4OR-Q J Oper Res (2013) 11:31–55
DOI 10.1007/s10288-012-0211-2

RESEARCH PAPER

A new model and a hyper-heuristic approach for two-dimensional shelf space allocation

Ruibin Bai · Tom van Woensel · Graham Kendall · Edmund K. Burke

Received: 16 April 2012 / Revised: 4 August 2012 / Published online: 29 September 2012

Abstract In this paper, we propose a two-dimensional shelf space allocation model. The second dimension stems from the height of the shelf. This results in an integer nonlinear programming model with a complex form of objective function. We propose a multiple neighborhood approach which is a hybridization of a simulated annealing algorithm with a hyper-heuristic learning mechanism. Experiments based on empirical data from both real-world and artificial instances show that the shelf space utilization and the resulting sales can be greatly improved when compared with a gradient method. Sensitivity analysis on the input parameters and the shelf space show the benefits of the proposed algorithm both in sales and in robustness.

Keywords Shelf space allocation · Two-dimensional · Retail · Multi-neighborhood search · Hyper-heuristics

R. Bai (✉)
Department of Computer Science, University of Nottingham Ningbo China,
199 Taikang East Road, Ningbo 315100, China
e-mail: ruibin.bai@nottingham.edu.cn

T. van Woensel
School of Industrial Engineering, Technische Universiteit Eindhoven,
Den Dolech 2, Pav. F05, 5600 MB Eindhoven, The Netherlands
e-mail: t.v.woensel@tue.nl

G. Kendall
School of Computer Science, University of Nottingham, Nottingham, UK
e-mail: graham.kendall@nottingham.ac.uk

G. Kendall
School of Computer Science, University of Nottingham, Semenyih, Malaysia

E. K. Burke
Department of Computing Science and Mathematics, University of Stirling,
Cottrell Building, Stirling FK9 4LA, UK
e-mail: e.k.burke@stir.ac.uk

MSC classification 90B80 Discrete Location and Assignment

1 Motivation

Retailers aim to maximize availability of the goods in their product line at a minimum cost to operations. In the stores, these goals have to be realized on the shelves. Shelf spaces are expensive resources for many retail stores and supermarkets. Space planning and shelf operation is a major area that can significantly improve a retailer's financial performance and customer satisfaction.

One of the responsibilities of marketing is designing an attractive presentation on the shelves in order to attract customers. The amount of shelf space allocated to a product is primarily a consequence of marketing decisions: i.e. the merchandizing category to which the product is assigned and the allocated number of facings (the number of slots on the front of a retail shelf). Planograms (Bai and Kendall 2005) represent plans of how retail products will be laid out on the shelves, and show retailers where and how each stock keeping unit (SKU) should be displayed. SKUs are used to uniquely identify a specific product in location and its characteristics. It is the smallest management unit in a retail store. Planograms determine the available shelf space for the operations. From both an operations and a marketing point of view, it is thus important to produce high-quality planograms.

A planogram contains important information for the execution of the operations. In general, when constructing planograms the retailer decides on the assortment composition (which items are in the assortment?), the location of the item in the stores (where are the aisles, section, and shelf locations?) and the amount of space allocated to each item (how many facings and consumer units?). In an empirical study by Van Woensel et al. (2006), it was observed that planogram integrity is a serious issue for retailers. Planogram integrity is the degree to which the planograms are followed in practice. The empirical evidence clearly demonstrated that the majority of differences are mainly due to facing differences. In second and third place of importance, it was found that assortment and location differences have an impact. A major consequence of a lack of planogram integrity is the loss of a substantial level of efficiency both in terms of the marketing strategy as well as in the operational executions.

An important underlying driver for these planogram integrity problems, could be related back to the inability of the planogram tools to cope with real-life situations. The store managers tried to remedy this by taking into account a greater number of factors and changing the proposed planograms. In general, the motivation relates back to the issue that planograms are mainly built based on marketing decision rules (e.g. the number of facings per product category) and are often one-dimensional (i.e. do not take into account the depth nor height of the shelves). This paper takes a significant step towards the development of more efficient planograming tools. In this paper, we propose to model shelves as two-dimensional spaces, whilst considering the location effects of shelf space on the demand. Moreover, we proposed an efficient hyper-heuristic method for tackling this two-dimensional plangram problem.

The paper makes the following contributions:

– Most of the shelf space allocation models discussed in the literature treat shelf space one-dimensionally, quantified either in volume or in shelf length. That is,

the capacity of a shelf is usually quantified as a single real value (see e.g. Corstjens and Doyle 1981; Dreze et al. 1994; Urban 1998; Yang 2001; Hwang et al. 2005; Bai and Kendall 2008). These models are useful if all the shelves and items concerned have similar height dimensions and it is impossible to improve the space utilization by manipulating items across shelves. These models are also useful when items cannot be stacked together for various reasons (for example wine and milk bottles). Our paper presents a two-dimensional approach taking into account not only shelf length but also shelf height. In addition, the paper proposes a hyper-heuristic based approach which can potentially handle large-sized problem instances.

– Today, many items can be stacked together in order to make full use of the shelf space with regard to the height dimension, but this is not explicitly considered in current planogram optimization tools. Of course, in practice shelf operators can still stack items on top of the facings allocated according to a one-dimensional model. However, this usually leads to decreased space utilization and productivity. We show, via a rigorous scale of experiments, that explicitly taking into account the height dimension will improve the space utilization.

The paper is structured as follows: Section 2 discusses related work. Section 3 formulates our model of the shelf space allocation. Methodologies that are used to optimize the problem model are presented in Sect. 4. Numerical examples are presented in Sect. 5 along with sensitivity analysis on parameter estimation errors and Sect. 6 concludes the paper.

2 Related work

The study of shelf space allocation dates back to the 1960s with some empirical experiments being carried out to study the influence of shelf space operations. For example, Kotzan and Evanson (1969) made empirical studies for three products from eight chain stores and found that a significant relationship existed between shelf facings and sales. Cox (1970) carried out similar experiments for products from two brands of two categories, salt and coffee cream. He reported a very weak relationship between shelf facings and sales.

Space elasticity has been widely used to measure the responsiveness of the sales with regards to the change of allocated space. Curhan (1972) defined space elasticity as "*the ratio of relative change in unit sales to relative change in shelf space*" and reported an average value of 0.212. However, this is just an average value: the actual value of the space elasticity can be very different, depending on the products, stores and in-store layout (Curhan 1973). Cross-product effects were also investigated in Dreze et al. (1994) where space manipulations were made to enhance complementary shopping by placing complementary products together. The results showed that complementary merchandizing resulted a positive boost in sales (over 5 %) on the tested products (toothbrush, toothpaste and laundry care). It was also found that, compared with shelf facings, location had a larger impact as long as a minimum number of shelf facings (to avoid out-of-stocks) was maintained. Shelves at the eyesight level are more favorable than shelves located at the top or bottom of the shelf fixtures. Quantitative models have been proposed to describe the relationship between shelf space

and sales. Corstjens and Doyle (1981) firstly formulated their model in a non-linear multiplicative way and incorporated cross elasticities, a set of problem parameters that reflect the interrelationships between the different products under consideration. The inventory and handling cost effects were also considered. Two parameter estimation approaches were also compared and discussed. Zufryden (1986) proposed a dynamic programming model for the shelf space allocation problem, which also took into account some non-space factors such as price, advertising, promotion, store characteristics, etc. However, the approach may only be suitable for small sized problems and becomes computationally expensive for larger problem instances.

Recognizing that shelf space allocation is also closely related with other retailing problems, some integrated models have been proposed. For example, Borin et al.'s model (1994) could simultaneously provide a solution for both product assortment and shelf space allocation. A simulated annealing algorithm was used to address the proposed model. Urban (1998) integrated product assortment and inventory control with a traditional shelf space allocation model. A genetic algorithm was proposed to solve the problem. However, these models usually involve a large number of parameters. Obtaining a reliable estimation of these parameters is generally challenging and time-consuming. Therefore, it is difficult to put these models into a real application. A simplified linear model was proposed in Yang (2001), which considered the location effect of shelves on sales. The model, in fact, is a varied type of bounded multi-knapsack problem, which is generally difficult to solve. The authors designed several greedy heuristics to address the problem model.

