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

Yao Xing
Zachry Department of Civil Engineering
Texas A&M University
Email: Y-Xing@ttimail.tamu.edu

Kai Yin
Zachry Department of Civil Engineering
Texas A&M University
Email: k-yin@ttimail.tamu.edu

Luca Quadrifoglio
Zachry Department of Civil Engineering
Texas A&M University
Email: quadrifo@tamu.edu
(Corresponding author)

Bruce X. Wang
Zachry Department of Civil Engineering
Texas A&M University
Email: bwang@tamu.edu

Submission date: August 1st, 2011
Number of words: 5240
Number of figures and tables: (7×250 = 1750)
Total: 6990
ABSTRACT

New quay cranes (QC) have been designed to increase the productivity in terminals by lifting more containers at once. However, their productivity also relies on the efficient cooperation with the vehicles carrying containers. This paper investigates the synchronization scheduling problem between the automated guided vehicles (AGVs) and these new QCs. The problem is formulated as a MILP model. Due to its complexity, a heuristic dispatch rule is proposed for practical purposes. Then in order to balance the computation time and the quality of solution, a neighborhood search method is designed by investigating the working sequences of AGVs. Numerical experiments show that both heuristics obtain good solutions within very short time and the latter performs generally better in terms of the objective value.
1. INTRODUCTION

The steadily increasing freight volume in addition to the development of jumbo container ships put a lot of pressure on freight transportation and 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 the efficient operation and management are becoming more attractive compared to physical enlargement of the terminal size. In recent years, automated Container Terminals (ACTs) (as shown in Figure 1) have been adopted in some of the main and busiest terminals, such as Hamburg, Singapore and Rotterdam. ACT refers to the unmanned terminal controlled by advanced equipment and high-level information network. Compared to traditional terminals, they have more advantages in reducing labor cost and operation cost (1).

![Figure 1: Layout of the Container Terminal Altenwerder, Hamburg, German. (source: http://www.hhla.de/de/Geschaeftsfelder/HHLA_Container/Altenwerder_(CTA)/Daten_und_Fakten.jsp)](image)

The operation process in an ACT is similar to that in the conventional container terminals. For the unload process, containers are lifted by the Quay Crane (QC) and then delivered by AGV to the storage area and positioned by yard cranes to specific storage blocks. The load process is operated in the reverse order. The difference between the conventional terminals and ACTs lies in the equipment and the operation systems. In an ACT, AGVs take the place of trucks and become the main horizontal transporter. A study shows that the application of AGVs could double the throughput of the terminal (2). Besides, they can reduce the labor cost as well as the emission since they do not need drivers and are powered by electricity instead of fossil energy (Figure 2 (a)). One other important handling equipment is the QC whose working speed greatly influences the efficiency and throughput of the whole terminal. New QCs are designed for faster loading/unloading operations to meet the demands of mega-vessels and the tandem/twin lift QC is one of them. Compared to conventional
single trolley cranes, they can double the productivity by lifting four adjacent 20ft or
two 40ft containers at once (Figure 2 (b)). For example, in the unloading process, a
tandem lift QC firstly moves its spreader for reaching the containers on the ship and
then hoisted two adjacent 40ft containers from the ship and then amount those two
containers onto two vehicles simultaneously.

Besides the adoption of new advanced pieces of equipment, synchronization and
cooperation of the operations is a more important and difficult issue within the
terminals. For example, Lind et al. (3) pointed out that “non-crane” delay would reduce
the tandem lift QC’s efficiency up to 50% and the “wasted” waiting time of the AGVs
is a main factor. In an ideal situation, the AGV arrives at the QC just at the moment
when the QC is ready to put containers on it or lift the container from it. If the AGV
arrives late, then the QC has to wait which results in the decrease in its working
productivity. If the AGV arrives early, then it has to wait for the QC, too. To minimize
those idle times, the dispatch sequences of all AGVs should be optimized based on the
working schedule of QCs. Such a problem can be classified as the classical vehicle
routing problem (VRP). When it comes to serving the tandem lift cranes, the situation
becomes more complex. The QC cannot start to mount/dismount containers onto/from
AGVs until both of the two AGVs are readily present there. However, the two
containers hoisted simultaneously by a QC may be stored in different blocks in the
yard area. As a result, the two AGVs dispatched to serve a tandem lift QC may travel
from different yard blocks to the same QC during the loading process or travel from
the same QC to different yard blocks during the unloading process. If we simply fix
two AGVs as a group serving a specific crane, the problem would degrade to the
traditional AGV’s dispatching problem in terminals. However, it would result in less
flexibility in the dispatch and reduce the efficiency of AGVs. Nowadays, the AGV
dispatching in container terminals follows like some simple rules, like
First-Come-First-Served, nearest vehicle rule, and so on. In some cases, especially
when there are unexpected events or accidents happened, the vehicles’ dispatchment
follows the experienced operator’s commends.

