# Dispatch Problem of Automated Guided Vehicles for Serving Tandem Lift Quay Crane

Yao Xing, Kai Yin, Luca Quadrifoglio, and Bruce X. Wang

**New quay cranes (QCs) have been designed to increase terminal productivity by lifting more containers simultaneously. But QC productivity relies on efficient cooperation with the vehicles carrying the containers. This paper investigates the synchronization scheduling problem between the automated guided vehicles and these new QCs. The problem is formulated as a mixed integer linear programming model. Because of the problem's complexity, a heuristic dispatch rule is proposed for practical purposes. Then, to balance the computation time and the quality of the solution, a neighborhood search method is designed by investigating the working sequences of automated guided vehicles. Numerical experiments show that both heuristics obtain good solutions within extremely short times and that the neighborhood search method generally performs better in relation to the objective value.**

Steadily increasing freight volumes, in addition to the development of jumbo container ships, put significant pressure on freight transportation and result in calls for higher productivity in the terminals, which are almost unanimously recognized to be the bottleneck of freight transportation and are suffering from inefficient operations and limited capacity. To solve those problems, the application of new, advanced equipment and efficient operation and management are becoming more attractive compared with physical enlargement of the terminal size. Today, automated container terminals (ACTs) (Figure 1) have been adopted in some of the busiest terminals, such as Hamburg, Germany; Singapore; and Rotterdam, Netherlands. ACT refers to the unmanned terminal controlled by advanced equipment and a high-level information network. Compared with traditional terminals, ACTs provide more advantages in reducing labor and operation cost (*1*).

When a vessel arrives at an ACT, unloaded containers are hoisted by the quay cranes (QCs) and then delivered by automated guided vehicles (AGVs) to the storage area and positioned by yard cranes to specific storage blocks. The loading process operates in the opposite direction. In an ACT, AGVs take the place of trucks and become the main horizontal transporter. A study showed that the application of AGVs could double the throughput of a terminal (*2*). In addition, it can reduce labor costs as well as emissions because AGVs do not need drivers and are powered by electricity (Figure 2*a*). The other critical equipment are the QCs, whose working speed directly influences the efficiency and throughput of the terminal. New QCs, like tandem-lift QCs, are designed for faster loading–unloading operations to meet the demands of megavessels. In comparison with conventional single-trolley cranes, QCs can double productivity by lifting four adjacent 20-ft or two 40-ft containers simultaneously (Figure 2*b*).

Besides the adoption of new, advanced pieces of equipment, synchronization and cooperation of operations are more important and difficult issues within terminals. For example, Lind et al. (*3*) pointed out that noncrane delay would reduce the tandem-lift QC's efficiency up to 50%. AGVs are expected to arrive at the QC just when the QC is ready to place or lift containers on or from it. Otherwise, the QC has to wait for the AGV, or the AGV has to wait for the QC. To minimize such idle time, the dispatch sequences of AGVs should coincide with the working schedule of the QCs. With tandem-lift cranes, the situation becomes more complex. The QC cannot start to load or unload containers until two AGVs are actually present. However, those two containers may be stored in different storage blocks. Thus, the two AGVs dispatched to serve a QC may travel from different yard blocks during the loading process or to different yard blocks during the unloading process. If two AGVs are simply fixed as a group, the problem would degrade to the traditional AGV-dispatching problem in terminals. However, it would result in less flexibility in the dispatch and reduce the efficiency of AGVs. Today, AGV dispatching in container terminals follows some simple rules, such as first come, first served; the nearest-vehicle rule; and so on. In some cases, especially when unexpected events or accidents happen, vehicle dispatch follows the operator's commands.

The rest of this paper is organized as follows. The next section offers a literature review of the AGV-dispatching problem in container terminals. Then the problem statement and the mathematical model of the problem are introduced. Next, three solution methods are described and numerical experiments performed. Conclusions close the paper.

## LITERATURE REVIEW

The objective of most transportation optimization in a terminal is to maximize the terminal's throughput or minimize the ship's turnaround time through optimization of the delivery schedule of AGVs.

Vis et al. (*4*) developed a minimum flow algorithm to determine the number of AGVs required at a semiautomated container terminal. Liu et al. (*5*) discussed, with simulations, the relationship between the number of AGVs and the terminal's layout. Duinkerken and Ottjes (*6*) developed a simulation to determine the sensitivity concerning a number of parameters like the number of AGVs, maximum AGV speed, and so on. Vis and Harika (*7*) pointed out how the

Zachry Department of Civil Engineering, Texas A&M University, College Station, TX 77801. Corresponding author: L. Quadrifoglio, quadrifo@tamu.edu.

*Transportation Research Record: Journal of the Transportation Research Board, No. 2273,* Transportation Research Board of the National Academies, Washington, D.C., 2012, pp. 79–86.
DOI: 10.3141/2273-10

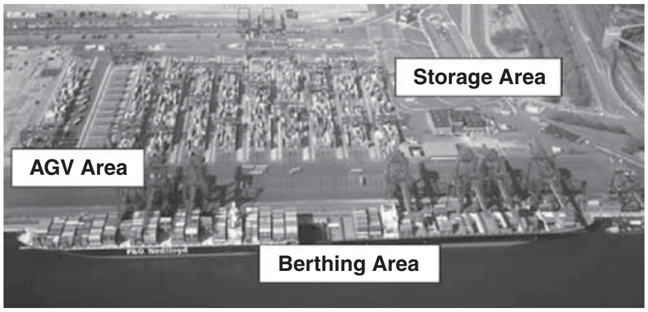


FIGURE 1 Layout of Container Terminal, Altenwerder, Hamburg, Germany. (Source: http://www.hhla.de/de/Geschaeftsfelder/HHLA_Container/Altenwerder_(CTA)/Daten_und_Fakten.jsp.)

design of the terminal and technical aspects of QCs affect the number of vehicles required and the choice for a certain type of equipment.

