International Journal of Computational Intelligence Systems, Vol. 6, No. 2 (March, 2013), 261-278

A Discrete Constrained Optimization Using Genetic Algorithms for A Bookstore Layout

Tuncay Ozcan[*]
Department of Industrial Engineering, Istanbul University
Avcilar, Istanbul, 34320, Turkey

Sakir Esnaf
Department of Industrial Engineering, Istanbul University
Avcilar, Istanbul, 34320, Turkey
E-mail: sesnaf@istanbul.edu.tr

Received 23 May 2012

Accepted 19 October 2012

Abstract

In retail industry, one of the most important decisions of shelf space management is the shelf location decision for products and product categories to be displayed in-store. The shelf location that products are displayed has a significant impact on product sales. At the same time, displaying complementary products close to each other increases the possibility of cross-selling of products. In this study, firstly, for a bookstore retailer, a mathematical model is developed based on association rule mining for store layout problem which includes the determination of the position of products and product categories which are displayed in-store shelves. Then, because of the NP-hard nature of the developed model, an original heuristic approach is developed based on genetic algorithms for solving large-scale real-life problems. In order to compare the performance of the genetic algorithm based heuristic with other methods, another heuristic approach based on tabu search and a simple heuristic that is commonly used by retailers are proposed. Finally, the effectiveness and applicability of the developed approaches are illustrated with numerical examples and a case study with data taken from a bookstore.

Keywords: Store layout, Shelf location, Genetic algorithms, Tabu search, Association rule mining.

1. Introduction

Location is an important factor affecting product demand in retail shelf space management. After the decision is made for products which include in the product assortment, the retail manager has to determine the locations of each selected product in the store. This is an important decision, which has a major effect on the sales performance of the retail store.[1] The other positive effect on product sales of effective store layout is provided by making it easier to find product items and creating a positive image. As in the famous beer diapers example, retailers create significant revenue potential by displaying products, which are often purchased together, side by side. The visual effect of product adjacencies occur at the point of encouraging unplanned purchasing

[*] Corresponding author, E-mail: tuncay.ozcan@istanbul.edu.tr , Tel: +90-212-4737070.

Co-published by Atlantis Press and Taylor & Francis

decisions, which are approximately 70% of purchasing decisions in a supermarket.[2] On the other hand, the shelf areas of a store are not regarded as equally important by customers. Customers pass throughout different areas of a shop with varying speeds and certain areas of the store draw more attention than others. In this regard, some examples of valuable store and shelf areas can be listed as follows[3]:

- Areas at the entrance of the store, especially the first shelf or other display areas that customer face immediately after entering the store.
- End caps of gondolas which are usually highly visible for people who do not enter into an aisle.
- Special presentation areas (e.g. off-shelf displays in a supermarket) are used to draw attention to specific products and new product introductions and create an additional demand.
- Check-out area, since all customers have to pass through it, this area is preferred for impulse items.
- Eye-level on shelves are more preferably since they are more visible; besides, because customers usually look through products from left to right in the western cultures, products on the lefts side will be seen in the first stage.

The first experimental study on the location effect was carried out by Dreze et al.[4] The location effect is displayed in a striking way by this study with data taken from a supermarket chain. In the scope of this study, 60 stores and 8 product categories from these stores were selected for the test and an average of 66000 observations were made per each product category of which contains an average of 115 product items. 32-week data were used to analyze of which the first 16 weeks before a change in shelf space and the second 16 weeks after the change in shelf space. The results of this study can be detailed as follows:

- Moving products from the worst horizontal position to the best horizontal position leads to an average of 15% increase in sales.
- The average sales difference between the worst and the best vertical position is above 39%. This finding reveals that vertical position has 2.5 times more effect upon product sales than horizontal position.
- Two units of visible amount at the eye-level, is more meaningful than five units of visible amount at the bottom shelves.
- Moving a unit product from the worst location to the best location according to horizontal and vertical positions in a store, will increase the demand of the product by an average of 60%.

The study of Dreze et al.[4] showed that shelf location had a large impact on sales, whereas changes in the allocated shelf space had much less impact as long as a minimum stock threshold was maintained. Consistent with these findings in another study, Hansen et al.[5] have shown that vertical location is approximately two times more effective than horizontal location on retailer's performance. In another study related to the location effect in retail stores, Chen et al.[2] investigated the relationship between spatial distances of the products displayed and their sales and analyzed the impact of the proximity of the shelf space on sales. For this purpose, a powerful algorithm based on association rule mining was developed.

The first shelf space model considering the location of the shelf was developed by Yang.[6] In numerous studies, the location effect is discussed along with shelf space allocation model. In these studies, the impact of the location of the shelf is added to the demand function, which is based on space elasticity and cross-elasticity.[1,7,8] Hwang et al.[7] dealt with the shelf space allocation problem, in which the rate of demand is a function of the inventory level displayed and the location of the shelf; and presented two different approaches, based on the gradient search and genetic algorithm, as the solution for the developed model. Hwang et al.[8] developed an integrated integer programming model for shelf-space design and allocation problem and proposed a genetic algorithm approach for solving the model. Hariga et al.[1] proposed a mixed integer nonlinear programming model, which decides on product assortment, stock replenishment, merchandising space and shelf space allocation, objective to maximize retailer's profitability under the constraints of shelf space and storage. Hansen et al.[5] presented a decision model for retail shelf space which integrates non-linear profit function, the effects of horizontal and vertical location and the cross-product elasticity. Murray et al.[9] developed a model which optimizes product prices for a product category, exhibit space, shelf space allocation and location in an integrated manner. In this study, unlike other models of shelf space allocation, both width and height of the shelf space were taken into account, stacking was allowed, and retailer's shelf space allocation decision was modeled to take account of cross-product interactions on demand, dependent on product pricing decisions.

Store layout design can play a critical role in effecting customer behavior and purchase intention for a retailer. However, most of the previous studies on the layout problem focus on manufacturing facilities or warehouse for the objective of minimizing the material handling cost. Although there is a vast amount of literature on manufacturing facility and warehouse layout, very few studies related to retail store layout have been published. Before reviewing these studies, it is useful to examine store layout types in retailing.

Retailers use three general types of store layout design: grid, racetrack and free form.[10] Each of these layouts can be briefly described as follows:

The grid layout is a rectangular arrangement of displays and has parallel aisles with merchandise on shelves on both sides of the aisles. Cash registers are located at the entrances/exits of the stores.[10] The advantages of this layout type are that of being cost-efficient, having minimal wasted space, the possibility of self-service and the ease of finding products. At the same time, this layout type allows more products to be displayed on the shelves. Due to these advantages, most supermarkets use the grid layout. On the other hand, this layout type has some disadvantages such as limited browsing, uninteresting arrangement and restricted stimulation.

The racetrack layout also known as a loop**,** is a store layout that provides a major aisle that loops around the store to guide customer traffic around different departments within the store. Cash register stations are typically located in each department bordering the racetrack.[10] Each individual area or sub-area is built for a particular shopping theme. The racetrack layout encourages unplanned purchasing. This layout type is used in department stores.

The free-form layout arranges displays and aisles in an asymmetric pattern. This layout type has some advantages like flexibility, allowance for browsing, relaxing environment and stimulation of impulse purchases. It is mainly used by name brand stores such as fashion stores.[11]

In the studies addressing retail store layout problem, firstly, Botsali and Peters[12] developed a network based layout design model for retail stores. Chen and Lin[13] developed an approach aiming to determine the location of the shelf for products and product categories in a supermarket retailer. In this study, the average profit per shelf space is used as a performance measure for shelf location decision. Seruni[14] proposed an approach to the product placement problem on the basis of market basket analysis. In another study, Cil[15] developed a new approach to supermarket layout using association rule mining and multidimensional scaling. Lastly, Yapicioglu and Smith[16] developed a model considering revenue maximization and adjacency satisfaction for a department store that uses racetrack layout. To solve this model, a tabu search heuristic was proposed.

