

https://dspace.org/

Some improved genetic-algorithms based heuristics for global optimization with innovative applications

Adewumi, Aderemi Oluyinka

2010-09-07
http://hdl.handle.net/10539/8621

Downloaded from a DSpace repository. https://dspace.org/

Mathematical and Computer Modelling 51 (2010) 109–126

Contents lists available at ScienceDirect

Mathematical and Computer Modelling

journal homepage: www.elsevier.com/locate/mcm

A multi-level genetic algorithm for a multi-stage space allocation problem

A.O. Adewumi [a,*], M.M. Ali [b]

[a] School of Computer Science, University of Kwazulu-Natal, Durban, South Africa

[b] School of Computational & Applied Mathematics, University of the Witwatersrand, Johannesburg, South Africa

ARTICLE INFO

Article history:
Received 28 February 2009
Received in revised form 28 August 2009
Accepted 29 September 2009

Keywords:
Space allocation
Hostel space allocation problem
Combinatorial optimization
Metaheuristics
Genetic algorithm

ABSTRACT

A new case of space allocation problem is considered. The study is based on a real-world multi-stage hostel space allocation for university students. A multi-level application of genetic algorithm metaheuristic with promising results is presented. Based on the case study, we examined the sensitivity analysis of various genetic algorithm operators in order to establish the baseline for practical deployment. The feasibility rate of the solutions obtained were also determined and presented.

© 2009 Elsevier Ltd. All rights reserved.

1. Introduction

Space allocation problem (SAP) is a domain specific combinatorial optimization problem (COP) that has attracted attention among researchers recently [1,2]. It is defined as the distribution of available areas of space of different sizes among a number of objects so as to ensure optimal space utilization and satisfaction of additional requirements and/or constraints [3]. The problem is similar to scheduling problems such as academic timetabling [4,5], open shop scheduling problem [6] with the associated resource efficiency issues. SAP impacts on various institutions ranging from companies, organizations to educational institutions. SAP in a higher institution context is defined as the allocation of entities (for example, staff, students, laboratories, lecture rooms) to available room space in order to satisfy as many requirements and constraints as possible [2]. Space allocation is a dynamic and continual exercise [7] and is carried out manually in most cases with the resultant time consumption and inefficiency [8]. In real-life cases with large problem instances therefore, the use of proven optimization techniques become inevitable.

In this paper, we handle a new class of SAP namely, the hostel space allocation problem (HSAP). HSAP arises in a tertiary institution from the need to optimize the allocation of available bed spaces in halls of residence (hostels) among a large number of demanding and eligible students based on a set of varying (and sometimes conflicting) requirements and constraints. The complexity of the problem emanates from the limited number of bed spaces compared with the large number of demanding students and the nature of the constraints imposed on the allocation process. HSAP, though a new case of SAP, is NP-hard like other SAPs [1,2]. The best practice for handling the problem in most Nigerian universities is the use of database or spreadsheet software for record maintenance without any recourse to known scheduling or an optimization algorithm that determines an optimal allocation. This makes the allocation process inefficient, time-consuming and unreliable due to human factors.

* Corresponding address: University of Kwazulu-Natal, School of Computer Science, Private Bag X54001, 4000 Durban, South Africa. Tel.: +27 31 2601176; fax: +27 31 2607001.
E-mail address: laremtj@gmail.com (A.O. Adewumi).

0895-7177/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.mcm.2009.09.004

Specifically, HSAP refers to the distribution of bed spaces in various hostels among different category of students so as to ensure optimal space utilization and the satisfaction of additional requirements and/or constraints. The number of students in each category varies just as the number of bed spaces varies from one hostel to the other. In this paper, we employ a multi-level heuristic driven by genetic algorithm (GA) to solve the HSAP under domain specific constraints. The study is based on a real-life multi-stage case of HSAP with data set obtained from one of the largest university (in term of student population and research) in Nigeria. Simulation was performed using the data set and result obtained is presented.

The rest of the paper is sub-divided as follows: Section 2 presents further introduction to SAP with specific and detailed description of HSAP as obtained in the case study. The motivation for the study is also presented in this section. The solution methodology based on GA is discussed in Section 3 while the results of simulation experiments are presented in Section 4. The paper ends with some concluding notes in Section 5.

2. The space allocation problem

SAP seeks to allocate given entities into available *scarce* resources while satisfying given requirements and constraints. It is a constrained multi-objective combinatorial optimization problem (COP) similar to the bin-packing and knapsack problems [3,9]. In higher institution context, this class of COPs varies greatly among institutions and requires frequent modifications due to the addition and/or removal of entities. It also has direct impact on the functionality of institutions [2,7]. Exact methods had earlier been applied to various (and often smaller) instances of real-life SAP. These include the use of mixed-integer goal programming [10], linear programming [11], and integer goal programming [12]. Like many other NP-hard COPs such as knapsack problem [13] and bin-packing problem [14], exact algorithm for SAP has exponential time complexity [2]. Handling the problem with relatively large instance will be computationally expensive hence the need to resort to approximation techniques which can solve it to a satisfactory solution quality level within reasonable computational time. Hence, Burke and Varley [7] suggested the application of heuristics to SAP.

For large instances of SAP, heuristics and their variants had been applied successfully of late. Bai [1] investigated the application of simulated annealing, GRASP and other metaheuristics to real-world shelf SAP. This problem arises from the availability of limited shelf space and large number of products to be displayed. Similarly, Landa-Silva [2] investigated the application of tabu search (TS) and GA metaheuristics to real-life instances of office SAP. These studies reveal, among other, (1) the multi-objective nature of SAP, (2) the complexity of the problem due to conflicting objectives and constraints, and (3) the difficult nature of real-world SAP. Experimental results obtained in their work proved the efficiency of metaheuristics in handling SAP. In this paper, we consider the problem of allocating bed spaces to undergraduate students in tertiary institution. This is a pioneer study with respect to both the instance of SAP and the application of metaheuristics to this instance. To the best of our knowledge, this is first application of metaheuristic to the problem in the context of developing countries. Ironically, the current manual approach being used in the case study neither preserve nor provide sufficient data as regard yearly generated allocation distribution on which our study could be benchmarked. Our goal therefore is to set the baseline and pioneer study for the practical automation of the allocation of students into hostels in order to achieve the **TREE** goals for the system namely, Transparency, Robustness, Effectiveness and Efficiency.

2.1. Basic notions

1. Bed space: A space created in the hostel to accommodate a single student.
2. Hall (Hostel) (H) : A building meant to accommodate a number of students located at various zones within the campus. Each hall has a predefined capacity and consists of set of rooms which are located at various floors and block (wings). Note that the terms hall and hostel can be used interchangeably.
3. Students (S): The number of students that must be accommodated in a given category.
4. Capacity: the maximum number of students that can be accommodated in a given entity (e.g. hall, floor, room). This is calculated as the total number of bed spaces in that entity.
5. Category (C): A grouping into which eligible *students* are divided for allocation purpose. Each category has varying number of *students* with varying degree of allocation priority.
6. Allocation (A): The assignment of students in a given category into an allocation entity (hall, block, floor). At the Floor/block level, it refers to the number of students allocated to the floor/block.
7. Room Capacity (R): Total number of bed spaces in a given room within a given hall.
8. Block Capacity (B): Summation of R for a given block in a given hall.
9. Floor Capacity (F): Summation of all B for a given floor in a given hall.
10. Hall Capacity (H): The capacity of all F within a given hall.

2.2. HSAP – background

Most universities in Nigeria are owned by the federal or state (provincial) governments with some private universities springing up of late. Apart from academic competence, one of the factors that influence the choice of university prospective students is the availability of residential accommodation (preferably on-campus). Most first generation universities (in terms

of relative date of establishment) have on-campus residential accommodation for students. However, given the current increase in the number of applicants and subsequent undergraduate intakes, residential accommodation is fast becoming a scarce resource. Some university can only accommodate less than twenty percent of the student population on their hostel facilities [15]. Given other administrative considerations such as security, it became imperative therefore to find an innovative way of allocating students to the scarce resource so as to help fulfil the academic, research, and other missions of the institutions. This brings about the HSAP as a new case of SAP. The increased number of requests for bed space and limited number of space available calls for an optimal allocation strategy. HSAP can thus be seen as the process of allocating scarce resources (bed spaces) to large number of competitive 'customers' (students) under given hard and soft constraints. This process is carried out every session.

The institution under study currently has a student population of over 35,000 spread over various faculties with twelve main halls of residence on the main campus to accommodate undergraduate students. These facilities consist of six male and six female hostels. The hostels are zoned based on their physical location (see Table 1). There is no mixed allocation of male and female undergraduate students in the same hostel hence there are designated hostels for male and female. Hostels are divided into blocks (wings) located on various floors of the building (see Appendix A). A room in a typical hostel is identified with a label *XAB*,where *X* represents the block label that the room falls, *A* is a 1-digit number referring to the floor number while *B* is a 2-digit code referring to the room number. The rooms have varying capacity depending on the number of bed space therein. The current capacity for the hostels is shown in Table 1.

