Proceedings of the 2005 International Conference on Simulation and Modeling
V. Kachitvichyanukul, U. Purintrapiban, P. Utayopas, eds.

FORMING BATCHES OF CUSTOMER ORDERS IN A WAREHOUSE

Nakorn Indra-Payoong

Transport and Logistics Optimisation Group
Maritime College
Burapha University
Chonburi, Thailand 20131

Kasame Pinthong

Water Resources and Coastal Engineering Section
281 Sukhumvit 71
SEATEC Group, Co., Ltd.
Bangkok, Thailand 10110

ABSTRACT

This paper addresses the problem of forming batches of customer orders in a warehouse. The problem is modelled as a constraint satisfaction problem in which the business goal and operational requirements are presented as soft and hard constraints respectively. A set of binary decision variables is used of whether to assign an order or not to each potential batch. The very high number of possible order combinations led us to use a heuristic method that can find a good batch forming solution within a viable time. The method also incorporates a self-learning mechanism. With sufficient search history, the algorithm predicts and generates a good assignment of orders to batches. Computational tests on randomly generated problems are conducted to evaluate the proposed model and solution method.

1 INTRODUCTION

Grouping members or items into batches is a decision problem occurred in business and industry, e.g. grouping patients into hospital's rooms or even a selection of food for menu. The objective of batch forming problem is to group items into batches such that items are formed with maximum similarity.

In warehouse operations, when customer orders consist of few items, efficient order picking can be achieved by forming (or grouping) orders into batches. When the size of a warehouse and the number of customer orders are large, a warehouse company encounters the problem in finding an efficient order picking operation. The poor operation incurs additional operating cost, i.e. wages paid to order pickers and the maintenance cost for pick equipments used.

Typically, order picking process accounts for about 55% of warehouse operating cost (Frazelle, 1996). Therefore, a decision support tool that provides an effective order picking plan would bring substantial cost-savings. We address the problem of forming batches in a grocery warehouse in which thousand customer orders a day are obtained. Each time when a warehouse receives customer orders, order pickers retrieve items listed in the orders; afterwards, those items will be delivered to warehouse's customers.

A basic concept for grouping customer orders into batches exploits the similarity between orders and combines them together. There is a cost paid for forming batches if orders are combined with less similarity. In this paper, a picker route is expressed in terms of aisle to traverse by the pickers. The business goal is to assign orders sequentially to batches of pick lists such that the picker route interference is minimised; in other words, batches of orders are formed with maximum similarity. A set of binary decision variables is used of whether to assign and order or not to each potential batch. We introduce a binary matrix of aisles to be traversed by all customer orders in which a traversal routing policy is assumed, i.e. when entering an aisle, an order picker completely traverses the aisle, and U-turn is not allowed. Then, the mismatches of aisles for the combinations of orders are used to evaluate how well the orders are formed.

The paper is organised as follows. Section 2 reviews the existing methods for batch forming problem. Section 3 defines soft and hard constraints and models the problem as a constraint satisfaction problem. Section 4 describes the solution methods for the batch forming problem. Computational results are given in Section 5. Conclusions are finally, discussed in the last section.

2 LITERATURE REVIEW

The batch forming problem is the problem of assigning customer orders into batches to enable efficient order picking and of finding batches of picking route for every batch. There are several variants of this problem, depending on the method to pick items listed in the orders, e.g. manual, semi-automatic or automatic picking, the capacity of a pick equipment, and the business goal of the problem.

In general, the business goal can be categorised into two groups: the distant of pickers traversed and the similarity of orders in batch (Choe, 1991) . The first category aims to minimise the total distance (or travel time) of pickers traversed in retrieving orders. The second tries to maximise the similarity of orders in batch, in other words, to minimise the total number of mismatches amongst customer orders.

An early attempt in forming batches of customer orders in a warehouse was proposed by Elsayed and Stern (1983), the objective of the problem is to minimise the total distance travelled by order pickers. They combined the orders in terms of common item locations and minimum distance between item locations. The heuristic method was proposed to solve this problem. The heuristic composes of three steps: seed selection, order congruence, and addition of orders.

