Building Agent-Based Models of Seaport Container Terminals

José M. Vidal
Computer Science and Engineering Dept.
University of South Carolina
Columbia, SC, 29208
vidal@sc.edu

Nathan Huynh
Civil and Environmental Engineering Dept.
University of South Carolina
Columbia, SC, 29208
huynh@cec.sc.edu

ABSTRACT

Agent-based models are increasingly being used to simulate and analyze various transportation problems, from traffic flow [15] to air traffic control [1]. One transportation industry that has not received as much attention from the multi-agent systems community is seaport container terminals. It can be argued that the operations that take place at a container terminal are as complex as that of an airport. A seaport container terminal faces a myriad of operational challenges such as optimizing berth space, minimizing ship turnaround time, maximizing use of resources, and reducing wait time of drayage trucks. Due to environmental concerns, terminal operators and port planners are focusing on the problem of reducing the in-terminal wait time of drayage trucks. In this paper, we present our multiagent model of a container yard operation, its implementation using NetLogo, and some initial test results. We model yard cranes as opportunistic utility-maximizing agents using several different utility functions for comparison purposes. By using a representative layout of a terminal our simulation model allows us to analyze the behavior of the cranes and evaluate the collective performance of the system. We demonstrate that it is possible to build a realistic and useful model of yard crane operation. Our test results show that utility functions that give higher precedence to nearby trucks lead to much better results than those that favor serving trucks on a mostly first-come first-serve order.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

General Terms

Experimentation

Keywords

Simulation techniques, tools and environments, emergent behavior, yard cranes, seaport container terminals

1. INTRODUCTION

Agent-based models have been often used to analyze automobile and air traffic. In this paper we continue that trend, but focus on an entirely topic: supply chain and logistics. In particular, we build an agent-based model of a seaport container terminal. These terminals facilitate the movement of containers between the sea and land. Once import containers are discharged from a vessel, they are stored in a container yard (see Figure 1). Rubber-tired gantry cranes (see Figure 2) or commonly known as "yard cranes" then move these containers onto the trucks. The problem faced by the crane operators is deciding which truck to service first since many arrive at the same time. The process whereby a truck goes to a seaport terminal to pick up or deliver a container with both the trip origin and destination in the same urban area is known as *drayage*.

Drayage activities play an important role in supply chain and logistics. From seaport terminals, drayage drivers and trucks transport import containers to first receivers where consolidation, stripping, transfers, and intermodal activities are undertaken. They also deliver containers to final receivers directly or via key rail intermodal terminals across the nation. This process is reversed for export containers. Drayage operations are now widely recognized as a critical emissions, congestion, and capacity issue for major container ports and rail intermodal terminals. Public agencies are rapidly developing policies and programs to reduce related emissions.[1] Concurrently, drayage firms and terminal operators are working to improve drayage operations that are highly inefficient at the present. Despite the relatively short distance of the truck movement compared to the rail or barge haul, drayage accounts for a large percentage (between 25% and 40 %) of origin to destination expenses [9]. In turn, high drayage costs seriously affect the profitability of an intermodal service.

The seaport container terminals have long been identified as bottlenecks and sources of delay for port drayage. The time drayage trucks spent in the queue at the entry gate, container yard, and exit gate are often exceedingly long during peak times at busy terminals. Drayage trucks are diesel-fueled, heavy-duty trucks that transport containers, bulk, and break-bulk goods to and from ports and intermodal rail yards to other locations[2]. Truck idling in the queues is a contributing source of emissions and noise at terminals. High truck turn time is the result of demand ex-

Cite as: Building Agent-Based Models of Seaport Container Terminals, José M. Vidal and Nathan Huynh, *Proc. of 6th Workshop on Agents in Traffic and Transportation*, Kluegl, Ossowski, Chaib-Draa and Bazzan (eds.), May, 11, 2010, Toronto, Canada

[1]For example see `www.epa.gov/cleandiesel`.

[2]See the California Environmental Protection Agency `www.arb.ca.gov/msprog/onroad/porttruck/drayage truckfactsheet.pdf`

ceeding supply. Truck turn time refers to the time it takes a drayage truck to complete a transaction such as picking up an import container or dropping off an export container. It is a measure of a terminal's efficiency in receiving and delivering containers. For terminals that stack their containers, demand is mainly the number of drayage trucks coming to the terminal to pick up or drop off containers. Supply is the number of yard cranes available to serve these drayage trucks. Supply is typically low on high volume vessel days because the majority of the yard cranes are assigned to work the vessel. In such a scenario, drayage drivers must wait for a longer period of time before a yard crane is available to perform the load or unload move. This waiting process can take a considerable amount of time.

The solution of adding more yard cranes to reduce truck turn time may seem obvious for terminals that stack their containers. However, the high initial investment, plus maintenance and operating costs of these cranes often prohibit terminals from freely buying more. Also, once a drayage truck arrives at its destination in the yard, its turn time is not only dependent on the number of cranes available, but also the service strategy in which the cranes follow. To date, no study has adequately examined the effect of crane service strategy on truck turn time. The challenging issues inherent in this problem, coupled with the limitation of existing research, motivate this study. In addition, this study addresses the practical challenges of increasing supply chain efficiency while reducing the carbon footprint. Specifically, this study investigates how to deploy yard cranes in an effective manner to reduce drayage trucks in-terminal wait time. Reducing the drayage trucks in-terminal dwell time is equivalent to reducing local and regional particulate matter (PM 2.5), nitrogen oxides (NOx), and greenhouse gas (GHG) emissions. PM 2.5 emissions from diesel engines are recognized by the Environmental Protection Agency (EPA) as a serious health issue.

The following describes the study's innovative, decentralized approach to modeling yard cranes by using agent-based modeling and utility maximization to investigate the effectiveness of different crane service strategies. While agent-based models have been widely used in many different discipline, they are relatively unexplored in the area of drayage and port operations.

