Agent-Based Container Terminal Optimisation*

(Extended Abstract)

Stephen Cranefield
University of Otago
Dunedin, New Zealand
stephen.cranefield@otago.ac.nz

Roger Jarquin
Jade Software Corporation
Christchurch, New Zealand
rjarquin@jadeworld.com

Guannan Li
University of Otago
Dunedin, New Zealand
gli@infoscience.otago.ac.nz

Brent Martin
University of Canterbury
Christchurch, New Zealand
brent.martin@canterbury.ac.nz

Rainer Unland
Universität Duisburg-Essen
Essen, Germany
rainer.unland@icb.uni-due.de

Hanno-Felix Wagner
Universität Duisburg-Essen
Essen, Germany
hanno-felix.wagner@stud.uni-due.de

Michael Winikoff
University of Otago
Dunedin, New Zealand
michael.winikoff@otago.ac.nz

Thomas Young
University of Canterbury
Christchurch, New Zealand
thomas.young@pg.canterbury.ac.nz

ABSTRACT

Container terminals play a critical role in international shipping and are under pressure to cope with increasing container traffic. The problem of managing container terminals effectively has a number of characteristics that suggest the use of agent technology would be beneficial. This paper describes a joint industry-university project which has explored the applicability of agent technology to the domain of container terminal management.

Categories and Subject Descriptors

H.4.2 [**Information Systems Applications**]: Types of Systems—*logistics*; I.2.11 [**Artificial Intelligence**]: Distributed AI—*multi-agent systems*

General Terms

Algorithms

Keywords

Container Terminal Management, Container Terminal Optimisation, Logistics

1. INTRODUCTION

A container terminal[1] consists of a number of different areas. The *apron* is the (limited size) area directly beside the ship. The bulk of the container terminal is taken up with the *yard* where containers are stored. Quay Cranes (QCs) unload containers from the ship to the apron, while Straddle Carriers (SCs) clear the apron by moving containers to the yard and stacking them. Loading is the opposite (yard→apron→ship). Additionally, containers enter and leave the port on trucks and trains, and these need to be served by SCs. This process sounds simple, but is made complicated by a range of factors and constraints. For instance SCs need to be shared between the QCs, and also between QCs and trucks/trains. Additionally, some containers are refrigerated ("reefers"), and these cannot be without power for an extended period. Furthermore, the environment is dynamic: issues may arise during operations such as machines breaking down. Thus, container terminals' characteristics (distribution, cooperation, complexity, and dynamicity) make them a natural candidate for agent-based solutions[2].

The key metric for container terminal efficiency is ship turnaround time: any delays to a ship's schedule are bad (and may involve a financial penalty to the port). Some of the decisions that the terminal operators need to make as part of day-to-day operations are: Where should an incoming ship dock? How should QCs be allocated to a ship? How should SCs be allocated between QCs, yard rearrangement operations, and trucks and trains? Where should a given (incoming) container be placed in the yard?

This paper reports on a joint industry-university project that investigated the application of agents to container terminal optimisation. The industry partner was Jade Software Corporation, whose portfolio of products includes Jade Master Terminal (JMT), a comprehensive container terminal management solution. JMT is already used in some ports which gave us the opportunity to evaluate our system with real (but anonymised) data.

In our work we have focused on the last two questions listed above, and have explored them in the context of an agent-based container terminal emulation platform that we have developed.

2. AN AGENT-BASED SIMULATOR

The *ContMAS*[3] port emulation platform consists of several types of agents (Figure 1) and is designed to be highly configurable. It is structured into core agents, user interface agents, administrative

*Author order is alphabetical. An expanded version of this paper can be found at http://eprints.otago.ac.nz/1057/

[1]The details, especially the types of machines, vary between ports.

Cite as: Agent-Based Container Terminal Optimisation (Extended Abstract), S. Cranefield, R. Jarquin, G. Li, B. Martin, R. Unland, H. Wagner, M. Winikoff and T. Young, *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, Tumer, Yolum, Sonenberg and Stone (eds.), May, 2–6, 2011, Taipei, Taiwan, pp. 1113-1114.

[2]We are not the first to propose this, but space precludes a discussion of related work.

[3]Available at http://www-stud.uni-due.de/~sehawagn/contmas/page/index_en.html under an LGPL licence.