The allocation process is multi-stage involving (1) the allocation of various categories of students to hall by the office of the Dean of Student Affairs (DSA) based on certain constraints and requirements, and (2) allocation of students under each category by the Hall Manager to a particular room in a block and floor based on another set of constraints and requirements. The bed spaces are then shared among the allocated students to a room. The physical allocation process involves (1) completion and submission of application form by interested students, (2) sorting, validation, categorization and data entry of applications by the office of the Students Affairs, (3) generation of hostel allocation list by the Students Affairs office, (4) releasing of the lists showing hall of residence, and (5) allocation to rooms by the Hall Manager.

Presently, eight categories of students are identified with differently assigned priority. They are (1) Final year students (Fy), (2) Scholars — those with cumulative grade point average (CGPA) $\geq$4.20 of 5.00 scaling (Sc), (3) Foreign students (Fo), (4) Health students (Ht) — physically-challenged students or those with peculiar health conditions, (5) Fresher — first year and direct entry students (Fr), (6) Sports men and women (Sp), (7) Discretionary (Ds) — based on peoples' requests, and (8) Others (Ot) — students at other levels who desire to be accommodated. Priorities are assigned based on administrative consideration by the institution authority. Some considerations include the need for final year students to concentrate on their projects work, physically-challenged students to be accommodated close to the health care service provider (medical centre) and scholars being accommodated to encourage them to maintain the academic status, etc.

2.2.1. Constraints

The HSAP as an instance of COP has its constraints imposed by the requirements of the system. These constraints need to be satisfied in order to achieve the overall goal of the allocation process and that of the institution's authority. In this paper, we classify the constraints into two namely, hard and soft constraints. Hard constraints are to be strictly enforced during the allocation process while soft constraints may be slacked but sometimes with necessary penalty. The different set of constraints that exist at different stages of the allocation process falls under these two categories.

In the current study, hard constraints are more prominent as a result of some administrative considerations mentioned earlier and thus must be strictly enforced. A major consideration is ease of accessibility to required facilities by certain categories of students. For example sport men and women must be accommodated close to the sport centre. Thus certain categories of students must be allocated to specified halls. The general breakdown of this as obtained in the university is presented in Table 2. It should be noted that male and female allocations are mutually exclusive, thus the same process and experiment are considered separately for each.

Other requirements classified as hard constraints are: (1) all Fo category must be accommodated, (2) allocation of all Ht due to their health status, (3) Ht must be allocated to the lowest floor possible in the given hall, (4) all Sp should be allocated and (5) all allocated Fy should be allocated to the highest floor possible in the given hall. Other requirements considered as soft constraints in the HSAP are: (1) as many Fy as possible are offered accommodation, and (2) as many Sc as possible should be offered accommodation. It should be noted that though the allocation of Fy is considered soft constraint in terms of *students*allocated yet its takes pre-eminence over other allocations (with the exception of Ht, Sp and Fo) according to the assigned allocation priority. The order of priority, which is another constraint, is (1) Fy, (2) Sc, (3) Fr, (4) Ds and (5) Ot. These constraints applies at different levels of category, hall and block/floor allocations.

The quality of an allocation (*solution*) at any of the stages in HSAP can be measured in terms of (1) satisfaction or non-violation of given hard constraints, (2) *Students* allocated under each category, (3) space utilization factor, and (4) satisfaction of any additional requirements (soft constraints). We define *space utilization factor* (U) as a measure of space usage. This is taken as the average of the allocated space (A) over the available space when considering a given entity i.e.

$$U = \frac{\text{Allocated Space } (A)}{\text{Capacity}} \tag{1}$$

Table 1
Capacities of halls of residence (source dsa office, University of Lagos).

Zone (Area)	Hall name	Sex	Hall capacity	Total capacity	
				Female	Male
A (Main campus)	Jaja	Male	660	866	1104
	Mariere	Male	444		
	Moremi	Female	866		
B (New hall)	Eni Njoku	Male	800	1564	1768
	Aliyu Makama Bida	Female	764		
	Fagunwa	Female	276		
	Madam Tinubu	Female	524		
	Sodeinde	Male	968		
C (Gate/Education)	El-Kanemi	Male	526	1158	1038
	Kofo Ademola	Female	512		
	Queen Amina	Female	646		
	Saburi Biobaku	Male	512		
Grand total capacity				3588 7498	3910

Table 2
Specified halls for certain categories.

Category	Specified halls	
	Male	Female
Ht	Jaja	Moremi
Sc	Mariere	Moremi
Sp	El-Kanemi	Queen Amina

with expected value in the range $0 \leq U \leq 1$. A value of $U > 1$ implies *space overuse*. Space overuse can arise sometimes due to the effect of the crossover operator of GA but a repair algorithm is used to correct this (see Section 3.3). Eq. (1) is a general definition of utilization factor which can be appropriately indexed when used in connection with different levels of the allocation process.

In a general sense, an optimal solution in SAP would mean an allocation in which all hard constraints are satisfied with no space overused and other additional requirements (constraints) are satisfied. However, this ideal solution is not always achievable with the application of heuristics [2]. In a more realistic scenario therefore, an optimal solution would be one where all entities are allocated and space utilization is the best possible, i.e. the amount of space wasted and overused has been reduced to the lowest possible and additional requirements and constraints are satisfied [2].

For allocation purpose, the data set consisting of the capacity of each hostel, floor within the hostels, and the number of eligible (determined by the students' affair section) students under each category are necessary. An instance of real-life data set obtained for the current study is shown in Appendix A.

2.2.2. Solution stages

This paper is basically concerned with the allocation of various categories of students into hostels in order to maximize bed space utilization and satisfy specified hard and soft constraints. The initial stage of the allocation seeks to determine the number of students in each category to be allocated that will not exceed the total capacity of available hostels. At this stage, the fixed-choice allocation of all applicants in the specified categories (Ht, Sp and Fo) are given priority. We then use a greedy approach to determine the number of applicants in the flexible-choice allocation of the remaining five categories (Fy, Sc, Fr, Ds and Ot) based on the given allocation priority.

The remaining two stages involve the use of the category allocation to distribute the students into various hostels and block/flock respectively under associated constraints. We use GA metaheuristic successively to handle the allocation at these two stages. Thus the execution of the algorithm at a stage depends on the result of the previous stage. Next we discuss in detail the GA component for the last two stages.

3. Genetic algorithm

GA [16] as a metaheuristic has been widely used in search of solutions to various forms of complex COPs [2,6,16]. A metaheuristic is a master strategy that guides the exploration of other heuristics in search of solutions to a given problem [17]. GA is an adaptive stochastic search metaheuristic inspired by evolutionary biology ideas of inheritance, mutation, natural selection, and recombination (crossover) [16]. Computer simulation of GA allows populations of abstract representations (chromosomes) of candidate solutions (individuals) of an optimization problem evolve toward better

a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}
a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}
a_{31}	a_{32}	a_{33}	a_{34}	a_{35}	a_{36}
a_{41}	a_{42}	a_{43}	a_{44}	a_{45}	a_{46}
a_{51}	a_{52}	a_{53}	a_{54}	a_{55}	a_{56}
a_{61}	a_{62}	a_{63}	a_{64}	a_{65}	a_{66}
a_{71}	a_{72}	a_{73}	a_{74}	a_{75}	a_{76}
a_{81}	a_{82}	a_{83}	a_{84}	a_{85}	a_{86}

Fig. 1. A chromosome representation for hall allocation with $m = 8$ and $n = 6$.

solutions. The evolution starts with an initial population of completely random individuals and moves through a number of generations in search of optimal solution. During each successive generation, the fitness of the individuals in the population are evaluated, multiple individuals are stochastically selected from the current population, modified (mutated or recombined) to form a new population, which becomes the current population in the next iteration of the algorithm. GAs offer a generational improvement in the fitness of chromosomes which after some generations creates other chromosomes that supposedly contain optimized variable settings [18,19].