The dispatching and routing problem of AGVs can be formulated by a mixed integer program (MIP) model. Kim and Bae (*8*) suggested a network-based MIP model for AGV dispatching and provided a heuristic algorithm to minimize the total idle time of a QC resulting from the late arrivals of AGVs. Choi and Tcha (*9*) proposed an approach based on column generation to solve the vehicle routing problem. In this approach, the feasible columns are generated by emulating dynamic programming schemes, and the experiment with the benchmark tests confirms that the proposed approach outperforms all existing algorithms. Meersmans and Wagelmans (*10*) considered an integrated problem with the scheduling of different equipment at automated terminals. They presented a branch-and-bound algorithm and a heuristic beam search algorithm to minimize the makespan (the difference between the start and finish of a sequence of tasks) of their schedule. Lim et al. (*11*) proposed an auction-based assignment algorithm in the sense that it makes dispatching decisions through communication between related vehicles and machines for matching multiple tasks with multiple vehicles. Their method takes into account future events, and its performance is evaluated through a simulation study. Grunow et al. (*12, 13*) proposed a flexible-priority rule for dispatching multiload AGVs, and an MIP formulation was developed for the optimal solution in instances of small problems. A hybrid approach combining the MIP model with a heuristic was also proposed for real application. Briskorn et al. (*14*) solved the assignment of jobs to AGVs both with a heuristic based on the greedy-priority rule and with an exact algorithm. They formulated the assignment without due times and solved it on the basis of a rough analogy to inventory management, avoiding the estimates of driving times, completion times, due times, and tardiness. Homayouni et al. (*15*) solved the integrated scheduling of QCs and AGVs by using a simulated-annealing algorithm. They investigated the effects of initial temperature and the number of trials on the algorithm and compared the results from the simulated-annealing algorithm with ones from the mixed integer linear programming (MILP) mode.

However, the QCs discussed in all those papers were limited to the conventional single-trolley QCs. Very few papers investigated the application and performance of tandem-lift QCs in container terminals, and even fewer papers studied the AGV dispatching rules serving them. From the modeling perspective, operations of tandem-lift QCs are much more challenging. The existing papers about tandem-lift QCs are mainly limited to the introduction of its configurations and productivities (*3*). Bae et al. (*16*) compared the performance of different vehicles combining with the QCs of various types by using simulation software. Lin and Chao (*17*) developed a two-phase method for choosing a suitable advanced QC for terminal operators. The first phase identifies the determinants influencing selection of QCs by applying exploratory factor analysis, and the second phase applies the process of fuzzy analytic hierarchy to compare alternatives.

## MATHEMATICAL MODELS AND METHODOLOGY

For this problem, one 40-ft container or two 20-ft containers hoisted by a QC simultaneously are viewed as one unit, and each container group contains two such units. The terminal layout used in this paper is a typical ACT layout with cross-lane AGV paths (Figure 3).

### Problem and Objective

AGVs conduct only two kinds of tasks: picking up a container from its original position and dropping it off at its destination. For any task conducted at the QC's side, the QC has a preplanned start time, which is controlled by the QC's working schedule. If the AGV arrives later than the QC, the QC has to wait and, as a result, such


(a)

(b)

FIGURE 2 Containers on (*a*) AGV and (*b*) tandem-lift QC.

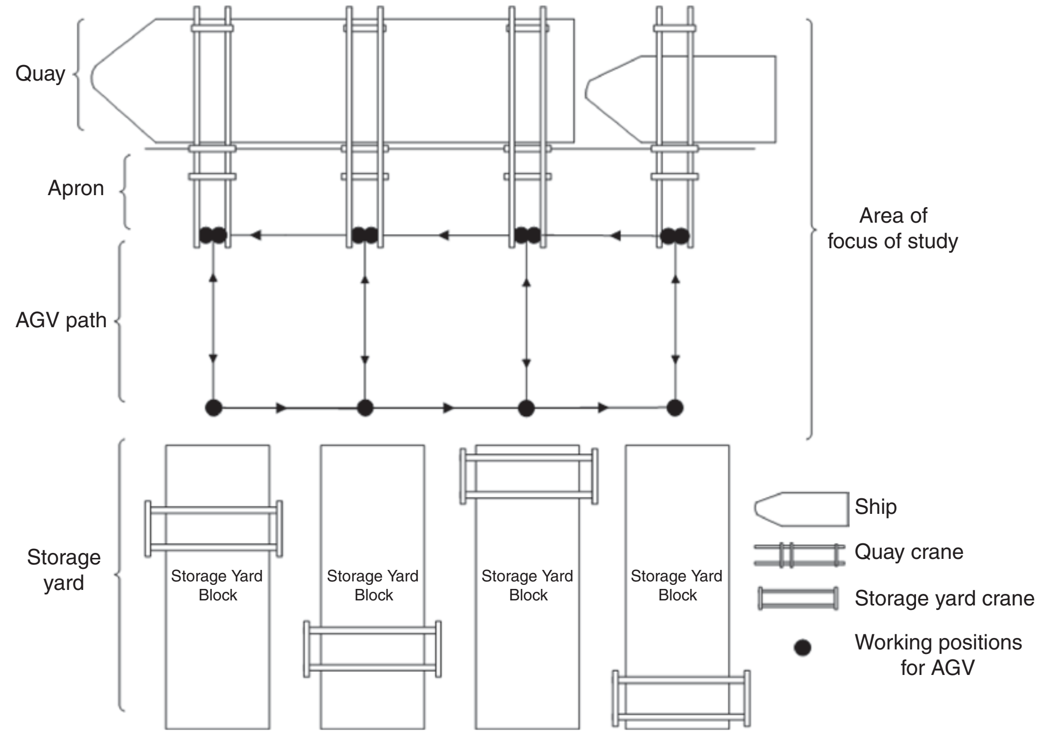


**FIGURE 3 ACT layout for cross-lane AGV paths.**

lateness reduces the QC's productivity. The objective of the scheduling problem investigated in this study is to minimize the overall lateness for all QCs during the planning horizon.

