Available online at www.sciencedirect.com
ScienceDirect
European Journal of Operational Research 184 (2008) 480–489
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

Production, Manufacturing and Logistics

Packing items to feed assembly lines

Mauricio C. de Souza [a,*], Carlos R.V. de Carvalho [a], Wellington B. Brizon [b]

[a] *Departamento de Engenharia de Produção, Universidade Federal de Minas Gerais, Rua Espirito Santo 35, cep 30160-030, Belo Horizonte, MG, Brazil*
[b] *Métodos de Logística, FIAT Automóveis s/a, Betim, MG, Brazil*

Received 18 February 2005; accepted 8 September 2006
Available online 18 January 2007

Abstract

We treat a practical application of packing problems in feeding assembly lines. This study was motivated by a real situation encountered in the shop floor of a major automobile industry plant in Brazil. The assembly line feed problem (LFP) consists in how pack the items in the available containers to meet the line work centers' requirements with a minimum total cost over the planning horizon. LFP is a variable-sized bin packing problem that has two special features: (i) a cardinality constraint on each bin's size; and, (ii) a cost structure such that each bin's cost varies according to the items that are packed in it. We propose an integer programming model and a GRASP heuristic for LFP. Numerical results on real-life test instances are reported.

Keywords: Packing; Metaheuristics; Assembly lines; Shop floor logistics; GRASP

1. Introduction

High-volume discrete manufacturing systems are traditionally organized in flow lines to achieve productivity gains and cost savings. Such flow lines have been widely used for assembly operations in the automobile industries. In these cases, the manufacturing systems are composed by independent parallel lines – each one dedicated to a different product family – having a serial arrangement of work centers (see Buzacott and Shanthikumar [5] for a comprehensive treatment of flow line systems). As we are concerned with assembly lines, a feed process must supply the work centers with all the necessary purchased or intermediate items to complete the required operations. Important managerial decisions in controlling assembly line systems are related therefore to their feed process to meet a time based (daily for instance) demand. The choice among possible feeding line policies depends upon their incurred costs that typically have two components: (i) holding costs; and, (ii) handling costs. In this paper, we are interested in modelling and solving packing problems encountered in assembly line feed processes.

Packing problems arise in the assembly line feed process because standardization of work methods

* Corresponding author.
E-mail addresses: mauricio.souza@pesquisador.cnpq.br (M.C. de Souza), carlos@dep.ufmg.br (C.R.V. de Carvalho), wellington.brizon@fiat.com.br (W.B. Brizon).

0377-2217/$ - see front matter
doi:10.1016/j.ejor.2006.09.091

plays a central role in achieving high productivity objectives. The materials handling from the storage area to the assembly line is then done in standard containers. In order to effectively control the system the containers should always carry the same quantity of items – as it is actually done in Just-In-Time systems (see for instance [17]). The use of nonstandard containers or irregularly filled containers disrupts the production flow through the assembly line.

The present study was conducted in a major automobile industry plant located in Brazil. The company has currently a remarkable position in the Brazilian market with a share of 25.3% in 2003. Moreover, it is one of the main exporters in Brazil. The plant's capacity is evaluated in 2300 vehicles a day. The whole production process comprises five stages: mechanics, presses, body shop, painting, and final assembly. The final assembly is composed by four independent lines where vehicles are aggregated according to its characteristics, types, and similar assembling pieces. The system operates in paced lines, this means that a work-in-process vehicle passes from one work center to another within a cycle time (the standard specified amount of time that a product must be turned out from a work center).

A feed process must supply the work centers with all the necessary purchased or intermediate items to complete the required operations within a cycle time. To allow trade-off policy possibilities between handling costs and holding costs, a certain number of different containers may be available. This includes plastic and metallic recipients and pallets – each one of these containers being available in different sizes. The problem faced when feeding an assembly line is how pack the items in the available containers to meet the work centers' requirements with a minimum total cost over the planning horizon.

This problem – LFP for short – is a variable-sized bin packing problem that has two special features: (i) a cardinality constraint on each bin's size; and, (ii) a cost structure such that each bin's cost varies according to the items that are packed in it. Friesen and Langston [15] presented, to our knowledge, one of the first approaches to the variable-sized bin packing problem. In this case, an unbounded number of bins from a finite collection of bin sizes are available to pack a set of items. The objective is to minimize the total space used in the packing. The authors proposed three efficient approximation algorithms – based on the First Fit Decreasing and Next Fit strategies – with guaranteed worst-case performance bounds. Some recent approaches for the variable-sized bin packing problem include the particular setting in metal cutting industries treated by Chu and La [6]. The authors proposed four greedy approximation algorithms based on the concept of absolute and relative waste. Kang and Park [16] presented two greedy algorithms – based on the First Fit Decreasing and Best Fit Decreasing – and analyzed then for three special cases: (i) the sizes of items and bins are divisible, respectively; (ii) only the sizes of bins are divisible; and, (iii) the sizes of bins are not divisible. Seiden and van Stee [24] established new upper and lower bounds for multidimensional generalizations of bin packing, which includes the d-dimensional variable-sized bin packing. Alves and Valério de Carvalho [1] studied strategies in stabilizing and accelerating column generation methods designed for the variable-sized bin packing problem. For this purpose, the authors introduced new dual-optimal inequalities and explored the principle of model aggregation. The online variant has been also treated in the literature, see for instance [8,9,26].

