An Efficient Continuous-time Model for Container Yard Crane Work

Wenkai Li
Yong Wu
Matthew E. H. Petering
Mark Goh
Robert de Souza

The Logistics Institute—Asia Pacific
National University of Singapore
Block AS6, Level 5, 11 Law Link
Singapore 119260, SINGAPORE

Abstract
The operations of a container terminal is often bottlenecked by slow YC (yard crane) movements. Long PM (prime mover) queues in front of YCs is a common sight. Efficient YC scheduling to reduce the PM waiting time thus plays a key role in increasing a terminal’s throughput. This paper develops an efficient continuous-time YC scheduling MILP model by treating real world operational constraints such as inter-crane interference, fixed crane separation distances and simultaneous storage/retrieval jobs handling. We show how the model can be solved more efficiently when included with a heuristics. Solution results show that the model can yield near-optimal solutions in reasonable time.

1. Introduction
Globalization has become an irreversible trend with the rapid development of IT and the decreasing cost of communication and transportation. Today, a substantial proportion of international cargo (about 70%) is shipped via standardized containers using container vessels. At the same time, due to intense competition among ports, a terminal operator is always under pressure to reduce its turnaround time and to increase its service level. Superposed by the increased container handling amount and shorter turnaround times, container terminal decision-makers and operations executives want and need efficient computational tools which can de-bottleneck a terminal’s container flow and thus increase the terminal’s throughput.

Figure 1 shows a simplified typical container terminal. Vessels mooring at the berths are waiting for quay cranes (QCs) which are handling their containers. PMs shuttle between QCs and YCs to move these containers from the berth to the yard and vice versa. After their arrival at the yard blocks, PMs queue in front of YCs until they are served. Typically, the job handling time of an YC is usually twice as long as that of a QC (see Ng and Mak, 2005a for details). Container flow is thus often constrained by the slower YC operations.
Even though a QC is technically capable of making 40 moves per hour when PMs are always on hand at the quay to deliver/receive the appropriate containers to/from the QC, the average QC rate at most container ports is currently far less than that value. While PMs are often waiting in front of YCs, QCs are hungry for PMs to deliver their containers. An efficient YC schedule is therefore the key for QC rates to approach the 40 moves per hour.

![Figure 1. A typical container terminal](image)

There are some unique operation constraints which have to be considered in YC scheduling. First, two YCs sharing the same bi-directional lane cannot cross gantry one other, i.e., a crane located at the left part of a yard cannot reach the right part of the yard if another crane is on the lane between two parts. Second, YCs in the same lane that are working simultaneously must remain separated by at least 160 feet of safety (Figure 2). This constraint is necessary because YCs need to leave some space for the PMs to park between the YCs. If YCs are closer than this, the PMs (e.g., PM1 in Figure 2) leaving the upstream YC (e.g., YC1 in Figure 2) will not be able to pull out of the handling lane and into the bypass lane before encountering the queue of PMs at the downstream YC (e.g., YC2 in Figure 2). Another problem is that there is not enough space for the PMs (e.g., PM2 in Figure 2) that are attempting to reach the downstream YC (e.g., YC2 in Figure 2) to move from the bypass lane into the handling lane. Third, there are two kinds of moves, namely storage and retrieval moves, need to be handled by YCs in a yard block. It can be seen from Figure 1 that, when PMs carrying containers from vessels to yard blocks arrive, YCs have to place those containers onto the block. When empty PMs arrive to carry containers from yard to vessel, the YCs have to retrieve these containers from the block to these PMs.

![Figure 2. Separation distance between working YCs](image)
2. Literature review

Many papers have considered the scheduling of a single YC where containers are grouped and the crane must retrieve containers from specified groups according to a fixed sequence, without due dates or release times, while minimizing travel distance. Since containers belonging to a specific group may be stored in multiple locations, both the YC route and number of containers picked up at each slot are decisions to be made (Kim and Kim (1999), Kim and Kim (2003), Kim and Kim (1997), and Narasimhan and Palekar (2002)). Recently Ng and Mak (2005a) developed a continuous-time model for single YC scheduling problem. They proposed a branch-and-bound algorithm to solve the problem, considering only one type of move, the storage move. If retrieval moves are to be considered, their branch-and-bound algorithm cannot be applied because the procedure to find the lower bound can only be used for storage moves. It took about 444.72 seconds in average for their algorithm to solve an YC scheduling problem with 25 jobs. Ng and Tsang (2005b) later developed a genetic algorithm to solve a single YC scheduling problem. Again, only one type of move, the storage move, is considered.