To make the problem suitable for the mathematical model without substantially affecting the characteristics of the real processes, the authors make three assumptions:

1. Each QC's loading–unloading sequences and the containers' storage plan are already known, and it is always true in the terminal's operation.
2. All the AGVs are homogeneous, and they run at the same speed all the time. Although not perfectly accurate, this assumption simplifies the model by disregarding acceleration–deceleration when the AGVs turn or approach the cranes.
3. A yard crane is always available to serve the AGVs.

## MILP Formulation

A MILP formulation is proposed in this paper to describe the AGV dispatch problem. Because they are similar to the classic vehicle routing problem, most proposed MILP models share similar structures (*18, 19*). In this problem, besides the known flow constraints and time constraints, additional time constraints are needed for the QC's operation because it cannot start until both AGVs have arrived. In the model, the whole layout is viewed as a network. The cranes are viewed as the nodes, and the guide paths are the arcs in this network. The notations needed for the formulation follow:

$V$ = set of all AGVs pooled to serve the QCs;
$C$ = set of all containers needed to be discharged or loaded during planning horizon;
$Q$ = set of all QCs discharging or loading containers;
$Y$ = set of all yard cranes in stack area;
$\text{pick}_c$ = AGV picks up container $c$ at QC or yard crane;
$\text{drop}_c$ = AGV drops off container $c$ at QC or yard crane;
$P = \{\text{pick}_1, \text{pick}_2, \ldots, \text{pick}_c\}$, $c \in C$ = set of all pickup tasks;
$D = \{\text{drop}_1, \text{drop}_2, \ldots, \text{drop}_c\}$, $c \in C$ = set of all drop-off tasks;
$T = P \cup D$ = set of all tasks, including pickup and drop-off tasks;
$S = T \cup 0$, where 0 is dummy start task for each AGV;
$E = T \cup 0$, where $e$ is dummy end task for each AGV;

$$\text{qt}(c) = \begin{cases} \text{pick}_c, \text{ if } \text{loc}_{\text{pick}_c} \in Q \\ \text{drop}_c, \text{ if } \text{loc}_{\text{drop}_c} \in Q \end{cases} = \text{quay-side task of container } c;$$

$$\text{yt}(c) = \begin{cases} \text{pick}_c, \text{ if } \text{loc}_{\text{pick}_c} \in Q \\ \text{drop}_c, \text{ if } \text{loc}_{\text{drop}_c} \in Q \end{cases} = \text{yard-side task of container } c;$$

$\text{QT} = \{\text{qt}(1), \text{qt}(2), \ldots, \text{qt}(c), \ldots\}$, $c \in C$ = set of all quay-side tasks;
$\text{YT} = \{\text{yt}(1), \text{yt}(2), \ldots, \text{yt}(c), \ldots\}$, $c \in C$ = set of all yard-side tasks;
$\text{twin}(c)$ = container hoisted simultaneously with container $c$, $c \in C$; and
$\text{pre}_{\text{qt}(c)}$ = qt($c$)'s predecessor in QC's working sequence, $c \in C$.

### *Parameters*

$\text{dis}(i, j)$ = distance AGV needs to travel from node $i$ to node $j$, $i, j \in N$;
$h$ = time crane needs to load or unload container onto or from AGV; and
cycle = interval in QC's working sequence decided by its working speed.

*Variables*

$x^v_{i,j}$ = binary variable = 1 if AGV $v$ is dispatched to complete the task $j$ immediately after completing task $i$, $i \in S$, $j \in E$;
$\text{start}_i$ = start time of task $i$, $i \in S$;
$\text{arrive}_i$ = time AGV arrives at node where task $i$ is, $i \in S$;
$\text{leave}_i$ = time AGV leaves after completion of task $i$, $i \in S$; and
$\text{ready}_i$ = ready time of a quay-side task $i$, $i \in$ QT.

## Formulation of MILP

Objective:

$$\min \sum_{i \in QT} \text{start}_i - \text{ready}_i \tag{1}$$

subject to

$$\sum_{i \in P} x^v_{0,i} = 1 \qquad \forall v \in V \tag{2}$$

$$\sum_{i \in D} x^v_{i,e} = 1 \qquad \forall v \in V \tag{3}$$

$$\sum_{m \in S} x^v_{m,i} = 1 \qquad \forall v \in V,\ \forall i \in T \tag{4}$$

$$\sum_{n \in E} x^v_{i,n} = 1 \qquad \forall v \in V,\ \forall i \in T \tag{5}$$

$$\sum_{m \in E} x^v_{i,m} = \sum_{n \in S} x^v_{i,n} \qquad \forall v \in V,\ \forall i \in T \tag{6}$$

$$\sum_{m \in S} x^v_{m,\text{pick}_c} = x^v_{\text{pick}_c,\text{drop}_c} \qquad \forall v \in V,\ \forall c \in C \tag{7}$$

$$\text{arrive}_j \geq \text{leave}_i + \text{dis}\left(\text{loc}_i, \text{loc}_j\right) + M * \left(x^v_{i,j} - 1\right) \qquad \forall v \in V,\ \forall i \in S,\ \forall j \in T \tag{8}$$

$$\text{arrive}_j \leq \text{leave}_i + \text{dis}\left(\text{loc}_i, \text{loc}_j\right) + M * \left(1 - x^v_{i,j}\right) \qquad \forall v \in V,\ \forall i \in S,\ \forall j \in T \tag{9}$$

