final year (Fy), scholars (Sc), foreign (Fo), physically challenged (Ht), fresher (otherwise called the first year) (Fr), sports men and women (Sp), discretionary list (Ds) and others (Ot). Priority orders are also assigned to these categories for allocation purpose at some stages. To be categorized as Ht student, the applicant must be registered at the university health centre. Discretionary students are usually based on individual request made by senior member of staff in the university. Detail on the category and allocation are contained in paper **[A]** and **[B]**.

Some of the requirements and conditions identified for the HSAP include:

- Capacity constraint must not be violated. For example, number of students allocated to a hall must not exceed the capacity of the hall.
- Fixed allocation - allocations of certain categories of students must be to specified halls. For example, Ht students must be allocated to designated halls that are close to health care facility for easy access. Sp students must be accommodated in the same hall very close to the sports centre. Sc students also have designated halls.
- Compulsory allocation – all applicants within certain categories must be accommodated for some administrative considerations. This affects Fo, Sp and Ht categories in our case.
- As many of Fy,Sc, Fr, Ds and Ot students as possible should be accommodated in prioritized order as listed, Fy having the highest priority.
- If possible, allocation should be such that students from the same department are located close to each other. This was introduced when there was a security problem on campus but had since been relaxed. Hence, we did not consider this.
- Ht students should be allocated to the lowest possible floor in their designated halls – for conveniences.
- Fy students should be allocated to the highest possible floor in their designated halls, possibly for concentration and avoidance of distractions.

We classified these requirements/constraints into either hard or soft constraints for the purpose of our study (see Chapters 5&6). Where necessary, some of these constraints were assigned appropriate weights for computational experiment purposes. The quality of a solution (allocation) is measured in terms of the following:

- the number of students allocated under each category
- satisfaction or no violation of hard constraints.

- space utilisation, i.e the amount of space that is wasted (space not used) and the amount of space that is overused (categories with less space allocated than needed).
- satisfaction of any soft requirements/constraints.

An optimal solution for SAP is one where all the entities are allocated, no space is wasted or overused and every additional requirements and constraints have been satisfied. In most cases of NP-hard problems, this is not always achievable with heuristic allocation. A more realistic optimal solution for SAP will be one in which all entities are allocated and the space utilization is the best possible, i.e. the amount of space wasted and overused has been reduced to the minimum and the additional requirements and constraints have been all satisfied. To minimize the penalties in a solution for a SAP, no hard constraints should be violated and as many as possible soft constraints should be satisfied **[**14**]**.

GAs have shown proven performance in initial studies of similar problems for which the search space is large or not fully understood; domain knowledge is scarce and expert knowledge is difficult to encode; no mathematical model or analysis is available; and where benchmarking standard is unavailable [60,83,86]. Similar baseline studies employed GAs due to robustness and efficiency of the algorithm [13,44]. We designed a GA data structure for representing above problem at the hall and floor levels and employ various heuristics to handle different levels of allocations. Simulation experiments were conducted to determine the best algorithms combinations and/or GA parameters that give the best solution for hostel space distribution. Promising results are reported in papers [**B**] and **[C]**.

2.2 Modeling the HSAP

We present in this section, the mathematical models for the description of the HSAP. We strive to present a generic view of the problems such as can be easily adapted to any case instance. As pointed out earlier, HSAP is a multi-stage problem. For the purpose of modeling, we identify three stages of allocations namely, the category, hall, and floor allocations. However, our model is limited to only the first two stages of allocations. The models are based on some modified forms of the bounded knapsack problems [84]. This is done essentially since the problem involves placing some items (students) into available knapsack (hostel space) in order to satisfy certain constraints and requirements. Comprehensive details on knapsack problems and its various forms is provided in [84].

We present the discussion for each of these in turn below.

2.2.1 Category Allocation

The allocation at this stage depends upon the priorities set by the administrators (Students Affair's office) and the total capacities of the available halls. As later discussed in Chapter four, the categories are divided into fixed-choice and flexible-choice allocation. We assume that individuals are selected into a single knapsack (described by the total capacities of all the halls), depending on their level of priority and some assigned weights. We then model this stage as a modified form of bounded knapsack problem. Since it is not feasible to allocate all applicants in each flexible-choice category, the problem can therefore not be modified as a binary knapsack problem. The allocation at this stage is therefore done subject to the following restrictions and further assumptions:

- The two broad categories of fixed and flexible must be handled separately with the latter given the first priority. Weight in the range of [0,1] are assigned accordingly to these two categories. We assign a weight of 1 to all fixed-choice while flexible categories are assigned variable values in [0,1]. However, since all categories must be granted the minimal allocation possible, we ensure that no category is assigned a weight of 0.
- A cost function is introduced for the model. However, for the flexible category, this function is designed to follow the order of allocation priority of the categories involved.

Let T represents the total capacity for all the halls (the size of the knapsack) and p_i represents the cost of allocating a category i to T, and w_i represents the number of applicants in category i (equivalent to the weight of each items for the knapsack), $i = 1,\ldots,m$; where m is the number of categories. Let k represents the number of categories under fixed-choice. We assume that the categories are ordered such that the fixed-choice comes before the flexible choice (since they are given first priority in allocation). Therefore, the number of flexible choice categories is m-k.

Next we define T_F and T_V as the total number of applicants in the fixed-choice and flexible- choice categories respectively. That is,

$$T_F = \sum_{i=1}^{k} w_i \qquad (1.1)$$

Therefore

$$T_V = T - T_F$$

Furthermore, we assume that the priority of a given category in the flexible-choice increases with the number of applicants in the category. We therefore defined the cost function as:

$$p_i = \begin{cases} 1 & \text{for } i = 1,\ldots.k \quad \text{(fixed)} \\ \dfrac{w_i}{T_V} & \text{for } i = k+1,\ldots,m \quad \text{(flexible)} \end{cases} \qquad (1.2)$$

From equation 1.2, it is obvious that the category with higher priority will have higher cost function value assigned than those of lower priorities thus enforcing the priority requirement of the allocation process.

Put together, the model for the category allocation stage becomes:

Maximize $\sum_{i=1}^{m} p_i x_i$

Subject to $\sum_{i=1}^{m} p_i w_i x_i \le T,$ (1.3)

where

$$x_i = \begin{cases} 1 & \text{for } i = 1,....k \quad \text{(fixed)} \\ 0 \le x_i \le 1 & \text{for } i = k+1,...,m \quad \text{(flexible)} \end{cases}$$

2.2.1 Hall Allocation

At this stage, allocation is done into respective halls based on certain constraints (weights) on some categories of students. At this stage, we are essentially assigning students into the available hall. Similar to the first stage, we also have the fixed and the flexible groups. We seek to allocate students such that certain categories of students (fixed) must be allocated to designated halls while others (flexible) are distributed to remaining space in all the halls in order to maximize the distribution spread of each category. Note that after the first stage, the overall total number of applicants is equivalent to the total number of available bed space in the halls. Other assumptions follow as in the category allocation stage.

We define a variable, hall ratio, r_j, as

$$r_j = \frac{h_j}{T_V} \quad (1.4)$$

h_j is the capacity of hall j, $j = 1,\ldots,n$, where n is the total number of halls. The hall ratio is used essentially to enforce the distribution spread of students in flexible group across all the available halls.

Let p_{ij} be the cost of allocating student in category i to hall j and w_{ij} be the number of students in category i allocated to hall j (this represents the weight of class i for hall j). We then seek to

Maximize $\sum_{j=1}^{n}\sum_{i=1}^{m} p_{ij} x_{ij}$

subject to

$$x_{ij} = \begin{cases} w_{ij} & \text{for } i = 1,\dots,k \text{ and } j \text{ is fixed (constant)} \\ 0 \le x_{ij} \le w_{ij} & \text{for } i = k+1,\dots,m;\ j = 1,\dots,n \text{ (flexible)} \end{cases} \qquad (1.5)$$

and

$$\sum_{i=1}^{m} x_{ij} = h_j \qquad \text{for each } j,\ j = 1,\dots,n \text{ (hall is filled up)}$$

Next, we need to determine the cost function, p_{ij}. Since the allocation of the number of students depends on the number of applicants and the hall ratio, we cannot allocate more than the expected portion of a given category to a given hall. Therefore the cost function is formulated as follows:

$$p_{ij} = \begin{cases} 1 & \text{for } i = 1,\dots,k \text{ and } j = \text{fixed (specified hall number)} \\ 0 \le p_{ij} \le r_j & \text{for } i = k+1,\dots,m;\ j = 1,\dots,n \text{ (flexible)} \end{cases} \qquad (1.6)$$

2.3 Related Works

Many practical, real-world instances of SAPs have been studied in literature. It is interesting to note that most instances arose from challenges facing one institutions or the other just as in our study. For example, the automated scheduling, optimization and planning group of the University of Nottingham was formed to find automated solutions to practical SAPs for different institutions in the United Kingdom [25,29]. With much successes recorded on baseline heuristic applications, the group later extended their work to higher level heuristics for other instances of SAP (for example, see [15,25,28,29,30,31,32,76]). Michalewicz and Fogel [85] submitted that in practical setting, the use of heuristics have proved to be often superior to exact methods. This accounts for their preference in handling real-world problems.

Before the application of heuristics and metaheuristics, several attempts were made to use exact methods to solve smaller instances of SAPs. Early studies on space planning and utilization in university environment include [16,65,88,97,105]. Most early studies however focused more on capacity-related issue, that is, "how much space is required to deliver the educational programs of the university or college?" However, location-related issues, that is, “where to place an entity” is of more relevance in space planning and management. In HSAP, the question of “how much space?” is naturally handled from capacity constraints and the number of applicants. HSAP therefore seek more to address the issues of “who to allocate and where to allocate them”, the solution of which affects the overall goal of the allocation process and the university in general. Part of early attempt to address the “where” issue was done by Sharma and Kurma [106] who studied the problem of space allocation to academic departments in a high rise building of an Australian educational institution. Two main objectives of the study were to minimize student pedestrian movement within the building and to maximize intra-departmental interaction. A cost-minimization model was used to solve the problem as a transportation problem. The resulting assignment of space was found to be better than the existing deployment of teaching department accommodation in terms of objective satisfaction. The study is however for a small instance/data set with inability to handle other multi-objectives that arose from the given instance.

Ritzman et al. [97] formulated a mixed-integer goal programming model to study the planning of academic facilities involving the reassignment of 144 offices to 289 members in 6 academic departments within the Ohio State University. The objective of the study was to make the reassignment of offices as fair as possible while avoiding conflicts such as minimizing the distances between the rooms assigned to each department and its administrative office, and ensure that each department obtains a fair share of the available high quality offices. The study however revealed that the mixed integer goal programming model was rather too complex for the problem than a standard Linear Programming. Benjamin et al. [18] employed linear goal problem to study the multi-objectives allocation of 15 sections to a new computer integrated manufacturing laboratory at the University of Missouri-Rolla. The objectives include developing new courses relying on the laboratory facilities, increase the students' use of the laboratory facilities and stimulate the graduate-level and funded research. The goals were prioritized using the analytic hierarchy process, a multi-objective decision making technique [18] which rank the alternatives of problems in hierarchical structure using pair wise comparison. The basic assumption was that the objectives of a problem can be represented in a hierarchical structure. The priority structure was incorporated into linear goal programming model that determines the optimum resource allocation. Results obtained were measured by the ability to fulfilled stated objectives as no comparison was made to other methodology. Giannikos et al. [55] studied the reorganization of academic space distribution in six major sites at the University of Westminster using integer goal programming. The main objective is to assigning enough and adequate type of offices to each school while avoiding repeated allocation of the same entity to different offices. Other objectives were minimizing the distance between offices assigned to a school and its administrative centre and minimizing the number of people that have to be relocated. The objectives were ranked according to their importance hence the use of pre-emptive goal programming to obtain a satisfactory solution.

The use of heuristics or metaheuristic to solve real-world instances of SAPs was popularized by the automated scheduling, optimization and planning group of the University of Nottingham then led by Burke [25]. The group has maintained a focused effort since 1998 to address the space allocation problem in the context of academic institutions. Specially, office space allocation [77] and on-the-shelf space allocation [14] were among those researched by the group. Their initial work on space allocation for higher institutions was based on genetic algorithm using data obtained from higher institutions in the United Kingdom. Subsequent works expanded to the use of other heuristics and their variants for different instances of SAPs. One of such employed hill climbing (HC), SA and GA methodologies to automatically generate solutions to the SAP [25]. The HC was applied in two ways: random selection of rooms (also called as random fit) and selection of room with the lowest penalty (best fit). The GA used roulette wheel method in the selection process. The GA was tested with various population sizes and various initial populations. It was tested with the random fit HC (random selection of rooms), best fit HC (selection of room with the lowest penalty) and SA initialized population. Results showed that SA performed the best though with longer convergence while random fit HC performed the worst which has faster convergence. Subsequent work after this have employed several variants of HC, SA and GA to handle the SAP [30,31,32]. SA and HC variants showed great performances when it comes to reorganizing allocation problem. This is likely because most conflicting resources were already allocated hence these local search heuristics serves to improve existing allocation. Based on the instance studied and data set used, the results of GA were shown to be better when improved with local search heuristics. Most of the works used domain-specific instances obtained from institutions in the United Kingdom.

Furthermore, Burke and Newall [27] presented a multi-stage evolutionary algorithm for the timetabling problem. The multi-stage algorithm decomposes a larger problem into smaller components which can be effectively handled by evolutionary algorithm. The algorithm was able to fix the events in the timetable before considering the next subset of events. This approach produced faster and better quality solutions to series of sub-problems than would have been if the larger problem is handled as an entity. Alkan and Ozcan [10] developed a steady state GA to find solution to a small portion of real-world course timetable data obtained from the Faculty of Engineering and Architecture (FEA), Yeditepe University, Istanbul in Turkey. This was a pioneer study into the instance and case considered and GA was found suitable for such with promising results. The study however did not make any distinction between hard and soft constraints. However, initial experimental results obtained in these work showed the viability of applying metaheuristics which eventually prompted further studies by the researchers. In one of such subsequent studies, Alkan and Ozcan [91] employed a variety of operators applied to memetic algorithm in search of solution to the same data set. Operators used include violation directed mutations, crossovers and violation directed hierarchical HC method. Initialization was done randomly and the population passed through the HC heuristic. A random, low probability mutation was applied. An additional mutation was also used to guide the search while appropriate penalty values were computed by a factor. The algorithm employed the one point crossover and uniform crossover as well as a new crossover selection based on ranking strategy. Results obtained favoured the use of genetic search combined with HC heuristic.