In this study, store layout problem is considered for a bookstore retailer. In terms of the layout problem, the specificities of a bookstore can be summarized below:

- Bookstores usually use the grid layout. Impulse purchases are stimulated by special displays and promotional areas.
- The shelves have different importance ratings according to their positional locations in store.
- The cross sales between the product categories is affected by the distance and adjacency.
- Layout decision is about the determination of in-store shelf locations of product categories. After determining the shelf location for product categories to be displayed, the horizontal and vertical locations of product items are determined in this shelf area according to the criteria such as product name and author information, ranked in alphabetical order.
- The main objectives of a bookstore layout are not only to determine the shelf locations of the product categories considering the location effects of the shelves and revenues of the product categories, but also to provide desired adjacencies between the product categories for increasing cross sales effect.
- The shelf areas assigned to the same product categories must have integrity. In other words, if two shelves are assigned to the same product category, a shelf must be either on the right, left or opposite of another shelf. This constraint is a hard constraint for bookstore layout problem.

As mentioned above, the vast majority of the published work on layout problem has focused on manufacturing facilities. In these studies, various algorithms have been developed for the facility layout problems such as CRAFT, ALDEP and CORELAP. While CRAFT attempts to minimize material handling cost, ALDEP and CORELAP attempt to maximize a nearness rating. These criteria are not suitable for retail stores, especially bookstores. In bookstore layout, the layout design objective is to maximize profit by increasing direct and cross sales. For this reason, the existing methods developed for manufacturing facilities can't be directly applied to the bookstore layout problem. In the few studies related to retail store layout, Botsali and Peters[12] developed a network based layout design and Yapicioglu and Smith[16] used a racetrack layout. Models proposed in these studies do not satisfy the specificities and constraints of a bookstore layout. Therefore, this study develops a new model and solution approaches to optimize bookstore layout.

The remainder of this paper is organized as follows. In Section 2, a mixed integer mathematical model is developed for the store layout problem. In Section 3, due to the NP-Hard nature of the developed model, a heuristic approach based on genetic algorithms is developed for solving large-scale real-life problems. For analyze the performance of the genetic algorithm based heuristic, another heuristic approach based on tabu search and a simple rule that is commonly used by retailers are proposed. In Section 4, firstly, the performance analysis of the developed approaches is realized with problem instances different number of product categories and shelves. Then, a case study conducted with data from a book retailer will be detailed. In the final part, the results of the study are discussed.

Unlike the mentioned studies, a store layout problem is investigated for a bookstore retailer in this study. The contribution of this study to the existing literature is that a new original model is developed based on association rule mining considering location effects based on position and proximity factors. Additionally, a modified genetic algorithm based approach which is differentiated by the generation of the initial solution and crossover and mutation operators from classical genetic algorithm, is designed for solving this model.

2. Model Development

In this study, firstly, a mathematical model based on association rules has been developed for the determination of the shelf locations of the product items and product categories in a bookstore retailer. On the other hand, the bookstore retailer location decision is about the determination of in-store shelf locations of product categories. As mentioned above, the horizontal and vertical locations of product items are determined according to the criteria such as product name and author information, ranked in alphabetical order. Location effect is a function of the positional location of the shelf according to criteria such as proximity to the check-out area or entrance and the shelf space distances of product categories according to the obtained association rules.

In this study, following notation is used where j denotes the number of product categories ($j=1,2,..,m$) and k denotes the set of shelves in the store ($k= 1,2,...,o$).

S_j the number of shelves that will be assigned to the product category j

LE_k the location effect of the position of the shelf k

$SR_{j,l}$ the strength of association rule between product category j and l

$s_{j,l}$ the support of association between product category j and l

$c_{j,l}$ the confidence of association rule between product category j and l

$D_{k,t}$ the minimum distance between shelf k and t

$LR_{k,t}$ binary parameter indicating the relationship of location between shelf k and t

LE_j the location effect of the position of the category j according to shelves assigned to category j

LA_j the location effect of the proximity to other categories according to the association rules of category j

w_s the importance weight of location criteria s

NP_j the net profit value of category j

$X_{j,k}$ binary variable indicating the status that product category j is assigned to shelf k

Definitions of model parameters and variables and forms of calculation are as follows:

S_j and NP_j are values that are obtained as a result of shelf space allocation model or retailer preferences. $LR_{k,t}$ takes value of 1 in case shelf k is on the right, left or opposite side of shelf t, otherwise is set to value of 0. If $LR_{k,t}$ takes value of 1, same category can be displayed on shelf k and t. This parameter carries critical importance to provide cohesion of shelves assigned to the same category. LE_k value is determined by evaluation of shelf alternatives according to criteria, which is effective on the product demand, such as proximity to check-out area, proximity to the entrance, and number of related shelves. At this point, the performance of each shelf according to each criterion is normalized between [0,1]. The location effect of the position of shelf *k* according to criteria *n* can be calculated as below.

If the expectancy is larger-the-better (e.g., the benefit),

$$LE_k(n) = \frac{LE_k(n) - \min LE_k(n)}{\max LE_k(n) - \min LE_k(n)} \tag{1}$$

If the expectancy is smaller-the-better (e.g., the cost and distance),

$$LE_k(n) = \frac{\max LE_k(n) - LE_k(n)}{\max LE_k(n) - \min LE_k(n)} \tag{2}$$

The location effect of the position of the shelf by weighting each criteria performance can be calculated using Eq. (3). The importance weight of each criteria can be gathered from retailer's opinions or using methods such as AHP, ANP, TOPSIS etc.

$$LE_k = LE_k(1).w_1 + ... + LE_k(s).w_s \tag{3}$$

The location effect arising from the shelf position in-store of category according to the location effect of the position of shelves, which categories are assigned to, can be calculated by Eq. (4).

$$LE_j = \frac{\sum_{k=1}^{o} LE_k . X_{j,k}}{\sum_{k=1}^{o} LE_k}, \quad \forall j \tag{4}$$

$D_{k,t}$ value indicates the minimum value of distance between shelves. Distance values between shelves are normalized like Eq. (2) in order to calculate the location effect of the proximity to the other categories (LA_j) during the application of the model.

$$D_{k,t} = \frac{\max D_{k,t} - D_{k,t}}{\max D_{k,t} - \min D_{k,t}} \tag{5}$$

The location effect of the proximity to the other categories (LA_j) of each category by using distance values between shelves can be calculated as below.

$$LA_j = \frac{\sum_{l=1}^{m}\sum_{k=1}^{o}\sum_{t=1}^{o} X_{j,k}.X_{l,t}.SR_{j,l}.D_{k,t}}{\sum_{l=1}^{m} SR_{j,l}}, \quad \forall j \tag{6}$$

$SR_{j,l}$ value refers to the strength of association rule between categories *j* and *l*. The calculation of this value can be explained as following.