$$\text{arrive}_{\text{drop}_c} > \text{leave}_{\text{pick}_c} \qquad \forall v \in V, \forall c \in C \tag{10}$$

$$\text{start}_i \geq \text{arrive}_i \qquad \forall i \in P \cup D \tag{11}$$

$$\text{start}_i \geq \text{ready}_i \qquad \forall i \in QT \tag{12}$$

$$\text{start}_{\text{qt}(\text{twin}(c))} = \text{start}_{\text{qt}(c)} \qquad \forall c \in C \tag{13}$$

$$\text{leave}_i = \text{start}_i + \text{handle}_i \qquad \forall i \in P \cup D \tag{14}$$

$$\text{ready}_i = \text{start}_{\text{pre}_i} + \text{cycle} \qquad \forall i \in QT \tag{15}$$

$$\text{leave}_0 = 0 \tag{16}$$

$$\text{arrive}_i \geq 0 \qquad \forall i \in P \cup D \tag{17}$$

$$\text{leave}_i \geq 0 \qquad \forall i \in P \cup D \tag{18}$$

$$\text{start}_i \geq 0 \qquad \forall i \in P \cup D \tag{19}$$

$$\text{ready}_i \geq 0 \qquad \forall i \in QT \tag{20}$$

The objective in Equation 1 is to minimize the total idle time of all quay-side tasks, which is also the idle time of all QCs during the planning horizon. The constraints in Equations 2 and 3 assign a dummy start and a dummy end task to each AGV. Equations 4 and 5 ensure that each task is assigned once and only once. The constraint in Equation 6 is the flow balance constraint, and the one in Equation 7 ensures that the AGV picking up a container has to deliver it to the destination node. The constraints in Equations 8 and 9 define an AGV's arrival time when it is assigned a task. Its arrival time equals its leave time from the last task plus the travel time between these two tasks' locations. The constraint in Equation 10 defines that, for each container *i,* the drop-off task for that container cannot be earlier than the pick-up task for it. The constraint in Equation 11 represents that each task starts after the AGV's arrival. The constraint in Equation 12 ensures that, for those quay-side tasks, their actual start time would not be earlier than the AGV's ready time. The constraint in Equation 13 ensures that, for the quay-side tasks, the start times of the two containers in the same container group must be the same. Equation 14 defines that the leave time from a task equals its start time plus the handle time. Equation 15 ensures that two successive tasks served by the same QC must be set apart by at least the time required for the QC to perform all necessary movements. Equation 16 sets the leave time from Dummy Task 0. The constraints in Equations 17 through 20 are nonnegative constraints.

## Dispatch Rule

Because of the complexity of the problem, it is impossible to obtain an optimal solution by solving the MILP model. Therefore two heuristic methods are proposed to solve the problem. The first is a two-phased dispatch rule and the second a neighborhood search method.

In a QC's working schedule, a precedence relationship exists among those containers. A container group cannot start to be unloaded–loaded until all the container groups before it in the QC's working sequence have been discharged–loaded. The basic idea for this dispatch rule is to minimize the total lateness by assigning the most prioritized available containers to the AGVs whose delivery of them generates the least lateness overall. The names and definitions of the four indices involved in the rule are as follows:

- Penalty index ($P^{m,n}_{i,j}$) represents the QC's idle time when AGV $i$ and $j$ are dispatched to transport containers $m$ and $n$.

$$P^{m,n}_{i,j} = \text{start}_{\text{qt}(m)} - \text{ready}_{\text{qt}(m)} = \text{start}_{\text{qt}(n)} - \text{ready}_{\text{qt}(n)} \qquad m, n \in C,\ i,\ j \in V \tag{21}$$

- Waiting index ($W^{m,n}_{i,j}$) represents the two AGVs' waiting time if they arrive before the QC is ready to load–unload containers from–onto them. When the AGVs wait for a QC, it is likely to cause congestion under the crane, and their idling at one QC may result in causing another QC to wait for the AGVs' arrival. Therefore, the

**TABLE 1 Comparison Criteria in First Phase**

| Order | Index | Favored Value | Reason |
|---|---|---|---|
| 1 | $P_{i,j}^{m,n}$ | Small | QC's idle time is smaller. |
| 2 | $W_{i,j}^{m,n}$ | Small | AGV's idle time is smaller. |
| 3 | $A_{i,j}^{m,n}$ | Small | AGV's, QC's, or both idle times are smaller. |

reduction of AGVs' idle time is also helpful to the minimization of QC's idle time.

$$W_{i,j}^{m,n} = \max\left\{0, \max\left(\text{arrive}_{\text{qt}(m)}, \text{arrive}_{\text{qt}(n)}\right) - \text{ready}_{\text{qt}(m)}\right\} \qquad m, n \in C, i, j \in V \qquad (22)$$

• Arrival index ($A_{i,j}^{m,n}$) represents the gap between the arrival times of AGV *i* and *j* when they are dispatched to pick up or drop off containers *m* and *n*. The reason for comparing this index is the same as for the waiting time index.

$$A_{i,j}^{m,n} = \left|\text{arrive}_{\text{qt}(m)} - \text{arrive}_{\text{qt}(n)}\right| \qquad m, n \in C, i, j \in V \qquad (23)$$

• Layer index ($L^{m,n}$) represents a container group's order in the QC's working sequence. If a container group is the *i*th one in a QC's working sequence, then its *L* is *i*.

