International Journal of Production Research

ISSN: 0020-7543 (Print) 1366-588X (Online) Journal homepage: http://www.tandfonline.com/loi/tprs20

Joint optimisation of order batching and picker routing in the online retailer's warehouse in China

Jianbin Li, Rihuan Huang & James B. Dai

To cite this article: Jianbin Li, Rihuan Huang & James B. Dai (2017) Joint optimisation of order batching and picker routing in the online retailer's warehouse in China, International Journal of Production Research, 55:2, 447-461, DOI: 10.1080/00207543.2016.1187313

To link to this article: http://dx.doi.org/10.1080/00207543.2016.1187313

Published online: 19 May 2016.

International Journal of Production Research, 2017
Vol. 55, No. 2, 447–461, http://dx.doi.org/10.1080/00207543.2016.1187313

Taylor & Francis
Taylor & Francis Group

Joint optimisation of order batching and picker routing in the online retailer's warehouse in China

Jianbin Li[a], Rihuan Huang[a] and James B. Dai[b]*

[a] *School of Management, Huazhong University of Technology and Science, Hubei, P.R. China;* [b] *Economics and Management School, Wuhan University, Hubei, P.R.China*

(*Received 1 August 2015; accepted 25 April 2016*)

Order picking is the core of warehouse operations and considerable researches have been conducted on improving its efficiency. In this paper, we aim at the joint optimisation of order batching and picker routing based on a famous and typical online retailer of China, which mainly focuses on fast-moving consumer goods. An integer programming is formulated to minimise the total travelling distance involving with order batching and picker routing. In the stage of order batching, an effective batching procedure based on similarity coefficient which is measured by overlapping channels between orders is proposed. In the stage of picker routing, an improved ant colony optimisation algorithm with local search is proposed. Based on those simulated orders generated by actual transaction data, numerical experiments are conducted to verify the performance of the algorithm we proposed. Results show that the proposed joint optimisation algorithm has potential advantages under various order sizes and order structures, which implies that it is effective and efficient particularly in the online retailing of fast-moving consumer goods.

Keywords: warehousing systems; ant colony optimisation; routing; order batching; efficiency analysis

1. Introduction

The rapid development of B2C electronic commerce draws public attraction on efficiency of warehouse operations. As the core of warehouse operations, order picking determines performance of warehouse operations in terms of both cost and responsiveness. Order picking is dealing with the retrieval of stock keeping units (SKUs) from their storage locations to satisfy a given demand specified by one or more customer orders. These SKUs are then sent to the sorting process for shipment of each order. Generally speaking, order picking accounts for about 60% of total labour cost in warehouse, while 90% of picking time is spent on travelling. Therefore, considerable researches have been focusing on improving the performance of warehouse operations by improving the efficiency of order picking, such as the minimisation of response time and travelling distance. There are numerous factors affecting the efficiency of order picking and some of key factors are warehouse design, storage location allocation and picking policy.

As for order picking, warehouse design mainly refers to the design of picking system. If a picker travels to storage location to pick SKUs, the system is defined as a 'picker-to-part' system. If SKUs are brought to pickers, which means there is no interface between storage position and pickers, the system is defined as a 'part-to-picker' system. 'Part-to-picker' system is relatively new and has been widely used in practice in United States. One of 'part-to-picker' systems is automated storage and retrieval system (AS/RS). Generally, an AS/RS consists of racks served by cranes running through aisles between the racks. An extensive literature review of AS/RSs has been presented in Roodbergen and Vis (2009) discussing different types of AS/RSs including single unit-load aisle-captive AS/RS, mini-load AS/RS. The mini-load AS/RSs are very common in warehouse of online retailers where pickers could take the required amount of units from the bins and then the AS/RS moves the remainder of the load back into the storage rack (SR). Therefore, the mini-load AS/RSs have received much attention (see Vasili et al. 2008; Lerher, Šraml, and Potrč 2011; Lerher, Edl, and Rosi 2014). Mostly, a AS/RS is single-deep, which means it is a two-dimensional (2D) system. Recently, to improve order picking efficiency and to save floor space, multi-deep (or three-dimensional (3D)) storage systems have been introduced on the market (Yu and de Koster 2012). For instance, a new system based on AS/RS, called vertical lift modules (VLMs) is being concentrated on, where insertion/extraction device is travelling vertically and extracts trays from the shelves and brings them to the operator putting it on pick shelf (or pick window). Due to their high efficiency, VLMs of both single-tray and double-tray have received attention (see Dukic, Opetuk, and Lerher 2015; Rosi et al. 2016).

*Corresponding author. Email: bin_dai@whu.edu.cn

Beside AS/RS, another automated system that belongs to 'part-to-picker' is very important as well: shuttle-based storage and retrieval system (SBS/RS). SBS/RSs are designed to meet the demand of faster delivery time, smaller order size and larger product variety (Lerher, Ekren, Sari et al. 2015). The SBS/RS is composed of multiple parallel aisles of SRs, elevator (or lift) intended for each aisle of the SR, tier-captive shuttle carriers, input and output location, buffer position in each tier and roller conveyors (Lerher, Ekren, Dukic et al. 2015). Compared to mini-load AS/RS, SBS/RS brings high-throughput capacity, while its capital and maintenance costs are relatively high (Lerher 2015). Therefore, more researches about SBS/RS need to be done to make it more accessible.

However, automated system is not so popular in China due to its high investment cost. Warehouses of most online retailers prefer to adapt 'picker-to-part' system. Therefore, improving efficiency of pickers draws much importance. In the three key factors that affecting performance of order picking mentioned above, warehouse design and storage location allocation are difficult to be changed once a warehouse starts to operate. Hence, we try to optimise order picking process instead. One of the picking strategies used in practice of electronic commerce is the order batching policy, to allocate a set of orders into several subsets named order batch. Picking of an order batch is usually finished by a single picker. From the perspective of optimisation, a complete order batching and picking policy is composed of two aspects: how to choose appropriate orders to form a batch (referred as order batching in the following sections for convenience) and how to find the best picking sequence for an order batch which means picker routing.

The order batching strategy is not only a short-term and operational decision, but also a long-term and strategic one because it involves both operating process (e.g. routing for order pickers) and strategic decision such as warehouse layout, see Rouwenhorst et al. (2000). The objective of the order batching strategy is to minimise the picking route distance in a manual sorting system particularly (see Gademann and Velde 2005). In a multi-block warehouse, SKUs in a batch can be picked either by several workers in distinct blocks at one time (Lambert, Stock, and Ellram 1998) or by a single picker crossing different zones. The batch size is a critical factor determining the picking efficiency, and usually it can be decided by the expected batch processing time according to Petersen (2000). However, in practice, the batch size will be constrained by the capacity of retrieval vehicles. Order batching problem has been proved as a NP-hard problem and many researchers aim on exploring more effective heuristic solutions. For instance, Chen and Wu (2005) use clustering algorithm for two-binary integer programming to batch orders to maximise similarity of orders in a batch. Some other scholars consider a 'seed order heuristic'. For each batch, one order is chosen as a seed order to be compared with other orders. Seed order can be chosen by different rules such as the order with most SKUs by de Koster, van der Pooft, and Wolters (1999). And this seed order heuristic is also used in batching picking in a warehouse with double aisles by Ho, Su, and Shi (2008).

The essence of picker routing problem is a travelling salesman problem (TSP) which has been shown to be NP-hard, and many effective algorithms have been adapted to solve TSP. However, warehouse layout must be taken into consideration when designing heuristics to construct a picking route because it could decide whether the TSP to be solved is symmetric or asymmetric. For example, a warehouse with two depots could lead to a asymmetric TSP (Gharehgozli, Yu et al. 2014). Considering a traditional warehouse layout without any middle aisle, Goetschalckx and Donald Ratliff (1988) shows that traversal policy, return policy and other improved heuristics based on these two policies are applicable. While in a multi-block warehouse with several blocks separated by aisles or a double-block warehouse with only one middle aisle, Roodbergen and de Koster (2001a) has shown that warehouse with one middle aisle can have a significant shorter routing distance than that of those warehouses without aisles. Furthermore, Vaughan (1999) finds that the layout with middle aisles impacts the flexibility and efficiency of order picking. Roodbergen and de Koster (2001b) compares the average time of order picking and finds that the double-block warehouse layout is preferred. The S-shape heuristic is tested to be effective and feasible for practical use by Le-Duc and de Koster (2007).