The rest of this paper is organized as follows: the next section reviewed the
existing literatures of AGV’s dispatching problem in container terminals. Then the
problem statement and the mathematical model of the problem are introduced. Three

Figure 2: (a) AGV (b) tandem lift QC
solution methods are then described and numerical experiments are executed, followed by the conclusions.

2. LITERATURE REVIEW

The operation and control problem of AGVs in the container terminals have been discussed very comprehensively from different aspects. The objective of most studies is to maximize the terminal’s throughput or minimize the ship’s turnaround time through the optimization of AGVs’ operation.

Qiu and Hsu (4) presented a bi-directional path layout and an algorithm for routing AGVs without conflicts with minimization of the space requirement of the layout. Vis al. (5) developed a minimum flow algorithm to determine the number of AGVs required at a semi-automated container terminal. Liu al. (6) discussed relationship between the number of AGVs and the terminal’s layout with simulations. Duinkerken and Ottjes (7) developed a simulation to determine the sensitivity concerning a number of parameters like number of AGVs, maximum AGV speed, crane capacity and stack capacity. Vis and Harika (8) pointed out that the design of container terminal and technical aspects of quay cranes impact the number of vehicles required and the choice for a certain type of equipment.

The dispatching and routing problem of AGVs is closely related to the vehicle routing problem (VRP) and can be formulated by an MIP model. Kim and Bae (9) suggest a network-based MIP model for AGV dispatching and provide a heuristic algorithm to minimize the total idle time of a quay crane resulting from the late arrivals of AGVs. Choi and Tcha (10) proposed an approach based on column generation to solve the heterogeneous fleet VRP. The feasible columns are generated by emulating dynamic programming schemes and the experiment with the benchmark tests confirms that the proposed approach outperforms all the existing algorithms.

Meersmans and Wagelmans (11) considered an integrated problem containing the scheduling of different equipment at automated terminals. They presented a branch and bound algorithm and a heuristic beam search algorithm in order to minimize the makespan of their schedule and near optimal solutions are obtained in a reasonable time. Lim et al. (12) proposed an auction-based assignment algorithm in the sense that the dispatching decisions are made through communication among related vehicles and machines for matching multiple tasks with multiple vehicles. Their method takes into account the future events and its performance is compared through a simulation study. Grunow et al. (13, 14) proposed a flexible priority rule for dispatching multi-load AGVs. And an MIP formulation is developed for the optimal solution in small problem instances. A hybrid approach combining the MIP model with heuristic is also proposed for real application. The numerical experiments show that the developed pattern-based off-line heuristic outperforms conventional on-line heuristics. Briskorn et al. (15) solved the assignment of jobs to AGVs both with a greedy priority rule based heuristic and with an exact algorithm. Then they formulated the assignment without due times and solve it based on a rough analogy to inventory
management avoiding the estimates of driving times completion times, due times and tardiness. Homayouni et al. (17) solved the integrated scheduling of quay crane and AGV by using simulated annealing (SA) algorithm. They investigated the effects of initial temperature and the number of trials on the algorithm and compared the results obtained from SA algorithm with ones from the MILP mode.

In addition, the deadlock prediction and conflict prevention is another complex issue in the AGVs’ control. Leong (17) integrated his AGV dispatching scheme model with the deadlock prediction and avoidance algorithm in previous study and compared the performance of current scheme and the new proposed one with simulation model. Kim et al. (18) proposed a graphical representation method for reserving grid-blocks in advance to prevent the deadlocks. Their approach can guarantee deadlock-free schedules for AGVs to cross the same area at the same time.

However, the QCs discussed in all those papers are limited to the conventional single trolley QCs, while very few papers investigated the application and performance of tandem lift QCs in container terminals and even fewer papers studied the AGVs’ dispatching rules serving them. From the modeling perspective, operations of tandem lift QCs are much more challenging than that of conventional QCs. The existing papers about the tandem lift QC are mainly limited to the introduction of its configurations and productivities (3). Bae et al. (19) compared the performance of different vehicles in a cross-lane path layout combining with the QCs of various types by using simulation software. Lin and Chao (20) developed a two-phased 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 fuzzy analytic hierarchy process to compare alternatives.

3. MATHEMATICAL MODELS AND METHODOLOGY

3.1 Problem Statement

In this problem, a QC lifts two 40ft containers or 20ft containers one time and an AGV’s capacity is one 40ft container or two 20ft containers. Therefore, two 20ft containers or one 40ft container is viewed as one unit in this paper. In the rest of the paper, a container refers to such a container unit. And for any two containers hoisted by a QC simultaneously, we say they are in the same container group.