2. LITERATURE REVIEW

Much of the research directly related to yard cranes' work schedule has been carried out using mathematical programming techniques (e.g integer programs or mixed integer programs). As such, these studies seek to optimize the work flow of cranes for a given set of jobs with different ready times in the yard. The "jobs" considered vary from study to study, and they could be either drayage trucks, or other yard handling equipment such as prime movers and internal transfer vehicles. Given that the scheduling problem is NP-complete, many studies proposed algorithms or heuristics in order to solve the real-world large-scale problem in a reasonable amount of time, including dynamic programming-based heuristic [10], branch and bound algorithm [11], Lagrangean relaxation [19], and simulated annealing [7]. In the study by Kim et al. [6], a simulation study was performed to compare the performances of several heuristic rules:

First-come-first-serve: trucks are served in the order of their arrival time at the yard. Uni-directional travel: a yard crane travels in one direction and serves trucks until there are no more trucks remaining in the direction of the travel. After serving all the trucks in the direction of travel, the yard crane starts to travel in the opposite direction. Nearest truck first: a yard crane serves the truck that is located nearest to it. Shortest processing time: a yard crane serves the truck with the shortest transfer time, which is the sum of the travel time and the time for transferring the corresponding container to and from the truck. The transfer time includes the time for re-handling containers on top of the target container in the case of a delivery operation.

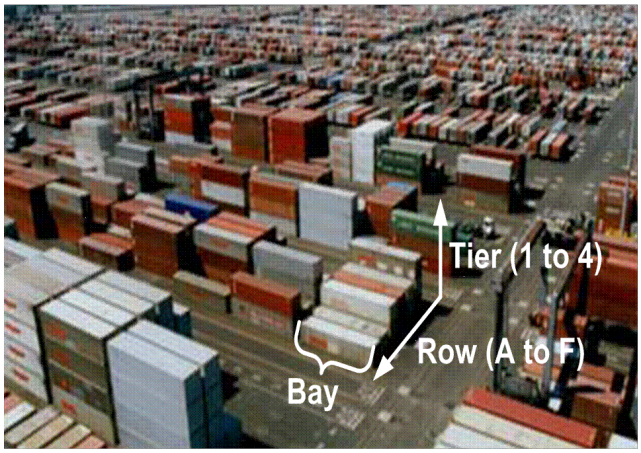

Figure 1: Illustration of bay, row, and tier in a yard block.

This study differs from the aforementioned mathematical programming related work in several ways. First, it takes a decentralized view instead of a centralized one. That is, the resulting cranes work flow is not governed by one optimal schedule. Rather the work flow stems from the individual decisions made by the crane operators. Second, it does not make any assumption regarding the ready times of the jobs. In this study, the number of drayage trucks that arrive to the yard is assumed to be Poisson distributed. Lastly, this study relies on agent-based simulation instead of a mathematical program. The agent-based feature also differentiates this study from the work of Kim et al. [6]. Moreover, each agent (i.e. crane operator) makes his decision based on a utility and not a prescribed heuristic rule.

Within the agent-based modeling community there have been some attempts at building simulations of container ports, but these address different parts of the problem. For example, in [2] the authors try to find optimal solutions for the placement of containers in the yard while assuming that the cranes use a fixed policy. In contrast, our research focuses on finding the optimal strategies for the cranes to use to minimize the trucks' wait time given random truck arrivals. Some preliminary work on simulating the ships and their allocation is presented in [12], and similarly in [14]. Most recently, the work of Henesey *et al* investigates the movement of containers from the ship into the yard [5], and looks at various policies for the sequencing of ships, berth allocation, and the use of simple stacking rules [4]. Their SimPort implementation demonstrates that a sophisticated agent-based simulation of a container port can be used to make prescriptive recommendations on how to manage the

system. Our research differs from these attempts in that we do not try to model the full system, from ships to trucks, nor do we try to make recommendations on how the complete system would work best. Instead, we focus on one small part of the problem—the yard cranes' service strategies—which we believe can be improved. We believe it is unlikely that a seaport will completely change their workflow because of the results of a simulation. On the other hand, small incremental changes that have been shown to have immediate positive effects on the bottom line are likely to be adopted. Our research aims to work within the real-world constraints of a working seaport, improving its overall efficiency via continuous small improvements.

There has also been some research done in building agent-based models of traffic system. For example, some have built simulations of automated intersections [1], or of air traffic control systems [15, 16], or studied other agent-based methods for solving the urban traffic problem [3, 17]. Our simulations thus take us one step further towards being able to build complete models of transportation systems, including vehicular traffic, freight, and intermodal facilities.

3. PROBLEM DESCRIPTION

A typical drayage move involves either a delivery of an export container to the seaport terminal or pickup of an import container. A drayage driver arriving to pick up a loaded import container may encounter one of three basic systems.

1. At wheeled terminals the driver will simply locate and retrieve the container on its chassis in the parking area.
2. At stacked terminals, the driver will usually first retrieve a chassis and then position the chassis in the container storage stacks to receive the container from a lift machine (typically yard crane).
3. At some stacked and straddle carrier terminals, the drayage driver will retrieve a chassis and then proceed to a designated transfer zone. A lift machine then brings the container to the waiting driver.

At stacked terminals, the containers are stacked on top of one another in separate yard blocks. Each yard block has about 80 20-foot bays, each bay has 6 rows, and each row has 4 tiers (Figure 1). A yard block is used for storing import containers, export containers, or both. Import containers are typically stored in the available blocks designated for imports and where it is most convenient for the stevedores to facilitate the vessel operations. As import containers are discharged from a vessel, they are stacked in the allocated space without any segregation. Export containers, on the other hand, are methodically segregated by 1) vessel, 2) port of discharge, 3) size, and 4) weight. This is done so that when export containers are transferred from the yard to the vessel, no re-handling (i.e. reshuffling of containers to retrieve the desired one) is required. Note that both the import and export processes are done in a manner to minimize the turn-around time of vessels.