Elsayed and Unal (1989) proposed the order batching algorithm for automated storage and retrieval (S/R) systems based on timesaving criterion of forming batches. Several heuristics were used, i.e. EQUAL, SL, MAXSAV and Cwright. They developed a travel time estimate procedure to calculate the time saved in forming batches. Gibson and Sharp (1992) developed the order batching heuristic for a manual order picking warehouse. They assumed that the location of the items in terms of aisle is known. To form a batch, an order in the top of the list is chosen as a seed order. Then, the orders are successively chosen in the batch based on the sequential minimum distance of the orders until the number of orders reaches the batch size limit.

Elsayed et al. (1993) considered time window batching problem in a warehouse. They proposed the heuristic for batching of orders with due times. The business goal is to minimise earliness and tardiness penalties. Gademann et al. (2001) addressed the problem of batching order in a warehouse. The objective is to minimise the minimum lead time of any of the batches. They presented a branch-and-bound algorithm to solve the problem of moderate size. The 2-opt heuristic is used for providing tight upper bounds.

Ruben and Jacobs (1999) compared the performance of different order batching strategies. They demonstrated that the turnover based assignment strategy reduces the distance traversed by order pickers in the warehouse, but increases traffic in the aisle. Random assignment of items results an increase in travel time of the pickers, however it does not cause the congestion. Family based assignment strategy provides the best results by reducing the travel distance of the pickers and has the aisle traffic in control. Sundaram and Centeno (2004) considered batch forming problem in a warehouse. The objective is to form batches of orders so that the commonality of aisles amongst the orders that form the batch is maximised. The exhaustive and partitioning methods were proposed to solve the problem. Under the exhaustive method, all possible combinations of orders for each batch are generated. After generating order combinations, a ranking method is used to determine the best combination. For the large size problem, the partitioning heuristic is used to obtain the good combination within a little computational time.

In this paper we consider the problem of forming batches for customer orders in a warehouse in which thousand customer orders a day are received. The objective is to minimise the total number of mismatches amongst orders in terms of aisle travelled by pickers. We present a heuristic learning algorithm to find a good batch forming solution for a practical order picking operation.

3 CONSTRAINT-BASED MODELLING

Real-world problems tend to have a large number of constraints, which may be hard or soft. Hard constraints require that any solutions will never violate the constraints. Soft constraints are more flexible, constraint violation is tolerated but attracts a penalty. Naturally, a real-world problem can be thought of as a constraint satisfaction problem (CSP). There are two critical advantages of using constraint-based modelling. Firstly, it is a clean separation between problem modelling and solution technique. If new problem conditions are introduced, we only need to model such conditions as constraints. Secondly, problem-specific knowledge can influence the search naturally. This is done by applying problem-specific weights, reflecting their relative importance, directly to constraints in order to enhance a solution algorithm within a CSP framework (Indra-Payoong et al., 2003).

We consider the order batching for a warehouse with parallel aisles which are connected by cross-aisles at the front, (the middle), and the back of each aisle. Figure 1 illustrates a typical parallel-aisle warehouse.

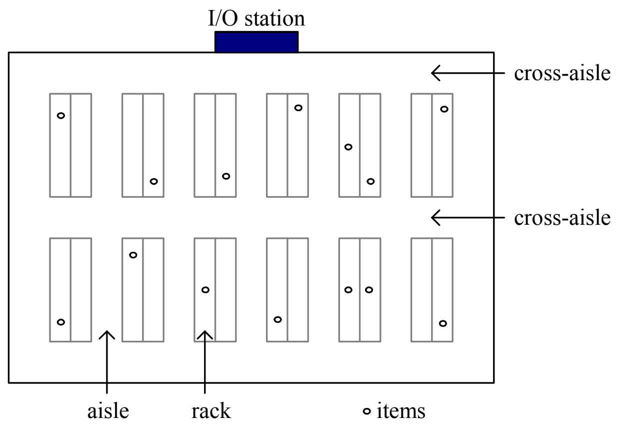

Figure 1: A typical parallel-aisle warehouse

A binary decision variables is used of whether to assign an order or not to each potential batch. To simplify our discussion, the following notation will be used.

