Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in DISCRETE MATHEMATICS

Electronic Notes in Discrete Mathematics 47 (2015) 77–84

www.elsevier.com/locate/endm

General Variable Neighborhood Search applied to the picking process in a warehouse

Borja Menéndez [a,1,2] Eduardo G. Pardo [a,1,3]
Abraham Duarte [a,1,4] Antonio Alonso-Ayuso [b,5]
Elisenda Molina [c,6]

[a] *Dept. Ciencias de la Computación, Universidad Rey Juan Carlos, Móstoles, Madrid, Spain*

[b] *Dept. Estadística e Investigación Operativa, Universidad Rey Juan Carlos, Móstoles, Madrid, Spain*

[c] *Dept. Estadística, Universidad Carlos III, Getafe, Madrid, Spain*

Abstract

The order batching problem is a part of the picking process of items in a warehouse. A set of items conform an order and a set of orders conform a batch. The problem consist of grouping the orders received in the warehouse in different batches. Each batch have to be collected by a single picker without exceed a capacity limit. The objective is to minimize the total time needed to collect all the items. In this paper we propose a General Variable Neighborhood Search algorithm to tackle the problem. Our approach outperforms other previous methods in the state of the art.

Keywords: Order Batching Problem, Heuristics, VNS, GVNS.

http://dx.doi.org/10.1016/j.endm.2014.11.011

1 Introduction

In the last few years, warehousing has become a very important issue for the industry, specially as a part of the supply chain. Among others, the success of warehousing depends on the picking process (i.e., the way that items are retrieved from the warehouse). This process may consume up to 60% of the total time of all labor activities in the warehouse [5], which can suppose more than half of the total operating cost. When dealing with the picking process of items in a warehouse it is possible to identify two different optimization problems: the batching of a set of orders and the routing to collect them. The objective of both problems is to minimize the total time needed by one operator to collect all the items in a set of orders. In this paper we focus our attention in the Order Batching Problem (OBP) and we propose an algorithm based on the Variable Neighborhood Search (VNS) methodology to address it.

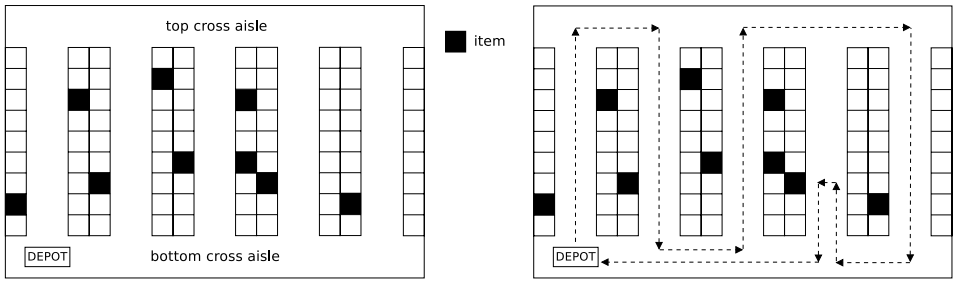
The rest of the paper is organized as follows. In Section 2 we describe the problem and the associated literature. In Section 3, we propose a General VNS (GVNS) algorithm to tackle the problem. Computational experiments and the associated conclusions are presented in sections 4 and 5 respectively.

2 Order Batching Problem

An item represents an object that must be collected from a warehouse. An order is a list of items packed together that must be retrieved by the same picker. Given a set of orders received in a warehouse at the same time, there are two basic order-picking strategies: *strict-order picking* and *order batching*. In the first one, each picker collects all the items included in one order, then another order is assigned to the picker and so on. In the second one, several orders are put together into batches. Each batch is assigned to a picker, who can retrieve the items of any order grouped into the assigned batch. Since several orders are picked simultaneously, efficient routes can be conceived and travel times can be reduced.

[1] This research has been partially supported by the Ministerio de Economía y Competitividad of Spain (Grant No. TIN2012-35632-C02-02) and Comunidad de Madrid (Grant No. S2013/ICE-2894).

