SIMULATION AND OPTIMISATION FOR MANAGEMENT OF INTERMODAL TERMINALS

Gianluca Bontempi, Luca Maria Gambardella, Andrea Emilio Rizzoli
IDSIA,
Istituto dalle Molle di Studi sull'Intelligenza Artificiale
Lugano,
Switzerland
E-mail: (gianluca, luca, andrea)@idsia.ch

KEYWORDS

discrete simulation, optimisation, decision support systems, scheduling, transportation.

ABSTRACT

This paper presents a decision support system for improving the management of intermodal container terminals. It is implemented as a modular architecture which integrates a forecasting model, a planner and a simulation module. While the forecasting module estimates container traffic, the planning module uses this information to generate efficient policies for storage, resource allocation and scheduling. The performance of management policies is assessed via computer simulation. Genetic algorithms, taboo search, and dynamic programming techniques are used to implement management policies. Some experimental results are presented.

INTRODUCTION

This paper presents a decision support system based on an integrated architecture to aid the management staff of a modern intermodal container terminal in short and long-term decision making.

An intermodal container terminal is a terminal where containers enter and leave by multiple means of transport, as trucks, trains, air cargoes and vessels (I/O transport means). This research is applied to the case study of La Spezia container terminal, located on the shores of Tyrrhenian sea in Italy.

Containers arrive at La Spezia terminal by train, vessel or truck and are stored in the terminal yard. Containers then leave the terminal by the same means to reach their final destinations. The flow of containers is composed of an *import flow*, i.e. containers arriving at the terminal from the sea, to be directed to the final destinations by trucks and trains, and an *export flow*, i.e. containers leaving the port on vessels.

Containers are stacked up to the fifth level on the yard by rail-mounted cranes (*yard cranes*) which unload trucks and trains. *Quay cranes* unload vessels and place containers on *shuttle trucks* which move them to storage locations in the yard. Loading a vessel is a similar process, where the shuttle receives the container from the yard cranes and moves it to the proper quay.

The amount of work processed by a container terminal depends on the quantity of containers in transit, import and export.

Storing containers on the yard, *allocating resources* in the terminal, and *scheduling vessel loading and unloading operations* (L/U operations, for brevity) are major problems in an intermodal container terminal. To solve these problems we define an architecture composed of three different but strictly connected modules (see Figure 1):

– a simulation model of the terminal, described in terms of entities (work force, transport means, storage areas, etc.) and

processes (vessel load/unload, shuttle truck movements, crane operations, etc.);
- a set of forecasting models to analyse historical data and to predict future events (Box *et al.*, 1994; Vemuri and Rogers, 1993), thus providing estimates of the expected import and export flows;.
- a planning system to optimise L/U operations, resource allocation, and container locations on the yard.

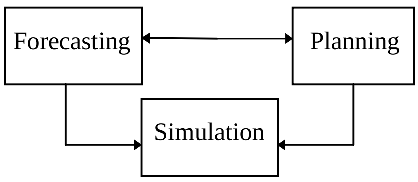

Figure 1. The modular system architecture.

This architecture supports the management staff in the evaluation of:
- alternative vessels loading and unloading sequences in terms of time and costs;
- alternative resource allocations procedures;
- different policies for container storage both in terms of space and cost of operations.

This allows terminal operators to assess "what-if" scenarios; for instance, what happens if the terminal undergoes an increased input/output throughput, or even if structural changes are made (e.g.: new berths are built), etc.
As the forecasting module is described in our previous paper (Gambardella *et al.*, 1996), in the following sections we introduce the other two modules of our architecture: the planner and the terminal simulator. For each topic we present the major problems, the resolution methodologies and the experimental results obtained at the current state of the project.

PLANNING OF CONTAINER OPERATIONS

The management of container terminals encounters three main problems: efficient storage of the containers on the yard, allocation of resources for vessel L/U operations, scheduling of L/U operations in order to improve resource usage and reduce the costs. All these problems are strictly interconnected and hard to be solved as a whole. To overcome this, we assign a different time horizon to each problem and use the solution of the long term problem (container storage policy) as input to the mid term problem (resource allocation problem), the solution of the mid term problem (resource allocation policy) as input to the short term one (L/U scheduling problem, producing an L/U scheduling policy).