Association rule mining was first presented by Agrawal et al.[17]. A formal statement of the problem is as follows: Let $I=\{i_1,i_2,...,i_m\}$ be a set of items. Let *D* be a set of customer transactions, where each transaction *T* is an itemset such that $T \subseteq I$. We say that a transaction *T* contains *X*, a set of some items in *I*, if $X \subseteq T$. An association rule is an implication of the form $X \Rightarrow Y$, where $X \subset I$, $Y \subset I$ and $X \cap Y = \emptyset$. The association rule $X \Rightarrow Y$ holds in the transaction set *D* with confidence if *c%* of transactions in *D* that contain *X* also contain *Y*. In other words, it is the rate of number of transactions containing *X* and *Y* together to the number of transactions containing *X*. The association rule $X \Rightarrow Y$ has support *s* in the transaction set *D* if *s%* of transactions in *D* contain $X \cup Y$. Support value (*s%*) is calculated as the rate of transaction data containing *X* and *Y* to the total number of transaction data. The association rule is valid for sets with support and confidence values which are above the user-specified minimum support (S_{min}) and minimum confidence (C_{min}) values. Support degree measures the importance (or range) of association rule, confidence degree measures the accuracy of association rules. The strength of association rule can be measured by support degree and confidence degree.[18] The higher the confidence and support values, the stronger the association rule is. In other words, if the confidence value and the support value are high, the measure rule support multiplied by rule confidence is also high. That's why, $SR_{j,l}$ value of the model is determined with multiplying the support and confidence values of category pairs which have higher values than specified minimum support and minimum confidence values.

$$SR_{j,l} = s_{j,l}.c_{j,l}, \quad \forall j,l \tag{7}$$

Model constraints can be expressed as follows:

1. Number of shelf space assigned to product categories must be equal to the number of shelf space determined with the solution of shelf space allocation problem or retailer preferences.

$$\sum_{k=1}^{o} X_{j,k} = S_j, \quad \forall j \tag{8}$$

2. In order to display products together which belong to the same category, a product category must be assigned to each shelf space.

$$\sum_{j=1}^{m} X_{j,k} = 1, \quad \forall k \tag{9}$$

3. Categorical integrity of shelf combinations assigned to product categories is an indispensable element in terms of merchandising and aesthetic values in shelf space management, such as displaying of history or children books together. The condition that a shelf is either on the right, left or opposite of another shelf states two shelves can be assigned to the same product category. That's why, one of suitable shelf combinations must be chosen according to the number of shelf space area that will be assigned to product categories.

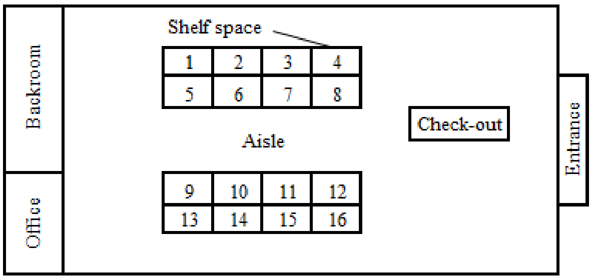


Fig. 1. A store shelf area layout for numerical example

For example; suppose that selection of 4 shelves for category A is needed according to the store layout plan presented in Figure 1. Shelf combinations such as {1,2,3,4} or {5,6,9,10} create an appropriate solution, while {5,6,8,12} or {3,4,7,8} states an inappropriate one.

The integrity of shelf combinations assigned to product categories must be controlled with the constraint below.

$$\sum_{k=1}^{o} X_{j,k} - \sum_{k=1}^{o}\sum_{t=1}^{o} X_{j,k}.X_{j,t}.LR_{k,t} \le 1, \quad \forall j \tag{10}$$

The objective of the model is to determine the shelf location of categories in store maximizing the retail's profit according to location effects based on position and proximity factors. While location effect based on position factor is related with the net profit value of category, location effect based on proximity factor is only related with cross sales effect (CS_i). In this case the objective function may be expressed as below:

$$MAX \sum_{j=1}^{m} NP_j.(LE_j + CS_j.LA_j) \tag{11}$$

Here, the value of cross sales effect can be calculated as below:

$$CS_j = \sum_{l=1}^{m} SR_{j,l}, \quad \forall j \tag{12}$$

The basic structure of the objective function of the developed model is as below:

1. It will tend to place product categories with higher expected profit to shelves with higher location effect.
2. It will place product categories with strong associate rules closer in order to support and increase cross sales effect between products.

3. Heuristic Approaches for Store Layout Problem

Due to NP-Hard nature of the model which is detailed in Section 2, heuristic or meta-heuristic methods should be employed to solve the large scale problems. In this study, firstly, a genetic algorithm based heuristic is proposed. In order to compare the performance of the genetic algorithm based heuristic with other methods, another heuristic approach based on tabu search and a simple rule that is commonly used by retailers are proposed.

3.1. *A modified genetic algorithm based approach*

If meta-heuristics such as particle swarm optimization, artificial bee colony, differential equation are applied in order to solve this layout problem, the obtained solutions do not satisfy the constraint defined in Eq. (10). In order to remain in the feasible solution space of the problem, neighborhood search heuristics and genetic algorithms can be used. Genetic algorithms offer significant opportunities by modifying crossover and mutation operators and the generation of the initial solution to provide good feasible solutions for the layout problem. Therefore, in this study, an original heuristic approach based on genetic algorithms has been developed for the solution of real life problem of a bookstore containing hundreds of shelves.

The main point which makes it difficult to implement genetic algorithm to discrete constrained optimization problems is the presence of epistatic genes. The change in values of these genes by crossover and mutation operations may be cause the violation of model constraints. At the same time, values of these genes affect values of other genes.

In the model presented in Section 2, the randomly determination of the initial solution, which is used in the classical form of genetic algorithm, cannot be applied in the proposed model, because of the location constraint in Eq. (10) which states the necessity to provide a certain integrity of shelf allocation for the same product category. Meanwhile, assignments of product categories to shelves also affect the existence of alternative feasible shelf combinations for other categories.

As mentioned above, shelf combinations such as {1,2,3,4} or {5,6,9,10} create an feasible selection, while {5,6,8,12} or {3,4,7,8} states an unfeasible selection for category A according to the store layout plan presented in Figure 1. On the other hand, since the selected shelves for category A cannot be used for other categories and the number and existence of feasible shelf combinations for other categories is affected from this selection. At the same time, the classical application of crossover and mutation operators of genetic algorithm will prevent the satisfaction of categorical integrity constraint defined in Eq. (10) and will cause inappropriate shelf combinations. In this respect, a modified genetic algorithm based approach which is differentiated in the generation of the initial solution and crossover and mutation operators from classical genetic algorithm, has been developed for the solution of store layout problem. This approach may be detailed as below:

Coding Structure

Throughout the developed approach, every individual in each population is represented as a permutation type. The first element of the solution set states the first shelf assigned to first category. That’s why element number of a feasible solution sets is equal to the number of shelves. In a store layout problem including 16 shelves and 6 categories, let the number of shelves for categories A,B,C,D,E,F are 4,4,3,2,2,1 respectively. Table 1 presents an appropriate solution example for genetic algorithm coding structure. According to this coding structure a code sequence such as 5-6-9-10-13-14-15-16-1-2-3-7-11-8-12-4 states that category A is assigned to shelves 5-6-9-10, category B is assigned to shelves 13-14-15-16, category C is assigned to shelves 1-2-3, category D is assigned to shelves 7-11, category E is assigned to shelves 8-12 and category F is assigned to shelf 4.

Table 1: An example of encoding for an individual

Category	A	A	A	A	B	B	B	B	C	C	C	D	D	E	E	F
Shelf	5	6	9	10	13	14	15	16	1	2	3	7	11	8	12	4

Fitness Function

In the developed approach, the objective function which is defined by Eq. (11) is used as fitness function to evaluate the population individuals.

Notations

In the developed approach, the notations used are as follows:

CSS	the category selection set
|*CSS*|	the number of elements of the category selection set
SC	the set of feasible shelf combinations
SC_p	the set of shelf combinations which includes *p* number of shelves
F_i	the fitness value of individual *i*
SP_i	the selection probability of individual *i*
G_{max}	maximum iteration number
N_s	the number of individuals created in population
PS	population size
P_e	elitism rate
P_c	crossover rate
P_m	mutation rate

Algorithm Steps

The main steps of the algorithm can be described as follows:

Step 1: Preparation Stage

This stage includes creation of necessary data sets before the beginning of the algorithm and calculation and definition of model parameters.