Most U.S. seaport terminals use rubber-tired gantry (RTG) cranes, often referred to as yard cranes, to load and unload containers in the yard blocks. Figure 2 shows a cross section view of a yard block, and it illustrates how a yard crane is positioned in a block. On any given day, the yard cranes are

Figure 2: Yard crane working the stacks.

assigned to either support the vessel operation or support the road operation. Vessel operation has higher priority, so the number of yard cranes available to support road operation is the total number of yard cranes available minus the number of yard cranes assigned to vessel operation. Road operation refers to the landside process where drayage trucks come to drop off export containers and/or pick up import containers. Vessel operation refers to the waterside process where import containers are transferred from a vessel to the yard and export containers are moved from the yard to the vessel.

A typical import process involves a drayage driver moving a loaded container from the seaport terminal to the consignee location and then returning an empty to the terminal. The process of taking a loaded container out of the terminal begins with the shipping line in charge of the container requesting drayage service. The manifest is transferred to the drayage company and at the same time to the terminal. The drayage company then creates a pickup order and subsequently dispatches the driver. In order to take a loaded container out of the terminal the driver first arrives at the terminal gate. At this stage, the driver must scan or show his driver's license and then provide the container number to the gate clerk. He must also specify whether he needs to pick up a chassis. If there are no issues with his transaction, the driver receives a pick-up ticket and is cleared to enter the terminal. If the driver does not need a chassis, he then proceeds to the pre-designated pick up area and waits to be serviced by a yard crane (Figure 3).

Depending on the availability of yard cranes and their service strategies, this wait can be a source of extensive delay. Once the yard crane arrives at the bay where the truck has been waiting, the crane operator must locate the requested container and must often re-handle other containers on top before reaching the target container. After the container is loaded onto his truck, the driver must verify that it is the correct container and undamaged. He then must lock the chassis and proceed to the radiation inspection station. After the radiation inspection by Customs and Border Protection (CBP), the driver scans or shows the pick-up ticket and waits for the clerk to perform the damage inspection of the container and issues an Equipment Interchange Report (EIR), ending the out procedure and allowing the truck to exit the terminal.

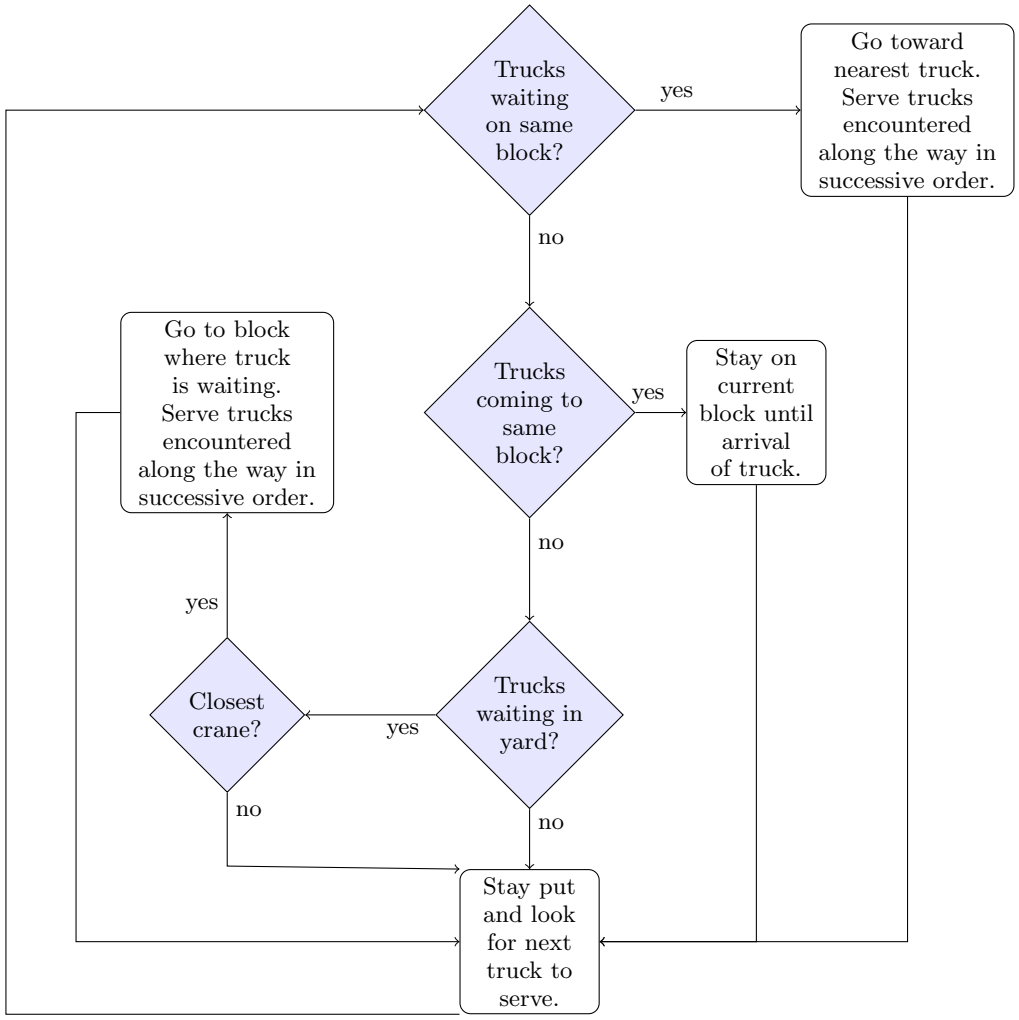

Figure 4: Service strategy for yard cranes at the port of Houston.

Figure 3: Yard crane deposits container on a truck.

The yard cranes are operated by operators who are given the freedom to make judgment calls on how to go about the yard to serve drayage trucks. At the Port of Charleston, the operators generally aim to minimize the trucks' wait time. However, they are given the flexibility to pick the next truck that makes the best sense based upon all the information they may know. At the Port of Houston, crane operators are also given the flexibility to use their judgment. In a series of interviews with different operators there, they appear to follow a strategy that is more distance-oriented (Figure 4).

To analyze the effectiveness of different yard crane service strategies, our initial models focus on stacked terminals equipped with RTGs and on the import drayage process. Also, we focus entirely on the container yard. We do not model the operations at the gate and berth (see Figure 5). We assume standard 40 foot long containers and yard blocks composed of 40 40-foot bays.