Some related and similar works in other industries could also be referred such as optimisation of operations in a container block which could be regard as a warehouse. In this case, scheduling of cranes has large impact on the efficiency and many existing studies about it could be found (see Gharehgozli et al. 2014, 2014a, 2014b). An other case is batching problem in steel-making industry, which is more complicated in some extent because optimal sequence in a batch also needs to be decided (e.g. Tang and Wang 2008; Wang and Tang 2008).

To the best of our knowledge, most existing literatures focus on a single process in order picking, and joint optimisation receives few attention. The main reason might be that it is difficult enough to solve order batching or picker routing, respectively, since either problem has been proved as NP-hard. For order batching, decision-makers often faces multiple goals in designing batching policy, such as shorter picking distance, higher utilisation of retrieval vehicle and less order holding time. The problem is, in most cases these goals cannot be achieved at the same time. Therefore, finding a balance to improve performance of the whole warehouse operation is critical and not easy as well. As for picker routing which could be regarded as TSP, numerous works about it have been presented and it still has been solved perfectly. The main difficulty is that most exact algorithms could not solve the problem in a reasonable time when problem size becomes large, while heuristics and intelligent optimisation algorithms are even unable to find the optimal solution every time. Hence, either order

batching or picker routing is hard to find an optimal solution. As a result, it will be even harder to solve both problems at the same time.

However, the order batching and picker routing policy is highly interrelated in warehouse operations. In addition, an important fact of online retailers in China is that most work in warehouse operations is finished by manpower due to low labour cost. Certain online retailers even employ amount of temporary workers to deal with heavy work induced by promotion. Usually, workers decide a picking route based on their own experience, which is inefficient for most of the time. Situation turns worse when it comes to temporary workers. As warehouse management system (WMS) is becoming a common tool for application, order picking can be optimised to significantly minimise workers' individual difference.

Won and Olafsson (2005) design two heuristics to solve order batching and picker routing jointly and minimise both order holding time and order picking time. However, only 2-opt heuristic is adapted for routing in this research. Tsai, Liou, and Huang (2008) design a multiple-GA method to solve the same problem and receive better performance. In this study, order batching and picker routing are solved by two different genetic algorithms (GA). One of the limitation is that it might fail to make full use of capacity of retrieval vehicle in batching process. Besides, GA does not perform well in picker routing compared to other intelligent algorithms. Recently, some researches of related problems mainly focus on designing a better intelligence algorithm, especially for order batching, to minimise the total routing distance. For instance, Kulak, Sahin, and Taner (2012) design a cluster-based tabu search algorithm, Chen et al. (2015) propose a hybrid-coded GA and Cheng et al. (2015) present a hybrid algorithm incorporates particle swarm optimisation and ant colony optimisation. These new algorithms may be able to acquire a better solution, but they also induce a longer computation time when constructing order batches, which is unrealistic for application. All of these motivate us to propose a more efficient solution on joint optimisation of order batching and picker routing under consideration of practical use, particularly in the online retailers' warehouse in China which mainly focus on fast-moving consumer goods.

In this paper, we examine a joint optimisation of order batching and picker routing policy by formulating an integer programming mainly to minimise the total picking distance, and then an effective heuristic solution is presented. We mainly consider to minimise the picking distance for the reason that less time spent on picking could also reduce holding time of other orders. Additionally, order waiting time is taken under consideration in the algorithm proposed in this paper. Besides the performance of heuristics, we consider the practical applicability in designing the heuristic solution as well. Our research is based on investigation in Yihaodian, a well-known Chinese B2C online retailer focusing on fast-moving consumer goods (FMCG) at first and now other categories as well.

The main contribution of this paper is: (1) optimisation of warehouse operations is divided into three specific aspects for further research; (2) an integer formulation considering practical situation is given; (3) we design an effective algorithm to batch orders and construct a picking route to minimise total picking cost. To be specific, we batch orders to maximise similarity of orders in the one order batch where similarity refers to that orders in the batch demand SKUs stored at near location. Besides, we combine two local search processes with ant colony optimisation to improve the result of picker routing; (4) a procedure to effectively simulate orders based on historic selling data is given; (5) the algorithm is proved to be effective than other algorithms and practical under various situations.

The rest of the paper is organised as follows. In Section 2, detailed description of this problem and related background is presented. Section 3 first introduces some method to simplify the practical problem for convenience of research, and then proposes an integer formulation of order batching and picking. In Section 4, a heuristic of order batching and picker routing is prescribed. Section 5 gives a numerical example based on real transaction data to compare the efficiency of the heuristic proposed and traditional routing policy in practice. Lastly, Section 6 concludes main works in the paper and suggests several potential research directions.

2. Problem description

As a cruel fact, profiting from China is difficult for most online retailers, which forces them to reduce cost in every aspect. As a major part in daily warehouse operations, order picking has attracted considerable attention on improving efficiency. Thus far, the 'picker-to-part' system is widely used in practice, and in a 'picker-to-part' system, order picking generally can be optimised in the four following ways:

- *Warehouse layout*: The optimisation of warehouse layout is to arrange different items into warehouse for efficient and cost-effective operation. Once the layout is decided, it will bring high cost to re-layout.
- *Position allocation*: Allocating SKUs to storage locations tends to find the best assignment to reduce the total picking distance. For instance, SKUs with high outbound frequency should be placed in the nearest positions. Besides, storage requirements for certain SKUs must be taken into consideration.

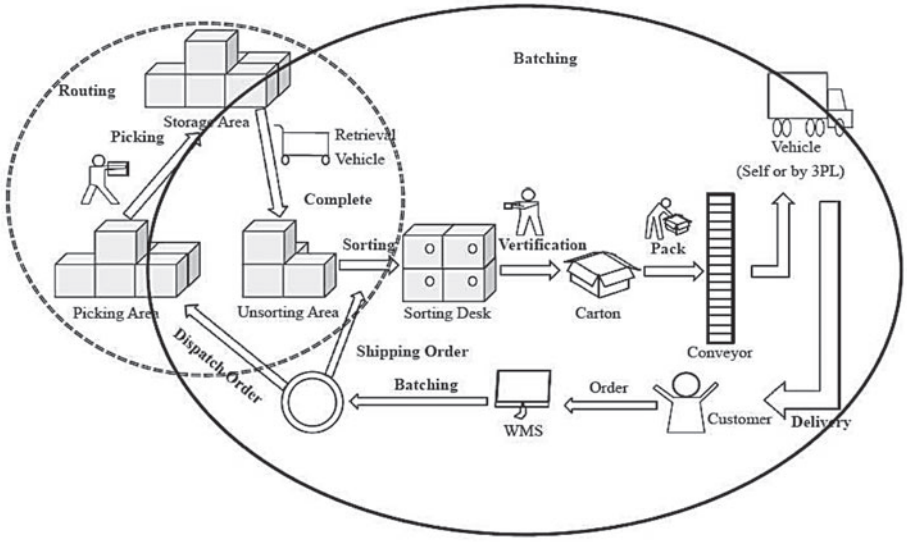


Figure 1. Effect of order batching and picker routing on warehouse operations.

- *Order batching*: A picking tour usually fulfils an order batch, containing several or more orders. This is a useful strategy to improve efficiency. The problem is to identify the best group of orders to form an order batch. In addition, order batching is a trade-off between cost and waiting time for consumers.
- *Picker routing*: Finding the shortest route to pick all demanded SKUs has received extensive attention. This problem can be transferred to a TSP. For a traditional warehouse layout, distance usually is described using rectilinear distance because it is close to practical situation.

