A CONTAINER STORAGE HANDLING MODEL FOR HIGH TECH AUTOMATED MULTIMODAL TERMINALS

Brad Casey and Erhan Kozan

School of Mathematical Sciences
Queensland University of Technology
Brisbane , Queensland, Australia
{b.casey,e.kozan}@qut.edu.au

ABSTRACT

Multimodal Container Terminals are complex systems which require careful planning and control in order to perform efficiently. The containers may require being stacked in multiple levels to maximise storage usage. The position of the container in the storage area affects the handling time, and this time dramatically increases with rehandling jobs that must be performed to access that container. A model for minimising transfer tardiness in the storage area is developed in this paper. A computer program based on meta-heuristic and simulation techniques are designed and implemented to optimise non-trivial problems. A performance and sensitivity analysis are carried out and suggestions are made for future research in this area.

Key Words: Multimodal, Container

1. INTRODUCTION

Multimodal Container Terminals (MMCT) are complex systems which require careful planning and control in order to perform efficiently. The containers may require being stacked in multiple levels to maximise storage usage. The position of the container in the storage area will affect the handling time, and this time will dramatically increase with rehandling jobs that must be performed to access that container. Synchronisation between the various sections of the terminal must also be achieved in order to prevent bottlenecks in the system and the subsequent delays that will occur as a result.

The problem that will be modelled and solved in this paper will be the optimisation of the storage and handling of containers in the storage area of the MMCT. Several sub-problems must be solved when optimising the storage and handling of containers, including

- Where is the most optimal position in the storage area to store each container, so that the least amount of rehandling is performed, and the container is delivered to its destination by its scheduled departure time? Usually the best position is closest to the departure location, so that the container can be stored as long as possible, and quickly moved to the location when needed. However, there will many containers that will depart from the same location, and they cannot all be stored in exactly the same position, so decisions must be made to work out how far from the optimal position the container should be stored while still allowing for a quick departure.
- Which machines should be allocated to each container to move it between the locations in the storage area, and when should they be moved? The machines can only carry a

limited number of containers at a time, usually only one, so decisions must be made to make sure the work is balanced evenly between the machines while still delivering the containers to their destinations on time.

- How are containers dealt with that need to be moved out of the way to reach other containers? Some models make use of a temporary space where the containers are stored while the containers below them are removed, and then they are placed back onto the stack. Other models, including the model developed in this paper, move the containers to another position in the storage bay.
- How is traffic dealt with in the storage area, including both within a location and between the various locations?

A model for minimising the rehandling and transfer tardiness in the storage are was developed in this paper. A computer program based on Operations Research techniques was designed and implemented to optimise non-trivial problems. A performance and sensitivity analysis was carried out and suggestions are made for future research in this area.

2. LITERATURE REVIEW

Over the last decade and a half, there has been a substantial amount of research done on the study of operations at container terminals. The following section reviews the important papers that have contributed to this field of study.

Vis and Koster (2003) and Steenken, Voss and Stahlbock (2004) have both presented a comprehensive review on the literature on recent research related to container terminals. Vis and Koster (2003) separated the problems involved in operating a container terminal into distinct categories, including unloading and loading of the ship, stacking of containers and various other categories. They concluded that in general it is considered necessary to simplify the problem due to its complexity before using analytical models, while on the other hand, simulation models may be used, however these are generally time-consuming to develop and test. They also suggested that more work could be done on joint optimisation of several different types of handling equipment, as most of the papers reviewed focused on single types of equipment. Steenken, Voss and Stahlbock (2004) first provide a historical overview of containers and container terminal operations. They then classified the different types of handling equipment involved in container terminal operations, as well as providing references to relevant literature. Finally they categorised the various processes involved in container terminal logistics and provided references to literature containing information on the various optimisation methods that have been used to solve these logistical problems. They concluded that there is a need for integrated optimisation between the different areas of the container terminal, as currently there are only a few studies on integrated problems.

There have been quite a large number of papers published on stacking containers in the storage area. Early work was done by Chung, Randhawa and McDowell (1988) where they proposed a methodology of utilizing a buffer space as a method to increase the utilization of the equipment and reduce the total container loading time. They developed a simulation model to compare the proposed methodology with the then-current practice at the Port of Portland. Cao and Uebe (1993) presented an algorithm for solving a special capacitated multi-commodity p-median transportation problem (CMPMTP), which arises in container terminal management. They extended the existing work by applying a Lagrangean relaxation to the CMPMTP, and used a heuristic branch-and-bound algorithm to search for a better solution. De Castilho and Daganzo (1993) presented methods for measuring the amount of handling effort required when two basic strategies are adopted, one that tries to keep all stacks the same size, and the other that segregates containers according to arrival time. Taleb-Ibrahimi, De Castilho and Daganzo (1993) describes handling and storage strategies for export containers at marine terminals and quantifies their performance according to the amount of space and number of handling moves they require.

Kozan and Preston (1999) use Genetic Algorithm techniques to reduce container handling/transfer times at the multimodal terminals. When containers are stacked to multi levels or high, more handling time is needed to retrieve a container at the lower level of the stack. Total throughput time of containers as a function of cranes, forklifts/highstackers and terminal transfer trucks are used to measure the performance of the system. Kozan (2000) discusses the major factors influencing the transfer efficiency of container terminals and a network model is designed to analyse container progress in the system to minimize the total handling and travelling time of containers. This paper considers various types of handling and transfer equipment as well as the location of containers in the yard.

Preston and Kozan (2001a) determine an optimal storage strategy for various container handling schedules. They minimise the ship turn around time of container ships by genetic algorithms and design a scheduling model and applied to container terminals taking into account factors such as container handling equipment, labour resources, storage capacities and terminal layout. Major factors influencing container transfer efficiency are analysed to optimise resource usage resulting in lower operating costs while achieving a desired level of customer service. Similarly optimising the storage location to match a particular transfer schedule is developed by Preston & Kozan (2001b) in a later study and some improvement could be gained.

Chen (1999) presented the findings of a study into yard operations in the container terminal, focusing on what effect unproductive container movements has on the overall efficiency of the terminal. Kim and Kim (1999a) considered how to allocate storage space for import containers, by analysing cases where the arrival rate of containers is constant, cyclic and dynamic. Spaces are allocated for each arriving vessel to minimise the expected total number of rehandles, and mathematical models and solution procedures were suggested for obtaining the optimal solution. Kim and Kim (1999b) continued their work from Kim and Kim (1997), presenting a more in-depth study of how to optimally route transfer cranes in a container yard during operations of export containers at port terminals.

Kim, Park and Ryu (2000) proposed a methodology for determining the storage location of an arriving export container considering its weight. They formulated a dynamic programming model to determine the storage location to minimise the number of relocation movements expected for the loading operation. A decision tree was also developed from the set of optimal solutions to support real time decisions. Chung, Li and Lin (2002) considered the problem of scheduling the movements of cranes in a container storage yard to minimise the total unfinished workload at the end of each time period. A mixed integer linear program was formulated and a new solution approach called the successive piecewise-linear approximation method was developed. Kim and Kim (2002) discuss a method of determining the optimal amount of storage space and the optimal number of transfer cranes for handling import containers. A cost model was developed that consisted of the space const, investment cost of transfer cranes, and operating cost of transfer cranes and trucks. A deterministic model was developed for the minimisation of the cost to the terminal operator, and a stochastic model was developed for the minimisation of cost to both the terminal operator and the customers. Kim and Park (2002) discussed how to allocate space for outbound containers arriving at a storage yard, where the main objectives are to utilize space efficiently and to make loading operations more efficient. A mixed integer linear program was formulated, and two heuristic algorithms were suggested based on the duration-of-stay of containers and the sub-gradient optimisation technique. Zhang et. al. (2002) addresses the crane deployment problem, where given the forecasted workload of each block in the storage yard in each period of a day, the objective is to find the times and routes of crane movements among blocks so that the total delayed workload in the yard is minimised. A mixed integer programming model was formulated and solved by Lagrangean relaxation.