4. MODEL DESCRIPTION

We model the yard as a 2-dimensional grid where each cell fits exactly one container, which would make them about 40 feet wide by 10 feet tall. The yard contains 4 blocks each one

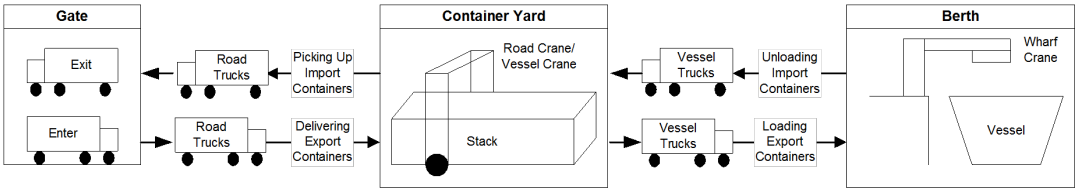

Figure 5: Typical operations and processes at seaport container terminals.

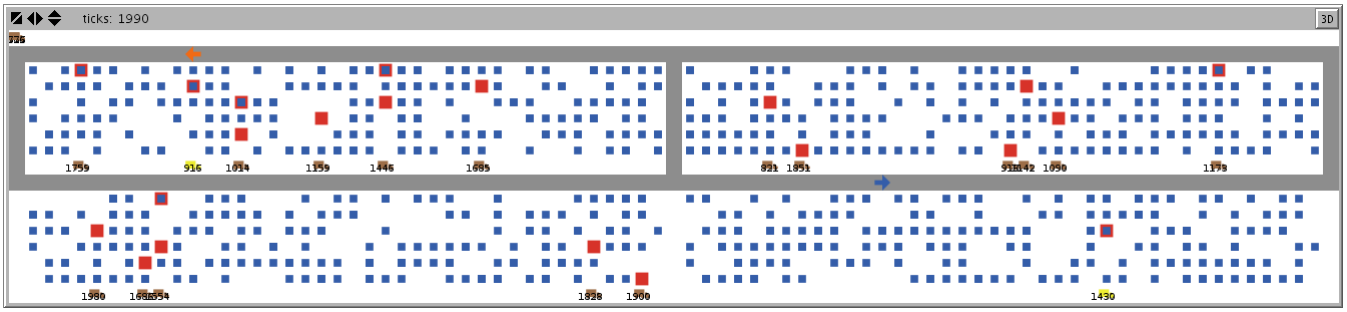

Figure 6: Screenshot of our agent-based model of yard cranes at a seaport container terminal.

with 40 bays, each bay has 6 rows of containers which can be stacked up to 4 high. The cranes move only on a specified track. The trucks appear below the bay where their desired container is located and are assumed to not interfere with the movement of the cranes. Figure 6 shows a screenshot of our implemented agent-based model. Notice that the cranes are represented by arrows which represent the direction in which the crane is moving. The track on which the cranes can move is represented by the gray lanes. Notice that the crane path allows the cranes to service all 4 blocks. Although not shown in Figure 6, our model includes several plots and other GUI elements which allow us to better understand what is happening in the program and to control its behavior.

Initially, the field is populated with randomly-placed containers. At run time the trucks arrive with a probability given by a Poisson distribution and each is assigned to a randomly chosen container. If the bay for that container is free then the truck is placed on that bay, otherwise the truck is placed on a waiting area. When the bay is freed, because the truck on the bay has been given the container it wanted, then the waiting truck is immediately moved to the bay.

We model the crane operators as utility-maximizing agents that can constantly re-evaluate their utility function. More formally, we say that there is a set C of cranes so that each crane $c \in C$ has a utility function $u_c(t)$ over all trucks $t \in T$ in the yard. We consider several different utility functions, all of which weigh different aspects of the world. One of those factors is the shortest path between the crane and the truck. As can be seen in Figure 6, there are a number of different paths that a crane can take to get to a container, some much longer than others. Thus, we let $\text{PATH}(c, t)$ be the shortest path between crane c and truck t. Similarly, we define $\text{DISTANCE}(p)$ to be the distance of a path p, $\text{HAS-TURN?}(p)$ to be a Boolean function that returns 1 if path p requires the crane to turn, that is, move from one of the top two blocks to one of the bottom two blocks or vice-versa, and $\text{OTHER-CRANE?}(p, c)$ to be a Boolean function that returns 1 if p passes over the current location of some crane other than c.

In our model, each crane c has a current goal g_c which can be either empty ($\emptyset$) or contain a truck t, which means that the crane's current goal is to go to the location of truck t, or it can have the value *deliver-container*, which means the crane is currently trying to move a container from the stack onto the truck. The cranes are opportunistic but we also implement a *decommitment-penalty* which can be set to 0 for completely opportunistic behavior or to larger numbers to make the cranes more committed to their current goal. Specifically, if $g_c = \emptyset$ or $g_c = t$ from some t, then the crane updates its goal at every time by first determining the optimal truck to service (t^*) and then switching to that truck only if its utility beats that of the current goal by more than the *decommitment-penalty*, as such:

$$t^* \leftarrow \arg_{t \in T} \max u_c(t) \tag{1}$$

$$g_c \leftarrow \begin{cases} t^* & \text{if } u_c(t^*) > u_c(g_c) + \textit{decommitment-penalty} \\ g_c & \text{otherwise,} \end{cases} \tag{2}$$

where we let $u_c(\emptyset) = 0$.