However, these simple heuristics will only return a sub-optimal solution in many cases due to their inability to escape from local optimality. To solve the weakness of these simple heuristics, Lim et al. (2004) proposed several approaches, including a network flow procedure, tabu search and a modified squeaky-wheel algorithm. Experimental results, over several problem instances, showed that the modified squeaky-wheel algorithm is able to obtain a large improvement over the simple greedy heuristics used in Yang (2001). Recently, another shelf space allocation model was proposed in Hwang et al. (2005) which considers both the inventory and location influence on the demand. Both a gradient-based procedure and a genetic algorithm were used to solve the problem and experimental results on several small numerical instances showed that the genetic algorithm is able to obtain superior results. Bai and Kendall (2008) proposed an integrated inventory and shelf space allocation model specifically to handle fresh produce. A modified generalized reduced gradient approach was used to search for good quality solutions. Murray et al. (2010) and Hwang et al. (2009) have presented shelf space allocation models related to the one presented in our paper. Specifically, Hwang et al. (2009) looked into a simultaneous shelf space design and allocation problem. A mathematical model was developed and two solution procedures were proposed based on a genetic algorithm. This was applied to some very small test instances of four products. Murray et al. (2010) developed a model that jointly addresses a retailer's decisions for product prices, display facing areas, display orientations and shelf-space locations in a product category. This comprehensive model is solved by branch-and-bound procedures on some small test instances. Our paper is distinct by the use of hyper-heuristics and the ability to solve larger, more realistic sized instances.

Hyper-heuristics (Burke et al. 2003a; Ross 2005) have recently received increasing attention partly because one of their motivating goals is to establish a higher level of generality across different problem solving scenarios. A hyper-heuristic searches across a space of heuristics rather than the more usual heuristic approach of dealing with the solution space directly. Hyper-heuristics have successfully been applied to several difficult search problems, including timetabling and rostering (Burke et al. 2003b; Rattadilok et al. 2005; Bilgin et al. 2007; Bai et al. 2012), scheduling (Hart et al. 1998; Ouelhadj and Petrovic 2010), and packing and layout optimization (Terashima-Marin et al. 2005; Bai et al. 2008; Dowsland et al. 2007; Terashima-Marín et al. 2010).

3 Model development

In a similar way to that presented in Yang (2001), it is assumed that retailers can prevent going out-of-stock by improving their supply chain. Space allocation is carried out within a given category (space allocation between different categories is usually a strategic problem rather than a quantitative one). Shelves are multi-level with each level having a different impact factor. The problem can be formulated as follows: we are given a number of shelves m with each shelf j having a demand impact factor γ_j ($\gamma_j \geq 1$) and we are also given a number of n SKUs. The least popular shelves (i.e. either the bottom or the top shelves) have an impact factor of one and shelves that are close to "eye-level" have γ_i values greater than one. An SKU can be stacked on the top of another but is limited to being stacked within the same SKU. The problem is to allocate an appropriate number of facings to each SKU i such that the total sales of the SKUs are maximized. The notations used in the model are defined in Table 1.

For simplicity, we define *length facing* as the number of facings that are allocated to an item along the shelf length and *stack coefficient* as the maximum number of facings that can be allocated in height. Therefore, the total number of facings of an item is the product of length facings and the stack coefficient.

We use the same deterministic demand function proposed by Hwang et al. (2005). In their paper, they formulated the demand function as a multiplicative form of direct space effect, cross space effect and location effect:

$$F_i = \alpha_i \times s_i^{\beta_i} \times \left[\prod_{k \neq i}^{n} s_k^{\beta_{ik}} \right] \times \Upsilon_i \tag{1}$$

where s_i is the total number of facings allocated to item i. For a stacking coefficient value π_{ij}, s_i can be calculated as

$$s_i = \sum_{j=1}^{m} x_{ij} \times \pi_{ij} \tag{2}$$

and Υ_i the average location effect obtained as

$$\Upsilon_i = \frac{\sum_{j=1}^{m} x_{ij} \pi_{ij} \gamma_j}{s_i} \tag{3}$$

Table 1 List of variables used in the analysis

	Description
L_j	Length of shelf j
H_j	Height of shelf j
l_i	Length of SKU i
h_i	Height of SKU i
π_{ij}	Stack coefficient of item i on shelf j, and $\pi_{ij} = \lfloor \frac{H_j}{h_i} \rfloor$
F_i	Demand function of item i over time
M_i	Sales realized on item i
α_i	Scale parameter for demand function of item i
β_i	Space elasticity for item i
β_{ik}	Cross elasticity of item k on item i and β_{ik} is asymmetric
γ_j	Location impact factor of shelf j
p_i	Unit selling price of item i
$s_i^{\min}$	Minimum number of facings for item i
$s_i^{\max}$	Maximum number of facings for item i
x_{ij}	length facing of shelf j allocated to item i
x_i	Total shelf length facings allocated to item i, i.e. $x_i = \sum_{j=1}^{m} x_{ij}$
s_i	Total facings allocated to item i
y_{ij}	$y_{ij} = 1$, if item i is placed on shelf j
	$y_{ij} = 0$, if item i is not placed on shelf j
z_i	Shelf ID on which product i is displayed

The model can then be formulated as follows

$$\max \sum_{i=1}^{n} M_i = \sum_{i=1}^{n} p_i F_i \tag{4}$$

subject to:

$$\sum_{i=1}^{n} l_i x_{ij} \leq L_j, \forall j \tag{5}$$

$$y_{ij} \leq \pi_{ij}, \forall i, j \tag{6}$$

$$s_i^{\min} \leq s_i \leq s_i^{\max}, \forall i \tag{7}$$

$$x_{ij} \in \{0\} \cup Z^+, \forall i, j \tag{8}$$

$$y_{ij} \in \{0, 1\}, \forall i, j \tag{9}$$

$$y_{ij} \leq x_{ij}, \forall i, j \tag{10}$$

$$y_{ij} \times L_j / l_i \geq x_{ij}, \forall i, j \tag{11}$$

$$\sum_{j=1}^{m} y_{ij} = 1, \forall i \tag{12}$$

Table 2 A numerical example taken from Hwang et al. (2005)

Item$_i$	p_i	l_i	s_i^{min}	$s_i^{\max}$	α_i	β_{ik}			
						1	2	3	4
1	5.89	1	1	6	62.26	0.500	0.011	−0.014	−0.010
2	4.37	1	1	6	70.11	0.010	0.250	−0.009	0.014
3	4.91	1	1	6	66.92	−0.005	−0.006	0.400	−0.011
4	6.14	1	1	6	61.22	−0.008	0.009	−0.015	0.340

Shelf data: $m = 4$, $L_j = \{3, 3, 3, 3\}$, $\gamma_j = \{1.3, 1.2, 1.1, 1.0\}$

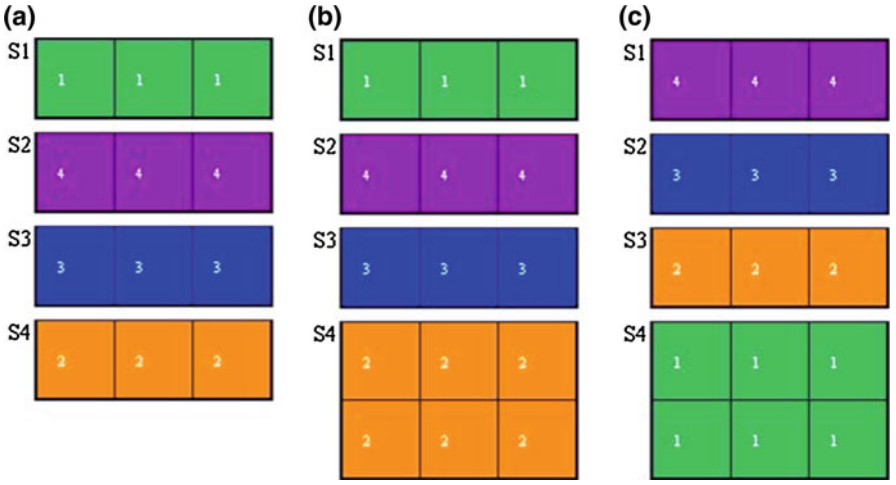

Fig. 1 A comparison of a 1D planogram with a 2D planogram. **a** Optimal solution for the 1D planogram. **b** The 1D planogram when shelf height changed. **c** Optimal solution of the 2D planogram

Constraints (5) and (6) make sure that the items allocated to each shelf do not exceed the capacity of the shelf (both in length and height). Constraint (7) defines the minimum and maximum facings which can be allocated to each item. Constraints (8), (9), (10) and (11) define the integrality of decision variables x_{ij} and y_{ij} and their relationships. Constraint (12) is a cluster constraint which ensures that the same type of items have to be displayed together on one shelf. The cluster constraint is not considered in previous models, which means that the same type of items can be displayed on several different shelves. In practice, retailers usually want to keep the same type of items in the same place to present a large attractive block. Therefore, this constraint is included in our model.