For the scheduling of multiple yard cranes, Cheung et al. (2002), Linn et al. (2003), and Zhang et al. (2002) developed methods for allocating yard cranes among yard blocks in an entire terminal and for scheduling cross-gantry moves. However, these approaches did not generate detailed work schedules for the YCs. From what we know, Ng (2005c) is the only paper which has proposed a method for constructing detailed work schedules for multiple YCs in a block at a container terminal with inter-crane interference being considered. He proposed a heuristic-based algorithm to solve this multiple YC scheduling problem. However, this study also does not consider retrieval moves and does not enforce the constraint that yard cranes must be separated by a minimum distance which may make the YC work schedule inoperable. So far, the algorithms applied on the YC scheduling problem (Ng and Mak (2005a); Ng and Tsang (2005b); Ng (2005c)) in the literature only consider storage moves. This is in part due to the ease of developing a heuristic algorithm for storage moves. Incorporating retrieval moves becomes challenging.

The YC scheduling problem has been proven to be an NP-hard problem (Ng, 2005c). In general, despite recent developments in model formulation, commercial solvers and computing power, the scheduling problems are still limited to small-sized problems. Ng and Mak (2005b) developed a simple continuous-time model for yard crane scheduling without considering most of the realistic operation constraints, such as inter-crane interference, crane safety distance separation and simultaneous handling of storage/retrieval jobs. They found that CPLEX failed to find the optimal solution within 30 minutes for problems with more than 20 container moves. Thus, MILP-based continuous-time models can become intractable for real-sized problems. Most of the research so far have used MILP-based models as benchmarks and have developed heuristics-based models for yard crane scheduling. Ng (2005c) proposed a discrete-time MILP model as the benchmark for his heuristics-based algorithm. He
found that, CPLEX took about 10 minutes on average to find the optimal schedule for a problem with 2 YCs and 10 jobs in a 40-slot yard block.

3. Mathematical Model
An efficient continuous-time MILP (mixed integer linear programming) model is developed in this paper. Important operational constraints, which will be described in detail in the following sections, are considered and modeled. The notations used for the indexes, sets, parameters and variables in the mathematical formulation are defined in Appendix I.

3.1 Assumptions
- The job handling time of an YC is usually 2~4 minutes (Ng and Mak, 2005a). We assume that the job handling time of all YCs is 3 minutes (i.e., 20 moves/hour).
- 20 to 30 moves in a two-hour time window are used in the scenarios tested in this paper. Each yard block contains 40-60 container slots.
- Assume that PMs are always available for YCs. In a container terminal, YC-YC interference is usually a more serious problem than PM-PM interference. YCs have more difficulty substituting for one another than PMs during actual operations (e.g. there are fewer YCs than PMs, YCs are much slower than PMs, and the minimum separation distance is larger for two YCs than for two PMs). Thus, more attention should be paid to scheduling YCs than to dispatching PMs. YCs also exhibit less variable traveling and handling times than PMs, which means that they can follow a predetermined sequence of moves with a less variable result than PMs. Thus, generating YC schedules with the assumption that PMs are always available becomes very worthwhile.
- the minimum difference in slot numbers allowed for two YCs at the same time is assumed to be 8 slots (160 feet) in this paper (Separation=8).
- YC acceleration/deceleration stages are taken into account when calculating its gantry time. The average YC gantry speed is 100m/min and the acceleration/deceleration is 0.3m/s/s.

3.2 Objective Function
Minimize \( PMWT = W_{re} \sum_{m \in R} RE_m + W_{rd} \sum_{m \in R} RL_m + W_{sd} \sum_{m \in S} STL_m \)  

(1)
The objective is to minimize a linear combination of the total amount of retrieval earliness, the total amount of storage delay and the total amount of retrieval delay. Since a retrieval delay directly leads to the QC schedules being delayed, the weight for total retrieval delay, \( W_{rd} \), is larger. In this paper, we set \( W_{re} = W_{sd} = 1 \) and \( W_{rd} = 2 \).

3.3 Constraints
A container move should be assigned to a yard crane

\[ \sum_{c \in C} W_{mc} = 1, \quad \forall m \in M, \ NC > 1 \]  

(2)
When the number of YCs is greater than one, eq (2) ensures that each move is assigned to exactly one YC. If there is only one YC, the value of $W_{mc}$ is fixed as 1 and this variable can be deleted. To maintain the consistency of constraints across multi/single-YC scenarios, we keep this variable and add the following equation when there is only one YC:

$$W_{mc} = 1, \quad \forall m \in M, \text{ NC}=1$$

**Identifying overlapping container moves $n$ against move $m$.**

Different from discrete-time formulation in which nearby container moves are synchronized to the boundaries of time intervals, container moves can overlap to any extent in a continuous-time formulation. Overlapping container moves (referred to as $n, n \neq m$) against move $m$ are moves whose job finishing times, $T_{en}$ ($=T_{sn} + HT$), are greater than the job starting time of move $m$, $T_{sm}$, and their job starting time, $T_{sn}$, is smaller than the job finishing time of move $m$, $T_{em}$. This is illustrated in Figure 3. In Figure 3, moves 3 and 4 are overlapping moves against move $m$ while moves 1, 2, 5, 6 are not.