[2] Email:borja.menendez@urjc.es
[3] Email:eduardo.pardo@urjc.es
[4] Email:abraham.duarte@urjc.es
[5] Email:antonio.alonso@urjc.es
[6] Email:emolina@est-econ.uc3m.es



(a) Warehouse layout. (b) Combined.

Fig. 1. Warehouse layout and routing strategy.

In this paper, we will center our attention in the order batching strategy, where it is necessary to group all the orders into batches considering the weight constraint (i.e., the picker has a maximum load capacity and the total load of each batch can not exceed it). The objective of the paper will be to conform the batches with the aim of minimizing the total time needed to collect them. It is important to notice that the routing problem associated to the picking process will not be tackled in this paper since it has been widely studied in the literature (see [4,11] for a review). Therefore we will make use of an efficient routing algorithm previously available.

There exist different variants of the OBP. In particular, we consider the variant with only one operator working at the same time and where the warehouse has parallel aisles of equal length. The aisles are connected by crossing aisles at the front and the back of each aisle. With this layout, it is possible to perform routes from both ends of the aisle. Additionally, the depot where the warehouse operator must drop the collected items is the starting and ending point for all routes. This depot can be located either in one of the corners of the warehouse or in the center of one of the cross aisles. Figure 1a illustrates a warehouse layout with 5 parallel aisles, 2 cross aisles and a depot at the left bottom corner.

The routes followed by the picker are usually determined by routing strategies. Although some routing schemes have been proved to be optimal, there are some tours that seem to reduce the number of in-aisle congestion like *traversal* or *largest gap* ([3]). These tours are prevalent in practice, since pickers seem to accept only straightforward and non-confusing routing schemes.

In particular, in this paper we will use a combination of the two previously mentioned strategies (*traversal* and *largest gap*). This strategy is named *combined* (see Figure 1b) and it was proposed by Roodbergen and De Koster [11]. The *combined* strategy make use of ideas from both: an aisle is either

entered and crossed (*traversal*) or entered and left from the same side of the aisle (*largest gap*). The criteria used to know whether an aisle is fully traversed or not is to check pairs of adjacent aisles (where the term "adjacent" is only referred to aisles which have at least one item to collect). The option to choose is selected taking into account which one gives the shortest route when combined with the best option for the next aisle.

The OBP has been proved to be $\mathcal{NP}$-hard if the number of orders per batch is greater than two [7]. In this sense, several heuristic methods have been developed to tackle the problem. According to De Koster *et al.* [3], most order batching heuristics can be classified into one of these three categories:

- **Basic methods** are simple constructive methods, as First-Come-First-Served. This strategy adds orders to batches in the sequence they arrive. New batches are added to the list when an order can not be added to the current batch. These methods are not very effective.
- **Seed methods**, introduced by Elsayed [6], generate batches sequentially. These methods produce solutions in two different steps: (i) an initial order (the so-called "seed") is chosen and selected for the new batch. (ii) Orders are sequentially added to the batch following a *proximity* to the seed criterion, as long as the picker capacity is not exceeded [3].
- **Saving methods** are inspired in the Clark and Wright savings algorithm [2]. These methods start with one batch per order and then they combine pairs of batches. There are two different variants: Clark and Wright I, make use of the original savings matrix during the whole process; Clark and Wright II updates the savings matrix each time two batches are combined.

However, there are other heuristic approaches that do not fit into the previous categories: Hsu *et al.* [9] and Tsai *et al.* [12], respectively, presented a Genetic Algorithm each one for this problem. Later, Albareda-Sambola *et al.* [1] proposed a Variable Neighborhood Descent (VND) method which is, as far as we know, the best method in the literature for the OBP.

3 General Variable Neighborhood Search

Variable Neighborhood Search (VNS) is a metaheuristic based on systematic changes of neighborhood (Mladenovic and Hansen [10]). This idea helps to escape from local optima. There are different schemes based on this methodology. Specifically we have selected the General VNS (GVNS) one to address the OBP.