### *First Phase*

In the first phase, any two AGVs are combined for an available container group and each combination is called a comb. The task of the first phase is to determine which comb is the best one for a container group according to its indices' values. Table 1 listed the criteria in measuring a comb's indices' values. The order in the first column refers to the importance of the index. For example, if only one comb performs the best in the first index, then it is viewed as the best comb, and the first phase ends. Otherwise, the combs performing equally well in the first index are compared against the

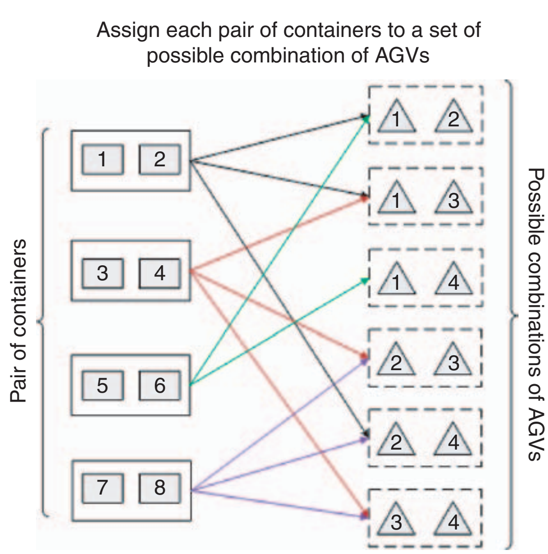


**FIGURE 4 Example of first phase.**

**TABLE 2 Comparison Criteria in Second Phase**

| Order | Index | Favored Value | Reason |
|---|---|---|---|
| 1 | $P_{i,j}^{m,n}$ | Large | QC's idle time must be equal to or larger than $P_{i,j}^{m,n}$ when it discharges–loads container *m* and *n*. Any other assignment must result in larger idle time. |
| 2 | $W_{i,j}^{m,n}$ | Small | AGV's idle time is smaller. |
| 3 | $A_{i,j}^{m,n}$ | Small | AGV's, QC's, or both idle times are smaller. |
| 4 | $L^{m,n}$ | Small | All containers behind *m* and *n* in QC's working sequence would be influenced if there is idling when QC discharges/loads *m* and *n*. |

second index. If two or more combs perform equally well on all three indices, then all of them are entered into the second phase. Smaller values are preferred for all three indices.

At the end of the first phase, at least one comb is associated with each container group. But overlap may exist in the AGV assignment among different container groups (Figure 4). This problem will be solved in the second phase.

### *Second Phase*

This phase is designed to solve the problem mentioned in the first phase by comparing the containers' priorities. The comparison criterion is illustrated in Table 2 in the same way as in Table 1. But in the second phase, the order of each index is not unchanged all the time. To explore more solutions, the orders of the four indices are changed and each set of different indices' orders is allowed to be a strategy. For example, Strategy P-W-A-L means that index $P_{i,j}^{m,n}$ is the most important one, $W_{i,j}^{m,n}$ is the second-most important one, and so on. Similarly, Strategy P-A-W-L means that index $P_{i,j}^{m,n}$ is the most important one, $A_{i,j}^{m,n}$ is the second important one, and so on. Therefore, in the second phase, the available container groups' priorities are measured in accordance with different strategies. Among all the solutions generated from strategies, the best one is taken as the final solution.

Now, the method with the QCs' working sequences will be introduced in Table 3. The two numbers in one cell represent the two containers discharged–loaded simultaneously by the QC. Assume that, in the last round of the assignment, AGVs 1 through 4 are dispatched to Containers 7, 8, 3, and 4, respectively. For AGV 1, its current position is $\text{loc}_{\text{drop}(7)}$, and the moment that it finishes dropping off Container 7 and is ready for the next task is $\text{leave}_{\text{drop}(7)} = \text{start}_{\text{pick}(7)} + h + \text{dis}(\text{loc}_{\text{pick}(7)}, \text{loc}_{\text{drop}(7)}) + h$. In the same way, the current positions of AGVs 1 through 4 and the times they finish delivering Containers 7, 8, 3, and 4 can be obtained.

**TABLE 3 Working Sequences for QC1 and QC2**

| Order | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| QC1 | 1, 2(U) | 3, 4(U) | 9, 10(L) | 13, 14(L) |
| QC2 | 5, 6(U) | 7, 8(U) | 11, 12(L) | 15, 16(L) |

NOTE: U represents that the container group will be unloaded by the QC from the vessel. L represents that the container group will be loaded by the QC to the vessel.

In the first phase, any two AGVs for container groups (9, 10) and (11, 12) are combined and the best combs for each of them are determined. From the criteria in the first phase, the best comb for (9, 10) is AGVs 3 and 4 and the best one for (11, 12) is AGVs 4 and 3. Next, the priorities of these two container groups are compared in the second phase by using Strategy P-W-A-L. Because $P_{3,4}^{9,10} = 5 > P_{4,3}^{11,12} = 3$, (9, 10) has higher priority than (11, 12) and AGVs 3 and 4 are dispatched to deliver Containers 9 and 10. As a result, the available container groups in the next round should be (11, 12) and (13, 14).

## Neighborhood Search

To obtain a better solution, another heuristic approach is proposed to solve the problem. The practical advantage of such a heuristic method is that it can solve optimization problems to near-optimal within an acceptable time. Generally, the approach starts with an initial solution and then searches for a better solution within its predefined neighborhood. On the basis of evaluation, each new neighbor could be either accepted or rejected. For such a method, the results largely lie in the design of neighborhoods and searching strategies. The neighborhood search method is a classic and effective heuristic algorithm. Although it has been widely applied to many vehicle routing problems, it has not been used in solving an AGV dispatching problem combined with tandem-lift QCs, to the best of the authors' knowledge. The proposed method does not reflect only the characteristics of this problem but also helps speed up the search process.

The main idea of the proposed neighborhood search is to first restrict the candidate solution within a small neighborhood and then generate more candidates by enlarging the neighborhood if there is no improvement within the neighborhood. A feasible solution can be viewed as a matrix of containers. The containers in a row represent the containers delivered by the same AGV, and the column where a container resides represents the order in which the container is delivered. The proposed neighborhood search is composed by two categories: intracolumn exchange and intercolumn exchange (Figure 5).