Chapter Three

Paper [A]: A Multi-level Genetic Algorithm for a Multi-stage Space Allocation Problem

“That some achieve great success, is proof to all that others can achieve it as well.”
– Abraham Lincoln

"There is no one giant step that does it. It's a lot of little steps."
- Peter A. Cohen

“You may never know what results come of your action, but if you do nothing there will be no result”
- Mahatma Gandhi

Necessity is the mother of invention
- Plato

Space management can be carried out more efficiently when the building design process has been thoroughly planned. Paper **[A]** present the initial work on the multi-level application of GA to the multi-stage HSAP. The paper presents some results obtained from simulation experiments based on dataset obtained from the University of Lagos in Nigeria. This work was motivated by the need to overcome some obvious bottlenecks in the manual approach adopted by the institution. There was a need for an effective and efficient means of allocating hostel accommodation to students especially on the main campus which has the higher concentration of student population. Some of the problems with the manual approach include piecewise release of allocation list, untimely release of list, human manipulations and errors, and of course, the cumulative effects of all these on academic performance. In a bid to overcome these problems, the university authority sought for ways to accommodate more students. This led to the semi-privatization of hostels, industrial collaboration to build more hostels on build-operate-and-transfer agreement, and encouragement of students to seek off-campus residence. However, the introduction of some of these measures had led to more serious security and moral concern for the administration. This is why we believe that proper and efficient management of existing facility in order to ensure even distribution of students into hostel based on stated requirements will help the authority to overcome part of these problems. A complete automated hostel allocation system that incorporates efficient optimization techniques is therefore inevitable. This will ensure that a four-point goal of transparency, reliability, efficiency and effectiveness (referred to as the *TREE* goal) are achieved. Moreover, proper allocation will reduce stress for students and facilitates better academic performance. The desired end results is for the allocation list to be released on time (and at once), and be favourable to as many students as possible.

The structure of the overall automated system is given in the sequence diagram in Figure 3.1. There are three entities identified that influence the overall system. They are: the applicants (students), who must apply for hostel space; the Accommodation Officer, who enters and validates all applications; and then the allocation system which distributes available bed spaces among eligible applicants in a way as to meet stated requirements/constraints. Some basic definitions of important terminologies are provided in Section 2.1 of paper **[A]**.

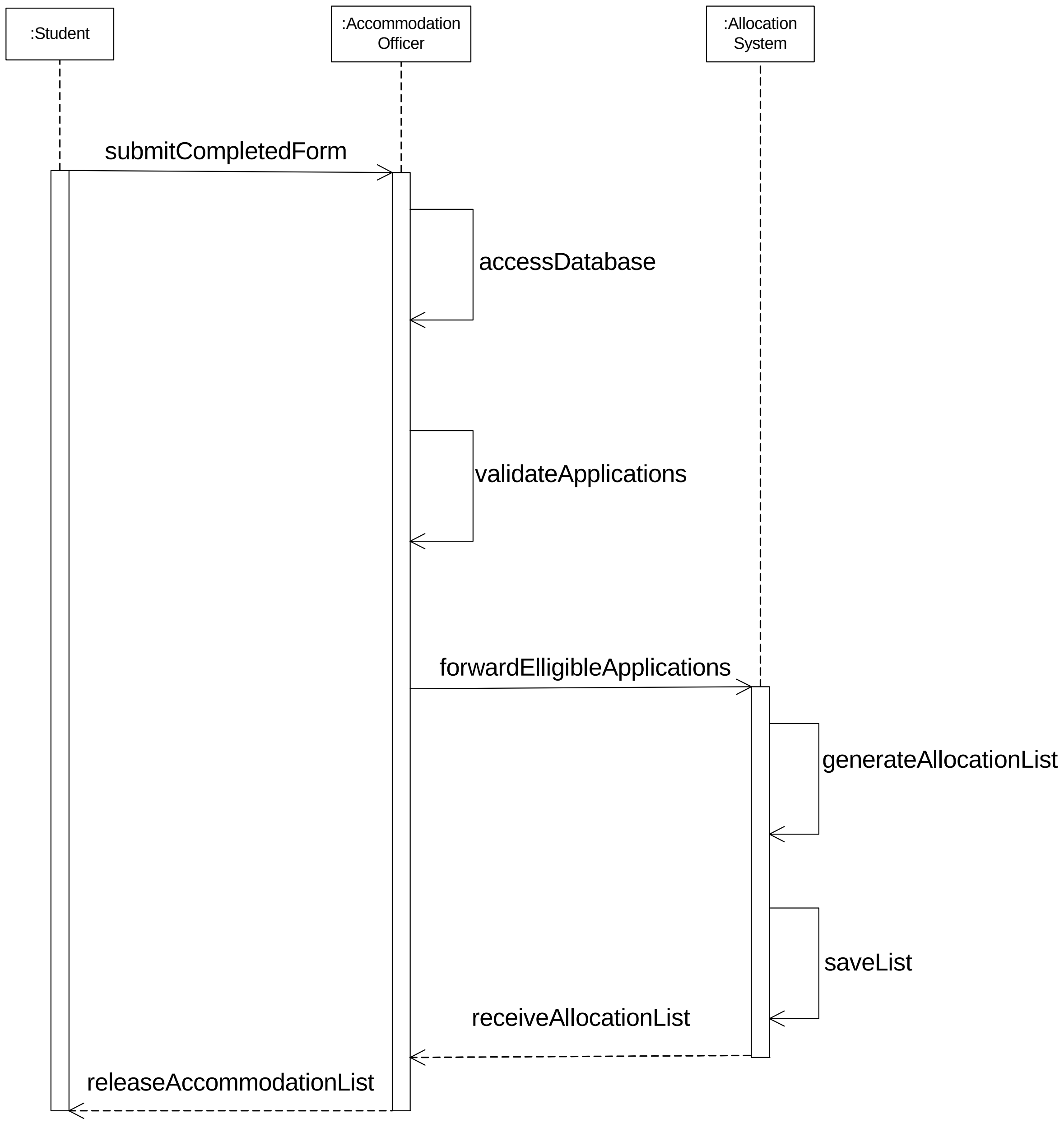

Figure 3.1: A sequential diagram for the HSAP

Since this thesis is concerned with determining the viability of metaheuristic application to the allocation distribution, we concentrated our simulation experiments on the allocation system. Aside the other subsystem, the allocation process consists of three stages namely, category allocation, hall allocation and block/floor allocation. The mathematical model HSAP had been earlier presented in Chapter 3, Section 2.3. Category allocation stage determines number of applicants in each category that can be accommodated without violating hall capacity constraints (Table 1 of paper **[A]**). Allocation at this stage must also take into consideration the allocation priority and mandatory requirements (Chapter 2, subsection 2.1.2.1). Results obtained from the category allocation passed to the hall allocation stage. Since there are separate hostels for undergraduate male and female students, we handled the allocation for these two in a mutually exclusive manner at the last two stages. The hall allocation stage determines the number of students in each category to be allocated to various hostels based on other set of constraints. The block/floor allocation stage takes the resulting distribution for each hostel and determines the number of students under each category to allocate to each block/floor within the hostel. This is done also in consideration of requirements/constraints that guard the distribution. To achieve the capacity constraints imposed at different stages, we classified the allocation into either fixed or flexible. For example, at the category allocation stage, the allocation of Ht, Sp and Fo categories are treated as fixed since all eligible students in these categories must be allocated. Other categories are treated as flexible allocation based on the given allocation priority (Section 2.2.2 of paper **[A]**). A sequential diagram illustrating the solution framework for the allocation system is shown in Figure 3.2.

We employed a simple greedy heuristic algorithm (Figure 3.3 where, C_1, C_2, C_3 are fixed allocation and $C_4,\ldots,C_8$ are flexible or free choice allocation) to handle the category allocation while remaining two stages were handled by two different but similarly designed and inter-dependent GAs (paper **[A]**, Section 3). The general structure of the GA metaheuristics is given in Figure 3.4.

In **[A]**, we classified the HSAP requirements given in Chapter 2, subsection 2.1.2.1 into either hard constraints or soft constraints. Hard constraints represent absolute limitations imposed on the system while soft constraints are necessary but no so important requirements that affect the overall quality of the allocation.

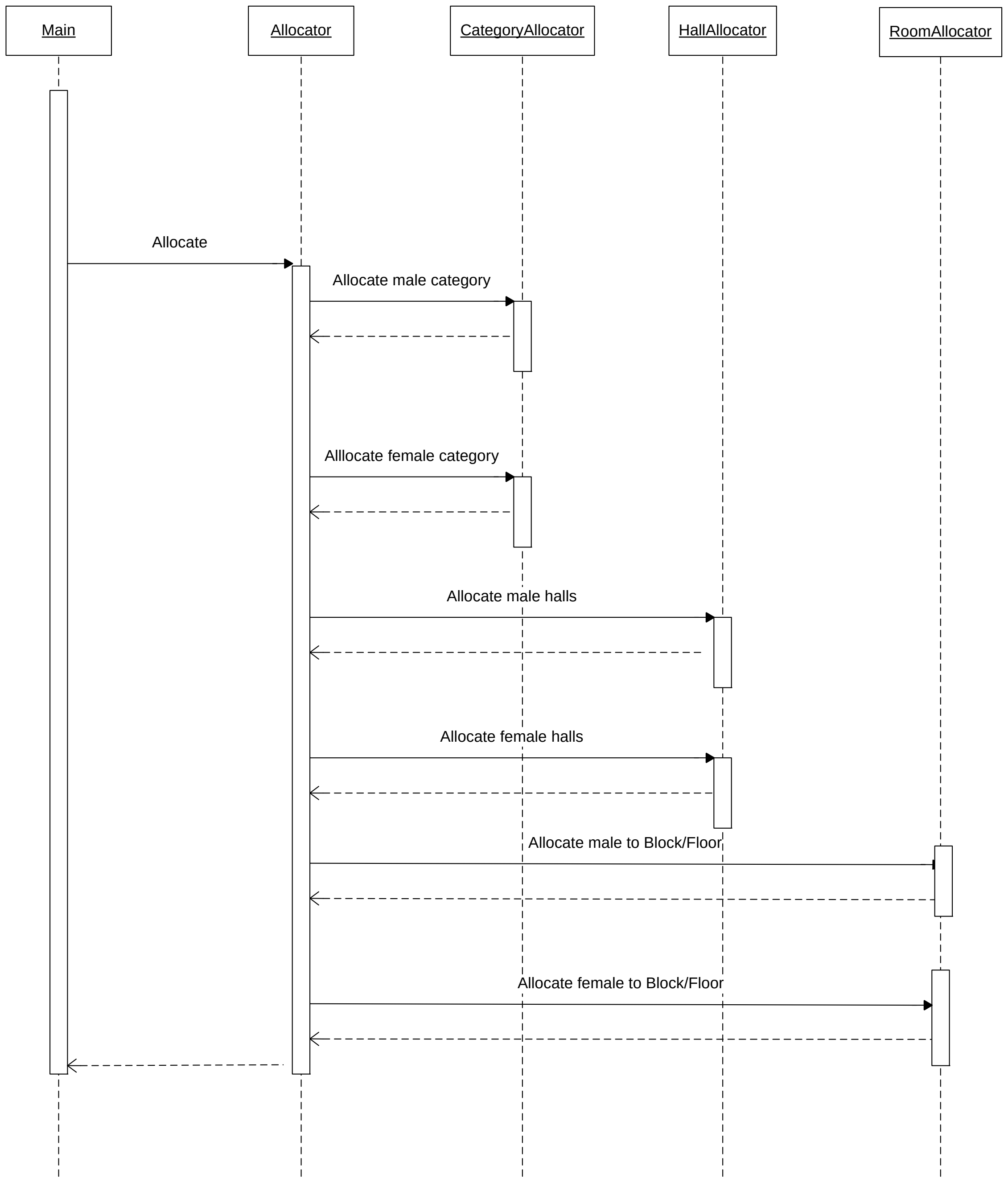

Figure 3.2: A sequential diagram of the hostel allocation generation subsystem

```
i.    Initialize: Set the total capacity of all Halls to TH, $C_i$ = 0 for all categories, Appls[i] = Total number
      of eligible applicants for category i; $C_i$ = allocation for category i, i = 1,2,…8, indexed such that the
      fixed categories, Ht, Sp and Fo are the first three, i.e. $C_1$, $C_2$, $C_3$.
ii.   Allocate Fixed Choice: Set $C_i$ = Appls[i], i = 1,2,3;
      Sum up the students in Fo, Ht and Sp given TFc and subtract from TH.
iii.  Allocate Free Choice:
      Initial:  Set rem = TH - TFc and Bool Ok ← FALSE;
      Prioritize: Set free choice categories $C_i$, i = 4,..,8 in order of priority such that $C_4 > C_5 > ... > C_8$.
      while (NOT Ok)
          rem = TH – TFc
          int remNew = rem;
         Allocate: Set $C_i$ = Min{remNew, Appls[i]},
                   Set remNew = remNew - $C_i$, i = 4,..,8 in order of priority
         CheckOk()
      End while
iv.   Calculate Unallocated:  Unallocated[i] = Appls[i] - $C_i$, for i = 4 to 8
v.    CheckOk: If $C_i$ >= 0,for all i = 4,..,8 Set Ok = TRUE
```