GAs have been successfully applied to a wide variety of problems because of their simplicity, global perspective, and inherent parallel processing [20]. The algorithm has been shown to be an effective tool for initial study of metaheuristic application to some real-life cases of COPs. Most large scale problems similar to HSAP with no specific benchmarking basis employed GA for a baseline study. Corne et al. [21] employed GA for solving the examination timetabling problem. They applied a repair mechanism to overcome the infeasibilities due to the direct chromosome representation that generate infeasible offspring solutions. The solution obtained was found to be better than a manual solution. Annevelink and Broekmeulen [22] studied the application of GA to space allocation planning in pot-plant nurseries in the field of horticulture. The work was a pioneer that seeks the use and incorporation of GA metaheuristic into decision support system for the complex-natured space allocation planning in the nurseries. The result obtained proved the viability of GA for such study. Similarly, Dean [23] investigated the use of GA to optimize staff scheduling problem within the domain of nurse rostering. The study was based on a real-life case of assigning employees to time slots in a way that satisfies given constraints. The study employed two different chromosome representations, namely bit string and two-dimensional array to work on a data set of nurse shift-duty rostering within a four-week schedule. Simulation results obtained proved the efficiency of GA and the use of two-dimensional array chromosome structure over the bit string [23]. Andresen et al. [6] showed that GA performs better for minimizing mean flow time in an open shop for which the author reported that there were no previous results available. Open shop scheduling problems arise in many applications including quality control centers, teacher-class assignments, and examination scheduling [24,25], thus having similarity to SAP. Therefore, for HSAP, being a pioneer study also, we employed GA to explore solutions to our real-life instance.

Issues of interest in the application of GA to COPs include genetic representation of the chromosome, definition of domain specific fitness function, choice of GA operators (selection, crossover and mutation), the establishment of genetic parameters for controlling the algorithm and the termination of the algorithm [26]. We present below a brief description of our GA representation and operators for HSAP.

3.1. Solution representation

As said earlier, we apply GA successively to the two stages of hall and block/floor allocations. We design two separate but similar chromosome representations for these two stages. The chromosome structure for hall allocation is given in Fig. 1. The structure has the form $(c_1(a_{11} \ldots a_{1n}), c_2(a_{21} \ldots a_{2n}) \ldots c_m(a_{m1} \ldots a_{mn}))$, where c_i represents category i and a_{ij} represents the allocation of category i to hall j, m is the number of categories and n is the number of halls. The population for hall allocation thus consists of Ψ individuals, where ψ is the population size. For computational purpose, this is represented as a three-dimensional array of integers. An integer in the array with subscripts p, i, *and* j indicates the number of students in category i allocated to hall j in the solution represented by the individual p. At this level of the allocation process, various categories are allocated to the given halls. The allocation at this level is a two-dimensional entity.

The next is the allocation into various rooms within the block/floor for each hall of residence based on the result obtained from the hall allocation and associated constraints. This involves allocation of students under each category into a block and floor within the hall. The chromosome structure for this is illustrated in Fig. 2. It has the form $(c_1(b_{11}(a_{111} \ldots a_{11x}) \ldots b_{1w}(a_{1w1} \ldots a_{1wx})) \ldots c_m(b_{m1}(a_{m11} \ldots a_{m1y}) \ldots b_{mw}(a_{mw1} \ldots a_{mwy})))$ where c_i represents category i, b_{il} represents block l in category i's allocation, a_{ilk} represents category i's allocation on floor $k$of block l, m is the number of categories, w is the number of blocks while x and y are the varying number of floors in the respective blocks (the number of floors varies with the hall shown in Appendix A). The allocation at this level is a three-dimensional entity. Fig. 2 presents four sections (one for each block), each with 8 rows representing the eight categories. In this representation, the blocks have 3, 1, 2 and 4 floors respectively. The population consists of Ψ individuals. For implementation purpose, the population is represented as a four-dimensional array of integers. An integer in the array with subscripts p, i, l and k is the number of students in category i allocated to floor k in block l in the solution represented by the chromosome of individual p.

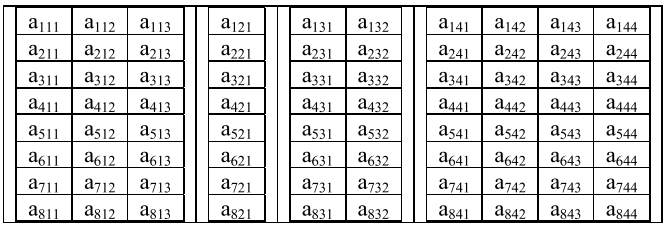


Fig. 2. A chromosome representation for block and floor allocation with $m = 8$, and $w = 4$.

3.2. Fitness evaluation

The fitness of value of an individual in a population is computed as a measure of the degree of satisfaction of given constraints that influence the allocation. The degree of satisfaction is computed as a combined factor of space utilization and weights assigned to conflicting constraints. These two are used in computing the fitness value of the individual (current allocation). At both the hall and block/floor (room) allocation levels, fitness values are set of real numbers in [0,1]. A value of 0 indicates complete violation of all given constraints while a value of 1 indicates non-violation of the constraints. Necessary constraints were considered separately at each levels of allocation.

For the hall allocation, the fitness of an individual in a population is computed as:

$$f = \sum_{q} w_q u_q \in [0, 1] \tag{2}$$

where u is the utilization factor (Eq. (1)) at the hall level given as:

$$u_q = \frac{a_{ij}}{s_i}, \tag{3}$$

w_q is the weight assigned to constraint q, $q = 1, 2, 3$ representing constraints presented in Table 2, a_{ij} is the number of allocations of category i to hall j, s_i is the total number of students in category i where i, j are fixed. The computation is repeated for all individuals in the population while the best fitness for the population is being noted. At this level, three major hard constraints are clearly identified (Table 2). Weights were assigned to these three constraints as follows: $W_{Ht} - 0.4$; $W_{Sc} - 0.3$; $W_{Sp} - 0.3$.

For the block and floor allocation, two constraints are specified namely, for Ht (allocation to the lowest possible floors in halls) and Fy (allocation to the highest possible floors in the halls). These constraints are assigned weights as follows: where a hall has Ht with no Fy allocation: $W_{Ht} = 1.0, W_{Fy} = 0.0$; where a hall has Fy with no Ht allocation: $W_{Fy} = 1.0, W_{Ht} = 0.0$; and where the hall has both Ht and Fy allocations: $W_{Ht} = 0.7, W_{Fy} = 0.3$. Ht category is given a higher priority because of the hard constraint assigned to them as a result of their peculiar condition. The fitness value of an individual is computed as in Eq. (2) where u is replaced with the utilization factor at the floor allocation level given as:

$$u_q = \frac{T_i}{s_{ij}}, \tag{4}$$

and

$$T_i = \sum_{k=1}^{\rho} a_{ik}, \tag{5}$$

w_q is the weight assigned to constraint $q, q = 1, 2$ representing floor level soft constraints (highest and lowest floor possible) as discussed in Section 2.2.1, a_{ik} is the number of allocations of category i to floor k, ρ is an integer representing the highest (or lowest) floor for the hostel under consideration and s_{ij} is the total number of students in category i allocated to hall j.

3.3. GA operators

The allocation at hall level is mutually exclusive from that of block and room allocation hence GA operations were performed at both levels. Since both allocations come with different criteria and constraints, the GA operators, though similar, are applied in a slightly different way.

The initial population of individual for hall allocation is generated randomly for each category of students. For block/floor allocation, the same process is repeated with the random selection of floors within various blocks in the hall. During successive generations, a proportion of existing populations are selected to breed new population set. Several selection methods have traditionally been used in GAs with roulette-wheel selection (RWS) as the most popular. At this stage we experiment with both the RWS and the stochastic universal sampling (SUS) [27] selection strategies. Our main result is based on a modified form of RWS. Individual solutions are selected through a fitness-based process, where better solutions

(as measured by a fitness function) have better chances of being selected. The algorithm was implemented to allow for selection of small proportion of less fit individuals thus making room for diversity in the population and prevention of premature convergence. The fitness values of all individuals are sorted in ascending order. The sorted fitness values are summed to compute the cumulative fitness of each solution as well as the total fitness of the entire population. The relative fitness of each individual in the population is calculated by dividing the cumulative fitness of the individual by the total fitness of the population. A random number is then generated in the interval [0,1]. The first individual whose cumulative fitness is greater than the generated number is selected for participation in reproduction. Where fitness proportionate selection chooses several solutions from the population by repeated random sampling, our implementation of SUS uses a single random value to sample all of the solutions by choosing them at evenly spaced intervals. The individuals are mapped to contiguous segments of a line, such that each individual's segment is equal in size to its fitness exactly as in roulette-wheel selection. Here equally spaced pointers are placed over the line as many as there are individuals to be selected. The distance between the pointers is 1/*popsize* and the position of the first pointer is given by a randomly generated number in the range [0, 1/*popsize*]. SUS ensures a selection of offspring which is closer to what is expected in RWS.

The crossover of selected parents is done for each category of students based on a generated random number, r, in the interval [0, 1]. If r is less than 0.5, then the first child inherits the allocation of the category from the first parent while the second child inherits from the second parent, otherwise the reverse takes place. A repair algorithm is invoked to ensure that the offspring produced do not exceed the capacity of the hall (in case of hall allocation) or the capacity of the floor (in case of block and floor allocation). We implemented both the one-point and two-point crossover for comparative purposes.