The proposed algorithm is presented in Algorithm 1. It starts with a feasi-

ble solution, f, provided by a constructive algorithm (in this case it is a random solution). There are two more parameters that configures the algorithm: *maxiter* is the maximum number of iterations and k_{max} is the maximum number of neighborhoods to be explored. In each iteration, the solution is randomly perturbed by the *Shake* procedure with size k (step 5) obtaining a new solution, f', and then a local search procedure based on a Variable Neighborhood Descent (VND) algorithm is applied to f', obtaining an improved solution f'' (step 6). If f'' is better than f (step 7), f is updated with f'' as the best solution found until the moment and k is set to 1 (step 11), otherwise, k is incremented. This process is repeated until k_{max} is reached (step 13).

Notice that we do not report the pseudo-codes of the *Shake* nor the *VND* procedures, since they follow their classical implementation (see Hansen and Mladenovic [8]). In particular, the *Shake* procedure applies a random perturbation to the solution within the k-th neighborhood. This perturbation consists of swapping k times, two orders from different batches, at random, in such a way that each swap results in a feasible solution. *VND* is based on changes of neighborhood structures in a deterministic way. Specifically, the procedure included in step 6 of Algorithm 1 uses two neighborhood structures: (i) all the feasible solutions reachable by the *insertion* of any order in a different batch and (ii) all the feasible solutions reachable by the *swap* of two orders from different batches.

Algorithm 1 GVNS algorithm

```
1: procedure GVNS(f, maxiter, k_max)
2:     for i ← 0; i < maxiter; i ← i + 1 do
3:         k ← 1
4:         repeat
5:             f' ← Shake(f, k)
6:             f'' ← VND(f')
7:             if f'' < f then
8:                 f ← f''
9:                 k ← 1
10:            else
11:                k ← k + 1
12:            end if
13:        until k ≥ k_max
14:    end for
15:    return f
16: end procedure
```

4 Computational experiments

In this section we present the experiments performed to empirically compare our proposal with previous algorithms in the state of the art. Particularly, we have selected the best previous algorithm for the OBP (see Albareda-Sambola *et al.* [1]). In order to have a fair comparison between the algorithms we have used a subset of 600 test cases proposed in their paper. For further information we refer the reader to the description of Warehouse 1 in Section 5.1 of that paper.

From the whole data set we selected 20 representative instances in order to adjust the parameters of our GVNS. According with the obtained results, variables k_{max} and *maxiter* were set to 5, since larger values would increase the CPU time and the improvement obtained in the results is negligible.

In Table 1 we report the results obtained by our GVNS and by the results obtained by the best method proposed in Albareda-Sambola *et al.* [1] which is named "Exchange". We present the average of the value of the objective function (Avg.), the average of the deviation to the best known value (Dev.(%)), the average of the CPU execution time (CPUt) in milliseconds and the number of best solutions found (#Best). Both methods were implemented in Java 6 and were run in the same computer (Intel Core i7 with 3.4 GHz and 4 GB of RAM with Linux Mint Debian Edition 64 bit OS). Additionally, both algorithms used the same routing strategy previously mentioned (*Combined*).

Results in Table 1 are divided in groups taking the number of orders of each instance into account. In particular we divided our experiment in 5 groups of 50, 100, 150, 200 and 250 orders with 120 instances per group. Finally, we also show the results over the whole data set (see row Global) considering all the instances together. According to the results in Table 1, GVNS outperforms the state of the art in almost all the subsets in terms of Dev.(%) and #Best in shorter time. The only exception is the set formed by instances of 50 orders, where the Exchange method performs slightly better. If we consider the whole data set, the GVNS obtains a deviation of 0.55% to the best known values, meanwhile Exchange obtains a deviation of 1.12% in almost double of the CPU time than GVNS. The number of best values obtained by each algorithm are also remarkable, since GVNS is able to match 335 best know values, meanwhile Exchange matches 134.