Figure 3.3: Structure of the greedy heuristic for category allocation

```
i.     Initialize: Generate initial population, NewPopulation, randomly
ii.    Evaluate: Calculate_Fitness (NewPopulation)
iii.   Set: Set CurrentPopulation = NewPopulation
iv.    While (NOT Terminal conditions)
v.         For counter = 1 to PopulationSize do
vi.            Selection:
                 Parent1 = Heuristic_Select (CurrentPopulation)
                 Parent2 = Heuristic_Select (CurrentPopulation)
vii.           Crossover:
                 Heuristic_Cross (Parent1, Parent2, NewPopulation)
viii.          Repair:
                 Heuristic_Repair (NewPopulation)
ix.            Mutation:
                 Mutate_Population (NewPopulation)
x.             Evaluate: Calculate_Fitness (NewPopulation)
xi.            Replace:
                 Replace_Population (Current Population, New Population)
xii.   endwhile
xiii.  Display Output
```

Figure 3.4: General structure of the genetic algorithms

A major problem with the system is the non-availability of archive data (past allocation) that can be used as benchmark for our study. Even the available allocation data at the time of study were very scanty and disjointed as the process was done in a piecemeal manner. Hence, the quality of a solution is determined by the degree of requirements/constraints satisfactions. To measure this, we define a space utilization factor, U, such that $0 \leq U \leq 1$ (paper **[A]**, subsection 2.2.1). This was used for appropriate fitness evaluations at both the hall and block/floor allocations. The overall goal therefore is to allocate hostel space such that utilization is maximized, that is, all hard constraints are satisfied and as many soft constraints are met as possible. A sample of the final allocation distribution obtained for both hall and block/floor allocation shows a high degree of satisfaction of given hard and soft constraints (see paper **[A]**, Tables A.1-A.3 and B.1)

GA researchers often report statistics based on GA parameters [66]. Some of these statistics are averaged over many different runs of the GA on the same problem [50,87]. Other statistics reported include the best fitness found in a run and the generation at which it was found, the size of the population, the rate of mutation and crossover, and the type and strength of selection [87]. For a problem domain therefore, an important experiment carry out a sensitivity analysis to determine the best parameter combination that gives the best results. This led to series of simulation experiments we conducted to determine the GA parameters values and the best combination of GA operators that give the best results for our problem instance. The experiment setup, results and conclusions reached are reported in Section 4 of paper **[A]**.

Furthermore, we carried out series of simulation experiments to determine the rate at which feasible solutions, that is allocation that do not violate any stated hard constraint, are obtained by the combined GA. This was conducted using different values of parameter combinations for each experiment. In an experiment, the crossover rate (P_c), mutation rate (P_μ), population size (N) were fixed and the algorithm executed 50 times independently. For this study, the following combinations (P_c, P_μ, N) were chosen – (0.1, 0.6, 90), (0.2, 0.5, 90), (0.2, 0.8, 50), (0.3, 0.3, 100) and (0.3, 0.7, 70). Any other combinations with good results can be used also. The number of generation was fixed at 1000 for all experiments. For each execution, the number of generation evolved and total number of feasible solutions over all generations were noted. The feasibility rate is computed as the average of the number of feasible solutions to the total number of solutions. One of the results obtained is reported in paper **[A]** (Table 3, Figures 9 and 10). We provide in Table 3.1, Figures 3.5 and 3.6, another set of results for parameter set (P_c = 0.1, P_μ = 0.6, N = 90). The results for this combination, (P_c = 0.1, P_μ = 0.6, N = 90), follows similar pattern with that reported in paper [A] for combination (P_c = 0.3, P_μ = 0.3, N = 100).

The GA metaheuristic framework reported in paper **[A]** does not aim to compete with other state-of-art problem specific methods but to provide a generalized approach for handling HSAP with solutions that are "*good enough, soon enough and cheap enough*" [**13**]. This implies solutions that are of good quality, converges, and whose time and space complexity are reasonable. Based on the cumulative results and observations from conducted experiments, the following parameter combinations are recommended (depending on computing resources consideration):

- For speedy execution (that is solutions requiring fewer number of generations to converge): (P_c = 0.2-0.5, P_μ = 0.7-0.9, N = 80-100)
- For accuracy (solutions with very high fitness values that are near optima): (P_c = 0.3-0.4, P_μ = 0.6-0.9, N = 70-100)
- For minimal use of resources (solutions requiring less amount of intermediate processing): (P_c = 0.2-0.4, P_μ = 0.3-0.7, N = 60-80)
- For consistent optimal results (i.e. solutions with good mix of high accuracy, speedy execution and minimal resource usage): (P_c = 0.3-0.5, P_μ = 0.3-0.7, N = 60-90).

From our results, we conclude that GA metaheuristic is highly efficient in handling the HSAP. It gives results that meet stated requirement thus will be very useful in improving the hostel space allocation process. We however note that there are rooms for improvements on the results obtained especially if the results of the initial stage (category allocation) can be enhanced. This in turn will affect the results of the remaining two stages. This led to our study and presentation in Chapter 4.

Table 3.1 - Results of experiment to determine rate of feasibility on the combination (0.1, 0.6, 90).

Run#	Male			Female		
	no. of gens	# feasible solutions	Feasibility rate(%)	no. of gens	# feasible solutions	Feasibility rate(%)
1	244	6739	30.69	253	18003	79.06
2	181	9905	60.80	176	8814	55.64
3	186	4812	28.75	213	8541	44.55
4	192	6471	37.45	184	4313	26.04
5	189	8586	50.48	187	4896	29.09
6	295	8038	30.28	167	3229	21.48
7	339	7282	23.87	390	6892	19.64
8	135	3067	25.24	144	9705	74.88
9	122	5946	54.15	146	7218	54.93
10	234	6399	30.39	243	15593	71.30
11	288	5376	20.74	278	11953	47.77
12	272	11538	47.13	102	7114	77.49
13	245	7745	35.13	144	7568	58.40
14	277	14455	57.98	319	6073	21.15
15	195	9424	53.70	223	7280	36.27
16	250	10010	44.49	149	7675	57.23
17	230	5196	25.10	196	4582	25.98
18	271	16064	65.86	172	5635	36.40
19	257	7573	32.74	200	3478	19.32
20	226	10190	50.10	231	9329	44.87
21	237	4093	19.19	166	4282	28.66
22	262	4738	20.09	294	14279	53.96
23	277	15909	63.82	156	4891	34.84
24	269	13991	57.79	251	6209	27.49
25	203	5221	28.58	172	5191	33.53
26	280	9260	36.75	250	825	3.67
27	195	9702	55.28	254	10921	47.77
28	314	17196	60.85	164	1557	10.55
29	271	6343	26.01	262	16951	71.89
30	269	14590	60.26	255	8788	38.29
31	98	1	0.01	60	1	0.02
32	237	6748	31.64	308	17061	61.55
33	226	8388	41.24	242	10149	46.60
34	274	4553	18.46	161	5568	38.43
35	283	11781	46.25	312	11420	40.67
36	391	17983	51.10	189	11829	69.54
37	299	12249	45.52	233	13565	64.69
38	277	14160	56.80	167	8656	57.59
39	230	9698	46.85	317	5602	19.64
40	209	12658	67.29	200	2659	14.77
41	266	11277	47.11	185	7364	44.23
42	163	6599	44.98	250	11415	50.73
43	211	10678	56.23	352	21156	66.78
44	221	3483	17.51	163	11261	76.76
45	291	11337	43.29	308	5659	20.41
46	214	3749	19.47	262	9544	40.47
47	194	5491	31.45	256	8762	38.03
48	274	13946	56.55	209	8425	44.79
49	315	6796	23.97	196	3013	17.08
50	352	14450	45.61	187	8502	50.52

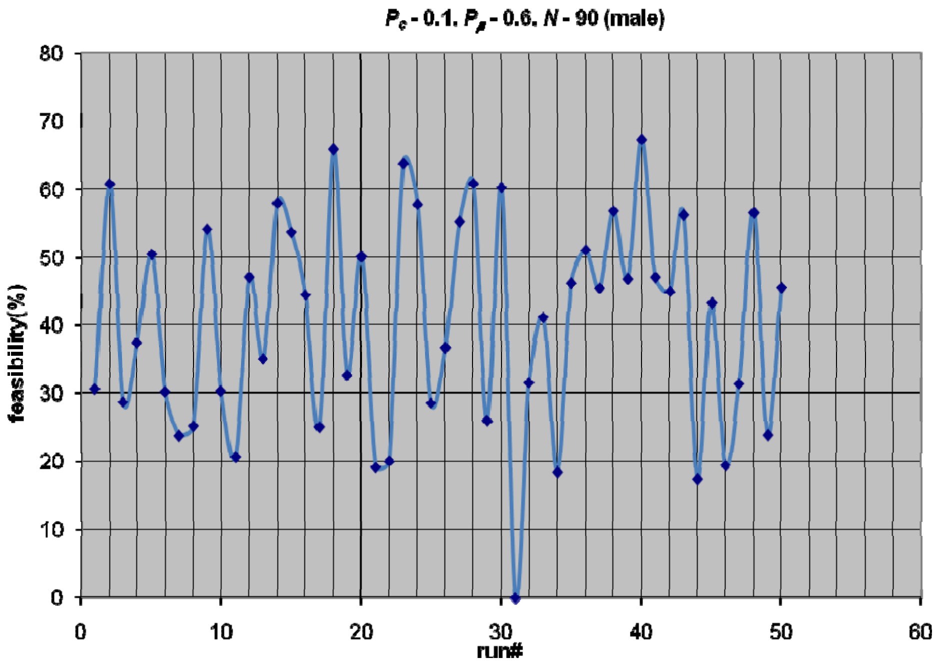

Figure 3.5: Feasibility study for male allocation (P_c = 0.1, P_μ = 0.6, N = 90)

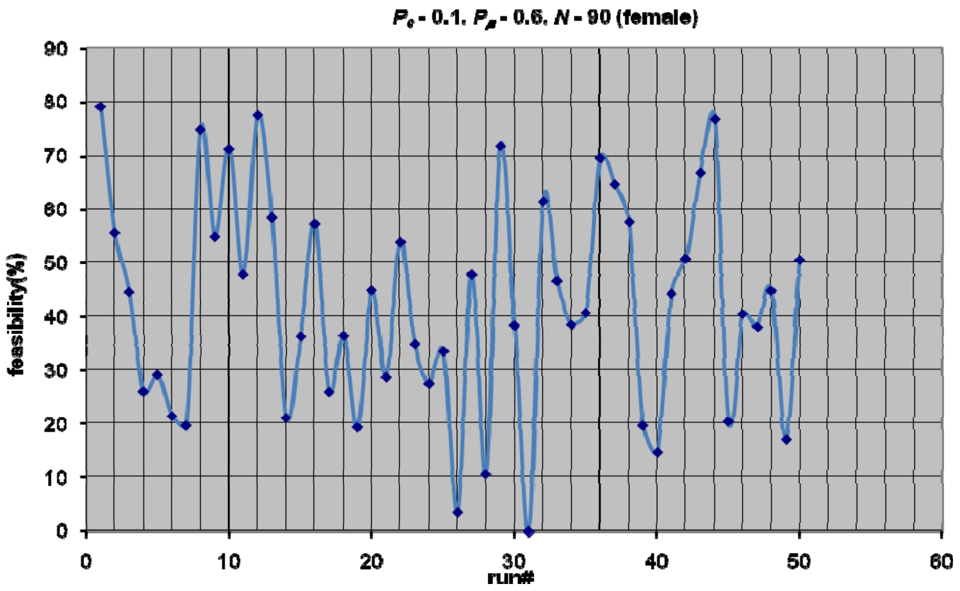

Figure 3.6: Feasibility study for female allocation (P_c = 0.1, P_μ = 0.6, N = 90)

INCLUDED ARTICLE

Paper [A]

Multi-level Genetic Algorithm for a Multi-stage Space Allocation Problem

Adewumi, A.O. and Ali, M.

Mathematical and Computer Modeling 51 (2010) 109_126
DOI:10.1016/j.mcm.2009.09.004

© 2010 Elsevier Ltd. Reproduced with permission

Chapter Four

Paper [B]: A Hierarchical Heuristic Strategy for Hostel Space Allocation Problem

"Nearly every man who develops an idea works at it up to the point where it looks impossible, and then gets discouraged. That's not the place to become discouraged."

- Thomas Edison

"Success seems to be connected with action. Successful people keep moving. They make mistakes, but they don't quit."

- Conrad Hilton

“That which we persist in doing becomes easier - not that the nature of the task has changed, but our ability to do has increased”

- Ralph Waldo Emerson

Paper **[B]** presents some further studies on the HSAP reported in Chapter 3. The mathematical models follow what we defined also in Chapter 3, Section 2.3. However, for comparative study purpose, we further seek to combine the models for the first two stages into a single mathematical model. This is presented later in this chapter.

For better understanding, we present a graphical illustration of the problem and a breakdown of constraints in paper **[B]** (Section 2.1, Figure 1, Tables 2 & 3). To understand the layout of the hostels, we present a graphical layout according to the zoning (see Table 1, paper **[B]**) in Figure 4.1**.** For generalization purpose, we use zone-based hostel identification (hostel ID) instead of the actual names used in paper **[A]**. For example, HA1 refers to hostel 1 in zone A. For cross-examination purposes therefore, Table 4.1 gives the names of the hostels with the corresponding ID used in paper **[B]**.

Table 4.1: Hostel names and identification used

Zone (Area)	**Hostel Names**	**Hostel ID**	**Sex**
A (Main Campus)	Jaja	HA1	Male
	Mariere	HA2	Male
	Moremi	HA3	Female
B (New Hall)	Eni Njoku	HB1	Male
	Aliyu Makama Bida	HB2	Female
	Fagunwa	HB3	Female
	Madam Tinubu	HB4	Female
	Sodeinde	HB5	Male
C (Gate/Education)	El-kanemi	HC1	Male
	Kofo Ademola	HC2	Female
	Queen Amina	HC3	Female
	Saburi Biobaku	HC4	Male

We developed a multi-level structure heuristics and metaheuristics, jointly called a hierarchical heuristic strategy, to solve the HSAP. Having successfully applied GA metaheuristics in our earlier study (Chapter 2), we set out this new study to 1) test other heuristics on the first two levels, that is, category allocation and hall allocation, 2) develop a heuristic for hall allocation that will maximize the distribution spread of categories of students into available hostel space. This, in turn, is to prevent 1) clustering of the same category into the same hall, and 2) bias distribution in which category of higher priorities are allocated to the detriment of those of lower priorities. We therefore aim at given at least some students in the lower category some chance of being accommodated while still observing the allocation priority requirement.

