An Efficient Evolutionary Solution to the joint Order Batching - Order Picking Planning Problem

Riccardo Lucato
Universität Hildesheim
Hildesheim, Germany
lucato@uni-hildesheim.de

Jonas Falkner
Universität Hildesheim
Hildesheim, Germany
falkner@ismll.uni-hildesheim.de

Lars Schmidt-Thieme
Universität Hildesheim
Hildesheim, Germany
schmidt-thieme@ismll.uni-hildesheim.de

ABSTRACT

This paper introduces an innovative solution to the joint Order Batching - Order Picking Planning Problem. Previous research on this field has mainly focused on solving the two problems separately. Failing to consolidate the two components, however, this approach leads to sub-optimal solutions. In this work we propose a Genetic Algorithm for Joint Optimization (GAJO), which optimizes the integrated problem. Thorough experimentation shows GAJO to be both more effective and faster than the baseline models.

CCS CONCEPTS

• **Theory of computation → Evolutionary algorithms**; • **Computing methodologies → Genetic algorithms**; *Planning and scheduling*; • **Applied computing →** *Operations research*;

KEYWORDS

Order Batching Problem, Order Picking Planning Problem

ACM Reference Format:
Riccardo Lucato, Jonas Falkner, and Lars Schmidt-Thieme. 2020. An Efficient Evolutionary Solution to the joint Order Batching - Order Picking Planning Problem. In *Genetic and Evolutionary Computation Conference Companion (GECCO '20 Companion), July 8–12, 2020, Cancún, Mexico.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3377929.3389935

1 INTRODUCTION

In a typical warehouse, the order picking process consists of two major components: the task of travelling through the warehouse to retrieve the merchandise and that of finding efficient groupings of orders to minimize the incidence of redundant picking sub-tours. The global objective, which is the minimization of the total walking distance across all orders, therefore consists of two sub-targets: first efficiently batching the orders together, combinatorial problem defined in the literature as the Order Batching Problem (OBP), second finding the shortest possible picking path for the items of every batch of orders, known in the literature as the Order Picking Planning Problem (OPP). While the separate optimization of the two problems through two-stage approaches finds a rather extensive coverage in the literature, their joint optimization through a single algorithm is still an open challenge. This represents a noteworthy research gap, which the present paper intends to fill.

GECCO '20 Companion, July 8–12, 2020, Cancún, Mexico

ACM ISBN 978-1-4503-7127-8/20/07…$15.00
https://doi.org/10.1145/3377929.3389935

2 PROBLEM FORMULATION

The set of constraints K to which the OBP is subject is [4], [3]:

$$K : \begin{cases} \sum_{j \in J} c_j \cdot a_{ij} \leq C, \ \forall i \in I & (1) \\ \sum_{i \in I} a_{ij} \cdot x_i = 1, \ \forall j \in J & (2) \end{cases}$$

where J is the set of customer orders, C is the carrying capacity of the picking device, c_j the capacity required for order j, a_i a vector of binary entries a_{ij} describing whether an order j is included in a batch i ($a_{ij} = 1$) or not ($a_{ij} = 0$), I is the set of all feasible batches and x_i is a binary decision variable stating whether batch i is actually constructed ($x_i = 1$) or not ($x_i = 0$). The set of constraints H of the OPP is [5]:

$$H : \begin{cases} \sum_{p \in V} y_{pq} = 1, \ \forall q \in V & (3) \\ \sum_{q \in V} y_{pq} = 1, \ \forall p \in V & (4) \\ h_p - h_q = (n+1) y_{pq} \leq n, \ \forall (p,q) \in E : p, q \neq 0 & (5) \end{cases}$$

in which V is the set of vertices $V = \{0, ..., n\}$ of the complete graph representing the problem, E is the set of edges $E = \{(p,q) : p, q \in V, p \neq q\}$, y_{pq} is a binary variable describing whether edge (p,q) is contained in the tour ($y_{pq} = 1$) or not ($y_{pq} = 0$) and h_p is the position of vertex p in the tour. Since the set of constraints M of the joint OBP-OPP can be written as

$$M = K \cup H \cup a \tag{6}$$

with a ensuring compatibility between solution of OBP and OPP

$$a : \left\{ q \in \{ j : j \in i \}, \forall q \in V_i, \forall i \in I \right\}, \tag{7}$$

then, letting d_{pq} be the distance between edges p and q, the mathematical formulation of the joint OBP-OPP can be expressed as:

$$\begin{aligned} \min \quad & \sum_i \sum_{(p_i, q_i) \in E_i} d_{p_i q_i} \cdot y_{p_i q_i} \\ \text{sub} \quad & M \end{aligned} \tag{8}$$

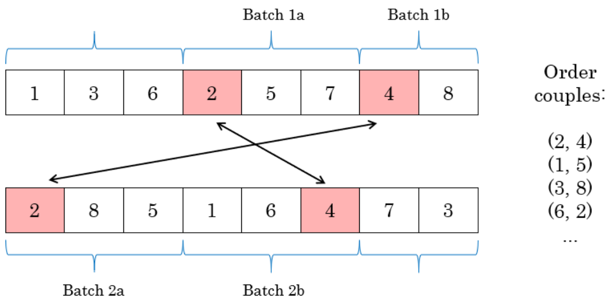

Figure 1: Working scheme of the proposed S-C Crossover Operator, which progressively homogenizes the batches.