Step 1.1: Choose criteria that will be used to evaluate location effects based on shelf position. Determine performance values of shelves according to these criteria and calculate the location effect of the shelf position (LE_k) with the help of Eq. (1)-(3).

Step 1.2: Determine minimum distance values between shelves in the store layout ($D_{k,t}$).

Step 1.3: Find the support (%s) and confidence (%c) values of associate rules between product categories using Apriori algorithm, set rules and calculate the strength of these rules ($SR_{j,l}$).

Apriori algorithm is widely used for generating association rules. The usage of Apriori algorithm in the developed heuristics may be summarized as follows:

Firstly, there are frequent item-sets which have one element, meaning that they include one product category and provide the minimum support condition. Item-sets with one element which do not provide the minimum support condition are pruned and no new frequent item sets are produced from these pruned itemsets in the next stages. Frequent item-sets with two elements, which include two different product categories, are produced by frequent item-sets with one element which are found in the previous scan. Support values of candidate sets are calculated during the scan and the candidates which provide the minimum support condition become frequent item-sets. Then, confidence values of the found frequent item-sets are calculated and controlled as if they provide the minimum confidence condition. The basic idea of the Apriori algorithm is that if an itemset with element *k* provides the minimum support condition, then the subsets of this set also provide the minimum support condition. In the store layout problem, the generation process of frequent item-sets is ended upon the completion of the generation of frequent itemsets with 2 elements that satisfy the condition of being consistent with the developed model structure.

Step 1.4: Generate feasible shelf combinations according to number of shelves of product categories.

Let *SC* is the set of feasible shelf combinations. Here; under the condition that $\min(S_j) \leq p \leq \max(S_j)$, SC_p states the set of shelf combinations which includes *p* number of shelves for each element and is a subset of *SC* ($SC_p \subset SC$).

For example; as presented in figure 1, according to the sample store layout which includes 16 shelves, the appropriate set of shelf combinations for the product category, to which 4 shelves are required to assign, is composed of 7 different elements such as; SC_4= {(1,2,3,4),(5,6,7,8),(5,6,9,10),(6,7,10,11),(7,8,11,12), (9,10,11,12),(13,14,15,16)}.

Step 1.5: Define parameters such as population size (*PS*), elitism rate (P_e), crossover rate (P_c), mutation rate (P_m) and iteration number (G_{max}) of the genetic algorithm.

Step 2: Generating the Initial Solution

Step 2.1: Generate category selection set that includes all product categories ($j=1,...,m$).

Step 2.2: Select the category, which has the highest number of shelves to be assigned to, (j: $S_j \geq S_l$; $j \neq l$; $l=1,...,n$) and delete the chosen category from the selection set.

Step 2.3: Control the set of feasible shelf combinations according to the number of shelves that will be assigned to the chosen category ($SC_p | \; p=S_j$). Randomly select an element if number of elements for the SC_p is greater than 0. Else, go back to step 2.1 by deleting all shelf allocations.

Step 2.4: Delete all shelf combinations including at least one shelf of which the chosen element includes, and update sets of SC_p and *SC*.

Step 2.5: Control category selection set (*CSS*). If $|CSS|>0$ go back to Step 2.2. Else, increase the number of individuals ($N_s=N_s+1$) and keep the created individual.

Step 2.6: Control the number of the created individuals. If $N_s < PS$ go back to Step 2.1. Else, stop since the generation of the initial solution has been done.

Step 3: Calculation of the fitness values of individuals for the initial solution

Throughout the heuristic algorithm, the objective function given in Eq. (11) has been used as the fitness function. Calculation of the fitness value of individuals takes place as below.

Step 3.1: Calculate the location effect based on position in store for categories (LE_j) with the help of Eq. (4) by using the location effect based on position for shelves (LE_k), which is calculated in Step 1.1, and shelf allocation of categories ($X_{j,k}$) in Step 2.
Step 3.2: Calculate the location effect of the proximity to other categories for categories (LA_j) with the help of Eq. (6) by using distance values between shelves ($D_{k,t}$) in Step 1.2, the strength of associate rule between categories ($SR_{j,l}$), and shelf allocation of product categories ($X_{j,k}$) in Step 2 in which substep through 2.1-2.6.
Step 3.3: Calculate fitness values of individuals (F_i) with the help of Eq. (11) by using the location effect based on position (LE_j), which is calculated in Step 3.1, the location effect based on proximity (LA_i), which is calculated in Step 3.2, and net profit values of categories (NP_j).

Step 4: Generating new individuals and populations
Creating new individuals and populations is provided by elitism, crossover and mutation operators in order.

Step 4.1: Elitism
Elitism includes the direct transfer of certain number of individuals with the best fitness value into the next population, in order to prevent the state of non-existence of the best individuals in the population in result of crossover and mutation processes.

Step 4.1.1: Sort individuals in decreasing order according to fitness values (F_i).
Step 4.1.2: Take number of individuals equal to multiplication of population size and elitism rate into the next population without making any change.

Step 4.2: Crossover
Crossover includes selection of two individuals from the population and production of new individuals by characteristics of these two individuals. In the suggested approach, selection of parents is performed with the commonly used roulette wheel technique.

Step 4.2.1: Sort individuals in decreasing order according to fitness values (F_i).
Step 4.2.2: Calculate the selection probability (SP_i) of each individual by dividing the fitness value of each individual to the total fitness value using Eq. (13).

$$SP_i = F_i / \sum_{i=1}^{PS} F_i, \quad \forall i \tag{13}$$

Step 4.2.3: Calculate the cumulative selection probability of each individual (Q_i) by adding selection probabilities greater than itself to calculated selection probability.

$$Q_i = SP_i + \sum\nolimits_{SP_l > SP_i} SP_l, \quad \forall i \tag{14}$$

Step 4.2.4: Create a random number in the range [0,1]. Select the individual which corresponds to the smallest Q_i value and is greater than the random number as the parent 1.
Step 4.2.5: Create a second random number in the range [0,1]. Select the individual which corresponds to the smallest Q_i value and is greater than the random number as the parent 2. Check if the same two individuals are chosen or not. If the two individuals are same, delete the second parent allocation and repeat Step 4.2.5.
Step 4.2.6: Create a random number (r) in the range [0,1] for the crossover process. If $r < P_c$, start the process, else go to Step 4.3.

In the developed model structure, application of the classical crossover operators of genetic algorithm will violate the categorical integrity of assigned shelves to product categories and cause the allocation of unfeasible shelf combinations. In this respect, an original crossover operator for the proposed heuristic approach has been set out. The structure of this crossover operator is as below:

Step 4.2.7: Determine the last genes, in which shelf allocation for product categories has been performed, as crossover points. For example; in Table 2, the points 4,8,11,13,15 and 16, where the last shelf allocations for categories have been made according to the gene number, are determined as crossover points and multi-point crossover is applied.
Step 4.2.8: Copy genes which come to the first crossover point from parent 1 and genes which come to the second crossover point from parent 2 and generate the first child changing parents at the crossover points in

the order of parent 1-parent 2-parent 1-parent 2. If genes to be copied coming from parent 2 or parent 1 has been used before, select a random element from the set of shelf combinations according to the number of genes to be assigned to.

Table 2: Determining crossover points in the proposed approach

	A	A	A	A	B	B	B	B	C	C	C	D	D	E	E	F
Parent	5	6	9	10	13	14	15	16	1	2	3	7	11	8	12	4
Cross at				*				*			*		*		*	*

Step 4.2.9: Copy genes which come to the first crossover point from parent 2 and genes which come to the second crossover point from parent 1 and generate the offspring 2 changing parents at the crossover points in the order of parent 2-parent 1- parent 2-parent 1. If genes to be copied coming from parent 2 or parent 1 has been used before, select a random element from the set of shelf combinations according to the number of genes to be assigned to.