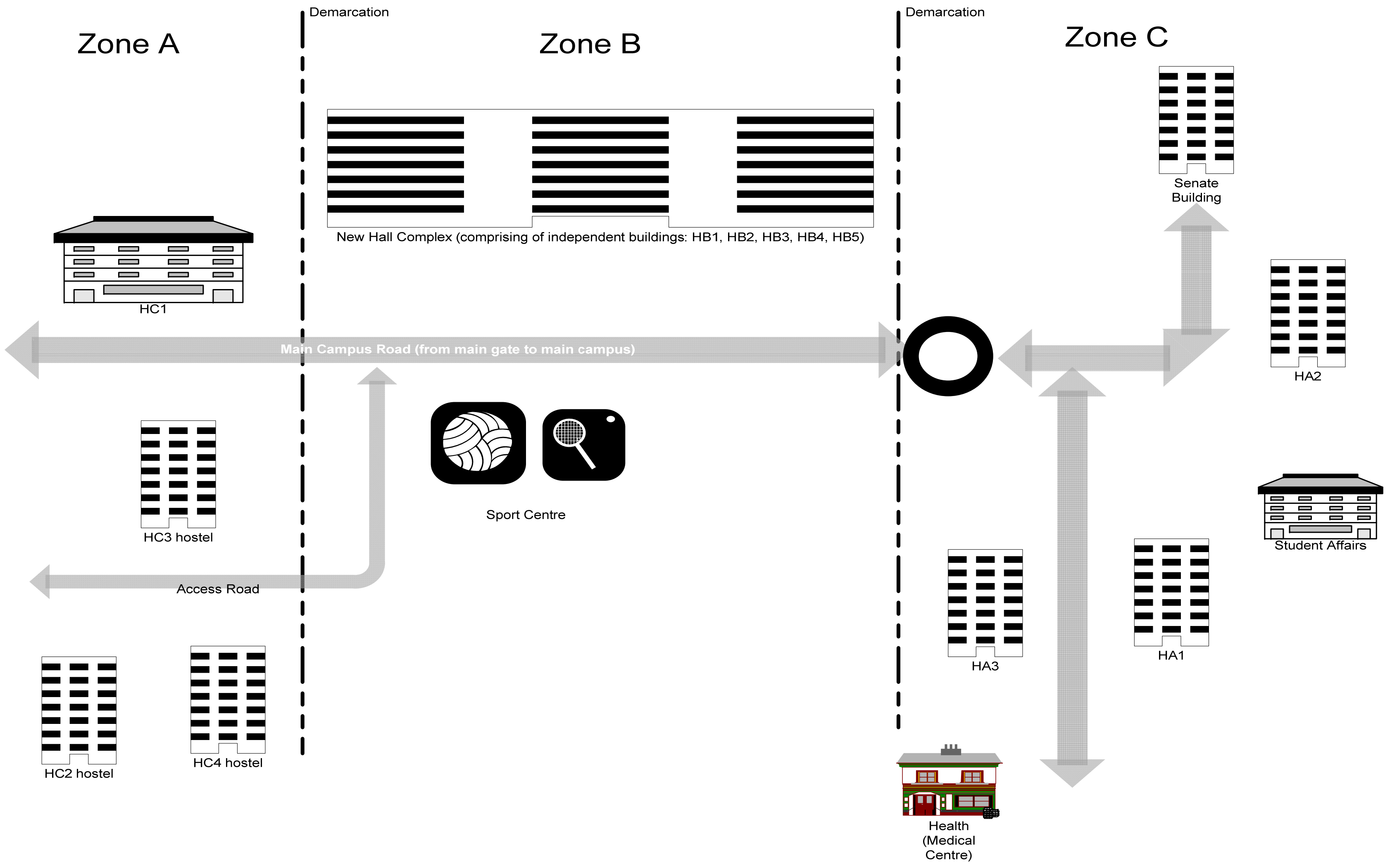

Figure 4.1: Graphical layout of hostels distribution and zoning

For ease of experimentation, we divided the allocation at the first two levels into either *fixed-choice* or *free-choice* depending on the strict hard constraints to affect allocation at each level. For example, at the category allocation level, all Ht, Fo and Sp must be accommodated hence they are regarded as *fixed-choice*.

Different heuristics were designed to handle the first two stages while GA metaheuristics, *FaGA,* still drives the final floor level allocation. For experimental and comparative study purposes, two different heuristics, *CAH1* and *CaH2,* were designed for the category allocation stage. *CaH1* is still a greedy-like heuristics as in the last Chapter (paper **[B]**, Section 3.1). *CaH2* heuristic uses a percentage ranking system to determine the number of students to allocate in each category (paper **[B]**, Section 3.2). Similarly, GA metaheuristic, *HaGA* and a new heuristic, *HaNH*, were designed for the hall level allocation. *HaNH* heuristic uses a parameter called, hall ratio (HRj, j=1..n), to distribute students in each category into various hostel (paper **[B]**, Section 3.3). The block/floor level allocation was handled by a GA metaheuristic, *FaGA*. The algorithms for the heuristics are provided in paper **[B]** (Section 3). Both *HaGA* and *FaGA* metaheuristics (paper **[B]**, Section 3.5) are similar to the one in the last Chapter. As noted in Chapter 3, a succeeding stage depends on the results obtained from the previous stage. The overall structure of the solution methodology is provided in Figure 2 of paper **[B]**.

Results obtained by the heuristics pair for the category level are compared and presented. The *fixed-choice* categories are first allocated to specified halls as required. The heuristics then seek to distribute the *free-choice* into remaining hall capacities so as to 1) follow the given allocation prioritization and 2) produce an allocation that maximizes distribution spread into various hostels. Figure 4.2 gives a graphical summary of the results of the *CaH1* and *CaH2* heuristics that are reported in paper **[B]** (Tables 5 & 6). The *y-axis* represents the utilization factor, *U*, obtained by dividing the actual number of students allocated, C_i, by the total number of applicants, *Appl[i]*, for each category. The *x-axis* represents the categories in *free-choice* allocation for category allocation stage. C4, C5, C6, C7, and C8 in the both represent the Fy, Sc, Fr, Ds and Ot student categories respectively. Note that the *fixed-choice* categories of Ht, Fo and Sp have a mandatory allocation to specified hall. Figure 4.2 show that both *CaH1* and *CaH2* meet prioritization requirement as shown by the sloppy nature of the graphs. However, *CaH2*, gives an allocation that is better spread among the *free-choice* categories. *CaH1* gives a more biased distribution that favours categories of higher priority and neglect those of lower priorities, depending on the available hall capacity. *CaH2* therefore produces solution of better quality than *CaH1*.

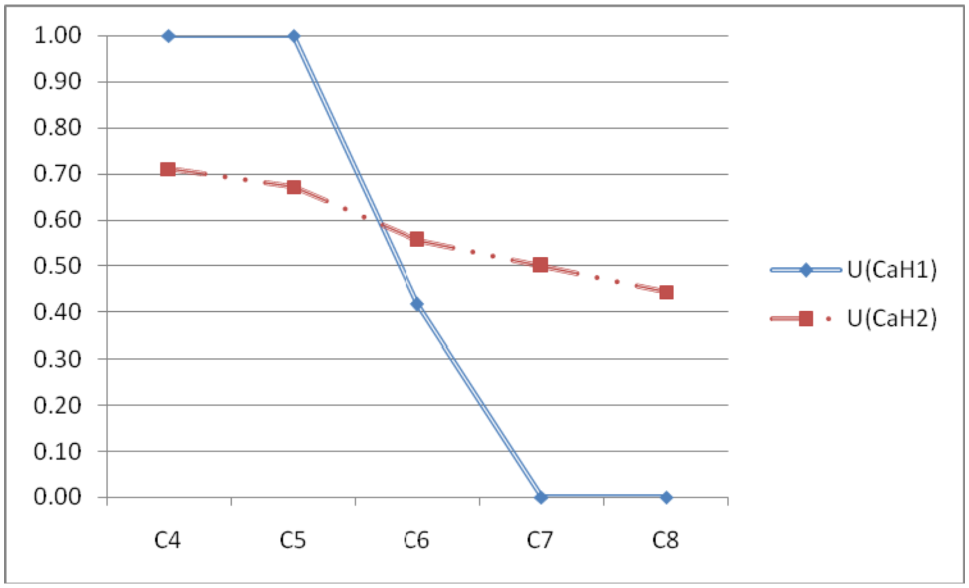

Figure 4.2: Comparative study of category allocation based on *CaH1* and *CaH2* heuristics

The hall distribution obtained with the application of *HaGA* and *HaNH* are reported in Tables 7 & 8 of paper **[B]**. Since hall allocation level depends on results from the category allocation, we expect similar pattern of distribution from both *HaGA* and *HaNH* based on given input. We therefore only reported the results based on the combination of *CaH1* with both *HaGA* and *HaNH* in paper **[B]**. Two pie chart illustrations for results of *HaGA* and *HaNH* are presented in Figures 4.3 and 4.4.

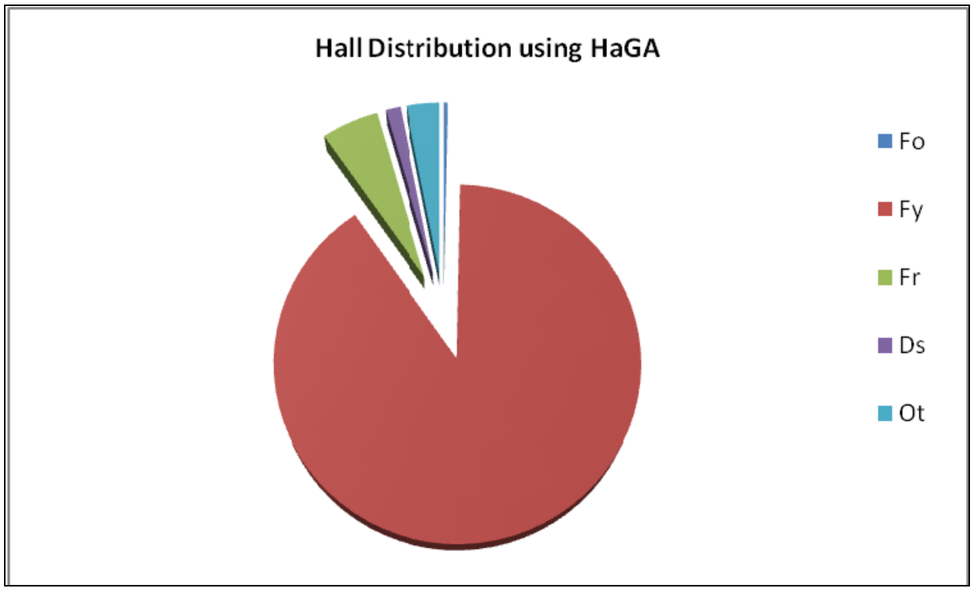

Figure 4.3: Hall distribution based on HaGA

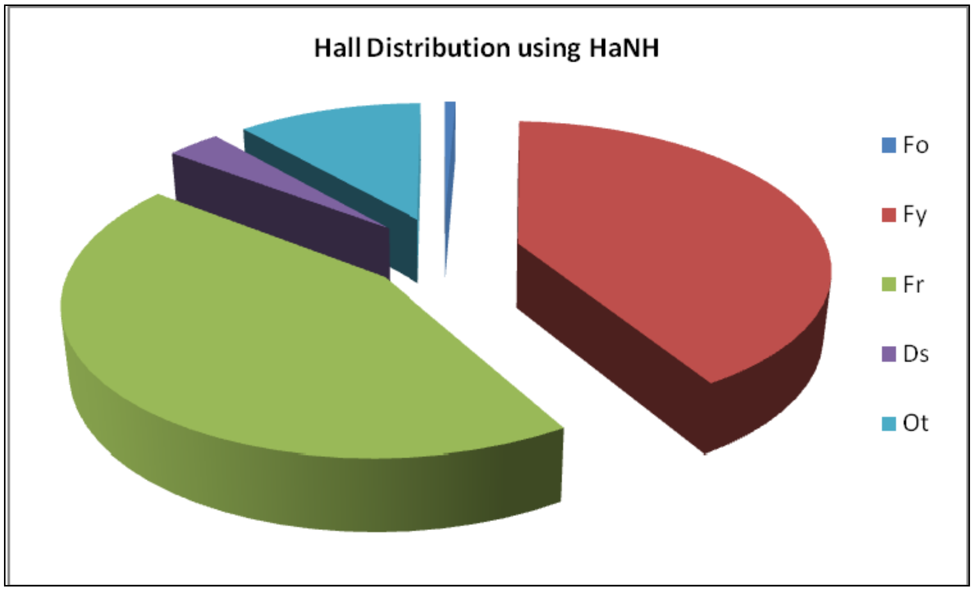

Figure 4.4: Hall distribution based on HaNH.

Observation from the results in Tables 7 and 8 of paper **[B]** both *HaGA* and *HaNH* satisfy the hard constraints for the *fixed-choice* allocation. However, Figures 4.3 and 4.4 show that *HaNH* heuristic maximizes the distribution spread of the *free- choice* categories more than the *HaGA* heuristic at the hall allocation stage. For example, though Fy category has priority over Fr category, *HaGA* heuristic however, allocated them to hall based on this priority but also considering the number of applicants in the two categories thus removing the possibility of priority-based lopsided or biased allocation that favours Fy when HaGA heuristics was applied.