Kim, Kang and Ryu (2004) applied a beam search algorithm to solve the load-sequencing problem in port container terminals, where the operational efficiency of transfer cranes and quay cranes were maximised while satisfying various constraints on stacking containers onto vessels. Ng (2005) examined the problem of scheduling multiple yard cranes to perform a given set of jobs with different ready times in a yard zone with only one bi-directional travelling lane. An integer program was formulated, and then a dynamic programming-based heuristic was developed to solve the problem.

3. MODEL FORMULATION

The notation in the model is defined as follows:

Constants

H (L) : Height (Length) of a standard container

v^H (v^V) : The horizontal (vertical) velocity of the machines

$h_{\max}$: The maximum number of containers allowed to be stacked in a storage bay.

Location Variables

L : The set of all locations

L^{TS} (L^{SB}) : The set of all transfer stations (storage bays)

l_c^A (l_c^D) : The arrival (departure) transfer station for container *c*

l_c^S : The storage bay that container *c* will be stored.

n_l : The number of containers that can be stored horizontally in location *l*

U_l : The storage capacity of location *l*

D_{l_1,l_2} : The shortest distance between locations l_1 and l_2

E_{l_1,l_2} : 0 if the closest end of location l_1 to location l_2 is the end near stack 1 of location l_1, 1 otherwise.

Container Variables

C : The set of all containers

x_c^A (x_c^D) : The stack that container *c* is placed in the arrival (departure) transfer location l_c^A (l_c^D)

$x_{c,t}^S$ ($y_{c,t}^S$) : The stack (level) that container *c* is stored in storage bay l_c^S at time *t*

$p_{c,l,x,y,t}$: 1 if container *c* is stored at position (x, y) in location *l* at time *t*, 0 otherwise.

Machine Variables

M : The set of machines used in the storage area

m_c^A : Machine assigned to carry container c from its arrival transfer station l_c^A to storage bay l_c^S.

m_c^D : Machine assigned to carry container c from storage bay l_c^S to its departure station l_c^D.

$q_{m,l,t}$: 1 if machine m is in location l at time t, 0 otherwise.

$r_{m,c,t}$: 1 if machine m is carrying container c at time t, 0 otherwise.

Job Variables

J_m : The set of jobs that machine m has been allocated to perform

J_m^A (J_m^D) : The set of arrival (departure) jobs that machine m has been allocated to perform

j_c^A : The job to move container c from arrival transfer station l_c^A to storage bay l_c^S

j_c^D : The job to move container c from storage bay l_c^S to departure transfer station l_c^D

C_j^R : The set of containers that must be rehandled in job j

$d_{m,j}^T$: The total distance that machine m travelled for job j.

Event Variables

t_c^{SAT} : The scheduled arrival time of container c at its arrival transfer station l_c^A

t_c^{AAT} : The time that container is placed down in the arrival transfer station l_c^A

t_c^{SDT} : The scheduled departure time of container c from its departure transfer station l_c^D

t_c^{DT} : The actual departure time of container c from its departure transfer station l_c^D

t_c^{PUCA} : The time that container c is picked up from the arrival transfer station l_c^A

t_c^{PDCD} : The time that container c is placed down in the departure transfer station l_c^D.

t_c^{PUCS} : The time that container c is picked up from the storage bay l_c^S to be taken to the departure transfer station l_c^D.

t_c^{PDCS} : The time that container c is place down in the storage bay l_c^S after leaving the arrival transfer station l_c^A.

$t_{c,j}^{PUCR}$: The time that container c is picked up for rehandling in order to reach job container j

$t_{c,j}^{PDCR}$: The time that container c is placed down after being rehandled in order to reach job container j

$t_{m,j}^{SJ}$: The time that machine m starts job j

$t_{m,l,j}^{MENL}$: The time that machine m enters location l for job j

$t_{m,l,j}^{MEXL}$: The time that machine m exits location l for job j

There are a number of assumptions that were made in order to simplify the model. They are as follows:

- Only standard 20 foot containers are considered in this model.
- Only straddle carriers will be considered in the model.
- The storage bays consist of only one row each.

- The horizontal and vertical velocity of the vehicles is considered to be constant.
- The berths, TTA, ROIT and RAIT are not explicitly included in this model.
- The straddle carriers always take the shortest distance between the storage bays and transfer stations, and delays caused by traffic are ignored.
- The containers that are being rehandled in the storage bay are moved to another stack and left there until they depart from the storage bay or rehandled in a future job.
- The stacks in the transfer stations are only one level high.
- The straddle carriers do not move horizontally when lifting or dropping a container and that when moving horizontal it carries the container at the highest possible level.

The model's objective function and constraints are defined as follows:

Minimise $$\sum_{m\in M}\sum_{j\in J} d^{T}_{m,j} \qquad (1)$$

Equation (1) calculates the total distance travelled by all of the machines and minimises this value. This will also minimise the amount of rehandling performed as well as help with maintaining the due dates for the containers.

Subject to

Capacity Constraints

$$U_l = h_{\max} n_l \qquad \forall l \in L^{SB} \qquad (2)$$

$$U_l = n_l \qquad \forall l \in L^{TS} \qquad (3)$$

$$\sum_{c\in C}\sum_{x=1}^{n_l}\sum_{y=1}^{h_{\max}} p_{c,l,x,y,t} \le U_l \qquad \forall t,\, l \in L^{SB} \qquad (4)$$

$$\sum_{c\in C}\sum_{x=1}^{n_l} p_{c,l,x,1,t} \le U_l \qquad \forall t,\, l \in L^{TS} \qquad (5)$$

Equation (2) and (3) calculates the capacity of the storage bays and locations. Equations (4) and (5) ensure that the capacity of these locations is not exceeded at any time.

Container Constraints

$$\sum_{l\in L^{SB}}\sum_{x=1}^{n_l}\sum_{y=1}^{h_{\max}} p_{c,l,x,y,t} + \sum_{l\in L^{TS}}\sum_{x=1}^{n_l} p_{c,l,x,1,t} + \sum_{m\in M} r_{m,c,t} \le 1 \qquad \forall t, c \in C \qquad (6)$$

$$\sum_{c\in C} p_{c,l,x,y,t} \le 1 \qquad \forall t, l \in L^{SB},\ 1 \le x \le n_l,\ 1 \le y \le h_{\max} \qquad (7)$$

$$\sum_{c\in C} p_{c,l,x,1,t} \le 1 \qquad \forall t, l \in L^{TS},\ 1 \le x \le n_l \qquad (8)$$

$$\sum_{c\in C} p_{c,l,x,y,t} \le \sum_{c\in C} p_{c,l,x,y-1,t} \qquad \forall t, l \in L^{SB},\ 1 \le x \le n_l,\ 1 < y \le h_{\max} \qquad (9)$$

$$x_c^A = \sum_{x=1}^{n_l} x p_{c,l,x,1,t} \qquad \forall c \in C,\ t = t_c^{AAT},\ l = l_c^A \qquad (10)$$

$$x_c^D = \sum_{x=1}^{n_l} x p_{c,l,x,1,t} \qquad \forall c \in C,\ t = t_c^{PDCD},\ l = l_c^D \qquad (11)$$

$$x_{c,t}^S = \sum_{x=1}^{n_l} x \sum_{y=1}^{h_{\max}} p_{c,l,x,y,t} \qquad \forall t, c \in C,\ l = l_c^S \qquad (12)$$

$$y_{c,t}^S = \sum_{y=1}^{h_{\max}} y \sum_{x=1}^{n_l} p_{c,l,x,y,t} \qquad \forall t, c \in C,\ l = l_c^S \qquad (13)$$

Equation (6) ensures that a container can only be in one place at a time, either in a location or being carried by a machine. Equation (7) and (8) ensures that there is at most one container in each position in the storage bays and transfer stations, while Equation (9) ensures that if containers in the storage bay are not on the bottom level, they must have a container beneath them. Equations (9) to (13) retrieves the positions the containers are stored in the transfer stations and the storage bays.