For an online retailer which has operated for several years, the warehouse layout has been decided and it may induce very high cost to change the layout. Although the position allocation is made considering various factors, e.g. storage condition and storage by categories, such allocation strategy will not be pursued here. Therefore, the retailer will try to reduce the picking cost by optimising order batching and order picking routes. In this paper, a joint optimisation of these two parts is considered because order batching and order picking routes have internal relation and each aspect affects the final outcome.

Figure 1 shows the effect of order batching and picker routing on warehouse operations. The solid circle refers to order batching and the other one represents the picker routing. It indicates that these two aspects cover most work in order fulfilment, and this is the underlying reason that the paper concentrates on the investigation of this problem.

The following assumptions are made based on practice to simplify the problem:

(a) The distance between any two positions is known for a given warehouse layout, and it is measured by the rectilinear distance.
(b) Both the horizontal speed and vertical speed of the picker is constant and known.
(c) One order cannot be split into multiple batches, which indicates that the capacity of a retrieval vehicle is enough to satisfy at least one order.
(d) Order picking starts and ends in the sorting area. Usually only one sorting area exists in a warehouse.
(e) The same SKUs stored at different positions (i.e. more than one position) will be regarded as different SKUs.

For assumption (a), in practice, aisle and channel might be crowded sometimes and therefore the picker is forced to find another path which is not the shortest. This condition will not be extended in this paper and rectilinear distance between any two positions is set to be the shortest one. Assumption (b) is to support the translation of the vertical distance to the horizontal distance given in the next section. Assumption (d) is for convenience of management and it is common in practice.

3. Joint optimisation formulation

For a modern multi-block stereoscopic warehouse, the storage location of a SKU can be determined in terms of block, row, column, level, bay and position. In practice, each rack is divided into several bays, each of which contains several positions.

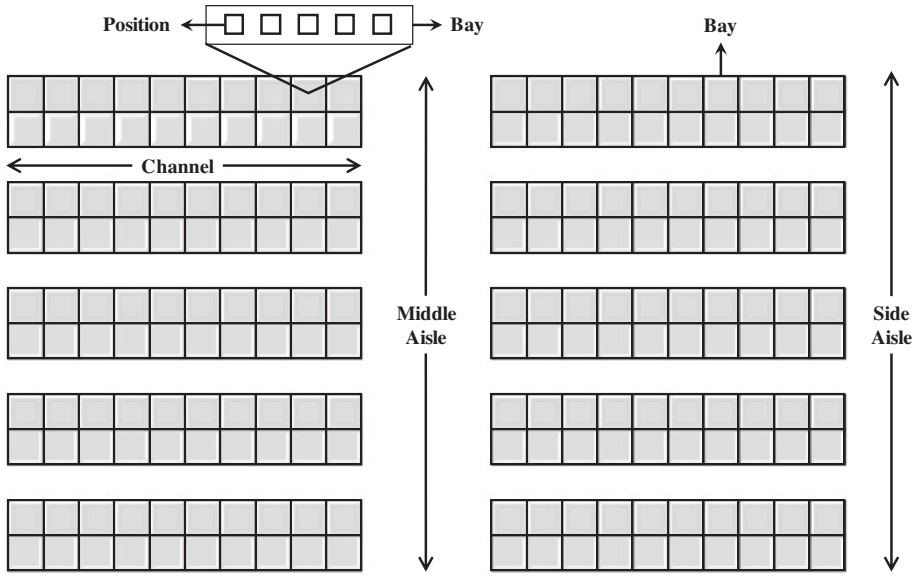


Figure 2. A typical warehouse layout.

Table 1. Notation of parameters in the formulation of joint optimisation problem.

Notation	Description
F	Number of levels in a SR, indexed by f
H	Height between two adjacent levels in a SR
v	Constant horizontal speed of the retrieval vehicle
h	Constant vertical speed of the retrieval vehicle
ω	Transferred coefficient where $\omega = v/h$
m, n	Index for a storage location
c^{mn}	Rectilinear distance between location m and n
P	Number of SKUs, indexed by p and q
c_{pq}	Rectilinear distance between the SKU p and q
v_p	Unit size of the SKU p
I	Number of orders, indexed by i and j
S_{ij}	Similarity coefficient between the order i and j
V	Storage capacity of the retrieval vehicle (in units)
O_{ip}	Quantity of the SKU p demanded by the order i
G_i	Set of SKUs demanded by order i
K	Number of order batches, indexed by k
G^k	Set of SKUs demanded by the batch k
A_{ip}	Channel at where the SKU p in the order i locates
A_i	Set of Channels where SKUs in the order i locate

For instance, A02-14-08-03-06 refers to position 06 on the third level of bay 08 located in row 02, column 14 in block A. A row and a column can locate a rack while a bay and a position can locate a SKU in the rack. To make it clear, we define the aisle between rows as corridor and aisle between columns as channel. Figure 2 shows a typical layout of a double-block warehouse.

The aforementioned configuration of code for storage locations can improve efficiency and avoid mistakes in daily operations, which makes it prevalent in practice. However, it is unnecessary in research. Therefore, to reduce the complexity of the problem, we further simply the definition to an one-dimensional (1D) code by the following procedure:

Step 3.1: Encode each position by an unique code.

Step 3.2: For storage positions at the same planar location but in different levels, transfer the vertical picking distance into a horizontal picking distance using a transferred coefficient $\omega = v/h$, where v and h are the horizontal and vertical speed of the retrieval vehicle, respectively.

Let F denote the number of levels (indexed by f), and H is the height between two adjacent levels in a rack. In practice, the medium level is usually comfortable for picking from the perspective of the total picking distance. We define the medium level to be the basic level. Thus, the vertical picking distance of the level f can be transferred into a horizontal picking distance by:

$$\text{transferred horizontal distance of the level } f = \omega H \left| f - \frac{F}{2} \right| \tag{1}$$

In this way, the picking distance between the storage location n and m, denoted by c^{mn}, can be evaluated by summing the horizontal distance and the transferred horizontal distance. To clarify the discussion, parameters and their definitions are presented in Table 1.

In stage of picking, a picker is given SKUs needed to be retrieved while distance discussed above is between two locations. Therefore, it would be more intuitionistic if distance between any two SKUs is calculated. Once decision of storage location allocation is known, the rectilinear distance between the SKU p and q is given by:

$$c_{pq} = c^{mn} \qquad \forall p(q) \in P \text{ stored in the storage location } m(n) \tag{2}$$

Note that distance between any two nodes is assumed to be symmetric in this paper, which means $c_{pq} = c_{qp}$ for any p and q.

In the joint optimisation of both order batching and picker routing, we aim to identify the optimal order batching policy and the optimal picking routes. The order batching policy indicates the total number of order batches K (indexed by k) and the allocation of orders to batch k, which is defined by Equation (3). The optimal picking route indicates the visiting sequence the SKU p and q, denoted by y_{pq}^k, which is defined by Equation (4).

$$x_i^k = \begin{cases} 1 & \text{order } i \text{ is allocated in batch } k \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

$$y_{pq}^k = \begin{cases} 1 & \text{The SKU } q \text{ is picked just after the SKU } q \text{ during the picking of batch } k \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Therefore, the joint optimisation of order batching and picker routing can be formulated by an integer program which is given by:

$$\min \sum_{k \in K} \sum_{p \in G^k} \sum_{q \in G^k} c_{pq} y_{pq}^k \tag{5}$$

$$\text{s.t.} \sum_{k \in K} x_i^k = 1 \qquad \forall i \in I \tag{6}$$

$$\sum_{i \in I} \sum_{p \in P} x_i^k O_{ip} v_p \le V \qquad \forall k \in K \tag{7}$$

$$G^k = \bigcup\nolimits_{i \in I} x_i^k G_i \qquad \forall k \in K \tag{8}$$

$$\sum_{p \in G^k} y_{pq}^k = 1 \qquad \forall k \in K, \forall q \in G^k \tag{9}$$

$$\sum_{q \in G^k} y_{pq}^k = 1 \qquad \forall k \in K, \forall p \in G^k \tag{10}$$

$$\sum\nolimits_{p \in D, q \notin D} y_{pq}^k \ge 1 \qquad \forall k \in K, D \text{ is any subset of SKUs set in batch } k \tag{11}$$

$$x_i^k, y_{pq}^k = 0 \text{ or } 1 \tag{12}$$

The objective function (5) is to minimise the total routing distance in the order picking. By assuming a constant velocity of the retrieval vehicle, picking distance could be transferred into picking time. In other words, minimising total distance equals minimising total picking time. This assumption is common in existing research such as Won and Olafsson (2005) and Tsai, Liou, and Huang (2008). The constraint (6) ensures that each order can only be allocated in one single order batch, and $K \le I$. Constraint (7) limits that total number of orders in a batch cannot exceed the capacity of the retrieval vehicle.

