Applying Simulated Annealing Algorithm for Cross-Docking Scheduling

Alireza Boloori Arabani, Faraz Ramtin, S. Nima Rafienejad,

Abstract— **Cross docking is a novel approach in material handling which can reduce inventories, lead times, and customer response time. In this strategy, products are unloaded from inbound trucks, sorted and categorized based on their characteristics, moved and loaded onto outbound trucks for delivery to demand points in distribution network. In this paper we address simulated annealing algorithm (SA) to find the best sequence of inbound and outbound trucks, so that the objective, minimizing the make span, can be satisfied. Furthermore, the efficiency of applied parameters is evaluated by Taguchi plan of parameter settings.**

Index Terms—**Cross-docking, inbound trucks, outbound trucks, receiving, shipping.**

I. Introduction

In the cross docking system a fewer stock is handled in a temporary storage in comparison with ordinary warehouses. As a result, not only total operational costs are reduced by cutting unnecessary inventory in the distribution system, but the customer satisfaction can be improved by more convenient shipment.

If we consider five main steps for a warehousing center consisting of receiving, sorting, storing, retrieving and shipping, then the most important role of cross-docking systems will be that they put aside storing and retrieving that ends in reduction of costs. For implementing a cross-docking system successfully, we should consider some points as follows: (1) short delivery lead times; (2) the required information of arrival and departure schedules should be obtainable in advance; (3) reducing the amount of delivery time by synchronizing the homogeneous products; (4) having good bases for both software and hardware infrastructure.

In this study, a particular type of cross docking system is probed in which the scheduled inbound trucks come into the receiving dock and the products are unloaded from them and loaded on the scheduled outbound trucks in the shipping dock. The objective is to pass the products from the inbound trucks to the outbound trucks in as least as possible time, so that the make span can be reduced. The general characteristics of this system belong to [1].

The rest of this paper contains the following sections. Section 2 mentions the related literatures briefly. Section 3 points out the main procedures to solve the cross docking problem. The applied simulated annealing algorithm is explained thoroughly in section 4. The Taguchi plan and parameter setting is clarified in section 5. Section 6 deals with the computational results and implementation of the algorithm, and finally Section 7 is the conclusion.

The authors are with the department of Industrial Engineering, Amirkabir University of Technology, 424 Hafez Avenue, Tehran, Iran (Alireza Boloori Arabani is the corresponding author, phone: +98-21-77633696, +98-912-5498037; e-mail: alirezaboloori@aut.ac.ir, alirezaboloori@yahoo.com).

II. Literature Review

As the publications allocated to the general concept of cross docking we can refer to some instances. Waller *et al*. [2] consider level of inventory in the retailer's warehouses in order to analyze the role of cross-docking determining this level. They even address a model for identifying the benefits of a cross docking system, such as lowering inventory level, and satisfying customer services. Ross and Jayaraman [3] proposed two new heuristics to determine the location of a distribution center or cross docking center in a supply chain. In this study, they not only applied the location planning for setting the place of cross docking distribution center in a network design, but also the network design was measured by means of some parameter setting approaches. Lee *et al*. [4] considered the cross-docking system as an applicable concept by which reduction of inventory and increase of customer service and responsiveness can be obtained. Since they aim to implement the cross-docking system operationally, a mixed model for scheduling of both cross-docking and vehicle-routing was addressed in their paper. In another research and in order to clarify the location of a temporary storage, Vis and Roodbergen [5] analyzed a model (with the objective of minimizing the total distance that is paced by forklift trucks conveying products from the inbound trucks to the storage or from storage to outbound trucks) in which they primarily focus on the location of the storage. Another significant measure taken is that they consider this model as a minimum cost flow problem which is quite common in networks problems categories.

On the other hand, there are a few papers worked on the field of scheduling. Magableh *et al*. [6] probed a dynamic distribution and logistics network in which cross-docking systems are exerted for shipping their products from

suppliers to customers. In this case, a simulation model is presented evaluate the amount of risk involved in the transportation and handling of the products. Chen and Lee [7] considered a cross docking system in that the applied operations are alike flow shop environment with two machines in which a job at the first machine must be finished at first, and then the job is allowed to be processed on the second machine. They also came up with a polynomial approximation algorithm and presented a branch and bound algorithm to solve the problem. Miao [8] analyzed an assignment problem with a truck dock in which the number of trucks and number of docks are unequal. The objective of their work is to minimize two measures simultaneously: shipping cost and the number of shipments remained unfulfilled. They also consider their problem affected by three main factors; arrival and departure times of each truck, shipping time of containers through the docks, and the total obtainable capacity of cross-dock. McWilliams [9] considered a cross docking environment in which the delivered packages act as the main products conveyed via a centralized terminal. The main specification of this terminal is that the process of sorting the parcels may require the queuing networks. For dealing with such network, two different algorithms, simulated annealing and iterative improvement, were developed to compare the performance of the delivery schedules of products.