As stated earlier, for ease of usage and comparative purpose, we try to combine the mathematical models for the category and hall allocation stages into a single model. This is presented as follows:

We assume the following constraints: (a) All Fo must be allocated, (b) All Ht must be allocated, (c) Ht must be allocated at the lowest oor as possible in a given hall, (d) All Sp must be allocated, (e) All allocated Fy should be allocated to the highest oor as possible in a given hall, (f) As many Fy, as possible, should be accommodated, (g) As many Sc, as possible, should be accommodated, and (h) The order of priority of allocation is Fy, Sc, Fr, Ds and Ot. The first five constraints are hard constraints while the others are soft constraints. The objective is to maximize bed space utilization so as to satisfy specified hard and soft constraints.

Let ω_{ij} be the satisfaction weight if a student of the category i is allocated in the hall j and x_{ij} be the number of student of category i allocated in the hall j, $i = 1,\ldots,$m and $j = 1,\ldots,n,$ m is the total number of categories while n is the total number of halls. We then define a satisfaction function as

$$S = \sum_{i=1}^{m}\sum_{j=1}^{n} \omega_{ij} x_{ij} \qquad (4.1)$$

Next we define L_i and U_i to be the lower and upper bound, respectively, of the number of students to be allocated in category *i*, while h_j the total bed space capacity of the hall *j*.

In order to satisfy the hard constraints (a), (b) and (d) above, we set their lower bound of the number of students allocated in the fixed-choice categories to be equal to the number of eligible students in those categories. For the flexible-choice categories, we set the lower bound to be 0 and the upper bound to be the number of eligible students within the concerned category. The formulation thus becomes:

$$\text{Maximize } S = \sum_{i=1}^{m}\sum_{j=1}^{n} \omega_{ij} x_{ij} \qquad (4.2)$$

subject to

$$\sum_{i=1}^{m} x_{ij} \le h_j \qquad \text{for } j = 1,\dots,n \qquad (4.3)$$

$$L_i \le \sum_{j=1}^{n} x_{ij} \le U_i \qquad \text{for } i = 1,\dots,m \qquad (4.4)$$

$$x_{ij} \in \{0,1,\dots.\} \qquad \text{for } i = 1,\dots,m \text{ and } j = 1,\dots,n \qquad (4.5)$$

Equation (4.3) stipulates that the total number of students allocated in hall *j* should not exceed the capacity of the hall, while equation (4.4) represents the constraint of lower and upper bound. We however assume in this model that the values of ω_{ij} is assigned by the accommodation officer in charge at the Students Affair's Office based on the order of priority assigned to the allocation of each category and also on constraints (c), (e) and (h).

For comparative purpose, we employed simulated annealing (SA) algorithm [see 47,49,73] to compute the solution for male student for the first two category and hall allocation stages based on the new model. We used the same set of given input as employed in main experiment described earlier in this Chapter and in paper **[B]**. In the SA implementation, we chose the cooling function to be $\phi(t) = T_0\alpha^t$ with $0 < \alpha < 1$.

Since the experiment is for comparative purpose, we set $\alpha = 0.9$, and the initial "temperature" T_0 = 100. The algorithm was set to stop after a certain number of iterations.

Using the same set of input as in Appendix A of paper **[B]** for male student categories only, the SA generated the hall distribution results as presented in Table 4.2.

Table 4.2: Comparative Results obtained using Simulated Annealing

Category	HA1	HA2	HB1	HB5	HC1	HC4	Percentage (%)
Fy	404	150	110	110	110	340	86.19
Sc	5	65	65	65	65	6	63.02
Fo	3	3	3	3	4	4	100
Ht	10	10	10	10	15	15	100
Fr	5	162	200	200	200	5	57.95
Sp	200	40	40	40	40	40	100
Ds	8	10	15	15	30	2	66.66
Ot	25	0	300	325	62	100	47.76
Total	660	440	743	768	501	512	

A study of the above results shows a good level of satisfaction of some give constraints by the SA algorithms. However, one could notice that the distribution spread objective was better satisfied by our earlier *HaGA* and *HaNH* heuristics than the SA (compare Tables 4.2 with Tables 7 & 8 of paper **[B]**), thus our heuristics proved to be better than SA algorithm. Similarly, the hard constraints regarding the fixed-choice allocation is better satisfied by *HaGA* and *HaNH* than the SA (compare Tables 4.2 with Tables 7 & 8 of paper **[B]**).

INCLUDED ARTICLE

Paper [B]

A Hierarchical Heuristic Strategy for Hostel Space Allocation Problem

Adewumi, A.O. and Ali, M.

Submitted to the Journal of the Operational Research Society (JORS)

Chapter Five

Paper [C]: A Heuristic Solution to the University Timetabling Problem

“That some achieve great success, is proof to all that others can achieve it as well.”
– Abraham Lincoln

"There is no one giant step that does it. It's a lot of little steps."
- Peter A. Cohen

“You may never know what results come of your action, but if you do nothing there will be no result”
- Mahatma Gandhi

TTP is a special class of NP-hard optimization problem that come up every year in educational institutions [54,61]. The use of computer-based solution is limited by the complexity of the problem. This explains the drive for the application of global optimization methods in solving the TTP. GAs have proved to be robust for this kind of problems [5,7,75]. In paper [C], we present a flexible representation and solution approach for the LTTP as obtained in the instanced considered (see Chapter 2, subsection 2.1.1). Further detail of the LTTP case instance is presented in Section 2.1 of paper **[A].** The overall goal of the LTTP is to assign lectures (courses/classes) into a set of time slots in such a way that satisfy given constraints and optimize a set of objectives. Common hard constraints considered in a typical UTTP include: lecturer must teach only one class at a time; a classroom cannot be allocated more than one course at a time; lecturer may only teach courses in his specialty (for example, a Computer Science Lecturer cannot be assigned to teach a Chemistry course); and the same class of students must not be doubly booked for compulsory courses at the same time. There are also soft constraints that influence the solution quality of timetable. However, in a real-world scenario especially with large student population, it is almost impossible to satisfy all soft constraints. In most cases, as in the current case instance, they are completely overlooked and where need be, assigned minimal weight for fitness evaluation purposes. Some soft constraints might include non-consecutive allocation of classes to Lecturers, preferential timeslot or classroom allocation, and proximity requirement (for example, classrooms may be booked close to the home department of a course). In our study instance, most soft constraints are implicitly taking care of by the arrangement of classrooms around department/faculty and the solution approach we adopted. The main emphasis therefore is to concentrate on the non-violation of hard constraints.

As stated in paper **[C]** (Section 2.1), the arrangement of most classrooms around faculties makes it easy for us to adopt a bottom-up approach in timetable construction. This makes it possible to build the timetables around departments which eventually accumulate into faculty and university timetable. Our approach reported in the paper allows most constraints to be specified as file inputs. Construction of timetable at higher level therefore only requires appending relevant courses, lecturer and classroom data into appropriate files for processing. The simulation effort was thus concentrated on testing the viability of GA for the problem and carrying out sensitivity analysis to determine GA parameter combinations that give quality solution for it. Our approach implicitly handles some hard constraints thus making the definition of the fitness function (paper **[C]**, subsection 3.2). The two-dimensional chromosome representation adopted (paper **[C]**, Section 3.1) also implicitly takes care of class clash constraint.

Furthermore, building of timetable around departments implicitly handles some fundamental constraints. For example, conflict of classes for students, conflict of classes for lecturers, and lecturer teaching courses in area of specialty are taken care of at the departmental level. The issue of *when* for the LTTP is taken care of naturally (see Chapter 2, subsection 2.2, paragraph 2). The focus then is essentially on timetabling as a SAP that is concerned with the question of *where,* that is, room allocation to courses. There are two fundamental constraints that are universal to all timetabling and SAPs in general, and that no feasible solution may violate. These are: no entity can be in more than one location at any one time. For each time period, there should be sufficient resources available for all the events that have been scheduled for that time period. This implies then that 1) lecturer must not be doubly booked (having two different courses taken by the same lecturer allocated to the same time slot); 2) room capacity must be appropriate with the size of the class and all classes (courses) must be assigned to rooms; and 3) class clash error, which makes students at the same level to be assigned the same timeslot for two separate courses, must be avoided. Aside, only one class can be assigned to one room at any one time. The fitness evaluation therefore was designed as a measure of the degree of violation of these hard constraints (see paper **[C]**, Section 3.2). The function determines the number of lecturer doubly booked errors, room too small errors, and related class errors and use them to compute the degree of fitness of a generated solution. The fitness function therefore takes values in [0,1], with 0 representing a high quality (optima) solution and 1 representing complete violation of all hard constraints.

The GA metaheuristic employed is presented in Section 3 of paper **[C]**. A simple class diagram representing the solution framework is given in Figure 5.1.

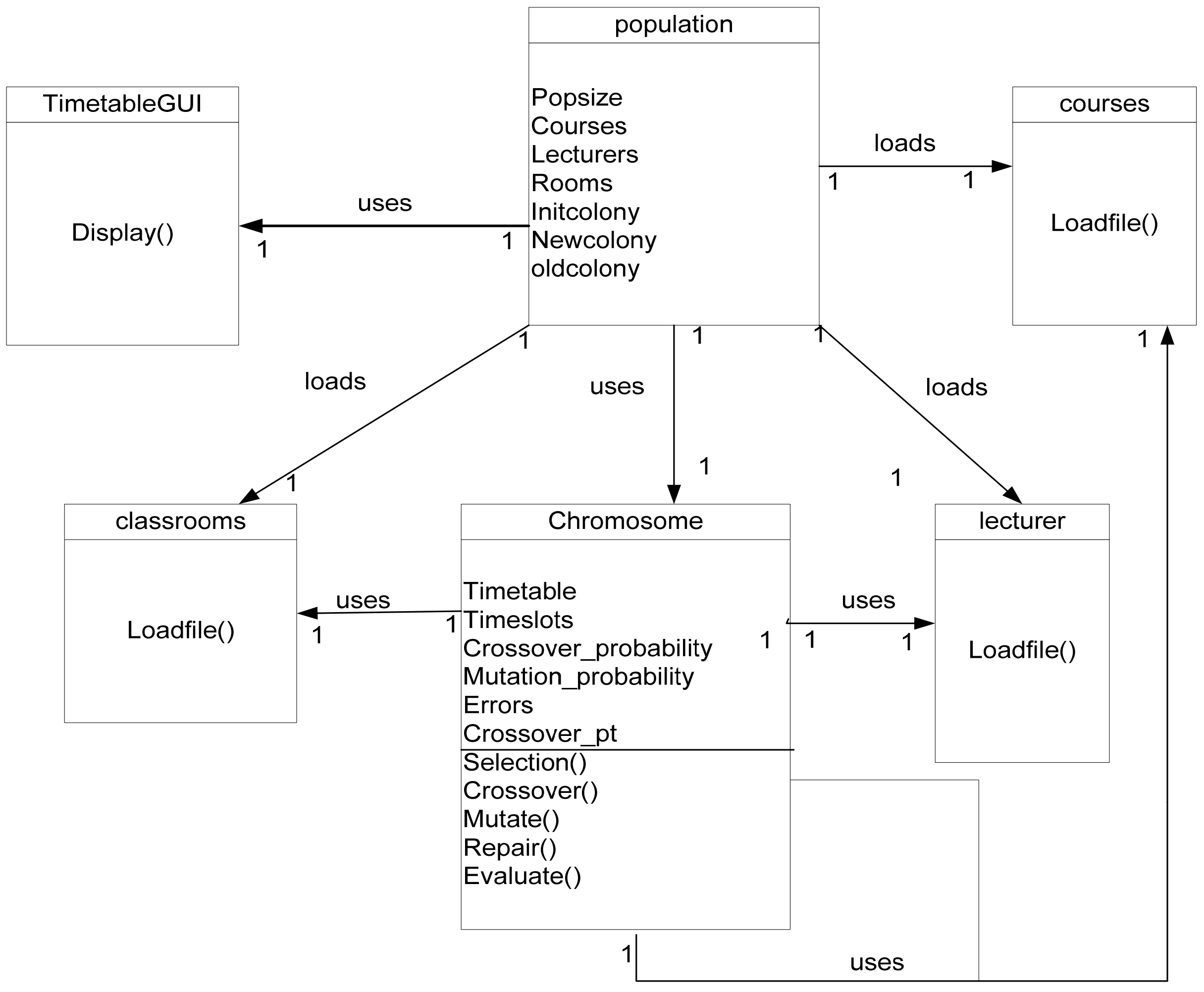

Figure 5.1: A class diagram framework of the solution approach

As seen in Figure 5.1, the main interface of the program developed is TimetableGUI which make used of the population class. The population class in turn called the classroom, lecturer and courses subclasses which load relevant constraint data inputs into the system. The population class then invokes the chromosome class to execute the GA metaheuristics given in paper **[B]**, Section 3. The repair strategy was done in two stages. The first stage ensures that offspring generated after applying GA operators are within the defined search space. It is essential that each class was booked in the chromosome. The second stage ensures that there is exactly one booking of each class in the generated offspring.

Our attempt is the first reported application of metaheuristic to LTTP in the context of our case study. The objective is to test the viability of GA in handling the problem instance in our case study. The need to study the new instance of SAP, that is, the HSAP, for the same institution prevented testing of other heuristics or variants on the LTTP. We have however shown that it is viable to apply metaheuristic to this problem instance.

INCLUDED ARTICLE

Paper [C]

A Heuristic Solution to the University Timetabling Problem

Adewumi, A.O., Sawyerr, B.A. and Ali, M.