Mutation depends on the specified mutation rate and a randomly generated variable. At the hall level, two halls and two allocated categories are selected randomly. A random integer, r, is generated where $r \in \{0, \alpha\}$ and α is the smaller of the allocations for the first category in the first hall and the second category in the second hall. The allocation of the first category in the first hall is increased by r while the allocation of the second category in the same hall is decreased by r. The reverse is performed for the second hall to ensure that the hall capacities are not exceeded. The same process is carried out for the block and floor allocation while substituting floors for halls.

For this problem, we define an optimal solution as one which satisfied all the given constraints (i.e. $\sum w_q u_q = 1$ for all the entities considered). GA is made to terminate if any of the following three conditions are met: (1) an optimal solution is found, (2) for fifty consecutive generations, the average fitness of the current population is at least 95% that of the previous average fitness and 80% of the best fitness found throughout execution, (3) execution has been carried out over a specified number of generation without (1) or (2) being satisfied.

4. Simulation experiments and results

The need to strike a balance between explorative and exploitative features of GA makes it imperative to carry out parameter tuning via simulation experiment [16]. The tuning of parameters such as mutation probability, recombination probability and population size in order to find reasonable settings for a problem class is inevitable [28]. GA is an adaptation process, hence it is natural to expect, that during its action its parameters are adapted on the base of some internal dynamics of the algorithm [29]. It is necessary therefore to find an optimal parameter setting that would accelerate the process convergence of the algorithm. Though some literature suggest basic general recommendations of parameter setting for the implementation of GA [26] yet domain specific experimental instance is needed to determine the best parameter setting. We report here some results obtained from simulation experiments for the current study. Simulation coding was done using an open-source eclipse®integrated development environment (IDE) for Java developer, release 3.3 running on Windows Vista business operating system running on an Intel core 2 CPU at 1.66 GHz with 1 GB of RAM.

We conduct series of simulation experiments based available data set for the case study to find combinations of suitable parameters values such as population size (N), crossover rate (P_c) and mutation rate (P_μ) that give optimal (or good sub-optimal) results. To see the effects of varying P_c and P_μ, we conducted the following experiments. We study the effect of P_c in [0.1, 1.0]. 10 different values of P_c in [0.1, 1.0] with an increment of 0.1 are used. For each fixed P_c, we then conducted 5 runs each for P_μ in [0.0, 0.1] with an increment of 0.01. The average is then computed, hence there are 10 average fitness values. The same process is repeated to see the effect of P_μ in two different intervals i.e. [0.0, 0.1] and [0.1, 0.9]. In these experiments, P_c was varied from 0.1 to 1.0. The results are summarized in Figs. 3–8.

We first study the convergence of GA using Fig. 3 where x-axis represents the # of generations and y-axis represents the fitness values. For each generation, the overall best fitness value was noted as well as the average fitness value for all individual chromosomes in the generation. The average fitness is the sum of all the fitness values divided by N, where N is the population size. Both set of values were then plotted against the number of generations. It is clear from Fig. 3 that as the number of generations move away from forty-five, the best fitness and average fitness values were seen to converge.

Next, we study the sensitivity of P_c. We vary P_c in [0.1, 1.0] with an increment of 0.1 in order to determine the effect of crossover rate on the performance of the algorithm. At each value of P_c, the value of P_μ is varied from 0.0 to 1.0 with 0.1 increment giving 10 different fitnesses. The average fitness for each 10 values was taken for each value of P_c and plotted against P_c. The result is summarized in Fig. 4. It is clear from Fig. 4 that the best effect of P_c was observed in the range 70%–80% with some good results also at 10%–20%, 40%–50%. Value of P_c higher than 80% seems to worsen the performance of the algorithm.

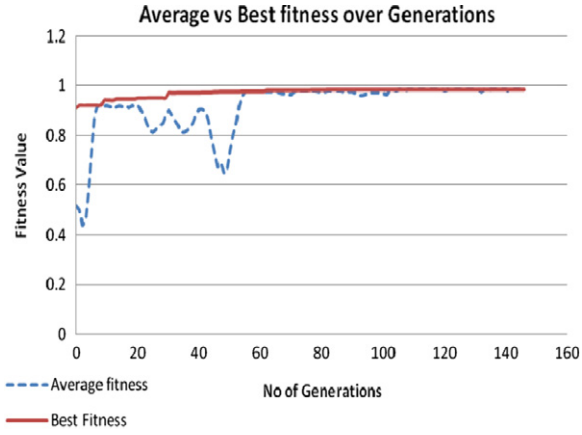


Fig. 3. Graph of average vs best fitness over # of generations ($N = 30$, $P_c = 0.1$, $P_\mu = 0.2$).

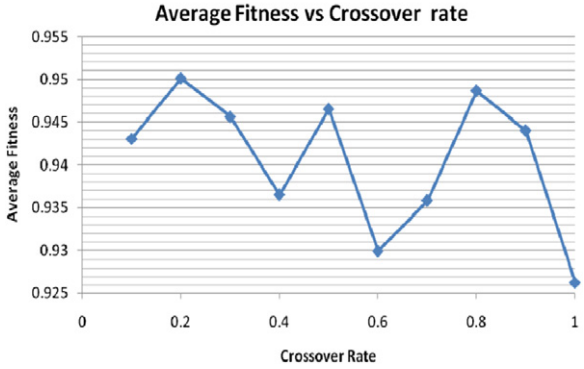


Fig. 4. Average fitness vs crossover rate.

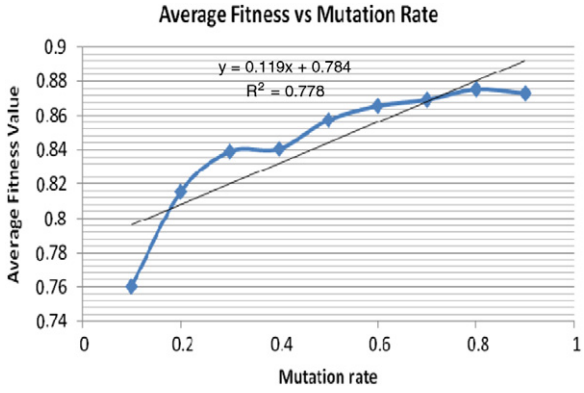


Fig. 5. Average fitness vs mutation Rate ($0.1 - 0.9$).

Furthermore, we study the effects of P_μ, but unlike P_c, here we used two different intervals for P_μ i.e. $P_\mu \in [0, 0.1]$ and $P_\mu \in [0.1, 0.9]$. As with P_c, for each value of P_μ, 10 values of P_c were used and the average fitness is calculated. The results are summarized in Figs. 5 and 6. Fig. 5 shows clearly an improvement in the algorithm with values of P_μ between 40%–80%. However, better results were obtained with values of P_μ in [0, 0.1] in Fig. 6 compared with Fig. 5. For further comparative purpose, Microsoft Excel was used to plot the line of best fit (regression line) over each graph of the two sets of P_μ as shown in Figs. 5 and 6. It is clear that P_μ gives better performance for the algorithm in the interval [0, 0.1] with correlation coefficient (R^2) 0.922 (Fig. 6) compared with the interval [0.1, 1.0] with correlation coefficient (R^2) 0.778 (Fig. 5).

We now study the effect of population size N on the performance of the algorithm. The values of P_μ and P_c are fixed while the value of N is varied in [10, 100] with increment of 10. While previous studies are based on the combination of both male and female allocation, this study is conducted for male and female allocation exclusively. The result obtained for

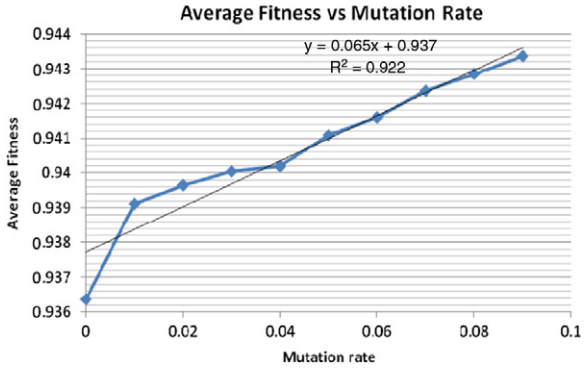


Fig. 6. Average fitness vs mutation Rate $(0.0 - 0.1)$.

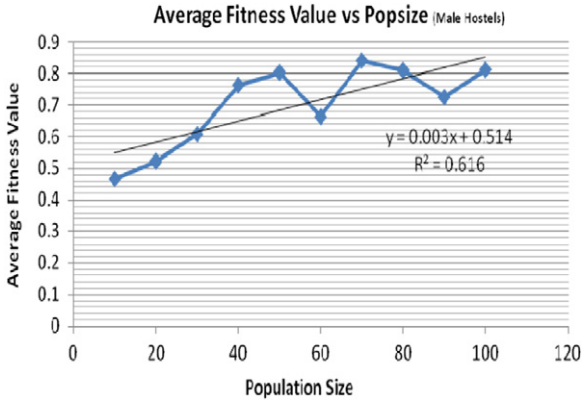


Fig. 7. Average fitness vs popsize ($P_c = 0.1$, $P_\mu = 0.1$).