The constraint (8) calculates the SKU set in an order batch. The constraint (9)–(11) are typical constraints in TSP ensuring the solution is a Hamilton cycle. Constraint (9) and (10) ensures the uniqueness of picking route. Specifically, constraint (9) means that only one SKU could be arranged to be picked just before picking SKU q. Constraint (10) ensures that any SKU p, the picker could only go to pick one SKU right after picking SKU p. Finally, constraint (11) ensures a complete picking route, avoiding the situation where a subset of SKUs to be picked forms a circle and other SKUs are not put in the route.

The problem could be simplified into a traditional order routing problem if we only consider decision variable y_{pq}^{k} and delete constraint (6)–(8). However, the problem could not be simplified into an order batching problem easily because the objective must be changed when only batching is considered, but constraint (6) and (7) are still two necessary constraints order batching problem. Actually, constraint (8) acts as a link to connect order batching and order routing by finding SKUs set in a given order batch so that picking route for the batch could be decided.

To show the complexity of the problem, consider the case where routing is not considered. Then the problem is simplified to order batching, which has been proved as an NP-hard problem (Gademann, van den Berg, and van der Hoff 2001; Gademann and Velde 2005). On the other hand, in the case without order batching, it is actually a TSP as mentioned above and it has been proved to be an NP-hard problem as well. Therefore, as a more complicated one, joint optimisation of order batching and picking is NP-hard as well.

The model could be extended to be more realistic in several directions. The first one is to consider order waiting time in batching process. To be specific, an order cannot be held in system for too long, otherwise it will affect customer satisfaction. Therefore, the objective function could be modified to be a multi-objective optimisation problem. The second one is to allow the routing problem to be asymmetric because distance between two nodes could be different due to congestion. In practice, there will be many pickers in the warehouse at the same time, making pickers unable to choose the shortest route every time. In this case, the distance will be asymmetric.

4. Heuristic solution

In this section, we purpose a heuristic solution approach to jointly optimise order batching and picker routing. The retailer makes decisions to minimise order picking cost after the actual content of orders is observed. Once a number of orders are realised, the retailer batches these orders and compute picking route for each batch to minimise the picking distance or time. Both sets of decisions will affect the cost induced by order picking. Storage location allocation is assumed to be decided and the retailer seeks to save transportation cost in fulfilling orders in an efficient way. When customers place their orders, the retailer aims to prepare the SKUs for shipment in the shortest time, which mostly depends on the efficiency of picking. Two approaches will be used in this stage: order batching and picking route computation.

It is called order batching that grouping orders into a number of sets, each of which can be finished by a single picking tour. This is a common strategy in practice. A batch is composed based on different criterion, such as capacity of picking equipment and expected waiting time of an order. Usually, a set of orders that in sequence of arriving time will be grouped in a batch, which means first in first out (FIFO) principle. However, batches composed in this way may require long picking time due to stochastic demand of orders. In other words, orders in a batch using FIFO principle have few similarities from the perspective of picking. Such effect is much more significant in FMCG. Hence, in this section, we discuss how to generate orders considering optimisation effect and practicability. Besides, for a given order batch (or a given sets of demand SKUs), the picker has multiple choices of picking sequence which affects efficiency. Therefore routing algorithm will be discussed in this section as well.

It is easy to recognise that generating similar orders is a feasible method. The point is the measurement of similarity. Considering the fact that moving from a channel to another usually takes more travelling time, we batch orders that demand SKUs in similar channels. On one hand, an order that demands SKUs in one channel can be fulfilled easily with other orders which will visit the same channels. On the other hand, for an order that demands SKUs in wide apart channels, batching more orders makes long tour between channels and more worthy.

Note that for $p \in G_i$, A_{ip} is the channel where SKU p located. A_i is set of channels involved in order i, which means $A_i = \bigcup_{p \in G_i} A_{ip}$. Let S_{ij} denote similarity coefficient between order i and order j, where S_{ij} is size of set $A_i \bigcap A_j$. In other words, S_{ij} refers to number of overlapping channels between order i and j. To be specific, order batching strategy using similarity coefficient in terms of overlapping channels is named batching with overlapping channels (BOC), distinguished from the FIFO strategy.

The main idea of BOC is to combine two similar orders as one which is regarded as a new order until picking equipment is unable to fulfil a larger order in one trip. Batching process starts when certain quantity (denoted by N) of orders are placed waiting for fulfilment. These orders form an unprocessed order list. Two strategies can be applied in batching: static batching and dynamic batching. Static batching generates orders into several batches without adding new orders to the list, while dynamic batching replenish orders to maintain N orders in unprocessed list once a batch is composed. In static batching, the

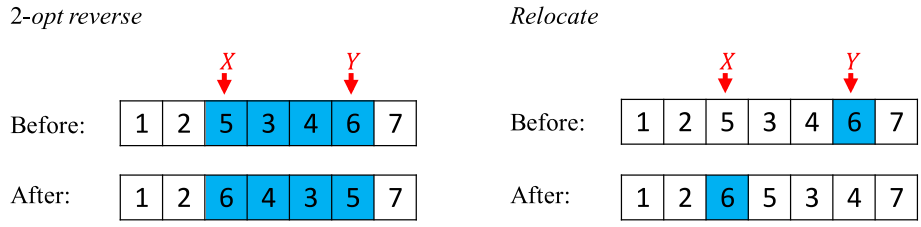


Figure 3. An example for local search process.

next order list is added only when the current order list is finished, which leads to relatively long waiting time for orders in the system. Therefore dynamic batching will generate better performance.

However, in dynamic batching, orders with more SKUs tend to be batched together at first when applying similarity coefficient by overlapping channels. In result, some orders of small size have to wait for a longer time. Hence, we adapt channel overlapping rate with seed order to evaluate similarity. The order with most SKUs in the list is chosen as the seed order. Other orders in the list are only compared to the seed order. Let A_{i^*} be set of channels of seed order i^*, then the overlapping rate of an order j is defined as:

$$S_{i^*j} = \frac{\text{size of } A_{i^*} \bigcap A_j}{\text{size of } A_j} \tag{13}$$

For instance, if order j has five channels, and three of them are also included by the seed order i^*, then the channel overlapping rate S_{i^*j} is 60%. This indicates orders with less SKUs may have a higher overlapping rate, which benefits the balance of workload.

Besides, a feasible s-shape picking route can be easily computed when a batch is decided because channels to be visited are already known in the batching process. This picking route can be regarded as sequence of visiting channels included in this batch from the nearest one to the farthest one. If the picking distance has to be further optimised using intelligent optimisation algorithms, such a route can be an initial solution.