Machine Constraints

$$\sum_{m \in M} q_{m,l,t} \le 1 \qquad \forall t, l \in L \qquad (14)$$

$$\sum_{l \in L} q_{m,l,t} \le 1 \qquad \forall t, m \in M \qquad (15)$$

$$\sum_{c \in C} r_{m,c,t} \le 1 \qquad \forall t, m \in M \qquad (16)$$

Equation (14) ensures that each location can have at most one machine occupying it at all times, while Equation (15) ensures that a machine cannot be in more than one location at a time. Equation (16) ensures that a machine is only carrying one container at a time.

Event Constraints

$$t_c^{SAT} \le t_c^{AAT} \qquad \forall c \in C \qquad (17)$$

$$t_c^{AAT} \le t_c^{PUCA} \qquad \forall c \in C \qquad (18)$$

$$t_c^{DT} = \max\left(t_c^{SDT}, t_c^{PDCD}\right) \qquad \forall c \in C \qquad (19)$$

$$t_c^{PDCS} < t_{c,j}^{PUCR} < t_c^{PUCS} \qquad \forall c \in C \qquad (20)$$

$$t_c^{PDCS} < t_{c,j}^{PDCR} < t_c^{PUCS} \qquad \forall c \in C \qquad (21)$$

$t_{m,l,k}^{MEXL} < t_{m,j}^{SJ}$ $\qquad \forall m \in M,\ j \in J_m$, k is the job performed before j (22)

$t_{m,i,j}^{MENL} \ge t_{m,j}^{SJ} + \frac{D_{i,l}}{v^H}$ $\qquad \forall m \in M,\ j \in J_m^A,\ c \mid j = j_c^A$, l is the location the machine is at before job $j,\ i = l_c^A$ (23)

$$t_c^{PUCA} = t_{m,i,j}^{MENL} + \frac{\{E_{i,l}(n_i - x_c^A) + (1 - E_{i,k})x_c^A\}L}{v^H} + \frac{h_{\max} H}{v^V} \qquad \forall m \in M, j \in J_m^A, c \mid j = j_c^A,$$

l is the location the machine is at before job j, $i = l_c^A$ (24)

$$t_{m,i,j}^{MEXL} = t_c^{PUCA} + \frac{\{E_{i,k}(n_i - x_c^A) + (1 - E_{i,k})x_c^A\}L}{v^H} + \frac{h_{\max} H}{v^V} \quad \forall m \in M, j \in J_m^A, c \mid j = j_c^A, i = l_c^A,$$

$k = l_c^S$ (25)

$$t_{m,k,j}^{MENL} \geq t_{m,i,j}^{MEXL} + \frac{D_{i,k}}{v^H} \quad \forall m \in M, j \in J_m^A, c \mid j = j_c^A, i = l_c^A, k = l_c^S \tag{26}$$

$$t_c^{PDCS} = t_{m,k,j}^{MENL} + \frac{\{E_{k,i}(n_k - x_{c,t}^S) + (1 - E_{k,i})x_{c,t}^S\}L}{v^H} + \frac{(h_{\max} - y_{c,t}^S)H}{v^V} \qquad \forall m \in M, j \in J_m^A,$$

$c \mid j = j_c^A$, $i = l_c^A$, $k = l_c^S$, t is the time c is placed down in the storage bay. (27)

$$t_{m,k,j}^{MEXL} = t_c^{PDCS} + \frac{\min(x_{c,t}^S, n_k - x_{c,t}^S)L}{v^H} + \frac{(h_{\max} - y_{c,t}^S)H}{v^V} \quad \forall m \in M, j \in J_m^A, c \mid j = j_c^A, i = l_c^A,$$

$k = l_c^S$, $t = t_c^{PDCS}$ (28)

$$t_{m,i,j}^{MENL} \geq t_{m,j}^{SJ} + \frac{D_{i,l}}{v^H} \qquad \forall m \in M, j \in J_m^D, c \mid j = j_c^D,$$ l is the location the machine is at before job j, $i = l_c^S$ (29)

$$t_{b,j}^{PUCR} = \begin{cases} t_{m,i,j}^{MENL} + \dfrac{\{E_{i,l}(n_i - x_{c,t}^S) + (1 - E_{i,l})x_{c,t}^S\}L}{v^H} + \dfrac{(h_{\max} - y_{b,t}^S)H}{v^V} & b \text{ is first to be rehandled} \\ t_{b-1,j}^{PDCR} + \dfrac{|x_{c,t}^S - x_{b-1,s}^S|L}{v^H} + \dfrac{(2h_{\max} - y_{b-1,s}^S - y_{b,t}^S)H}{v^V} & \text{otherwise} \end{cases}$$

$\forall m \in M, j \in J_m^D, b \in C_j^R$, $c \mid j = j_c^D$, $t = t_{m,i,j}^{MENL}$, s is the time the previous rehandled container is placed down, l is the location the machine is at before job j, $i = l_c^S$ (30)

$$t_{b,j}^{PDCR} = t_{b,j}^{PUCR} + \frac{|x_{c,t}^S - x_{b,s}^S|L}{v^H} + \frac{(2h_{\max} - y_{b,t}^S - y_{b,s}^S)H}{v^V} \quad \forall m \in M, j \in J_m^D, b \in C_j^R,$$

$c \mid j = j_c^D$, $t = t_{m,i,j}^{MENL}$, s is the time b is placed down. (31)

$$t_c^{PUCS} = \begin{cases} t_b^{PDCR} + \dfrac{|x_{c,t}^S - x_{b,s}^S|L}{v^H} + \dfrac{(2h_{\max} - y_{c,t}^S - y_{b,s}^S)H}{v^V} & b \text{ exists} \\ t_{m,i,j}^{MENL} + \dfrac{\{E_{i,l}(n_i - x_{c,t}^S) + (1 - E_{i,l})x_{c,t}^S\}L}{v^H} + \dfrac{(h_{\max} - y_{c,t}^S)H}{v^V} & \text{otherwise} \end{cases} \quad \forall m \in M, j \in J_m^D,$$

$c \mid j = j_c^D$, b is the container above c, $t = t_{m,i,j}^{MENL}$, $s = t_{b,j}^{PDCR}$, l is the location the machine is at before job j, $i = l_c^S$ (32)

$$t_{m,i,j}^{MEXL} = t_c^{PUCS} + \frac{\{E_{i,k}(n_i - x_{c,t}^S) + (1 - E_{i,k})x_{c,t}^S\}L}{v^H} + \frac{(h_{\max} - y_{c,t}^S)H}{v^V} \quad \forall m \in M, j \in J_m^D, c \mid j = j_c^D, i = l_c^S, k = l_c^D, t = t_{m,i,j}^{MENL} \quad (33)$$

$$t_{m,k,j}^{MENL} \geq t_{m,i,j}^{MEXL} + \frac{D_{i,k}}{v^H} \quad \forall m \in M, j \in J_m^D, c \mid j = j_c^D, i = l_c^S, k = l_c^D \quad (34)$$

$$t_c^{PDCD} = t_{m,k,j}^{MENL} + \frac{\{E_{k,i}(n_k - x_c^D) + (1 - E_{k,i})x_c^D\}L}{v^H} + \frac{h_{\max} H}{v^V} \quad \forall m \in M, j \in J_m^D, c \mid j = j_c^D, i = l_c^S, k = l_c^D \quad (35)$$

$$t_{m,k,j}^{MEXL} = t_c^{PDCD} + \frac{\min(x_c^D, n_k - x_c^D)L}{v^H} + \frac{h_{\max} H}{v^V} \quad \forall m \in M, j \in J_m^D, c \mid j = j_c^D, i = l_c^S, k = l_c^D. \quad (36)$$

$$d_{m,j}^T = D_{l,i} + D_{i,k} + \{(E_{i,l} + E_{i,k})(n_i - x_c^A) + (2 - E_{i,l} + E_{i,k})x_c^A\}L + \{E_{k,i}(n_k - x_{c,t}^S) + (1 - E_{k,i})x_{c,t}^S + \min(x_{c,t}^S, n_k - x_{c,t}^S)\}L + 2\{2h_{\max} - y_{c,t}^S\}H$$