Engineering Computations: International Journal for Computer-Aided Engineering and Software 26 (8) 2009 972-984

© Emerald Group Publishing Limited 0264-4401

Chapter Six

Paper [C]: A Comparative Study of Some Real Coded Genetic Algorithms for Unconstrained Global Optimization

"I find that a great part of the information I have were acquired by looking up something and finding something else on the way"

- Frankling P. Adams

Paper **[D]** is the outcome of the study on GA application to unconstrained global optimization problems with continuous variables. The paper attached to this thesis is a revised version of the original submission based on the Reviewer's comments received from the journal Editor. In the paper, we presented a set of new RCGAs that have ability to perform both local and global exploratory search. The algorithms were developed as a hybridization of the SRCGA (see paper **[D]**, Section 2) with local search heuristics namely, PS and vector projection. A limited version of PS heuristic (see paper **[D]**, subsection 3.1) was use to modified the crossover operator for RCGA in order to improve its robustness and efficiency. We further introduced a new vector projection global exploratory method (see paper **[D]**, subsection 3.3). We combined these algorithms in such a way that give three new variants which were all tested along side with SRCGA on 57 test problems. The variants are RCGA-PS (RCGA with PS incorporated into the crossover procedure, - see paper **[D]**, subsection 3.2); RCGA-P (RCGA with incorporated projection based exploratory mechanism at the end of each generation of the SRCGA - see paper **[D]**, subsection 3.4); and RCGA-PS-P (similar to RCGA-PS but augmented with projection based exploration at the end of each iteration of the RCGA-PS - (see paper **[D]**, subsection 3.5). These algorithms, alongside with SRCGA, were tested on various dimensions of the test problems, ranging between 2 and 30. All the algorithms used the same GA parameter values as shown in paper **[D]**, subsection 4.1 and Table 1. Each algorithm was run independently for 100 trials on each of the 57 benchmark problems to determine its success rate.

Statistical analysis was conducted to determine how the new modified algorithms fare in comparison with the SRCGA. Criteria used for results evaluation include best fitness values, mean best fitness value, mean function evaluations, success rate, standard deviations, and p-value from ANOVA test (see paper **[D]**, Section 5, Tables 2, 3, 4, 5, Appendix II & III). We also applied the Success Performance (SP) index for ranking of the algorithms (see paper **[D]**, Section 5, Table 6, Appendix I). Graphically, box-plots and multiple comparison (MCx) graphs were generated to compare the algorithms (see paper **[D]**, Appendices IV & V). RCGA-PS, RCGA-P and RCGA-PS-P were compared with SRCGA using the stated criteria (see paper **[D]**, subsection 5.1). Aside these comparative studies, experimental results from the four algorithms are compared with similar studies in literature (see paper **[D]**, subsection 5.2).

In all, we discovered that RCGA-PS, RCGA-P and RCGA-PS-P perform better than SRCGA thus the local and global exploratory algorithms introduced helped to improved the performance of RCGA with RCGA-PS-P giving the best performance. RCGA-PS-P also performed better than recent algorithms from literature.

Paper **[D]** has been reviewed by two Reviewers appointed by the Journal of Optimization Methods and Software of the Elsevier Science (see Appendix A). As noted by the Reviewers, Ackley, Griewank, Rastringn, Rosenbrock and Schwefel problems constitute a group of five test problems which possess varying level of difficulty as the dimension increased from 2 to 30. Our initial study on these five problems ranged from 2 to 10 dimensions. We therefore conducted more experiment on this group of test problems using dimensions 10, 20 and 30. Experimental results obtained are reported in Table 6.1. The results further confirm the superiority of the improved RCGAs over SRCGA with RCGA-PS-P still performing best.

Table 6.1: Comparative study of SRCGA, RCGA-PS, RCGA-P and RCGA-PS-P on selected problem with dimension 10, 20 and 30

Dimension = 10

Pro. #.	Global min	Min				SR				MBF of successful runs				MFE of successful runs			
		SRCGA	RCGA-PS	RCGA-PS-P	RCGA-P	SRCGA	RCGA-PS	RCGA-PS-P	RCGA-P	SRCGA	RCGA-PS	RCGA-PS-P	RCGA-P	SRCGA	RCGA-PS	RCGA-PS-P	RCGA-P
39	0.00000	8.08E-04	5.50E-05	**0.00E+00**	**1.00E-06**	100	100	100	100	2.57E-03	9.00E-05	3.40E-05	4.00E-05	1,000,100	49,618	1,988	1,276
41	0.00000	2.22E-03	**3.60E-05**	**0.00E+00**	**0.00E+00**	2	52	100	100	2.74E-03	2.90E-03	2.00E-05	2.30E-05	1,000,100	1,981,670	1,455	878
46	0.00000	9.95E-01	2.50E-05	**0.00E+00**	**0.00E+00**	0	6	100	100	-	6.83E-05	2.00E-05	2.50E-05	-	42,062	1,239	758
47	0.00000	7.75E-03	9.20E-05	1.12E-03	1.64E+00	1	81	35	0	7.75E-03	4.84E-04	6.62E-03	-	1,000,100	2,775,232	3,799,851	-
49	-4189.82890	-3.62E+03	**-4.19E+03**	**-4.19E+03**	-3.42E+03	0	41	32	0	-	-4.19E+03	-4.19E+03	-	-	40,401	50,023	-
52	0.00000	4.20E-05	1.50E-05	**0.00E+00**	**0.00E+00**	100	100	100	100	8.50E-05	7.20E-05	1.70E-05	2.60E-05	30,913	15,074	714	446

Dimension = 20

Pro. #.	Global min	Min				SR				MBF of successful runs				MFE of successful runs			
		SRCGA	RCGA-PS	RCGA-PS-P	RCGA-P	SRCGA	RCGA-PS	RCGA-PS-P	RCGA-P	SRCGA	RCGA-PS	RCGA-PS-P	RCGA-P	SRCGA	RCGA-PS	RCGA-PS-P	RCGA-P
39	0.00000	1.21E-02	8.20E-05	**0.00E+00**	**0.00E+00**	0	100	100	100	-	9.60E-05	3.70E-05	3.90E-05	-	1,834,080	2,044	1,252
41	0.00000	1.84E-02	8.00E-05	**0.00E+00**	**0.00E+00**	0	19	100	100	-	8.61E-04	2.20E-05	2.40E-05	-	818,737	1,448	865
46	0.00000	4.98E+00	2.98E+00	**0.00E+00**	**0.00E+00**	0	0	100	100	-	-	2.50E-05	3.10E-05	-	-	1,311	739
47	0.00000	1.12E+00	4.19E-02	3.30E-01	1.14E+01	0	0	0	0	-	-	-	-	-	-	-	-
49	-8379.65780	-6.03E+03	-8.38E+03	-8.26E+03	-5.56E+03	0	1	0	0	-	-8.38E+03	-	-	-	234,214	-	-
52	0.00000	5.90E-05	4.80E-05	**0.00E+00**	**0.00E+00**	100	100	100	100	9.20E-05	8.50E-05	2.20E-05	2.40E-05	202,580	25,961	732	448

Dimension = 30

Pro. #.	Global min	Min				SR				MBF of successful runs				MFE of successful runs			
		SRCGA	RCGA-PS	RCGA-PS-P	RCGA-P	SRCGA	RCGA-PS	RCGA-PS-P	RCGA-P	SRCGA	RCGA-PS	RCGA-PS-P	RCGA-P	SRCGA	RCGA-PS	RCGA-PS-P	RCGA-P
39	0.00000	2.65E-02	2.37E-04	**1.00E-06**	**0.00E+00**	0	100	100	100	-	6.23E-04	3.60E-05	3.60E-05	-	2,799,978	1873	1126
41	0.00000	3.96E-02	7.80E-05	**0.00E+00**	**0.00E+00**	0	39	100	100	-	2.00E-03	2.20E-05	2.40E-05	-	2,384,116	1326	807
46	0.00000	1.49E+01	9.95E+00	**0.00E+00**	**0.00E+00**	0	0	100	100	-	-	2.20E-05	2.80E-05	-	-	1212	730
47	0.00000	6.73E+00	1.28E-03	8.12E+00	2.12E+01	0	2	0	0	-	1.34E-03	-	-	-	2,800,846	-	-
49	-12569.48670	-8.56E+03	-1.21E+04	-1.23E+04	-8.13E+03	0	0	0	0	-	-	-	-	-	-	-	-
52	0.00000	7.00E-05	6.20E-05	**0.00E+00**	**0.00E+00**	100	100	100	100	9.60E-05	9.10E-05	1.50E-05	1.90E-05	762,717	41,550	700	412

INCLUDED ARTICLE

Paper [D]

A Comparative Study of Some Real Coded Genetic Algorithms for Unconstrained Global Optimization

Sawyerr, B.A., Adewumi, A.O. and Ali, M.M.

Journal of Optimization Methods and Software, *to appear*

Chapter Seven

Conclusion and Future Works

Every exit is an entrance to somewhere else

- Tom Stoppard

“Every man’s life ends the same way. It is only the details of how he lived”

- Ernest Hemingway

In order to draw some conclusions from the investigation presented in this thesis, it is important to consider our initial aims and scope of the research. We set out to test the efficiency and robustness of GA metaheuristics for real-world instance global optimization problems. One of the overriding aim was to carry out an investigation on the suitability of applying metaheuristic techniques to tackle the space allocation problem in academic institutions. We were concerned with allocating a set of entities into the available room space so that the space utilization is maximized. The emphasis was in obtaining a set of high quality (i.e. not necessarily optimal) allocations that are also structurally non-similar (i.e. diverse with respect to the solution space) so that the institution decision-makers can select the most appropriate solution.

Global optimization problems abound in many real-world instances. It is a known fact that real-world problems are characterized by real-time objectives, inconsistent constraints, optimum seeking in a changing environment and huge search space. The work in this thesis was devoted to the design and improvement of a population based method, namely GA, for solving both discrete and continuous global optimization problems. The discrete problems are real-life instances of SAP, one of which is new in literature. We develop GA metaheuristics to handle both problems. We also design some new heuristics for the HSAP and showed that they are very efficient in giving quality results.

HSAP is becoming a big concern to university authorities especially in Africa. As stated in Chapter 1, the overall aim of the work on HSAP is to investigate heuristic methods that can be used to generate automated and optimized solution in the context of the case study. Bearing this in mind, the thesis discussed several issues and potential constraints that are involved in hostel space allocation distribution. Due to the diverse manual way of handling the problem as a result of changes in personnel, we devised an abstracted and simplified version problem with the advantages of practicability and ease of implementation. This was done by identifying categories of students to be accommodated and constraints that guide their allocation at various stages. We consider this a major contribution of the thesis as this is the first reported study in literature for the problem instance to the best of our knowledge. The success record in metaheuristic application to

HSAP will opened a new door of research in this area. We therefore believe that our work, parts of which are already published, will inspire other research to do more in this area. We have presented some of the results for HSAP in this thesis at two international conferences (see [1,3]). In one of the conferences, our paper [3] was short listed as a finalist paper for the Operation Research (OR) in developing country prize [69]. We hope the work can be adapted to other instances from other institutions especially in developing countries. The deployment of a computer based solution with incorporated optimization method will help to achieve the 4-points *TREE* goals of transparency, robustness and reliability, effectiveness and efficiency. We also hope to evolve more, and possibly better, heuristics for HSAP and apply them to more instances from Institutions especially in South Africa. In one of such current study, we are adaptation a newly designed GA based on integrated crossover rule to the discrete HSAP. The new algorithm was originally designed to work for continuous problems [71].

Unlike some classical problems (such as bin packing, traveling salesman etc.) which have large benchmark data sets available in the literature, HSAP does not have benchmark data available that allow us to compare the proposed algorithm with other approaches. We also noted that it would have taken a considerable amount of work if we compared the algorithms used with every other search techniques. Moreover, the initial objective is to show the feasibility of metaheuristics approach to the problems within the context of our case study. Opportunity abounds therefore to 1) test the discrete problems with more metaheuristics and possibly hybrid techniques, 2) gather more data set from other institutions Nigeria and other developing countries, and 3) develop and formulate mathematical models for a generalized version of the HSAP as a standard benchmark problem based on fund availability.

Furthermore, we used RCGA to solve unconstrained global optimization problems with continuous variables. Two new local search heuristics based on PS and vector-projection were introduced to improve the RCGA metaheuristic. Three set of improved algorithms namely, RCGA-PS, RCGA-P and RCGA-PS-P were introduced and their results compared with standard RCGA using 57 test problems with varying dimensions.

Numerical comparisons have shown that all new algorithms are better than SRCGA with RCGA-PS-P being the best performer. RCGA-PS-P has also been shown to rank better than some obtained from literature (see Chapter 6, paper **[D]**, subsection 5.2). We therefore believe that hybridizing GA with good global and/or local exploratory heuristics will help improve it performance, efficiency and robustness in handling even difficult unconstrained global optimization problems. Our future works include the design and deployment of improvement heuristics for RCGA to handle constrained global optimization problems.

From the GA application to our instance of LTTP, we observed that the application of the metaheuristic helps to find a good quality solution as well as reduce the time to find such solution. Though, an optima solution to LTTP is always desirable, it is however ideal find a near-optima solution that reduces the amount of infeasibility in the timetable. The program developed for the LTTP can be readily scaled to a more comprehensive UTTP. This can be achieved by appending appropriate constraint data into appropriate input file created and slight modification of the program to adapt it to the new environment.

Hybridization has been one way of trying to come up with methods that are applicable to a wide range of problems. Through hybridization, a lot of methods that are more reliable can be developed. Our work with unconstrained global optimization problems shows that hybridizing GA with other heuristics will improve it robustness and efficiency. Therefore, more research is still needed in finding even more efficient global optimization methods that are applicable to a wide range of complex optimization problems. In the same vein and as future work on SAPs considered, it would be interesting to further improve the GA metaheuristics by hybridizing it with other local search heuristics in order to improve its exploration capability. In one of such work, we are trying to adapt a new integrated crossover rule developed for continuous global optimization problem into this discrete problem. It is hoped that hybridization will greatly improve the efficiency of the algorithm and solutions of the problems. Furthermore, hybridization of current metaheuristics with some exact methods, such as linear programming, branch-and bound, dynamic programming can be explored. Metaheuristics are believed to be able explore a large search space within a short time while exact methods can explore a specific small area exhaustively. Hybridization of the two may lead to a better quality solution within reasonable computational time. Furthermore, different fitness evaluation methods can be designed for the two discrete problems to assess the fitness of different individuals within the same population.

