Proceedings of the 2005 Winter Simulation Conference
M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, eds.

SIMULATION-BASED SCHEDULING FOR PARCEL CONSOLIDATION TERMINALS: A COMPARISON OF ITERATIVE IMPROVEMENT AND SIMULATED ANNEALING

Douglas L. McWilliams

Department of Industrial Technology
401 North Grant Street
Purdue University
West Lafayette, IN 47906, U.S.A.

ABSTRACT

This research explores the application of a simulation-based scheduling algorithm to generate unload schedules for processing feeder trailers in a parcel consolidation terminal. The study compares the performance of iterative improvement and simulated annealing to produce quality schedules. The paper reports the results from a number of experimental test problems.

1 INTRODUCTION

The parcel delivery industry is a significant segment of the transportation industry. According to the Bureau of Transportation Statistics' Commodity Flow Survey of 2002, the parcel delivery industry moved approximately 12.1% ($1.03 trillion) of goods in the United States in 2002. Factors that have increased the demand of the parcel delivery industry are just-in-time production principle (JIT), global manufacturing, and electronic retailing. For instance, JIT production and global manufacturing require reliable, high-speed, low-cost delivery service to transport goods in small batch sizes; and consumers continue to purchase goods such as personal computers, compact discs, books, jewelry, and apparel over the Internet, which require deliveries from the retailers' remote locations to the customers' sites.

To improve the delivery service, many companies—USPS, UPS, FedEx, and DHL—have invested in information technology and in high-speed automated sorting systems, with the objective of the high-speed automated sorting systems reducing the amount of time required to process large volumes of parcels through the parcel consolidation terminals. However, most high-speed sorting systems seldom operate at maximum throughput because the stated capacity does not consider the labor constraints of the systems. The unloading of incoming trailers at receiving docks and the loading of outgoing trailers at shipping docks continue to be relatively slow labor-intensive operations. These manual loading and unloading operations are the bottlenecks that limit the throughputs. Therefore, the challenge in the parcel delivery industry is to maximize the throughputs of the parcel consolidation terminals, which can only be achieved by considering the loading and unloading constraints.

A typical sorting system in a parcel consolidation terminal consists of a complex network of conveyors to move parcels from receiving docks to shipping docks (Figure 1). The terminal in Figure 1 has three receiving docks represented by nodes 1, 2, and 3, and nine shipping docks represented by nodes 10 through 18. The shaded rectangles represent incoming or feeder trailers to be processed at the receiving docks. A large number of feeder trailers must be processed at the receiving docks. The clear rectangles represent outgoing trailers. Each dock is usually allocated at a particular destination. Nodes 4, 5, and 6 represent initial diverting points, and nodes 7, 8, and 9 represent final diverting points. Parcels entering the sorting system are diverted to appropriate induction line conveyors. The induction line conveyors move the parcels downstream to mainline conveyors that feed output cells. At nodes 7, 8, and 9, the parcels are diverted to the correct shipping docks for loading onto outgoing trailers.

Not only do the manual loading and unloading operations have a negative impact on system performance. But the flow pattern of parcels through the system may also have an adverse affect on performance. The reason is that each incoming trailer contains a batch of heterogeneous parcels. The batch is heterogeneous because each parcel is not bound for the same shipping dock. Based on the parcels' destinations, each parcel takes a predefined route through the system. For example, at a small terminal, regardless of the destinations for the parcels, each parcel is loaded onto the same trailer. At the large consolidation terminals, the parcels get unloaded and routed to outgoing trailers for the appropriate destinations in the delivery system. When an incoming trailer is unloaded at a receiving dock, all parcels are unloaded before the trailer leaves the dock. Given that consolidation terminals have multiple re-

ceiving docks, many incoming trailers are unloaded simultaneously. The outcome may result in an excessive number of specific parcel types entering the system. In this case, a parcel type is based on the flow of a parcel over a particular lane of the conveyor network to a particular shipping dock. If an excessive number of specific parcel types enters the system, the flow of parcels to a particular shipping dock may exceed the loading capacity to the particular outgoing trailer. Parcels begin to form a queue in the lane to the shipping dock. If the queue length propagates upstream, the flow of parcels over the other lanes to the other shipping docks may becomes blocked because of the common conveyors. The end results will be flow congestion, idle workers, and loss capacity.