Industrial applications related to LFP include the modelling of pull type Just-In-Time systems. Mathematical programming approaches to pull type systems have been proposed by Bitran and Chang [4] and Watanabe and Hiraki [25], among others, for the purpose of reducing the in-process inventory and improving product quality. In these cases, a mathematical model assists managers in determining the number of circulating Kanbans in the system.

In the next section, we formulate the assembly line feed problem as an integer programming model. We propose a GRASP heuristic for LFP in Section 3, where we give a description of its construction and local search phases. In Section 4, we report numerical results on real practical instances from the studied automobile industry's shop floor. Concluding remarks are made in the last section.

2. Assembly line feed problem

For every work center, a production schedule determines how many of each item is necessary to complete its operations within specified periods of a finite horizon. Therefore, we have a demand d_{it} for item i, $i = 1, \ldots, I$, with which the system must be supplied in period t, $t = 1, \ldots, T$, of the finite horizon. As mentioned in the previous section, we

have $k = 1, \ldots, K$ different container's sizes available to feed the assembly line. A handling cost b_k is associated with a container of size k. This cost is estimated as the ratio that the amount of resources destined to handling activities, i.e. labor and equipments, has to the total quantity of materials transported. In the feeding process, a container carries only one kind of item in order to control the system with standard work methods. Thus, when we pack an item i in a container k, a work center will be supplied with a fixed quantity q_{ik} every time the container arrives from the storage area. We may consequently have a holding cost incurred when the amount supplied exceeds the demand. The holding cost is c_i per unity of item i remaining besides the line at the end of a period.

The assembly line feed problem (LFP) consists in how pack the items in the available containers to meet the work centers' demand with a minimum total cost over the planning horizon. We will first consider an unbounded version of the assembly line feed problem (ULFP) where there is no limit over the number of containers of size k available. Because ULFP is then separable over the items, each item is treated as a different instance (hence no need for subscript i). As it will be shown, ULFP is NP-Hard as it has the change-making problem as a special case. Considering a certain item, a frequency variable f_t^k indicates the number of times we will feed the line using container k in period t. A stock variable s_t denotes the amount remaining besides the line at the end of period t. We model ULFP as follows:

$$\min \quad \sum_{t=1}^{T} \left(cs_t + \sum_{k=1}^{K} b_k f_t^k \right) \tag{1}$$

$$\text{s.t.} \quad s_{t-1} - s_t + \sum_{k=1}^{K} q_k f_t^k = d_t, \quad t = 1, \ldots, T, \tag{2}$$

$$s_t, f_t^k \geqslant 0 \text{ and integer.} \tag{3}$$

The change-making problem (CMP) is that of a cashier having to assemble a given change (in our case d) using the least number of coins of specified values (in our case q_k). Lueker [20] has proved that even in its unbounded version CMP is NP-Hard. CMP can also be viewed as an unbounded knapsack problem and has applications in unidimensional cargo-loading and cutting stock problems (see Martello and Toth [21]). ULFP has CMP as a special case by setting $T = 1$ and an arbitrary large holding cost c, which means that the optimal solution of this special case of ULFP will only have strictly positive stock variables if the corresponding CMP is unfeasible.

Additional constraints related to the operational control of the system must be taken into account when developing a model for LFP. A nonquantitative but nevertheless important issue in practice is that the chosen feed process' policy must be easily enough described to be implemented in the shop floor. Operational constraints stand that we can choose only one size of container to feed the line with a given item in the whole planning horizon (since most part of the factory's external suppliers are required to deliver in the right container to feed the line). We also have coupling constraints on the different items because the number of containers of a given size available for materials handling is limited.

We make use of a binary variable x_{ik} that indicates if we pack ($x_{ik} = 1$) or not ($x_{ik} = 0$) item i in container k to feed the line. We have now frequency and stock variables indexed by item, i.e. f_{it}^k and s_{it}, respectively. To write down the bounding constraints on the limit of available containers, we have to know how many containers of size k we must have simultaneously in the process to feed the line with item i. Note that this number is disassociated to its frequency (we can have for example a situation where we feed the line 12 times in a period employing two containers).