$\forall m \in M, j \in J_m^A, c \mid j = j_c^A$, l is the location the machine is at before job j, $i = l_c^A$, $k = l_c^S$, $t = t_c^{PDCS}$ (37)

$$d_{m,j}^T = D_{l,i} + D_{i,k} + \{(E_{i,l} + E_{i,k})(n_i - x_{c,t}^S) + (2 - E_{i,l} + E_{i,k})x_{c,t}^S\}L + (h_{\max} - y_{c,t}^S)H + 2L\sum_{b \in C_c^R} \left|x_{c,t}^S - x_{b,s}^S\right| + H\sum_{b \in C_c^R}(2h_{\max} - y_{c,t}^S - y_{b,s}^S) + \{E_{k,i}(n_k - x_c^D) + (1 - E_{k,i})x_c^D + \min(x_c^D, n_k - x_c^D)\}L + 2h_{\max} H$$

$\forall m \in M, j \in J_m^D, c \mid j = j_c^D$, l is the location the machine is at before job j, $i = l_c^S$, $k = l_c^D$, $t = t_{m,i,j}^{MENL}$, s is the time that the rehandle containers were placed down after being picked up from the stack containing the job container (38)

Equation (17) ensures that a container is not placed in its arrival transfer station before its scheduled arrival time, while Equation (18) ensures that a container is not picked up from the arrival transfer station before it is placed down in the location. Equation (19) finds the departure time of each container. Equations (20) and (21) ensures that a container is first placed down in the storage bay, then rehandled for other jobs, then finally picked up from the storage bay. Equations (22) – (36) track the time of the events for a job, while Equations (37) and (38) track the distance travelled by the machine during a job.

4. SOLUTION TECHNIQUES

Due to the timely nature of this problem, a good, near-optimal solution that is found in a reasonable amount of time is preferred to an optimal solution that would take an eternity to calculate. Therefore, a number of Meta-heuristic techniques have been implemented and tested on a benchmark problem. These techniques tested were Genetic Algorithm (GA), Tabu Search (TS), and a hybrid of TS and Simulated Annealing (SA), which will be named TabuSA for the remainder of this paper.

GA is part of the family of Evolutionary Algorithms, all of which take their philosophy from Darwin's theory of natural selection and from genetics. Figure 1 outlines the GA process used in this paper.

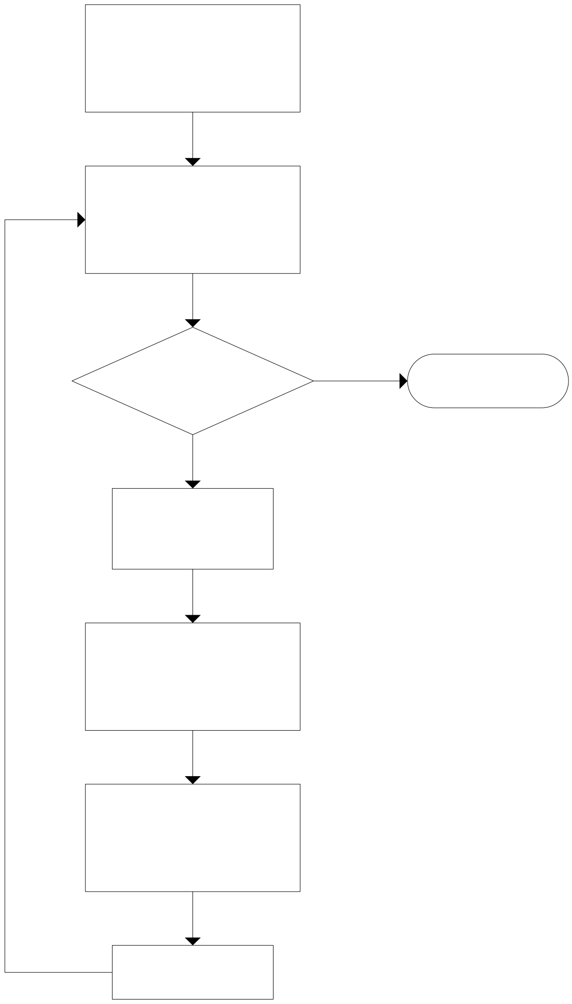

Figure 1. Genetic Algorithm Process

The GA begins with the creation of an initial group of solutions or *chromosomes* called a **population**, and this population forms the initial **generation**. The objective value or *fitness* of each solution is calculated, as well as their infeasibility, using an appropriate technique. If the termination criteria are met then the GA terminates. Otherwise a new generation is formed from this initial generation by performing the following process. Firstly, a specified number of the best solutions from the current generation are added to the new generation, which is known as **Elitism**. Once this is completed, pairs of chromosomes are chosen from the current generation, and combined using a **crossover** operation to generate children chromosomes, which are added to the new generation. In order to maintain some diversity in the population, these children chromosomes are then randomly **mutated** by altering some of the values in the solution. Finally, the fitness and infeasibility of these new chromosomes are calculated, and the process repeats itself until the termination criteria are met.

Simulated Annealing has origins in the field of material science and physics, as it is similar to the physical annealing process of condensed matter. An initial solution is generated either randomly or by using a constructive heuristic. This solution is set as the best solution. A group of solutions, called a **neighbourhood**, are obtained from the current solution by modifying it in a well-defined manner. The solution with the best objective value in the neighbourhood is found to be compared against the current solution and current best solution. If the new solution has a better objective value than the current solution, it will replace it, and if it is better than the best solution found so far, the new solution will become the best solution. However, if it has a worse fitness than the current solution, it will only replace the current solution with some probability. This probability, also known as a **cooling parameter**, gradually becomes smaller as the process continues towards termination. SA allows worse solutions to replace the current solution near the beginning of the annealing process, thus allowing the SA to move in and out of local minima, while providing the ability of refining the best solution towards the end of the annealing process.

Tabu Search is similar to Simulated Annealing in that it can move in and out of local minima; however the decision of whether a solution should replace the current solution is determined by a list of tabu alterations or mutations. As with Simulated Annealing, Tabu Search begins with an initial solution that is set as the current best solution. A neighbourhood of solutions are obtained through the mutation of the current solution. The best solution from this neighbourhood is chosen to replace the current solution. Once the new solution has replaced the current solution, the mutation that created the new solution from the old solution is added to a list of tabu mutations that can not be performed in future iterations. Usually for storage purposes, the tabu list is of a finite size, and once it becomes full, the oldest tabu mutation is removed from the list to allow a new tabu mutation to be added. The tabu list is used to prevent cycles from occurring when in one iteration, the tabu search moves away from a local minima, then in a future iteration, the tabu search will return to that local minima. Since, in practice the tabu list is finite, cycles that are longer than the list size may still occur.

One of the disadvantages of using Tabu Search is that the probability of choosing a worse solution is not guaranteed to reduce towards the end of the search process. In order to guarantee this reduction, TabuSA was developed in this paper. The tabu-list from Tabu Search is used in conjunction with the cooling parameter from Simulated Annealing. This combines the tabu-list's ability to prevent cycles from occurring with the cooling parameter's ability to refine the solution as more iterations are performed.