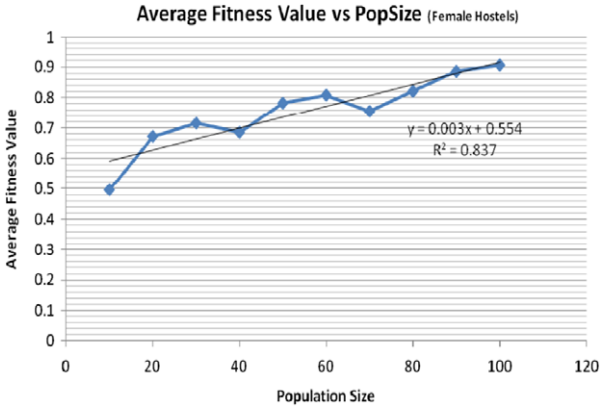


Fig. 8. Average fitness vs popsize ($P_c = 0.1 P_\mu = 0.1$).

the two allocations are presented separately in Figs. 7 and 8. It is clear from the two figures that the performance of the algorithm tends to improve with increasing value of N. However, in Fig. 7, the performance show decreasing trend for value of N in [70, 90] and N in [45, 60]. Fig. 8 gives a slightly better performance with very good results as N moves towards 100. To further compare the male and female allocations, a regression line was fitted on each of the two results. Interestingly, both the male and female allocations show the same trend (slope of both trend lines = 0.003 (Figs. 7 and 8)). However, the correlation coefficient for the female allocation (Fig. 8) is higher ($R^2 = 0.837$) than that of the male (Fig. 7) ($R^2 = 0.616$). This is likely due to the fact that the number of female applicants and constraints related to them are generally lesser than that of the male.

Finally, a study to determine the rate of feasibility of the solutions was conducted. We define a feasible solution as one that does not violate any of the hard constraints. For this study, a fixed set of values of P_c, P_μ, and N were used and the number of

Table 3
Results of experiment to determine rate of feasibility on the combination (0.3, 0.3, 100).

Run#	Male			Female		
	No. of gens	# feasible solutions	Feasibility rate (%)	No. of gens	# feasible solutions	Feasibility rate (%)
1	247	15 091	61.10	256	9 309	36.36
2	261	8 884	34.04	194	12 840	66.19
3	471	17 535	37.23	297	9 905	33.35
4	389	15 586	40.07	193	6 390	33.11
5	252	9 279	36.82	125	2 721	21.77
6	290	9 161	31.59	225	8 518	37.86
7	334	6 911	20.69	170	7 252	42.66
8	267	10 011	37.49	209	12 124	58.01
9	291	11 070	38.04	251	7 265	28.94
10	248	7 176	28.94	166	4 204	25.33
11	279	10 072	36.10	153	6 252	40.86
12	269	10 423	38.75	61	1 859	30.48
13	275	11 264	40.96	251	14 711	58.61
14	322	9 657	29.99	343	17 005	49.58
15	286	10 685	37.36	254	2 495	9.82
16	323	11 187	34.63	220	7 822	35.55
17	159	4 324	27.19	341	16 210	47.54
18	320	13 205	41.27	97	5 242	54.04
19	328	11 943	36.41	257	12 638	49.18
20	321	8 784	27.36	182	8 157	44.82
21	467	15 535	33.27	385	17 007	44.17
22	181	5 413	29.91	241	7 905	32.80
23	262	12 786	48.80	251	10 000	39.84
24	239	6 817	28.52	130	1552	11.94
25	237	8 969	37.84	286	8929	31.22
26	267	9 353	35.03	310	13 716	44.25
27	193	6 556	33.97	416	24 099	57.93
28	287	11 194	39.00	337	11 988	35.57
29	282	9 746	34.56	220	10 582	48.10
30	224	7 432	33.18	210	7 543	35.92
31	242	7 293	30.14	177	1644	9.29
32	313	12 552	40.10	129	4 989	38.67
33	133	2 935	22.07	233	10 513	45.12
34	290	7 741	26.69	237	13 291	56.08
35	259	11 140	43.01	210	9 561	45.53
36	264	10 826	41.01	293	8 662	29.56
37	292	8 713	29.84	345	15 209	44.08
38	372	14 435	38.80	51	1 988	38.98
39	322	8 138	25.27	221	10 351	46.84
40	295	13 908	47.15	317	18 364	57.93
41	210	9 048	43.09	104	1 334	12.83
42	252	11 793	46.80	97	2 498	25.75
43	330	10 965	33.23	122	2 880	23.61
44	225	8 810	39.16	193	10 194	52.82
45	280	13 983	49.94	325	13 929	42.86
46	343	11 588	33.78	188	10 701	56.92
47	462	16 197	35.06	364	15 619	42.91
48	396	11 715	29.58	256	10 611	41.45
49	238	5 891	24.75	69	923	13.38
50	313	11 233	35.89	195	6 098	31.27

generations also fixed at 1000. If the algorithm does not stop based on the stopping conditions discussed then it stops after 1000 generations. The algorithm was made to execute 50 times independently. For each successful execution, the number of generations evolved, total number of feasible solutions over all generations and the feasibility rate (number of feasible solutions as a percentage of the total number of solutions) were computed. One of the results obtained using the values of $P_c = 0.3$, $P_\mu = 0.3$, and $N = 100$ is presented in Table 3 sorted by # of runs. Two separate scatter graphs of percentage of feasible solutions against the # of runs are presented in Figs. 9 and 10 for male and female allocations respectively. The two graphs reveal the random nature of GA (the rate of feasible solutions covers a very wide range and also conforms to Normal Distribution). There were quite large numbers of feasible solutions scattered across various computed number of generation. Interestingly also, the feasibility rate show some improvement as the number of generations increases for both male and female allocations.

A summary of a generated allocation distribution from the simulation experiment is presented in Appendix B. The generated outputs show a summary of the category, hall and block/floor allocations respectively based on given data set in Appendix A. The category allocation gives the male and female breakdown of the number of students allocated (and unallocated) under each category. Fig. 11 gives a summary of the category allocation based on the priority given. We observe

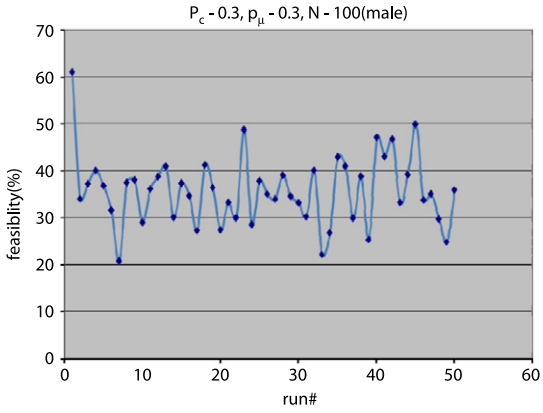


Fig. 9. Feasibility study for male allocation.

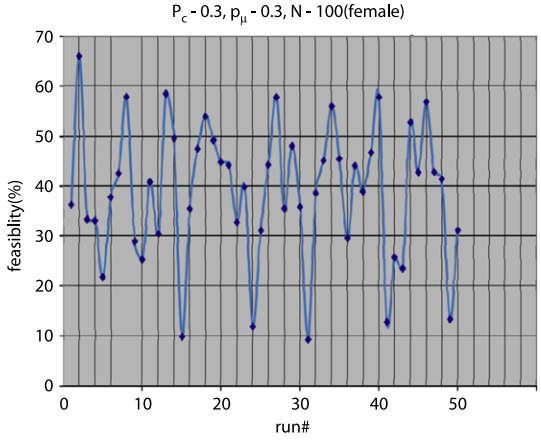


Fig. 10. Feasibility study for female allocation.

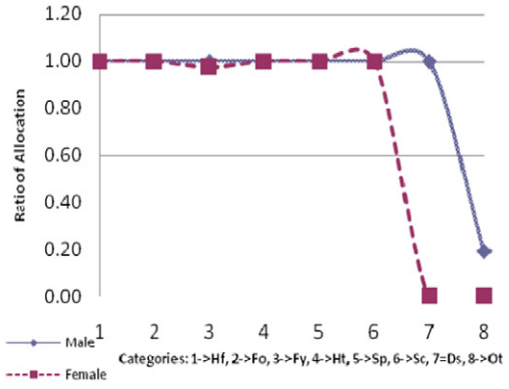


Fig. 11. Distribution of category allocation.

that the allocation decreases with decreasing priority of the various categories. The hall allocation gives the breakdown of the distribution of the results of the category allocation into the various halls. The block and floor allocation gives the number of students under each category that are allocated to a particular block/floor for each of the halls of residence. Fig. 12 depicts the hall allocation for one of the male halls (Jaja). In the figure, the *y*-axis represents the ratio of the category allocation to the capacity of the wing (block). For example, if a wing has a capacity of 18 and an allocation of 9 is done for a category under consideration, then the ratio is 0.5. From the figure, we observe that all health students were allocated to ground floor of A and B wings (with a proportion of 1) while spill over were allocated to the first floor of A wing. This satisfied a major constraint specify for Ht category.