As other papers addressing the novel issue of cross docking systems, we can refer to Alvarez-Perez *et al*. [10], Bartholdi and Gue [11], Boysen [12], Boysen *et al*. [13], Bozer and Carlo [14], Wang and Regan [15].

III. HEURISTIC APPRAOCH FOR OBTAINING OBJECTIVE FUNCTION

According to what mentioned before, this work follows up the main guidelines and frameworks of Yu and Egbelu [1]. In order to calculate the objective function they developed a heuristic algorithm consisting of two general steps in which the first step is based on two approaches conveying products from the inbound truck to the outbound truck: (1) these products can be conveyed from the inbound truck to the outbound truck straightly without being kept in the temporary storage or (2) these products can be stored in the temporary storage – after they are unloaded from the inbound trucks – and then loaded onto outbound trucks. With a brief look at this procedure, it is quite apparent that the more commodities carried through the first way, or similarly the fewer commodities passed through the second way, the more suitable the makespan is acquired. These two approaches are carried and the optimal unscheduled inbound truck can be determined based on one of the following three strategies:

- The smallest number of products shipped to the temporary storage determines the best inbound truck.
- The largest number of products shipped to the outbound truck directly determines the best inbound truck.
- The smallest ratio of the number of products conveyed from an inbound truck into the temporary storage to the number of products conveyed directly from an inbound truck into the outbound truck determines the best inbound truck.

In the second step, the optimal sequences, or permutations of inbound and outbound trucks should be identified. In other words, in this step each scheduled outbound truck must recognize its relevant sequence of inbound trucks. We also have three strategies in this phase as:

- The smallest number of products shipped to temporary storage clarifies the related outbound truck and its associate inbound trucks.
- The shortest time by which the outbound truck dwells in the shipping dock clarifies the related outbound truck and its associate inbound trucks.
- The smallest ratio of the number of products shipped to temporary storage to the number of products required for the outbound truck clarifies the concerned outbound truck and its associate inbound trucks as the best ones.

Now we put our emphasis on calculating the makespan via the method described by Yu and Egbelu [1] and Yu [16]. For this sake, we first introduce the required notations as follows:

M total operation time (makespan)

L total lateness

R number of inbound trucks in the set

S number of outbound trucks in the set

N number of product types in the set

N_T total number of products

D truck changeover time

V moving time of products from the receiving dock to the shipping dock

$r_{[i]k}$ number of units of product type k that was initially loaded in the i^{th} positioned receiving truck in the receiving truck sequence

$s_{[j]k}$ number of units of product type k that was initially needed in the j^{th} positioned receiving truck in the receiving truck sequence

$t_{[i][j]}$ total number of products type k which transfers the i^{th} positioned receiving truck in the receiving truck sequence to the j^{th} positioned receiving truck in the receiving truck sequence

$$v_{[i][j]} = \begin{cases} 1 & \text{If } t_{[i][j]} > 0 \\ 0 & \text{else} \end{cases}$$

$C_{[i]}$ time at which the i^{th} positioned receiving truck in the receiving truck sequence leaves the receiving dock (completion time for inbound trucks)

$C_{[j]}$ time at which the j^{th} positioned shipping truck in the shipping truck sequence leaves the shipping dock (completion time for outbound trucks)

Then the makespan is measure by:

$$M = C_{[S]} \tag{1}$$

in that

$$C_{[i]} = \sum_{k=1}^{N} r_{[i]k} \quad (i = 1) \tag{2}$$

$$C_{[i]} = C_{[i-1]} + D + \sum_{k=1}^{N} r_{[i]k} \quad (2 < i < R) \tag{3}$$

$$C_{[j]} = \max(h_1, h_2) \tag{4}$$

where

$$h_1 = \max_{1 \le i \le R} \left\{ v_{[i][j]} \left(C_{[i]} - \sum_{k=1}^{N} r_{[i]k} + \sum_{k=1}^{N} t_{[i][j]} + V \right) \right\} \tag{5}$$

$$h_2 = C_{[j-1]} + D + \sum_{k=1}^{N} s_{[j]k} \tag{6}$$

IV. Applied Simulated Annealing

SA is a local search meta-heuristic that is derived from the annealing process of solids, or crystals. Actually, in this process a solid material is heated in first and then cooled gradually so that it can reach to its possible lattice framework representing the lattice minimum energy state in which the solid's imperfections are as fewest as possible. Due to this fact and for solving the combinatorial optimization problems, the main procedure in the SA resembles this cooling behavior in order to prepare a novel strategy. The core of the SA is laid in a way allowing escaping from local optima in order to find the possibly best global solution, even if the new created solution aggravates the fitness function (objective function's value).