Objective: For AGVs, they only conduct two kinds of tasks – pick up a container from its original position and drop it off at its destination. For any task conducted at the QC’ side, it has a pre-planned start time according to the QC’s working schedule. If the AGV arrives later than it, then the QC has to wait for the AGV and such lateness reduces the QC’s productivity as a result. 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, we assume that:

1. The work schedule of each QC (loading/unloading sequences of container pairs)
and the containers’ storage plan are already known, and it is always true in the terminals’ operation.

2. All the AGVs are homogeneous and they run at the same speed all the time. Although not perfectly accurate, we simplify our model by disregarding acceleration/deceleration when they turn or approach the cranes.

3. Finally, we assume that there is always an available yard crane serving the AGVs.

![Figure 3: The cross lane layout.](image)

### 3.2 Mixed Integer Linear Programming Formulation

A Mixed Integer Linear Programming (MILP) formulation is proposed in this paper to describe the AGVs’ dispatch problem. Since it is similar to the classic vehicle routing problem, most of proposed MILP models share the similar structures \((21, 22)\). In this problem, besides the known flow constraints and time constraints, additional time constraints are needed for QCs’ operation since 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 path are the arcs in this network. The notations needed for the formulation are listed in the following:

- \( V \) is the set of all AGVs pooled to server the QCs;
- \( C \) is the set of all containers needed to be discharged or loaded during the planning horizon;
- \( Q \) is the set of all the QCs discharge or load containers;
- \( Y \) is the set of all the YCs in the stack area;
- \( \text{pick}_c \) AGV picks up container \( c \) at a quay crane or yard crane;
- \( \text{drop}_c \) AGV drops off container \( c \) at a quay crane or yard crane;
\( P = \{ \text{pick}_1, \text{pick}_2, \ldots, \text{pick}_c \}, c \in C \) is the set of all pick-up tasks;

\( D = \{ \text{drop}_1, \text{drop}_2, \ldots, \text{drop}_c \}, c \in C \) is the set of all drop-off tasks;

\( T = P \cup D \) is the set of all tasks, including pick-up and drop-off tasks;

\( S = T \cup 0 \) where 0 is the dummy start task for each AGV;

\( E = T \cup e \) where e is the dummy end task for each AGV;

\[ q_t(c) = \begin{cases} \text{pick}_c, & \text{if } \text{loc}_{\text{pick}} \in Q \\ \text{drop}_c, & \text{if } \text{loc}_{\text{drop}} \in Q \end{cases} \]

is the quay-side task of container c;

\[ y_t(c) = \begin{cases} \text{pick}_c, & \text{if } \text{loc}_{\text{pick}} \in Q \\ \text{drop}_c, & \text{if } \text{loc}_{\text{drop}} \in Q \end{cases} \]

is the yard-side task of container c;

\[ QT = \{ q_t(1), q_t(2), \ldots, q_t(c), \ldots \}, c \in C \] is the set of all quay-side tasks

\[ YT = \{ y_t(1), y_t(2), \ldots, y_t(c), \ldots \}, c \in C \] is the set of all yard-side tasks

\( \text{twin}(c) \) is the container which is hoisted simultaneously with container \( c, c \in C \);

\( \text{pre}_{q_t(c)} \) is the \( q_t(c) \)'s predecessor in a QC’s working sequence, \( c \in C \)

**Parameters**

\( \text{dis}(i, j) \) is the distance an AGV needs traveling from node \( i \) to node \( j \), \( i, j \in N \)

\( h \) is the time a crane needs to amount or dismount a container onto or from an AGV;

\( \text{cycle} \) is the interval in the QC’s working sequence, which is decided by it working speed;

**Variables**

\( x_{i,j}^v \) is the binary variable equals to 1 if AGV \( v \) is dispatched to complete the task \( j \) immediately after completing task \( i \), \( i \in S, j \in E \)

\( \text{start}_i \) is the start time of task \( i \), \( i \in S \)

\( \text{arrive}_i \) is the time when an AGV arrives at the node where task \( i \) is, \( i \in S \);

\( \text{leave}_i \) is the time when an AGV leaves after the completion of task \( i \), \( i \in S \);

\( \text{ready}_i \) is the ready time of a quay-side task \( i \), \( i \in QT \)

**MILP Formulation:**
Objective: $\min \sum_{i \in QT} start_i - ready_i$ (1)

Subject to:

1. $\sum_{v \in V} x_{0,v} = 1$, $\forall v \in V$ (2)
2. $\sum_{v \in V} x_{i,v} = 1$, $\forall v \in V$ (3)
3. $\sum_{v \in V, i \in T} x_{i,v} = 1$, $\forall v \in V, \forall i \in T$ (4)
4. $\sum_{v \in V, i \in T} x_{i,v} = 1$, $\forall v \in V, \forall i \in T$ (5)
5. $\sum_{m \in S} x_{i,m} = \sum_{n \in S} x_{i,n}$, $\forall v \in V, \forall i \in T$ (6)
6. $\sum_{m \in S} x_{m,\text{pick}} = x_{\text{pick},\text{drop}}$, $\forall v \in V, \forall c \in C$ (7)

$\text{arrive}_j = \text{leave}_i + \text{dis}(\text{loc}_i, \text{loc}_j) + M \cdot (x_{i,j} - 1)$, $\forall v \in V, \forall i \in S, \forall j \in T$ (8)

$\text{arrive}_j = \text{leave}_i + \text{dis}(\text{loc}_i, \text{loc}_j) + M \cdot (1 - x_{i,j})$, $\forall v \in V, \forall i \in S, \forall j \in T$ (9)

$\text{arrive}_{\text{drop}} > \text{leave}_{\text{pick}}$, $\forall v \in V, \forall c \in C$ (10)

$\text{start}_i \geq \text{arrive}_i$, $\forall i \in P \cup D$ (11)

$\text{start}_i \geq \text{ready}_i$, $\forall i \in QT$ (12)

$\text{start}_{q(twinc)} = \text{start}_{q(c)}$, $\forall c \in C$ (13)

$\text{leave}_i = \text{start}_i + \text{handle}_i$, $\forall i \in P \cup D$ (14)

$\text{ready}_i = \text{start}_{\text{preq}} + \text{cycle}$, $\forall i \in QT$ (15)

$\text{leave}_0 = 0$ (16)

$\text{arrive}_i \geq 0$, $\forall i \in P \cup D$ (17)

$\text{leave}_i \geq 0$, $\forall i \in P \cup D$ (18)

$\text{start}_i \geq 0$, $\forall i \in P \cup D$ (19)

$\text{ready}_i \geq 0$, $\forall i \in QT$ (20)
The objective (1) is to minimize the total idle time of all quay-side tasks, which is also the idle time of all QC's during the planning horizon. Constraints (2) and (3) assign a dummy start and end task to each AGV. Equation (4) and (5) ensure that each task is assigned once and only once. Constraint (6) is flow balance constraint and (7) ensures that the AGV which picks up a container has to deliver it to the destination node. Constraint (8) and (9) define an AGV’s arrival time when it is assigned to a task. It equals to its leave time from the last task plus the travel time between these two tasks’ locations. Constraint (10) defines that for each container i, the drop off task of that container cannot be earlier than the pick-up task of it. Constraint (11) represents the each task starts after the AGV’s arrival. Constraint (12) ensures that for those quay-side tasks, their actual start time would not be earlier than its ready time. For those yard-side tasks, because we assume there is always a yard crane waiting to serve the AGV, there is no need to add such constraint to them. Constraint (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 the leave time from a task equals to 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. Constraints (17)-(20) are non-negative constraints.

Due to the complexity of the problem, it is impossible to obtain the optimal solution by solving the MILP model. So next, we propose two heuristic methods – a two-phased Dispatch rule and neighborhood search.

### 3.3 Dispatch Rule

Due to the computational difficulties of solving MILP model, two heuristic methods are proposed to solve the problem. They are designed to look for a near-optimal solution within reasonable computation time. The first one is a two-phased dispatch rule and the second one is neighborhood search method.

In a QC’s working schedule, there is precedence relationship among those containers. A container group cannot start to be discharged/loaded until all the container groups before it in the QC’s working sequence has been discharged/loaded. The basic idea of this dispatch rule is to minimize the total lateness by assigning the most “prioritized” available containers to the AGVs whose delivery of them generates least lateness as a whole. Before the illustration of the rule, we listed the definitions and meanings of the four indices involved:

- **Penalty index** ($P_{i,j}^{m,n}$) represents the QC’s idle time when AGV $i$ and $j$ are dispatched to transport container $m$ and $n$.

\[ P_{i,j}^{m,n} = \text{start}_{q(m)} - \text{ready}_{q(m)} = \text{start}_{q(n)} - \text{ready}_{q(n)}, m,n \in C, i,j \in V \]  

(21)

