Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2017) 25: 794 – 805

doi:10.3906/elk-1504-250

A modified genetic algorithm for a special case of the generalized assignment problem

Murat DÖRTERLER[1,*], **Ömer Faruk BAY**[2], **Mehmet Ali AKCAYOL**[3]
[1]Department of Computer Engineering, Gazi University, Teknikokullar, Ankara, Turkey
[2]Department of Electrical and Electronics Engineering, Gazi University, Teknikokullar, Ankara, Turkey
[3]Department of Computer Engineering, Gazi University, Maltepe, Ankara, Turkey

Received: 28.04.2015 • **Accepted/Published Online:** 01.03.2016 • **Final Version:** 10.04.2017

Abstract: Many central examinations are performed nationwide in Turkey. These examinations are held simultaneously throughout Turkey. Examinees attempt to arrive at the examination centers at the same time and they encounter problems such as traffic congestion, especially in metropolises. The state of mind that this situation puts them into negatively affects the achievement and future goals of the test takers. Our solution to minimize the negative effects of this issue is to assign the test takers to closest examination centers taking into account the capacities of examination halls nearby. This solution is a special case of the generalized assignment problem (GAP). Since the scale of the problem is quite large, we have focused on heuristic methods. In this study, a modified genetic algorithm (GA) is used for the solution of the problem since the classical GA often generates infeasible solutions when it is applied to GAPs. A new method, named nucleotide exchange, is designed in place of the crossover method. The designed method is run between the genes of a single parent chromosome. In addition to the randomness, the consciousness factor is taken into consideration in the mutation process. With this new GA method, results are obtained successfully and quickly in large-sized data sets.

Key words: Genetic algorithm, optimization, generalized assignment problem

1. Introduction

Many examinations are held for various purposes in Turkey. Examinations such as matriculations, distance learning, and employment and certificate examinations are planned and held throughout the country because of the huge numbers of applicants. School buildings are used as examination centers; classrooms of schools are used as examination halls for such examinations.

Among the aforementioned exams, matriculations receive the most applicants. In 2011, 1,759,998 test takers took the exam around the country. The number of test takers in Ankara and İstanbul was 127,957 and 266,712 respectively.

On the days the examinations are held, traffic jam and congestion occur in the cities, particularly in metropolises. This situation is likely to cause the test takers to arrive late at the exam centers. They sometimes might have to walk to arrive at the examination centers. The test takers strive to be at the exam centers on time and this makes them lose their motivation. Considering that such examinations are held once a year and that this situation directly influences the rest of the lives of the test takers, removing the problems of getting to the examination centers is important.

*Correspondence: dorterler@gazi.edu.tr

One of the possible solutions to minimize the negative effects of the issue is to assign the test takers to the closest exam center as much as possible. The main constraint in this solution is that the capacities of the examination halls should not be exceeded. This solution can be considered as a specific case of the generalized assignment problem (GAP). Heuristic methods are comparatively feasible for solving large-scale NP-hard problems.

The genetic algorithm (GA), one of the foremost heuristic methods, is used to solve the GAP. Nevertheless, not only the previous GA studies for GAPs but also our own trials indicated that the capacity constraint of the GAP is violated when the classical GA is implemented. Additional or alternative operators were proposed in order to solve GAPs in previous studies. In this study, a different chromosome structure and a nucleotide exchange (NE) method are proposed due to the special case of the problem.

In this study, structures of genes and chromosomes are presented as object-oriented rather than numerical vectors. Each significant datum denoted in the gene is called a nucleotide, as designated in the natural sciences. We developed and applied a NE method, creating offspring from a single parent instead of two parents at crossover level. In addition to that, controlled mutation is applied using a factor of consciousness at mutation level. As a result of these new approaches, successful results were obtained.

In the second section of the study, the GAP and previous studies on GAPs are presented. In Section 3, the GA and the new method are elaborated. In Section 4, the structure of the data set simulating the problem and the results of the simulations are explained. Experimental results and analyses are provided in Section 5.