To identify overlapping moves, binary variables $A_{1mn}$ and $A_{2mn}$ are defined:

$$A_{1mn} = \begin{cases} 1 & \text{if } T_{en} > T_{sm} \\ 0 & \text{otherwise} \end{cases}$$

That means if move $n$ happens before $m$ and they are non-overlapping, then $A_{1mn}$ equals to 0. For example, move 1 in Figure 3.

$$A_{2mn} = \begin{cases} 1 & \text{if } T_{sn} < T_{em} \\ 0 & \text{otherwise} \end{cases}$$

That means if move $n$ happens after $m$ and two moves are non-overlapping, then $A_{2mn}$ equals to 0. For example, move 6 in Figure 3.

For moves whose job finishing time equals the job starting time of move $m$, i.e., $T_{en} = T_{sm}$, or moves whose job starting time equals the job finishing time of move $m$, i.e., $T_{sn} = T_{em}$, we define these moves as non-overlapping moves against $m$. For example, moves 2 and 5 in Figure 3 are also defined as non-overlapping moves against move $m$. 

![Figure 3. Overlapping container moves](image-url)
It is clear from Figure 3 that, both $A_{1mn}$ and $A_{2mn}$ of overlapping moves take a value of 1. We can then identify the overlapping moves using $A_{1mn}$ and $A_{2mn}$ and constraints (3a) to (4b) as follows:

$$T_{S_n} + HT \geq T_{S_m} - \text{BigM}*(1 - A_{1mn}), \ \forall m, n \in M, \ m \neq n, \ NC > 1$$  (3a)

$$T_{S_m} \geq T_{S_n} + HT - \text{BigM}*A_{1mn}, \ \forall m, n \in M, \ m \neq n, \ NC > 1$$  (3b)

$$T_{S_m} + HT \geq T_{S_n} - \text{BigM}*(1 - A_{2mn}), \ \forall m, n \in M, \ m \neq n, \ NC > 1$$  (4a)

$$T_{S_n} \geq T_{S_m} + HT - \text{BigM}*A_{2mn}, \ \forall m, n \in M, \ m \neq n, \ NC > 1$$  (4b)

Constraints (3a) and (3b) are used to define the value of $A_{1mn}$. For move 1 in Figure 3, we have $T_{S_n} + HT < T_{S_m}$, this forces $A_{1mn}$ to take a value of zero because otherwise (3a) is not satisfied. In the meantime (3b) is also satisfied when $A_{1mn}$ is zero. For move 3 in Figure 3, $A_{1mn}$ is forced to take a value of 1 because $T_{S_n} + HT > T_{S_m}$. Similarly, constraints (4a) and (4b) are used to define the value of $A_{2mn}$. Note that constraints (3a) and (3b) are only used for scenarios with multiple cranes. For scenarios with a single crane, (3a) and (3b) should be replaced by (5a)-(5b) as follows:

$$T_{S_m} \geq T_{S_n} - \text{BigM}*(1 - A_{2mn}), \ \forall m, n \in M, \ m \neq n, \ NC=1$$  (5a)

$$T_{S_n} \geq T_{S_m} - \text{BigM}*A_{2mn}, \ \forall m, n \in M, \ m \neq n, \ NC=1$$  (5b)

For single-crane scenarios, all moves are handled by one crane and hence there exist no overlapping moves (a crane can only handle one container move at a time). Thus, (5a) and (5b) are used to ensure that, if the job starting time of move $n$ is greater than move $m$, i.e., $T_{S_n} > T_{S_m}$, we have $A_{2mn} = 0$, otherwise $A_{2mn} = 1$. For single-crane scenarios, $A_{1mn}$ is not necessary and we can fix its value to 0:

$$A_{1mn} = 0, \ \forall m, n \in M, \ m \neq n, \ NC=1$$

Then we can define a 0-1 variable $Y_{mn}$:

$$Y_{mn} = \begin{cases} 1 & \text{if } A_{1mn} = A_{2mn} = 1 \\ 0 & \text{otherwise} \end{cases}$$

That is, if move $n$ is the overlapping move against move $m$, then $Y_{mn} = 1$. Constraints (6a) to (6c) are used to ensure this:

$$Y_{mn} \geq A_{1mn} + A_{2mn} - 1, \ \forall m, n \in M, \ NC>1$$  (6a)

$$Y_{mn} \leq A_{1mn}, \ \forall m, n \in M, \ NC>1$$  (6b)

$$Y_{mn} \leq A_{2mn}, \ \forall m, n \in M, \ NC>1$$  (6c)
Note that constraints (6a)-(6c) ensure that value of $Y_{mn}$ only be either 0 or 1. Thus, $Y_{mn}$ is defined as a continuous variable.