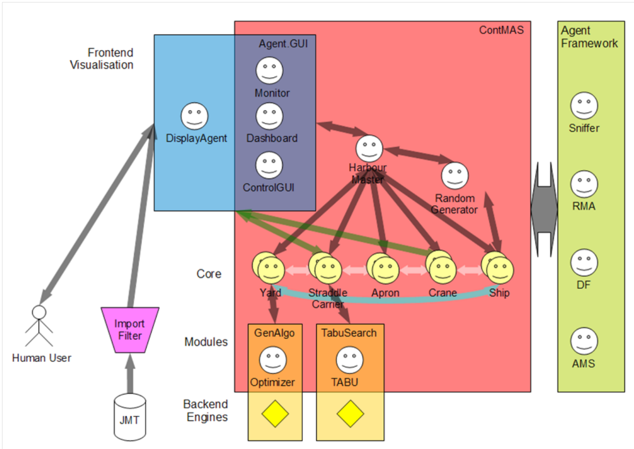

Figure 1: Emulation Architecture

agents and module agents. The core agents are called *ContainerHolderAgents*. Those are the agents which can pick up, transport ("hold"), store and put down containers, one for each individual device or other actor, such as cranes, ships, straddle carriers, yard areas or apron areas. There are several other agents in the model. The *HarbourMaster* controls the set-up and events such as creation of a new agent, e.g. for a newly arriving ship. The *ControlGUIAgent* provides the graphical interface for the human user. The *RandomGenerator* provides random numbers or events for simulations. Finally, *ContMAS* can be extended with *advisors* (e.g. *GenAlgo*, *TabuSearch*) which provide advice to specific agents. We have used advisors to integrate external (centralized) algorithms to improve the management of straddle carriers (Section 3) and yard allocation (Section 4). While agents can get advice, they remain autonomous, and may ignore the advice, thus our approach can combine the advantages of a centralized and a decentralized solution.

All negotiations between the agents are carried out by means of an extended contract net protocol: Any agent currently holding a container, e.g. a ship, initiates a call for proposals (CFP) to other suitable agents, e.g. cranes. They respond with a REFUSE or PROPOSE message, in the latter case containing the possible time of pick-up. The initiating agent then decides on one of the proposals and sends an ACCEPT message to that agent; all other agents get a REJECT message. Through this message exchange, the issuing agent and the determined contractor established a time and place to meet physically to hand over the container in question. Both agents move independently and can also negotiate with other agents about more containers in the meantime, thus building up a local plan. When the agreed upon time is reached, both agents should have moved to their negotiated position and the initiating agent issues a REQUEST to execute the appointment, i.e. to hand over the container, which the contractor will acknowledge with an INFORM message. At this point, the administration over the container changes from the initiating agent to the contractor, which can itself become an initiator and issue a CFP for the next step of transportation, e.g. from crane to apron.

3. STRADDLE CARRIER MANAGEMENT

One of the problems that we focus on is the management of Straddle Carriers. If Straddle Carriers are not managed well, then Quay Cranes can be idle, waiting for containers to be provided for loading, or for apron space to clear up so that they can unload containers from the ship.

We have developed a negotiation-based optimisation strategy[4] to allocate container moves to Straddle Carriers. The process for deriving a solution has two phases: initial allocation and optimisation. In the initial allocation phase each container in turn is put up for auction and is allocated to the machine with the cheapest bid, and inserted into its schedule (a list of container moves with associated source, destination, start and end times). In the optimisation phase, we try and improve the initial allocation by repeatedly modifying it (reallocating a container to a different position, or to a different machine), picking the best candidate modified solution.

This process is done before machines begin performing moves, and develops a complete scheduled plan for unloading a ship. A strength of the approach is that should something go wrong, the schedule can be updated to reflect necessary changes, and the allocation process re-run. For example, should a Straddle Carrier break down, the solution is updated by removing the Straddle Carrier in question, putting its allocated container moves back into the list of moves to be allocated, and then re-running the allocation process to allocate these container moves to other Straddle Carriers.

We have implemented our approach for container management using a Tabu Search framework (OpenTS[5]) and have evaluated it using real real (anonymised) data from the local port, showing that our approach is able to find solutions, and that the optimisation phase does improve the solution.

4. YARD MANAGEMENT

Deciding where to place a container in the yard is important and difficult. The decision can significantly affect efficiency, e.g. extra time will be needed if a container needs to be extracted from beneath another container ("overstow"). It is complex because the environment is dynamic and unpredictable (e.g. containers arrive at unpredictable times, or a ship may not arrive at all).

Given a sequence of expected container moves and a representation of the current yard state, we create a population of yard allocations for incoming containers, and use an evolutionary algorithm to find a good allocation. A genome is a sequence of (container id, yard location) genes, where each gene represents a move of a particular container to a [lane,bay,tier] location within the yard, and order is significant. The fitness is calculated by simulating the moves encoded in the genome, using a 'Manhattan' distance cost. We use a mutation operator that sets the location of a random gene to a random location in the yard, and a crossover operator that identifies locations unique to the second parent, and then switches those for the locations of a random proportion of genes in the first parent, leaving the order of moves untouched[6]. This approach has been implemented and integrated with *ContMAS*.

5. CONCLUSION

Overall, our conclusion is that taking an agent-based approach has proven to be a natural choice, and we have found that the agent paradigm supports the natural modeling of such an environment with a high level of detail and flexibility. Initial evaluation is promising, but more extensive evaluation is still to be done.

[4]The description here is necessarily brief, and omits discussion of how we deal with the various constraints that apply.

[5]http://www.coin-or.org/Ots

[6]This can result in invalid genomes, e.g. where the crossover results in a container to be in mid-air, which are repaired by dropping mid-air containers down the stack to a supported position.