Long term optimisation: containers storage
The objective of long term optimisation is to find one or more efficient *container storage policy*. Our approach to find such a policy consists of two steps: first, we solve a job-shop scheduling problem to determine the yard configuration (arrangements of containers on the yard) that optimises L/U operations; then, the policy consists in storing containers on the yard minimising the deviation from that configuration.
Different yard configurations are ranked according to a quantitative performance index. This index measures the performance (e.g. the maximum delay or makespan) of a L/U scheduling policy with respect to the given yard configuration used as initial condition. We assume that the efficiency of a generic L/U scheduling policy depends on the yard configuration.
Let us consider the following example. We have three vessels moored at the quay-side which are to be loaded within the next 3 hours. Their containers (100 per vessels) are stored on the same yard area, where three rail-mounted cranes operate. We want to estimate how three alternative yard configurations affect the performance of loading operations. We measure the performance by employing the *makespan* index, that is the completion time of the whole loading operation. The three configurations to be compared are represented in Figure 2.

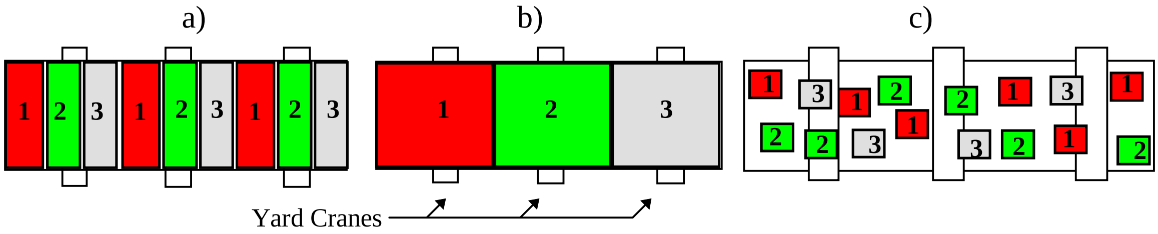

Figure 2. Three alternative storage policies. Areas labelled with "1" store containers to be loaded on vessel "1", and so on.

In configuration a) containers have been grouped in 9 distinct clusters (3 per vessel). In configuration b) only 3 large cluster (one per vessel) have been created. Configuration c) is a random placement configuration.

The L/U scheduling policy is obtained running a job shop scheduling algorithm (presented in section 4) and the resulting makespan is used to compare the three configurations. For comparison purposes, we set the time required for the completion of each loading operation equal to one. Different runs of the algorithm returned a statistical distribution of the makespan for the three different initial configurations.

We see how the second configuration has better performance (lower makespan) than the other two configurations (see Figure 3). The result is trivial but the method can provide an experimental setting to evaluate and compare different approaches to the problem.

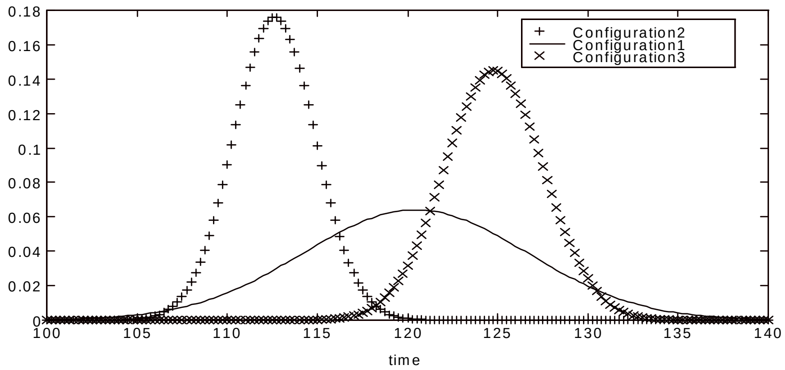

Figure 3. Makespan statistical distribution for different terminal configurations

Mid-term optimisation: resource allocation

The role of mid-term optimisation is to solve the problem of resource allocation for vessel loading and unloading operations. Resource allocation spans a time horizon limited to a few shifts of the working force.

The problem can be formulated as a complex mathematical programming problem, with the goal of maximising the profits over a limited time horizon. The objective function depends on the costs for resource usage, the lateness in vessel loading/unloading and the income of the terminal for each operation.

At this detail level, we model the terminal with a discrete-time dynamic system and a functional that is additive over time.