For single-crane scenarios, $Y_{mn}$ is fixed to zero:

$$Y_{mn} = 0, \quad \forall m, n \in M, \; NC=1$$

**Relation of job starting times of consecutive container moves at the same YC.**

If move $n$ happens after $m$, and both jobs $m$ and $n$ are handled by the same YC, then constraint (7) holds:

$$T_{s_n} \geq T_{s_m} + HT + GH_{mn} - \left[3 - W_{mc} - W_{nc} - (1 - A^2_{mn})\right] \cdot \text{BigM}, \quad \forall m, n \in M, \; m \neq n, \; \forall c \in C$$

(7)

Constraint (7) is valid for both multi/single-crane scenarios. It states that, if both moves $m$ and $n$ are handled by YC $c$ (i.e. $W_{mc}=W_{nc}=1$), and move $n$ happens after $m$ (i.e. $A^2_{mn}=0$), then job $n$ should start after the job finishing time of move $m$, $T_{s_m}+HT$, plus the YC gantry time from slot $m$ to slot $n$, $GH_{mn}$. The parameter $GH_{mn}$ is defined as:

$$GH_{mn} = \begin{cases} |\text{slot}_m - \text{slot}_n| \cdot 0.06 + 0.09 & \text{when } |\text{slot}_m - \text{slot}_n| \neq 0 \\ 0 & \text{when } |\text{slot}_m - \text{slot}_n| = 0 \end{cases}$$

In the above definition, $|\text{slot}_m - \text{slot}_n|$ is the absolute slot number difference. If moves $m$ and $n$ are handled by the same YC consecutively and they are located at the same slot ($\text{slot}_m = \text{slot}_n$), then the YC does not need to move and the gantry time is zero. If moves are located at different slots, the gantry time (in minutes) of the YC is $|\text{slot}_m - \text{slot}_n| \cdot 0.06 + 0.09$ by taking into account the YC acceleration/deceleration stages.

**Relation of job starting times of consecutive container moves at different YCs.**

If move $n$ happens after move $m$, and moves $m$ and $n$ are handled by different YCs, then constraint (8) holds:

$$T_{s_n} \geq T_{s_m} + HT + \text{OverLapT}_{mn} - \left[3 - W_{mc} - (1 - W_{nc}) - (1 - A^2_{mn})\right] \cdot \text{BigM}, \quad \forall m, n \in \text{NearJob}_{mn}, \; \forall c \in C, \; NC > 1$$

(8)

Constraint (8) is only valid for multi-crane scenarios. It states that, if moves $m$ and $n$ are near jobs (their absolute slot distance is less than Separation) and handled by different YCs (i.e. $W_{mc}=1, W_{nc}=0$), and move $n$ happens after $m$ (i.e. $A^2_{mn}=0$), then job $n$ should start after the job finishing time of move $m$, $T_{s_m}+HT$, plus the YC gantry time, OverLapT$_{mn}$. Set NearJob$_{mn}$ is defined as:

$$\text{NearJob}_{mn} = \{m, n \in M : |\text{slot}_m - \text{slot}_n| < \text{Separation} \land m \neq n\}$$

That is, set NearJob$_{mn}$ includes all moves whose slot number differences are less than Separation. The parameter OverLapT$_{mn}$ is defined as (illustrated in Figure 4):

$$\text{OverLapT}_{mn} = (\text{Separation} - |\text{slot}_m - \text{slot}_n|) \cdot 0.06 + 0.09$$
In Figure 4, moves \( m \) and \( n \) belong to \( \text{NearJob}_{mn} \). At the left of Figure 4, move \( m \) is handled by \( \text{YC} \ c \) and move \( n \) handled by \( \text{YC} \ cc \). Because the slot difference of \( m \) and \( n \) is less than \( \text{Separation} \), \( \text{YC} \ cc \) cannot handle move \( n \) immediately after \( \text{YC} \ c \) finishes handling \( m \). \( \text{YC} \ cc \) has to wait for \( \text{YC} \ c \) to gantry aside until the slot difference between \( \text{YC} \ c \) and slot \( n \) is \( \text{Separation} \). The slot number that \( \text{YC} \ c \) has to gantry is \( \text{Separation} - |\text{slot}_m - \text{slot}_n| \). Again, the acceleration/deceleration stages of \( \text{YC} \ c \) are taken into account when calculating the actual gantry time. The feasible schedule of moves \( m \) and \( n \) is shown at the right of Figure 4.

![Figure 4. Gantry time between near moves](image)

**Moves handled by neighboring YCs**

For overlapping moves, if they are handled by neighboring YCs, their slot difference should be greater than \( \text{Separation} \). Constraint (9) is used to ensure this:

\[
W_{n, cc} \leq 2 - Y_{mn} - W_{mc}, \quad \forall c, cc \in C, \forall m, n \in \text{LeftJob}_{mn}, \quad cc > c, c < NCW_2 \tag{9}
\]

Without loss of generality, the slots of a yard block are numbered increasingly from left to right. We also assume that moves are numbered increasingly according to the slot number they are located in, i.e., if \( m < n \), then \( \text{slot}_m \leq \text{slot}_n \).