Subscribes:

- i : batch, $i = 1, 2, 3, \ldots, M$
- j : order, $j = 1, 2, 3, \ldots, N$
- k : aisle, $k = 1, 2, 3, \ldots, L$

Decision variable:

- x_{ij} : 1, if batch i consists of order j, 0 otherwise

Parameters:

- b_i : batch i
- B : Size of batch (number of orders)
- m_i : number of mismatches in batch i

In a CSP-model, optimisation criterion and operational requirements are represented as soft and hard constraints respectively. The criteria are handled by transforming them into soft constraints. This is achieved by expressing each criterion as an inequality against a tight bound on its optimal value. As a result, such soft constraints are rarely satisfied.

A feasible solution for a CSP representation of the problem is an assignment to all decision variables in the model that satisfies all hard constraints, whereas an optimal solution is a feasible solution with the minimum total soft constraint violation. For a constraint satisfaction problem, the violation v_i of constraint i is defined as follows:

$$\sum_j a_{ij} x_j \le b_i \Rightarrow v_i = \max\left(0, \sum_j a_{ij} x_j - b_i\right) \qquad (1)$$

where: a_{ij} are coefficients, b_i is a tight bound and x_j are constrained variables. Note that violations for other types of linear and non-linear constraints can be defined in an analogous way.

When all variables are assigned a value, the violation of the hard and soft constraints can be tested and quantified for evaluating solutions.

3.1 Soft constraint

The total number of mismatches – the aim is to minimise the total number of mismatches for all customer orders.

In this paper, a picker route is expressed in terms of aisle to traverse as a picker moves along aisle to pick items from rack. We introduce a binary matrix of aisles to be traversed by all customer orders in which a traversal routing policy is assumed, i.e. when entering an aisle, a order picker completely traverses the aisle, and U-turn is not allowed. The binary matrix is illustrated in Table 1.

Table 1: The binary matrix

Order/Aisle	1	2	3	4	5	6
1	1	1	1	0	1	0
2	0	1	1	1	0	1
3	1	0	1	1	1	1
4	0	1	0	1	0	1
5	1	0	0	1	1	0
6	0	0	1	1	1	1

In Table 1, suppose that order 1 and order 2 are formed batch i (x_{i1}, $x_{i2} = 1$), the number of mismatches of aisles to be traversed by these orders (m_i) is equal to 4. That is, in aisle 1, 4, 5, and 6, the item locations are different. Therefore, the total number of mismatches constraint can be defined as:

$$\sum_{i=1}^{M} m_i = 0 \qquad (2)$$

3.2 Hard constraint

Batch size – this constraint ensures the orders must not exceed the batch size, which is defined as:

$$\sum_{j=1}^{N} x_{ij} \le B; \quad \forall i \qquad (3)$$

4 SOLUTION METHOD

We propose a constraint-based local search (CLS) for solving the batch forming problem.

4.1 CLS

CLS is a heuristic method proposed by Indra-Payoong et al (2004). The CLS can be considered as a general algorithmic framework which can be applied to different optimisation problems with relatively few modifications to make them adapted to specific conditions or problems.

The CLS starts with an initial random assignment, in which some hard constraints in the model can be violated. In the iteration loop, the algorithm randomly selects a violated constraint. Having selected a violated constraint, the algorithm randomly selects one variable in that constraint and another variable, either from the violated constraint or from the search space. Then, two flip trials are performed, i.e. changing the current value of the variable to its complementary binary value. Whenever all hard constraints are satisfied, the algorithm stores the soft violation penalties as feasible objective values, together with the associated variable values. The algorithm continues until stopping criterion is met, i.e. a feasible solution is found or if no improvement has been achieved within a specified number of iterations. The procedure of CLS is outlined in Figure 2.

```
proc CLS
    input soft and hard constraints
    $A$ := initial assignment
    while not stopping criterion do
        $C$ := select-violated-hard-constraint $(A)$
        $P$ := select-two-variables $(C, A)$
        $h$ := total hard violation
        $A_1$, $A_2$ := flip$(A, P)$
        if $(h_1 < h_2)$ then $(A \leftarrow A_1)$
        else $(A \leftarrow A_2)$
        if $h = 0$ then $A$ is feasible, record $A$
        end if
    end while
    output a feasible solution found
end proc
```

Figure2: The CLS procedure

The procedure can be readily modified to give a set of feasible solutions and to make more use of the soft constraints, which in the procedure of Figure 2 are ignored.