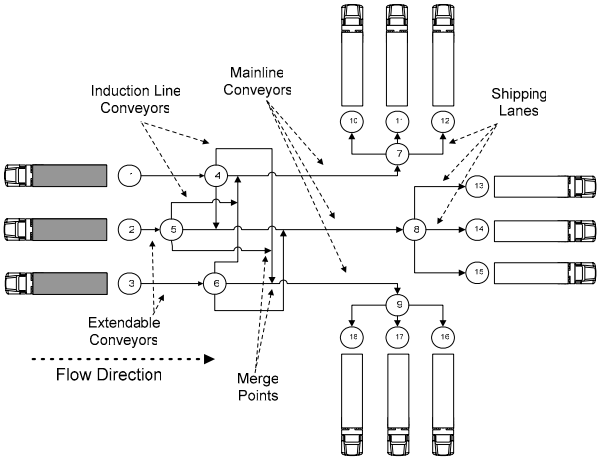

Figure 1: A Basic Sorting System.

To reduce the risk of blocking and thus the time required to process a set of parcels through the system, the numbers of specific parcel types entering the sorting system must be metered at the receiving docks. However, controlling the numbers of specific parcel types flowing through a particular lane at any give time is a difficult task.

This paper addresses the trailer-scheduling problem by considering the application of SA to solve the problem. In Section 2, the relevant literature is discussed. In Section 3, the problem statement and assumptions are presented. A detailed description of the simulation-based scheduling algorithm is presented in Section 4. The algorithm is applied to small, medium, and large size problems; Section 5 shows results of the algorithm. Section 6 contains the conclusion and future research direction.

2 RELEVANT LITERATURE

The trailer-dock assignment problem is a similar problem to the trailer-scheduling problem. The difference in the problems stems from the fact that a trailer-dock assignment problem is a static workload balancing problem (typically a design issue) and this trailer-scheduling problem is a dynamic workload balancing problem (typically an operational issue). Peck (1983), Tsui and Chang (1990 1992), Gue (1995 1997), Bartholdi and Gue (1998 2000), and Hwang and Cho (2002) have considered the trailer-dock assignment problem in the less-than-truckload (LTL) industry. Masel and Goldsmith (1997) and Masel (1998) have considered the trailer-dock assignment problem in the parcel delivery industry, and McWilliams and Stanfield (2005) have studied the trailer-scheduling problem in the industry.

Peck (1983) studied the effects of trailer-dock assignment in LTL terminals. A "greedy" heuristics is proposed to improve productivity by assigning incoming trailers and outgoing trailers to docks to minimize the total distance material handlers travel during the transfer operations. The solution of the algorithm is evaluated using a simulation model of an LTL terminal. The results show a 5% to 15% reduction in the time spans of transfer operations. Tsui and Chang (1990) proposed an iterative improvement heuristics to the trailer-dock assignment problem with the objective of minimizing total forklift travel distance. Tsui and Chang (1992) proposed an improved version of the assignment heuristics. The model results in optimal assignments for small size problems. Computational time increases significantly as the number of trailers and number of docks increase.

Gue (1995 1999) proposed a solution methodology for the trailer-dock assignment problem. While previous studies sought to minimize total travel distance, this solution methodology minimized total transfer time. Since transfer time is a composition of both move time and wait time, minimizing total transfer time accounts for the total wait time. A greedy algorithm is developed to solve the problem. Results show a 3% to 30% reduction in total transfer time that equates to a 2% to 5% reduction in overall labor cost. Bartholdi and Gue (2000) improved the solution methodology by replacing the greedy algorithm with simulated annealing. The results show labor cost reductions of approximately 20% to 45%.