The key to determining how many containers k are simultaneously required to feed the line with item i is the average lead time r_{ik} expressed in period time units. This because the number of containers k we must have simultaneously in period t to feed the line with item i is given by $w_{it}^k = \left\lceil \frac{d_{it} r_{ik}}{q_{ik}} \right\rceil$ (see for instance [17, Chapter 17]). The containers spend some time to be filled, some time besides the line, and some time in transit. The first step is to check, in the main storage area, the item to fill the container with. The full container is then moved to the destination work center's inbound stocking, and exchanged by the empty one. The latter container is carried to cleaning, and sent back to the main storage area. A full container spends $\frac{q_{ik}}{d_{it}}$ time units to empty. Let us consider, as an illustrative example, the wheel mounting bolt. The demand is 320 units per hour, and let us assume it is packed in the medium container which filled contains 1000 bolts. The checking and filling operations in the main storage area take two hours in average.

Once the container is filled, it takes 15 minutes to reach the destination work center. Once the container is empty, the cleaning and moving back operations take 45 minutes Thus the average lead time is 6.125 hours, and we must have $\left\lceil \frac{320*6.125}{1000} \right\rceil = 2$ medium containers to feed the line with wheel mounting bolts.

We model LFP as follows:

$$\min \quad \sum_{t=1}^{T} \sum_{i=1}^{I} \left(c_i s_{it} + \sum_{k=1}^{K} b_k f_{it}^k \right) \tag{4}$$

$$\text{s.t.} \quad s_{i(t-1)} - s_{it} + \sum_{k=1}^{K} q_{ik} f_{it}^k = d_{it}, \quad i = 1, \ldots, I, \ t = 1, \ldots, T, \tag{5}$$

$$\sum_{k=1}^{K} x_{ik} = 1, \quad i = 1, \ldots, I, \tag{6}$$

$$M x_{ik} - \sum_{t=1}^{T} f_{it}^k \geqslant 0, \quad i = 1, \ldots, I, \quad k = 1, \ldots, K, \tag{7}$$

$$\sum_{i=1}^{I} w_{it}^k x_{ik} \leqslant l_k, \quad k = 1, \ldots, K, \quad t = 1, \ldots, T, \tag{8}$$

$$x_{ik} \in \{0, 1\}, \ s_{it}, f_{it}^k \geqslant 0 \text{ and integer.} \tag{9}$$

The objective function (4) minimizes the total cost over the planning horizon. In constraint (5) we satisfy the work centers' demand. Constraints (6) and (7) are the operational ones due to shop floor control procedures discussed above. These constraints enforce that for each item only one size of container will be employed to feed – in the whole horizon – the line with it. The parameter M in (7) is an upper bound on the total number of times a container k will be moved from the storage area to the line in the whole horizon. Its value can be given by $M = \max\left\{\sum_{t=1}^{T} \left\lceil \frac{d_{it}}{q_{ik}} \right\rceil : i = 1, \ldots, I\right\}$, where $\bar{k}$ is the smallest size of container. Thus, if we pack item i in container k to feed the line, constraint (7) permits as much handling as needed, otherwise, it ensures that item i will not be transported in container k. Constraint (8) limits to l_k the number of containers of size k simultaneously in use in a given period.

LFP is a variable-sized bin packing problem with an additionally cardinality constraint on each bin's size. Let us consider for that (i) each size k of container as a single bin of capacity l_k; and, (ii) each item i as a one piece item of weight w_{it}^k that depends on the container in which and on the period when it is packed. Thus, constraint (8) ensures that the total weight of the items packed in bin k does not exceed its capacity l_k whatever is the period t of the planning horizon. We have an implicit unitary cardinality constraint since we aggregate all the available containers of size k into only one bin of capacity l_k. In the maximum cardinality bin packing problem, given a set of items and a set of bins with the same capacity, the objective is to maximize the number of items packed without exceeding bin capacities. It is a NP-Hard problem, see for instance Labbé et al. [18]. If we set all holding and handling costs to zero, solve this LFP special instance is equivalent to decide whether the given set of items can or cannot be packed in the given set of capacitated bins. This is the decision problem associated with the maximum cardinality bin packing problem. We have therefore that even finding a feasible solution for LFP is a NP-Complete problem.

Another particular feature of LFP is its special cost structure. We do not have a fixed cost associated to each bin (usually proportional to its size, see for instance [15,16]). But each bin's cost varies according to the items that are packed in it – due to more or less materials holding and handling. Given values of x_{ik}, $i = 1, \ldots, I$, $k = 1, \ldots, K$, feasible to constraints (6), (8) and (9) we obtain in $O(T)$ the values of s_{it} and f_{it}^k, $i = 1, \ldots, I$, $t = 1, \ldots, T$, $k = 1, \ldots, K$ that minimizes (4), since both the holding cost per unit of item i and the handling cost per transportation of container k are considered constant over the planning horizon. For a container $\bar{k}$ and an item $\bar{i}$ such that $x_{\bar{i}\bar{k}} = 1$, constraints (5) and (7) lead to the following expressions for the values of $s_{\bar{i}t}$ and $f_{\bar{i}t}^{\bar{k}}$, $t = 1, \ldots, T$, that minimizes (4) under such a packing policy:

$$s_{\bar{i}t} = s_{\bar{i}(t-1)} - d_{\bar{i}t} + q_{\bar{i}\bar{k}} f_{\bar{i}t}^{\bar{k}} \tag{10}$$

and

$$f_{\bar{i}t}^{\bar{k}} = \left\lceil \frac{d_{\bar{i}t} - s_{\bar{i}(t-1)}}{q_{\bar{i}\bar{k}}} \right\rceil. \tag{11}$$

Constraints (6), (8), and (9) define thus a variable-sized bin packing problem with bin's unitary cardinalities whose objective is a function of the packing variable x_{ik} given by (4), (10), and (11).