The process of generating new individuals with crossover operator may be explained with given example in Table 3:

Table 3: Generating new individuals with crossover operator

	A	A	A	A	B	B	B	B	C	C	C	D	D	E	E	F
Parent 1	5	6	9	10	13	14	15	16	1	2	3	7	11	8	12	4
Cross at				*				*			*		*		*	*
Parent 2	7	8	11	12	1	2	3	4	13	14	15	5	9	6	10	16
Offspring 1	5	6	9	10	1	2	3	4	13	14	15	7	11	8	12	16
Offspring 2	7	8	11	12	13	14	15	16	1	2	3	5	6	9	10	4

In the given example with Table 3, gene values of the first child, which come to the first crossover point, are taken from the first parent and 5-6-9-10 values are assigned. Gene values which come to the second crossover point are copied from the second parent and 1-2-3-4 values are assigned. Gene values which come to the third crossover point must be taken from the first parent. However, since 1,2,3 values which are the 9, 10 and 11. genes of the first parent have been used before, 13,14,15 values are assigned by randomly being selected from set of shelf combinations. Same scenario occurs for the fourth crossover point and thus 7,11 values are assigned with random selection. Section to the fifth crossover point is taken from the first parent and the section to the sixth crossover point is taken from the second parent so that the creation of the first offspring is completed. The creation of the second offspring has been done in a similar way using the order of parent 2-parent 1-parent 2-parent 1. As the explained in Step 4.2.1-4.2.3, new individuals are generated considering location effects and fitness values. Another important point is that while crossover points are selected randomly in the classical GA, the last genes of product categories are determined as crossover points in the proposed approach.

Step 4.3: Mutation

Mutation is a genetic algorithm operator used to provide diversity in population. Since a mutation operator, used in classical genetic algorithm applications, as in the form of randomly changing two genes of an individual, will prevent the fitness of created individuals to constraints, the mutation operator in the suggested heuristic algorithm is differentiated. Steps of the mutation process may be summarized as below:

Step 4.3.1: Generate a random number (r) in the range [0,1] for mutation process. If $r < P_m$, start the process, else go to Step 4.4.

Step 4.3.2: Randomly choose two categories with equal number of shelves to be assigned (S_j=S_l).

Step 4.3.3: Exchange the chosen category elements with each other.

The process of creating new individuals with mutation operator may be explained with the example in Table 4.

Table 4: Generating new individuals with mutation operator

	A	A	A	A	B	B	B	B	C	C	C	D	D	E	E	F
Parent 1	5	6	9	10	13	14	15	16	1	2	3	7	11	8	12	4
Offspring 1	5	6	9	10	13	14	15	16	1	2	3	8	12	7	11	4

In the given example in Table 4, categories of D and E, which have equal number of shelves to be assigned in Step 4.3.2, are chosen. In exchange of elements of these categories the individual 5-6-9-10-13-14-15-16-1-2-3-8-12-7-11-4 is created. While mutation points are selected randomly in the classical GA, the genes of two product categories which have equal numbers of shelves are changed in the proposed approach.

Step 4.4: Check the number of new individuals created in result of elitism, crossover and mutation. If N_s < PS, go back to Step 4.2. Create new individuals with crossover and mutation operators by repeating Step 4.2-Step 4.3.

Step 5: Calculation of fitness values of new individuals
Calculate fitness values of new individuals of the population created in result of elitism, crossover and mutation by applying processes defined in Step 3.

Step 6: Repeat of Step 4-5 until stopping criterion is satisfied
Repeat Step 2-4 until the determined maximum iteration number or the target fitness value is satisfied. When stopping criteria is satisfied, the fitness value of the objective function of the proposed approach for the developed model is obtained.

3.2. *A tabu search based approach*

In order to compare the performance of the genetic algorithm based heuristic with other methods, another heuristic approach based on tabu search is proposed.

As mentioned before, in order to satisfy the constraints of this layout problem, neighborhood search heuristics can be also used. At this point, tabu search based heuristic is proposed for solving the bookstore layout problem. Tabu search (TS) can be described as a neighborhood search method. It was introduced for combinatorial optimization problem by Glover.[19,20,21]

In the TS based heuristic, coding structure, fitness function and preparation stage are same with the step 1 of the GA based heuristic. The following steps of the tabu search based heuristic as follows:

Step 2: Generating the initial solution
Generate a initial solution using Step 2.1-2.5 of the GA based heuristic. Due to the fact that tabu search is not a population based algorithm, only a single individual is generated in the tabu search based heuristic.

Step 3: Move operator
Generate neighbouring solutions around the current solution by using one of two different move operators.

In the first of two move operators, two elements are selected and the places of these elements are exchanged in the permutation. In order to remain in the feasible solution space of the layout problem, the set of feasible element pairs is generated. At the same time, the candidate list is reduced with this set.

The generation of neighboring solutions with the first move operator may be explained with an example in Table 5. In this example, the set of feasible element pairs is determined as S= {(1,4), (7,12), (8,11)}.

Table 5: Generating neighboring solutions with first move operator in the tabu based heuristic

	A	A	A	A	B	B	B	B	C	C	C	D	D	E	E	F
Current solution	5	6	9	10	13	14	15	16	1	2	3	7	11	8	12	4
Neighboring solution	5	6	9	10	13	14	15	16	4	2	3	8	12	7	11	1

In the other move operator, generation of neighboring solutions is same with the mutation operator of the genetic algorithm based heuristic. A candidate list is generated using category pairs which have equal number of shelves. Neighboring solutions are generated by exchanging the elements of category pairs with each other. As an example, Table 4 can be examined.

In this step of the tabu search based heuristic, one of the two move operators is randomly selected and neighboring solutions is generated.

In the tabu search based heuristic, the notations used are as follows:

S_o	the best solution
S_c	the current solution
S_n	the neighboring solution
F_o	the fitness value of the best solution
F_c	the fitness value of the current solution
F_n	the fitness value of the neighboring solution
NM	the number of the non-improving moves
NM_{limit}	the upper limit of the non-improving moves

Step 4: Determine tabu tenure and generate an empty tabu list.
In the tabu list, the most recent reverse moves (element pairs or category pairs) that are exchanged are kept.
Step 5: Calculate the fitness value (F_n) of neighboring solutions using Eq. (11).
Step 6: Select the neighboring solution, which has the maximum fitness value from the candidate list. Delete the chosen neighboring solution from the candidate list.
Step 7: If the selected neighborhood move is prohibited on the tabu list, check aspiration criterion. If the neighborhood move passes the aspiration criterion, set S_c=S_n, otherwise go to the Step 6.
If the selected neighborhood move is not prohibited on the tabu list, set S_c=S_n, enter reverse move at the top of the tabu list, push all other moves in the tabu list one position down and delete the move at the bottom of the tabu list.
In the tabu search based heuristic, the aspiration criterion used is as follows: If a move leads to a neighboring solution that is better than the best solution encountered so far, the tabu status of the move is revoked.
Step 8: Compare the fitness value of the current solution and the best solution. If F_c>F_o, set F_o=F_c and S_o=S_c, NM=0 else NM=NM+1
Step 9: Diversification strategy
Check NM. If NM>NM_{limit}, reset the tabu list, go to the Step 2 and generate an initial solution.
Step 10: Termination criterion
If the best solution (S_o) has not been improved for a certain number of consecutive moves, the search terminates.

3.3. *A simple rule of thumb*

As a rule of thumb, net profit per shelf space area criteria can be used for product categories to solve the bookstore layout. This simple rule is commonly used by retailers in practice. At the same time, this rule is proposed by Chen and Lin[3] for shelf allocation decision in literature.