With the above assumptions, we first define set \( \text{LeftJob}_{mn} \) as:

\[
\text{LeftJob}_{mn} = \{m, n \in M: \text{slot}_n - \text{slot}_m < \text{Separation} \land m \neq n\}
\]

![Figure 5. Definitions of LeftJob, HighJob and LowJob](image)
That is, those moves that are placed at the left of move $m$ and those moves that are placed at the right of move $m$ but their slot difference against move $m$ is less than Separation belongs to LeftJob$_{mn}$ (Figure 5).

Constraint (9) is only valid for multi-crane scenarios. It states that if moves $m$ and $n$ are overlapping, i.e., $Y_{mn}=1$, and move $m$ is handled by YC $c$, i.e., $W_{mc}=1$, then YC $c+1$ cannot handle moves which belong to LeftJob$_{mn}$. For example, In Figure 5, YC 2 cannot handle moves that belong to LeftJob$_{mn}$ when YC 1 is handling move $m$. This is required by Separation apart of YCs and inter-crane interference.

**A crane can handle only one job at a time**
This is enforced by constraint (10) below:

$$ W_{n,c} \leq 2 - Y_{mn} - W_{mc}, \quad \forall c \in C, \forall m, n \in M, \text{NC}>1 \quad (10) $$

Constraint (10) is only valid for multi-crane scenarios. It states that, If moves $m$ and $n$ are overlapping, i.e., $Y_{mn}=1$, and YC $c$ is handling move $m$, i.e., $W_{mc}=1$, then YC $c$ cannot handle move $n$, i.e., $W_{nc}=0$.

**A Storage move can take place no earlier than its target time**
This is enforced as follows:

$$ T_{Sm} \geq T_{rget_{m}}, \quad \forall m \in S_{m} $$

**Computing the retrieval earliness, $RE_{m}$, of move $m$:**

$$ RE_{m} \geq T_{rget_{m}} - T_{Sm}, \quad \forall m \in R_{m} \quad (11) $$

**Computing the retrieval lateness, $RL_{m}$, of move $m$:**

$$ RL_{m} \geq T_{Sm} - T_{rget_{m}}, \quad \forall m \in R_{m} \quad (12) $$

**Computing the storage lateness, $STL_{m}$, of move $m$:**

$$ STL_{m} \geq T_{Sm} - T_{rget_{m}}, \quad \forall m \in S_{m} \quad (13) $$

**Moves locate at the first and last Separation slots**
Moves locate at the first Separation slots (see Figure 5) can only be assigned to the first YC. This is also the result of inter-crane interference. Similarly, moves at the last Separation slots can only be assigned to the last YC. This is enforced as follows:
\[ W_{mc} = 1, \quad \forall m \in LowJob_m, c = 1 \]
\[ W_{mc} = 1, \quad \forall m \in HighJob_m, c = NC \]

Set LowJob\(_m\) is defined as:

\[ LowJob_m = \{ m \in M: \text{slot}_m \leq \text{Separation} \} \]

LowJob\(_m\) includes moves locate at the first Separation slots (see Figure 5). Similarly, HighJob\(_m\) is defined as:

\[ HighJob_m = \{ m \in M: \text{slot}_m > (\text{NSL} - \text{Separation}) \} \]

Where NSL is the total number of slots. HighJob\(_m\) includes moves locate at the last Separation slots (see Figure 5).

4. **Further reduction of the number of integer variables of the model**

The number of integer variables and the number of constraints can be further reduced by detailed analysis of the constraints (2)-(13). Firstly, from Figure 3, we see that if move \( n \) is used as the reference move, then when \( A_{1mn} = 0 \), we have \( A_{2mn} = 0 \); when \( A_{1mn} = 1 \), we have \( A_{2mn} = 1 \), etc. In short, \( A_{1mn} \) equals to \( A_{2mn} \) and \( A_{1mn} \) can be replaced \( A_{2mn} \). Thus, binary variable \( A_{1mn} \) and constraints (3a)-(3b) can be eliminated from the model. \( A_{1mn} \) in constraints (6a)-(6b) is then replaced by \( A_{2mn} \).