The main steps of implementing our applied SA algorithm are as follows: (1) it starts from an initial (current) solution (sequence) s created randomly; (2) then at each iteration a neighborhood of the current solution $N(s)$ is generated according to a local search scheme that will be described later; (3) next, the fitness functions (equal to makespan) of each solution in the new neighborhood is measure and the best solution is selected among them called $s' \in N(s)$; (4) if s' has a better fitness function in comparison with the s, then we select the s' as the current solution; otherwise the s' will be selected as the current solution if $r \le p$. The related probability p is acquired as:

$$p = \exp\left(\frac{f(s) - f(s')}{T} \right) \tag{7}$$

furthermore, r is a number having uniform distribution and randomly generated in $[0,1]$; $f(s)$ and $f(s')$ are the fitness functions of the s and s' respectively and T is a term called temperature. After the stopping criterion (maximum number of iterations) is met in the current temperature, we apply the cooling schedule (will be explained later) of the SA by which the temperature should be decreased (the counter of temperatures is called *epoch*). Then, the mentioned procedure is repeated in the new temperature until we can conduct the SA algorithm for all epochs. By reducing the temperatures step by step basically expect to reach to more favorable solutions and are likely to not stick in the local optima; (5) the *diversification scheme* which prevents the search procedure from being trapped in local optimal solutions.

A. Local Search Scheme

In the case of local search we generally have four different schemes illustrated in Fig. 1 given by:

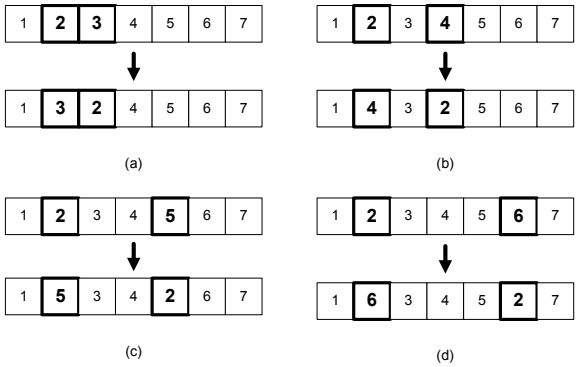

Fig. 1. Local search schemes: (a) adjacent intervals; (b) with one interval interchange; (c) with two intervals interchange; (c) with three intervals interchange

For each of local search schemes, we use a combination of the so called schemes. In other words, for the first scheme, we use the (a) section of Fig. 1. For the second scheme we apply sections (a) and (b) altogether. For the third scheme section (a), (b) and (c) are exerted and for the fourth scheme we use all the schemes altogether.

B. Cooling Schedules

Now, we go through the cooling schedule by which we can determine the temperature in each epoch trough one of the strategies from equations 8 to 11 which are equal to sections (a) to (d) in Fig. 2 in which the behaviors of these strategies can be seen. In these states the ways of decreasing the temperature for each temperature are illustrated. Before this note that T_0 is the initial temperature tuned for the first epoch and T_f is the final temperature tuned for the final epoch; i is the counter of the epochs and N is the total number of epochs considered primarily.

$$T_i = T_0 - i\frac{(T_0 - T_f)}{N} \tag{8}$$

$$T_i = \frac{(T_0 - T_f)(N+1)}{N(i+1)} + T_0 - \frac{(T_0 - T_f)(N+1)}{N} \tag{9}$$

$$T_i = \frac{1}{2}(T_0 - T_f)\left(1 - \tanh\left(\frac{10i}{N} - 5\right)\right) + T_f \tag{10}$$

$$T_i = T_0 - i^{\frac{\ln(T_0 - T_f)}{\ln(N)}} \quad (11)$$

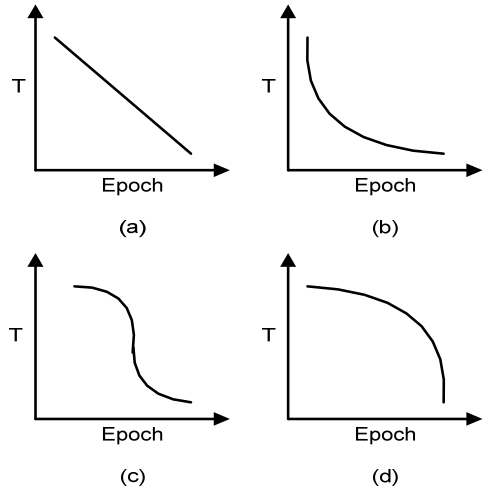

Fig. 2. Cooling behavior of temperatures in each epoch [17]