With a feasible solution, the intercolumn exchange is conducted by exchanging every container with another one within the same column. One can notice that, at any time, only limited containers require deliveries by AGVs, and the order in which they are delivered by different AGVs are quite close to each other. Thus, the intercolumn exchange is restricted to being executed only between two adjacent columns. Among all the new solutions, the promising ones are filtered, and more possible improvement is investigated by exchanging every container within the same column (intracolumn exchange). The filter criterion is higher at first. If no improvement occurs, the neighborhood is extended by lowering the filter criterion. The iteration process is illustrated in Table 4.

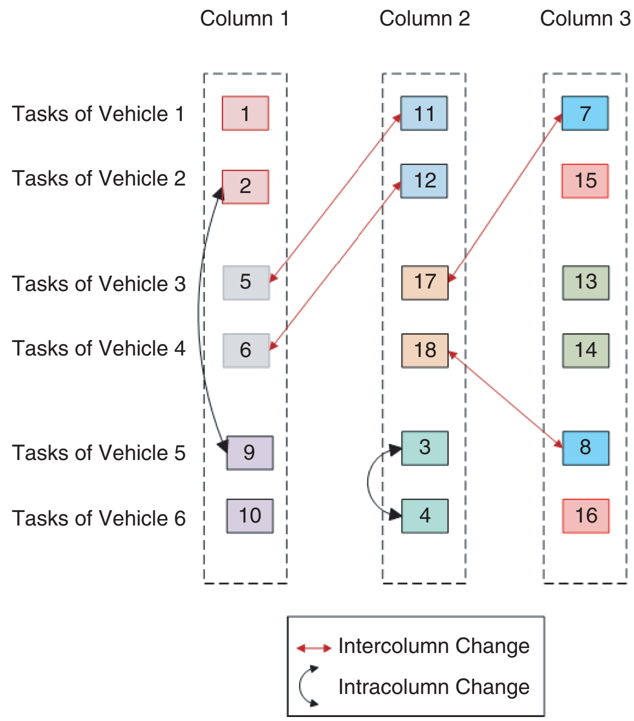


**FIGURE 5 Example of intracolumn and intercolumn exchanges.**

Moreover, during each step of iteration, the changes in every candidate solution are recorded and saved in a taboo list to prevent redoing of these changes. For each candidate, only the last 100 changes in the taboo list are saved with those before them being discarded. In addition, the initial solutions are chosen by the worst ones among the feasible solutions obtained from the priority rules introduced earlier. The whole flow of the neighborhood search is illustrated in Table 4.

**TABLE 4 Process of Neighborhood Search**

| Step | Description |
|---|---|
| Initial solution | Feasible solutions obtained from priority-rules. |
| Step 1 | Apply intercolumn neighborhood search between any two neighbored columns. Save all qualified solutions as candidates. |
| Step 2 | For the candidates from Step 1 |
| 2.1 | Select the higher-quality candidates. Perform the intracolumn exchange. If there is any improvement in objective value, the solutions are updated. |
| 2.2 | Otherwise, enlarge the search space by adding those lower-quality candidates from Step 1 and perform the intracolumn exchange to them. If there is still not any improvement, use the solutions from Step 2.1 for the next round of intercolumn exchanges. |
| Step 3 | End the whole process after a predefined number of intercolumn exchanges. |

## NUMERICAL EXPERIMENTS

In this part, the problem is solved with the three methods introduced earlier. On the basis of the problem size, the experiments can be divided into three categories: small (S), medium (M), and large (L). Each of them contains 10 test cases. The containers' storage plans are generated randomly in the C++ program. By considering the layout and the AGVs' speed variation in different terminals, the parameters are set on the basis of published papers (*3, 7, 20*):

- The average operation speed of a tandem-lift QC is about 60 moves per hour.
- The speed of the AGVs is about 6 m/s.
- The distance between adjacent cranes is 90 m.
- The number of AGVs is different in different scenarios.

To simplify the problem, all distances are set to be normalized at one time unit (measured in AGV travel time) for a trip between two adjacent working stations, and the cycle times of QCs are normalized at four time units. In each experiment, the storage position of each container on the ship and storage block, as well as the QCs' working schedule, are randomly generated.

The MILP model was formulated and solved by using optimization programming language and the commercial optimization solver CPLEX, version 12.1. The two heuristic methods were coded in C++. All computations were conducted on a personal computer with a 2.66-GHz–2.66-GHz Intel Core 2 Quad CPU on a Microsoft Windows platform and a 4.00-GB RAM.

Table 5 shows the computation results for the proposed three methods. They can be compared in the following two main respects: objective value and computation time.

### Objective Value

Optimal solutions can be found for all the small problems. For the medium and large ones, the CPLEX solver cannot obtain the optimal solution even after running 10 h, but the results from CPLEX provide a benchmark for the two heuristic methods. The results in Table 5 show that, except for only one case (16_4_5), the heuristic results are as good as or better than the best integer solutions found by CPLEX. In addition, the neighborhood search method always performs better than the dispatch rule, with one exception. The possible reasons for this performance may include that (*a*) the dispatch rule is myopic