2. The generalized assignment problem

The GAP is a well-known NP-complete combinatorial optimization problem [1]. It is a type of one-to-many assignment problems that recognizes capacity limits [2]. The GAP considers finding the minimum cost assignment of n tasks to m agents provided that each task should be assigned to one agent only. On the other hand, an agent may be assigned more than one task, subject to the agents' available capacity [3].

The mathematical formulation of the GAP is:

$$\text{Minimise } \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \tag{1}$$

$$\text{Subject to: } \sum_{i=1}^{m} x_{ij} = 1 \quad j = 1, ..., n, \tag{2}$$

$$\sum_{j=1}^{n} a_{ij}x_{ij} \leq b_i \;\; i = 1, ..., m, \tag{3}$$

$$x_{ij} = 0 \; or \; 1, \tag{4}$$

where c_{ij} is the cost of assigning agent i to task j, and if agent i is assigned to task j, $x_{ij} = 1$, else 0; a_{ij} is the amount of agent i's capacity consumed by task j in the case that it is assigned to i; and b_i is the available capacity of agent i. The first constraint indicates that each task is assigned to only one agent; the second constraint is that the resource demands of the tasks that are assigned to an agent must not exceed the capacity of the agent.

The GAP was introduced by Ross and Soland [3]. The definition of the GAP covers many real-world applications such as vehicle routing [4], grouping and loading for flexible manufacturing systems [5], assigning ships to overhaul [6], assigning jobs to computers in computer networks [7], and land use allocation [8]. Cattrysse and Van Wassenhove mentioned cardinal applications and algorithms for GAPs in their survey study [9].

The algorithms proposed to solve GAPs can be implemented as exact and heuristic. The branch-and-price algorithm [10], cutting plane algorithm [11], and branch-and-cut algorithm [12] are significant exact algorithms proposed in the last two decades. The existing exact algorithms are not effective for more difficult large-scale problems. Hence, large-scale GAP problems are often tackled by applying heuristics to obtain approximate solutions [13]. Various heuristic methods were used to solve GAPs. Osman designed both tabu search and hybrid simulated annealing/tabu search methods for GAPs [14]. Tabu search was also used by Diaz and Fernandez [15]. The differential evolution algorithm [16], bees algorithm [17], and neighborhood search algorithm [18] are other notable heuristics approaches applied to GAPs.

The GA is a prominent heuristic method used to solve GAPs. When GAP studies approached with GAs are analyzed in the literature, the classical GA often generates infeasible solutions because it violates the capacity constraint of Eq. (3). Therefore, additional processes are needed to solve GAPs by GAs [13,19–21].

The study of Chu and Beasley [13], which inspired later studies, used classical crossover, binary tournament selection, and mutation operators. They presented an unfitness function in order to evaluate the degree of infeasibility of the individuals together with a fitness function to evaluate cost. A heuristic operator involving two local improvement steps was applied to each offspring individual after they were created. The first step attempts to improve the feasibility of the offspring and the second step attempts to improve the cost of the offspring. However, the heuristic methods do not guarantee a feasible solution.

In Wilson's study [20], classical crossover and tournament selection was implemented with potentially infeasible initial individuals. In addition to that, a 2-phase local search method was proposed. This method improves the feasibility of a selected individual in the first phase and searches for a better fitness value in the second phase. If a feasible individual is obtained, the GA is stopped and the solution is selected; otherwise, the solution that fits best is selected at the end of the crossover. The local search method is applied to one individual only; however, it does not ensure feasibility. Even if this study could obtain a feasible solution, the probability of getting an optimum solution is very low.

Feltl and Raidl [21] proposed several improvements for the study of Chu and Beasley [13]. They presented two alternative heuristic methods to increase the proportion of feasible individuals in the initial population. Selection and replacement schemas are suggested to eliminate infeasible individuals. A heuristic mutation solution is used to decrease the probability of turning feasible individuals into infeasible ones.