Masel and Goldsmith (1997) used a simulation study to evaluate various trailer-dock assignments for consolidation terminals in the parcel delivery industry. Masel (1998) proposed a list-scheduling heuristic to construct adequate trailer-dock assignments. No computational results are provided for either approach. McWilliams and Stanfield (2005) proposed a simulation-based scheduling algorithm (SSA) to assign incoming trailers to unload docks at the parcel consolidation terminals. The SSA embeds an iterative improvement heuristics to search for good unload schedules. The results show a 15% to 25% reduction in the required time to complete the transfer operations. This study compares the performance of simulated annealing and iterative improvement to generate good solutions to the problem.

3 THE PROBLEM STATEMENT

Moving parcels from incoming trailers to outgoing trailers is a transfer operation. The trailer-scheduling problem of minimizing the time span of the transfer operation is defined as follows. A parcel consolidation terminal has a set of incoming trailers. Each incoming trailer contains a batch of heterogeneous parcels. The parcels must be unloaded at receiving docks, routed to appropriate shipping docks, and then loaded onto outgoing trailers. The objective is to assign the incoming trailers to the receiving docks with the objective of minimizing the required time to complete the transfer operation. The time starts when the first parcel enters the system and ends when the last parcel exits the system. The basic assumptions in this study as stated in McWilliams and Stanfield (2005) are as follows:

1. All trailers are available at the beginning of the transfer operation;
2. Empty incoming trailers are instantaneously replaced with nonempty incoming trailers;
3. Full outgoing trailers are instantaneously replaced with empty outgoing trailers;
4. Setup time for trailers are negligible and equal to zero;
5. No priority exists for any one trailer;
6. No trailer can be preempted once its unloading or loading has begun;
7. The number of receiving docks and the number of shipping docks remain fixed throughout the duration of the transfer operations;
8. Receiving docks are identical—a receiving dock can process any incoming trailer and receiving docks have equal and constant service rates;
9. Shipping docks are identical—a shipping dock can process any outgoing trailer and shipping docks have equal and constant service rates;
10. Parcel movement through the sorting system is via a network of conveyors;
11. Conveying times are negligible and equal to zero; and
12. Loading, unloading, and sorting times are known and deterministic.

4 METHODOLOGY

A conveyor network is a queuing network. The complexity of queuing networks makes it extremely difficult to develop analytical models of their stochastic behavior. Thus, this study assesses the use of an SSA with simulated annealing to generate unload schedules. An SSA is the implementation of computer simulation to optimize systems that are too complex to be modeled analytically (Andradóttir, 1998). SSA involves finding the best schedule where the performance evaluation is based on results from a discrete-event simulation model of the real-world system. Figure 2 illustrates the conceptual design of the SSA. The proposed SSA embeds a local search heuristics, a list-scheduling algorithm, and a deterministic simulation model. This study compares the performance of iterative improvement and simulated annealing to find good unload schedules.

4.1 Iterative Improvement

Iterative improvement is a local search heuristics that uses permutation representation for the solution structure. It starts with a generated initial random solution s, which is stored as the current solution. At each iteration, the heuristic generates a new solution s' using a particular move strategy. The heuristic then compares the quality of the solution s', $C(s')$, with the quality of the current solution s, $C(s)$. If $C(s') \leq C(s)$, the heuristics replaces the current solution with the new solution; otherwise, the heuristics keeps the current solution. This procedure continues until a termination criterion is satisfied, i.e., maximum iterations or maximum computational time.

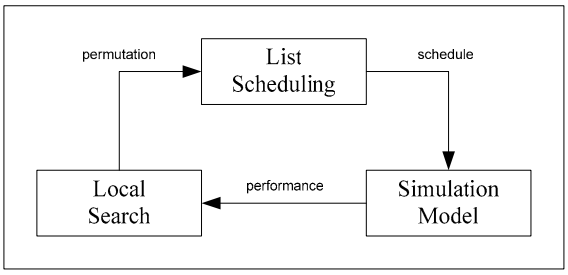

Figure 2: The Conceptual Design of the SSA.