The optimisation program was implemented in C++ using object-oriented techniques. The representation of a solution to the model was developed to be used in the crossover and mutation operations of the GA, TS and TS/SA hybrid without too much trouble, while still providing enough information to simulate and calculate the objective value and infeasibility of the solution. The solution is stored in an array of integers which are divided into groups that represent different variables in the model.

The majority of the running time of the optimisation program is spent running simulations for the calculation of the fitness and infeasibility of solutions. The simulation uses the values from the solution representation to initialise the various objects, and then iterates through the algorithm until all containers have passed the storage area.

5. RESULTS

The computer program was run on three test cases in order to compare the performance of the various solution techniques, as well as perform a sensitivity analysis. Each test case consisted of the following:

- 100 containers, with random inter-arrival times that were obtained from an exponential distribution, and random departure times obtained from a triangular distribution.
- 4 Transfer Stations, each with a capacity of 8 containers
- 20 Storage Bays, each with a capacity of 15 containers, 5 containers long by 3 containers high.

Figure 2 illustrates the layout that was used in the test cases. The tests were run with 2, 4, 6, 8 and 10 straddle carriers, to see the effect that changing the number of straddle carriers has on the result and the performance of the solver. The Genetic Algorithm were run on 2 2.8 GHz Pentium 4 PCs with 512 Mb of RAM, while the Tabu Search and Tabu SA tests were run on 1.7 GHz Pentium 4 PCs with 512 Mb of RAM. The first test for the Tabu Search test cases were run on a 2.4 GHz Pentium 4 PC with 512 Mb, and due to the difference in speed to the rest of the Tabu Search tests, these test will be ignored when comparing CPU times.

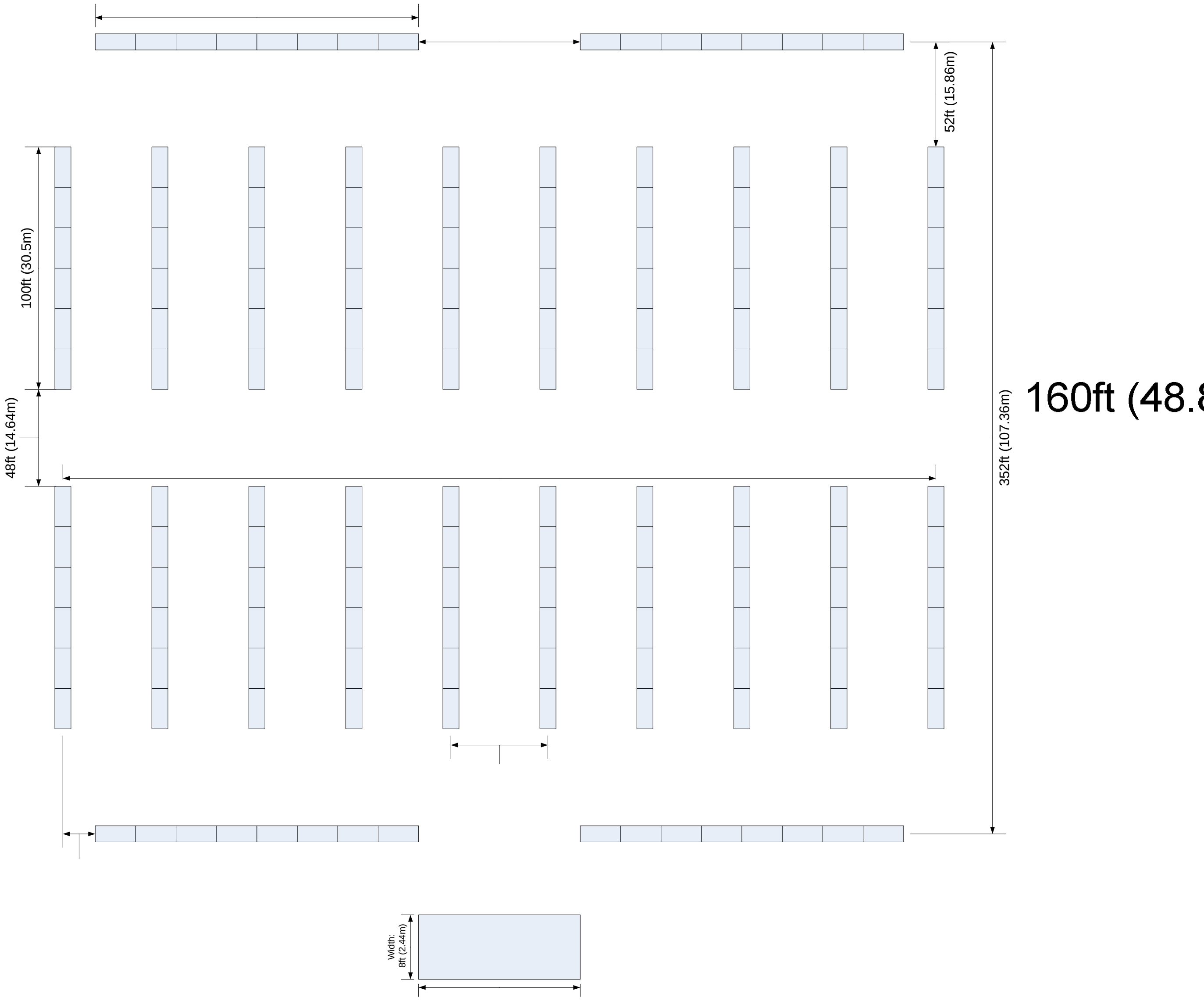

Figure 2. Test Problem Layout

Unfortunately, none of the solutions that were obtained during the tests satisfied all container departure time constraints, that is, there were some tardy containers. This was due to the fact the initial solutions were generated randomly and some of the containers arrived at the departure transfer station late. A similar problem also occurred in a previous research project, and it was improved by generating the initial solutions using a problem specific greedy generation heuristic. When a generation heuristic is implemented, it will have an algorithm similar to the simulation algorithm, however instead of allocating machines to containers, the locations will be checked for containers that can leave their current destination, and these would be allocated to the machines that can get them to their destination in the fastest time. The checks that were performed when deciding whether a job is valid would also have to be used in this heuristic.

The Genetic Algorithm tests were run with the following parameters:

- 50 Generations
- 50 Chromosomes per generation
- Single Point Crossover
- Elitism of 10 solutions

- Probability of Crossover 80%
- Probability of Mutation 1%

The Tabu Search tests were run with the following parameters

- 100 Iterations
- Tabu List Size – 25 mutations
- Maximum Neighbourhood size – 25 Solutions
- 200 Initial Solutions

The Tabu SA tests were run with the same parameters as the Tabu Search tests, with the additional cooling parameter set at 0.95

Figures 3, 4 and 5 are graphs of the infeasibility for test 1 as it decreases over the iterations for the Genetic Algorithm, Tabu Search and Tabu Search/Simulated Annealing Hybrid techniques respectively. The plots for the other tests are omitted, but they demonstrate very similar behaviour to these plots.

The results shown in these figures suggest that the Genetic Algorithm performs better than the other two solution techniques, as it manages to reduce the final solution infeasibility further from the initial solution infeasibility. The Genetic Algorithm initial solution infeasibility is higher than the Tabu Search and TS/SA Hybrid in most tests, since it only begins with 50 solutions in its initial iteration, while the other two begin with 200 solutions, and therefore they have a better chance of producing an initial solution with a smaller infeasibility. However, the GA manages to reduce the best infeasibility to below 200000 in most cases, and in fact, some of the tests that were run produced a final infeasibility below 100000. The Tabu Search and TS/SA hybrid, on the other hand, could only reduce the infeasibility from around 400000 for the initial infeasibility to around 300000. This implies that the GA performs better in the large search space that this problem provides, and converges towards a solution more quickly within the large search space. The opposite was found with previous research, where the Tabu Search and TS/SA performed better within the smaller feasible region of that problem, as they are better at refining solutions that are already reasonably good.