- **Waiting index** ($W_{i,j}^{m,n}$) represents the two AGVs’ waiting time if they arrive before the QC is ready to dismount/amount containers from/onto them. When
the AGVs wait for a QC, it is likely to cause the congestion under the crane and their idling at one QC may result in that another QC has to wait for the AGVs’ arrival. Therefore, the reduction of AGVs’ idle time is also helpful to the minimization of QC’s idle time.

\[ W_{i,j}^{m,n} = \max\{ 0, \max(\text{arrive}_{q(m)},\text{arrive}_{q(n)}) - \text{ready}_{q(m)} \}, m,n \in C, i,j \in V \] (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 container \( m \) and \( n \). The reason for comparing this index is the same as \( W \).

\[ A_{i,j}^{m,n} = |\text{arrive}_{q(m)} - \text{arrive}_{q(n)}|, m,n \in C, i,j \in V \] (23)

- Layer index \( L_{i,n}^{m} \) 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 find out which comb is the best one for a container group according to its indices’ values. Table 1 listed the criterion in measuring a comb’s indices’ values. The order in the first column refers to the importance of the index. For example, if there is only one comb performs the best in the 1st index, then it is viewed as the best comb for that two containers and the first phase ends. Otherwise, the combs performing the equally best in the first index are compared according to the second index. If there are more than one comb perform the equally best on all the three indices, then all of them are saved and entered into the second phase. The “small” in the second column represents that the comb with smaller value of that index is favored in the first phase and the reason is listed in the third column.

Table 1 Comparison criterion in the first phase

<table>
<thead>
<tr>
<th>Order</th>
<th>Index</th>
<th>F_value</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( P_{i,j}^{m,n} )</td>
<td>small</td>
<td>The QC’s idle time is smaller.</td>
</tr>
<tr>
<td>2</td>
<td>( W_{i,j}^{m,n} )</td>
<td>small</td>
<td>The AGVs’ idle time is smaller.</td>
</tr>
<tr>
<td>3</td>
<td>( A_{i,j}^{m,n} )</td>
<td>small</td>
<td>The AGV’s and/or QC’s idle time is smaller.</td>
</tr>
</tbody>
</table>

At the end of the first phase, there is at least one comb associated to each
container group. But there may be overlap in the AGVs assignment among different container groups (see Figure 3 for an example). 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 the 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. For exploring more solutions, we change the orders of these four indices and let each set of different indices’ orders 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 important one, and so on. 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, the available container groups’ priorities are measured according to different strategies in the second phase. Among all the solutions generated from strategies, the best one is outputted as the final solution.

Table 2 Comparison criterion in the second phase

<table>
<thead>
<tr>
<th>Order</th>
<th>Index</th>
<th>F_value</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$P_{i,j}^{m,n}$</td>
<td>Large</td>
<td>The 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</td>
</tr>
</tbody>
</table>
larger idle time.

2 \[ W_{i,j}^{m,n} \] small The AGVs’ idle time is smaller.

3 \[ A_{i,j}^{m,n} \] small The AGV’s and/or QC’s idle time is smaller.

4 \[ L_{m,n} \] small All the containers behind \( m \) and \( n \) in the QC’s working sequence would be influenced if there is idling when QC discharges/loads \( m \) and \( n \).

Now we will introduce the method with the QCs’ working sequences 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 assignment, AGV1 is dispatched to container 7, 8, 3 and 4, respectively. For AGV1, its current position is \( loc_{drop(7)} \) and the time moment that it finishes dropping off the container 7 and ready for the next task is

\[
leave_{drop(7)} = start_{pick(7)} + h + dis(loc_{pick(7)}, loc_{drop(7)}) + h.
\]

In the same way, we can get the current positions of AGV 1-4 and the times they finish delivering container 7, 8, 3, and 4.

In the first phase, we combine any two AGVs for container group \((9, 10)\) and \((11, 12)\) and find out the best combs for each of them. According to the criterion in the first phase, the best comb for \((9, 10)\) is AGV 3 and 4 and the best one for \((11, 12)\) is AGV 4 and 3. To solve the AGV overlap in the two combs, we compare the priorities of these two container groups in the second phase. Assume that they are first compared using strategy P-W-A-L. Because \( P_{3,4}^{9,10} = 5 > P_{4,3}^{11,12} = 3 \), container group \((9,10)\) has higher priority than \((11, 12)\), AGV 3 and 4 are dispatched to delivery container 9 and 10. As a result, the available container groups in the next round should be \((11, 12)\) and \((13, 14)\).

Table 3 The working sequences for QC1 and QC2.