4.2 Simulated Annealing

Simulated Annealing (SA) is a meta-strategy local search heuristics. Similar to iterative improvement, SA starts with a generated initial random solution s, which is stored as the current solution. At each iteration, the heuristic generates a new solution s' using a move strategy. The heuristic then compares the quality of the solution with the quality of the current solution s. If $C(s') \leq C(s)$, the heuristics replaces the current solution with the new solution; otherwise, SA uses random acceptance to determine if the new solution is to replace the current solution. The random acceptance strategy allows occasional uphill moves to be accepted with certain probabilities to avoid producing poor local optimum solutions. The probability of acceptance equation is

$$P = \exp\left(\frac{\Delta C_{ss'}}{c_k}\right),$$

where $\Delta C_{ss'} = C(s') - C(s)$ and c_k is the temperature (control parameter). The equation is the Metropolis criterion (Me-

tropolis et al., 1953). For a non-homogeneous SA algorithm, the temperature c_k is incrementally decreases in magnitude as the algorithm progresses according to the cooling schedule. At each increment of c_k, a number of solutions are evaluated. The decreasing of c_k causes the acceptance probability of degenerative solutions to lower as the algorithm progresses. Kirkpatrick et al. (1982) propose a cooling schedule for SA. The temperature c_k for $k > 0$ is calculated by $c_{k+1} = \alpha \times c_k$ where α is the control parameter for the cooling schedule. Limiting the maximum value for k sets the stopping criterion.

4.3 Move Strategy

Each new solution *s'* is a neighbor of the current solution *s*. The size of the neighborhood *NB*(*s*), a subset of the set *S* of feasible solutions, is defined by a specified move strategy. To obtain new solutions, small changes (perturbations) are made to the current solution. The selected move strategy has a crucial influence on the quality of solutions obtainable and the required computational time. If the neighborhood is too small, the search is very restricted, and thus it may be nearly impossible to reach good solutions. On the other hand, if the neighborhood is too large, a good solution may be found; however, the computational time to find the solution may be significantly high. Various move strategies were evaluated in McWilliams and Stanfield (2005). The best move strategies are evaluated under simulated annealing. The move strategies are trailer-insertion (TIS), 2-trailer swap (2TS), 3-trailer swap (3TS), and random (RND).

TIS. The set of solutions obtainable by removing a trailer from one position in the permutation and inserting it into another position (Figure 3). The size of the neighborhood is $O(n^2)$.

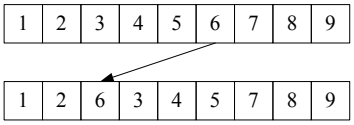

Figure 3: The Trailer Insertion Move Strategy.

2TS. The set of solutions obtainable by exchanging the positions of two trailers in the permutation (Figure 4). The size of the neighborhood is $O(n^2)$.

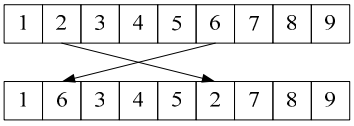

Figure 4: The 2-Trailer Swap Move Strategy.

3TS. The set of solutions obtainable by exchanging the positions of three trailers in the permutation (Figure 5). The size of the neighborhood is $O(n^3)$.

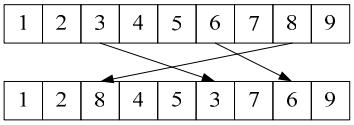

Figure 5: The 3-Trailer Swap Move Strategy.

RND. The universal set of all solutions obtainable by generating a new random permutation (Figure 6). The size of the neighborhood is $O(n!)$.

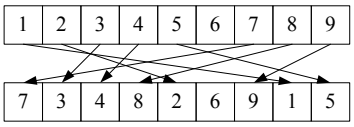

Figure 6: The Random Move Strategy.

4.4 List-scheduling algorithm (LSA)

To compute the quality of each new solution, a LSA maps each permutation sequence to a corresponding unload schedule. Each solution (permutation) is a priority list, where incoming trailers appear on the list in decreasing order of priority. The LSA constructs a feasible schedule from each list systematically by assigning one trailer after the other to the first available receiving dock. Figure 7 shows two consecutive permutations: s_1 and s_2. At iterations 1 and 2 respectively, the LSA retrieves permutations s_1 and s_2 then maps the appropriate unload schedules during the simulation runs.