**TABLE 5 Computational Results from Three Methods**

| | | MILP Model Solved by CPLEX | | Neighborhood Search | | | | Dispatch Rule | |
|---|---|---|---|---|---|---|---|---|---|
| Method | Case Number | BI | Time (s) | Obj1 | Time_B | Time_F | Gap1 | Obj2 | Gap2 |
| S | 16_4_1 | 36 | 420 | 36 | 7.26 | 133.86 | 0.00 | 36 | 0.00 |
| | 16_4_2 | 34 | 1,387 | 34 | 26.92 | 439.42 | 0.00 | 35 | 2.94 |
| | 16_4_3 | 30 | 609 | 30 | 15.44 | 197.81 | 0.00 | 30 | 0.00 |
| | 16_4_4 | 29 | 6,590 | 29 | 0.09 | 95.97 | 0.00 | 29 | 0.00 |
| | *16_4_5 | 36 | 8,792 | 37 | 115.06 | 481.5 | 2.78 | 36 | 0.00 |
| | 16_4_6 | 36 | 782 | 36 | 0.03 | 12.95 | 0.00 | 36 | 0.00 |
| | *16_4_7 | 33 | 2,151 | 33 | 96.757 | 466.247 | 0.00 | 34 | 3.03 |
| | 16_4_8 | 32 | 3,539 | 32 | 0.49 | 48.93 | 0.00 | 32 | 0.00 |
| | *16_4_9 | 32 | 1,263 | 32 | 100.58 | 460.22 | 0.00 | 32 | 0.00 |
| | 16_4_10 | 33 | 999 | 33 | 5.43 | 85.1 | 0.00 | 33 | 0.00 |
| M | 18_6_1 | 22 | — | 22 | 12.19 | 473.07 | 0.00 | 22 | 0.00 |
| | 18_6_2 | 19 | — | 19 | 16.74 | 467.97 | 0.00 | 19 | 0.00 |
| | 18_6_3 | 26 | — | 24 | 9.33 | 470.12 | –7.69 | 24 | –7.69 |
| | 18_6_4 | 25 | — | 20 | 25.83 | 833.99 | –20.00 | 21 | –16.00 |
| | 18_6_5 | 28 | — | 26 | 108.42 | 1,030.7 | –7.14 | 28 | 0.00 |
| | 18_6_6 | 24 | — | 23 | 17.64 | 319.89 | –4.17 | 24 | 0.00 |
| | 18_6_7 | 20 | — | 20 | 9.24 | 484.73 | 0.00 | 20 | 0.00 |
| | 18_6_8 | 37 | — | 34 | 17.96 | 755.49 | –8.11 | 35 | –5.41 |
| | 18_6_9 | 28 | — | 26 | 14.47 | 805.33 | –7.14 | 26 | –7.14 |
| | 18_6_10 | 20 | — | 20 | 21.15 | 393.1 | 0.00 | 20 | 0.00 |
| L | 24_6_1 | 54 | — | 38 | 201.62 | 1,755.11 | –29.63 | 38 | –29.63 |
| | 24_6_2 | 61 | — | 41 | 802 | 2,369.1 | –32.79 | 47 | –22.95 |
| | *24_6_3 | 55 | — | 39 | 1,377.9 | 3,039.8 | –29.09 | 39 | –29.09 |
| | *24_6_4 | 190 | — | 54 | 2,039.92 | 3,732.48 | –71.58 | 54 | –71.58 |
| | 24_6_5 | 59 | — | 41 | 300.51 | 2,127.74 | –30.51 | 47 | –20.34 |
| | *24_6_6 | 63 | — | 41 | 685.39 | 2,486.98 | –34.92 | 45 | –28.57 |
| | 24_6_7 | 80 | — | 34 | 384.84 | 2,745.89 | –57.50 | 34 | –57.50 |
| | 24_6_8 | 46 | — | 39 | 53.32 | 1,714.79 | –15.22 | 47 | 2.17 |
| | 24_6_9 | 51 | — | 43 | 892.52 | 3,802.41 | –15.69 | 50 | –1.96 |
| | 24_6_10 | 47 | — | 38 | 222.41 | 1,985.34 | –19.15 | 42 | –10.64 |

NOTE: BI represents the best integer solution found by CPLEX. For the M and L test cases, the "—" in the column for time represents the computation time as 10 h. The computation time of dispatch rule is only a few seconds, so it is not listed in the table. Time_B is the time when the neighborhood search finds the best solution. Time_F is the time when the neighborhood search finishes the whole search process. Obj1 = objective value obtained from neighborhood search; Obj2 = objective value obtained from dispatch rule; Gap1 = 100% * (Obj1 – BI)/BI; Gap2 = 100% * (Obj2 – BI)/BI.

and only makes the best decision on the basis of the current situation, and (*b*) when one strategy is executed in the priority measurement, all the containers are measured with it without change. The dispatch rule ignores the possibility that, for some measurements, a different strategy could generate a better result for the whole problem. However, comparison of all the priority strategies in every measurement is impossible within reasonable computation time. Compared with the dispatch rule, the neighborhood search method overcomes these disadvantages by exchanging AGVs' working sequences according to several principles. The reason for this is that the generation of new solutions equals the application of different priority strategies or the knowledge of future events in decision making.

### Computation Time

Obviously, MILP is the most time-consuming method, and the priority-based dispatch rule is the fastest method. The computation time of neighborhood search is much shorter than that of the MILP but longer than the dispatch rule. By a comparison of Time_B and Time_F in the neighborhood search method, some redundancy in computation time can be uncovered, but solutions of higher quality must be obtained. Sometimes changing one container in an AGV's working sequence cannot improve the solution. When that change is combined with another change, the objective value would be improved. If the process is ended too early or the solutions are filtered too strictly, those solutions with the possibility of improvement would be lost. However, if too many solutions are kept at every step of iteration, the computation time would dramatically increase. To keep a balance between computation time and the objective value, the whole search process is ended when no improvement occurs in the objective value after three consecutive iterations. In up to 20% of experiments (which have been marked by an asterisk in Table 5), this process finds better solutions by continuing to search after no improvement in two iterations.

## CONCLUSIONS AND DISCUSSION OF RESULTS

Both of the QCs and AGVs introduced in this paper are advanced equipment designed to increase the productivity in container terminals. However, neither of them can achieve this purpose without efficient operation control. The AGV dispatching problem, combined with the tandem-lift QC, is still a relatively new topic in this area, and little research has been conducted, especially from the aspect of mathematical modeling.