To illustrate the necessity to use a 2D model, we now consider one of the 1D numerical examples presented in Hwang et al. (2005). The example is drawn from the Dataset1 with 4 "mixed" items (i.e. including both substitutive and complementary items) and four shelves. For completeness, the details of the problem instance are also presented in Table 2.

The optimal solution (obtained by an exhaustive search procedure) for this 1D problem instance is displayed in Fig. 1a. As can be seen, this example assumes that

both shelves and items have unit height sizes. In reality, however, shelves are often in different heights (ranging from 100 to 1,000 mm based on our own data) and items can be stacked if the shelf space and other conditions allow it. Now suppose that the fourth shelf is two units in height instead of one. If the 1D planogram solution is still used, one would probably come up with a solution by stacking three more of item 2 onto the fourth shelf (corresponding to Fig. 1b). This solution gives an objective value of 2,492.55. However, the true optimal solution is 2,616.29 which is shown in Fig. 1c. The retailer could potentially lose 123.74 worth of sales in this case. This simple example shows that using a 1D planogram model can lead to considerable losses. It should be noted that, in this example, we only slightly changed the original data instance (one shelf only). In practice, both shelves and items may be of a different height, which justifies the necessity of using a two-dimensional planogram model, although item heights are not critical: (1) if the shelves are of the same height, the problems with items of different heights could be easily transformed into a one-dimensional problem; (2) even if the items are of the same height, the problems with shelves of different heights can be transformed into a one-dimensional problem.

4 Solution approaches and model extension

The problem presented in (4) subject to (5)–(12) is an integer nonlinear programming model with a complex form of objective function. The decision variables are x_{ij}. Due to the cluster constraint (12), each item can only be displayed on one shelf. Therefore, the solution can be represented by the following two vectors: let z_i be the shelf ID on which item i is displayed. Let x_i be the shelf length facings that are allocated to item i. This reduces the number of decision variables to $2n$. Given this, the problem is still very difficult to solve in a closed form. A complete search algorithm is also prohibitive even for a medium sized problem. We therefore turn our attention to heuristic approaches. This is also supported in the literature where heuristic approaches have been widely used to solve several shelf space allocation or related problems (Borin et al. 1994; Urban 1998; Yang 2001; Lim et al. 2004; Hwang et al. 2005; Bai et al. 2008).

4.1 A gradient approach

Because of the limited shelf space resources, items compete against each other for space. A gradient approach iteratively allocates shelf space to the items that can produce the largest reward per unit space in terms of the objective value (in this case, sales). However, due to the nonlinearity of the objective function, the ratio of sales to the allocated space varies with different shelf facing values s_i. The partial derivative of the objective function with regard to decision variables x_{uv} ($u = 1, \ldots, n$, $v = 1, \ldots, m$) is quite complex and can be presented as follows:

$$\frac{\partial \sum_{i=1}^{n} M_i}{\partial x_{uv}} = \sum_{i \neq u} \left[p_i \alpha_i \beta_{iu} \pi_{uv} s_i^{\beta_i - 1} s_u^{\beta_{iu} - 1} \prod_{k \neq i,u} s_k^{\beta_{ik}} \sum_{j=1}^{m} (x_{ij} \pi_{ij} \gamma_j) \right]$$

$$+p_u\alpha_u\pi_{uv}s_u^{\beta_u-2}\prod_{k\neq u}s_k^{\beta_{uk}}\left[(\beta_u-1)\sum_{j=1}^{m}(x_{uj}\pi_{uj}r_j)+s_u\gamma_v\right] \quad (13)$$

Due to the minimum facing constraint (7), most constructive heuristics for shelf space allocation problems adopt a two-phase procedure, which we also use in this paper. The first phase of the algorithm tries to allocate space to each item so that the minimum facing requirements are satisfied. In the second phase, the algorithm then repeatedly allocates the remaining space to the most preferable items according to some criteria. For example, the partial derivative function (13) would be a good candidate for this purpose. We can simply repeatedly increase the value of x_{uv} that gives the largest partial derivative value according to function (13), assuming there is still sufficient space available on shelf v. If this is not the case, the second largest partial derivative value is chosen, and so on. However, there is a problem. The existence of the cluster constraint (12) means that each item can only be displayed on one of the shelves. Therefore, once the first facing of an item is allocated to a given shelf, the remaining facings of this item have to be on the same shelf. This means that the first phase of the procedure is paramount in terms of the accommodating shelf for each item. If function (13) is adopted directly in the first phase of the algorithm, the algorithm tends to allocate the first facing of most of the items to the most attractive shelves and leaves some of the less attractive shelves empty. To solve this problem, the gradient algorithm used in this paper selects shelves based on the following rules, instead of function (13). At each iteration, the shelf with the largest residual length is selected, with ties broken by favoring larger shelf heights and then the location impact factor.

The algorithm can be described as follows:

S1. Initial Phase

S1.1 Select a shelf v with the largest residual capacity, with ties broken by favoring larger shelf height and then the larger location coefficient.
S1.2 For each item u that has not been initialized, set $s_u = s_u^{\min}$, $x_u = \lceil s_u/\pi_{uv} \rceil$, $s_u = x_u \times \pi_{uv}$, $z_u = v$.
S1.3 Calculate the partial derivative according to function (13), select the item that has the largest partial derivative value and that can be accommodated by the shelf v. Label this item to be initialized.
S1.4 If all items are initialized, go to S2, otherwise, go to S1.1.

S2. Iterative Phase

S2.1 For each item and its accommodating shelf, update the corresponding derivative value according to function (13). Sort the items according to the descending order of their derivative values.
S2.2 Select the first item u in the list, denote v be the shelf on which u is placed, set $x_u = x_u + 1$, $s_u = s_u + \pi_{uv}$. If this results in an infeasible solution, set $x_u = x_u - 1$, $s_u = s_u - \pi_{uv}$ and exclude this item from further consideration.
S2.3 Stop if no more facings can be allocated to any shelf, otherwise go to S2.1.

1: **Initialization**:
2: **Generate** an initial solution S_0, set $S = S\_0; S_{best} = S\_0$;
3: **Associate** each neighborhood operators N_i $(i = 1, ..., 5)$ three counters $C_i^{accept} = 0$,
4: $C_i^{new} = 0$, $C_i^{total} = 0$, a minimum weight $w_{\min}$ and set initial weight $w_i = w_{\min}$;
5: **Set** initial and stopping non-improving acceptance ratio r_s and r_e. Estimate the
starting temperature t_s and stopping temperature t_e by r_s and r_e respectively;
6: **Set** total iterations K, iterations at each temperature $nrep$ and the length of a
7: single learning period LP;
8: **Calculate** temperature deduction rate $\eta = (t_s - t_e) \cdot nrep/(K \cdot t_s \cdot t_e)$;
9: **Set** $t = t_s$; $t_{imp} = t_s$; $iter = 0$; $C_a = 0$; f_r=false;
10:**Iterative improvement:**
11:**Do**
12: **Select** a neighborhood (N_i) based on probability $P_i = w_i / \sum_{i=1}^{5} w_i$;
13: **Generate** randomly a candidate solution S' from current solution S in neighborhood N_i;
14: **Let** $\delta = obj(S) - obj(S')$, where $obj(S)$ and $obj(S')$) are the objective values of solutions S and S' respectively;
15: $iter++$; $C_i^{total}++$;
16: **If** $(\delta < 0)$
17: $S = S'$; $t_{imp} = t$; f_r=false; $C_i^{accept}++$; $C_i^{new}++$; C_a++;
18: **If** $(obj(S) > obj(S_{best}))$, $S_{best} = S$;
19: **If** $(\delta > 0)$
20: $C_i^{new}++$;
21: **If** $(\exp(-\delta/t) < random(0, 1))$ $S = S'$; $C_i^{accept}++$; C_a++;
22: **Endif**
23: **If** $(\delta = 0$ && new solution generated) S=S'; $C_i^{new}++$; $C_i^{accept}++$; C_a++;
24: **If** (f_r**=true**) $t_{imp} = t_{imp}/(1 - \eta t_{imp})$; $t = t_{imp}$;
25: **Else if** $(\mathtt{mod}(iter, nrep) = 0)$ $t = t/(1 + \eta t)$;
26: **Endif**
27: **If** $(\mathtt{mod}(iter, LP)$=0)
28: **If** $(C_a/LP < r_e)$
29: f_r=true; $t_{imp} = t_{imp}/(1 - \eta t_{imp})$; $t = t_{imp}$; $S = S_{best}$;
30: **For each** $i = 1, ..., 4$
31: **If** $(C_i^{total} = 0)$ $w_i = w_{\min}$;
32: **Else** $w_i = c_i^{new}/c_i^{total} + w_{\min}$
33: **Endif**
34: $C_a = 0, C_i^{accept} = 0, C_i^{new} = 0, C_i^{total} = 0$
35: **Endfor**
36: **Else**
37: **Foreach** $i = 1, ..., 4$
38: **If** $(C_i^{total} = 0)$ $w_i = w_{\min}$;
39: **Else** $w_i = C_i^{accept}/C_i^{total} + w_{\min}$
40: **Endif**
41: $C_a = 0, C_i^{accept} = 0, C_i^{new} = 0, C_i^{total} = 0$
42: **Endfor**
43: **Endif**
44: **Endif**
45:**Until** $iter > K$