V. PARAMETER SETTING

Be we go through explanation of the parameter setting, it seems be necessary to note that the problem size (size of inbound and outbound trucks) is divided into two categories: small to medium size in which there are generally 6 to 14 trucks, and medium to large size in which there are 15 to around 20 trucks. We refer to these categories as *class* 1 and *class* 2 that each has 4 problem sets. As there are some parameters in the applied SA algorithm and each of them has some levels, we should specify the best level of each parameter so that the algorithm can facilitate us the best possible results. We first point out the parameters and their levels in Table I.

Table I
SA algorithm's parameters and their levels

Factor's Name	Factor's Value
Local search	Sections (a), (b), (c) and (d) of Fig. 1
Cooling schedule	Sections (a), (b), (c) and (d) of Fig. 2
Initial temperature (T_0)	100, 200, 300, 400
Final temperature (T_f)	20, 40, 60, 80
Number of epoch	5, 10, 15, 20

According to this table we have five parameters each having 4 levels; in fact we have to consider $5^4 = 625$ different conditions that affect negatively on our computational efforts. By applying *Taguchi* plan, we are able to consider fewer conditions by which the algorithm can be run under 16 different conditions for each problem set and there are 4 replications for each condition. Hence for each problem set 64 different executions will be held. In each run the best makespan is registered and the relative percentage deviation (*RPD*) will be calculated as a mean of evaluation for each algorithm given by:

$$RPD = \frac{M_i - M_{\min}}{M_{\min}} \quad (12)$$

where M_i is the makespan acquired for each of the 64 cases and $M_{\min}$ is the minimum makespan obtained among these 64 cases. If a parameter's level has the least *RPD* it will become the most effective level of the parameter. The behavior of the *RPD* values is depicted in Fig. 3 in which the numbers 1 to 5 represent five parameters described respectively in Table I.

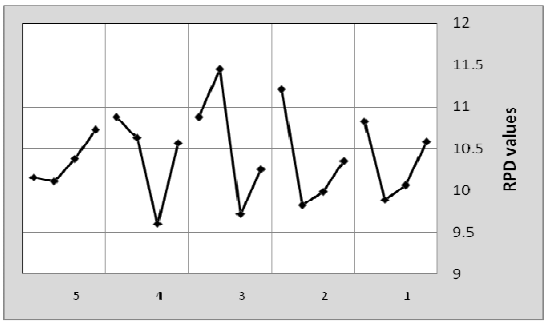

Fig. 3. *RPD* values of parameters in SA algorithm

By means of this figure, the best levels of each algorithm can be determined in that we have: for "local search" the best level is section (c) of Fig. 1, for "cooling schedule" the best level is section (c) of Fig. 2, for "initial temperature" the best level is 200, for "final temperature" the best level is 40 and finally for "number of epoch" the best level is 15.

VI. COMPUTATIONAL RESULT

For implementation of the SA algorithm, 8 problem sets were created at random in which 4 problem sets are dedicated to each class of problem size, class 1 and class 2 described before. Together with the problem proposed by Yu and Egbelu [1] we assess the SA algorithm on 9 different problem sets. The loading and unloading time per product is assumed to be one time unit in duration. The time for each truck's changeover is considered to be 75 time units and the time consumed to transfer each product via conveyor from receiving dock to shipping duck is 100 time units. All the applied algorithms were coded in C#.Net and these algorithms were run by a PC with a Pentium 1.8 GHz processor with 2GB of RAM. The outcomes of solving the 9 problem sets by both heuristic algorithm (presented by Yu and Egbelu [1]) and our SA algorithm are presented in the Table II. Due to this table, it is quite clear that our applied SA algorithm outperforms the heuristic proposed by Yu and Egbelu[1].

VII. CONCLUSION

In this paper we developed a meta-heuristic algorithm (Simulated Annealing Algorithm) in order to schedule the inbound and outbound trucks coming into and going out of cross-docking terminal. The SA algorithm was applied for 9 different problem sets, compared with the heuristic algorithm addressed by Yu and Egbelu [1]. This cross-

docking terminal had just one dock with several vehicles. However, as a hint for future research, this can be extended for many cross-docking terminals. Besides, this cross-dock can be linked to other cross-docks and storage centers so that a distribution network in a supply chain can be obtained so that many operations can be proceed more effectively. Moreover, this problem can be analyzed as a part of location – allocation problem in which the products not only can be distributed but also the site of storage centers can be determined.