We consider three specific utility functions. The first one is a **distance-based** utility function which tries to capture the effective distance between the crane and a truck, giving higher priority to trucks that are closer to the crane. This distance is mostly just the path length between the crane and the truck, but also includes elements that consider the need for making a turn (as these take a longer time), the fact that there is another crane in the path (since then the path is blocked), whether or not the crane needs to change direction, and whether this crane is indeed the closest one to the truck. That last term provides the cranes with a slight implicit form of coordination. We believe that this utility function roughly captures what an operator means when he says he intends to always serve the closest truck. More formally, we define this utility as

$$
\begin{aligned}
u_c^{\text{distance}}(t) = & -\text{DISTANCE}(\text{PATH}(c,t)) \\
& - p_1 \cdot \text{OTHER-CRANE?}(\text{PATH}(c,t)) \\
& - p_2 \cdot \text{HAS-TURN?}(\text{PATH}(c,t)) \\
& - p_3 \cdot \text{CHANGE-HEADING?}(\text{PATH}(c,t)) \\
& - p_4 \cdot \text{NOT-CLOSEST?}(c,t),
\end{aligned} \tag{3}
$$

where the $p_1 \cdots p_4$ are fixed penalty constants. Their values are set to be high enough such that a crane will never choose a truck for which any one of the terms are true (that is, there is another crane on the way, or the crane must take a turn, or it must change its heading, or there is another crane closest) if there is another truck somewhere in the yard for which all terms are false. Note that the distance has a negative sign because the more a crane has to travel the less utility it receives from that delivery. Also, we let CHANGE-HEADING?(p) be a Boolean function which returns 1 if the crane needs to change its current heading in order to follow path p, and we let NOT-CLOSEST?(c,t) return 1 if crane c is not the one currently closest to t, or 0 otherwise.

Similarly, we define a **time-based** utility function that gives higher priority to the trucks that have been waiting the longest, but also taking into account the other terms. Formally, the time-based utility is given by

$$
\begin{aligned}
u_c^{\text{time}}(t) = & \text{WAIT-TIME}(t)) \\
& - p_1 \cdot \text{OTHER-CRANE?}(\text{PATH}(c,t)) \\
& - p_2 \cdot \text{HAS-TURN?}(\text{PATH}(c,t)) \\
& - p_3 \cdot \text{CHANGE-HEADING?}(\text{PATH}(c,t)) \\
& - p_4 \cdot \text{NOT-CLOSEST?}(c,t),
\end{aligned} \tag{4}
$$

where WAIT-TIME(t) is the time that truck t has been waiting.

Finally, we define a **time-and-distance** utility function which merges these two into one, as such:

$$
u_c^{\text{time-distance}}(t) = -\text{DISTANCE}(\text{PATH}(c,t)) + u_c^{\text{time}}(t) \tag{5}
$$

In modeling the yard crane gantry speed and handling times, actual or empirical data are used. A typical yard crane can gantry (i.e. traverse along the yard block) at a speed of 135 meters per minute [13]. Thus, it takes a crane about 6 seconds to gantry from one 40-foot bay to the next. As mentioned previously, a truck's wait time is a combination of the time it takes a crane to arrive at the bay where the truck is parked and the time it takes the crane to perform both rehandling and delivery moves. The steps involved in performing a rehandle are as follows. These steps are repeated for every container that is sitting on top of the target container.

1. Position spreader bar on top of container to be rehandled
2. Lower the spreader bar
3. Lock the spreader bar to the container
4. Hoist the container
5. Trolley to the desired stack
6. Lower the container
7. Unlock the twist lock
8. Bring the spreader bar back to its normal position

The steps involved in performing a delivery move are similar to a rehandle move. The key difference is in step 5 where instead of setting a container onto a stack, the crane operator sets the container onto the truck, which could take much longer time if the truck is not properly positioned. If the target container is at the bottom of a stack that is four high, then a crane will need to perform three rehandling moves and one delivery move. Data gathered previously by the authors show that the average rehandling time to be about 40 seconds and the delivery time to be about 87 seconds.

4.1 Solution Concepts

Given the problem definition, there are several solution concepts we might consider. The most obvious one is maximizing the throughput of the port. That is, servicing the most trucks possible in a given fixed amount of time. By definition, this measure is the same as minimizing the total wait time of the trucks in that same fixed amount of time. However, it is possible that a solution that minimizes the total wait time does so at the expense of one, or a few, unlucky trucks who must spend a very long time in the truck. Thus, another solution concept tries to minimize the maximum wait time of a truck. This solution is more egalitarian and might thus be preferred by the truck drivers. A third possibility, one which we have not explored yet, would to try to even out the amount of work each crane performs so that they all do about the same amount of work.

5. MODEL IMPLEMENTATION

Our model is implemented in NetLogo [18], an agent-based simulation platform and programming language. We modeled four yard blocks, each one with 40 bays of 40-foot containers, and each stack has six rows of containers that can be stacked up to four high. The cranes can move around these four blocks and can position themselves at any bay. The model is implemented to work for any number of cranes. The containers are distributed randomly across the four blocks and are never stacked more than four high in any one row. No new containers are added during a run since we are only concerned with evaluating the cranes' strategies. We also implemented trucks, each of which is assigned a randomly chosen container. If there is another truck already waiting at the bay where the container resides then the truck is made to wait in a holding area until the other truck is serviced and departed, thus clearing the spot for the waiting truck. The waiting truck then takes its position on the next tick.

Our model implements a discrete simulation where every tick corresponds to one second of real-world time. At every tick, the model creates and positions any new trucks that might have arrived during that tick, asks the cranes to perform their chosen action for that tick, and updates the graphs and plots. Since the cranes' actions take more than one second to execute, our implementation incorporates wait times for each action. For example, it takes six seconds for the crane to move from one stack to the next one. Instead of having the crane move one sixth of the distance each time, our implementation makes it wait for the first five seconds and then perform the move on the sixth second. This delay technique is used for all other actions: moving a container

```
main()
1  while user has not stopped program
2        do generate truck arrivals
3           for c ∈ C
4                do g_c ← ∅
5                   c.GO()
6           tick ← tick +1

go()
 1  if g_c ∈ T or g_c = ∅
 2     then t* ← arg_{t∈T} max u_c(t)
 3          if u_c(t*) > u_c(g_c) + decommitment-penalty
 4             then g_c ← t*
 5                  g_c^t ← ticks-to-move
 6  if g_c^t ≠ 0
 7     then g_c^t ← g_c^t − 1
 8          return
 9  if g_c ∈ T
10     then move to the first in PATH(c, t)
11          if we are at g_c
12             then g_c ← deliver-container
13                  g_c^t ← ticks-to-deliver
14             else
15                  g_c^t ← ticks-to-move
16  elseif g_c = deliver-container
17     then take step in delivery
18          if container delivered
19             then g_c ← ∅
```

Figure 7: Implementation methods. MAIN is the main loop and GO is a method implemented by every crane c. Note that $u_x(\emptyset)$ is assumed to be 0.

from one row to another (40 seconds) and moving a container from a row to the truck (87 seconds). By using this wait technique, it is easy to change the times each action takes to suit the real-world data. It also lets the simulation display an accurate representation of what is happening in the model.