Fig. 2 Pseudo code of the proposed multiple neighborhood search algorithm (Bai et al. 2012)

4.2 A multi-neighborhood approach

Metaheuristics (Glover and Kochenberger 2003) have been widely used to tackle highly constrained optimization problems from various fields. Multiple neighborhood search approaches have recently emerged as a popular meta-heuristic technique because of their ability to handle difficult constraints and obtain high quality solutions. Utilizing multiple neighborhoods could increase the accessibility of the search space and also improve the efficiency of the local search.

In this paper, a hybrid multiple neighborhood approach is proposed to solve this two-dimensional shelf space allocation problem. The multiple neighborhood approach utilizes a collection of neighborhoods in hybridization with a simulated annealing algorithm and a hyper-heuristic learning mechanism. This approach has shown considerable generality and competitiveness across large number of datasets of two very different combinatorial optimisation problems (bin packing and university course timetabling) (Bai et al. 2012). The increased generality is achieved by preventing the hyper-heuristic from using problem-dependent information. Figure 2 gives a pseudo-

code of this approach. Based on a given initial solution and a set of neighborhoods, this hyper-heuristic approach intelligently changes the neighborhood preferences during the search and the simulated annealing is used to determine whether a given neighborhood move suggested by the hyper-heuristic is accepted or rejected. More specifically, each neighborhood is associated with a weight w_i to represent its preference in comparison to the other neighborhoods. At each iteration, a neighborhood is stochastically ranked by the probability $p_i = w_i / \sum_{i=1}^{n} w_i$. Initially, weights are set to a predefined common minimum $w_{\min}$ and then updated periodically (defined by a learning period, or LP). The performance of each neighborhood in a given learning period is monitored and measured by two criteria, namely the acceptance ratio and the percentage of new solutions being generated. During normal "annealing" phase, the weight w_i is set to the acceptance ratio of each heuristic in the previous period and then stays unchanged during the learning period. If, however, at the end of the period the average acceptance ratio among all neighborhoods falls below a threshold, a "reheating" strategy is triggered to diversify the search and neighborhoods that are more likely to generate new solutions are favoured. Interested readers could refer to Bai et al. (2012) for more discussion about this approach and its relationship with some other metaheuristics, for example, iterated local search, variable neighbourhood search (VNS) and adaptive large neighborhood search approaches.

4.2.1 Neighborhood structures

The following neighborhood structures are used.

N1 **Swap:** This neighborhood includes all the neighboring solutions that can be generated by swapping one shelf length facing of two items i, k that are sharing the same shelf in the current solution. i.e. $x_i = x_i + 1, x_k = x_k - 1$.

N2 **Shift:** This neighborhood aims to improve the current solution by changing an item's current accommodating shelf. The neighborhood consists of all the solutions that can be generated by moving all the facings of an item from its current shelf to a different shelf.

N3 **Interchange:** This neighborhood enables the algorithm to interchange two items' (i, k) accommodating shelves if they are different (i.e. $z_i \leftrightarrow z_k$). This neighborhood is different from N1 in that it operates on the facings of items from two different shelves while N1 operates on the items' facings on the same shelf.

N4 **Add_Facing:** This neighborhood consists of all the neighboring solutions that can be generated by increasing the length facing x_i of item i by one.

N5 **Delete_Facing:** This neighborhood function deletes a length facing of a randomly selected item.

4.2.2 Constraint handling

Since not all of the above neighborhood moves produce feasible solutions, a mechanism has to be used to ensure that the algorithm searches within feasible regions of the search space. In this paper, the following method is used. For neighborhoods N1, N4 and N5 that involve adding/deleting facings to/from a single shelf, if a neighborhood

move leads to an infeasible solution, the move is rejected and the search goes back to the previous point. For neighborhoods N2 and N3, if the solution is not feasible because of violations of shelf length constraint (6), the length facing variables x_i that are involved in the moves are adjusted to their maximum possible values while attempting to satisfy all the constraints. If this fails to generate a feasible solution, the current neighboring solution is discarded and another neighboring solution is examined. The complete multiple neighborhood search algorithm is shown in Fig. 2.

In this application, the parameters of the multi-neighborhood algorithm are set as follows: the initial and stopping non-improving acceptance ratios are set as $r_s = 0.1$ and $r_e = 0.01$. The number of iterations at each temperature is set to be equal to the number of total neighborhoods used (i.e., $nrep = 5$). This implies that each neighborhood is sampled, on average, once at each temperature level if the neighborhoods are selected uniformly. We set the length of a single learning period $LP = 5{,}000$ and the minimum weight for each heuristic $w_{\min} = 1/n$. The total iteration count is set to different values based on the size of the problem instances. We set $K = 100{,}000$ for the small instance Pn6, $K = 500{,}000$ for the medium instance Pn29 (see Sect. 5.2), and $K = 2 \times 10^6$ for all the artificially generated instances (see Sect. 5.4).

4.3 Model extension: no clustering constraint

As mentioned earlier, the inclusion of a cluster constraint (12) simplifies the problem in a sense that the search can concentrate on a smaller search space. However, the solution procedures proposed in this paper can be easily adapted to the problem without this cluster constraint. For the gradient approach, choosing both the best shelf and item will be based on function (13) only. For the multi-neighbourhood approach, two things need to change: firstly, feasibility check procedure will be relaxed accordingly. Secondly, new neighourhood moves can be introduced which reallocate some of facings to another shelf. The overall framework and the other aspects of the algorithm will remain the same. This also highlights one of the advantages of the proposed multi-neighbourhood approach, which could adapt to the changes in a business environment. However, this may not be the case for some mathematical programming techniques.

5 Numerical examples

In this section, we describe the empirical data available and then present some examples based on this data.

5.1 Empirical input data

We use empirical data on daily sales, product attributes and available shelf space obtained from a European supermarket chain. We focused on dry groceries which are delivered from the retailers' distribution center. Median sales per SKU are observed to be around 3.8 customer unit/week. The merchandising categories which were included in the datasets are similar to the categories reported in Broekmeulen et al. (2007). The

Table 3 Problem instance with six items

Item$_i$	h_i	l_i	p_i	s_i^{min}	s_i^{max}	α_i	β_i	β_{ik}					
								1	2	3	4	5	6
1	143	47	6.62	3	10	45.80	0.88	–	−0.001	0.008	0	0	0
2	143	47	5.94	2	8	93.57	0.73	0.029	–	0.022	0	0	0
3	150	50	4.59	3	8	42.59	0.88	0.016	0.026	–	0	0	0
4	150	76	3.92	1	10	68.52	0.34	0	0	0	–	0	0
5	143	47	7.93	1	6	22.74	0.80	0	0	0	0	–	0
6	150	50	6.09	3	10	16.92	0.66	0	0	0	0	0	–

Shelf data: $m = 3$, $L_j = 500$, $H_j = \{150, 300, 150\}$, $\gamma_j = \{1.0, 1.1, 1.0\}$

experiment contained SKUs from 44 stores. Usually, promotions are done in the stores using other shelf space allocations. Therefore, promotions are eliminated to obtain the true mean and the standard deviation of the weekly sales. The supermarket chain reported the shelf capacity for each store-SKU combination. Unfortunately, the chain did not have data on the exact position of the products on the shelves, nor did they report on the specific sales prices of the SKUs due to reasons of confidentiality.