3. GRASP for the assembly line feed problem

We developed a GRASP – Greedy Randomized Adaptive Search Procedure – strategy to obtain feasible solutions to LFP. GRASP is a multi-start

metaheuristic proposed by Feo and Resende [11,12] which has been widely used to obtain good quality solutions for many combinatorial problems. Resende and Ribeiro [22] provided an in-depth survey that covers GRASP from basic scheme to recent enhancements, implementation strategies and hybridizations. Festa and Resende [13,14] reported an annotated bibliography containing material such as: tutorials and surveys, enhancements and hybrid methods, parallel implementations, and successful applications in the operational research and computer science domains. In particular, GRASP has been applied to several managerial problems arising in manufacturing and scheduling. The reader is referred to the works of Bard and Feo [3], Feo et al. [10], Laguna and González-Velarde [19], Ríos-Mercado and Bard [23], and Yen et al. [27], to have an overview of how GRASP has been employed to address day-to-day industrial planning problems. Concerning packing, GRASP has been recently applied by Delorme et al. [7] to set packing problems arising in railway planning.

A GRASP iteration consists basically of two phases: (i) construction phase; and, (ii) local search phase. The construction phase builds a feasible solution whose neighborhood is investigated until a local minima in the local search phase. The best solution found after `Max_It` iterations is returned. Fig. 1 illustrates the GRASP framework.

3.1. Construction phase

GRASP uses a greedy randomized heuristic in its construction phase. A feasible solution is iteratively constructed one element at a time. The selection of the next element is guided by an evaluation function. Resende and Ribeiro [22] emphasize the greedy, probabilistic, and adaptive aspects of such a function. The greedy one is that the evaluation function leads to the creation of a restricted candidate list (RCL) formed by the best elements, i.e. those whose incorporation to the current partial solution results in the smallest incremental costs. The probabilistic one is that the element to be incorporated into the partial solution is randomly selected from those in the RCL. Finally, the adaptive aspect is that, once the selected element is incorporated to the partial solution, the candidate list is updated and the incremental costs are reevaluated.

Our greedy randomized heuristic GRH constructs a feasible solution packing one item at a time. For a given container k and item i, we define an evaluation function h as

$$h(i,k) = \sum_{t=1}^{T} (c_i s_{it} + b_k f_{it}^k),$$

where s_{it} and f_{it}^k are given by (10) and (11), respectively. Thus, all the possible values of $h(i,k)$ can be calculated in $O(IKT)$ time. For each item $\hat{i}$, the containers are sorted in non-decreasing order of $h(\hat{i},k)$. This preprocessing operation takes time $O(IK\log K)$.

GRH consists of a main loop executed either until all items have been packed or until it has been checked that the procedure will not be able to find a feasible solution. Initially all containers may be employed to feed the line with each item. Due to the bounding constraints (8), while constructing a feasible solution in a greedy fashion, we may not have a certain container available to feed the line with a given item. We denote by L_i the set of available containers to pack item i in an iteration of the construction procedure. Let J be the set of items that has not yet been packed, initially including all items. For each item $i \in J$, we check whether $L_i = \emptyset$. If this is the case, i.e. no matter the size there are not enough containers to pack item i, the procedure returns that no feasible solution was found. Otherwise, the restricted candidate list RCL is made up with the pairs (i,k) leading to the smallest

```
Procedure GRASP
1 for i = 1,...,Max_It do
2    Obtain a feasible solution s using a greedy randomized heuristic.
3    Obtain a feasible solution s' by applying a local search to s.
4    Let s* be the best solution found so far.
5    if (s' is better than s*) do
6       s* ← s'
7 return s*
```

Fig. 1. GRASP general framework.

incremental costs. To do this, we insert in RCL the pairs (i,k) whose incremental cost $h(i,k)$ is within an interval $[\underline{h},\bar{h}]$, where $\underline{h} = \min\{h(i,k) : (i,k) \in J \times L_i\}$, i.e. the greedy choice. The RCL is then defined by

$$\mathrm{RCL} = \{(i,k) \in J \times L_i : \underline{h} \leqslant h(i,k) \leqslant \underline{h} + \alpha(\bar{h} - \underline{h})\},$$

where α is a parameter such that $0 \leqslant \alpha \leqslant 1$. To restrain the RCL amplitude, we set $\bar{h}$ to the median of $h(i,\hat{k}_i)$, where, for $i = 1,\ldots,I$, $\hat{k}_i$ is the container in L_i with the least value of $h(i,k)$. This allows more flexibility in tuning parameter α, since often in practical situations the feed process deals with very different items having heterogeneous demands. We then select a pair (i^*,k^*) at random from RCL. This means that, in the solution under construction, item i^* is packed in container k^*. The last step is to remove item i^* from set J and update L_i for each $i \in J$. The main loop is executed $O(I)$ times. Build the RCL takes $O(IK)$ time, and update each L_i takes $O(KT)$ time. Thus, the main loop's complexity is $O(I^2KT)$.