In this paper, the problem was first formulated by an MILP model. Because of the complexity of solving such a model, two heuristic methods were proposed for practical purposes. Numerical experiments showed that both of the heuristic methods can obtain good solutions within a very short time, and the neighborhood search method generally performs better in relation to the objective value. The authors' future work on this problem will focus on improvement of the neighborhood search method as well as on heuristic dispatching rules for the daily operation in reality.

## ACKNOWLEDGMENT

The authors thank the University Transportation Center for Mobility at the Texas Transportation Institute for supporting this research.

## REFERENCES


1. Jeon, Y.H., and Y.M. Park. Status and Visions of Automated Container Terminals. *Proc., 15th International Offshore and Polar Engineering Conference,* Seoul, Korea, June 19–24, 2005.
2. Liu, C.I., H. Jula, K. Vukadinovic, and P.A. Ioannou. Comparing Different Technologies for Containers Movement in Marine Container Terminals. *Proc. 3rd IEEE International Conference on Intelligent Transportation Systems,* Dearborn, Mass., Oct. 1–3, 2000, pp. 488–493.
3. Lind, D., J.K. Hsieh, and M.A. Jordan. Tandem-40 Dockside Container Cranes and Their Impact on Terminals. Presented at ASCE Ports 2007 Conference, San Diego, Calif., 2007.
4. Vis, I.F.A., R. de Koster, K.J. Roodbergen, and L.W.P. Peeters. Determination of the Number of Automated Guided Vehicles Required at a Semi-Automated Container Terminal. *Journal of the Operational Research Society,* Vol. 52, No. 4, 2001, pp. 409–417.
5. Liu, C.I., H. Jula, K. Vukadinovic, and P.A. Ioannou. Automated Guided Vehicle System for Two Container Yard Layouts. *Transportation Research Part C,* Vol. 12, 2004, pp. 349–368.
6. Duinkerken, M.B., and J.A. Ottjes. A Simulation Model for Automated Container Terminals. *Proc., Business and Industry Simulation Symposium,* Washington, D.C., 2000.
7. Vis, I.F.A., and I. Harika. Comparison of Vehicle Types at an Automated Container Terminal. *OR Spectrum,* Vol. 26, 2004, pp. 117–143.
8. Kim, K.H., and J.W. Bae. A Look-Ahead Dispatching Method for Automated Guided Vehicles in Automated Port Container Terminals. *Transportation Science,* Vol. 38, No. 2, 2004, pp. 224–234.
9. Choi, E., and D.W. Tcha. A Column Generation Approach to the Heterogeneous Fleet Vehicle Routing Problem. *Computers and Operation Research,* Vol. 34, 2007, pp. 2080–2095.
10. Meersmans, P.J.M., and A.P.M. Wagelmans. *Effective Algorithms for Integrated Scheduling of Handling Equipment at Automated Container Terminals.* ERS-2001-36-LIS. ERIM Report Series Research in Management, Erasmus Research Institute of Management, Rotterdam, Netherlands, 2001.
11. Lim, J.K., K.H. Kim, K. Yoshimoto, J.H. Lee, and T.A. Takahashi. Dispatching Method for Automated Guided Vehicles by Using a Bidding Concept. *OR Spectrum,* Vol. 25, 2003, pp. 25–44.
12. Grunow, M., H.O. Gunther, and M. Lehmann. Dispatching Multi-Load AGVs in Highly Automated Seaport Container Terminals. Container Terminals and Automated Transport Systems, Part I. *Proc., International Symposium on OR and Its Applications 2005,* 2005, pp. 231–255.
13. Grunow, M., H.O. Gunther, and M. Lehmann Strategies for Dispatching AGVs at Automated Seaport Container Terminals. Container Terminals and Automated Transport Systems, Part II. *Container Terminals and Cargo Systems Design, Operations Management, and Logistics Control Issues,* Springer, Berlin, 2007, pp. 155–178.
14. Briskorn, D., A. Drexl, and S. Hartmann. Inventory-Based Dispatching of Automated Guided Vehicles on Container Terminals. *OR Spectrum,* 2006, pp. 611–630.
15. Homayouni, S.M., T.S. Hong, N. Ismail, and M.K.A. Ariffin. Optimization of Integrated Scheduling of Quay Cranes and Automated Guided Vehicles Using Simulated Annealing Algorithm. *Proc., 2011 International Conference on Industrial Engineering and Operations Management,* Kuala Lumpur, Malaysia, Jan. 22–24, 2011.
16. Bae, H.Y., R. Cheo, T. Park, and K.R. Ryu. Comparison of Operations of AGVs and ALVs in a Container Terminal. *Journal of Intelligent Manufacturing,* Vol. 22, No. 3, June 2011.
17. Lin, Y.J., and S.L. Chao. Evaluating Advanced Quay Cranes in Container Terminals. *Transportation Research Part E,* Vol. 47, 2011, pp. 432–445.
18. Dondo, R., and J. Cerda. A Reactive MILP Approach to the Multi-Depot Heterogeneous Fleet Vehicle Routing Problem with Time Windows. *International Transactions in Operational Research,* Vol. 13, 2006, pp. 441–459.
19. Karraman, S., and E. Frazzolo. Vehicle Routing Problem with Metric Temporal Logic Specifications. Presented at 47th IEEE Conference on Decision and Control, Cancun, Mexico, Dec. 9–11, 2008.
20. Stahlbock, R., and S. Voß. Operations Research at Container Terminals: A Literature Update. *OR Spectrum,* Vol. 30, 2008, pp. 1–52.


*The Intermodal Freight Terminal Design and Operations Committee peer-reviewed this paper.*