In fact, constraints (6a)-(6b) are used to ensure that the value of \( Y_{mn} \) be 1 when \( A_{1mn}=A_{2mn}=1 \) (or equivalently \( A_{2nm}=A_{2mn}=1 \)). We can further eliminate the continuous variable \( Y_{mn} \) from the model by replacing it with \( (A_{1mn}+A_{2mn}) \) and corresponding constraints (6a)-(6c) can be eliminated from the model. To achieve this, constraints (9) and (10) should be modified as follows:

\[ W_{n,cc} \leq 3 - A_{2mn} - A_{2nm} - W_{mc}, \quad \forall c, cc \in C, \forall m, n \in \text{LeftJob}_m, cc > c, c<NC \]  
\[ (9a) \]
\[ W_{n,c} \leq 3 - A_{2mn} - A_{2nm} - W_{mc}, \quad \forall c \in C, \forall m, n \in M, NC>1 \]  
\[ (10a) \]

5. **Heuristics integrated with the MILP model**

Even though the number of binary variables has been reduced significantly by defining two bi-index binary variables, \( W_{mc} \) and \( A_{2mn} \), compared with defining tri-index binary variables in some papers (Ng, 2005c), the size of the model increases dramatically as the number of moves increases. We observe that, for both storage and retrieval moves, the bigger the gap between the job finishing time of a move and its target time, the bigger the total PM waiting time. Thus, the job finishing time is usually near its corresponding target time in an optimal solution. We can reasonably assume that the job finishing time of each move is placed inside a certain range around its target time (job handling range). For example, we assume that the job finishing time of move \( m \) happens in the range \( (T_{lo}, T_{up}) \) in Figure 6. For a large
enough $|T_{up} - T_{lo}|$, the possibility of the job finishing time of move $m$ being placed beyond this range is very low in an optimal schedule. For moves whose target times are bigger than $T_{up}$, e.g., move $n1$ in Figure 6, we assume that it must be handled after move $m$. For moves whose target times are smaller than $T_{lo}$, e.g., move $n3$ in Figure 6, we assume that it must be handled before move $m$. Only moves located inside in the job handling range ($T_{lo}$, $T_{up}$), e.g., move $n2$, need to be carefully scheduled against move $m$. By applying this heuristics, the model size can be greatly reduced.

To apply the above heuristics into the model, we first define sets Ngb$_{mn}$, NgbH$_{mn}$ and NgbL$_{mn}$ as follows:

$$
\text{Ngb}_{mn} = \{m,n \in M : |\text{Trget}_n - \text{Trget}_m| < \text{Ngb}\text{n} \times HT \land m \neq n\}
$$

$$
\text{NgbH}_{mn} = \{m,n \in M : \text{Trget}_n - \text{Trget}_m \geq \text{Ngb}\text{n} \times HT \land m \neq n\}
$$

$$
\text{NgbL}_{mn} = \{m,n \in M : \text{Trget}_n - \text{Trget}_m \leq \text{Ngb}\text{n} \times HT \land m \neq n\}
$$

where Ngbn is a parameter used to define the job handling range. It is set to 3.5 in this paper. Sets Ngb$_{mn}$, NgbH$_{mn}$ and NgbL$_{mn}$ are defined to include moves $n$ whose target times are placed inside/above/below the job handling range of move $m$, respectively (shown in Figure 6). Some constraints should be modified to apply sets Ngb$_{mn}$, NgbH$_{mn}$ and NgbL$_{mn}$. First, constraints (4a)-(4b) should be defined within Ngb$_{mn}$:

$$
T_{s_n} + HT \geq T_{s_m} - \text{BigM} \times (1 - A_{2_{mn}}), \quad \forall m,n \in \text{Ngb}_{mn}, \ m \neq n, \ NC > 1 \quad (4ah)
$$

$$
T_{s_n} \geq T_{s_m} + HT - \text{BigM} \times A_{2_{mn}}, \quad \forall m,n \in \text{Ngb}_{mn}, \ m \neq n, \ NC > 1 \quad (4bh)
$$

![Figure 6. Definition of neighboring moves](image-url)
Since we assume that all moves \( n \) in NgbH\(_{mn} \) are handled after move \( m \) and all moves \( n \) in NgbL\(_{mn} \) are handled before move \( m \), part of \( A_{2mn} \) can be fixed:

\[
A_{2mn}^2 = 0, \quad \forall m, n \in \text{NgbH}_{mn} \\
A_{2mn}^2 = 1, \quad \forall m, n \in \text{NgbL}_{mn}
\]

Similarly, (5a)-(5b) should be defined within Ngb\(_{mn} \):

\[
T_{s_m} - T_{s_n} \geq \text{BigM}^\ast(1 - A_{2mn}), \quad \forall m, n \in \text{Ngb}_{mn}, m \neq n, \text{NC}=1 \quad (5ah)
\]

\[
T_{s_n} \geq T_{s_m} - \text{BigM}^\ast A_{2mn}, \quad \forall m, n \in \text{Ngb}_{mn}, m \neq n, \text{NC}=1 \quad (5bh)
\]

Constraint (7) should be replaced by (7a)-(7c) as follows:

\[
T_{s_n} \geq T_{s_m} + \text{HT} + \text{GH}_{mn} - [3 - W_{mc} - W_{nc} - (1 - A_{2mn})] \ast \text{BigM}, \\
\forall m, n \in \text{Ngb}_{mn}, \forall c \in C
\]

\[
T_{s_n} \geq T_{s_m} + \text{HT} + \text{GH}_{mn}, \quad \forall m, n \in \text{Ngb}_{mn}, \forall c \in C \quad (7a)
\]

\[
T_{s_n} \leq T_{s_m} - \text{HT} - \text{GH}_{mn}, \quad \forall m, n \in \text{Ngb}_{mn}, \forall c \in C \quad (7c)
\]

(7b) and (7c) simply ensures that moves \( n \) in NgbH\(_{mn} \) are handled after move \( m \) and all moves \( n \) in NgbL\(_{mn} \) are handled before move \( m \).