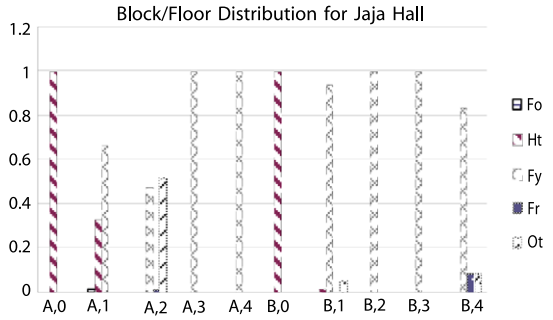


Fig. 12. Block/Floor allocation for a male hall.

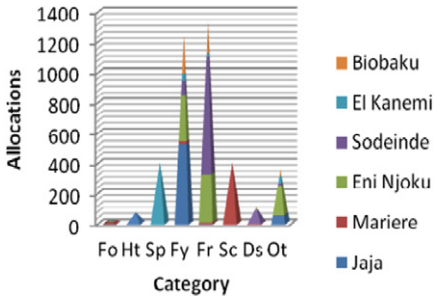


Fig. 13a. Hall allocation distribution using the main operators.

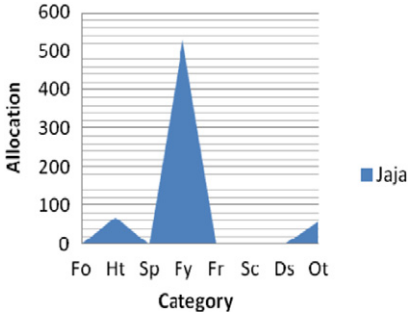


Fig. 13b. Allocation distribution using the main operator.

To avoid infeasible solutions and determine the best operator for the GA implementation of timetabling problem, Ross et al. [30] had to experiment with different operator options to generate offspring solutions. Similarly, we implemented various combinations of the selection and crossover operators aside the one we discussed earlier. We use combinations of SUS or RWS with either of one-point or two-points crossover operators, both with or without mutation operator. The experiment was performed only for the hall allocation level. Results obtained for the male allocation only are presented in Figs. 13a–17b. The (a) part of the figures gives a general distribution obtained for category allocations into male hostels while the (b) part gives the allocation distribution into one of the male hostel only. These were selected for illustration and study purposes.

Figs. 13a and 13b represent the results obtained from our normal operation described earlier (and shown in Appendix B). We observe that the generated distribution meets the hard constraint requiring the allocation all Ht, Sp and Sc to Jaja, El-Kanemi and Mariere halls respectively. We also note that a reasonable allocation spread is achieved by this algorithm (Fig. 13b) compared with other combinations used (Figs. 14a–16b). The distribution obtained with the combination of SUS selection scheme with two-point crossover operator is presented in Figs. 14a and 14b while Figs. 15a and 15b represents the results from the combination of SUS with one-point crossover operator. The last two combinations were repeated with the exclusion of mutation operator and the results presented in Figs. 16a, 16b, 17a and 17b respectively. We observe that the use of SUS selection operator with two-points crossover give a faster convergence. However, the results are not comparable to our main implementation in term of absolute constraints satisfaction. The same situation was observed with the use of SUS with one-point crossover howbeit, with slower convergence. Thus the use of our modified RWS and single-point

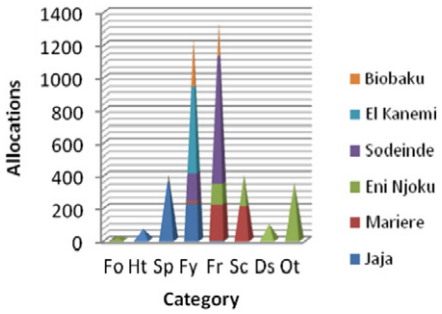


Fig. 14a. Hall allocation distribution using SUS with 2Pts CxOver.

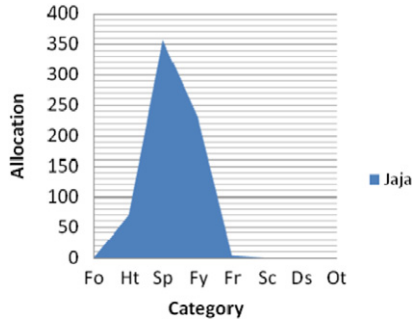


Fig. 14b. Allocation distribution using SUS with 2Pts CxOver.

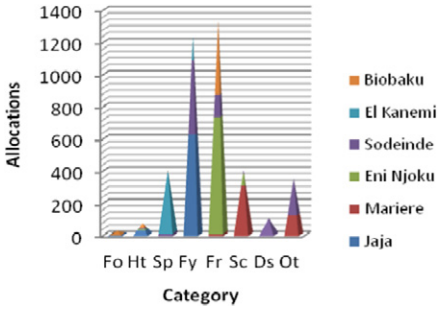


Fig. 15a. Hall allocation distribution using SUS with 1Pts CxOver.

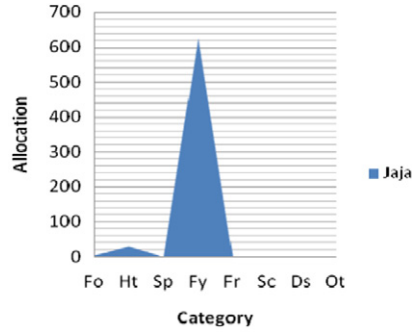


Fig. 15b. Hall allocation distribution using SUS with 1Pts CxOver.

crossover seems the best option for the given dataset. Possible reasons for this might include the nature of the chromosome representation and complexity of given constraints.

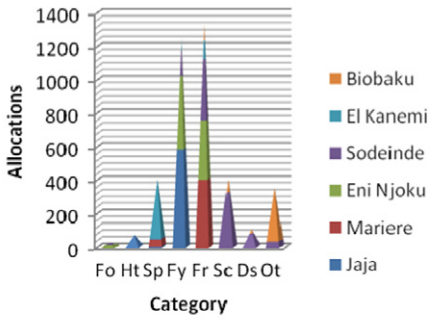


Fig. 16a. Hall allocation distribution using SUS with 2Pts CxOver (without mutation).

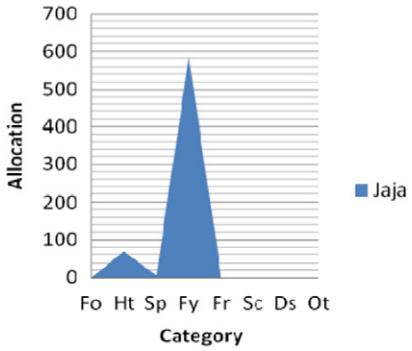


Fig. 16b. Hall allocation distribution using SUS with 2Pts CxOver (without mutation).

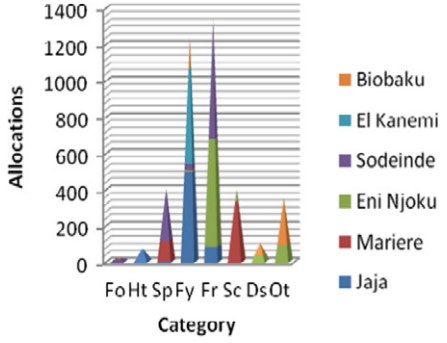


Fig. 17a. Hall allocation distribution using SUS with 1Pts CxOver (without mutation).

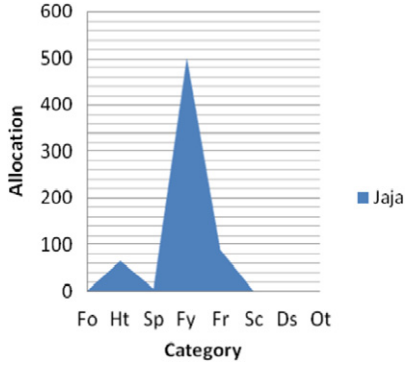


Fig. 17b. Allocation distribution using SUS with 1Pts CxOver (without mutation).

Table A.1
Sample statistics of applicants per category.

Category	Male	Female
Final year	1240	1420
Foreign	20	25
First year	1332	1367
Health Basis	70	80
Sport	400	500
Scholars	400	230
Discretion	100	60
Others	1800	1000

Table A.2
Statistics showing Hall, Block/floor capacities (Male).