To conclude the experiments we have carried out statistical tests to confirm that the differences obtained by our algorithm are statistically significant. In particular, we performed the Wilcoxon test for each group of instances separately and also for all the instances together. The *p-value* obtained by all the

Table 1
Comparison of GVNS and Exchange algorithms.

		Avg.	**Dev. (%)**	**CPUt (ms)**	**#Best**
50 orders	GVNS	3510.21	1.50	33.90	43
	Exchange	3501.24	1.42	11.42	45
100 orders	GVNS	6625.09	0.64	96.83	52
	Exchange	6656.01	1.11	79.92	31
150 orders	GVNS	9849.93	0.33	205.98	68
	Exchange	9918.03	1.06	250.17	22
200 orders	GVNS	12934.16	0.18	358.82	84
	Exchange	13041.03	1.02	584.25	18
250 orders	GVNS	16140.21	0.08	547.21	88
	Exchange	16268.48	0.95	1179.83	18
Global	GVNS	9811.92	0.55	248.55	335
	Exchange	9876.96	1.12	421.12	134

subsets separately was 0.000 indicating that there are significant differences between the evaluated algorithms except for the 50 orders per instance subset, where the obtained *p-value* was 0.743 (curiously, it is the only subset where Exchange slightly outperforms GVNS). Finally, if we consider all the instances together, the *p-value* was again 0.000, indicating that the differences between the algorithms are statistically significant.

5 Conclusions

In this paper, we dealt with the Order Batching Problem in a parallel aisle warehouse with two cross-aisles at the front and at back of the warehouse. In particular we proposed a General VNS algorithm to deal with the problem. We experimentally compared our proposal with the best previous algorithm in the state of the art over a referred set of instances. The results indicated that our algorithm outperforms previous methods in the state of the art. Additionally,

we confirmed that the differences were statistically significant.

References

[1] Albareda-Sambola, M., A. Alonso-Ayuso, E. Molina and C. Simón de Blas, *Variable neighborhood search for order batching in a warehouse*, Asia-Pacific Journal of Operational Research **1** (2009), pp. 655–683.

[2] Clark, G. and W. Wright, *Scheduling of vehicles from a central depot to a number of delivery points*, Operations Research **12** (1964), pp. 568–581.

[3] De Koster, R., E. van der Poort and M. Wolters, *Efficient orderbatching methods in warehouses*, International Journal of Production Research **37** (1999), pp. 1479–1504.

[4] De Koster, R., K. J. Roodbergen and R. Van Voorden, *Reduction of walking time in the distribution center of De Bijenkorf*, Technical report, Rotterdam School of Management, Erasmus University of Rotterdam (1998).

[5] Drury, J., *Towards More Efficient Order Picking*, IMM monograph No. 1, The Institute of Materials Management, Cranfield, UK (1988).

[6] Elsayed, E., *Algorithms for optimal material handling in automatic warehousing systems*, International Journal of Production Research **19** (1981), pp. 525–535.

[7] Gademann, N. and S. van de Velde, *Order Batching to Minimize total Travel Time in a Parallel-Aisle Warehouse*, *IIE Transactions* **37** (2005), pp. 63–75.

[8] Hansen, P. and N. Mladenovic, *Variable neighborhood search: Principles and applications*, European Journal of Operational Research **130** (2001), pp. 449–467.

[9] Hsu, C.-M., K.-Y. Chen and M.-C. Chen, *Batching orders in warehouses by minimizing travel distance with genetic algorithms*, *Computers in Industry* **56** (2005), pp. 169–178.

[10] Mladenovic, N. and P. Hansen, *Variable neighborhood search*, *Computers and Operations Research* **24** (1997), pp. 1097–1100.

[11] Roodbergen, K. J. and R. de Koster, R*outing methods for warehouses with multiple cross aisles*, International Journal of Production Research **39** (2001), pp. 1865–1883.

[12] Tsai, C.-Y., J. J. H. Liou and T.-M. Huang, *Using a Multiple-GA Method to Solve The Batch Picking Problem: Considering Travel Distance and Order Due Time*, *International Journal of Production Research* **46** (2008), pp. 6533–6555.