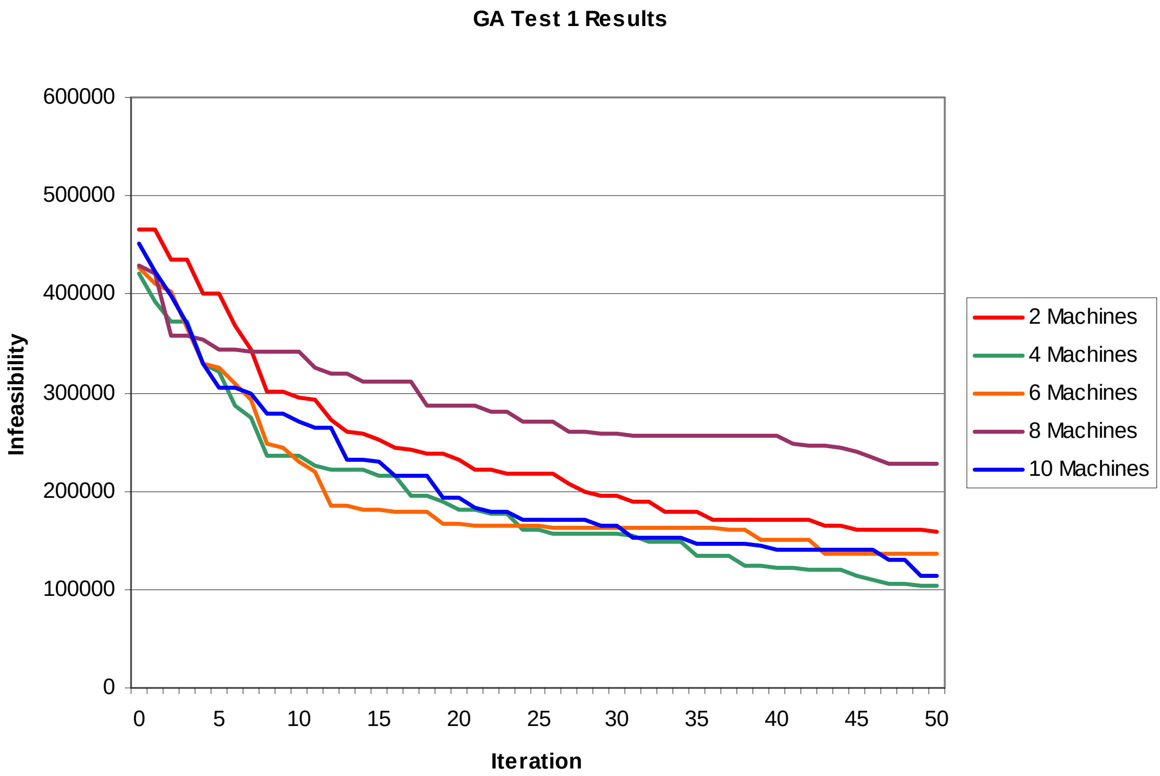

Figure 3. GA Test Case 1 Results

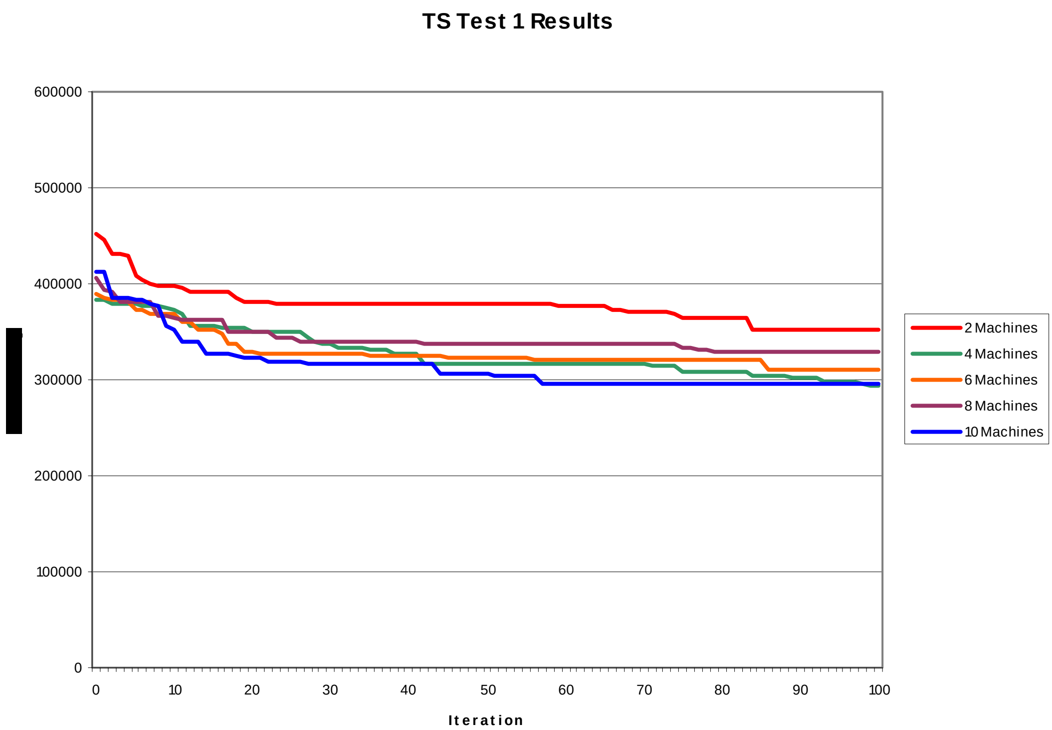

Figure 4. TS Test Case 1 Results

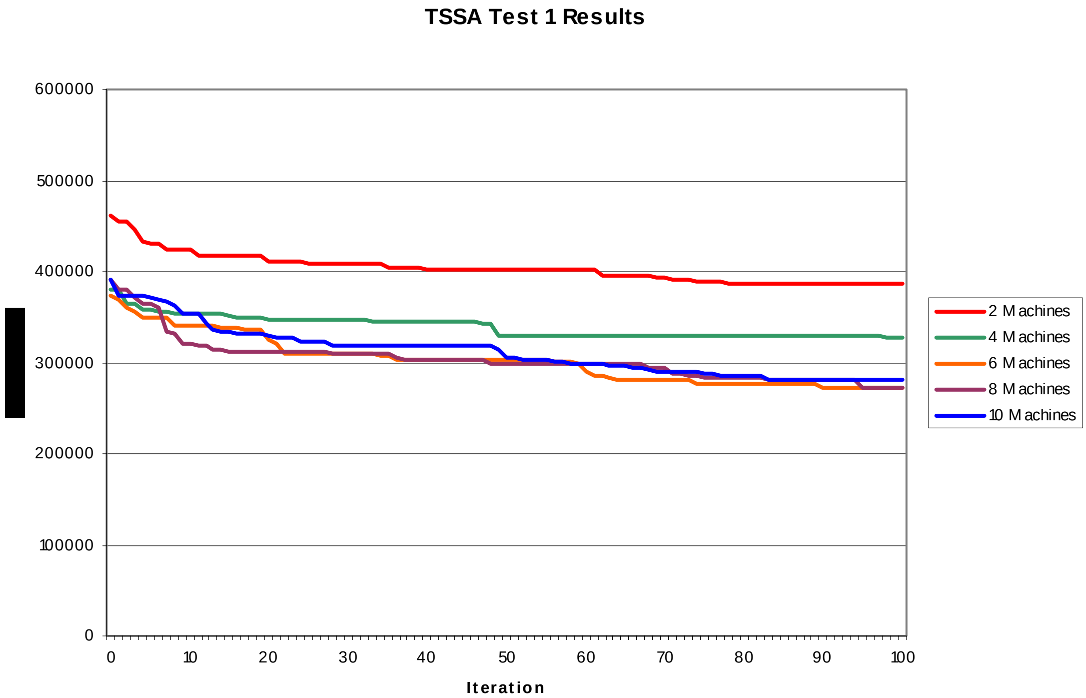

Figure 5. TabuSA hybrid Test Case 1 Results

In all of the tests except GA Test 1, the tests that only used two straddle carriers had higher infeasibility values over the majority of the iterations, which suggests that changing the number of machines may have a minor affect on the infeasibility obtained. Above 4 machines, however, there is no discernable trend in the infeasibility obtained, so it is possible that this may be related to the number of transfer stations in the problem, which in this case is four. The reasoning behind this is that with more than 4 straddle carriers, there is at least one straddle carrier per transfer station, and therefore it is less likely that a container will be delivered to its destination late. In fact, having more straddle carriers than transfer stations may have a negative effect on the container tardiness, as straddle carriers may have to queue up to enter the transfer station.