5.2 Two case studies

Due to data availability and commercial confidentiality limitations, only two instances, one small instance ($m = 3, n = 6$) and one medium-sized instance ($m = 5, n = 29$), were extracted from the data. For simplicity, they are denoted as Pn6 and Pn29. Detailed information of these two instances is provided in the Tables 3 and 4. The instances are based on a coffee category. Due to the limited data available, sales prices are drawn from the range [2.0,8.0] with even probabilities and cross elasticities were uniformly sampled within [−0.03,0.03] for the top ten SKUs (in sales). The elasticities for the other SKUs are set to be zero. Direct space elasticities β_i were estimated by simple linear regression based on the real-world data. As can be seen from Table 7, most of the direct space elasticities are in the range [0,1]. However, there are a few values outside this range. A negative value means that a large well displayed stock may lead to a decrease in demand (sometimes, a customer may associate poor quality or defects with some items that have large stocks). Some direct space elasticities are larger than one which means that allocating extra space to these items would lead to a larger increase in demand. Note that these estimated values are only valid within the corresponding facings ranges $[s_i^{\min}, s_i^{\max}]$. Once the allocated facings exceed this bound, the estimated direct elasticity values become invalid. All the other values were directly drawn from the real-world data.

Tables 5 and 6 present a comparison of solutions obtained by the gradient approach, the multi-neighborhood approach and the complete enumeration approach. All the algorithms were run on a PC with a Pentium IV 1.8GHZ CPU and 2GB RAM. Due to its stochastic nature, the multi-neighborhood approach was run 20 times for each problem instance (with the best solution and average objective value over 20 runs being presented in the tables). From Table 5, it can be seen that the gradient approach is very fast.

Table 4 Problem instance with 29 items

Item$_i$	h_i	l_i	p_i	s_i^{min}	s_i^{max}	α_i	β_i	β_{ik}									
								1	2	3	4	5	6	7	8	9	10
1	143	47	3.49	7	12	658.52	0.98	–	0.023	0.003	0.013	0.011	−0.030	0.029	0.018	0.021	−0.027
2	143	47	2.66	2	8	18.18	1.02	0.000	–	0.001	−0.014	0.029	0.011	−0.007	−0.004	0.010	0.010
3	143	47	4.80	8	16	685.40	0.35	−0.002	−0.016	–	−0.009	0.001	−0.020	−0.010	0.011	−0.007	−0.021
4	143	47	6.62	3	10	45.80	0.88	0.026	−0.025	0.028	–	−0.001	0.008	−0.028	−0.023	−0.018	−0.025
5	143	47	5.94	2	8	93.57	0.73	−0.012	0.002	−0.012	0.029	–	0.022	0.017	−0.017	−0.024	−0.020
6	150	50	4.59	3	8	42.59	0.88	−0.005	0.027	−0.006	0.016	0.026	–	−0.023	−0.001	−0.011	−0.027
7	143	47	4.12	3	10	125.21	1.06	−0.002	−0.009	−0.007	−0.008	−0.030	0.007	–	0.020	0.021	0.016
8	180	65	7.66	4	6	134.29	1.11	−0.025	−0.029	0.023	−0.017	0.022	−0.002	−0.012	–	−0.023	−0.023
9	180	65	2.60	7	10	70.81	1.12	−0.008	−0.002	−0.008	0.003	−0.012	−0.026	−0.020	−0.015	–	−0.004
10	153	56	5.85	3	8	671.83	0.15	0.015	−0.020	−0.022	0.021	0.029	−0.011	0.029	0.009	0.021	–
11	180	65	2.17	4	8	347.23	−0.08	0	0	0	0	0	0	0	0	0	0
12	150	55	3.48	3	8	122.73	0.95	0	0	0	0	0	0	0	0	0	0
13	150	76	3.92	1	10	68.52	0.34	0	0	0	0	0	0	0	0	0	0
14	143	47	7.93	1	6	22.74	0.80	0	0	0	0	0	0	0	0	0	0
15	150	50	6.09	3	10	16.92	0.66	0	0	0	0	0	0	0	0	0	0
16	143	47	5.92	2	4	646.55	−0.09	0	0	0	0	0	0	0	0	0	0
17	180	64	3.69	1	6	173.98	0.37	0	0	0	0	0	0	0	0	0	0
18	153	48	5.69	2	4	350.72	−0.56	0	0	0	0	0	0	0	0	0	0
19	143	48	6.22	4	8	219.20	0.18	0	0	0	0	0	0	0	0	0	0
20	150	47	6.21	1	8	43.82	0.84	0	0	0	0	0	0	0	0	0	0

Table 4 continued

Item$_i$	h_i	l_i	p_i	s_i^{min}	s_i^{max}	α_i	β_i	β_{ik}									
								1	2	3	4	5	6	7	8	9	10
21	150	45	7.69	2	6	56.26	1.12	0	0	0	0	0	0	0	0	0	0
22	145	50	2.56	1	8	56.79	0.25	0	0	0	0	0	0	0	0	0	0
23	150	47	2.96	1	10	21.07	0.40	0	0	0	0	0	0	0	0	0	0
24	150	47	4.29	2	4	104.58	0.29	0	0	0	0	0	0	0	0	0	0
25	150	50	6.78	2	8	330.30	0.25	0	0	0	0	0	0	0	0	0	0
26	180	65	3.01	3	10	138.38	0.37	0	0	0	0	0	0	0	0	0	0
27	180	65	6.76	5	10	319.28	0.06	0	0	0	0	0	0	0	0	0	0
28	150	45	3.84	1	8	13.20	0.77	0	0	0	0	0	0	0	0	0	0
29	150	45	6.93	1	8	3.16	0.98	0	0	0	0	0	0	0	0	0	0

Other cross elasticities $\beta_{ik} = 0$ Shelf data: $m = 5$, $L_j = 1600$, $H_j = \{180, 310, 150, 150, 300\}$, $\gamma_j = \{1.1, 1.3, 1.2, 1.1, 1.0\}$

Table 5 A comparison of solutions by different approaches for Pn6

Item$_i$	Gradient			Multi-neighborhood			Optimal solution		
	x_i	s_i	z_i	x_i	s_i	z_i	x_i	s_i	z_i
1	7	7	0	5	10	1	5	10	1
2	4	8	1	4	8	1	4	8	1
3	8	8	2	8	8	0	8	8	0
4	2	4	1	1	2	1	1	2	1
5	3	6	1	6	6	2	6	6	2
6	3	3	0	4	4	2	4	4	2
Objective (best/mean)[a]	8188.97			8975.50/8897.75			8975.50		
Deviation from optimum	8.8 %			0.9 %			–		
Time (in sec.)	<0.1			1.2			58.6		

[a] The best and mean results by the multi-neighborhood algorithm were based on ten independent runs. Both the gradient approach and the complete enumeration approach always obtained the same solution

For the small instance Pn6, our approach finds a solution in less than 0.1 s with an objective value that is 8.8 % away from the optimum. In contrast, the complete enumeration is quite slow, with a computational time of almost a minute. For Pn29, the complete enumeration approach (embedded with some simple bounding heuristics) failed even to find a feasible solution after 12 h of computational time. The multi-neighborhood approach seems to provide an appropriate compromise between the solution quality and the computational time. On average, among 20 runs, it obtains a solution that is only 0.8 % away from the optimum with much less computational time. For the small instance Pn6, the multi-neighborhood approach is able to find the optimal solution among 20 runs. For the instance Pn29, the multi-neighborhood approach could obtain 12.3 % ($=\frac{110640.14-97134.70}{97134.70}$) improvement over the gradient approach in terms of the objective function. Nevertheless, the multi-neighborhood approach consumes much more (but still a reasonable level of) CPU time. A graphical representation of the best solution by the multi-neighborhood approach is given in the Fig. 3.

5.3 Sensitivity analysis

5.3.1 Shelf space

Although shelf space is an expensive resource, in some cases, retailers may be able to invest more in space in order to increase sales. A question that retailers are interested in is how much extra sales can be realized by investing in more shelf space. The small instance is considered here because its optimal solution can be found reasonably quickly by the complete enumeration method. Figure 4 plots the relationship between the percentage of increased sales and the increase in shelf length (the shelves' height dimensions are the same). It can be seen that increasing shelf length has a positive effect on the sales. However, this relationship is extremely nonlinear. For example, when the shelf length is increased from 300 to 340, the sales increased by almost 15 %. However, expanding the shelf length from 780 to 1,000 does not increase sales at all.