The steps of this simple heuristic can be described as follows:

Step 1: Calculate the average profit per shelf space (AP_j) using Eq. (15) for each category.

$$AP_j = NP_j / S_j, \quad \forall j \tag{15}$$

Step 2: Sort categories in decreasing order according to average profit per shelf space (AP_j).
Step 3: Generate category selection set that includes all product categories ($j=1,...,m$).
Step 4: Select the category, which has the maximum average profit number and delete the chosen category from the selection set.
Step 5: Control the set of feasible shelf combinations according to the number of shelves that will be assigned to the category selected in Step 4 (SC_p| p=S_j). Select the element, which has the maximum location effect if number of elements for the SC_p is greater than 0. Else, go back to step 3 by deleting all shelf allocations.
Step 6: Control category selection set (CSS). If |CSS| >0 go back to Step 4. Else, stop since the allocation to the shelves of the product categories has been done.
Step 7: Calculate the objective function value using Eq. (11).

In this simple heuristic, the more profitable categories are allocated on the shelves with higher location effects. However, the distances and adjacencies between product categories are not taken into account.

4. Experimental Design and Case Study

The applicability of the developed approaches has been presented with a case study performed in a bookstore including 137 shelves and 30 product categories. The store layout used in the case study is shown in Figure 2.

Before the application of the developed model for determining shelf locations, necessary model parameters must be calculated with the preparation stage of the heuristic approaches. At this point, firstly location effects of the positions of shelves (LE_k) defined in Step 1.1 have been calculated. In this respect, in order to determine location effects based on position, criteria such as proximity to check-out area, proximity to entrance and related number of shelves are chosen by the retailer. While calculating performance values and location effects of shelves according to this criteria, Eq. (1)-(3) have been used and values in the interval [0,1] have been changed to values in interval [1,1.3] in order to state the minimum location effect as 1, similar to the study of Hwang et al.[7] Positional location effects calculated for the 16 shelves are shown in Table 6. For example; for shelf 1, proximity to check out is 13.05 meter, proximity to entrance is 18.9 meter and number of visible shelves for this shelf location is 12. After, table of distance values between shelves ($D_{k,t}$), which has 18632 elements, has been formed according to the store layout plan. Here, distance values refer to the shortest distance between two shelves and may be described as $D_{k,t}= D_{t,k}$.

In order to determine associate rules between categories which is another step of the preparation stage, sales transaction data of the retailer for the last 3 month period, including 56910 different sales transaction data, has been used. At this point, minimum support (s_{min}) and minimum confidence (c_{min}) values are accepted as 0.5% and 2% respectively. A sample data set for the gained associate rules is presented in Table 7. Generating of the sets of the feasible shelf combinations for the product categories according to the store layout is the most important and time-consuming phase of the preparation stage. As a result, 1152 different feasible shelf combinations have been obtained for all product categories.

Preceding the solution of the store layout problem presented in Figure 2, small and medium sized test problems are derived from data set of this case study for the performance analysis of the developed approaches. Due to the novelty of this research, no test problems from the literature were identified.

The problem size is defined by two parameters: number of shelves and number of product categories. Three different problem sizes ((10,5), (30.10), (50,20)) are considered and 5 randomly selected numerical examples are generated for each problem size. For each problem, 10 independent replications are carried out.

The model structure in Section 2 is also coded in LINGO optimization software in order to evaluate the efficiency of the developed approaches. All tests were performed on Intel Core Duo 1.86 Ghz CPU with 1 GB RAM. The parameter values of the GA based heuristic are shown in Table 8.

The parameter values of the TS based heuristic are as follows: For (10,5), (20,10) and (50,20) sized test problems, tabu tenures are set to be 3, 5 and 7, respectively. Due to neighboring moves being limited with the feasible solution space, a higher value of the tabu tenure is not appropriate in the store layout problem. As a diversification strategy, after 20 consecutive non-improving moves, the tabu list is reset and the search restarts from the generation of the initial solution defined in Step 2. For (10,5), (20,10) and (50,20) sized test problems, the termination criteria are 500, 1000 and 5000 non-improving moves, respectively.

In order to compare the performances of the proposed approaches, average deviation from the best known solution (Δ_{avg}), maximum deviation from the best known solution (Δ_{max}) and average CPU time (T_{avg}) are used. The deviation from the best solution is the difference between the solution generated by the approach and the best solution of all the approaches. Average deviation from the best solution can be calculated by using Eq. (16).

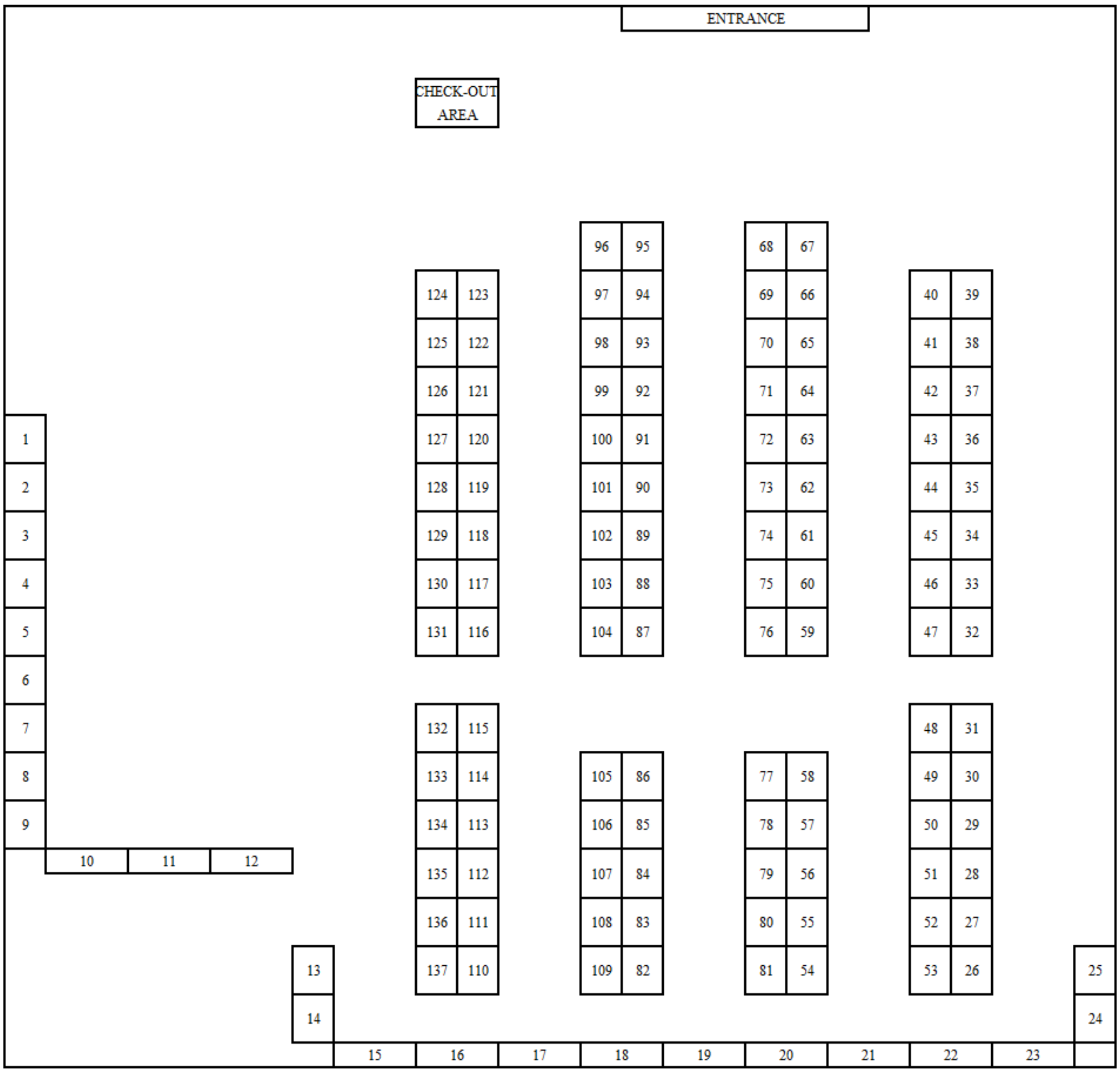


Fig. 2. Layout plan of bookstore retailer in the case study

$$\Delta_{avg} = \sum_{i=1}^{R}\left(\frac{(F_b - F_c)\times 100}{F_b}\right)\Big/ R \qquad (16)$$

In the Eq. (16), F_b denotes the best known solution, F_c is the obtained solution by using the heuristic approach and R is the number of replications.