The implementation algorithm is shown in Figure 7. At every tick we first create any new trucks that might have arrived and assign them to their appropriate spots. The cranes are then asked to GO(). First, crane uses its function to determine which is the best truck for it to service. If the crane has a current goal of serving a truck or no goal and there is a truck with a utility greater that *decommitment-penalty* then the crane switches to that one, thus implementing (2), as shown in lines (1) – (5). Lines (6) – (8) implement the time delay (skipped ticks) required for some of the longer actions. In lines (9) – (15) the crane moves one step in the path towards its chosen goal and then either re-sets its goal or changes its goal to one of delivering the container to the truck. Finally, in lines (16) – 19 the crane has the goal of delivering a container and takes a step in delivering it. This step might require the crane to move other containers around in its current bay if the desired container lies under other containers. In these cases the crane will require more ticks to deliver the container.

Our current implementation is fast enough for our needs, but it could be improved. Our analysis shows that line (2) of the GO() procedure is where the simulation spends most of its time as it has to check every single truck in the yard. Of course, we do not need to check every truck as there are some fairly simple heuristics that could be used to eliminate some trucks from consideration. Future version of our implementation will include such heuristics, thus enabling us to test much larger yards in the same amount of time.

Table 1: Simulation results for 2-crane scenario.

Distance-based (3)

De-commitment Penalty	Average Wait Time (minutes)	Min of Max wait time (minutes)
0	14.37	41.30
100	15.42	37.93
10,000	15.04	45.65

Time-based (4)

De-commitment Penalty	Average Wait Time (minutes)	Min of Max wait time (minutes)
0	68.97	68.95
100	65.49	72.58
10,000	53.84	56.18

Time-and-distance-based (5)

De-commitment Penalty	Average Wait Time (minutes)	Min of Max wait time (minutes)
0	68.04	86.38
100	65.42	67.97
10,000	52.24	56.77

6. SIMULATION RESULTS

Our first tests simply made sure that the program will be fast enough to be usable. Our current implementation running on a standard PC is able to simulate a whole day (8 hours) in just a few seconds, as long as the graphics display is turned off. Showing the animation of the cranes moving around the field significantly slows down the simulation, but this is not a problem as the visualization is only used for debugging when we do need it to go slow enough so we can see what the cranes are doing.

We mentioned several solution concepts for this problem in Section 4.1. For our initial tests we decided to focus on the average wait time of the trucks and the wait time of the truck that waits the most. The lower the average wait time the more trucks are being served in a day, as we keep the arrival rates constant. The maximum wait time tells us how the least quickly served truck had to wait. We varied the *decommitment-penalty* from 0, which implies a completely opportunistic crane, to 100, to 10,000 which implies a crane that once committed to a truck will never drop it for another one. Tables 1 and 2 show the average results from 100 runs for the various utility function and de-commitment penalties with trucks arriving following a Poisson distribution of with a mean of 40 trucks/hour. The tables show the average truck waiting time and the minimum of the maximum waiting time in each one of the runs, that is, the minimum of the list containing the maximum wait time for each run.

Table 2: Simulation results for 3-crane scenario.

Distance-based (3)

De-commitment Penalty	Average Wait Time (minutes)	Min of Max wait time (minutes)
0	6.53	19.05
100	6.82	20.78
10,000	6.77	19.90

Time-based (4)

De-commitment Penalty	Average Wait Time (minutes)	Min of Max wait time (minutes)
0	8.75	21.95
100	8.51	24.47
10,000	7.85	21.27

Time-and-distance-based (5)

De-commitment Penalty	Average Wait Time (minutes)	Min of Max wait time (minutes)
0	7.85	21.07
100	7.88	22.37
10,000	8.11	19.58

In Table 1 we note the surprising result that the average wait time for the time-based and time-and-distance-based utility functions is nearly four times as large as that of the distance-based utility. The reason for this is evident when viewing the simulation. When crane operators worked to minimize trucks' waiting time, they ended up making long runs from one end of the yard to another while ignoring nearby trucks. The model indicates that, on average, the two cranes covered a total distance of 16.25 miles when following the distance-based utilities and 25.41 miles when following the time-based utilities. The resulting effect is that many more trucks end up waiting longer.

Another surprising discovery from this study is how effective the distance-based utility is in minimizing the maximum waiting time across all the runs. It was expected that the time-based utility with the de-commitment penalty set to 10,000 would yield the lowest min-max wait time because the cranes would effectively "chase" after these longer waiting trucks. As shown in the third column of Table 1, the min-max wait times of the time-based utilities are higher than that of distance-based utilities. As explained above, when the cranes "chase" after the longer waiting trucks, they are less efficient because they are spending more time traveling to their target trucks. It would have been more efficient if they use that time to serve nearby trucks.