Jaja (660)		Mariere (440)				Eni Njoku (800)				Sodeinde (968)				El-Kanemi (526)				Biobaku (512)	
Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap
A0	18	A0	0	C0	0	F0	20	R0	20	E0	20	J2	60	10	40	110	40	10	80
A1	100	A1	40	C1	40	F1	60	R1	60	E1	60	J3	60	20	40	120	40	11	144
A2	100	A2	40	C2	40	F2	60	R2	60	E2	60	T0	12	30	40	130	46	12	144
A3	100	A3	40	C3	40	F3	60	R3	60	E3	60	T1	60	40	40			13	144
A4	12	B0	0	D0	0	G0	20	S0	20	H0	12	T2	60	50	40				
B0	18	B1	40	D1	28	G1	60	S1	60	H1	60	T3	60	60	40				
B1	100	B2	40	D2	28	G 2	60	S2	60	H2	60	W0	12	70	40				
B2	100	B3	40	D3	28	G3	60	S3	60	H3	60	W1	60	80	40				
B3	100									J0	12	W2	60	90	40				
B4	12									J1	60	W3	60	100	40				

A thorough study of the GA generated hall and block/floor allocations in our main experiment reveals a very high degree of satisfaction of given hard and soft constraints at both levels thus showing the viability of the metaheuristics in handling this problem.

5. Conclusion

We examines a new domain of SAP namely, HSAP with emphasis on a case study from a developing country. The problem is multi-stage in nature and a multi-stage GA was also developed to provide solutions based on given hard and soft constraints. Results obtained show the practicability of employing metaheuristics in handling this class of problems. To the best of our knowledge, this is the first application of metaheuristics to solve HSAP. Our results thus provide a firm foundation for decision making by relevant authority in the university on the allocation of hostel space that will achieve the four-point *TREE* objectives of transparency, reliability, efficiency and effectiveness for hostel allocation. The use of heuristics can thus form a basis on which automated system can be built for the allocation process. Future research will include the use of other metaheuristics and their variants for HSAP as well as experimenting with more data set that can be gathered from other institutions.

Acknowledgements

This work has been supported by the University of Lagos, Nigeria and the University of Witwatersrand, South Africa. The support and effort of Mr. Gbenga Bastos and the School of Computational & Applied Mathematics, Wits University are appreciated.

Appendix A. Sample input into the experiment

The tables in this appendix contain data set used for the experiment as obtained as at the time of the work. The Table A.1 gives the statistics of applicants per category. The Tables A.2 and A.3 give the capacity, number of block/floor with label used for the male and female halls respectively. A label *W 0*, means block *W* floor 0 (i.e. ground floor). *Cap* refers to the/number of bed spaces. A capacity of zero (0) indicates that the particular block/floor is reserved for special purpose and is not to be allocated. The number in bracket against each of the hostel's name indicates the total capacity (number of bed spaces) for the hostel.

Appendix B. Sample of generated hostel space allocation for hall, block and floor

See Table B.1.

Table A.3
Statistics showing Hall, Block/floor capacities (Female).

Moremi (866)				Makama (764)				Fagunwa (276)		MTH (524)				Amina (646)		Kofo (512)	
Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap	Blk/Flr	Cap
A0	0	D2	60	D0	0	U3	80	C0	0	A0	30	M2	32	A0	60	10	80
A1	30	D3	60	D1	32	V0	40	C1	30	A1	40	M3	32	A1	60	11	144
A2	30	E0	60	D2	32	V1	40	C2	30	A2	40	N0	32	A2	60	12	144
B0	40	E1	60	D3	32	V2	40	L0	40	A3	40	N1	32	B0	40	13	144
B1	40	E2	60	K0	48			L1	40	B0	30	N2	32	B1	100		
B2	40	E3	60	K1	80			L2	40	B1	40	N3	32	B2	100		
C0	40	F0	0	K2	80			P0	0	B2	40			C0	26		
C1	40	F1	30	K3	80			P1	24	B3	40			C1	100		
C2	40	F2	30	U0	20			P2	24	M0	0			C2	100		
D0	0	G0	43	U1	80			S0	0	M1	32						
D1	60	H0	43	U2	80			S1	24								
								S2	24								

Table B.1
Sample of generated allocation distribution for Hall, Block and Floor.

Category allocation				
	Allocated		Unallocated	
	Male	Female	Male	Female
Foreign:	20	25	0	0
Health:	70	80	0	0
Sport:	400	500	0	0
Final:	1240	1420	0	0
First:	1332	1333	0	34
Scholar:	400	230	0	0
Discretion:	100	0	0	60
Other:	348	0	1452	1000
Total:	3910	3588	1452	1094
Gross:	7498		2546	

Hall allocation						
	Jaja	Mariere	Eni Njoku	Sodeinde	El-Kanemi	Biobaku
Foreign	1	10	3	1	5	0
Health	70	0	0	0	0	0
Sport	0	0	0	0	400	0
Final	529	18	295	97	51	250
First	2	12	313	776	21	208
Scholar	0	400	0	0	0	0
Discretion	0	3	0	79	5	13
Other	58	1	189	15	44	41
	Moremi	Makama	Fagunwa	MTH	Amina	Kofo
Foreign	6	14	1	1	3	0
Health	80	0	0	0	0	0
Sport	0	0	0	0	500	0
Final	292	347	73	217	18	473
First	258	403	202	306	125	39
Scholar	230	0	0	0	0	0
Discretion	0	0	0	0	0	0
Other	0	0	0	0	0	0

Block and floor allocation										
	Jaja									
Block, Floor	A, 0	A, 1	A, 2	A, 3	A, 4	B, 0	B, 1	B, 2	B, 3	B, 4
Foreign	0	1	0	0	0	0	0	0	0	0
Health	18	33	0	0	0	18	1	0	0	0
Final	0	66	47	100	12	0	94	100	100	10
First	0	0	1	0	0	0	0	0	0	1
Other	0	0	52	0	0	0	5	0	0	1

	Mariere											
Block, Floor	A, 1	A, 2	A, 3	B, 1	B, 2	B, 3	C, 1	C, 2	C, 3	D, 1	D, 2	D, 3
Foreign	0	0	0	10	0	0	0	0	0	0	0	0

Table B.1 (*continued*)

Final	0	0	0	0	0	2	0	0	3	0	0	13							
First	0	0	0	1	0	9	0	1	0	0	1	0							
Scholar	40	40	40	29	40	29	40	35	37	28	27	15							
Discretion	0	0	0	0	0	0	0	3	0	0	0	0							
Other	0	0	0	0	0	0	0	1	0	0	0	0							
		Eni Njoku																	
Block, Floor	F, 0	F, 1	F, 2	F, 3	G, 0	G, 1	G, 2	G, 3	R, 0	R, 1	R, 2	R, 3	S, 0	S, 1	S, 2	S, 3			
Foreign	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0			
Final	0	0	53	60	0	0	54	45	1	0	4	47	0	0	30	1			
First	19	60	6	0	3	60	0	0	19	1	56	7	20	3	0	59			
Other	0	0	0	0	17	0	6	14	0	59	0	6	0	57	30	0			
			Sodeinde																
Block, Floor	E, 0	E, 1	E, 2	E, 3	H, 0	H, 1	H, 2	H, 3	J, 0	J, 1	J, 2	J, 3	T, 0	T, 1	T, 2	T, 3	W, 0	W, 1	W, 2	W, 3
Foreign	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
Final	0	0	0	5	0	2	1	2	0	0	1	60	0	1	0	24	0	0	0	1
First	20	1	60	55	12	58	59	32	3	60	59	0	12	59	59	36	12	60	60	59
Discretion	0	47	0	0	0	0	0	26	6	0	0	0	0	0	0	0	0	0	0	0
Other	0	12	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0
			El-Kanemi																
Block, Floor	1, 0	2, 0	3, 0	4, 0	5, 0	6, 0	7, 0	8, 0	9, 0	10, 0	11, 0	12, 0	13, 0						
Foreign	0	0	0	0	1	2	1	1	0	0	0	0	0						
Sport	15	40	40	40	39	38	2	39	8	13	40	40	46						
Final	5	0	0	0	0	0	33	0	4	9	0	0	0						
First	0	0	0	0	0	0	0	0	20	1	0	0	0						
Discretion	0	0	0	0	0	0	0	0	0	5	0	0	0						
Other	20	0	0	0	0	0	4	0	8	12	0	0	0						
		Biobaku																	
Block, Floor	1, 0	1, 1	1, 2	1, 3															
Final	1	0	144	105															
First	60	138	0	10															
Discretion	13	0	0	0															
Other	6	6	0	29															
		Moremi																	
Block, Floor	A, 1	A, 2	B, 0	B, 1	B, 2	C, 0	C, 1	C, 2	D, 1	D, 2	D, 3	E, 0	E, 1	E, 2	E, 3	F, 1	F, 2	G, 0	H, 0
Foreign	0	0	0	0	0	0	0	1	0	0	2	0	0	0	2	0	0	1	0
Health	0	0	0	0	0	40	0	0	0	0	0	0	0	0	0	0	0	1	39
Final	1	25	3	2	38	0	13	30	0	29	58	0	4	60	23	0	4	1	1
First	29	5	18	20	2	0	27	9	60	15	0	0	56	0	1	2	0	12	2
Scholar	0	0	19	18	0	0	0	0	0	16	0	60	0	0	34	28	26	28	1
		Makama																	
Block, Floor	D, 1	D, 2	D, 3	K, 0	K, 1	K, 2	K, 3	U, 0	U, 1	U, 2	U, 3	V, 0	V, 1	V, 2					
Foreign	0	3	0	0	0	0	0	0	9	0	0	0	2	0					
Final	5	29	24	1	0	3	80	0	0	80	80	5	0	40					
First	27	0	8	47	80	77	0	20	71	0	0	35	38	0					
		Fagunwa																	
Block, Floor	C, 1	C, 2	L, 0	L, 1	L, 2	P, 1	P, 2	S, 1	S, 2										
Foreign	1	0	0	0	0	0	0	0	0										
Final	0	30	0	0	17	1	24	0	1										
First	29	0	40	40	23	23	0	24	23										
		MTH																	
Block, Floor	A, 0	A, 1	A, 2	A, 3	B, 0	B, 1	B, 2	B, 3	M, 1	M, 2	M, 3	N, 0	N, 1	N, 2	N, 3				
Foreign	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0				
Final	0	0	12	40	1	1	7	29	0	29	32	1	1	32	32				
First	30	40	28	0	29	39	33	11	32	2	0	31	31	0	0				
		Amina																	
Block, Floor	A, 0	A, 1	A, 2	B, 0	B, 1	B, 2	C, 0	C, 1	C, 2										
Foreign	2	0	1	0	0	0	0	0	0										
Sport	58	60	59	40	100	31	26	100	26										
Final	0	0	0	0	0	0	0	0	18										
First	0	0	0	0	0	69	0	0	56										