Based on discussion above, the dynamic batching process is presented as follows:

Step 4.1: Choose the order with the most SKUs as the seed order i^*;
Step 4.2: Calculate the similarity coefficient S_{i^*j} of other orders with order i^*, $S_{i^*j} = -1$ if combined order of i^* and j exceeds the capacity constraint.
Step 4.3: If $S_{i^*j} = -1$ for every order j in the list, then go to Step 4.7, otherwise go to Step 4.4.
Step 4.4: Sort orders by S_{i^*j} in descending order;
Step 4.5: Select order (i^*, j) with the highest S_{i^*j}, if multiple orders have the same similarity coefficient, choose the order j that arrives first;
Step 4.6: Combine order j and i^* as a new order i_t, add order i_t back to the set of unprocessed orders. If the set contains only order i_t then go to Step 4.7, otherwise go to Step 4.2;
Step 4.7: Output the order batch. Add new orders to the set of unprocessed orders in the sequence of arrival time, ensure the set remains N orders and then go to Step 4.1.

The initial route has to be improved by intelligent optimisation algorithms. In the rest of this section, ant colony algorithm (ACO) will be adapted. ACO tries to imitate the behaviour of a colony of ants when searching for food (Colorni, Dorigo, and Maniezzo 1991). The ants release a substance called pheromone on the route they are searching to exchange information with others so ants can judge a promising path by amount of pheromone. ACO has been adapted to solve TSP and relevant problem with promising performance (Dorigm and Gambardella 1997).

We consider a combination of ACO and local search (ACOLS) by following process: each time when a feasible route is constructed by an ant, we perform two local search processes in sequence on the solution, *2-opt reverse* and *relocate* used in Zhang et al. (2013). Both processes first uniformly choose two nodes X and Y in the route. In *2-opt reverse*, sequence between node X and node Y will be reversed. In *relocate*, node Y will be inserted before node X. New route will be compared to the old one and the better one will be accepted. For each process, the terminal condition is that the route has not been improved in 100 continuous iterations. Main idea of two local search processes is shown in Figure 3. These two local search processes can search the neighbourhood and improve the solution. When two processes is finished and out the final route, pheromone will be updated based on this route. Therefore, the whole algorithm of batching and picking is named BOC-ACOLS.

Table 2. The setting of SKUs and orders.

Setting of SKUs					Setting of orders					
Type	W_s	$AvgD_s$	$maxD_s$	P_s	Type	R_o	$Size_o$	Type	R_o	$Size_o$
A	$0.0030P$	9	12	0.03	a	$0.28I$	2	g	$0.05I$	8
B	$0.0097P$	4	7	0.07	b	$0.19I$	3	h	$0.04I$	9
C	$0.0400P$	3	6	0.25	c	$0.15I$	4	i	$0.01I$	10
D	$0.5800P$	2	4	0.60	d	$0.12I$	5	j	$0.006I$	11
E	$0.3673P$	1	2	0.05	e	$0.08I$	6	k	$0.004I$	12
sum	P	–	–	1	f	$0.07I$	7	sum	I	–

In next section, BOC-ACOLS will be compared to other algorithms to test efficiency. We mainly consider three algorithms. The first one is a common batching and picking strategy applied in practice of warehouse operation including Yihaodian's warehouses: FIFO policy in batching and s-shaping heuristic (SS) in routing, shorten as FIFO-SS. As it is mentioned in former part of this section, FIFO policy is to batch orders in the sequence of arriving time until the retrieval vehicle cannot hold one more next orders. SS is to pick SKUs in a route shaped like an 'S', which is easy for a picker to understand and execute. The second heuristic is sequential order batching and picking (SBP) proposed by Won and Olafsson (2005). It jointly considers order picking time and order holding time in the batching process and uses the well-known 2-opt heuristic method to calculate the picking route. The third one is a multi-GA method used by Tsai, Liou, and Huang (2008). GA is a widely used evolutionary algorithm to solve different problems. The multi-GA uses two different GA in batching process and routing process separately. All these algorithms will be tested in a numerical experiment presented in the next section.

5. Numerical experiment

In this section, some numerical results are presented to compare the efficiency of heuristics. These orders are simulated based on the transaction data of Yihaodian, a famous online retailer in China mentioned above, by the following procedure:

Step 5.1: Divide all SKUs into S types (indexed by s). For a certain type s, there are four parameters: number of SKUs in this type (referred as W_s), average demand in an order (referred as $avgD_s$), maximum demand in an order (referred as $maxD_s$) and the possibility of appearance (referred as P_s).
Step 5.2: Set O order types, indexed by o. For a certain type o, there are two parameters: number of orders in this type (referred as R_o) and number of SKUs it contains (referred as $Size_o$). Order types are simply different in terms of number of SKUs. In this example, the number of SKUs in an order varies from 2 to 12. Therefore O is set to be 11.
Step 5.3: For each order type o, draw R_o orders randomly from all I orders to form the order type o.
Step 5.4: For each order i, retrieve order type o and $Size_o$ of order i. For each SKU vacancy in $Size_o$ do:

Step 5.4.1: Decide the SKU type s using the probability P_s.
Step 5.4.2: Retrieve SKUs list of type s, choose a SKU p randomly from the list and add it to order o.
Step 5.4.3: If SKU p already exists in order i, then go back to Step 5.4.2.
Step 5.4.4: Generate a random demand quantity of SKU p in order i using average demand $avgD_s$ and maximum demand $maxD_s$. The demand quantity is assumed to follow a Poisson distribution in this paper.

Step 5.5: Output all the simulated orders.

In this numerical experiment, the SKU type is classified by sales for a few SKUs account for major sales. Note that the performance of simulation improves as the number of SKU types increases. Detailed settings of parameters are shown in Table 2. The data settings are based on sales data of Yihaodian in November 2013. Actually, the classification can be more detailed with more advanced testing computer. All the notations for order simulation process is given in Table 3.

The simulated results are close to collected data, which are shown in Table 4. In Table 4, SKU is sorted by sales volume in a descending order, and we evaluate the simulation results by calculating the proportion of aggregated sales volume from the top $\delta\%$ of SKU in two sets of orders, the real one and simulated one. Value of $\delta\%$ is specified in the first row. As it is mentioned above, the simulated data will be closer to the real one if SKUs are divided into more classes. In this section, we only consider five classes due to the hardware constraint.

Table 3. Notation of parameters in the order simulation.

Notation	Description
S	All the SKUs are divided into S types, indexed by s
W_s	Number of SKUs in type s
$AvgD_s$	Average demand in an order for each SKU of type s
$maxD_s$	Maximum demand in an order for each SKU of type s
P_s	Possibility of appearance of type s in an order
O	All the orders are divided into O types, indexed by o
R_o	Number of orders of type o
$Size_o$	Number of SKUs in a type o order

Table 4. Evaluation of the order simulation.

SKU	0.3 (%)	1 (%)	5 (%)	55 (%)	63 (%)	70 (%)
Real data	10.00	20.00	55.00	98.00	99.00	99.50
Simulated data	10.18	21.04	51.70	97.98	98.23	98.58

Table 5. Configurations of a typical warehouse.

Positions	2000	Bin size	1.2 m × 1.2 m
Columns	100	Middle aisle width	4.8 m
Rows	40	Side aisle width	1.2 m
Aisles	3	Channel width	1.2 m
Channels	11	Vehicle capacity V	50 units, 100 units
Horizontal speed v	0.25 m/s	Vertical speed h	0.25 m/s

We consider the storage structure with a middle aisle, for example. It contains two identical zones connected by a middle aisle. Each zone consists of 1000 positions (50 columns and 20 rows). Each position is specified for one SKU and assumes that storage quantity of all the SKUs is enough to cover demand. Such a storage can be regarded as part of a whole warehouse and it contains all typical components in the warehouse layout. The detail description of the storage structure is shown in Table 5. In the numerical experiment, we consider the number of SKUs to be 2000. For simplification, the unit size of all SKUs is assumed to be equal and therefore the capacity of the retrieval vehicle could be set to 50 units or 100 units for different kinds of order setting. Orders are assumed to arrive chronologically at a speed of 900 orders per hour on average. In practice, retailers could receive much more orders.