Table 6 Solutions obtained by the gradient method and the multi-neighborhood approach for Pn29

Item$_i$	Gradient			Multi-neighborhood (MN2)		
	x_i	s_i	z_i	x_i	s_i	z_i
1	6	1	12	6	12	1
2	8	3	8	2	2	3
3	8	4	16	8	16	1
4	10	2	10	10	10	2
5	8	2	8	8	8	2
6	3	3	3	3	3	3
7	5	4	10	10	10	2
8	6	0	6	6	6	1
9	10	1	10	10	10	4
10	4	1	8	3	3	0
11	4	0	4	4	4	4
12	8	3	8	8	8	3
13	3	2	3	3	3	0
14	6	2	6	6	6	2
15	7	3	7	5	10	4
16	2	2	2	1	2	1
17	6	0	6	6	6	0
18	2	4	2	1	2	1
19	4	4	8	8	8	3
20	4	1	8	8	8	0
21	6	3	6	3	6	1
22	1	2	1	3	3	3
23	1	2	1	1	1	0
24	4	0	4	2	4	1
25	4	4	8	4	8	1
26	5	0	5	4	4	4
27	5	4	5	6	6	0
28	1	2	1	8	8	3
29	3	4	6	4	8	4
Objective (best/mean)[a]	97134.70			110640.14/110007.113		
Time (in sec.)	<0.5			43.4		

[a] The best and mean results by the multi-neighborhood algorithm were based on 20 independent runs. The gradient approach always returned the same solution

In general, when the shelf space is very limited, an increase in the shelf space has a larger impact on the sales than when the shelf space is already large. When the shelf length reaches over 1,400, it no longer has an impact on sales. This is partially due to the diminishing return demand function we used. In addition, the number of facings allocated to the items may reach the maximum allowed upper bounds, which define the valid ranges of the demand function.

Table 7 Average impact of errors in parameter estimation

α_i		β_i		β_{ik}	
$X\%$	$M_e\%$	$X\%$	$M_e\%$	$X\%$	$M_e\%$
2.7	1.28	2.5	2.31	3.2	1.52
5.4	2.57	5.6	4.61	6.4	1.49
7.6	3.80	7.4	7.59	9.6	1.53
10.6	5.48	10.4	7.03	12.9	1.63
14.2	6.61	12.3	9.37	16.3	1.54
16.8	7.14	15.2	11.67	18.9	1.82
17.7	13.76	16.0	16.94	23.0	1.82
21.2	10.64	17.4	12.69	26.0	1.67
22.4	12.13	20.5	12.19	27.0	1.83

$X\%$ mean absolute percent error in parameter estimation
$M_e\%$ mean percent deviation from optima

5.3.2 Sensitivity of the parameter estimation error

A number of parameters were estimated in the problem model such as, the scale coefficient (α_i), the space elasticity (β_i), and the cross elasticities (β_{ik}). This could introduce some estimation errors into the model. In this section, we analyze the effect of these estimation errors over the objective function. The small instance was used again due to its known optimal objective. The sensitivity analysis method is similar to the approach used in Borin and Farris (1995). For each estimated parameter set (α_i, β_i, or β_{ik}), a random error is added to each element of the considered parameter set. The errors were sampled from a normal distribution with mean values set to zero. The errors have mean 0, while the resulting parameters have means to be the true values and increasingly larger standard deviations. Let $X\%$ be the mean absolute percent error from the true values. The enumeration search was again carried out to obtain the global optimal objective values for both the original instance (without error) and the instance with parameter errors. Let M_{orig} and M_{error} be the corresponding optimal objective values. A relative objective error, $M_e\%$, was calculated by $|M_{orig} - M_{error}|/M_{orig}$. For each parameter set, this process was repeated 50 times. The average relative objective errors are presented in Table 7.

Figures 5, 6 and 7 plot the distributions of the resulted errors in sales with respect to the errors in three parameter sets: α_i, β_i, and β_{ik}. Each square box represents a given percentage error in sales. Hence, areas crowded with square boxes mean more values appearing in that range. The solid line plots the trend of the average relative error in sales across different parameter errors. The evidence suggests that the model is reasonably robust. In general, average errors in sales range from 1.28 % to less than 17 %. Larger errors in parameter estimation generally result in larger errors in sales prediction by the model, except for the cross elasticity. Within three parameter sets, the errors in α_i and β_i lead to larger errors in sales than the β_{ik}. Cross elasticities have least impact on the sales. This is probably due to the fact that their values are much smaller compared with α_i and β_i.

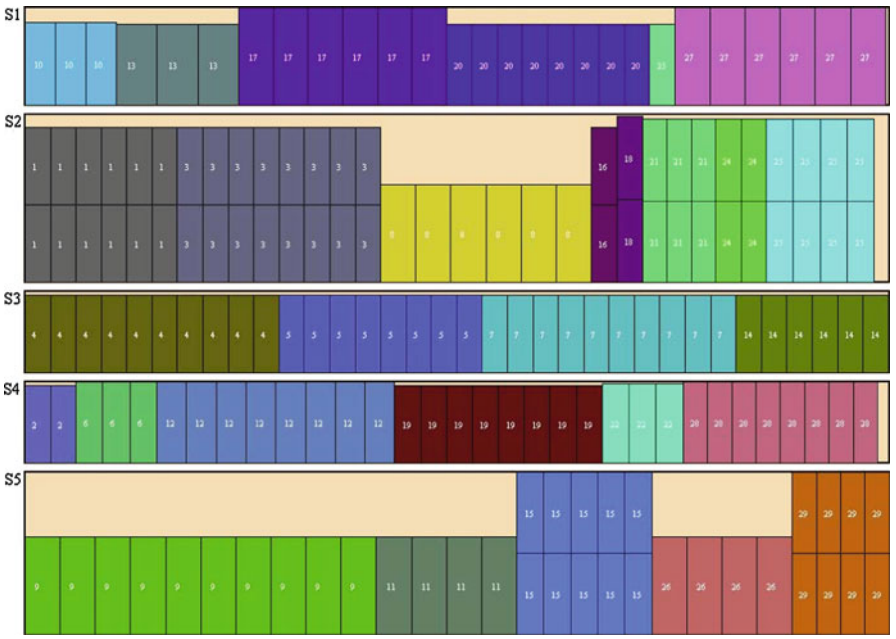

Fig. 3 The best solution obtained by the multi-neighborhood approach out of 20 runs

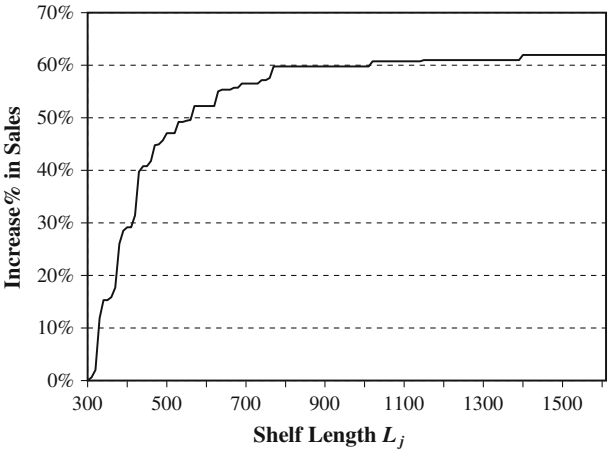

Fig. 4 The impact of shelf space over sales

5.4 Larger instances

In order to fully test the model and the solution methods, we generated two artificial data sets (Set1 and Set2), each of which is comprised by 20 larger instances. The parameters that are used to create these instances are the same as those used by Hwang