Computational results and performance comparisons of the GA based heuristic and benchmark approaches are given in Table 9.

It can be observed from Table 9, LINGO software can only solve the small size (10,5) problems. For the second (30,10) and third (50,20) problems, the LINGO software is run for 24 hours and during this time 1 trillion iterations are done for each problem, but this software is not able to find an optimal solution. Numerical examples showed that LINGO had difficulties handling this model with medium or large size problem.

Table 6: Location effects of the positions of the shelves in the case study

Shelf	Proximity to Check-out	Proximity to Entrance	Related Number of Shelves	Location Effect (LE_j)
1	13.05	18.9	12	1.2690
2	13.95	19.8	13	1.2636
3	14.85	20.7	13	1.2482
4	15.75	21.6	13	1.2328
5	16.65	22.5	13	1.2174
6	17.55	23.4	13	1.2020
7	18.45	24.3	13	1.1866
8	19.35	25.2	13	1.1712
9	20.25	26.1	13	1.1559
10	20.25	26.1	14	1.1658
11	19.35	25.2	15	1.1911
12	18.45	24.3	17	1.2264
13	19.80	25.65	4	1.0741
14	20.70	26.55	4	1.0587
15	20.70	26.55	4	1.0587
…	…	…	…	…
137	19.80	25.65	3	1.0641

Table 7: Associate rules of product categories in the case study

Category 1	Category 2	Support Number	Support	Confidence	Rule Support
Category A	Category E	1201	0.0422	0.2427	0.0102
Category A	Category D	1201	0.0422	0.9290	0.0392
Category C	Category E	1145	0.0402	0.3837	0.0154
Category D	Category B	1145	0.0402	0.2176	0.0087
Category B	Category C	960	0.0337	0.3217	0.0108
Category E	Category B	960	0.0337	0.1974	0.0003

Table 8: Parameter values of GA based approach

Parameter	Values
Max. iteration number (G_{max})	1000
Population size (PS)	20
Elitism rate (P_e)	0.20
Crossover rate (P_c)	0.80
Mutation rate (P_m)	0.01

The results in Table 9 showed that the GA based heuristic is better than the two benchmark heuristics in terms of solution quality in all test problems. For (30,10) problem, maximum performance gap and average performance gap from the GA based heuristic of the TS based heuristic are 3.10% and 8.05%, respectively. Similarly, for the problem of (50,20), maximum and average performance gaps from the GA based heuristic of the TS based heuristic are 5.38% and 17.63%, respectively. Compared to the simple rule, the GA based heuristic has a significant performance advantage. For (30,10) and (50,30) size problems, the maximum performance difference between the two heuristics are 16.18% and 31.90% respectively. Similarly, for (30,10) and (50,30) size problems, the average performance difference between two heuristics are 7.56% and 13.54% respectively. These values indicated that the GA based heuristic can be efficient for large and also real world problems.

On the other hand, the TS based heuristic is superior to the GA based heuristic in terms of average CPU time. However, the performance difference between two heuristics are not considered important. The TS based heuristic is approximately 1.27 and 1.47 times faster than the GA based heuristic for the problems of (30,10) and (50,20) respectively. Although, the simple rule of thumb requires very low CPU time, the average and maximum deviation from the best known solution of this rule are very high.

Table 9: Performance comparisons of the GA based heuristic and benchmark approaches for test problems

Problem Size	Performance measures	LINGO	GA based heuristic	TS based heuristic	A rule of thumb
(10,5)	Average deviation from the best known solution (%)	0	1.26	1.32	4.37
	Maximum deviation from the best known solution (%)	0	3.55	6.95	10.52
	Average CPU time (min)	3.42	5.14	4.45	0.32
(30,10)	Average deviation from the best known solution (%)	-	0	3.10	7.56
	Maximum deviation from the best known solution (%)	-	0	8.05	16.18
	Average CPU time (min)	N/A after 24h	74.64	58.82	0.57
(50,20)	Average deviation from the best known solution (%)	-	0	5.38	13.24
	Maximum deviation from the best known solution (%)	-	0	17.63	31.90
	Average CPU time (min)	N/A after 24h	145.26	98.77	0.84

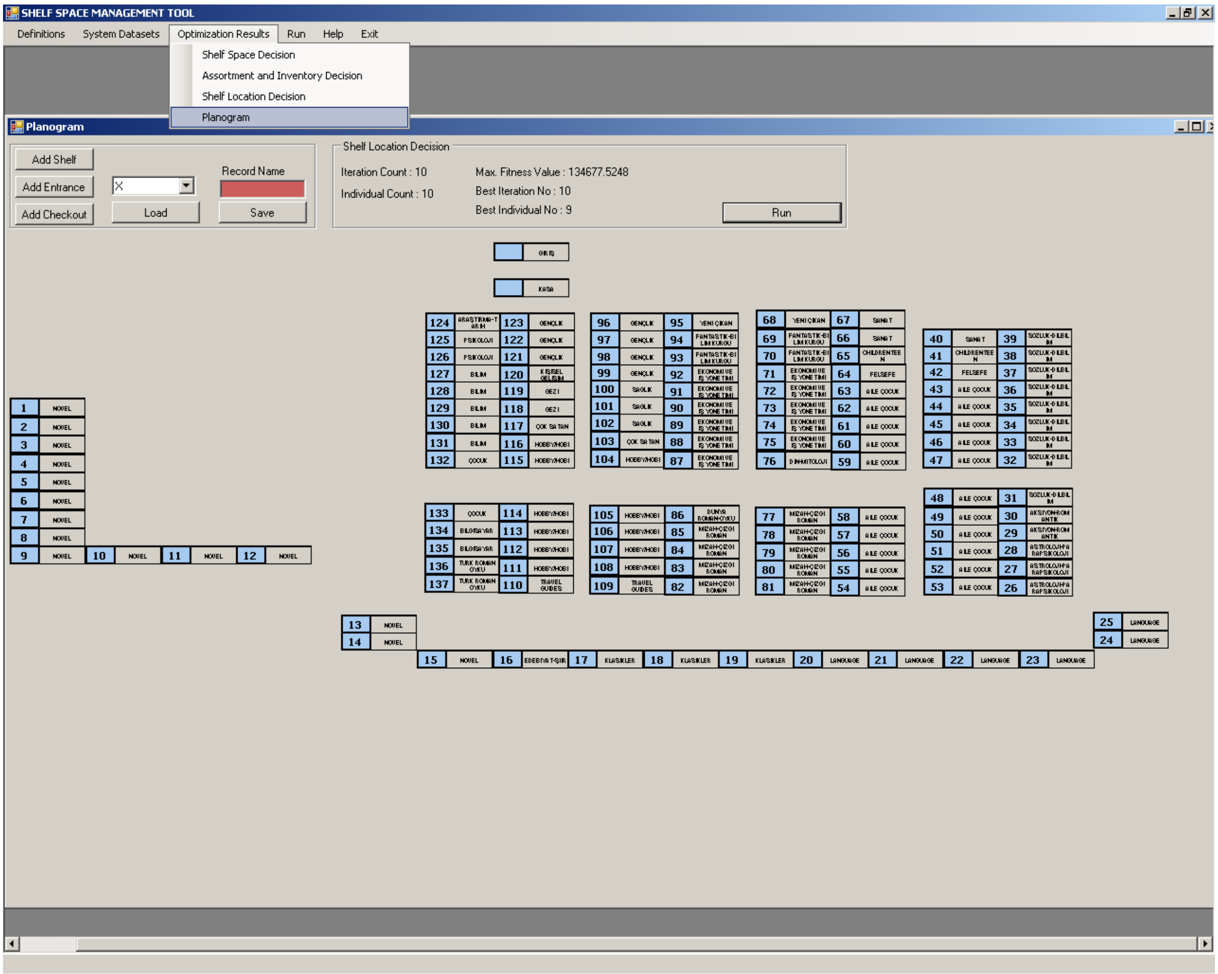
For the solution of the real life problem which is specified in Figure 2, the final step of the preparation stage includes describing genetic algorithm parameters such as the elitism rate, crossover rate, mutation rate, population size and maximum iteration number, in order to apply the heuristic algorithm. At this point, for GA based heuristic, parameter values are used in Table 8. In the case study, the parameter values of the TS based heuristic are as follows: Tabu tenure is set to be 10, termination criteria is 10000 non-improving moves. Additionally, after 20 consecutive non-improving moves, the diversification strategy is applied. The heuristic algorithms have been coded using SQL, which is consistent with database of the retailer.