REFERENCES

[1] W. Yu and P. J. Egbelu. "Scheduling of inbound and outbound trucks in cross docking systems with temporary storage". European Journal of Operational Research, vol. 184, pp. 377 – 396, 2008.
[2] M. A. Waller, C. R. Cassady and J. Ozment. "Impact of cross-docking on inventory in a decentralized retail supply chain". Transportation Research Part E, vol. 42, pp. 359 – 382, 2006.
[3] A. Ross and V. Jayaraman. "An evaluation of new heuristics for the location of cross-docks distribution centers in supply chain network design". Computers and Industrial Engineering. Vol. 55, pp. 64 – 79, 2008.
[4] W. H. Lee, J. W. Jung and K. M. Lee. "Vehicle routing scheduling for cross-docking in the supply chain". Computers and Industrial Engineering, vol. 51, pp. 247 – 256, 2006.
[5] I. F. A. Vis and K. J. Roodbergen. "Positioning of goods in a cross-docking environment". Computers and Industrial Engineering, vol. 54, pp. 677 – 689, 2008.
[6] G. M. Magableh, M. D. Rosetti and S. Mason. "Modeling and Analysis of a Generic Cross Docking Facility". In: Proceedings of the 2005 Winter Simulation Conference. Florida: Orlando, pp. 1613 – 1620.
[7] F. Chen and C. Y. Lee. "Minimizing the makespan in a two-machine cross-docking flow shop problem". European Journal of Operational Research, doi:10.1016/j.ejor.2007.10.051, 2007.
[8] Z. Miao, A. Lim and H. Ma. "Truck dock assignment problem with operational time constraint within cross-docks". European Journal of Operational Research, vol. 192, pp. 105 – 115, 2009.
[9] D. L. McWilliams. "Simulation-Based Scheduling for Parcel Consolidation Terminals: A Comparison of Iterative Improvement and Simulated Annealing". In: Proceedings of the 2005 Winter Simulation Conference. Florida: Orlando, pp. 2087 – 2093.
[10] G. A. Alvarez-Perez, J. L. Gonzalez-Valarde and J. W. Fowler. "Cross-docking-Just in time scheduling: an alternative solution approach". Journal of the Operational Research Society, vol. 60, pp. 554 – 564, 2009.
[11] J. J. Bartholdi III and K. R. Gue. "The best shape for a cross-dock". Transportation Science, vol. 38, pp. 235-244, 2004.
[12] N. Boysen. "Truck scheduling at zero-inventory cross-docking terminals". Computers and Operations Research, doi: 10.1016/j.cor.2009.03.010, 2009.
[13] N. Boysen, M. Fliedner and A. Scholl. "Scheduling inbound and outbound trucks at cross-docking terminals". OR Spectrum, doi 10.1007/s00291-008-0139-2, 2008.
[14] Y. A. Bozer and H. J. Carlo. "Optimizing inbound and outbound door assignments in less-than-truckload cross-docks". IIE Transactions, vol. 40, no. 11, pp. 1007 – 1018, 2008.
[15] J. F. Wang and A. Regan. "Real-time truck scheduling for cross-dock operations". Transportation Journal, vol. 47, no. 2, pp. 5 – 20, 2008.
[16] W. Yu. "Operational strategies for cross-docking systems". Iowa State University, Ph.D. Dissertation, 2002.
[17] M. Zandieh, M. Amiri, B.Vahdani and R. Soltani. "A robust parameter design for multi-response problems". Journal of Computational and Applied Mathematics, doi: 10.1016/j.cam.2008.12.019, 2009.

Table II

Best, average, and worst solutions acquired for SA algorithm and heuristic algorithm

Problem set	Problem size				Compound solution of heuristic	Simulated Annealing Algorithm		
	R	S	N	N_T		Worst	Average	Best
1	5	4	6	1030	1577	1577	1577	1577
2	12	9	11	6007	8270	8678	8073.9	7719
3	8	11	12	5293	7195	7890	7352.4	6952
4	9	14	8	5600	6109	7188	6251.7	5948
5	11	15	8	4786	6047	7335	6807.9	6594
6	17	15	17	9944	9944	10255	9408.3	8934
7	19	16	18	7964	11769	11616	11011.2	10653
8	15	15	15	5326	8152	8479	7741	7479
9	17	16	19	10852	10889	10611	10098	9611