<table>
<thead>
<tr>
<th>Order</th>
<th>QC1</th>
<th>QC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1,2(U)</td>
<td>5,6(U)</td>
</tr>
<tr>
<td>2</td>
<td>3,4(U)</td>
<td>7,8(U)</td>
</tr>
<tr>
<td>3</td>
<td>9,10(L)</td>
<td>11,12(L)</td>
</tr>
<tr>
<td>4</td>
<td>13,14(L)</td>
<td>15,16(L)</td>
</tr>
</tbody>
</table>

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

3.4 Neighborhood Search

The studied topic can be classified as one kind of combinatorial optimization problems. Due to the complexity of these problems, many optimization methods might fail to be either effective or efficient. During the last decades, metaheuristics...
have been recognized as one of the most practical approaches to tackle these complex problems. A neighborhood search method is proposed. Although it has been widely applied to many VRP related problems, it has not been used in solving AGV’s dispatching problem combined with tandem lift QCs, to the best of our knowledge. The proposed method does not only reflect the characteristics of this problem, but also helps speeding up the search process.

The practical advantage of such heuristic method is that it can obtain good results to many real optimization problems. However, the optimal solution cannot be always guaranteed. Generally speaking, the approach starts by an initial set of solutions, and then searches their neighborhoods by selecting and evaluating a candidate. Based on the evaluation, each candidate can be either accepted or rejected. Then the algorithm starts from these selected candidates and repeats the previous steps. For such method, the results largely lie in the design of neighborhoods and searching strategies. In this study, similar principles of metaheuristics are used to design an efficient algorithm to find good solutions in short time.

The main idea of the proposed neighborhood search is to first restrict the candidate solution within a small neighborhood set and then generate more candidates by enlarging the neighborhood if there is no improvement within this neighborhood. Let us consider again the objective. In order to minimize the total operation time, one has to determine the order of containers taken by each AGV. Hence, it is natural to define the neighbor of a solution based on the order of containers for each vehicle. As another dimension is the order of tasks for each vehicle, a candidate feasible solution can be viewed as a matrix with columns and rows, representing the same order of tasks for all AGVs and all (containers) tasks for each vehicle, respectively. The proposed neighborhood search of a candidate is composed by two categories: intra-column exchange and inter-column exchange. Basic ideas are illustrated for the example in Figure 4. The number in each column represents the number of containers, and the same color for two containers indicates that they are operated in pairs by a QC.

As shown in Figure 4, we exchange each container’s position with all rest tasks within the same column when performing the intra-column exchange. For each search step of iteration, the neighbors of one candidate solution must have only two positions different from this candidate in order to reduce the size of neighbor. While in inter-column exchange, we exchange a pair of containers with other pairs in the adjacent column. However, if a pair of containers are exchanged with others in any column, say, for example, one task in column 1 with one task in column 3 in Figure 4, and if there is precedence relationship between them in the QCs’ working schedule, such exchange could enlarge the waiting time for some QCs and AGVs even result in infeasible solution. Thus, we restrict the inter-column exchange only executed between two adjacent columns.
Figure 4: An example of intra-column exchange and inter-column exchanges.

These two neighborhood searches are repeated several times to look for better solutions. For the intra-column exchange, we firstly choose a certain number of better candidate solutions according to a higher threshold and then perform the intra-column exchange on them. If there is improvement to a candidate solution, the candidate solution is replaced by the improved one. If there is no candidate solution is improved, we enlarge the searching space by lower the threshold of candidates and apply the intra-column exchanges to the new qualified candidate solutions from the last inter-column exchange. If there is still not any improvement, we would merely start a new round of inter-column exchange. If there is no any improvement after a pre-defined time, the whole searching process ends. Obviously, the number of iteration depends on the number of AGVs and the number of containers in their schedules.

Moreover, during the each step of iteration, the changes of every candidate solution are recorded and saved in a tabu list preventing redo these changes. For each candidate, we only saved the last 100 changes in the tabu list and discard ones before them. In addition, the initial solutions are chosen by the worse ones among the feasible solutions obtained from the priority-rules introduced in the last part. The whole flow of the neighborhood search is illustrated as the following table.

Table 4: The process of neighborhood search.

| Initial Sol. | The feasible solutions obtained from the priority-rules. |
Step 1
Repeat intra-column neighborhood search for the pre-defined number iteration.
If candidate solutions have an improvement, then pass them to step 2.
Otherwise, the original ones are kept for the next steps.

Step 2
Apply inter-column neighborhood search between any two neighbored columns.
Save all qualified solutions as candidates.

Step 3
For the candidates from Step 2, do:

3.1 Select the higher quality candidates.
Perform the intra-column exchange in Step 1.
If there is any improvement in objective value, the solutions are updated.

3.2 Otherwise, enlarge the search space by and adding those lower quality solutions from Step 2 and perform the intra-column exchange to them.
If there is still not any improvement, use the solutions from Step 3.1 for the next round of inter-column exchanges.