The dynamic system is of the form:

$$\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k),\mathbf{u}(k),k)$$
$$k = 0,1,\ldots,N-1 \qquad (1)$$

where

– k indexes discrete time. The length of each time unit is equal to a shift of the work force.
– $\mathbf{x}(k)$ is the state of the terminal and summarises the information relevant for resource allocation. In this case $\mathbf{x}(k)$ is a vector $[y_1(k) \ldots y_n(k)$, $z_{1,1}(k) \ldots z_{1,r}(k)$,

… , $z_{i,1}(k)$ … $z_{i,r}(k)$, … , $z_{n,1}(k)$ … $z_{n,r}(k)$] where $y_i(k)$ represents the number of container moved up to time k for the *i*-th vessel ($i = 1,..,n$; where n is the number of moored vessels) and $z_{i,j}(k)$ $(0 \le z_{i,j}(k) \le 1 \;\; , \sum_{i=1}^{n} z_{i,j}(k) = 1 \quad \forall k, j \;)$ represents how containers for the *i*-th vessel are distributed in the *j*-th region of the yard ($j = 1,..,r$, where *r* is the number of yard regions).

– ***u****(k)* is the control action selected at time *k* with knowledge of the state ***x****(k)*. In our problem the set of actions U is the set of possible combinations of resources allocated to vessels (e.g. quay cranes) and yard areas (e.g. yard cranes, shuttles). An example of action is the following: *for the next time interval let us allocate 2 cranes for vessel 1, 1 crane for vessel 2, 2 cranes for yard region A and 4 shuttles***.**

– *N* is the number of working shifts taken in considerations by our model.

The functional *J* to be maximised is additive in the sense that at each time step it increases according to the difference between terminal incomes and costs.

This problem is made difficult by the large number of state variables and possible actions. We have envisioned Dynamic Programming and Genetic Algorithms among the techniques explored to solve the problem. Dynamic Programming is a well assessed technique to solve optimal control problems. It provides an optimal solution to the problem (1) by returning an admissible sequence of actions which maximises profits for the next *N* steps. However, in practice, it suffers of a complexity combinatorial explosion which is not manageable in our case due to the high number of variables. Thus, in order to apply this technique we limited our state variables to the subvector ***y***. This implies that we make the assumption that the distribution of containers in the yard does not change during the operations.

GA are a powerful tool in the field of global optimisation (Goldberg, 1989). They have been applied to a wide variety of real-world problems (Bruzzone and Signorile, 1996) and exhibited in many cases better efficiency than traditional methods. In our problem they provide the important property of being not sensitive to the state dimension of the problem. In a GA setting the resource allocation problem is seen as a common optimisation problem where the search space is the set of possible action sequences and the fitness function is the desired profit. The resolution of the problem is not affected by the complexity of the dynamics of the underlying system. Henceforth, we are able to model the terminal in a more complex fashion by considering the whole state vector ***x***. We employ the software tool Genesis (Grefenstette, 1987) to perform the GA optimisation.

Preliminary results obtained with the GA approach are encouraging: the algorithm returns good solutions in limited amount of computation time and for complex problem configurations.

Short term optimisation: L/U scheduling

The *short-term* optimisation is aimed at solving the scheduling of daily L/U operations.

We decomposed this problem in two steps:

1. *Off-line scheduling*, based on the assumption of a complete a priori knowledge of the terminal state. This implies to know at each instant, with certainty, the availability of cranes, the level of occupancy of the yard, the date of arrivals of the vessels. Unfortunately, there is no guarantee that this information is correct and there is the risk that an off-line scheduling is not robust enough to be effective in reality. Then a reactive adaptation of the scheduling is necessary.
2. *Reactive scheduling* (Kerr and Szelke, 1995), whose task is real time supervision of the L/U procedure. Its initial condition is provided by the previous level optimisation (off-line scheduling) and then it is adapted to unexpected events.

We represent the off-line scheduling as a NP-Hard job-shop scheduling (Morton and Pentico, 1993). A job-shop problem consists of a set of machines that perform operation on jobs. A job is composed of an ordered list of operations each one identified by the required machine and the processing time. In our representation each I/O transport mean is a job, whose set of operations is the list of containers to be processed (loaded and unloaded) by the intermodal terminal. The machines are the set of resources (shuttles, cranes) available into the terminal to execute this job. Many techniques have been used in the literature to solve a job-shop problem. We focused on Taboo-Search (Taillard, 1994) as an effective way to implement a local search method with strong computational properties. Given a set of vessels and the relative list of containers together with their placement in the terminal, the algorithm returns the optimal sequence of operations to be performed by terminal resources in order to minimise the maximum delay (makespan).

TERMINAL SIMULATION

Simulation plays a fundamental role in evaluating the performance of the management policies and it is the principal tool for "what-if" analysis (Hayut *et al.*, 1994). Using the simulation tool, terminal managers may assess the efficiency of alternative policies and to choose the most suitable one.