Figures 6, 7 and 8 are graphs of the cumulative CPU time for test 1 for the Genetic Algorithm, Tabu Search and Tabu Search/Simulated Annealing Hybrid techniques respectively. As before, the plots for the other tests are omitted, but they demonstrate very similar behaviour to these plots.

The results in Figures 6, 7 and 8 indicate that the CPU time is strongly affected by the number of machines used in the problem. The relationship between the number of machines and the CPU time appears to be logarithmic, with the difference between CPU times decreases as the number of machines used increases. This can be explained by how the simulation works. It requires checks for a number of jobs for each available straddle carriers, and these checks are computationally expensive. The number of checks required increases as the number of straddle carrier increases, and therefore the CPU time will increase.

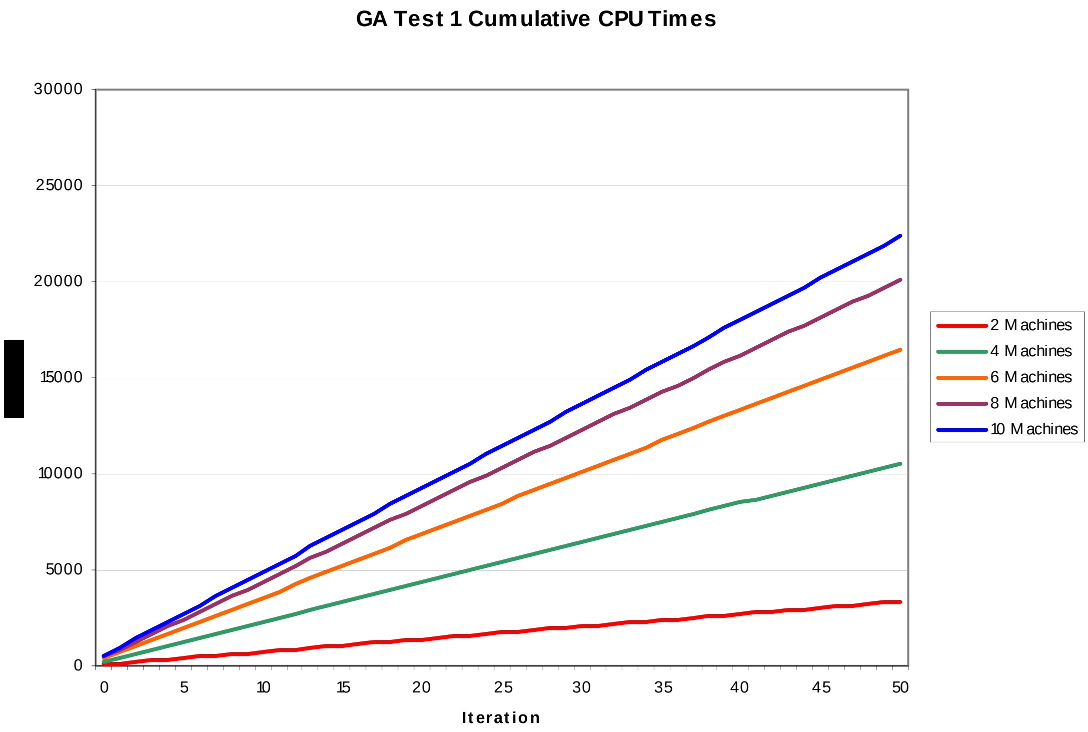

Figure 6. GA Test Case 1 Cumulative CPU Time

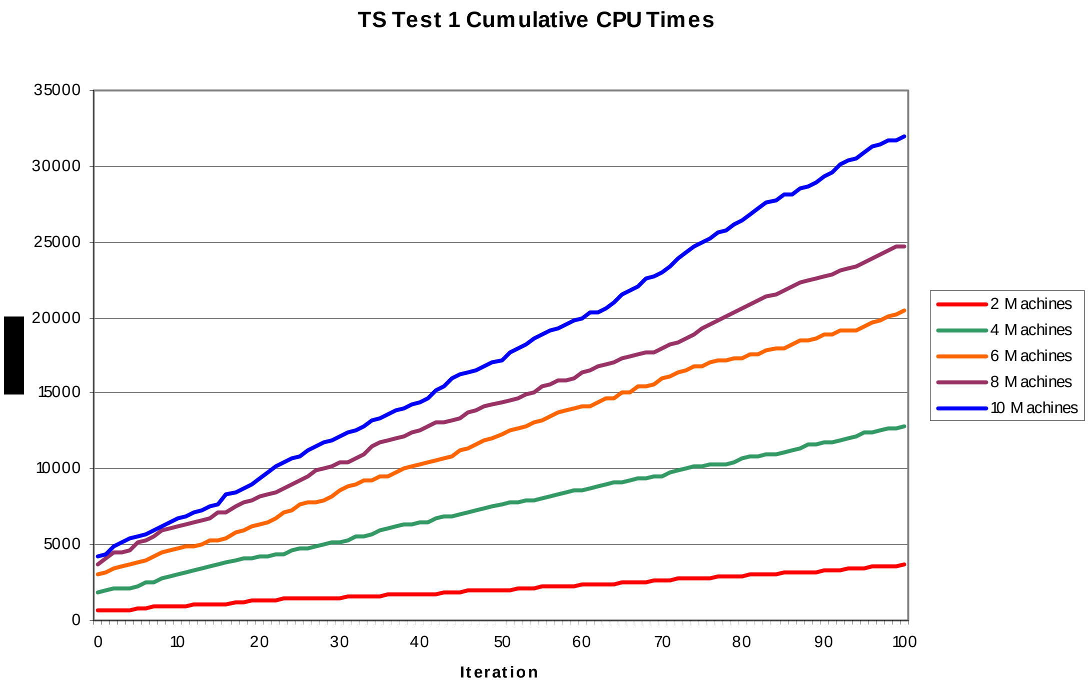

Figure 7. TS Test Case 1 Cumulative CPU Time

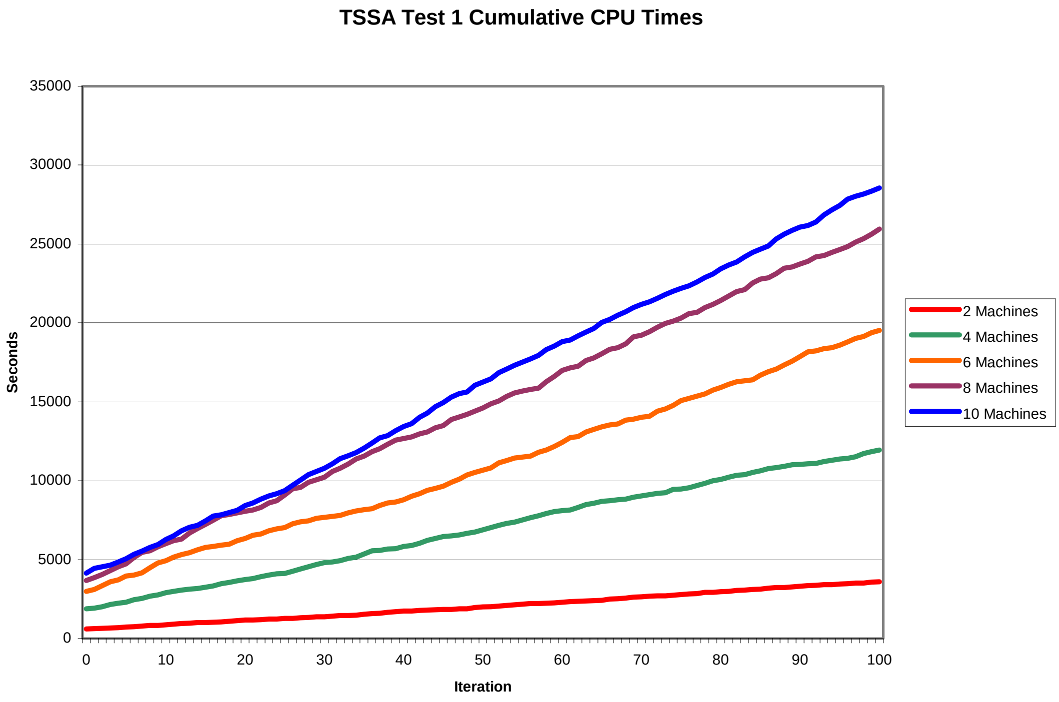

Figure 8. TSSA Test Case 1 Cumulative CPU Time

The GA graphs have a constant gradient while the Tabu Search and TS/SA Hybrid graphs have an increasing gradient that is not constant. This is due to the fact the GA must simulate a constant number of solutions during every iteration, while the Tabu Search and TS/SA Hybrid may simulate differing number of solutions depending on the point around which they are generating a neighbourhood. The Tabu Search and TS/SA hybrid also have larger initial iteration CPU times than the Genetic Algorithm due to the fact that they must simulate 200 solutions as opposed to only 50 solutions.

If the solution techniques were allowed to continue past the iterations that the tests were run for, they may have produced feasible solutions. However the time taken to compute this feasible solution by this method would take too much time, and finding a feasible solution is not guaranteed. Therefore, the generation heuristic mentioned earlier will have to be implemented before this program would be able to be used in practical problems. Once this heuristic is implemented, further tests will have to be performed to measure the performance of the three solution techniques, and the sensitivity of the best solutions to changes in the different input parameters.

6. CONCLUSION

The aim of this paper was to model the problem of storing and handling containers within the storage area of the MMCT, and to develop a computer program that could solve the model for non-trivial problems.

The model that was developed was simplified from what occurs in practice in order to focus on the main problems of storage allocation of containers, machine allocation to container jobs and rehandling of containers. The assumptions that were taken will have to be removed in order for this model to be used in a practical application. However the model that was developed does contain the main constraints that will be required in a realistic model, such as the capacity constraints, the container constraints and machine constraints.

The program that was developed does generate solutions for the model for non-trivial problems. Unfortunately, due to the initial solutions being generated randomly, these solutions do not satisfy all of the container departure constraints, and feasible solutions could not be obtained within the limited time frame that would be allowed in practice. This problem can be remedied by generating the initial solutions using a problem-specific generation heuristic that would generate feasible solutions, therefore allowing the solution techniques to concentrate on improving those feasible solutions. From the results that were obtained, the GA performed better than the TS and TS/SA Hybrid in obtaining solutions that were less infeasible. However from experience gained from a previous project, once feasible solutions are found, the TS and TS/SA Hybrid perform better in refining the feasible solutions. The generation heuristic should be implemented before the program should be used in practise, and further test should be run after this is completed to observe the performance of the solution techniques and to test the sensitivity of the solution to changes in the parameters, such as number of straddle carriers or maximum stack height.