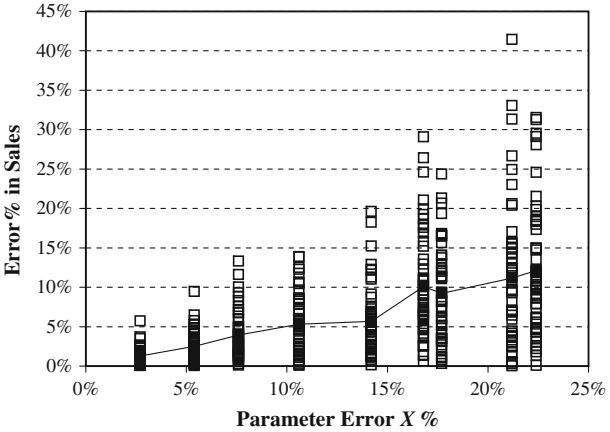

Fig. 5 Distribution of errors in sales with respect to errors in α_i

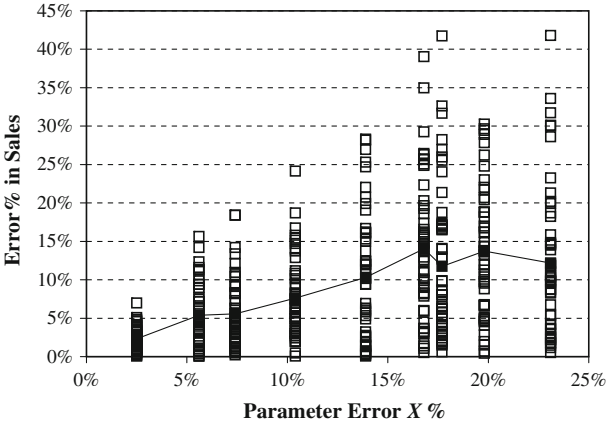

Fig. 6 Distribution of errors in sales with respect to the errors in β_i

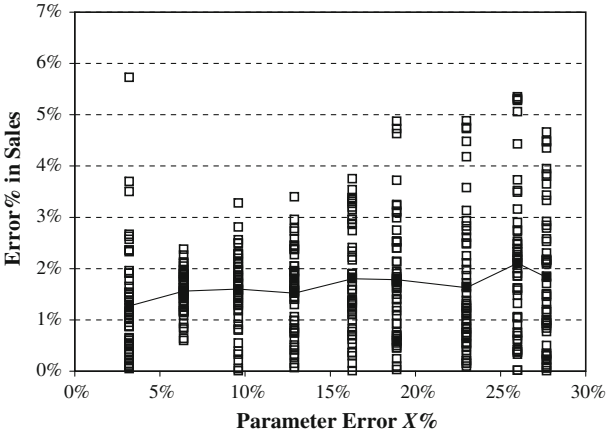

Fig. 7 Distribution of errors in sales with respect to the errors in β_{ik}

et al. (2009). Each instance in Set1 contains 50 items and five shelves (denoted as n50m5_01-n50m5_10). Instances in Set2 have 100 items and ten shelves (denoted as n1000m10_01-n100m10_10). These instances are publicly available for download from http://www.cs.nott.ac.uk/~rzb/files/2d-ssa-instances.zip.

Two versions of the simulated annealing hyper-heuristics were implemented and tested. In the first version (denoted as MN1), the neighborhoods are uniformly selected and hence does not have any adaptation. The second version is the one that we described in Fig. 2 where the probabilities of neighborhood calls are adaptively tuned during the search. All the other parameters for MN1 and MN2 are the same. In addition, we also implemented a first-decent VNS, consisting of two repetitive phases, local search phase and perturbation phase. The neighborhoods N1, N2, N3, N4 described in Sect. 4.2.1 were used as the first-descent local search neighborhoods and N5 was used as a perturbation method. The VNS algorithm successively searches through each of the local search neighborhoods (i.e. N1–N4) until it gets stuck at a local optimum. A perturbation is then applied to the current solution using neighborhood function N5 before the next iteration of the local search phase.

The following parameter settings were used: for both MN1 and MN2, we set $K = 2 \times 10^6$, $LP = 5{,}000$, which corresponds to around 280 s computational time for Set1 instances and 600 s for Set2 instances on the same machine that we used in the previous experiments. For VNS, computational time limit was set to 280 s for Set1 instances and 600 s for Set2 instances. Therefore, all the algorithms used approximately similar amounts of computational time. Each of the 20 instances was solved by each of these algorithms (MN1, MN2, VNS) 20 times from different random seeds.

Table 8 gives the computational results for these 20 instances by the gradient method, the VNS, the multi-neighborhood approach with random neighborhood selection (MN1), multi-neighborhood approach with adaptive neighborhood selection (MN2). A few observations can be made from the results: (1) The gradient method performed poorly, particularly for instances with 100 items and ten shelves. (2) The VNS does well for Set1 instances but is significantly inferior to MN1 and MN2 for instances in Set2. (3) In terms of average objective values, the hyper-heuristic approach with adaptive neighbourhood selection (MN2) performed generally better than the random neighbourhood selection (MN1). Nevertheless, the mean results by MN1 were better for four instances (e.g. n50_08, n100_02, n100_08, n100_10).

The two-tailed student's t-tests were also carried out to find out whether the proposed hyper-heuristic method (MN2) is statistically different from VNS or MN1. Table 9 presents the probabilities of the test results. We can see that, at 5 % confidence level, MN2 is significantly different from VNS for every instance. From both Tables 8 and 9, we can conclude that MN2 is significantly better than VNS except for 3 out 20 instances (n50m5_07, n50m5_08, and n50m5_10), for which VNS performed better. The differences in performance between MN1 and MN2 are not as obvious but overall MN2 is slightly better. For 8 out of 20 instances, there is no significant difference between them. For the remaining 12 instances, MN2 is significantly better than MN1 for 11 instances and MN1 outperformed MN2 statistically for one instance, n100m10_10.

Table 8 The performance of the proposed algorithm in comparison with other approaches

Instance	Gradient	VNS				MN1				MN2			
		Best	Worst	Mean	SD	Best	Worst	Mean	SD	Best	Worst	Mean	SD
n50m5_01	1383	2040	2011	**2020**	9	2040	1992	2010	11	2040	1992	2018	11
n50m5_02	1105	2101	2082	**2094**	5	2039	1975	2004	18	2039	1998	2018	10
n50m5_03	1651	2911	2718	2864	38	2896	2785	2840	24	2907	2816	**2870**	26
n50m5_04	2335	3972	3852	**3932**	26	3938	3832	3879	25	3951	3875	3910	22
n50m5_05	2685	4369	4267	**4314**	29	4335	4208	4260	40	4377	4253	4297	29
n50m5_06	2495	4636	4456	4558	53	4772	4544	4650	63	4799	4591	**4695**	51
n50m5_07	2369	4378	4251	4305	34	4412	4334	4364	24	4426	4333	**4376**	23
n50m5_08	1308	2013	1892	1975	24	2032	1998	2013	11	2046	2007	**2021**	10
n50m5_09	3512	7436	7232	7356	72	7616	7385	**7523**	70	7598	7281	7493	76
n50m5_10	3482	6487	6291	6394	54	6456	6187	6340	78	6513	6270	**6428**	61
n100m10_01	32128	347270	315775	330035	8583	395180	360286	376030	9118	401225	348535	**378953**	12505
n100m10_02	74214	814671	735019	760522	17479	889213	747627	**824317**	42779	874476	709707	820846	45283
n100m10_03	19611	234504	204919	218984	6998	262656	246531	254112	4424	262924	249455	**256943**	4674
n100m10_04	15915	113921	109302	111928	1153	121109	113795	117671	2080	121800	113079	**118918**	2583
n100m10_05	15392	179744	153959	170483	5447	202262	182786	194307	4426	200535	185548	**194736**	4166
n100m10_06	5755	30028	28506	29340	344	31277	29311	30616	545	31402	30033	**30678**	416
n100m10_07	23341	173881	130326	167358	9003	193604	175007	185598	5424	191584	177164	**185809**	3877
n100m10_08	26144	155829	145191	151104	3261	182590	159759	**171923**	6276	177540	154844	170636	6215
n100m10_09	8860	71113	64347	69226	1555	79285	74964	77283	1089	79812	76628	**78167**	979
n100m10_10	2781	23051	21678	22121	309	23606	22619	**23247**	272	23810	22225	22964	446
Average	12323	109218	97404	103546	**2724**	121066	107596	114749	3840	**120290**	**105332**	**114939**	4073

The gradient method was run once only since it is deterministic but 20 independent runs were carried out for each instance for VNS, MN1 and MN2. The best average results for each instance are highlighted in bold

Table 9 The probability results of the student's t tests between MN2, VNS and MN1

Set1	MN2 versus VNS (%)	MN2 versus MN1 (%)	Set2	MN2 versus VNS (%)	MN2 versus MN1 (%)
n50m5_01	2.4	4.2	n100m10_01	0.0	40.7
n50m5_02	0.0	0.8	n100m10_02	0.0	80.7
n50m5_03	4.5	0.2	n100m10_03	0.0	3.6
n50m5_04	0.0	0.0	n100m10_04	0.0	2.9
n50m5_05	0.0	0.2	n100m10_05	0.0	78.4
n50m5_06	0.0	1.1	n100m10_06	0.0	62.5
n50m5_07	0.0	10.2	n100m10_07	0.0	88.4
n50m5_08	0.0	3.2	n100m10_08	0.0	52.3
n50m5_09	0.0	18.8	n100m10_09	0.0	1.4
n50m5_10	1.7	0.0	n100m10_10	0.0	4.8