2.1. The problem as a special case of the GAP

When the description of the GAP is adapted to our problem, agents correspond to examination halls and tasks correspond to test takers. The minimum cost corresponds to total distance from their homes covered by test takers in order to arrive at the examination centers. The constraints of the GAP are also valid for our problem.

Though the definition of the GAP covers our problem thoroughly, there is a special case. The capacity consumed by each task taker is equal. Under the GAP formulation, a_{ij} always equals 1. The studies in literature are not feasible owing to the special case.

3. The proposed genetic algorithm

3.1. Background

It is possible to solve NP-hard combinatorial problems by means of deterministic or heuristic methods. Nevertheless, applying deterministic methods to these types of problems ends with either inability to reach the result or a longer duration until the appearance of the solution. On the other hand, heuristic algorithms such as the GA can provide good approximate solutions with comparatively much lower computational complexity.

The heuristic methods do not guarantee a globally optimal solution to be found. Nevertheless, they are easy and quick methods to arrive at an acceptable solution. The GA, one of the heuristic search methods, was first proposed by Holland [22]. Holland aimed both at improving the perception of natural adaptation and at designing artificial systems similar to the characteristics of natural systems [23]. The GA has been developed with the help of inspiration by natural selection [24]. GA methods can produce highly effective results for complex optimization problems.

The GA is an iteration process that aims to get a new solution set that contains better solutions from the first solution set built randomly. Each solution is called a chromosome and the set built from solutions is called a population in the GA. Each chromosome consists of genes, which are significant parts of the solution. After each one of the iterations, the evolved new-generation chromosomes replace the current chromosomes. A crossover process is used for creating new chromosomes. The crossover process provides copulation of parent chromosomes to create a new chromosome. The created chromosome is called an offspring. The created offspring may be subject to a mutation process. A mutation process changes either one or more genes of the offspring randomly. The fitness function is used for choosing the best fitting chromosomes among the offspring and parents so that new generations are created. This process, the imitation of the evolution process, repeats until a better result cannot be obtained or until a certain number of iterations occurs [25].

3.2. Coding method

The first step of GA design is to constitute the structure, which requires coding the solution. Each potential solution must be defined by a numeric vector. The coding has been denoted as a bit array in the first samples of the GA. On the other hand, using real data directly provides significant advantages [26]. In previous GA studies [13,20,21] mentioned in Section 2, each possible solution is represented by an integer vector $S=\{S_1, S_2, \ldots, S_n\}$. For each $j=1,\ldots,n$, the Sj indicates the agent to which job j is assigned. In our approach, structures of genes and chromosomes are defined as object-oriented. Various data of agent and job are kept together in a gene. Each significant datum kept in a gene is called a nucleotide, as designated in the natural sciences.

Let L represent the population consisting of a chromosome set and each chromosome stands for a solution of the problem. Chromosome C_k consists of an array of genes $(g_i | i = 1, ..., n)$. Genes represent meaningful units of a solution. The structure of the gene is defined as per the analysis of the problem.

Each one of the genes (g_i) indicates which test taker is assigned to which examination hall. Hence, if n number of test takers will participate in an exam, the chromosome is represented as $Ck = \{g_1, g_2, g_3, ..., g_n\}$ (Figure 1). Genes have an object-oriented structure and they contain the test taker's data (id, neighborhood, etc.) and examination hall data (id, neighborhood, position, etc.), in short, nucleotides, as depicted in Figure 2. The sequence of neither the test taker nor the examination hall matters in this representation.

Figure 1. Structure of a chromosome.

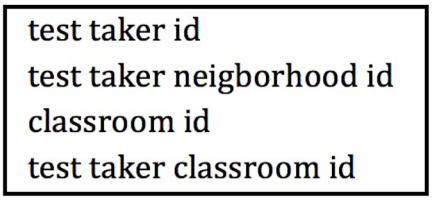

Figure 2. Structure of a gene and nucleotides.