3.2. Local search phase

GRASP uses local search in attempt to improve a solution built in the construction phase. Let y denote a feasible solution for LFP, i.e. values of x_{ik}, s_{it}, and f^k_{it}, $i = 1,\ldots,I$, $k = 1,\ldots,K$, $t = 1,\ldots,T$, feasible to (5)–(9). We define an exchange neighborhood $N(y)$ as the set of all feasible solutions that are obtained from y by applying two kinds of moves: (i) substitution move; and, (ii) swap move. Given a certain item i^*, let us denote by k^* the container in which i^* is currently packed. A substitution move changes k^* for a container k' that we still have available in sufficient number, i.e. such that $l^y_{k't} \geqslant w^{k'}_{i^*t}$, $t = 1,\ldots,T$, where l^y_{kt} denotes, for a solution y, the number of containers k that are not used in period t. A swap move exchanges k^* with a container k' that is currently employed to pack another item i'. This kind of move leads to a feasible solution only if the number of containers k^* (resp. k') available plus the number of containers k^* (resp. k') used in feeding the line with item i^* (resp. i') is greater than or equal to the number of containers k^* (resp. k') needed to feed the line with item i' (resp. i^*), i.e. only if $l^y_{k^*t} + w^{k^*}_{i^*t} \geqslant w^{k^*}_{i't}$ and $l^y_{k't} + w^{k'}_{i't} \geqslant w^{k'}_{i^*t}$ for $t = 1,\ldots,T$.

The local search procedure has as input parameter a feasible solution y. We evaluate all the $O(K)$ possible moves associated with each item, i.e. a substitution or a swap move for each container. For each container $k \neq k^*$, we first test if we can perform a substitution move replacing k^* for k. If this is not the case, we identify the set $\overline{J}_{i^*k}$ of those items, if there exists at least one, with which we can perform a swap move. This set is given by $\overline{J}_{i^*k} = \{\bar{i} \in J - \{i^*\} : x_{ik} = 1,$ and for $t = 1,\ldots,T, l^y_{k^*t} + w^{k^*}_{i^*t} \geqslant w^{k^*}_{\bar{i}t}$ and $l^y_{kt} + w^k_{\bar{i}t} \geqslant w^k_{i^*t}\}$. We adopt the best improvement strategy, that is, if the current solution is not a local optima, it is replaced by the best solution in its neighborhood. We proceed the search until a local optima has been found. Each solution has $O(IK)$ neighbors. The most time consuming operations are the identification of set $\overline{J}_{i^*k}$ and, in case $\overline{J}_{i^*k} \neq \emptyset$, the item that leads to best swap move. These two operations take respectively $O(IT)$ and $O(I)$ time. Thus, investigating once the whole neighborhood takes $O(I^2KT)$ time.

4. Computational results

We present numerical results obtained by applying GRASP on real-life LFP instances. The computational experiments were focused on real test instances that cover 10 serial work centers' requirements. Each work center has a demand of about 15–20 different items, resulting in instances containing a total of 191 different items. These items are handled in plastic containers available in three standard sizes – small, medium, and large. The operations planning horizon consists of seven periods since shop floor managerial decisions are taken based on a daily demand. An item's daily demands vary within an interval defined around a target value considered for the planning horizon. Thus, for each item i and period t, we have that d_{it} is an integer from the interval $[(1-\epsilon)\hat{d}_i, (1+\epsilon)\hat{d}_i]$, $\epsilon \in [0,1)$, where $\hat{d}_i$ is a given target value. The feed process must supply the work centers with all kinds of necessary items to complete their operations. This means that the items are heterogeneous, and that the range of its target values is from some units to 10,000 (with no particular distribution). The holding costs per unit also vary in a wide range – from two cents to 20 monetary units. There is no relation between quantities required and holding costs. We have on the one hand some large (resp. small) demand items incurring in high (resp. low) holding costs, and on the other hand some large (resp. small) demand items incurring in low (resp. high) holding costs.

The experiments were run on two groups of test instances distinguished by different scenarios on how the available containers are distributed among the standard sizes. In the first group, denoted by t_group1, given a container k, l_k is a percentage of the total number a_k of containers needed to feed the line employing only containers of size k. In the second group, denoted by t_group2, there is a predominance of a certain size in the factory's containers park. Therefore, there exits, for the t_group1 instances, the relation $\frac{l_k}{a_k} = u$ for $k = 1, \ldots, K$, where u is a constant value, that is not observed in t_group2.