We consider two classes of computational tests. Class 1 is a test with various quantity of orders and Class 2 is a test with various structures of orders. Class 1 is to examine the efficiency of algorithms under different problem size and Class 2 is to find whether the algorithms can deal with different demand structures. In Class 1, we test 200, 500, 1000, 5000 and 10,000 orders. In Class 2, number of orders is set to 5000, and four different order sets are given. These four order sets are designed in two dimensions: demand of each SKU (referred as order demand) and the number of SKUs in an order (referred as order size). Each dimension is divided into two levels: normal and large. In this experiment, for normal demand, the settings of $AvgD_s$ and $maxD_s$ are consistent with the data in Table 2 while the large one doubles the demand. Analogously, normal size means that $Size_o$ remains the same as the data in Table 2. In large size, we only consider orders with more than 5 SKUs. Division of demand is to simulate different situations in the B2C online retailing. Small demand may be accord with comprehensive retailers, but as for a retailer mainly selling FMCG such as snacks, nut, milk, customer may prefer to purchase more for two reasons: to reach the requirement of free shipping and customers do not need to carry goods home by themselves so purchasing more may bring higher utility. Considering the order size, actually, small size is more often in the online retailing. In this example, large size is mainly to test the efficiency of heuristics in some rare situations, e.g. November 11th, an online promotion day in China raised in recent years. Note that for orders in Class 2, the retrieval vehicle with a capacity of 50 units may be not enough; therefore, we assume capacity V is upgraded to 100 units to handle such kind of situation.

The setting of parameters for algorithms should be given before the experiment. For BOC-ACOLS, the size of unprocessed order list N is set to be 50. Number of ants equals to the number of SKUs to be picked, relative influence parameters

Table 6. Class 1: comparison under various quantity of orders ($V = 50$).

Order amount	Algorithm	Batch	Utl(%)	D_T (m)	Imp(%)	D_{avg}(m)	T_{avg}(s)
200	BOC-ACOLS	46	94.22	7,826.4	–	170.14	70.04
	FIFO-SS	51	84.98	45,096.0	82.65	884.24	–
	SBP	52	83.35	56,114.4	86.05	1079.12	–
	Multi-GA	60	72.23	46,934.4	83.32	782.24	9.28
500	BOC-ACOLS	108	96.74	19,665.6	–	182.09	65.77
	FIFO-SS	120	87.07	111,547.2	82.37	929.56	–
	SBP	122	85.64	142,104.0	86.16	1,164.79	–
	Multi-GA	148	70.59	119,157.6	83.50	805.12	8.12
1000	BOC-ACOLS	335	94.64	59,100.0	–	176.42	70.91
	FIFO-SS	377	84.10	335,738.4	82.40	890.55	–
	SBP	379	83.65	425,774.4	86.12	1123.42	–
	Multi-GA	450	70.45	358,816.8	83.53	797.37	10.58
5000	BOC-ACOLS	1,065	98.11	200,930.4	–	188.67	64.69
	FIFO-SS	1197	87.29	1,132,908.0	82.26	946.46	–
	SBP	1233	84.74	1,430,656.8	85.96	1160.31	–
	Multi-GA	1451	72.01	1,198,024.8	83.23	825.65	8.89
10000	BOC-ACOLS	2,144	98.03	399,844.8	–	186.49	64.82
	FIFO-SS	2,423	86.75	2,267,949.6	82.37	936.01	–
	SBP	2,465	85.27	2,864,808.0	86.04	1,162.19	–
	Multi-GA	2930	71.74	2,394,583.2	83.30	817.26	8.83

$\alpha_{\text{ACO}} = 1$ and $\beta = 5$, the pheromone evaporation coefficient $\rho = 0.5$. Ant-cycle model is adapted to update the pheromone and termination condition is not finding a better solution in 300 continuous iterations. Parameters of SBP mainly follows that of Won and Olafsson (2005) where both picking factor ϖ_1 and holding factor ϖ_2 are equal to 1. The most important parameter of SBP, the between batch time t, rangers from 5 to 30 min and t increases by 1 min each time. Note that we mainly compare the length of picking route computed by different algorithms while SBP considers picking time and holding time of orders. Therefore, results of SBP will be transferred into distance from time since speed of retrieval vehicle given in Table 5 is constant. Similar operations are done for multi-GA in Tsai, Liou, and Huang (2008). Besides, for multi-GA, number of batches is decided by Equations (11) and (12) in Tsai, Liou, and Huang (2008) while the parameter R_1 and R_2 need to be set by users. We let $R_1 = 1$ and $R_2 = 1.5$ based on some experiments. Each time multi-GA processes $N = 50$ orders which is the same as BOC-ACOLS. All tests are completed on a computer with a 2.7-Ghz Intel Core i5 CPU and 8 GB RAM using Matlab 2014b.

The result of Class 1 analysis is shown in Table 6. We use meter (m) to measure distance and second (s) to measure time. The third column gives the amount of batches calculated by each algorithms. 'Utl' in the fourth column refers to average utilisation of the retrieval vehicle by order batching. In the fifth column, D_T means the total distance in picking all batches. To compare efficiency of routing, 'Imp' in the sixth column calculates the percentage of distance improvement on the shortest distance compared to that of others. Then D_avg in the eighth column is the average picking distance of each order batch and T_avg in the final column is the average time in computation for each batch including batching and routing. Since both FIFO-SS and SBP are heuristics which finish computation in a very short time, we only compare the time efficiency between BOC-ACOLS and multi-GA.

Data in Table 6 shows that BOC-ACOLS outperforms other algorithms in allocating orders into less batches and computing a better route for each batch. Firstly, in order batching process, BOC-ACOLS divides orders into less batches compared to others and therefore achieves a higher utilisation of capacity. Multi-GA always leads to the most batches and one possible reason for it is that GA cannot converge to the optimal solution in many cases. As a result, its outcome may be even worse than that of FIFO-SS. Considering the picker routing, BOC-ACOLS always computes a shorter picking route with an improvement of more than 80%. FIFO-SS gets a higher efficiency than SBP mainly because SBP only adapts 2-opt heuristic in its routing. Though multi-GA computes more batches than others, GA still has advantage in picker routing compared to FIFO-SS and SBP.

Table 7. Class 2: Comparison under different order structures ($V = 100$).

Order set	Algorithm	Batch	Utl(%)	D_T(m)	Imp(%)	D_{avg}(m)	T_{avg}(s)
1	BOC-ACOLS	533	99.27	210,285.6	–	394.53	99.28
	FIFO-SS	569	92.99	1,006,334.4	79.10	1,768.60	–
Normal demand	SBP	585	90.44	1,371,892.8	84.67	2,345.12	–
Normal size	Multi-GA	698	75.80	1,197,472.8	82.44	1715.58	4.26
2	BOC-ACOLS	804	97.23	320,798.4	–	399.00	101.63
	FIFO-SS	849	92.08	1,515,410.4	78.83	1784.94	–
Normal demand	SBP	858	91.11	2,064,626.4	84.46	2406.32	–
Large size	Multi-GA	1058	73.89	1,805,275.2	82.23	1706.31	6.16
3	BOC-ACOLS	1090	97.67	198,513.6	–	182.12	66.34
	FIFO-SS	1232	86.41	1,135,027.2	82.51	921.29	–
Double demand	SBP	1238	85.99	1,427,940.0	86.10	1,153.42	–
Normal size	Multi-GA	1478	72.03	1,193,128.8	83.36	807.26	8.14
4	BOC-ACOLS	1723	92.76	302,395.2	–	175.51	61.05
	FIFO-SS	1919	83.29	1,701,847.2	82.23	886.84	–
Double demand	SBP	1923	83.12	2,142,960.0	85.89	1,114.38	–
Large size	Multi-GA	2247	70.29	1,797,580.8	83.18	799.99	13.07

Table 8. Comparison of CPU time for BOC-ACOLS and multi-GA to obtain the same routing distance.