Table 2 shows the wait time and min-max wait time results when there are three cranes available. Note the significant drop in the average wait time and min-max wait time across all three utility types. It is also interesting to note that with three cranes, the performance of the time-based utilities is very close to that of the distance-based utilities. This is because cranes do not have to cover as much distance with three cranes. The model indicates that, on average, the three cranes covered a total distance of 13.65 miles when following the distance-based utilities, 15.47 miles when following

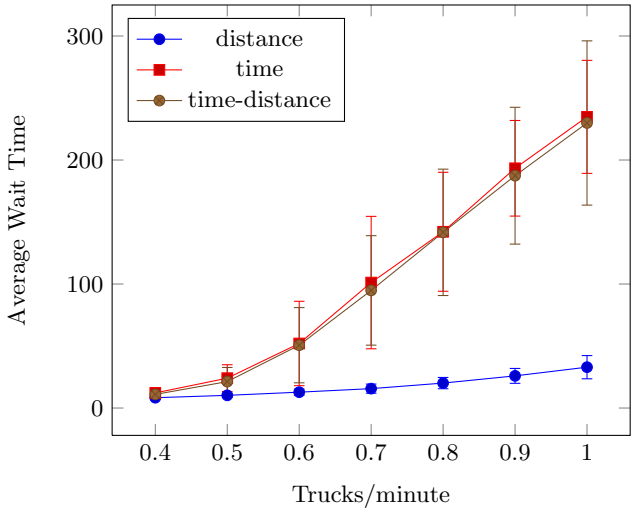

Figure 8: Average truck waiting time (over 100 runs) as a function of arrival rate for the three utility functions, with a *decommitment-penalty* of 0 and 2 cranes. The x-axis is the mean of a Poisson arrival process. The error bars represent the maximum and minimum wait time from all 100 runs.

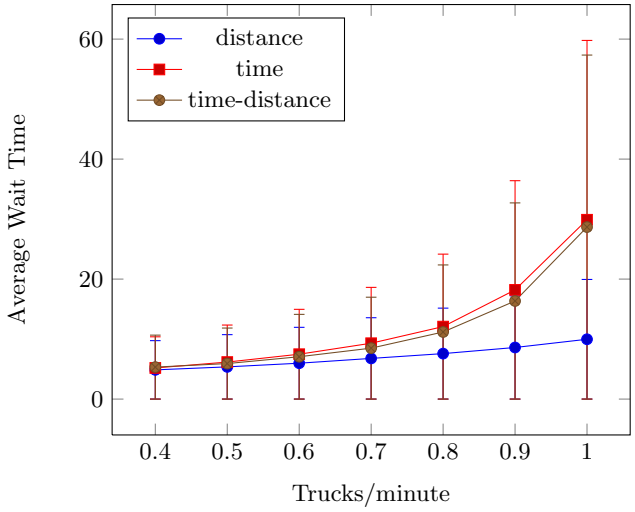

Figure 9: The same experiment as Figure 8 but with 3 cranes instead.

the time-based utilities, and 16.46 miles when following the time-and-distanced-based utilities.

After establishing that the distance-based utility performed the best overall, we were curious as to how it would fare as the truck arrival rate changed. Thus, we performed the same experiments as with 2 cranes but we varied the truck arrival rate from .4 trucks/minute to 1 truck/minute, where our previous results used a fixed rate of .667 trucks/minute (40 trucks/hour). The results of our tests are shown in Figure 8. As expected the distance-based utility performs best but it is noteworthy how flat its curve is while the other time-based utility functions explode as trucks arrive faster. On the other hand, as the arrival rate is made smaller, to .4, the difference between the various utility functions almost

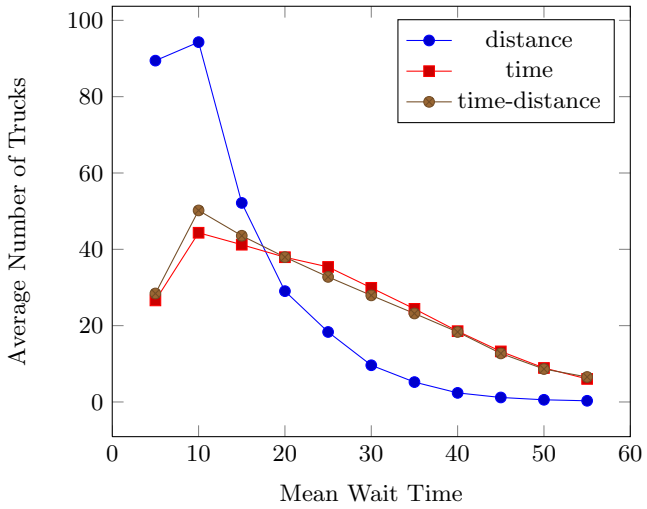

Figure 10: Distribution of average wait times with an mean truck arrival rate of .5.

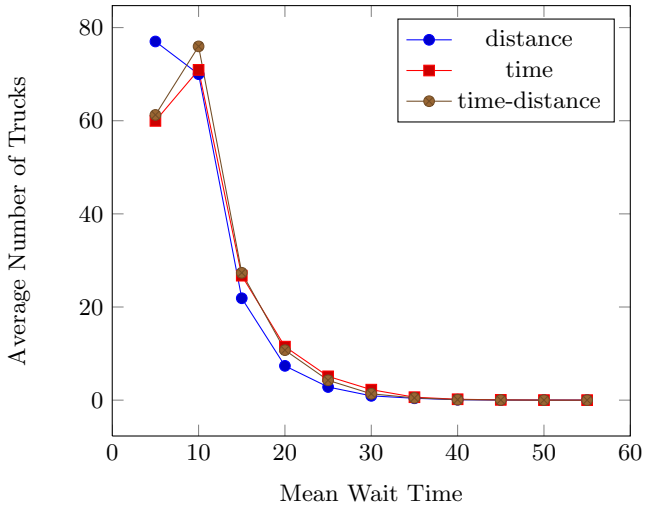

Figure 11: Distribution of average wait times with an mean truck arrival rate of .3.

disappears. The qualitative aspects of these results are not surprising. However, by using our agent-based model we can quantify exactly how much better one utility function is over the others given the specifics of the problem. Thus, terminal operators can use the data generated by our model to make business decision on how to direct their cranes.

Similarly, Figure 9 shows the results of the same experiment but with 3 cranes instead. As we might expect, three cranes are able to handle higher truck arrival rates thus there is a smaller, but still significant, difference between the distance-based utility function and the other ones.