To the best knowledge of the author, this thesis presents the first investigation on the application of metaheuristic techniques to HSAP in academic institution. It is also the first investigation to LTTP within the context of our case study.   It was shown that metaheuristics can produce good solutions in much shorter time than required when constructing allocations manually. GA metaheuristics separately and reasonable adapted to all problems studied and benchmark results were provided.

The experiences gathered from this thesis can also be beneficial to research in related areas such as space planning, task allocation, car space allocation, etc. Also, the algorithms described and tested in this thesis can be the starting point for further research and for the development of a fully automated system especially for the space allocation processes considered.

Chapter Eight

Contributions

"Little drops of water make the mighty ocean and the pleasant"

- Julia Abigail Carney

"We ourselves feel that what we are doing is a just a drop in the ocean. But the ocean would be less because of that missing drop"

- Mother Theresa

"The world is moved not only by the mighty shoves of the heroes but also by the aggregate of the tiny pushes of each honest worker."

- Helen Keller

The contributions of this thesis can be summarized as follows:

- ✓ . A description and investigation into a new instance of SAP, HSAP is provided. This we believe would created avenue for future rigorous research on the application of metaheuristics to such problems
- ✓ . For the first time, an investigation on the suitability of applying metaheuristics to solve HSAP is presented. It is shown that metaheuristic approach can produce solutions of better quality than those generated manually by student affair officers and in a much shorter time.
- ✓ The design of three heuristic algorithms (*CaH1*, *CaH2* and *HaNH*) for HSAP with promising results.
- ✓ A further study to show the robustness and efficiency of GA metaheuristics in solving both discrete and continuous global optimization problems.
- ✓ The development of three new RCGAs based on local exploratory PS and global exploratory vector projection is presented. We showed that these three algorithms perform better than SRCGA. This shows that proper hybridization of GA with other heuristics can improve its performance.

This thesis reports the original ideas of the author. The Supervisor provided some general ideas that helped to refine the works and papers in both scientific and linguistic aspects. Some assistance were received from one of the author of papers **[C]** and **[D]** who was still resident (and later came for a 6-months visit) and working at the University of Lagos where data set used for the SAPs were obtained.

References

[1] Adewumi, A.O and Ali, M.M. and Ayeni, J.O.A. A multi-level genetic algorithm for a multi-stage space allocation problem. Accepted for the 8th *International Conference on the Practice and Theory of Automated Timetabling* (PATAT 2008).

[2] Adewumi, A.O. and Ali, M. A multi-level genetic algorithm for a multi-stage space allocation problem. *Mathematical and Computer Modeling*, 51(1-2), (2010), 109-126

[3] Adewumi, A.O., Ayeni, J.O.A, Fasina E.P, and Ali, M.M. (2008) A genetic algorithm metaheuristic for a multi-stage hostel space allocation problem. *Proceedings of the International Federation of Operations Research Conference (IFORS 2008), Sandton, South Africa*, 2008.

[4] Alabi, A.T. Conflicts in Nigerian Universities: causes and management. *Ilorin Journal of Education*. Faculty of Education, University of Ilorin, Nigeria 21, 2002.

[5] AlAmoush, F.A. Improving primary school timetabling using genetic algorithms. M.Sc. Thesis. Universiti Utara Malaysia, 2007.

[6] Alberto, P., Nogueira, F., Rocha, H. and Vicente, L.N. Pattern search methods for user-provided points: Application to molecular geometry problems, *SIAM Journal on Optimization, 14* (2004), 1216–1236.

[7] Aldasht, M., Alsaheb, M., Adi, S., Qopita, M.A. University course scheduling using evolutionary algorithms. *Proceedings of the 4th International Conference on Computing in the Global Information Technology, International Multi-Conference.* (2009). 47-51.

[8] Ali, M.M. and Törn, A. Population set based global optimization algorithms: Some modifications and numerical studies, *Computers and Operations Research* 31(10) (2004) 1703–1725.

[9] Ali, M.M. and Torn, A. Topographical differential evolution using pre-calculated differentials. In G. Dzemyda, V. Saltenis and A. Zilinskas, (eds.). *Stochastic and*

Global Optimization, Kluwer Academic Publishers, (2002) 1-17.

[10] Alkan, A., and Ozcan, E. Memetic algorithms for timetabling, *Proceedings of the IEEE Congress on Evolutionary Computation*. (2003) 1796-1802.

[11] Andre, D. and Astro, T. Evolving team Darwin united. In Asada, M. and Kitano, H.(eds.). *RoboCup-98: Robot Soccer World Cup II*, *Lecture Notes in Computer Science*, 1604 (1999) 346-352.

[12] Androulakis, I.P., Maranas, C.D. and Floudas, C.A. αBB: A global optimization method for general constrained nonconvex problems. *Journal of Global Optimization* 7 (4) (1995) 337-363.

[13] Annevelink, E. and Broekmeulen, R.A.C.M. The genetic algorithm applied to space allocation planning in pot-plant nurseries. XII International Symposium on Horticultural Economics *Acta Horticulture (ISHS)* 340 (1995) 141-148.

[14] Bai, R, *An Investigation of Novel Approaches for the Optimizing Retail Shelf Space Allocation.* PhD Thesis, University of Nottingham, 2005.

[15] Bai, R. Burke, E.K. and Kendall, G. Heuristic, meta-heuristic and hyper-heuristic approaches for fresh produce inventory control and shelf space allocation. *Journal of the Operational Research Society* (2007), 1-11

[16] Bareither, H.D. and Schillinger, J.L. University Space Planning: Translating the Educational Programme of a University into Physical Facility Requirements. Technical Paper ED029467, University of Illinois, , Urbana, 1969.

[17] Beasley, J.E., Sonander, J. and Havelock, P. Scheduling aircraft landings at London Heathrow using a population heuristic. *Journal of the Operational Research Society*, 52(5) (2001) 483-493.

[18] Benjamin, C.O., Ehie, I.C. and Omurtag, Y. Planning facilities at the University of Missouri-Rolla. *Interfaces*. 22 (4) (1992) 95-105

[19] Bergey, P.K. and Ragsdale, C. Modified differential evolution: a greedy random strategy for genetic recombination. *Omega 33*, 3 (2005), 255 - 265.

[20] Bertsekas, D.P. *Nonlinear Programming,* 2nd Edition. Athena Scientific, 1999.

[21] Box G.E.P. Evolutionary operation: A method for increasing industrial productivity. *Applied Statistics.* 6 (1957) 81-101.

[22] Boyd, S. and Vandenberghe, L. *Convex Optimization.* Cambridge University

Press, 2004 .

[23] Brandimarte, P. *Numerical Methods in Finance and Economics - A Matlab-based Introduction.* 2nd Edition, John Wiley & Sons. 2006.

[24] Buczak, L., Wang, H., Darabi, H. Jafari, M.A. Genetic algorithm convergence study for sensor network optimization, *Information Sciences* 133(3) (2001) 267-282.

[25] Burke E.K and Varley D.B. Automating space allocation in higher education. Selected Paper from the 2nd Asia Pacific Conference on Simulated Evolution and Learning SEAL 98, *Lectures Notes in Artificial Intelligence*, 1998, 1585:66-73.

[26] Burke E.K, Kendall, G. and Soubeiga, E. A tabu search hyper-heuristics for timetabling and rostering. *Journal of Heuristics*, 9 (6) (2003) 451-470.

[27] Burke, E.K. and Newall, J.P. A multi-stage evolutionary algorithm for the timetable problem. *IEEE Transactions on Evolutionary Computation*, 13(1):63-74, Apr 1999.

[28] Burke, E.K. and Varley, D.B. Space allocation: An analysis of higher education requirements. In: Burke, E.K. and Carter, M.W. (eds.) *Lecture Notes in Computer Science*, 1408 (1998), 20-33.

[29] Burke, E.K., Beyrouthy, C., Landa-Silva, J.D., McCollum, B. and McMullan, P. SpaceMAP: Applying meta-heuristics to real world space allocation problems in academic institutions. In: *Proceedings of the 2004 International Conference on the Practice and Theory of Automated Timetabling (PATAT),* Pittsburgh USA, (2004) 441–456.

[30] Burke, E.K., Cowling, P. and Landa-Silva, J.D. and Petrovic, S. Combining hybrid metaheuristics and populations for the multiobjective optimization of space allocation problems, *Proceedings of the 2001 Genetic and Evolutionary Computation Conference (GECCO)* (2001) 1252-1259

[31] Burke, E.K., Cowling, P. and Landa-Silva, J.D. Hybrid population-based meta-heuristics approaches for the space allocation problem. *Proceedings of the Congress on Evolutionary Computation*. (2001) 232-239.

[32] Burke, E.K., Cowling, P., Landa-Silva, J.D., McCollum, B. Three methods to automate the space allocation process in UK universities. *Lecture Notes in*

Computer Science 2079 (2001) 254-273.

[33] Caserta, M. and Voß, S. Metaheuristics: intelligent problem solving. In: Maniezzo, V., Stützle, T., Voß, S. (eds.). *Matheuristics: Hybridizing Metaheuristics and Mathematical Programming - Annals of Information Systems*. Springer. 2009 1-32.

[34] Chandrakaisan S.D. The optimum combination of local searches for genetic operators in memetic algorithm for the space allocation problem. M.Sc. Thesis. University of Malaysia, 2008.

[35] Charles, A. and Dennis, E.J. Analysis of generalized pattern searches. *SIAM Journal on Optimization* 13 (3) (2003) 889–903.

[36] Cherkaev, A. Course note on methods of optimization. Online available at *http://www.math.utah.edu/~cherk/teach/opt/2009.html*

[37] Chinneck, J.W. *Practical Optimization: A Gentle Introduction*. Online Reading Material. Systems and Computer Engineering, Carleton University, Canada. Available at http://www.sce.carleton.ca/faculty/chinneck/po.html.

[38] Cioppa, A.D., Stefano, C.D., Marcelli, A. On the role of population size and niche radius in fitness sharing. *IEEE Transactions on Evolutionary Computation*, 8 (6) (2004) 580-592.

[39] Coello, C.A.C., Twenty years of evolutionary multi-objective optimization: A historical view of the field. *IEEE Computational Intelligence Magazine*. 1 (2006) 28-36.

[40] Coffman, E. G., Garey, M. R. and Johnson, D. S. Approximation algorithms for bin packing: A survey. In: Hochbaum, D.S (ed), *Approximation Algorithms for NP-Hard Problems*, PWS Publishing, 1997, pp. 46-93.

[41] Cowling, P., Kendall, G. and Soubeiga, E. A hyper-heuristic approach to scheduling a sales summit. In: Burke, E.K. and W. Erben, E. (eds.) *Lecture Notes in Computer Science*, Springer, (2001) 176-190.

[42] Dantzig, G.B. *Linear Programming and Extensions*. Princeton University Press, Princeton, NJ. 1963

[43] Darwin. C. *On the Origin of Species*, 6th edition, John Murray, 1859.

[44] Dean, J.S. Staff scheduling by a genetic algorithm with a two-dimensional

chromosome, In Burke, E.K. and Gendreau, M. (Eds.), *Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling – PATAT 2008*, Montreal, Canada, 2008, 15 pgs

[45] Dennis, J. E. and Torczon, Virginia J. Derivative-free pattern search methods for multidisciplinary design problems. *Proceedings of the 5th AIAA/ USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Florida. 1994.

[46] Donald, K.L. and Douglas, S.R. *Combinatorial Algorithms: Generation, Enumeration, and Search.* CRC Press, Florida, 1999.

[47] Dowsland, K.A. Simulated annealing. In: Reeves, C.R. (ed). *Modern Heuristic Techniques for Combinatorial Problems.* McGraw-Hill, 1995.

[48] Fogel, D.B., Fogel, G.B. and Ohkura, K. Multiple-vector self-adaptation in evolutionary algorithms, *BioSystems*, 61(2-3) (2001) 155-162.

[49] Gabere, M.N. *Simulated Annealing Driven Pattern Search Algorithms for Global Optimization.* M.Sc. Thesis, University of the Witwatersrand, 2005.

[50] Gabrani, G. Bhargava, P., Bhawana, Gill, G.S. Use of genetic algorithms for Indian music mixing, *ACM Ubiquity*, 9 (10) (2008) 11-17.

[51] Gage, R.L. and Gates, R.L. Technical challenges in the implementation of a space management system. Technical Report UC1170, NASA Research Center. http://proceedings.esri.com/library/userconf/proc05/papers/pap1170.pdf

[52] Gagliardi, J-P, Ruiz, A. and Renaud, J. Space allocation and stock replenishment synchronization in a distribution center. *International Journal of Production Economics*, 115 (2008) 19– 27

[53] Gendreau M. An introduction to tabu search. In: Glover F. & Kochenberger G.A (eds.). *Handbook of Metaheuristics.* Kluwer Academic Publisher, Boston, 2003 37-54.

[54] Ghaemi, S., Vakili, M.T., and Aghagolzadeh, A. Using a genetic algorithm optimizer tool to solve university timetable scheduling problem. *Proceedings of the 9th International Symposium on Signal Processing and Its Applications* (ISSPA 2007) (2007) 1-4.

[55] Giannikos, I. El-Darzi, E. and Lees, P. An integer goal programming model to

allocate offices to staff in an academic institution. *Journal of the Operational Research Society*, 46 (6) (1995) 713-720.

[56] Glover F & Kochenberger G.A (eds.). *Handbook of Metaheuristics.* Kluwer Academic Publisher, Boston, 2003.

[57] Glover F and Laguna M. Tabu Search. In**:** Reeves, C.R**. (**ed**)** .*Modern Heuristic Techniques for Combinatorial Problems***.** McGraw-Hill, 1995 70-150.

[58] Glover F and Laguna, M. *Tabu Search.* Kluwer Academic Publishers, 1997.

[59] Glover, F. (1986). "Future paths for integer programming and links to artificial intelligence." *Computers & Operations Research* 13, 533–549.

[60] Goldberg, D.E. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, 1989.