3 METHODOLOGY

We propose a novel approach involving a plain Genetic Algorithm. It is named Genetic Algorithm for Joint Optimization, or GAJO. This is a 11-step procedure, summarized below:

Step 1 - Encode initial solutions. Each chromosome consists of three portions, respectively describing batch sizes, batch contents in terms of orders and batch-wise picking sequence.
Step 2 - Compute fitness of each solution.
Step 3 - Create mating pool of fittest individuals.
Step 4 - Perform crossover and mutation on the second portions of random individuals from the mating pool, while leaving the remaining portions untouched. We introduce a new crossover operator called S-C Crossover: given two parent chromosomes, a similarity score is computed for each of their batches based on the average walking distance separating the items they contain. A list is then created of all unique pairs of orders to be picked. For each pair of orders on the list, one at a time, it is checked if swapping the position of the two orders leads to an increase in average similarity in all four batches. If this is the case, the two swaps are actually carried out. The procedure is repeated until the order pair list has been exhausted. Since this operation typically renders the chromosomes invalid due to incompatibility between second and third portions, the third portions of the resulting chromosomes are dropped. After performing mutation, the two incomplete chromosomes are added to a new population. The process is repeated, two insertions at a time, until the new population has the same size as the original one. Thus the former replaces the latter.
Step 5 - Select one incomplete chromosome from the new population, then create a number of feasible picking tours for its first batch and add them to a new, inner population.
Step 6 - Create mating pool from the inner population.
Step 7 - Perform crossover and mutation on random individuals in the inner population and add the offspring to a new inner population. Repeat the process until the new inner population has the same size as the original one, then replace the latter with the former.
Step 8 - Repeat steps 6 and 7 for a given number of iterations.
Step 9 - Repeat steps 5 through 8 for all batches of the current incomplete chromosome selected in Step 5.
Step 10 - Repeat steps 5 through 9 for all individuals in the population of incomplete chromosomes created in Step 4.
Step 11 - Repeat steps 2 through 10 for a given number of iterations.

Table 1: Assessment of the mean distance, standard deviation and runtime of the three algorithms over ten independent sets of experiments.

Algorithm	Distance Found	Standard Deviation	Runtime
Hybrid	20633.92 m	104.52	21'34"
Savings	5828.41 m	80.69	/
GAJO	5133.15 m	61.12	3'47"

4 EXPERIMENTS

Objective of the experiment is to find the shortest possible picking paths for several groups of orders received by the warehouse of a large German retail company at various points in time, by using three different algorithms: the PSO-ACO Hybrid Algorithm [1], the Savings Algorithm [2] and GAJO. A total of ten groups of orders of disparate sizes were considered, ranging from 200 to 3300 orders, and every algorithm was run five times on each group of orders. Comparison across algorithms was made on two levels: performance, meaning the length of the path found, and runtime for one iteration. Since the Savings Algorithm is a lightweight procedure intended to quickly return good-quality solutions, while the other two models are designed to improve their search over time, the former was kept out of the runtime-based analysis. Table 1 shows the average distance, standard deviation and runtime recorded by the three algorithms over the fifty runs. The paths found by the proposed GAJO are on average 75.13% shorter than those found by the Hybrid Algorithm and 11.93% shorter than those found by the Savings Algorithm. Furthermore, an iteration of GAJO is on average 82.46% faster than an iteration of the Hybrid Algorithm.

5 CONCLUSIONS

This work proposes an efficient algorithm for the optimization of the joint Order Batching - Order Picking Planning Problem. We point out that, to the best of our knowledge, this is the first successful attempt to the joint optimization of the OBP-OPP. The experimental results show that the proposed GAJO greatly outperforms both considered baseline models, while also being significantly faster.

REFERENCES

[1] C. Y. Cheng, Y. Y. Chen, T. L Chen, and J. J. W Yoo. 2015. Using a hybrid approach based on the particle swarm optimization and ant colony optimization to solve a joint order batching and picker routing problem. *International Journal of Production Economics* 170, C (2015), 805–814. https://doi.org/10.1016/j.ijpe.2015.03.021
[2] G. Clarke and J. W. Wright. 1964. Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Oper. Res.* 12, 4 (Aug. 1964), 568–581. https://doi.org/10.1287/opre.12.4.568
[3] N. Gademann and S. Velde. 2005. Order batching to minimize total travel time in a parallel-aisle warehouse. *IIE transactions* 37, 1 (2005), 63–75.
[4] S. Henn, S. Koch, K. Doerner, C. Strauss, and G. Wäscher. 2009. Metaheuristics for the Order Batching Problem in Manual Order Picking Systems. *Otto-von-Guericke University Magdeburg, Faculty of Economics and Management, FEMM Working Papers* (01 2009).
[5] C. E Miller, A. W. Tucker, and R. A. Zemlin. 1960. Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)* 7, 4 (1960), 326–329.