Computational results and performance comparisons of the GA based heuristic and benchmark approaches are shown in Table 10 for the case study. According to these results, the GA based heuristic is 8.14% better than TS based heuristic and 18.72% better than the simple rule of thumb.

As a result of running the algorithm, the obtained best store layout is presented with the developed planogram software in Figure 3. In this study, the developed software provides a decision support tool for store layout decision to a book retailer by producing automated planograms. A planogram is basically a diagram or map that illustrates where product items and product categories should be displayed on the shelves.

Table 10: Performance comparisons of the GA based heuristic and benchmark approaches for the case study

Performance measures	GA based heuristic	TS based heuristic	A rule of thumb
Deviation from the best known solution (%)	0	8.14	18.72
Average CPU time (min)	282.72	166.81	1.34



Shelf Location Decision

Iteration Count : 10

Individual Count : 10

Max. Fitness Value : 134677.5248

Best Iteration No : 10

Best Individual No : 9

Fig. 3. The obtained store layout plan for the retailer by the GA based heuristic approach

5. Conclusions

The store layout decision has a bigger role in customer perception. A retail store must be designed in a way to provide the best sales-area efficiency and increase purchasing.[3] In this context, in this study, firstly, a new mixed integer mathematical programming model based on associate rules has been developed for a bookstore retailer in order to determine the locations of shelves, where products and product categories is displayed in store. Then, because of the NP-Hard nature of the developed model, an original heuristic approach based on genetic algorithms has been designed for the solution of large scale real life problems. In order to compare the performance of the genetic algorithm based heuristic with other methods, a simple rule widely used by retailers and another heuristic approach based on tabu search are proposed. The effectiveness of the proposed approaches is evaluated with a case study and numerical examples including different product category and shelf numbers. Computational results showed that in terms of solution quality, the GA based heuristic has the best performance among these approaches. For small scale problem, the average objective value of the model found by the GA based heuristic approach is 98.74% of the optimal solution which is obtained using LINGO software. These findings revealed that the developed GA based approach is efficient for providing near-optimal results at a reasonable time for large sized store layout problems.

In this study, the store layout model based on associate rules contributes to the creation of a layout plan which reflects customer preferences better with directly analysis of the customer transaction data. At the same time, different from other studies in the literature, the original crossover and mutation operators of the heuristic approach based on the genetic algorithm, which is suggested for the solution of the developed model, are presented. On the other hand, this study can be extended by developing more effective software solutions for reducing the solution time. Moreover, store layout problem is considered independently in this study. However, in practice, shelf space management problems such as store layout, shelf space allocation, product assortment, pricing are taken into account jointly by retailers. On this account, another possible extension is to develop an integrated model for these shelf space problems.

References

1. M.A. Hariga, A. Ahmari and A. A. Mohamed, A joint optimization model for inventory replenishment, product assortment, shelf space and display area allocation decisions, European Journal of Operational Research 181 (2007) 239-251.
2. Y. Chen, J. Chen and C. Tung, A data mining approach for retail knowledge discovery with consideration of the effect of shelf space adjacency on sales, Decision Support Systems 42 (2006) 1503-1520.
3. J. Zentes, D. Morschett and H. Schramm-Klein, Strategic Retail Management: Text and International Cases, 1st edn. (Springer Publishing, 2007).
4. X. Dreze, S. J. Hoch and M. E. Purk, Shelf management and space elasticity, Journal of Retailing 70(4) (2007) 301-326.
5. J. M. Hansen, S. Raut and S. Swami, Retail shelf allocation: A comparative analysis of heuristic and meta-heuristic approaches, Journal of Retailing 86 (2010) 94-105.
6. M. Yang, An efficient algorithm to allocate shelf space, European Journal of Operational Research 131 (2001) 107-118.
7. H. Hwang, B. Choi and M. Lee, A model for shelf space allocation and inventory control considering location and inventory level effects on demand, International Journal of Production Economics 97 (2005) 185-195.
8. H. Hwang, B. Choi and G. Lee, A genetic algorithm approach to an integrated problem of shelf space design and item allocation, Computers and Industrial Engineering 56 (2008) 809-820.
9. C. C. Murray, D. Talukdar and A. Gosavi, Joint optimization of product price, display orientation and shelf space allocation in retail category management, Journal of Retailing 86(2) (2010) 125-136.
10. M. Levy and B. A. Weitz, Retailing Management, 7th edn. (McGraw Hill Publishing, 2008).
11. R. S. Inglay and R. S. Dhalla, Application of systematic layout planning in hypermarkets, Proceedings of the 2010 International Conference on Industrial Engineering and Operations Management (Dhaka, Bangladesh, 2010)
12. A. Botsali and C. Peters, A network based layout design model for retail stores, Proceedings of the 2005 Industrial Engineering Research Conference (Atlanta, USA, 2005) pp. 1-6.
13. M. Chen and C. Lin, A data mining approach to product assortment and shelf space allocation, Expert Systems with Applications 39 (2007) 8611-8625.
14. C. A. Seruni, Design of product placement layout in retail shop using market basket analysis, Teknologi 9(2) (2005) 43-47.
15. I. Cil, Consumption universes based supermarket layout through association rule mining and multidimensional scaling, Expert Systems with Applications 39(10) (2012) 8611-8625.
16. H. Yapicioglu and A. E. Smith, Retail space design considering revenue and adjacencies using a racetrack aisle network, IIE Transactions 44(6) (2012) 446-458.
17. R. Agrawal, T. Imielinski and A. Swami, Mining association rules between sets of items in large databases, Proceedings of the ACM SIGMOD International Conference on Management of Data (Washington, USA, 1993) pp. 207-216.
18. T. Yuan and P. Chen, Data mining applications in e-government information security, Procedia Engineering 29 (2012) 235-240.
19. F. Glover, Future paths for integer programming and links to artificial intelligence, Computers and Operations Research 13 (1986) 533-549.
20. F. Glover, Tabu search part I, ORSA Journal on Computing, 1 (1989) 190-206.
21. F. Glover, Tabu search part II, ORSA Journal on Computing, 2 (1990) 4-32.