3.3. Content of proposed GA

The proposed GA has been developed for solving the problem mentioned in the introduction. The aim of this study is to place the test takers into the optimally closest examination center taking into account the problem constraints. The main steps of the proposed GA are as follows:

1. Begin.
2. Generate an initial population of a predefined number of solutions. Each solution is generated by assigning each test taker to an examination hall randomly without violating the capacity constraint of Eq. (3).
3. Repeat until ending conditions are met.
   - 3.1. The fitness values of each solution are computed by using a fitness function.
   - 3.2. Repeat number of gene times for each chromosome.
     - 3.2.1. Parent solution is selected by means of predetermined selection method.
     - 3.2.2. The offspring solution is created by applying the NE method to the parent solution. The offspring takes its parent's place in the population.
   - 3.3. A randomly selected solution is subjected to a mutation process. The test taker denoted in the selected gene is reassigned to another examination hall, which has available capacity.
4. End.

3.3.1. Initiation

In this phase, a certain number of solutions is created randomly as the initial population. While each test taker is assigned to an examination hall randomly, the hall capacities are taken into account in the initial phase. Thus, each chromosome represents a feasible but suboptimal solution in the initial population. Successive phases are processed to improve the optimality of the solutions.

3.3.2. Evaluation of solutions

Evaluation of a solution is conducted using a fitness function. A fitness value is obtained for each solution at the end of the evaluation process. The fitness value is the cost described in the definition of the GAP and it is aimed to minimize the value. The fitness value of our problem is calculated by the following formula.

$$f_k = \frac{\sum_{j=1}^{n} d_j}{n} \tag{5}$$

Here, d_j indicates the distance traveled by the test taker who is assigned to examination hall j from his/her residence to the examination center.

3.3.3. Selection of new generations

The chromosomes, which will be the parents of the members of the new generation, are chosen to be as many as the number of the population among chromosomes of the current generation. In our approach only one parent is needed for creating a new offspring. Moreover, offspring creation causes the destruction of the parent. The selection of the parent can be carried out in different ways. Three popular selection methods are used in this study.

Random selection: The parent chromosome is selected from the current population randomly in this method. No criterion is used in this method, so the application of this method is quite easy. A chromosome can be selected more than once. Although obtaining the target results may take longer compared to the other selection methods, it is an effective way to avoid local solutions.

Roulette wheel selection: Fitness values (f_k) are worked out for each chromosome (C_k) by using a fitness function in this method. The selectable probability (p_k) of each chromosome in a population is calculated proportional to the fitness value (Figure 3). The selectable probabilities of the chromosomes ordered by fitness value are added until a randomly defined value is reached. The last chromosome added to the sum is selected as a parent chromosome.

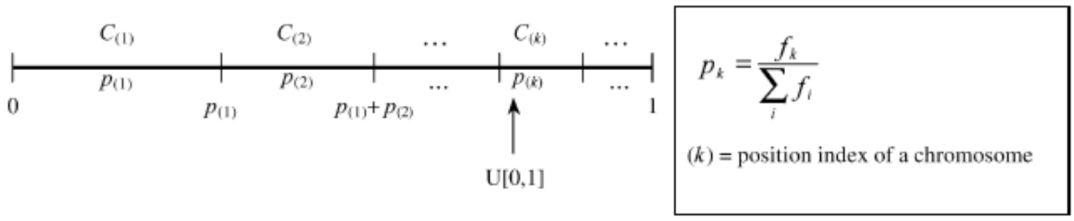

Figure 3. Roulette wheel selection method and formulation of selection probability [19].

Tournament selection: In this method two chromosomes are selected from the population randomly. The chromosome that has the better fitness value (f_k) is selected as the parent.