The simulation model is obtained by first performing an analysis of its requirements, then designing and implementing the software structures identified in the analysis stage. A calibration and validation stage is performed, and finally the model can be used to support the decision makers in managing the terminal.

The analysis of the problem, based on object-oriented analysis and design techniques (Booch, 1994), has identified two main hierarchies of *classes*: Terminal components and management policies. Terminal components are objects such as transport means, cranes, yard areas. Management policies (see section 0) are classified into resource allocation, container storage on the yard, and L/U scheduling.

In Figure 4 a part of the hierarchy of terminal components is reported. Trucks and trains are transport means, and the class `Truck` is then specialised into class `TranspTruck` (it is an I/O transport mean, since it moves containers in and out of the terminal) and class `Shuttle` (trucks servicing yard areas,

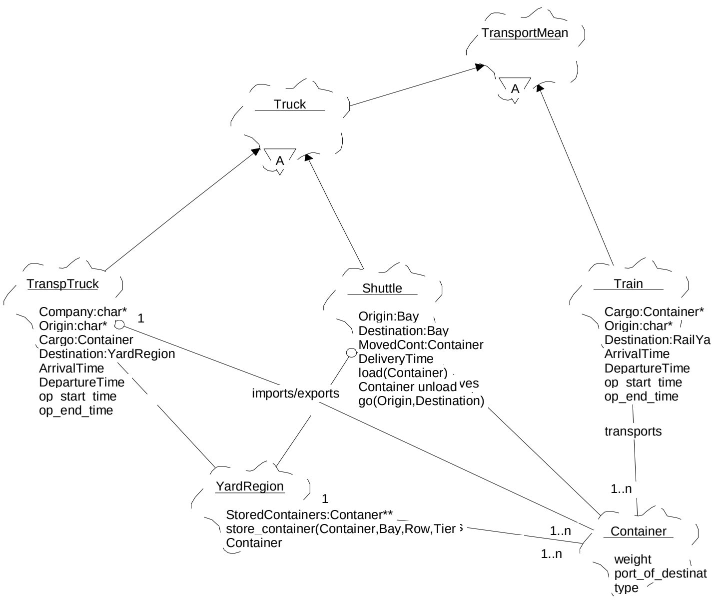

Figure 4. The class diagram for port component classes.

moving containers from point to point). Objects of class `Shuttle` and `TranspTruck` are in relation with `YardArea` objects: a truck arrives to a yard area to load/unload containers, and a shuttle moves to and from a yard area. Finally, shuttles, trucks and trains move containers and the corresponding objects are therefore related to class `Container`.

Each class has a set of *methods* which are used to perform the simulation. In Figure 4 we represent the sequence of method invocations describing the process of vessel unloading. Note that, while

the supervision and management of processes can be performed by either a human decision maker or by a set of management policies, in this case the `S1` object is an instance of the class `Scheduler` (supervisor of the unloading process) which uses a L/U scheduling policy previously generated (see section 0). The scheduler acts in co-ordination with other unloading schedulers, thus implementing an L/U procedure more effective than the one currently adopted in La Spezia terminal.

this policy is executed by means of a sequence of unloading orders sent to the quay crane. The unloaded containers are then moved by truck and stored in the yard. Since there may be more than one unloading vessel present on the quays, the yard cranes are accessed concurrently by many storage requests. The queue is handled according to the container storage policy.

The implementation of such a system is performed using the discrete event simulation language Modsim III (CACI, 1996) which supports the process-oriented simulation paradigm and the object-oriented programming paradigm. Monte Carlo techniques are applied in scenario evaluation, together with sensitivity analyses of model parameters.

CONCLUSIONS

An intermodal terminal is a complex dynamic system characterised by an high level of uncertainty and non-stationarity. Thus, a unique model representation is not able to properly describe the reality and does