Nodes	GA-$Time$ (s)	ACO-$Time_{avg}$ (s)	ACO-$Time_{min}$ (s)	ACO-$Time_{max}$ (s)
10	0.0167	0.0049	0.0001	0.0301
30	0.0203	0.0134	0.0009	0.1099
50	0.0299	0.0523	0.0232	0.1401

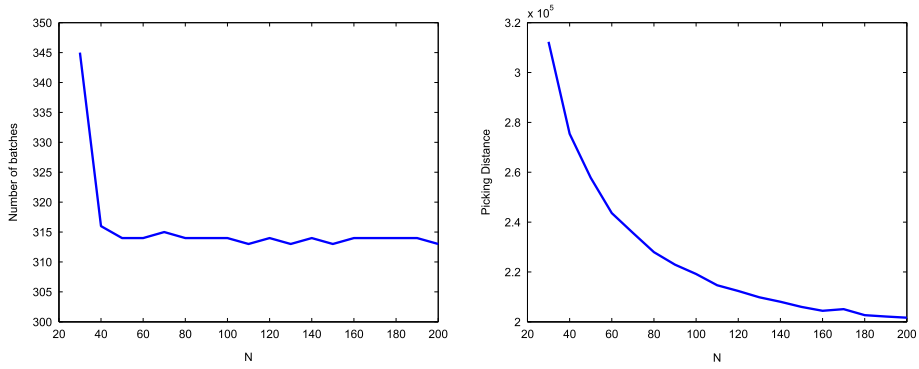
Table 7 presents the results of Class 2 test. The meaning of columns is the same as those in Table 6 and we obtain similar results. BOC-ACOLS is still the best among all algorithms from perspectives of both constructing less batches and finding a shorter route. As for order batching, BOC-ACOLS still outputs less order batches than that of others, which will induce less labour cost in order fulfilment. Besides, BOC-ACOLS can reduce the total picking distance by arranging similar orders into one batch. The data in the seventh column may be more persuasive because the BOC-ACOLS still computes the shortest distance for each batch. Utilisation is decreasing because both demanding units and order size is larger so it becomes more difficult to find suitable orders to construct a batch. Routing distance decreases in the case of double demand mainly because demanding units in an order increase so each batch contains less SKUs.

Although BOC-ACOLS is better than multi-GA in finding a shorter route, it also induces a longer computation time. Note that major part of computation time is spent on routing. Therefore, an additional test to prove the time efficiency of BOC-ACOLS is presented. In this test, we compare the time that BOC-ACOLS spent on finding a solution which is no worse than that of multi-GA. Three different sizes of nodes are used: 10, 30 and 50 because these three sizes are common problem sizes in the former tests. Firstly, a given size of nodes are selected uniformly from all P SKUs. Secondly, multi-GA is run for 10 times to output 10 final picking routes. Then we choose the best solution and record its distance as GA-$Dist$ with corresponding computation time as GA-$Time$. Thirdly, BOC-ACOLS is run for ten times. The terminal condition for BOC-ACOLS is finding a solution that is no worse than GA-$Dist$. Then computation time of BOC-ACOLS is recorded. The result is shown in Table 8. After this test, we run another similar experiment. In the second test, once CPU time exceeds GA-$Time$ then BOC-ACOLS stops and outputs the current solution. Result of the second test is shown in Table 9.

In Tables 8 and 9, unit to measure time is second and unit to measure distance is meter (m). The results show that BOC-ACOLS is more efficient than multi-GA in considering the time spent on finding the same qualified solution or the quality of the solution obtained in the same computation time. In the first test, BOC-ACOLS spends more time than multi-GA averagely when the number of nodes is large, and otherwise it is faster. In the second test, BOC-ACOLS obtains a shorter average picker routing distance than that of multi-GA in all cases. Though the gap decreases as the problem size increases,

Table 9. Comparison of picker routing distance obtained by BOC-ACOLS and multi-GA in the same time.

Nodes	GA-$Dist$ (m)	ACO-$Dist_{avg}$ (m)	ACO-$Dist_{min}$ (m)	ACO-$Dist_{max}$ (m)
10	142.7	81.2	24.6	158.7
30	389.3	283.8	184.7	480.7
50	486.0	456.7	438.7	764.3



(a) Impact of N on number of batches. (b) Impact of N on picking distance (m).

Figure 4. Sensitivity analysis of N.

BOC-ACOLS still has impressive outcome. In real applications, terminal condition of BOC-ACOLS can be adjusted to cut down the computation time only with a small reduction of solution quality.

Finally, note that the size of the unprocessed order list N has influence on the final picking distance and the order holding time. In other words, larger N brings a shorter picking route but a longer order holding time which may affect the customer satisfaction. A simple sensitivity analysis of N is given to explore how N affect the picking distance. Note that distribution of order arrival time is difficult to estimate, and the order holding time is not considered in this analysis. Results of sensitivity analysis are presented in Figure 4. Order set of normal size and normal demand with 5000 orders is used and the solution is computed by BOC-ACOLS.

In Figure 4, N changes from 30 to 200 to ensure a fairly stable performance. The total picking distance first decreases as N increases and then becomes stable when N reaches 150. When N is relatively small, more batches are generated because utilisation of capacity is not maximised. When N increases, number of batches does not decrease due to the capacity constraint. However, larger N may lead to longer order fulfilment time because the order has to wait for longer time after it is placed by customers. Therefore, in practice, online retailers should make a trade-off between efficiency and customer satisfaction. The sensitivity analysis presented in Figure 4 is based on a typical parameter setting used and results may change as other parameters change such as the capacity of retrieval vehicle. Hence, more discussion about the decision of N will not be pursued in the paper.

6. Conclusion

In this paper, a joint optimisation problem of order batching and picker routing in a warehouse of online retailers is investigated. We purpose an integer formulation aiming to minimise the picking distance which reflects both operating cost and order processing time to some extent. An effective algorithm using similarity coefficient by overlapping channels to batch orders and composing picking route which is further optimised by the ant colony optimisation with local search is presented. A numerical experiment based on actual sales data is given to test the efficiency of heuristics and results indicate that our heuristic can shorten the order picking distance by optimising order batching and picker routing, generating orders into less batches which reduce manpower spent on picking with even less workload for each batch.

In practice, problem size could be very large and is still increasing. Online retailing in China is growing at a high speed and competition is becoming more furious. For example, the biggest online shopping day in China, known as ‘Double 11’ or China’s singles’ day, brought 57.1 billion Yuan (about 9.3 billion dollars) to Alibaba in a single day of 2014. Other online retailers also followed and launched large promotion activities. Therefore, it has been a big challenge to improve efficiency of

warehouse operations. The problem size has become so large that traditional heuristics cannot meet requirements. Intelligent optimisation algorithms are then designed to solve this kind of problems but they usually take an unacceptable long time to obtain a good solution. In this paper, our algorithm can balance quality of solution and computation time. Besides, we consider the feasibility of algorithms by proposing a method to convert practical problems into formulation programs.

The computational example mainly implies that: (1) our algorithm can divide orders into less batches compared to practically used heuristics, which could save labour cost; (2) our algorithm can find a shorter picking route than other intelligent algorithms and practical heuristics, which will reduce order fulfilment time; (3) our algorithm could have higher time efficiency compared to other intelligent algorithms and computation time can be cut down without loss of solution quality; (4) decision-makers have to make a trade-off between cost and order fulfilment time by deciding number of waiting orders in WMS. Our work has been implemented in Yihaodian with positive feedback. For a typical online retailer in China, human labour will be prevailing in the foreseeable future due to low labour cost. Therefore optimisation in warehouse operations to save cost with realisation of such a fact should be a research focus.

With realisation of works in the paper, our research can be extended in the following ways: (1) combine the optimisation of warehouse layout and storage location allocation for a new warehouse; (2) in practical warehouse operations, the channel is usually narrow and congestion must be considered in batching orders; (3) when an effective algorithm is found to compute the picking route, then it demands how to balance the total picking distance among different batches for convenience of personnel performance evaluation.