4.5 Simulation Model

A detailed deterministic simulation model of the parcel consolidation terminal is used to evaluate each unload schedule. The model includes all internal operations that affect the operating performance of the system, such as the unloading, loading, and sorting operations. The speed and availability of these resources affect the time span of the transfer operations. For additional reading on the simulation modeling of parcel consolidation terminals, the reader is referred to Rohrer (1995). The evaluation function for schedule *s'* is

$$f = eval(s'), \text{ where } s' \in NB(s).$$

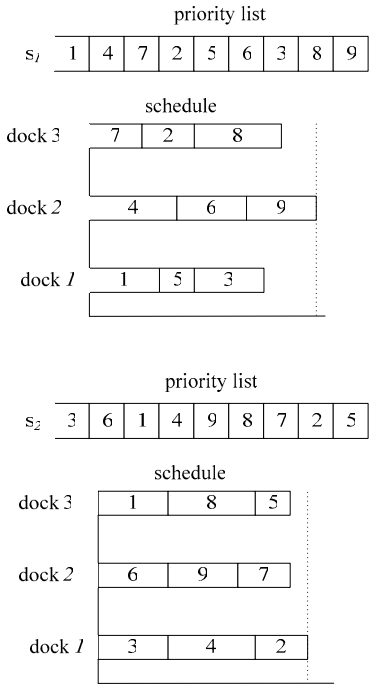

Figure 7: Illustration of the List-Scheduling Algorithm.

5 EXPERIMENTAL RESULTS

An experimental analysis is conducted to evaluate the SSA. The experiment determines the impact of simulated annealing and iterative improvement, along with the move strategies. Three sizes of parcel consolidation terminals are considered: "small," "medium," and "large." The small terminal has three receiving docks and nine shipping docks as in Figure 1. The medium terminal has 10 receiving docks and 32 shipping docks, and the large terminal has 50 receiving docks and 160 shipping docks. The hourly throughput capacities of the small, medium, and large terminals are 1,800, 6,000, and 30,000 parcels, respectively. These throughput capacities are so selected to reflect the real-world system.

The test problems are taken from McWilliams and Stanfield (2005). Twenty random instances are used for the small and medium size terminals, and one random instances is used for the large terminal. The constraints for the test problems are as follows. The cumulative loads from the incoming trailers to the shipping docks are evenly distributed. For the small, medium, and large terminals, the number of trailers in each instance is 30, 100, and 500, respectively. There are 600 parcels in each trailer; thus, the cumulative volume sizes for the small, medium, and large terminals are 18,000, 60,000, 300,000 parcels, respectively.

The local search heuristics are coded in C language and run on IBM workstations with 3 GHz and 1 GB RAM. The simulation models of the parcel consolidation terminals were implemented using Arena 7.0 simulation software package, a product of Rockwell Automation. The average computational time for each replication of the simulation models are 0.03, 0.20, and 2.00 minutes for the small, medium, and large terminals, respectively. Maximum iteration is used as the termination: 200 iterations for the small and medium instances and 1000 for the large instance.

Table 1 shows the summary statistics for the small instances. The table also shows the initial and worst solutions statistics. The worst performance is 1206 minutes. Compared to the initial and worst solutions, the SSA generates good unload schedules for this problem. Table 2 is an ANOVA table. With alpha equal to 0.05, the table shows that there is no statistical significance between the algorithms. The iterative improvement and the simulated annealing generate similar results; however, the table does show that there is statistical significance between the move strategies. Figure 8 is an interaction plot. The figure shows that RND results in the worst performance for the SSA under both iterative improvement and simulated annealing. In fact, simulated annealing performance best with pairwise interchange.

Table 1: Summary for Small Instances.