3.3.4. Nucleotide exchange

Offspring for new generations are created in this level. A crossover method is used at this level of classical GAs. However, when the crossover method is applied to the chromosomes, the capacity constraint of the GAP is most likely violated, as in previous studies [13,20]. Therefore, getting offspring by using the method of crossbreeding genes of two parents is abandoned all together. The NE method is developed and internalized.

Predefined exchange probability (ep) is used in NE. A random value is defined for each chromosome. Whether the chromosome is subjected to processing is decided, comparing the exchange probability and the random value of chromosome. NE takes place between each gene (g_i) of the chromosome and a random gene (g_r) of the same chromosome. Both of the genes must affect the fitness value negatively. As a result of this operation, while an offspring is created, the parents perish. The algorithm of the method, which is adapted to the problem, is as follows:

1. Begin
2. define ep
3. repeat for each chromosome (C_k) of the population
   - 3.1. If random value $<$ ep
     - 3.1.1. repeat for each gene (g_i) for chromosome
       - 3.1.1.1. get a gene (g_r) randomly
       - 3.1.1.2. If test takers of g_i and g_r are assigned outside of their neighborhood.
         - 3.1.1.2.1. Exchange test taker's data (nucleotides) between the genes.
   - 3.2. Update f_k of the chromosome
4. End.

Genes have an object structure and each gene consists of the data on the test takers and the examination halls to be assigned. The test takers of the subjected genes must be assigned to an examination center located outside their neighborhood. The mean of this condition is that these genes affect the fitness value negatively. It is expected that the fitness value will be improved after NE, so test takers' data are swapped between g_i and g_r and test takers are assigned new examination halls in the NE process (Figure 4). The data of examination halls stay the same while the test taker's data are changed in the gene. On the other hand, replacing examination hall data with the test taker's data does not change the result.

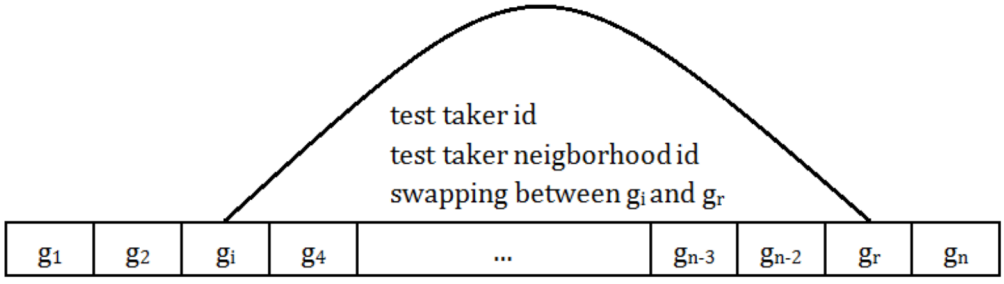

Figure 4. Properties exchange between two gene objects.

Genes consist of meaningful arrays of nucleotides in the perspective of the natural sciences. Mutation means an event of damage or a change in the sequence of nucleotides. According to this point of view, NE is a mutation. This mutation process makes the chromosome better thanks to conscious intervention even though it involves randomness. When this characteristic is considered, it is possible to state that the chromosome being subjected to the process of NE is similar to mutant individuals in science fiction.

3.3.5. Mutation

The value of the randomly selected gene is changed in the mutation process. The aim of the mutation is to avoid the local optimum solutions in the solution space. The number of mutation processes depends on the predefined mutation probability (mp). A randomly selected test taker is assigned to another hall with suitable capacity. A pseudocode for the algorithm of the method, which is adapted to the problem, is given below.

1. Begin
2. Define mutation probability (mp)
3. repeat for each chromosome (C_k) of population
    - 3.1. If random value $<$ mp
        - 3.1.1. get a chromosome randomly
        - 3.1.2. determinate a mutation point (g_r) randomly
        - 3.1.3. select an examination hall with suitable capacity randomly
        - 3.1.4. Change examination hall data of the gene at mutation with the selected one point
    - 3.2. Update f_k of the chromosome
4. End.