4.2 Variable selection

Once a violated constraint has been chosen, the algorithm randomly selects two variables in order to perform trial flips. For the batch forming problem, there is one set of binary variables, i.e. a variable x_{ij} represents whether order j is assigned to batch i or not. However, randomly choosing one variable from the violated constraint and another from the search space, the diversified exploration of the solution space may not be achieved because the number of assigned orders to batches (x_{ij} = 1) is significantly less than the number of unassigned orders to batches (x_{ij} = 0).

It is also noted that a violated constraint indicates the number of orders that exceeds the batch size, and CLS attempts to get ride of one assigned order (x_{ij} = 1) at each time in order to reduce the number of orders in the violated batch. However, when all batch size constraints are almost satisfied, we may only need a small change to a current assignment so that CLS can move around a feasible order combination more closely. Therefore, for the batch forming problem, two alternative variable selection schemes are introduced as follows:

Scheme 1:	Randomly select two assigned orders in the violated batch.
Scheme 2:	Randomly select any two orders in the violated batch.

The CLS selects one of the schemes at random so that a wide exploration of the solution space may be achieved. The scheme 2 is used not only to allow CBS moving around a feasible solution more closely but also to gain computational advantage in a performance of the algorithm.

4.3 Refined improvement

CLS has no feature that can move from a current feasible solution to a better feasible solution. This deceases the performance of CLS for the batch forming problem in which there are so many feasible order combinations to batches, and when a good bound strategy on an objective value cannot be obtained. Therefore, a refined-improvement procedure is introduced to improve a feasible orders combination found by CLS. The main concept is to allow only feasible improving solutions and to search more intensively on a feasible region currently found. When CLS finds a feasible combination, the refined-improvement procedure is called. The procedure of the refined improvement is outlined as follows:

Step 1: Given a feasible combination found by CLS, record the total mismatches.

Step 2: Perform interchange assignment, i.e. assign order to a new batch, and swap each remaining order in the new batch to the current assigned batch until all remaining orders in the new batch have been considered. This is illustrated in Figure 3.

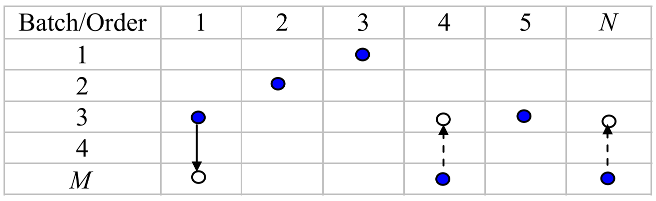

Figure 3: The interchange assignment

Step 3: If any interchange assignment is feasible and a new total mismatches is less than a current total mismatches, replace a current total mismatches with a new total mismatches; otherwise, undo interchange assignment.

Step 4: Repeat from step 2 until all orders have been considered and continue the refined improvement procedure until no improvement has been found.

4.4 Learning method

The learning method learns from the search history and extracts the problem-specific knowledge implicitly (Indra-payoong et al 2003). After a specified number of iterations, the search history is analysed. The predictive choice model (PCM) predicts good assignments of orders to batches. These assignments will be hold for a number of iterations in a probabilistic way, leading to good order combinations.

4.4.1 Violation history

Once a variable has been selected, the CLS has to choose a value for it. The concept is to choose a good value for a variable, e.g. the one that is likely to lead to a smaller total hard constraint violation in a complete assignment. In CLS, two variables are considered at each flip trial. The first variable is randomly chosen from those appearing in a violated constraint and considered as the variable of interest, the second variable is randomly selected, either from that violated constraint or from the search space, and is to provide a basis for comparison with the variable of interest.

Clearly, the interdependency of the variables implies that the effect of the variable value chosen for any particular variable in isolation is uncertain. Flipping the first variable might result in a reduction in total hard constraint violation. However, it might be that flipping the second variable would result in even more reduction in the violation. In this case, the flipped value of the first variable is not accepted.

Table 2 and 3 illustrate how the violation history is recorded and compared when the variable of interest in a current assignment, $x_{ij} = 0$ and $x_{ij} = 1$ respectively. In these tables, h is the total constraint violation, x_{ij}^* is the value of x_{ij} chosen in the flip trial. Note that only h_1, h_1', and x_{ij}^* are recorded for the violation history of x_{ij}.