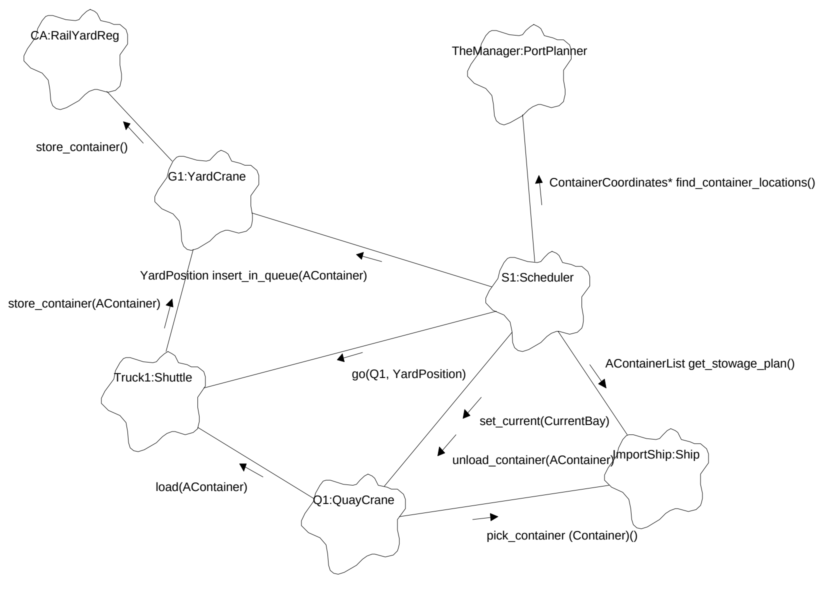

Figure 5. The object diagram for vessel unloading.

In Figure 5 it is shown a small part of the simulated system. Namely, we show how the scheduler retrieves the scheduling policy from the `PortPlanner` object and how

not adequately support the definition of efficient control policies. Our approach consists in a problem decomposition on different time scales which allows for an easier description and the adoption of different formalisms at the different problem levels. In particular, this paper focuses on the planning and the simulation modules of the architecture. We can manage terminal activities with success only if we split the whole problem on different time horizons and assume some *a priori* knowledge on the state of the terminal. Thus, simulation becomes an invaluable tool to test and validate the implemented policies in a realistic and non deterministic environment. We believe that only an integrated approach based on optimisation techniques and simulation methods may lead to a decision support system which is sufficiently robust to be effective in a non stationary environment.

ACKNOWLEDGEMENTS

The work presented in this paper is supported by the Swiss Commission for Technology and Innovation (CTI) under the project n. 3128.1 "A methodology for container flow forecasting and positioning in intermodal terminal".

REFERENCES

Booch, G., 1994. Object-oriented Analysis and Design: with Applications, 2nd Edition. Santa Clara (CA): Rational.

Box, G.E.P., Jenkins, G.M., Reinsel, G.C., 1994. *Time Series Analysis: Forecasting and Control*. London: Prentice Hall.

Bruzzone, A.G., Signorile, R., 1996. Genetic algorithms and simulation as support for planning a port terminal. In: A.G. Bruzzone, J.H. Kerckhoffs (eds.), *Simulation in Industry, 8th European Simulation Symposium*, 1996. Genoa: Society for Computer Simulation International.

Gambardella, L.M., Bontempi, G., Taillard, E., Romanengo, D., Raso, G., Piermari, P., 1996. Simulation and forecasting in intermodal container terminal. In: A.G. Bruzzone, JH. Kerckhoffs (eds.), *Simulation in Industry, 8th European Simulation Symposium*, 1996. Genoa: Society for Computer Simulation International.

Goldberg, D.E., 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MS: Addison Wesley.

Grefenstette, J.J., 1987. *A User's Guide to GENESIS*. Washington, D.C.: Navy Center for Applied Research in Artificial Intelligence.

Hayuth, Y., Pollatschek, M.A., Roll, Y., 1994. Building a port Simulator. *Simulation*, 63(**3**), 179-189.

Jacobs, R.A., 1995. Methods for combining experts' probability assessments. *Neural Computation*, 7, 867-888.

Kerr, R., Szelke, E. (eds.), 1995. *Artificial Intelligence in Reactive Scheduling*. London: Chapman & Hall.

Morton, T., Pentico, D., 1993. *Heuristic Scheduling Systems with Applications to Production Systems and Project Management*. New York: John Wiley & Sons.

Taillard, E.D., 1994. Parallel taboo search techniques for the job shop scheduling problem. *ORSA Journal on Computing*, 6(**2**), 108-117.

Vemuri, V.R., Rogers, R.D., 1993. *Artificial Neural Networks: Forecasting Time Series*. IEEE Computer Society Press.

BIOGRAPHY

Gianluca Bontempi had his *degree magna cum laude* from Politecnico di Milano, discussing a thesis on qualitative simulation with fuzzy differential equations. He is now research assistant at IDSIA-Lugano and Phd student at IRIDIA-ULB, Universitè Libre de Bruxelles with a thesis on multimodelling in system identification and control. His main research interests are on qualitative simulation, intelligent control for autonomous robotics, multimodel learning, optimisation and scheduling in real industrial problems.