The mutation operator mostly changes the value of the selected gene. As the classical crossover operator does, the classical mutation operator damages the structure of the feasible solution and converts it to an infeasible one by violating the capacity constraint of the GAP of Eq. (3). Therefore, a consciousness factor, which takes into account the capacity of the halls, is involved. Step 3.1.3 is repeated until a suitable examination hall with available capacity is found.

4. Simulation results and discussion

Due to the special case of the problem, the algorithms proposed in the previous GAP studies [13,20,21] are not suitable to solve the specific case of the GAP. Moreover, the data sets used in the previous GAP studies are not suitable for testing the qualification of the proposed GA. Thus, we develop a simulation environment to indicate the capability of the proposed GA. The virtual examination environment, which simulates the problem, is developed for proving the applicability of the proposed GA. The virtual environment is obtained by creating a data set, which is needed for an examination in a city randomly. The data set is obtained in accordance with the rules listed below.

- Sixteen districts are defined in the city.
- The districts can be divided into 4 to 8 neighborhoods randomly.
- Between 2 and 4 schools can be defined in each district randomly.
- Between 0 and 2 schools can be placed in neighborhoods randomly.
- Between 8 and 16 classrooms can be assigned in schools randomly.

- Between 16 and 32 capacities can be determined in classrooms randomly.
- The number of test takers must be determined randomly between the total capacity determined randomly and 90% of this capacity.
- Test takers must be placed into neighborhoods randomly.

The shortest distance between the test taker's neighborhood and the test taker's examination center is the base in the fitness function. A virtual map of the virtual city is created randomly to determine the distance. Rules of the map are as follows:

- The city is located on a 256 km^2 square-shaped area.
- District lands are defined as 16 km^2 square-shaped lands, dividing the city into 16 equal parts.
- Centers of neighborhoods are defined on coordinate planes randomly provided that they fall inside the district borders.
- Bird's-eye view distances between the test taker's residential neighborhood and the neighborhood of the examination hall are calculated.

Software is developed by using the Java programming language with object-oriented programming methods to process the rules. The data set, which is generated within the defined rules, is obtained for the first trials as follows:

- Number of test takers is 15,343.
- Number of neighborhoods is 93.
- Number of examination centers is 63.
- Number of examination halls is 712.
- Total capacity of the halls is 17,014.

The population size value is defined as 100; the exchange probability value is defined as 0.50 and the mutation probability value is defined as 0.01 in experimental trials. In terms of performance, the developed simulation software is run on Java RTE v. 1.8.0_20 for Windows 8 x64 installed on a machine with an Intel Core i5-2400 3.1 GHz CPU and 6 GB of RAM.

Computational experience showed that results of less than 2.1 km are approximate for the created data set. This value is determined as the stop criterion of trials, which are performed in order to measure the performance of the selection methods. Ten trials are performed for each selection method and the results are indicated in Table 1. According to these values, the roulette wheel performs better than the other selection methods. A GA with roulette wheel method reaches the stop criterion with minimum iterations. Though the roulette wheel has a relatively more complex algorithm, it provides minimum execution time thanks to fewer iterations. Performances of the tournament and random selection methods follow the roulette wheel consecutively. It can be stated that heuristic selection methods are more successful than random selection methods to achieve the goal (Figure 5).

Table 1. Performance values of the selection methods.

	Average number of iterations	Average execution time (ms)	Result (m)
Roulette wheel	24.2	1572	2091
Tournament	31.6	2123	2093
Random	61.8	4395	2095