Table 2: Violation history, $x_{ij} = 0$

Flip trial	Var. of interest x_{ij}				Compared variable $x_{ij}(q)$					x_{ij}^*
	Current		Flipped		q	Current		Flipped		
	Val	h_1	Val	h_1'		Val	h_2	Val	h_2'	
1	0	33	1	15	5	1	33	0	26	1
2	0	18	1	15	9	0	18	1	9	0
3	0	9	1	4	12	0	9	1	14	1
N	0	75	1	32	4	1	75	0	39	1

In flip trial 1 the selected variables are x_{ij} (current value 0) and, separately, $x_{ij}(5)$ (current value 1). The current assignment has violation = 33. Flipping x_{ij}, with $x_{ij}(5)$ fixed at 1, gives violation = 15; flipping $x_{ij}(5)$, with x_{ij} fixed at 0, gives violation = 26. Therefore, in this trial the CLS records x_{ij} = 1 as the better value. At some later iteration CLS chooses to flip x_{ij} again, this time (flip trial 2) with compared variable $x_{ij}(9)$. Flipping x_{ij}, with $x_{ij}(9)$ fixed at 0, gives violation = 15; flipping $x_{ij}(9)$, with x_{ij} fixed at 0 gives violation = 9. Even though flipping x_{ij} to 1 gives a better violation than the current assignment, in this flip trial the algorithm records $x_{ij} = 0$ as the better value as there is an assignment with $x_{ij} = 0$ which gives an even better violation. If we view the results of these flip trials as a random sample of the set of all assignments, the probabilistic model could be used to capture the behaviourial inconsistency in choice selection and to predict what would be a good value for x_{ij}. The collection method of violation history in Table 3 can be explained in an analogous way.

Table 3: Violation history, $x_{ij} = 1$

Flip trial	Var. of interest x_{ij}				Compared variable $x_{ij}(q)$					x_{ij}^*
	Current		Flipped		q	Current		Flipped		
	Val	h_1	Val	h_1'		Val	h_2	Val	h_2'	
1	0	33	1	15	5	1	33	0	26	1
2	0	18	1	15	9	0	18	1	9	0
3	0	9	1	4	12	0	9	1	14	1
N	0	75	1	32	4	1	75	0	39	1

4.4.2 Variable fixing

After a specified number of iterations, the trial history is analysed. Some variables may have high probability of a particular value given by the PCM. These variables will be fixed at their predicted value for a number of iterations determined by the magnitude of the probability. The search space would be intensified and the algorithm targets for an optimal solution. The x_{ij} may hold a current value 0 or 1 and its predicted value can either be 0 or 1 during the search. We categorise the fixing procedure into two groups: local fix and global fix.

Local fix. The local fix is a process of preventing the CLS assigning order to batch that may not lead to an optimal solution, i.e. fixing x_{ij} = 0 for the number of iterations.

Global fix. The global fix provides a strong propagation of consistency within each order for the potential number of batches. When the global fix is called (i.e. a predicted value of x_{ij} = 1), one order is exactly assigned to a batch, thereby it prevents the algorithm selecting the remaining potential batches for that order.

5 COMPUTATIONAL RESULTS

The batch forming model was tested on randomly generated problem data from small to very large sizes. All test cases were run on a typical personal computer. Each test case is run ten times using different random number seeds at the beginning of each run. If no improvement has been achieved within 1500 iterations, the CLS will terminate. The results are summarised in Table 4 . In this table, orders are either be formed in each interval time or each delivery day, $\Phi_{\max}$ denotes the highest value of the first feasible solutions, and Φ denotes the mean of the best feasible solutions.

Table 4: Computational results

Test	Aisles	Orders	$\Phi_{\max}$	Solutions		Reductions	Time
case				Φ	Batches	(%)	(min)
1	6	10	20	8	5	60	0.00
2	10	20	64	40	5	38	0.01
3	10	50	193	151	5	22	0.05
4	15	100	579	491	10	15	0.21
5	15	250	1345	1069	25	21	3.65
6	20	500	3604	2918	50	19	9.04
7	30	1000	7401	6288	100	15	17.30

The results indicate that CLS is a promising approach for the batch forming problem. We are able to obtain solutions of a good quality in a reasonable run-time.