Step 4
End the whole process after a pre-defined times of inter-column exchanges.

1. NUMERICAL EXPERIMENTS

In this part, the problem is solved with the three methods introduced previously.
Based on the problem size, the experiments can be divided into 3 categories – small (S), medium (M) and large (L); and each of them contain 10 test cases. The containers’ storage plans are generated randomly in C++ program. Considering the layout and the AGVs’ speed variation in different terminals, we set the parameters based on some published papers (3, 8, 23):

- The average operation speed of a tandem lift QC is about 60 moves/hr, implying that the cycle time is 60 seconds.
- The speed of the AGVs is about 6m/sec.
- The distance between adjacent cranes is 90m.
- The number of AGVs is different in different scenarios.

To simplify the problem, we set all distances (measured in AGV’s travel time) normalized at 1 time unit for a trip between two adjacent working stations, and the QC’s cycle time are normalized at 4 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 using OPL (a modeling language) and the commercial optimization solver CPLEX version 12.1 (24, 25). The two heuristic methods were coded in C++. All computations were conducted on a personal computer with a 2.66GHz/2.66GHz Intel Core 2 Quad CPU on a Microsoft Windows platform and a 4.00 GB RAM.

Table 5: Computational results from the three methods.

<table>
<thead>
<tr>
<th>No. Case</th>
<th>MILP model solved by CPLEX</th>
<th>Neighborhood Search</th>
<th>Dispatch Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BI  Time (sec)</td>
<td>Obj1  Time_B  Time_F  Gap1</td>
<td>Obj2  Gap2</td>
</tr>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16_4_1</td>
<td>36  420</td>
<td>36  7.26  133.86  0.00</td>
<td>36  0.00</td>
</tr>
<tr>
<td>16_4_2</td>
<td>34  1387</td>
<td>34  26.92  439.42  0.00</td>
<td>35  2.94</td>
</tr>
<tr>
<td>16_4_3</td>
<td>30  690</td>
<td>30  15.44  197.81  0.00</td>
<td>30  0.00</td>
</tr>
<tr>
<td>16_4_4</td>
<td>29  6590</td>
<td>29  0.09  95.97  0.00</td>
<td>29  0.00</td>
</tr>
<tr>
<td>16_4_5</td>
<td>36  8792</td>
<td>37  115.06  481.5  2.78</td>
<td>36  0.00</td>
</tr>
<tr>
<td>16_4_6</td>
<td>36  782</td>
<td>36  0.03  12.95  0.00</td>
<td>36  0.00</td>
</tr>
<tr>
<td>16_4_7</td>
<td>33  2151</td>
<td>33  96.757  466.247 0.00</td>
<td>34  3.03</td>
</tr>
<tr>
<td>16_4_8</td>
<td>32  3539</td>
<td>32  0.49  48.93  0.00</td>
<td>32  0.00</td>
</tr>
<tr>
<td>16_4_9</td>
<td>32  1263</td>
<td>32  100.58  460.22 0.00</td>
<td>32  0.00</td>
</tr>
<tr>
<td>16_4_10</td>
<td>33  999</td>
<td>33  5.43  85.1  0.00</td>
<td>33  0.00</td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18_6_1</td>
<td>22  -</td>
<td>22  12.19  473.07 0.00</td>
<td>22  0.00</td>
</tr>
<tr>
<td>18_6_2</td>
<td>19  -</td>
<td>19  16.74  467.97 0.00</td>
<td>19  0.00</td>
</tr>
<tr>
<td>18_6_3</td>
<td>26  -</td>
<td>24  9.33  470.12 -7.69</td>
<td>24  -7.69</td>
</tr>
<tr>
<td>18_6_4</td>
<td>25  -</td>
<td>20  25.83  833.99 -20.00</td>
<td>21  -16.00</td>
</tr>
<tr>
<td>18_6_5</td>
<td>28  -</td>
<td>26  108.42  1030.7 -7.14</td>
<td>28  0.00</td>
</tr>
<tr>
<td>18_6_6</td>
<td>24  -</td>
<td>23  17.64  319.89 -4.17</td>
<td>24  0.00</td>
</tr>
<tr>
<td>18_6_7</td>
<td>20  -</td>
<td>20  9.24  484.73 0.00</td>
<td>20  0.00</td>
</tr>
<tr>
<td>18_6_8</td>
<td>37  -</td>
<td>34  17.96  755.49 -8.11</td>
<td>35  -5.41</td>
</tr>
<tr>
<td>18_6_10</td>
<td>20  -</td>
<td>20  21.15  393.1  0.00</td>
<td>20  0.00</td>
</tr>
<tr>
<td>L</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24_6_1</td>
<td>54  -</td>
<td>38  201.62  1755.11 -29.63</td>
<td>38  -29.63</td>
</tr>
<tr>
<td>24_6_2</td>
<td>61  -</td>
<td>41  802  2369.1 -32.79</td>
<td>47  -22.95</td>
</tr>
<tr>
<td>24_6_3</td>
<td>55  -</td>
<td>39  1377.9  3039.8 -29.09</td>
<td>39  -29.09</td>
</tr>
<tr>
<td>24_6_4</td>
<td>190  -</td>
<td>54  2039.92  3732.48 -71.58</td>
<td>54  -71.58</td>
</tr>
<tr>
<td>24_6_5</td>
<td>59  -</td>
<td>41  300.51  2127.74 -30.51</td>
<td>47  -20.34</td>
</tr>
<tr>
<td>24_6_6</td>
<td>63  -</td>
<td>41  685.39  2486.98 -34.92</td>
<td>45  -28.57</td>
</tr>
<tr>
<td>24_6_7</td>
<td>80  -</td>
<td>34  384.84  2745.89 -57.50</td>
<td>34  -57.50</td>
</tr>
<tr>
<td>24_6_8</td>
<td>46  -</td>
<td>39  53.32  1714.79 -15.22</td>
<td>47  2.17</td>
</tr>
<tr>
<td>24_6_9</td>
<td>51  -</td>
<td>43  892.52  3802.41 -15.69</td>
<td>50  -1.96</td>
</tr>
</tbody>
</table>
Note:
BI represents the best integer solution found by the CPLEX;
For the M and L test cases, the “-” in the column of computation time represents the computation time is 10hrs;
The computation time of dispatch rule is only few seconds, so we do not list it 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;
Gap1 = 100% * (obj.1 - BI)/BI;
Gap2 = 100% * (obj.2 - BI)/BI.