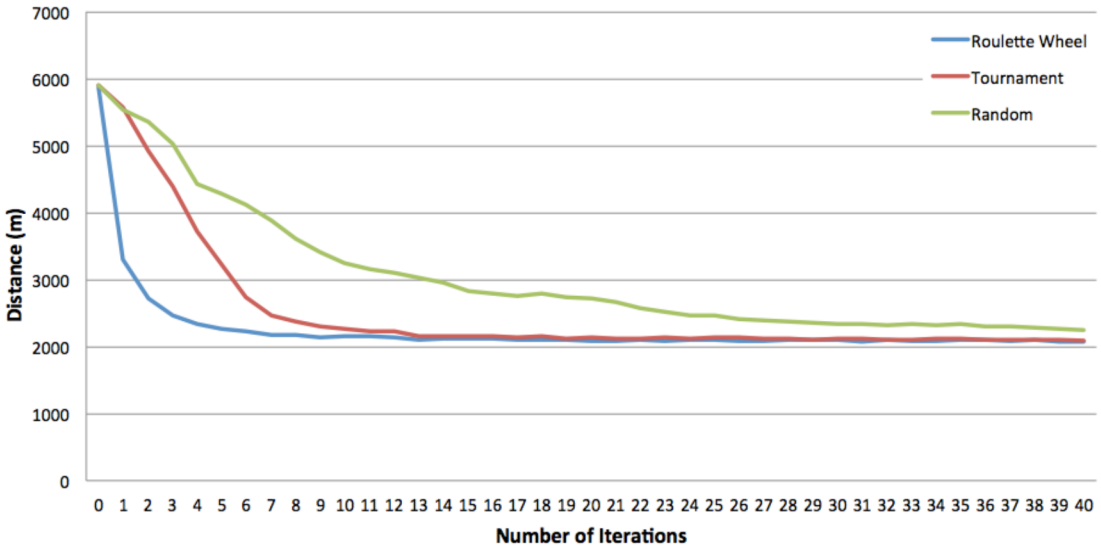

Figure 5. Average of traveled distance with different selection methods.

The proposed GA is also tested on data sets of five different examination environments, generated in accordance with the simulation rules. Although examination environments have the same acreage, they hold various numbers of test takers and sources such as number of exam centers or hall capacities. Locations of test takers' residences and examination centers are also different in the environments. The roulette wheel method is selected and the number of iteration is set to 30 for these trials. According to the values in Table 2, the proposed GA gives reasonable results and has stability under different conditions.

Table 2. Performance values of the proposed GA with various data sets.

	Number of neighbor oods	Number of examination centers	Number of examinati on halls	Capacity of the halls	Test takers	The best fitness value at start	The best fitness value at finish	Execution time (ms)
Env. A	93	63	712	17,014	15,343	5894	2089	1984
Env. B	96	61	763	18,255	17,760	5833	2581	2453
Env .C	102	64	780	18,808	18,277	6104	2629	2453
Env. D	93	64	746	17,969	16,921	6103	2043	2171
Env. E	90	68	848	20,358	19,374	5736	1940	2344

5. Conclusions

In this study, a new model of the GA is presented in order to assign test takers to the most appropriate examination halls in nationwide exams of Turkey. In the designed algorithm, the crossover method is abandoned.

The NE method is developed instead of crossover. Conscious effect is added to the mutation process so controlled mutation is implemented, too. A specific rule set is defined to test the proposed GA and an examination environment simulation is developed in a virtual city. When the new GA method is applied to the problem with simulation software, successful results are obtained. Furthermore, the results are obtained quite fast in spite of the large size of data set used.

The simulation is performed in a city created randomly with a random number of test takers, which are created randomly. Testing with real examination data may be effective to increase the applicability of the algorithm. The algorithm has been developed for a specific problem and fitness function with object-oriented programming methods. The proposed GA may be applied to different problems with different aims or constraints.

References

[1] Sahni S, Gonzalez T. p-Complete approximation problems. J ACM 1976; 23: 555-565.

[2] Pentico WP. Assignment problems: a golden anniversary survey. Eur J Oper Res 2007; 176: 774-793.

[3] Ross GT, Soland RM. A branch and bound algorithm for the generalized assignment problem. Math Program 1975; 8: 91-103.

[4] Fisher ML, Jaikumar R. A generalized assignment heuristic for vehicle routing. Networks 1981; 11: 109-124.