(*continued on next page*)

Table B.1 (*continued*)

	Kofo			
Block, Floor	1, 0	1, 1	1, 2	1, 3
Final	80	144	144	105
First	0	0	0	39

References

[1] R. Bai, An investigation of novel approaches for optimising retail shelf space allocation, Ph.D. Dissertation, School of Computer Science and Information Technology, University of Nottingham, UK, 2005.
[2] J.D. Landa-Silva, Metaheuristics and multi-objective approaches for space allocation. Ph.D. Dissertation. School of Computer Science and Information Technology, University of Nottingham, UK, 2003.
[3] E.K. Burke, P. Cowling, J.D. Landa-Silva, B. McCollum, Three methods to automate the space allocation process in UK universities, in: Proceedings of the 3rd International Conference on the Practice and Theory of Automated Timetabling, PATAT2000, Konstanz, Germany, 2000, pp. 374–393.
[4] A.O. Adewumi, J.O.A Ayeni, O. Abass, E.P. Fasina, Some comments on task allocation problem in distributed computing systems, J. Comput. Sci. Appl. 9 (1) (2003) 136–145.
[5] A.O. Adewumi, N. Ihemedu, J.O.A. Ayeni, An evolutionary-based approach to university course time-tabling problem, Poster Session, First Annual Research Conference and Fair, University of Lagos, 9(11) (2005), October.
[6] M. Andresen, H. Brasel, M. Morig, J. Tusch, F. Werner, P. Willenius, Simulated annealing and genetic algorithms for minimizing mean flow time in an open shop, Math. Comput. Modelling 48 (2008) 1279–1293.
[7] E.K. Burke, D.B. Varley, Automating space allocation in higher education, in: Selected Papers from the 2nd Asia Pacific Conference on Simulated Evolution and Learning SEAL 98, in: Lectures Notes in Artificial Intelligence, vol. 1585, 1998, pp. 66–73.
[8] E.K. Burke, D.B. Varley, Space allocation: An analysis of higher education requirements, in: E.K. Burke, M.W. Carter (Eds.), The Practice and Theory of Automated Timetabling II: Selected Papers from the 2nd International Conference on the Practice and Theory of Automated Timetabling PATAT 1997, in: Lecture Notes in Computer Science, vol. 1408, 1998, pp. 20–33.
[9] E.K. Burke, P. Cowling, J.D. Landa-Silva, On the performance of population-based metaheuristics for the space allocation problem: An extended abstract, in: Proceedings of the 4th Metaheuristics International Conference, MIC 2001, Porto Portugal, 2001, pp. 579–583.
[10] L. Ritzman, J. Bradford, R. Jacobs, A multiple objective approach to space planning for academic facilities, J. Mgmt. Sci. 25 (9) (1980) 895–906.
[11] C. Benjamin, I. Ehie, Y. Omurtag, Planning facilities at the university of missoury-rolla, J. Interface 22 (4) (1992) 95–105.
[12] J. Giannikos, E. El-Darzi, P. Lees, An integer goal programming model to allocate offices to staff in an academic institution, J. Oper. Res. Soc. 46 (6) (1995) 713–720.
[13] D. Pisinger, An exact algorithm for large multiple knapsack problems, European J. Oper. Res. 144 (1999) 528–541.
[14] A. Federgruen, G.V. Ryzin, Probabilistic analysis of a generalized bin packing problem and applications, Oper. Res. 45 (4) (1997) 596–609.
[15] Federal university of technology minna. student affairs department information, available on www.futminna.org/currentstudents/student_affairs.htm. Accessed in October, 2008.
[16] D.E. Goldberg, Genetic Algorithms in Search, Optimisation and Machine Learning, Addison Wesley, 1989.
[17] F. Glover, Future paths for integer programming and links to artificial intelligence, Comput. Oper. Res. 13 (5) (1986) 533–549.
[18] C.B. Lucasius, I.G. Kateman, Understanding and using genetic algorithms — Representation, configuration and hybridization (A tutorial), Chemo. Intel. Lab. Syst. 25 (1994) 99–145.
[19] Z. Ramadan, D. Jacobs, M. Grigorov, S. Kochhar, Metabolic profiling using principal component analysis, discriminant partial least squares, and genetic algorithms, Talanta 68 (2006) 1683–1691.
[20] S.N. Sivanandam, S.N. Deepa, Introduction to Genetic Algorithms, Springer, New York, 2008.
[21] D. Corne, H.L. Fang, C. Mellish, Solving the modular exam scheduling problem with genetic algorithms, in: The proceedings of the 6th international conference of industrial and engineering applications of ai and expert system, 1993, pp. 370–373.
[22] E. Annevelink, R.A.C.M. Broekmeulen, The Genetic Algorithm applied to Space Allocation Planning in Pot-Plant Nurseries, in: XII International Symposium on Horticultural Economics Acta Hort. (ISHS) 340 (1995) pp. 141–148.
[23] J.S. Dean, Staff scheduling by a genetic algorithm with a two-dimensional chromosome, in: E.K. Burke, M. Gendreau, (Eds.), Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling — PATAT, 2008, Montreal, Canada, August 18–22, 2008, p. 15.
[24] W. Kubiak, C. Sriskandarajah, K. Zaras, A note on the complexity of open shop scheduling problems, INFOR 29 (1991) 284–294.
[25] C.Y. Liu, R.L. Bulfin, Scheduling ordered open shops, Comput. Oper. Res. 14 (1987) 257–264.
[26] M. Obitko, Genetic algorithm — recommendation. Hochschule für Technik und Wirtschaft Dresden (FH) (University of Applied Sciences), Available on http://cs.felk.cvut.cz/~xobitko/ga/recom.html. Accessed in October, 2008.
[27] J.E. Baker, Reducing bias and inefficiency in the selection algorithm, in: Proceedings of the Second International Conference on Genetic Algorithms and their Application (Hillsdale), 1987, pp. 14–21.
[28] Wikipedia. Genetic Algorithm. Accessed via http://en.wikipedia.org/wiki/Genetic_algorithm in October 2008.
[29] W. Kosiński, S. Kotowski, Limit properties of evolutionary algorithms, in: W. Kosiński. Advances in Evolutionary Algorithms, I-Tech Education and Publishing, Vienna, Austria, November 2008, p. 468, ISBN 978-953-7619-11-4.
[30] P. Ross, E. Hart, D. Corne, Some observations about GA based timetabling. The practice and theory of automated timetabling II, in: E.K. Burke, M.W. Carter (Eds.), The Practice and Theory of Automated Timetabling II: Selected Papers from the 2nd International Conference on the Practice and Theory of Automated Timetabling PATAT 1997, in: Lecture Notes in Computer Science, vol. 1408, 1998, pp. 115–129.