Similarly, constraint (8) should be replaced by (8a)-(8c) as follows:

\[
T_{s_n} \geq T_{s_m} + \text{HT} + \text{OverLapT}_{mn} - [3 - W_{mc} - (1 - W_{nc}) - (1 - A_{2mn})] \ast \text{BigM}, \\
\forall m, n \in \text{NearJob}_{mn} \cap \text{Ngb}_{mn}, \forall c \in C, \text{NC} > 1
\]

\[
T_{s_n} \geq T_{s_m} + \text{HT} + \text{OverLapT}_{mn}, \quad \forall m, n \in \text{NearJob}_{mn} \cap \text{Ngb}_{mn}, \forall c \in C, \text{NC} > 1 \quad (8a)
\]

\[
T_{s_n} \leq T_{s_m} - \text{HT} - \text{OverLapT}_{mn}, \quad \forall m, n \in \text{NearJob}_{mn} \cap \text{Ngb}_{mn}, \forall c \in C, \text{NC} > 1 \quad (8c)
\]

6. Computational results

The model without heuristics (MODEL1) involves the objective function (1), constraints (2), (4a)-(4b), (5a)-(5b), (7), (8), (9a), (10a) and (11)-(13)). The model with heuristics (MODEL2) involves the objective function (1), constraints (2), (4ah)-(4bh), (5ah)-(5bh), (7a)-(7c), (8a)-(8c), (9a), (10a) and (11)-(13). In a typical container terminal, there are 40 slots, at most 60 slots, in a yard block. 0 to 20 moves are to be handled at each block (Ng and Mak, 2005a). The number of YCs in each block is 2 to 4. Twenty test scenarios with 20~30 moves in 40-slot blocks and 10 test scenarios with 20~30 moves in 60-slot blocks are randomly generated. CPLEX is used to solve the model in a Pentium 1.6GHz computer. The optimal solutions of all test scenarios are obtained by solving MODEL1. The mean time to obtaining the optimal solution for MODEL1 is 35269 seconds which is too long. However, these optimal solutions can be used as the benchmark for MODEL2. The results of MODEL2 are shown in Table 1.
In Table 1, the mean solution time to solve 40-slot blocks with 20–30 moves and 2 YCs is 25.0 seconds. For 60-slot blocks with 20–30 moves, the mean solution time is 41.9 seconds. This is faster than the solution time (444.72 seconds in average) required in the paper of Ng and Mak (2005a) solving 25 moves in a block. The average gaps between $MODEL2$ and the optimal solution are also shown. It can be seen that the average gap for total PM waiting time and objective value are 2% to 4%. A very low average TCT (Total Completion Time) gap (0.06%–0.09%) is obtained. This is 100 times smaller than the gap obtained in the literature (7.3% of TCT gap in Ng (2005c)’s paper). In fact, $MODEL2$ obtained optimal solutions for 18 out of the 30 scenarios tested.

<table>
<thead>
<tr>
<th>Solution Time, seconds</th>
<th>Mean</th>
<th>Maximum</th>
<th>Minimum</th>
<th>Total PM Waiting Time(^a)</th>
<th>TCT(^b)</th>
<th>Obj. Val.(^c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>40 slots</td>
<td>25.0</td>
<td>70.2</td>
<td>0.1</td>
<td>2.46</td>
<td>0.06</td>
<td>2.44</td>
</tr>
<tr>
<td>60 slots</td>
<td>41.9</td>
<td>70.1</td>
<td>0.2</td>
<td>3.59</td>
<td>0.09</td>
<td>3.85</td>
</tr>
<tr>
<td>Pure storage moves</td>
<td>7.5</td>
<td>36.8</td>
<td>0.1</td>
<td>1.54</td>
<td>0.04</td>
<td>1.54</td>
</tr>
<tr>
<td>Storage+Retrieval</td>
<td>40.0</td>
<td>70.2</td>
<td>0.1</td>
<td>3.25</td>
<td>0.08</td>
<td>3.36</td>
</tr>
</tbody>
</table>

\(^a\)Gap = (PMWT of $MODEL2$ – optimal PMWT)/optimal PMWT
\(^b\)Gap = (TCT of $MODEL2$ – optimal TCT)/optimal TCT
\(^c\)Gap = (objective value of $MODEL2$ – optimal objective value)/optimal Objective value

Table 1. Performance of the continuous-time scheduling model with heuristics

It is interesting to note that, $MODEL2$ generally takes a longer time to obtain a good solution when retrieval moves are involved in the scenarios. $MODEL2$ can obtain a good solution in about 7.5 seconds for scenarios with pure storage moves (this is also the case in the literature so far). When retrieval moves are involved, the average solution time of $MODEL2$ is 40 seconds. This may be because retrieval moves can be either handled before and after their target times while storage moves can only be handled after.