Future work that should be performed in this area of research and in particular on the model and program developed include:

- Implementing the generation heuristic to ensure feasible solutions are found
- Improving the model by removing the assumptions that were taken. For instance, other types of yard machines should be considered, such as gantry cranes and AGVs. Also, the paths between the locations in the storage area could be modelled in a more sophisticated to allow for more general storage area layouts and to take traffic into consideration.
- Profiling of the code in order to optimise the sections that the program spends the most time executing, therefore reducing the overall computation time and thus producing more optimal solutions in a timely manner.
- Parallelisation of the solution techniques to take advantage of new multi-core CPU technology, therefore allowing multiple solutions to be simulated at the same time and thus producing more optimal solutions in less time.
- Integration of the refined model and program into a system that optimises the overall MMCT system.

REFERENCES

Cao, B. and Uebe, G. (1993), An algorithm for solving capacitated multi-commodity p-median transportation problems. *Journal of the Operational Research Society*, **44(3)**, 259-269.

Chen, T. (1999), Yard operations in the container terminal – a study in the ‘unproductive moves’. *Maritime Policy & Management*, **26(1)**, 27-38

Cheung, R.K., Li, C.-L. and Lin, W. (2002), Interblock crane deployment in container terminals. *Transportation Science*, **36(1)**, 79-93.

Chung, Y.G., Randhawa, S.U. and McDowell, E.D. (1988), A simulation analysis for a transtainer-based container handling facility. *Computers & Industrial Engineering*, **14(2)**, 113-125.

De Castilho, B. and Daganzo, C.F. (1993), Handling strategies for import containers at marine terminals. *Transportation Research B*, **27(2)**, 151-166.

Kim, K.H. and Kim, H.B. (1999a), Segregating space allocation models for container inventories in port container terminals. *International Journal of Production Economics,* **59**, 415-423.

Kim, K.H. and Kim, K.Y. (1999b), An optimal routing algorithm for a transfer crane in port container terminals. *Transportation Science,* **33(1)**, 17-33

Kim, K.H. and Kim, H.B. (2002), The optimal sizing of the storage space and handling facilities for import containers. *Transportation Research B*, **36**, 821-835.

Kim, K.H., Kang, J.S. and Ryu, K.R. (2004), A beam search algorithm for the loading sequence of outbound containers in port container terminals. *OR Spectrum*, **26**, 93-166.

Kim, K.H., Park, Y.M. and Ryu, K.R. (2000), Deriving decision rules to locate export containers in container yards. *European Journal of Operations Research,* **124**, 89-101.

Kim, K.H. and Park, Y.T. (2002) A note on a dynamic space-allocation method for outbound containers. *European Journal of Operations Research*, **148**, 92-101.

Kozan, E. and Preston, P.(1999), Genetic Algorithms to Schedule Container Transfers at Multimodal Terminals, *International Transactions in Operational Research*, **6.3**, 311-329.

Kozan E. (2000), Optimising container transfers at multimodal terminals, *Mathematical and Computer Modelling* **31**(10-12), 235-243.

Preston, P. and Kozan, E. (2001a), An Approach to Determine Storage Locations of Containers at Seaport Terminals, *Computers and Operations Research,* **28**, 983-995.

Preston P. and Kozan E. *(2001b), A Tabu Search Technique Applied to Scheduling Container Transfers", Transportation Planning and Technology.* **24.2**, 135-154.

Ng, W.C. (2005), Crane scheduling in container yards with inter-crane interference. *European Journal of Operations Research*, **164**, 64-78.

Steenken, D., Voss, S. and Stahlbock, R. (2004), Container terminal operation and operations research – a classification and literature review. *OR Spectrum*, **26**, 3-49.

Taleb-Ibrahimi, M., De Castilho, B. and Daganzo, C.F. (1993), Storage space vs handling work in container terminals. *Transporation Research B,* **27**, 13-32.

Vis, I.F.A. and De Koster, R. (2003), Transhipment of containers at a container terminal: an overview. *European Journal of Operations Research*, **147(1)**, 1-16.

Zhang, C., Wan, Y.W., Liu, J. and Linn, R.J. (2002), Dynamic crane deployment in container storage yards. *Transportation Research B,* **36**, 537-555.