Figure 4 illustrates the improvement of the solution from aspects of the total number of mismatches versus the number of iterations for test case 4. It shows that as search proceeds CLS tends to roam progressively further from the best current solution and focuses intensively on good solution regions previously found. Although the actual values may differ amongst various cases, the characteristic shapes of the curves are similar.

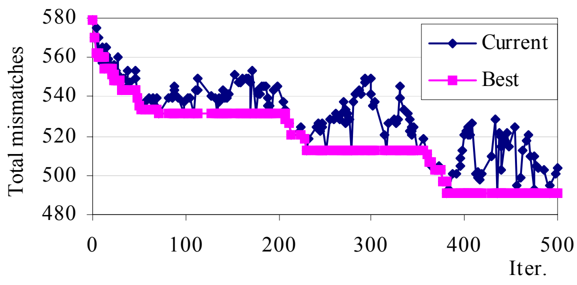

Figure 4: The solution improvement of CLS

6 CONCLUSIONS

This paper considers the difficult optimisation problem of forming batches of customer orders in a warehouse. The problem is modelled as a constraint satisfaction problem and the total number of mismatches amongst orders in terms of aisles traversed by pickers is used to evaluated how well the orders are formed. This quantified measure can readily be applied to a wide range of the problem conditions in which the size of a warehouse or the layout of aisles may vary.

The proposed solution framework is simple and convenient to use. It requires a little effort in designing the algorithm to solve the problem or to fit new problem conditions. This is particularly true for the batch forming problem in which the problem-specific knowledge cannot be obtained easily; as a result, traditional heuristic methods are less attractive as an algorithm designer faces the problem in finding a good short-cut to a good or optimal solution. The computational results have demonstrated the efficiency and the robustness of the proposed method for the batch forming problem.

REFERENCES

Choe, K.I. 1991. Aisle-based order pick systems with batching, zoning and sorting. PhD thesis, Georia Instiue of Technology, GA.

Elsayed, E.A. and M.K. Lee, S. Kim. and E. Scherer. 1993. Sequencing and batching procedures for minimizing earliness and tardiness penalty of order retrievals. *International Journal of Production Research* 31(3): 272-738.

Elsayed, E.A. and O.I. Unal. 1989. Order batching algorithms and travel time estimation for automated storage/retrieval systems. *International Journal of Production Research* 27 (7): 1097-1114.

Elsayed, E.A. and R.G. Stern. 1983. Computerized algorithms for order processing in automated warehouse systems. *International Journal of Production Research* 21 (4): 579-586.

Frazelle, E.H. 1996. World class warehousing. Logistics Resources International, Atlanta, GA.

Gibson, D.R. and G.P. Sharp. 1992. Order batching procedures. *European Journal of Operational Research* 58 (1): 57-67.

Indra-Payoong, N., R.S.K. Kwan, and L.G. Proll. 2003. A randomised algorithm with prediction. (Invited talk at) Scheduling Workshop on Application of Constraint Programming, 9-10 September, University of Huddersfield.

Indra-Payoong, N., R.S.K. Kwan, and L.G. Proll. 2004. Rail container service planning: a constraint-based approach. (To appear in) *Journal of Scheduling*.

Ruben, R.A. and F.R. Jacobs. 1999. Batch construction heuristics and storage assignment strategies for wall/ride and pick systems. *Management Science* 45 (4): 577-596.

Sundaram R. and M.A. Centeno. 2004. Forming batches: exhaustive vs. partitioning methods. In *Proceedings of the 2nd World Conference on POM*, Cancun, Mexico.

Gademann, A.J.R.M., 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: 385-398.

AUTHOR BIOGRAPHIES

NAKORN INDRA-PAYOONG is a Lecturer in the Maritime College, Burapha University. His research interests include quantitative and optimisation techniques for providing competitive advantages in business with particular emphasis on shipping, freight transport and logistics. His email address is <nakorn@buu.ac.th>.

KASAME PINTHONG is a senior engineer of SEATEC Group, Co., Ltd.. He received his M.Eng in Water Resources Engineering from King Mongkut University of Thonburi, Thailand. His interests include advanced algorithms, simulation and modelling techniques with focus on practical applications in water resources and coastal management. His email address is <kasemai@hotmail.com>.