7. Conclusion
This paper has developed an efficient continuous-time model for container yard crane work schedules. The number of discrete variables is significantly reduced compared with the other models appeared in the literature. To solve this NP-hard problem, a simple heuristics is applied to further reduce the size of the problem. It is found that near optimal schedules can be found in reasonable time.

References


**Appendix I. Definition of Sets and Parameters**

The notations used for the sets, variables and parameters in the mathematical formulation are as follows.

(a) Indices
- \(c, cc =\) yard cranes, \(c, cc = 1\) to \(C\)
- \(m, n =\) container moves, \(m, n = 1\) to \(M\)
- \(SL =\) slot number, \(SL = 1\) to \(TSL\)

(b) Sets
- \(C\) = number of yard cranes working in the yard.
- \(M\) = number of individual container moves to be scheduled
- \(TSL\) = number of slots in the yard
- \(R_m\) = set of yard retrieval moves
- \(S_m\) = set of yard storage moves (the total number of items in \(R_m\) and \(S_m\) is \(M\))
- \(\text{NearJob}_{mn}\) = moves \(n\) whose slot number differences against move \(m\) are less than \(\text{Separation}\)
- \(\text{Ngb}_{mn}\) = moves \(n\) whose target times are placed inside the job handling range of move \(m\)
- \(\text{NgbH}_{mn}\) = moves \(n\) whose target times are placed above the job handling range of move \(m\)
- \(\text{NgbL}_{mn}\) = moves \(n\) whose target times are placed below the job handling range of move \(m\)
LeftJob
= moves that locates at the left side of move \( m \) and those moves that locate at the right side of move \( m \) but their slot difference against move \( m \) are less than Separation

HighJob
= includes moves locate at the last Separation slots in a yard

LowJob
= includes moves locate at the first Separation slots in a yard

(b) Parameters

\( \text{BigM} = \) A big number, \( \text{BigM} = \text{Max} (\text{Trget} \_\_m) + 50 \).

\( \text{GH} \_\_mn = \) time for an \( YC \) to gantry from slot \( m \) to slot \( n \) when consecutive moves \( m \) and \( n \) are handled by the same \( YC \).

\( HT = \) yard crane handling time of a container move, 3 minutes is used in this paper

\( \text{NC} = \) total number of yard cranes

\( \text{Ngbn} = \) parameter used to defined the job handling range

\( \text{NSL} = \) total number of slots in the yard

\( \text{OverLapT} \_\_mn = \) time for \( YC \) to gantry aside when consecutive nearby moves \( m \) and \( n \) are handled by different \( YCs \).

\( \text{Separation} = \) minimum difference in slot numbers allowed for two \( YCs \) at the same time. Separation is assumed to be 8 slots in this paper.

\( \text{Slot} \_\_m = \) Slot number where move \( m \) takes place. Without loss of generality, we assume that if \( m < n \), then \( \text{slot} \_\_m \leq \text{slot} \_\_n \).

\( \text{Trget} \_\_m = \) target time for move \( m \) in the yard. For retrieval moves, this is the latest job starting time that meets the deadline set by the quay cranes. For storage moves, this is the earliest job starting time following the release of the job in the yard.

\( W \_\_re = \) weight assigned to total retrieval earliness in the objective function

\( W \_\_sd = \) weight assigned to total storage delay in the objective function

\( W \_\_rd = \) weight assigned to total retrieval delay in the objective function

c) Variables

\( \text{PMWT} = \) total prime mover waiting time

\( W \_\_mc = 0-1, \) binary variable to denote if container move \( m \) is assigned to crane \( c \)

\( A1 \_\_mn = 0-1, \) binary variable to denote if move \( n \) happens before the start of move \( m \)

\( A2 \_\_mn = 0-1, \) binary variable to denote if move \( n \) happens after the finish of move \( m \)

\( Y \_\_mn = 0-1, \) continuous variable to denote whether moves \( m \) and \( n \) are overlapping

\( Ts \_\_m = \) job starting time of container move \( m \)

\( \text{RE} \_\_m = \) amount of retrieval earliness for move \( m \)

\( \text{RL} \_\_m = \) amount of retrieval lateness for move \( m \)

\( \text{STL} \_\_m = \) amount of storage lateness for move \( m \)