For each group, we vary the following parameters resulting in twelve instances per group: (i) two values for ϵ; (ii) two handling costs policies; and, (iii) three values for l_k. Parameters (i) and (ii) are common to both groups t_group1 and t_group2 of test instances. We set $\epsilon = 5\%$ and $\epsilon = 20\%$ obtaining respectively a more and a less uniform daily demand. And we consider the following two handling costs policies: a fixed handling cost for all size of containers and a proportional handling cost. In the fixed cost policy, a handling cost of 10 monetary units is incurred every time a container is transported no matter its size. Let $\bar{q}_k$ be the average number of items carried when container k is full, i.e. $\bar{q}_k = \frac{\sum_{i=1}^{I} q_{ik}}{I}$. In the proportional cost policy, we have that $\frac{b_k}{\bar{q}_k}$ is equal to a constant for $k = 1, \ldots, K$. In the present experiments we fixed this constant by associating a handling cost of 10 monetary units to the medium size container.

Defining values for parameter l_k depends on the characteristics of each group of test instance. For the t_group1 instances, we first calculate the number a_k needed to feed the line using exclusively containers of size k, and then we set $l_k = \gamma a_k$ where $\gamma \in [0, 1]$ is a fixed value for $k = 1, \ldots, K$. Let $k = 1$, $k = 2$, and $k = 3$ denote respectively the small, the medium, and the large size of container. For the t_group2 instances, we set $l_k = \gamma_k a_1$, $\gamma_k \in [0, 1]$ for $k = 1, \ldots, K$, such that we have a predominance of a certain container size. Note that since $k = 1$ is the smallest container, we can have feasible instances when $\sum_{k=1}^{K} \gamma_k < 1$. We consider three different configurations denoted by sm, md, and lg. In the sm configuration we have $l_1 = 50\%a_1$, $l_2 = 5\%a_1$, $l_3 = 15\%a_1$. For the md configuration these values are $l_1 = 5\%a_1$, $l_2 = 60\%a_1$, $l_3 = 20\%a_1$. Finally, for the lg configuration $l_1 = 5\%a_1$, $l_2 = 5\%a_1$, $l_3 = 60\%a_1$.

We report numerical results for t_group1 and t_group2 respectively in Tables 1 and 2. The GRASP heuristic was ran on a SUN BLADE 100 Workstation ULTRASPARC 500 MHz with 1 GB of RAM memory. It was coded in C and was compiled with the gcc compiler version 3.3.2 with no optimization flags. Parameter Max_It was set to 500 iterations, and, at each GRASP iteration, α was randomly chosen in the interval $[0.1, 0.4]$. To analyze whether or not it is easy to find optimal solutions for practical LFP instances, we ran CPLEX version 9.0 with no time limit on a Pentium IV 3 GHz with Hyperthreading, 1 GB of RAM memory and 1 GB of swap memory.

In order to evaluate the proposed heuristic performance, we compare the GRASP results with the ones given by the firm's packing strategy.

Table 1
Computational results for the test instances of t_group1

ϵ	b_k	γ	PGP	GRASP			CPLEX			
			UB_p	UB_g	r (%)	s	UB_c	LB	o (%)	s
5%	f	0.8	397,288	301,239	24.18	26	424,144	106,364	74.92	1277
		0.6	386,896	307,551	20.51	26	366,836	106,238	71.04	1071
		0.4	410,918	324,428	21.05	27	447,480	106,952	76.10	1081
	p	0.8	306,207	293,342	4.20	27	323,195	135,912	57.95	1277
		0.6	311,616	297,568	4.51	34	365,207	136,080	62.74	2933
		0.4	367,176	300,127	18.26	60	341,868	136,105	60.19	1502
20%	f	0.8	426,998	310,867	27.20	26	499,606	111,649	77.65	1436
		0.6	423,298	317,445	25.01	26	543,414	110,359	79.69	1398
		0.4	435,326	336,429	22.72	28	433,467	110,910	74.41	1555
	p	0.8	327,817	308,444	5.91	26	358,125	145,963	59.24	1615
		0.6	332,026	308,554	7.07	28	404,610	146,108	63.89	1399
		0.4	383,456	311,041	18.88	90	414,896	146,168	64.77	1312

Table 2
Computational results for the test instances of t_group2

ϵ	b_k	γ	PGP	GRASP			CPLEX			
			UB_p	UB_g	r (%)	s	UB_c	LB	o (%)	s
5%	f	sm	408,315	303,443	25.68	47	351,923	106,604	69.71	1266
		md	586,784	307,361	47.62	146	–	106,431	–	4579
		lg	704,173	309,741	56.01	432	–	108,404	–	5444
	p	sm	350,126	298,467	14.75	50	353,971	135,982	61.58	1464
		md	597,391	326,143	45.41	227	443,317	141,919	67.99	1096
		lg	730,525	337,711	53.77	639	–	143,001	–	5094
20%	f	sm	409,598	331,144	19.15	34	484,482	120,158	75.20	1256
		md	586,189	316,910	45.94	195	493,048	111,600	77.37	982
		lg	701,307	319,356	54.46	640	414,814	112,295	72.93	1278
	p	sm	349,543	310,318	11.22	111	352,109	146,360	58.43	1078
		md	596,331	338,280	43.27	286	388,036	153,015	60.57	1086
		lg	728,548	349,552	52.02	774	–	154,467	–	5130