	Min	Mean	Max	Std Dev.
Initial	730	837	989	65
Worst	974	1037	1206	44
Iterative				
TIS	616	640	668	12
2TS	613	640	670	15
3TS	612	646	677	16
RND	648	690	720	15
SA				
TIS	616	641	660	11
2TS	615	635	671	13
3TS	619	645	678	17
RND	664	689	714	13

Table 2: ANOVA Summary for Small Instances.

Source	*DF*	*SS*	*MS*	*F*	*P*
Move strategy	3	63691	21230	97.76	0.000
Algorithm	1	6	6	0.03	0.864
Interaction	3	422	141	0.65	0.585
Error	152	33010	217		
Total	159	97130			

Table 3 shows the summary statistics for the medium instances. This table also shows the initial and worst solutions. Compared to the initial and worst solutions, SSA still generates good solutions. Table 4 is an ANOVA table of the medium instances. The table shows similar results as in Table 2. No statistical significance exists between iterative improvement and simulated annealing; however, statistical significance does exist between the move strate-

gies. The significance is illustrated in Figure 9. Finally, Figure 10 shows a plot of the large instance for 1000 iterations. The plot shows that simulated annealing performs best. The initial solution quality was 851 minutes, and the resulting solution quality for the iterative improvement and simulated annealing were 775 minutes and 752 minutes, respectively. The runtime for the large problem was 1847 minutes.

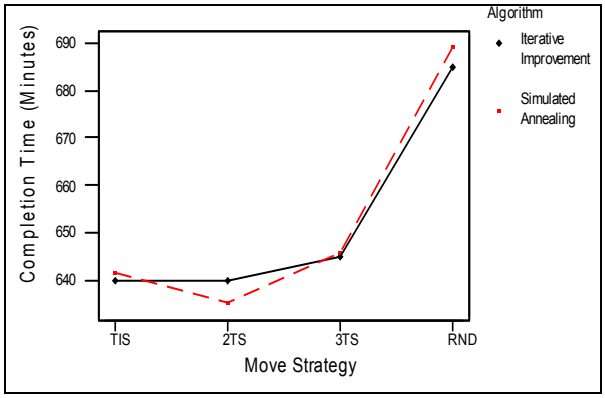

Figure 8: Interaction Plot for Small Problems.

Table 3: Summary for Medium Instances.

	Min	Mean	Max	Std Dev.
Initial	708	812	890	40
Worst	956	1134	1320	51
Iterative				
TIS	639	664	692	13
2TS	637	657	673	10
3TS	637	661	684	14
RND	672	703	713	9
SA				
TIS	637	665	695	14
2TS	639	661	718	16
3TS	635	664	682	12
RND	688	705	721	9

Table 4: ANOVA Summary for Medium Instances.

Source	*DF*	*SS*	*MS*	*F*	*P*
Move strategy	3	53204	17735	116.92	0.000
Algorithm	1	276	276	1.82	0.180
Interaction	3	75	25	0.17	0.919
Error	152	23056	152		
Total	159	76611			

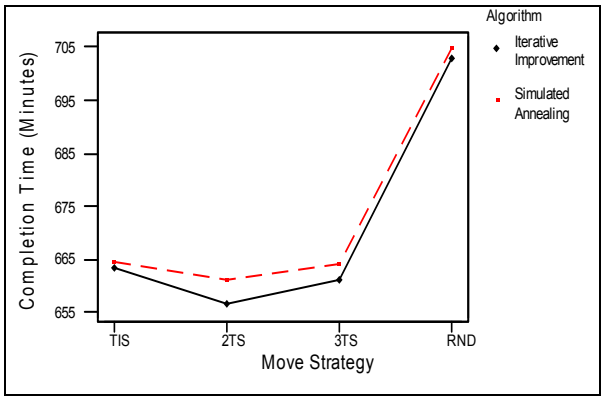

Figure 9: Interaction Plot for Medium Problems.