[61] Gyõri,S., Petres, Z., Várkonyi-Kóczy, A. R. Genetic algorithms in timetabling. A new approach, Technical paper, 2001. http://www.mft.hu/hallg/200107.pdf

[62] Hartl, R.F. A global convergence proof for a class of genetic algorithms. Technical Paper, Univerity of Technology, Vienna, 1990

[63] Hill T., Lundgren A., Fredriksson R., Schiöth H.B. Genetic algorithm for large-scale maximum parsimony phylogenetic analysis of proteins. *Biochimica et Biophysica Acta* 1725 (2005) 19–29.

[64] Himmelblau, D.M. *Applied Nonlinear Programming*, McGraw-Hill, 1972.

[65] Hoadley, J. A., and Anderson, J. I. Space Projection Model for Academic Departments, Technical Paper, R.M.I.T, Melbourne, 1973.

[66] Holland, J. H. *Adaptation in Natural and Artificial Systems.* Cambridge, MA: MIT Press, 1992.

[67] Holland, J.H. *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, 1975.

[68] Hooke, R. and Jeeves, T.A. Direct search solution of numerical and statistical problems, *Journal of the Association for Computing Machinery*, 8(2) (1961) 212- 229.

[69] Ittmann, H.W (ed.) International Federations of the Operational Research Societies (IFORS) Newsletter 2(1) (2008) pg. 13.

[70] Kaelo P. *Some Population Set-based Methods for Unconstrained Global*

Optimization. Ph.D Thesis, University of the Witwatersrand, 2005.

[71] Kaelo, P. and Ali, M. M. Probabilistic adaptations of point generation schemes in some global optimization algorithms. *Optimization Methods and Software*. 21(3) (2006) 343-357

[72] Kall, P. and Wallace, S.W., *Stochastic Programming.* John Wiley & Sons, Chichester, 1994

[73] Kirkpatrick, S., Gellat, C. and Vecchi, M. Optimization by simulated annealing. *Science*, 220 (1983) 671–680

[74] Kolda, T.G., Lewis, R.M. and Torczon, V. Optimization by direct search: New perspective on classical and modern methods. *SIAM Review, 45(3)* (2003), 385–482.

[75] Lalescu L. and Badica C. Timetabling experiments using genetic algorithms. In *Proceedings of the International 12th Turkish Symposium on Artificial Intelligence and Neural Networks (TAINN-2003),* Canakkale, Turkey, 2003.

[76] Landa-Silva, J.D. and Burke, E.K. Asynchronous cooperative local search for the office space allocation problem. *INFORMS Journal on Computing*, 19(4) (2007) 575-587.

[77] Landa-Silva, J.D. *Metaheuristics and Multiobjective Approaches for Space Allocation*. PhD Thesis. University of Nottingham, 2003.

[78] Lawler, E.L., Lenstra, J.K., Rinnooy-Kan, A.H.G. and Shmoys, D.B. (eds.). *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization.* John Wiley and Sons, 1985.

[79] Lee, J. *A First Course in Combinatorial Optimization*, Cambridge University Press; 2004

[80] Levy, A.B. *The Basics of Practical Optimization. SIAM, Philadelphia, PA,* 2009.

[81] Louis, S.J. and Rawlins, G.J.E. Predicting convergence time for genetic algorithms. Technical Report 370, Computer Science Department, Indiana University 1993.

[82] M&G. Varsities run out of housing. Mail & Guardians (South African) Newspaper article on higher learning, September 23, 2009 pg 31.

[83] Marczyk, A. Genetic algorithms and evolutionary computation. The TalkOrigin

Archive. Available at http://www.talkorigins.org/faqs/genalg/genalg.html

[84] Martello, S. and Toth, P. *Knapsack Problems - Algorithms and Computer Implementations*. Wiley, 1990.

[85] Michalewicz, Z. and Fogel, D.B. *How to Solve It: Modern Heuristics.* 2nd, Revised and Extended Edition, Springer, 2004.

[86] Mitchell, M. *An Introduction to Genetic Algorithms. The MIT Press, 1998.*

[87] Mitchell, M. Genetic algorithms: An overview. *Complexity*, 1 (1) 1995) 31-39.

[88] Musgrove, J. Space utilization in universities and polytechnics. *University building notes, Design notes No. 12*, Department of Education and Science and Great Britain University Grants Committee, 1972.

[89] Naso, D., Surico, M., Turchiano, B. and Kaymak, U. Genetic algorithms for supply-chain scheduling: A case study in the distribution of ready-mixed concrete, *European Journal of Operational Research* 177 (2007) 2069–2099.

[90] Osman, I.H. and Kelly, J.P. (eds.). *Meta-Heuristics: Theory and Applications*. Kluwer Academic Publishers, Boston, 1996.

[91] Ozcan, E. and Alkan, A. Timetabling using a steady state genetic algorithm, *Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling (PATAT 2002),* (2002) 104-107.

[92] Polya, G. *How to solve it*. Princeton University Press, Princeton, 1945.

[93] Powell M,J. Direct search algorithms for optimization calculations. *Acta Numerica*, 1998, 7: 287-336.

[94] Price K. An introduction to differential evolution. In: Corne, D., Dorigo, M. and Glover, F. (eds.) *New ideas in optimization*, McGraw-Hill, Cambridge, 1999, 79.108.

[95] Radda, S.I. Unethical practices in the Nigeria's University system: Pattern, causes and solutions. *Business Ethics Network of Africa (BEN-Africa) Conference Proceedings* Ghana, 2009.

[96] Resende, M.G.C. and de Sousa, J.P., (eds.) *Metaheuristics: Computer Decision-Making*. Kluwer Academic Publishers, 2003.

[97] Ritzman, L., Bradford, J., and Jacobs, R. A multiple objective approach to space planning for academic facilities. *Management Science* 25 (9) (1979) 895-906.

[98] Rudolph, G. Convergence analysis of canonical genetic algorithms. *IEEE Transactions on Neural Networks*, 5(1) 91994) 96-101.

[99] Rudolph, G. Self-adaptive mutations may lead to premature convergence. *IEEE Transactions on Evolutionary Computation*, 5(4) (2001) 410-414.

[100] Sahinidis, N.V. BARON: A general purpose global optimization software package. *Journal of Global Optimization* 8 (2) (1996) 201-205

[101] Salikon, M.Z.B.M. Examination timetabling using genetic algorithms – Case study: Kuittho. M.Sc. Thesis. Universiti Utara Malaysia, 2005.

[102] Shabha, G. An assessment of the effectiveness of e-learning on university space planning and design. *Facilities.* 22 (3-4) (2004) 78-86.

[103] Shabha, G. Virtual universities in the third millennium: an assessment of the implications of teleworking on university buildings and space planning. *Facilities.* 18 (5-6) (2000) 235 -244.

[104] Sharapov, R.R and Lapshin, A.V. Convergence of genetic algorithms. *Pattern Recognition and Image Analysis,* 16 (3) (2006) 392–397.

[105] Sharma, R. D. Academic staff and space allocation models for Australian CAEs, R.M.I.T., 1982.

[106] Sharma, R.D. and Kumar, S. Space allocation to academic departments in a high rise building of an Australian educational institution. *Research in Higher Education*, 23(1) (1985) 86-95

[107] Sivanandam, S. N., and Deepa, S. N. *Introduction to Genetic Algorithms,* Springer, 2007.

[108] Spencer, H. *The Principles of Biology*. Reproduced by the University Press of the Pacific, 2002.

[109] To C.C. and Vohradsky J. A parallel genetic algorithm for single class pattern classification and its application for gene expression profiling in Streptomyces coelicolor. *BMC Genomics* 8 (2007) 49

[110] Torczon V, Trosset M. W. From evolutionary operation to parallel direct search: Pattern search algorithms for numerical optimization. *Computational Science and Statistics*, 1998, 29: 396-401.

[111] Torczon, V. *Multi-Directional Search. A Direct Search Algorithm for Parallel*

Machines, Ph.D. Thesis, Rice University, Houston, 1989.

[112] Torczon, V. On the convergence of pattern search algorithms. *SIAM Journal on Optimization*, 12(4) (1997) 1075–1089.

[113] Vanderplaats, G. *Numerical Optimization Techniques for Engineering Design: With Applications*. McGraw-Hill, Inc, 1984.

[114] Varaiya, P. *Lecture Notes on Optimization*. Re-issue of Notes on Optimization, Van Nostrand-Reinhold, 1972.

[115] Vesterstrøm J.S and Riget J. Particle swarms: Extensions for improved local, multi-modal, and dynamic search in numerical optimization. M.Sc. Thesis submitted to the Department of Computer Science, University of Aarhus, 2002.

[116] Wai-Ho, A., Chan, K. and Yao, X. A novel evolutionary data mining algorithm with applications to churn prediction. *IEEE Transactions on Evolutionary Computation*, 7(6) (2003) 532-545.

[117] Weise T. *Global Optimization Algorithms - Theory and Application*. Self-published electronic book, 2006–2009, *http://www.it-weise.de/projects/book.pdf*

[118] Weise, T., Zapf, M., Chiong, R. and Nebro, A.J. Why is optimization difficult? In R. Chiong (ed.), *Nature Inspired Algorithms for Optimization, Studies in Computational Intelligence Series*, Springer, 193 (2009), pp. 1-50.

[119] Westerlund T. and Pettersson F. An extended cutting plane method for solving convex MINLP problems. *Computer & Chemical Engineering*. 19 (1) (1995) 131-136

[120] Wetter, M. and Polak, E. A convergent optimization method using pattern search algorithms with adaptive precision simulation. *Eighth International IBPSA Conference* (2003) 1398 – 1400.

[121] Wikipedia. Optimization (mathematics). Available at http://en.wikipedia.org/wiki/Optimization_mathematics

[122] Wren, A. Scheduling, Timetabling and rostering: A special relationship? In: Burke, E.K., and Ross, P. (eds.) *Lecture Notes in Computer Science*, Springer, 1153, (1996) 46-75.

[123] Wu, X., Sharif, B.S., Hinton, O.R. An improved resource allocation scheme for plane cover multiple access using genetic algorithm. *IEEE Transactions on*

Evolutionary Computation. 9 (1) (2005) 74–80.

[124] Yang X.S. *Introduction to Mathematical Optimization: From Linear Programming to Metaheuristics*, Cambridge Int. Science Publishing, 2008.

[125] Yang, M.-H An efficient algorithm to allocate shelf space. *European Journal of Operations Research*, 131 (2001) 107-118.

[126] Yin, P., Yu, S., Wang, P., Wang, Y. Task allocation for maximizing reliability of a distributed system using hybrid particle swarm optimization. *Journal of Systems and Software*, 80(5) (2007), 724-735.

[127] Zegordi, S. H., Abadi, I. N.K. and Beheshti Nia, M. A. A novel genetic algorithm for solving production and transportation scheduling in a two-stage supply chain Export. *Computers & Industrial Engineering*. In press, 2009.

Appendix A

Remi Adewumi <laremtj@gmail.com>

Optimization Methods and Software - Decision on Manuscript ID GOMS-2009-0069.R1

1 message

olbur@mai.liu.se <olbur@mai.liu.se> **Wed, May 5, 2010 at 5:45 PM**
To: laremtj@gmail.com

05-May-2010

Dear Dr Adewumi:

Ref: A Comparative Study of Some Real Coded Genetic Algorithms for Unconstrained Global Optimization

This is just to let you know that the final version of your paper will be forwarded to the production department for publication in Optimization Methods and Software.

You will receive proofs for checking, and instructions for transfer of copyright in due course.

The publisher also requests that proofs are checked and returned within 48 hours of receipt.

Thank you for your contribution to Optimization Methods and Software and we look forward to receiving further submissions from you.

Sincerely,
Oleg Burdakov
Editor-in-Chief, Optimization Methods and Software

This journal is participating in the PEER project (http://www.peerproject.eu/), which aims to monitor the effects of systematic self-archiving (author deposit in repositories) over time. PEER is supported bytheECeContentplusprogramme (http://ec.europa.eu/information_society/activities/econtentplus/index_en.htm). As your manuscript has been accepted for publication by Optimization Methods and Software, you may be eligible to participate in the PEER project. If you are based in the European Union, you are hereby invited to deposit your accepted manuscript in one of the partici¬pating PEER repositories. You may also choose to deposit in a non-PEER, institutional or subject repository in addition to, or as an alternative to deposit in a PEER designated repository. If depositing your accepted manuscript in a non-PEER repository, please set an embargo period of 12 months from the date of publication of the journal article for the public release of your accepted manuscript. For further information on PEER deposit, non-PEER deposit and embargo periods please visit the PEER Helpdesk: http://peer.mpdl.mpg.de/helpdesk.

Subject: Journal Submission System - Paper Converted
From: <parry@orsoc.org.uk>
Date: Fri, 23 Oct 2009 10:45:03 +0100
To: <laremtj@gmail.com>

Thank you for submitting your paper to the Journal of The OR Society (JORS). Your document has been successfully converted by the system and will shortly be assigned to an editor. The details of your paper are listed below for confirmation purposes. These details include a paper identification code, which you may use at any time to check the progress of your paper by simply accessing the URL -

http://www.orsoc.org.uk/jorssubmission/submission/papertrack.asp

- and entering your identification code into the field provided.

The editor will now review your paper to assess its appropriateness for publication in JORS, after which, if appropriate, the paper will be sent to referees for peer review. Once the editor has considered the referees' reports and arrived at a decision, I will be in touch once again.

To enable me to keep in contact with you during the publication process, please notify me of any changes in your contact details.

Kind regards
Sarah Parry
Editorial Administrator.

parry@orsoc.org.uk

Submitted Paper Details

Paper Title: A Hierarchical Heuristic Strategy for Hostel Space Allocation Problem (Paper ID - 11311)
Paper Author(s): Aderemi Adewumi and Montaz Ali
Paper Tracking Code: ORJSSHCBHTR

Office Use Only: JSSFEDCC9709

This message has been scanned for malware by SurfControl plc. www.surfcontrol.com