We aim, with such a comparison, to show GRASP potential in reducing operational costs in assembly line systems. The actual firm's packing strategy is a pure greedy procedure. It consists of the following steps to each item i sorted in non-decreasing order of unitary holding cost c_i. First, the set L_i of available containers to pack item i is built. If L_i is an empty set, the procedure returns that no feasible solution was found. Otherwise, the procedure continues by calculating the average demand $\bar{d}_i$ over the planning horizon. Let us denote by u_{ik} the number of items i that exceeds the average demand in a single period supply employing container k, i.e. $u_{ik} = \left\lceil \frac{\bar{d}_i}{q_{ik}} \right\rceil * q_{ik} - \bar{d}_i$. Item i is then packed in the largest container $k^* \in L_i$ such that $u_{ik^*} \leqslant u_{ik}$ for all $k \in L_i - \{k^*\}$. (That is, ties are broken by lesser handling.)

Tables 1 and 2 show in the first two columns the values of ϵ and if the handling cost policy is the fixed one (f) or the proportional one (p). The third column differs accordingly to the test instance group: for each t_groupl instance (in Table 1), it shows the value of γ constant for $k = 1, \ldots, K$; and, for each t_group2 instance (in Table 2), it shows the configuration sm, md, or lg of the factory's containers park. The next column gives results obtained by the firm's packing strategy, denoted by PGP. The last columns present the numerical results for GRASP and CPLEX. The upper bounds obtained with PGP, GRASP and CPLEX are denoted respectively by UB_p, UB_g and UB_c. We first give, for GRASP, the cost of the best solution found, the percentage reduction gap, and the CPU time in seconds to perform Max_It iterations. The reduction gap reflects GRASP contribution in reducing actual managerial costs, and it is given by $r = \frac{UB_p - UB_g}{UB_p}\%$. We then give, in the next three columns, upper and lower bounds and the percentage optimality gap obtained by CPLEX, i.e. $o = \frac{UB_c - LB}{UB_c}\%$. The last column gives CPU time in seconds until CPLEX ran out of memory.

GRASP was able to significantly reduce actual operational costs in feeding the studied assembly line. Over the 24 instances, reduction gaps are in average 27.87%, going up to 56.01%. Moreover, upper bounds provided by GRASP, in moderate computational times, are much better than the ones provided by CPLEX. It seems that these practical instances are not easy to solve, since optimality gaps provided by CPLEX 9.0 running until get out of memory are at least 57.95%. And indeed CPLEX was not able to find, after more than 4500 seconds of computation, feasible solutions for 4 out of 24 instances.

By analyzing Tables 1 and 2, it appears that the harder instances are the ones where the constant relation $\frac{l_k}{b_k}$ for $k = 1, \ldots, K$ does not hold. We remark first that it was precisely for 4 out 12 instances of t_group2 that CPLEX ran out of memory without find feasible solutions. Second, GRASP computational times increase when varying from the sm to the lg configuration while it remains almost constant in the t_groupl instances. The difference in the GRASP times increasing behaviour from t_groupl to t_group2 instances is explained by the number of iterations that the local search is actually activated. In 6 out of the 12 t_groupl instances the solutions constructed by the GRH procedure were already a local optima in

all the 500 GRASP iterations executed. This never occurred for t_group2 instances. On the contrary, in 8 out of the 12 t_group2 instances local search improved the solutions constructed by the GRH procedure in all the 500 GRASP iterations. This fact – more room for local search improvements – also explains why reductions in actual operational costs are more significant for t_group2 instances. In these cases, reduction gaps are in average 39.11%. And in particular, for instances of the lg configuration, the ones in which it is observed a predominance of the largest container in the factory's park, GRASP obtains sound reductions in actual operational costs: always more than 50%. We explain this performance by a more careful choice of which items to be packed in the scarce small and medium containers carried out by the local search.

5. Concluding remarks

We described an important application of packing problems arising in assembly line feed process. Particular features of this application such as variable sized containers, cardinality constraints, and special cost structure were identified. We proposed two approaches to deal with such an industrial problem: an integer programming model and a GRASP heuristic. It seems that these are difficult practical problems, since realistic instances could not be solved by a standard optimization package. Numerical results reported show however that metaheuristics may be used as valuable tools to reduce actual managerial costs. In particular, the proposed GRASP was able to find in moderate computational times solutions whose reduction gaps with respect to the firm's packing policy are in average 27.87%, going up to 56.01%. Further research directions are, in our point of view, related to the weakness of the lower bounds given by standard packages and to the possibilities in finding better feasible solutions. One way could be the development of hybrid metaheuristics combining lower bounding strategies with primal heuristics, as it has been successfully done by Alvim et al. [2] for the bin packing problem.