Table shows the computation results for the proposed three methods and we can compare them from the following two main aspects:

**Objective value:** For the small problems, all of them can find the optimal solutions. For the medium and large ones, the CPLEX solver cannot obtain the optimal solution even after running 10hrs but the results from CPLEX provide a benchmark for the two heuristic methods. The results show that except only one case (16_4_5), the heuristic results are as good as or even better than the best integer solutions found by the CPLEX. In addition, the neighborhood search method always performs better than the dispatch rule except only one experiment. The possible reasons may include: (1) The dispatch rule is myopic and only makes the best decision based on the current situation. (2) When one strategy is executed in the priority measurement, all the containers are measured with it without change. It ignores the possibility that for some measurement, a different strategy could generate better results for the whole problem. However, comparison of all the priority strategies in every measurement is impossible within reasonable computation time. Compared to it, the neighborhood search method overcomes these disadvantages by exchanging AGVs’ working sequences according to some principles. This is because the generation of new solutions equals to the application of different priority strategies or the knowledge of future events in decision making.

**Computation time:** Obviously, the MILP is the most time-consuming method and the priority-based dispatch rule is the fastest one. The computation time of neighborhood search is much shorter than the MILP, but longer than dispatch rule. By comparing Time_B and Time_F in the neighborhood search method, we can find out that there is some redundancy in the computation time, but it is necessary to obtain solutions of higher quality. Sometimes, changing one container in an AGV’s working sequence cannot improve the solution. When it is combined with another change, the objective value would be improved. If we end the process too early or filter solutions too strictly, we would lose those solutions with the possibility of improvement. 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, we end the whole search process when there is no improvement in the objective value after three iterations in row. In up to 20% experiments (which has been marked by asterisk), this process finds better solutions by keep searching after no improvement in two iterations.

4. CONCLUSIONS AND DISCUSSION

Both of the QCs and AGVs introduced in this paper are advanced equipment designed to increase the productivity in the container terminals. However, neither of them can achieve this purpose without the efficient operation control. The AGV’s 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 respect of mathematic modeling.

In this paper, the problem is firstly formulated by an MILP model. Due to the complexity of solving such model, two heuristic methods are proposed for practical purposes. Numerical experiments show that both of them can obtain good solutions within very short time and the latter one performs generally better in terms of the objective value. Our future work on this problem will focus on the improvement of neighborhood search method as well as the heuristic dispatching rules for the daily operation in reality.

5. ACKNOWLEDGEMENT

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

6. REFERENCE


4. Qiu, L., and Hsu, W. J., A bi-directional path layout for conflict-free routing of


23. Stahlbock, R., and Voß, S., Operations research at container terminals: a literature
1 update. OR spectrum (30), 2008, pp. 1-52.

2 24. ILOG OPL Studio 6.3, ILOG Inc.