Our final experiment shows the wait time distribution among the trucks. Figure 10 shows how many trucks, averaged over 100 runs, had to wait 0-5, 5-10, 10-15 minutes and so on till 55 minutes, the maximum wait time. We see that for the distance utility a great number of trucks does not have to wait more than 10 minutes, and only a small percentage has to wait more than 20 minutes. The time utilities, on the other hand, distribute wait times a bit more evenly. That is, a greater number of trucks have to wait longer. In Figure 11, we decreased the arrival rate to 0.3. As we saw in earlier results, for such low arrival rates we can expect the average wait times to be the same. The distributions largely confirm this; however, we do see that the distance utility yields about 20 more trucks with a wait time of less than 5 minutes. Another interpretation of this result is that when following the time utilities, in their effort to serve trucks in a first-come first-serve manner, the cranes miss the opportunities to serve nearby trucks.

7. CONCLUSION

This study introduced an agent-based utility maximization approach to modeling yard cranes at seaport container terminals to study how different service strategies affect truck turn time. The developed model provides a powerful tool terminal operators could use to assess the performance of various contemplated crane service strategies as well as the effect of having additional cranes or fewer cranes due to mechanical problems and/or scheduled maintenance. This study has identified a set of utility functions that properly captured the essential decision making criteria of crane operators in choosing the next truck to provide service to. Simulation results showed that if crane operators choose trucks that are closest to them without requiring the cranes to turn often (a time consuming process) and reverse heading, then the overall system performance in terms of average waiting time and the maximum waiting time of any truck will be better than if there were to choose trucks based on their waiting times.

Implementing the mentioned agent-based simulation model revealed some important lessons in modeling cranes as agents. Initially, we implemented the crane behaviors as procedures (e.g. choose nearest truck or choose longest waiting truck). While these procedures were easy to implement in NetLogo, as we incorporated additional complexities into the operators' decision making process, the procedures became unwieldy. The procedures ended up implementing ad-hoc rules which we could not fully explain or justify. For these reasons, we changed our approach to use utility functions and made the cranes utility-maximizing agents. By using utility functions we can clearly and explicitly capture how the cranes balance the various priorities: distance to truck, time truck spent waiting, etc. A caveat here is that the utility functions can make it harder to implement certain procedural knowledge, like "move to the closest truck and then keep going in that direction if there are more trucks waiting right behind that one." In this study, we have identified a set suitable utility functions and built a first model that implements these.

In future work, we plan to extend the model to handle larger yards and include explicit coordination with the incoming trucks, perhaps in the form of reservations or auctions. We hope to eventually build a detailed simulator of several yards as well as the trucks moving between them and their respective delivery sites. Such large-scale simulation will give us the ability to model truck traffic across a wide geographic span and see how it affects, or is affected by, seaport operations.

8. REFERENCES

[1] K. Dresner and P. Stone. A multiagent approach to autonomous intersection management. *Journal of Artificial Intelligence Research*, 31(1):591–656, 2008.

[2] L. M. Gambardella, A. E. Rizzoli, and M. Zaffalon. Simulation and planning of an intermodal container terminal. *Simulation*, 71:107–116, 1998.

[3] P.-L. Grégoire, C. Desjardins, J. Laumonier, and B. Chaib-draa. Urban traffic control based on learning agents. In *Proceedings of the 10th Internationnal IEEE Conference on Intelligent Transportation Systems*, 2007.

[4] L. Henesey, P. Davidsson, and J. A. Persson. Agent based simulation architecture for evaluating operational policies in transshipping containers. *Autonomous Agents and Multi-Agent Systems*, 18(2):220–238, 2009.

[5] L. Henesey, P. Davidsson, and J. A. Persson. Evaluation of automated guided vehicle systems for container terminals using multi agent based simulation. In *Multi-Agent-Based Simulation IX: International Workshop*, pages 85–96. Springer-Verlag, Berlin, Heidelberg, 2009.

[6] K. Kim. Sequencing delivery and receiving operations for yard cranes in port container terminals. *International Journal of Production Economics*, 2003.

[7] D. Lee, Z. Cao, and Q. Meng. Scheduling of two-transtainer systems for loading outbound containers in port container terminals with simulated annealing algorithm. *International Journal of Production Economics*, 107(1):115–124, 2007.

[8] K. Lewis. *Decision making in engineering design*. ASME Press, New York, 2006.

[9] C. Macharis. Opportunities for OR in intermodal freight transport research: A review. *European Journal of Operational Research*, 153(2):400–416, 2004.

[10] W. Ng. Crane scheduling in container yards with inter-crane interference. *European Journal of Operational Research*, 164(1):64–78, 2005.

[11] W. Ng and K. Mak. Yard crane scheduling in port container terminals. *Applied Mathematical Modelling*, 29(3):263–276, 2005.

[12] M. Rebollo, V. Julian, C. Carrascosa, and V. Botti. A MAS approach for port container terminal management. In *Proceedings of the Third Iberoamerican workshop on DAI-MAS*, 2001.

[13] D. Thurston. Utility function fundamental. In *Decision Making in Engineering Design* [8].

[14] T. Thurston and H. Hu. Distributed agent architecture for port automation. *Computer Software and Applications Conference, Annual International*, 0:81, 2002.

[15] K. Tumer and A. Agogino. Improving air traffic management with a learning multiagent system. *Intelligent Systems*, 24(1), Jan/Feb 2009.

[16] K. Tumer, A. K. Agogino, and Z. Welch. Traffic congestion management as a learning agent coordination problem. In A. Bazzan and F. Kluegl, editors, *Multiagent Architectures for Traffic and Transportation Engineering*, pages 261–279. Lecture notes in AI, Springer, 2009.

[17] M. Vasirani and S. Ossowski. A market-inspired approach to reservation-based urban road traffic management. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, pages 617–624, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems.

[18] U. Wilensky. NetLogo: Center for connected learning and computer-based modeling, Northwestern University. Evanston, IL, 1999. `http://ccl.northwestern.edu/netlogo/`.

[19] C. Zhang. Dynamic crane deployment in container storage yards. *Transportation Research Part B: Methodological*, 36(6):537–555, 2002.