Acknowledgements

The authors wish to thank the two anonymous referees for helpful suggestions in improving this paper. Thanks are also due to LaPO – Operational Research Laboratory, Computer Science Department, UFMG – colleagues for the use of CPLEX version 9.0. The first author was supported by CNPq grant 306884/2003-8, Brazil.

References

[1] C. Alves, J.M. Valério de Carvalho, Accelerating column generation for variable sized bin-packing problems, European Journal of Operational Research to appear (2005).
[2] A.C.F. Alvim, F. Glover, C.C. Ribeiro, D.J. Aloise, A hybrid improvement heuristic for the one-dimensional bin packing problem, Journal of Heuristics 10 (2004) 205–229.
[3] J.F. Bard, T.A. Feo, Operations sequencing in discrete parts manufacturing, Management Science 35 (1989) 249–255.
[4] G.R. Bitran, L. Chang, A mathematical programming approach to a deterministic Kanban system, Management Science 33 (1987) 427–441.
[5] J.A. Buzacott, J.G. Shanthikumar, Stochastic Models of Manufacturing Systems, Prentice Hall, 1993.
[6] C. Chu, R. La, Variable-sized bin packing: tight absolute worst-case performance ratios for four approximation algorithms, SIAM Journal on Computing 30 (2001) 2069–2083.
[7] X. Delorme, X. Gandibleux, J. Rodriguez, GRASP for set packing problems, European Journal of Operational Research 153 (2004) 564–580.
[8] L. Epstein, R. van Stee, On variable-sized multidimensional packing, Lecture Notes in Computer Science 3221 (2004) 287–298.
[9] L. Epstein, R. van Stee, Optimal online algorithms for multidimensional packing problems, SIAM Journal on Computing 35 (2005) 431–448.
[10] T.A. Feo, J.F. Bard, S. Holland, Facility-wide planning and scheduling of printed wiring board assembly, Operations Research 43 (1995) 219–230.
[11] T.A. Feo, M.G.C. Resende, A probabilistic heuristic for a computationally difficult set covering problem, Operations Research Letters 8 (1989) 67–71.
[12] T.A. Feo, M.G.C. Resende, Greedy randomized adaptive search procedures, Journal of Global Optimization 6 (1995) 109–133.
[13] P. Festa, M.G.C. Resende, GRASP: An annotated bibliography, in: C.C. Ribeiro, P. Hansen (Eds.), Essays and Surveys in Metaheuristics, Kluwer Academic Publishers, Norwell, 2001, pp. 325–367.
[14] P. Festa, M.G.C. Resende, An annotated bibliography of GRASP, AT&T Labs Research Technical Report, TD-5WYSEW, 2004.
[15] D.K. Friesen, M.A. Langston, Variable sized bin packing, SIAM Journal on Computing 15 (1986) 222–230.
[16] J. Kang, S. Park, Algorithms for the variable sized bin packing problem, European Journal of Operational Research 147 (2003) 365–372.
[17] L.J. Krajewski, L.P. Ritzman, Operations Management: Strategy and Analysis, third ed., Addison-Wesley, 1993.
[18] M. Labbé, G. Laporte, S. Martello, Upper bounds and algorithms for the maximum cardinality bin packing problem, European Journal of Operational Research 149 (2003) 490–498.
[19] M. Laguna, J.L. González-Velarde, A search heuristic for just-in-time scheduling in parallel machines, Journal of Intelligent Manufacturing 2 (1991) 253–260.

[20] G.S. Lueker, Two NP-Complete problems in nonnegative integer programming, Report No. 178, Computer Science Laboratory, Princeton University, Princeton, USA, 1975.
[21] S. Martello, P. Toth, Knapsack Problems: Algorithms and Computer Implementations, John Wiley & Sons, 1990.
[22] M.G.C. Resende, C.C. Ribeiro, Greedy randomized adaptive search procedures, in: F. Glover, G. Kochenberger (Eds.), Handbooks of Metaheuristics, Kluwer Academic Publishers, 2003, pp. 219–249.
[23] R.Z. Ríos-Mercado, J.F. Bard, Heuristics for the flow line problem with setup costs, European Journal of Operational Research 110 (1998) 76–98.
[24] S.S. Seiden, R. van Stee, New bounds for multidimensional packing, Algorithmica 36 (2003) 261–293.
[25] N. Watanabe, S. Hiraki, A modeling approach to a JIT-based ordering system, Annals of Operations Research 69 (1997) 379–403.
[26] G.J. Woeginger, G. Zhang, Optimal online algorithm for variable-sized bin covering, Operations Research Letters 25 (1999) 47–50.
[27] J. Yen, M. Carlsson, J.M. Garcia, H. Nguyen, Constraint solving for inkjet print mask design, Journal of Imaging Science and Technology 44 (2000) 391–397.