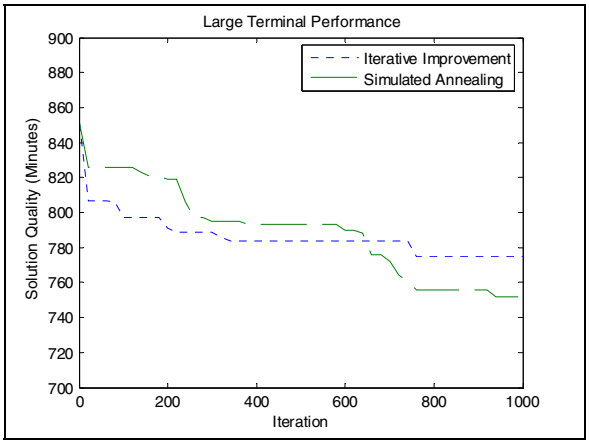

Figure 10: A Plot of Large Terminal Performance per Iteration.

6 CONCLUSIONS AND FUTURE RESEARCH

This research compared the performance of simulated annealing and iterative improvement in a simulation-based scheduling algorithm to find unload schedules for the trailer-scheduling problem. Four different move strategies were evaluated for the local search methods. The results show that the SSA produces unload schedules that reduce the required time of transfer operations by 15% to 25% for the small and medium instances. Each move strategy produced good schedules except the random move strategy. Also, there no statistical significance between iterative improvement and simulated annealing. The SSA provides insight for future research on the trailer-scheduling problem. Future research should include the application of distributed simulation or a meta-modeling approach to reduce required computational time to obtain given solutions. Other important issues involve systematically relaxing the assumptions.

REFERENCES

Andradóttir, S. 1998. Simulation optimization. *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice*. New York: John Wiley & Sons: 307-333.

Bartholdi, J., and K. Gue. 2000. Reducing labor costs in an ltl crossdocking terminal. *Operations Research* 48: 823-832.

Gue, K. 1995. Freight terminal layout and operations. Dissertation, Georgia Institute of Technology.

Gue, K. 1999. Effects of trailer scheduling on the layout of freight terminals. *Transportation Science* 33: 419-428.

Hwang, H., and G. Cho. 2002. Layout design for improving of freight terminal performance. *Proceedings of the 30th International Conference on Computers and Industrial Engineering*: 347-352.

Kirkpatrick, S., C. D. Gelatt Jr., and M. P. Vecchi. 1982. Optimization by simulated annealing, *IBM Research Report RC* 9355.

Masel, D. 1998. Adapting the longest processing time heuristic for output station assignment in a transfer facility. *Proceedings of the 1998 Industrial Engineering Research Conference*.

Masel, D., and D. Goldsmith. 1997. Using a simulation model to evaluate the configuration of a sortation facility. *Proceedings of the 1997 Winter Simulation Conference,* ed. S. Andradóttir, K. Healy, D. Withers, B. Nelson: 1210-1213.

Metropolis, N., A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. 1953. Equation of state calculations by fast computing machines, *J. of Chem. Physics* 21.

Peck, K. 1983. Operational analysis of freight terminals handling less than container load shipment. Dissertation, University of Illinois at Urbana-Champaign.

Rohrer, M. 1995. Simulation and cross docking. *Proceedings of the 1995 Winter Simulation Conference,* ed. C. Alexopoulos, K. Kang, W. R. Lilegdon, and D. Goldsman. 846-849.

Tsui, L., and C. Chang. 1990. A microcomputer based decision support tool for assigning dock doors in freight yards. *Computers and Industrial Engineering*, 19: 309-312.

Tsui, L., and C. Chang. 1992. An optimal solution to dock door assignment problem. *Computers and Industrial Engineering*, 23: 283-286.

AUTHOR BIOGRAPHY

DOUGLAS L. MCWILLIAMS is an assistant professor in the Department of Industrial Technology at Purdue University. He teaches courses in lean manufacturing and industrial distribution. He received his PhD and MSc degrees in industrial engineering from North Carolina A&T State University and Mississippi State University, respectively. His current research interest is the applications of simulation and operations research in the transportation industry. He is a member of INFORMS, IIE, APICS, and NAIT. His e-mail is dmcwilli@purdue.edu and his Web address is www.tech.purdue.edu/It/ Facstaff/mcwilliams/index.html.