[5] Mazzola JB, Neebe AW, Dunn CVR. Production planning of a flexible manufacturing system in a material requirements planning environment. Int J Flex Manuf Sys 1989; 1: 115-142.

[6] Gross D, Pinkus CE. Optimal Allocation of Ships to Yards for Regular Overhauls. Technical Memorandum 63095. Washington, DC, USA: Institute of Management Science Engineering of George Washington University, 1972.

[7] Balachandran V. An integer generalized transportation model for optimal job assignment in computer networks. Oper Res 1972; 24: 742-759.

[8] Cromley RG, Hanink DM. Coupling land use allocation models with raster GIS. J Geogr Syst 1999; 1: 137-153.

[9] Cattrysse DG, Van Wassenhove LN. A survey of algorithms for the generalized assignment problem. Eur J Oper Res 1992; 60: 260-272.

[10] Savelsbergh M. A branch-and-price algorithm for the generalized assignment problem. Oper Res 1997; 45: 831-841.

[11] Avella P, Boccia M, Vasilyev I. A computational study of exact knapsack separation for the generalized assignment problem. Comput Optim Appl 2010; 45: 543-555.

[12] Nauss RM. Solving the generalized assignment problem: an optimizing and heuristic approach. INFORMS J Comput 2003; 15: 249-266.

[13] Chu PCH, Beasley JE. A genetic algorithm for the generalized assignment problem. Comput Oper Res 1997; 24: 17-23.

[14] Osman IH. Heuristics for the generalized assignment problem: simulated annealing and tabu search approaches. OR Spectrum 1995; 17: 211-225.

[15] Dıaz JA, Fernández E. A tabu search heuristic for the generalized assignment problem. Eur J Oper Res 2001; 132: 22-38.

[16] Tasgetiren MF, Suganthan PN, Chua TJ, Al-Hajri A. Differential evolution algorithms for the generalized assignment problem. In: IEEE Congress on Evolutionary Computation; 18–21 May 2009; Trondheim, Norway. New York, NY, USA: IEEE. pp. 2606-2613.

[17] Ozbakir L, Baykasoglu A, Tapkan P. Bees algorithm for generalized assignment problem. Appl Math Comput 2010; 215: 3782-3795.

[18] Yagiura M, Iwasaki S, Ibaraki T, Glover F. A very large-scale neighborhood search algorithm for the multi-resource generalized assignment problem. Discrete Optim 2004; 1: 87-98.

[19] Liu YY, Wang S. A scalable parallel genetic algorithm for the generalized assignment problem. Parallel Comput 2015; 46: 98-119.

[20] Wilson JM. A genetic algorithm for the generalised assignment problem. J Oper Res Soc 1997; 48: 804-809.

[21] Feltl H, Raidl GR. An improved hybrid genetic algorithm for the generalized assignment problem. In: Proceedings of the 2004 ACM Symposium on Applied Computing; 14–17 March 2004; Nicosia, Cyprus. New York, NY, USA: ACM. pp. 990-995.

[22] Holland JH. Adaptation in Natural and Artificial Systems. Ann Arbor, MI, USA: University of Michigan Press, 1975.

[23] Goldberg DE. Genetic Algorithms in Search, Optimization and Machine Learning. Boston, MA, USA: Addison-Wesley Longman Publishing, 1988.

[24] Chatterjee S, Laudato M, Lynch LA. Genetic algorithms and their statistical applications: an introduction. Comput Stat Data An 1996; 22: 633-651.

[25] Zolfaghari S, Liang M. A new genetic algorithm for the machine/part grouping problem involving processing times and lot sizes. Comput Ind Eng 2003; 45: 713-731.

[26] Salomon R. The influence of different coding schemes on the computational complexity of genetic algorithms in function optimization. In: Fourth International Conference on Parallel Problem Solving from Nature; 22–26 September 1996; Berlin, Germany. Cham, Switzerland: Springer. pp. 227-235.