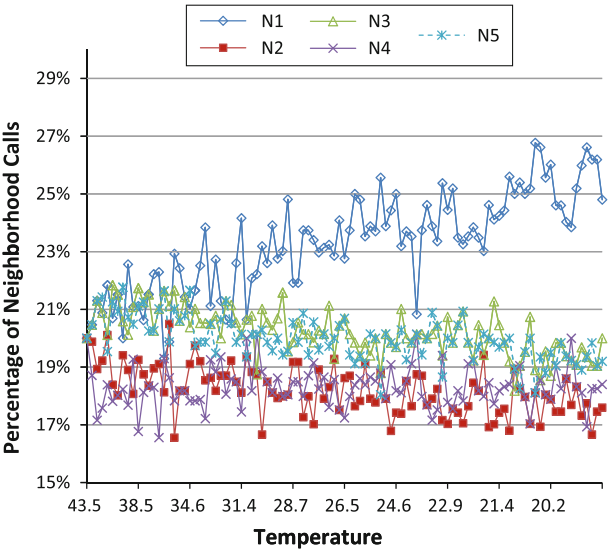

Fig. 8 Adaptation of neighbourhood calls at different annealing temperatures (for instance n50m5_01. Distributions for other instances are similar)

5.5 Adaptation of neighborhood selection

In our proposed algorithm, the selection of neighborhoods are based on an online learning mechanism. As we discussed in the previous section, this online selection mechanism provides a slightly better overall performance than the random uniform neighborhood selection for most instances. Figure 8 shows a typical dynamic adaptation process of different neighbourhood calls over different annealing temperatures. It can be observed that, although the probabilities with which neighborhoods were

chosen are the same at the beginning of the search, they are changed for different temperatures during the search. Overall, the first neighbourhood, N1, was selected more than the other neighborhoods. This trend intensifies when the annealing temperature decreases.

6 Conclusions

In this paper, we have presented a two-dimensional shelf space allocation model. The contributions of the paper are two fold: (1) rather than focusing on a single dimension, shelf length, the proposed shelf space allocation model adds the height dimension to the shelf space allocation decisions. As such, the new model is much more realistic than one-dimensional models. (2) An efficient simulated annealing hyper-heuristic approach is proposed for this 2D shelf space allocation problem. This approach can flexibly be adapted to other types of shelf space allocation problems (e.g. three-dimensional versions) thanks to the increased generality of hyper-heuristics compared to other heuristic methods. The performance of the algorithm was tested on two real-world instances and 20 artificial instances.

We have showed via a small example used in a previous study, that explicitly taking into account the height dimension will improve shelf space utilization and the resulting sales. Sensitivity analysis has shown the benefits of the proposed algorithm: if one has allocated limited shelf space to an SKU, a shelf space increase will have a relatively large impact on sales than if one already has abundant shelf space allocated. In addition, the shelf space allocation model is fairly robust against input parameter errors.

References

Bai R, Kendall G (2005) An investigation of automated planograms using a simulated annealing based hyper-heuristics. In: Ibaraki T, Nonobe K, Yagiura M (eds) Metaheuristics: progress as real problem solvers. Springer, Berlin, pp 87–108

Bai R, Kendall G (2008) A model for fresh produce shelf space allocation and inventory management with freshness condition dependent demand. Inf J Comput 20(1):78–85

Bai R, Burke EK, Kendall G (2008) Heuristic, meta-heuristic and hyper-heuristic approaches for fresh produce inventory control and shelf space allocation. J Oper Res Soc 59:1387–1397

Bai R, Blazewicz J, Burke EK, Kendall G, McCollum B (2012) A simulated annealing hyper-heuristic methodology for flexible decision support. 4OR Q J. Oper Res 10:43–66

Bilgin N, Özcan E, Korkmaz E (2007) An experimental study on hyper-heuristics and exam timetabling. In: Practice and theory of automated timetabling VI, Lecture Notes in Computer Science, vol 3867. Springer, pp 394–412

Borin N, Farris P (1995) A sensitivity analysis of retailer shelf management models. J Retail 71(2):153–171

Borin N, Farris PW, Freeland JR (1994) A model for determining retail product category assortment and shelf space allocation. Decis Sci 25(3):359–384

Broekmeulen RACM, Fransoo JC, van Donselaar KH, van Woensel T (2007) Shelf space excesses and shortages in grocery retail stores. Technical report, Eindhoven University of Technology, The Netherlands

Burke EK, Hart E, Kendall G, Newall J, Ross P, Schulenburg S (2003a) Hyper-heuristics: an emerging direction in modern search technology. In: Glover F, Kochenberger G (eds) Handbook of metaheuristics. Kluwer, Dordrecht, pp 457–474

Burke EK, Kendall G, Soubeiga E (2003b) A tabu-search hyperheuristic for timetabling and rostering. J Heuristics 9(6):451–470

Corstjens M, Doyle P (1981) A model for optimising retail space allocations. Manag Sci 27:822–833

Cox K (1970) The effect of shelf space upon sales of branded products. J Mark Res 7:55–58

Curhan R (1972) The relationship between space and unit sales in supermarkets. J Mark Res 9:406–412
Curhan RC (1973) Shelf space allocation and profit maximization in mass retailing. J Mark 37:54–60
Dowsland KA, Soubeiga E, Burke EK (2007) A simulated annealing based hyperheuristic for determining shipper sizes for storage and transportation. Eur J Oper Res 179(3):759–774
Dreze X, Hoch SJ, Purk ME (1994) Shelf management and space elasticity. J Retail 70(4):301–326
Glover F, Kochenberger GA (eds) (2003) Handbook of metaheuristics. Springer, Berlin
Hart E, Ross P, Nelson JA (1998) Solving a real-world problem using an evolving heuristically driven schedule builder. Evolut Comput 6(1):61–80
Hwang H, Choi B, Lee M-J (2005) A model for shelf space allocation and inventory control considering location and inventory level effects on demand. Int J Prod Econ 97(2):185–195
Hwang H, Choi B, Lee G (2009) A genetic algorithm approach to an integrated problem of shelf space design and item allocation. Comput Ind Eng 56:809–820
Kotzan J, Evanson R (1969) Responsiveness of drug store sales to shelf space allocations. J Mark Res 6:465–469
Lim A, Rodrigues B, Zhang X (2004) Metaheuristics with local search techniques for retail shelf-space optimization. Manag Sci 50(1):117–131
Murray CC, Talukdar D, Gosavi A (2010) Joint optimization of product price, display orientation and shelf-space allocation in retail category management. J Retail 86:125–136
Ouelhadj D, Petrovic S (2010) A cooperative hyper-heuristic search framework. J Heuristics 16(6):835–857
Rattadilok P, Gaw A, Kwan R (2005) Distributed choice function hyper-heuristics for timetabling and scheduling. In: Practice and theory of automated timetabling V, Lecture Notes in Computer Science, vol 3616. Springer, Berlin, pp 51–67
Ross P (2005) Hyper-heuristics. In: Burke EK, Kendall G (eds) Search methodologies: introductory tutorials in optimization and decision support techniques, Chap 17. Springer, Berlin, pp 529–556
Terashima-Marin H, Flores-Alvarez EJ, Ross P (2005) Hyper-heuristics and classifier systems for solving 2D-regular cutting stock problems. In: Proceedings of the 2005 ACM conference on genetic and evolutionary computation (GECCOR 2005), pp 637–643
Terashima-Marín H, Ross P, Farías-Zárate CJ, López-Camacho E, Valenzuela-Rendón M (2010) Generalized hyper-heuristics for solving 2D regular and irregular packing problems. Annals Oper Res 179(1):369–392
Urban T (1998) An inventory-theoretic approach to product assortment and shelf-space allocation. J Retail 74(1):15–35
Van Woensel T, Broekmeulen RACM, van Donselaar KH, Fransoo JC (2006) Planogram integrity: a serious issue. ECR J 6:4–5
Yang M-H (2001) An efficient algorithm to allocate shelf space. Eur J Oper Res 131:107–118
Zufryden F (1986) A dynamic programming approach for product selection and supermarket shelf-space allocation. J Oper Res Soc 37(4):413–422