Acknowledgements

The authors would like to thank editors and referees for their valuable suggestions which can significantly improve our manuscript.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported by the National Natural Science Foundation of China [71131004], [71171008], [71301122], [71571079]; Program for New Century Excellent Talents in University [NCE-13-0228]; Fundamental Research Funds for the Central Universities (Wuhan University).

References

Chen, M. C., and H. P. Wu. 2005. "An Association-based Clustering Approach to Order Batching Considering Customer Demand Patterns." *Omega* 33 (4): 333–343.

Chen, T. L., C. Y. Cheng, Y. Y. Chen, and L. K. Chan. 2015. "An Efficient Hybrid Algorithm for Integrated Order Batching, Sequencing and Routing Problem." *International Journal of Production Economics* 159: 158–167.

Cheng, C. Y., 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: 805–814.

Colorni, A., M. Dorigo, and V. Maniezzo. 1991. "Distributed optimization by ant colonies." Vol. 142. In *Proceedings of the First European Conference on Artificial Life*, 1991, 134–142. Paris, France.

de Koster, R., E. S. van der Pooft, and M. Wolters. 1999. "Efficient Orderbatching Methods in Warehouses." *International Journal of Production Research* 37 (7): 1479–1504.

Dorigm, M., and L. M. Gambardella. 1997. "Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem." *IEEE Transactions on Evolutionary Computation* 1 (1): 53–66.

Dukic, G., T. Opetuk, and T. Lerher. 2015. "A Throughput Model for a Dual-tray Vertical Lift Module with a Human Order-picker." *International Journal of Production Economics* 170: 874–881.

Gademann, N., J. P. van den Berg, and H. H. van der Hoff. 2001. "An Order Batching Algorithm for Wave Picking in a Parallel-aisle Warehouse." *IIE Transactions* 33 (5): 385–398.

Gademann, N., and S. Velde. 2005. "Order Batching to Minimize Total Travel Time in a Parallel-aisle Warehouse." *IIE Transactions* 37 (1): 63–75.

Gharehgozli, A. H., G. Laporte, Y. G. Yu, and R. de Koster. 2014. "Scheduling Twin Yard Cranes in a Container Block." *Transportation Science* 49 (3): 686–705.

Gharehgozli, A. H., Y. G. Yu, R. de Koster, and J. T. Udding. 2014a. "A Decisiontree Stacking Heuristic for Large Scale Reshuffling Problems At a Container Yard." *International Journal of Production Research* 52 (9): 2592–2611.

Gharehgozli, A. H., Y. G. Yu, R. de Koster, and J. T. Udding. 2014b. "An Exact Method for Scheduling a Yard Crane." *European Journal of Operational Research* 235 (2): 431–447.
Gharehgozli, A. H., Y. G. Yu, X. D. Zhang, and R. de Koster. 2014. "Polynomial Time Algorithms to Minimize Total Travel Time in a Two-depot Automated Storage/Retrieval System." *Transportation Science*. doi:10.1287/trsc.2014.0562.
Goetschalckx, M., and H. Donald Ratliff. 1988. "Order Picking in an Aisle." *IIE Transactions* 20 (1): 53–62.
Ho, Y. C., T. S. Su, and Z. B. Shi. 2008. "Order-batching Methods for an Order-picking Warehouse with Two Cross Aisles." *Computers & Industrial Engineering* 55 (2): 321–347.
Kulak, O., Y. Sahin, and M. E. Taner. 2012. "Joint Order Batching and Picker Routing in Single and Multiple-cross-aisle Warehouses using Cluster-based Tabu Search Algorithms." *Flexible Services and Manufacturing Journal* 24 (1): 52–80.
Lambert, D. M., J. R. Stock, L. M. Ellram. 1998. *Fundamentals of Logistics Management*. Boston, MA: Irwin/McGraw-Hill.
Le-Duc, T., and R. de Koster. 2007. "Travel Time Estimation and Order Batching in a 2-block Warehouse." *European Journal of Operational Research* 176 (1): 374–388.
Lerher, T. 2015. "Travel Time Model for Double-deep Shuttle-based Storage and Retrieval Systems." *International Journal of Production Research*. doi:10.1080/00207543.2015.1061717.
Lerher, T., M. Edl, and B. Rosi. 2014. "Energy Efficiency Model for the Mini-load Automated Storage and Retrieval Systems." *The International Journal of Advanced Manufacturing Technology* 70 (1–4): 97–115.
Lerher, T., Y. B. Ekren, Z. Sari, and B. Rosi. 2015. "Simulation Analysis of Shuttle based Storage and Retrieval Systems." *International Journal of Simulation Modelling* 14 (1): 48–59.
Lerher, T., Y. B. Ekren, G. Dukic, and B. Rosi. 2015. "Travel Time Model for Shuttle-based Storage and Retrieval Systems." *The International Journal of Advanced Manufacturing Technology* 78 (9–12): 1705–1725.
Lerher, T., M. Šraml, and I. Potrč. 2011. "Simulation Analysis of Mini-load Multi-shuttle Automated Storage and Retrieval Systems." *The International Journal of Advanced Manufacturing Technology* 54 (1–4): 337–348.
Petersen, C. G. 2000. "An Evaluation of Order Picking Policies for Mail Order Companies." *Production and Operations Management* 9 (4): 319–335.
Roodbergen, K. J., and R. de Koster. 2001a. "Routing Order Pickers in a Warehouse with a Middle Aisle." *European Journal of Operational Research* 133 (1): 32–43.
Roodbergen, K. J., and R. de Koster. 2001b. "Routing Methods for Warehouses with Multiple Cross Aisles." *International Journal of Production Research* 39 (9): 1865–1883.
Roodbergen, K. J., and I. F. A. Vis. 2009. "A Survey of Literature on Automated Storage and Retrieval Systems." *European Journal of Operational Research* 194 (2): 343–362.
Rouwenhorst, B., B. Reuter, V. Stockrahm, G. J. van Houtum, R. J. Mantel, and W. H. M. Zijm. 2000. "Warehouse Design and Control: Framework and Literature Review." *European Journal of Operational Research* 122 (3): 515–533.
Rosi, B., L. Grasic, G. Dukic, T. Opetuk, and T. Lerher. 2016. "Simulation-based Performance Analysis of Automated Single-tray Vertical Lift Module." *International Journal of Simulation Modelling* 15 (1): 97–108.
Tang, L. X., and G. S. Wang. 2008. "Decision Support System for the Batching Problems of Steelmaking and Continuous-casting Production." *Omega* 36 (6): 976–991.
Tsai, C. Y., J. J. H. Liou, and T. M. Huang. 2008. "Using a Multiple-GA Method to Solve the Batch Picking Problem: Considering Travel Distance and Order Due Time." *International Journal of Production Research* 46 (22): 6533–6555.
Vasili, M. R., S. H. Tang, N. Ismail, and S. Sulaiman. 2008. "Class-based Storage Assignments for Miniload AS/RS with Open-rack Structure." *International Journal of Engineering and Technology* 5 (2): 118–128.
Vaughan, T. S. 1999. "The Effect of Warehouse Cross Aisles on Order Picking Efficiency." *International Journal of Production Research* 37 (4): 881–897.
Wang, X. P., and L. X. Tang. 2008. "Integration of Batching and Scheduling for Hot Rolling Production in the Steel Industry." *The International Journal of Advanced Manufacturing Technology* 36 (5–6): 431–441.
Won, J., and S. Olafsson. 2005. "Joint Order Batching and Order Picking in Warehouse Operations." *International Journal of Production Research* 43 (7): 1427–1442.
Yu, Y. G., and R. de Koster. 2012. "Sequencing Heuristics for Storing and Retrieving Unit Loads in 3D Compact Automated Warehousing Systems." *IIE Transactions* 44 (2): 69–87.
Zhang, Z. Z., O. Che, B. Cheang, A. Lim, and H. Qin. 2013. "A Memetic Algorithm for the Multiperiod Vehicle Routing Problem with Profit." *European Journal of Operational Research* 229 (3): 573–584.