Universität Hamburg
Fakultät für Wirtschafts- und Sozialwissenschaften
Fachbereich Betriebswirtschaftslehre
Institut für Operations Research (Unternehmensforschung)

RMGC Simulation Model

Documentation of a Simulation Model for automated
Rail-Mounted-Gantry-Crane Systems at Seaport Container Terminals

Nils Kemme

20.09.2011

Contents

List of Figures

List of Tables

List of Symbols and Abbreviations

$\overline{\delta}$	mean dwell time over all containers
$\overline{\delta}^{\text{ts}}$	mean dwell time over transshipment containers
$\overline{\delta}^{\text{ie}}$	mean dwell time over import-export containers
δ_c	dwell time of container c
ϵ	correction parameter for random generation of container deliveries by vessel v
π^{bf}	block-factor
π^{fill}	average filling rate of the yard block
π^{length}	length of the quay wall (m)
π_i^{qcm}	number of QC moves per operating hour at vessel of type i
π^{teu}	TEU-factor, average container length (TEU)
π^{through}	terminal throughput (containers)
π^{ts}	transshipment factor
ρ^{block}	fraction of total storage slots located in the modelled yard block
ρ_{ik}^{class}	fraction of class k for transport mode i
ρ_{ig}^{dest}	fraction of containers with destination g among all containers departing with mode i
$\rho_{it}^{\text{timeslot}}$	fraction of arrivals in time slot t of mode i
ρ_w^{weight}	fraction of containers in weight class w
$a_c^{\text{container}}$	arrival date of container c
a_v^{vessel}	arrival time of vessel v
b_v^{vessel}	berthing time of vessel v
c	container index
C_i	number of containers delivered by mode i

c_{ij}	fraction of containers delivered by mode i and picked up by mode j
$d_c^{\text{container}}$	departure time of container c
d_v^{vessel}	departure time of vessel v
g	destination index
$\underline{H}_{it}$	first hour of time slot t for arrivals of mode i
$\overline{H}_{it}$	last hour of time slot t for arrivals of mode i
i	transport mode index
j	transport mode index
k	transport mode class index
l	deep-sea vessel tour index
$\underline{L}_{ik}$	minimum length of a ship in class k of mode i
$\overline{L}_{ik}$	maximum length of a ship in class k of mode i
n_i^{class}	number of classes of transport mode i
n_i^{dest}	number of destination classes for containers of mode i
n_i^{timeslot}	number of daily time slots for arrivals of mode i
n^{vcp}	number of weekly repeated vessel tours
n^{weight}	number of weight classes
n^{x}	length of the yard block (bays of TEU)
n^{y}	width of the yard block (lanes of TEU)
n^{z}	height of the yard block (tiers of TEU)
p_c^{x}	bay (of TEU) container c is positioned in the yard block
p_c^{y}	row (of TEU) container c is positioned in the yard block
p_c^{z}	tier (of TEU) container c is positioned in the yard block
$\underline{S}_{ik}$	minimum number of moves per call in class k of mode i
$\overline{S}_{ik}$	maximum number of moves per call in class k of mode i
S_v	number of containers delivered by vessel v
S_v^{add}	random additive number of containers delivered by vessel v
S_v^{min}	minimum number of containers delivered by vessel v
t	time slot index

T	length of simulation period (weeks)
T^s	length of warm-up period (weeks)
t^{os}	period of time the stowage plan is created before starting the vessel loading
$t^{\text{wsprocess}}$	waterside processing time for container unloading/loading and horizontal transport by QCs and transport vehicles
v	vessel index
w	weight class index
CCFS	combined cost function stacking
DRMGC	double rail mounted gantry crane
EDT	empty driving time
ETA	estimated time of arrival
ETD	estimated time of departure
FCFS	first-come-first-served
GCR	gross crane rate
ISO	International Organization for Standardization
lshk	housekeeping job for a landside departing container
lsin	landside storage job
lsout	landside retrieval job
lsshu	shuffle job caused by landside departing container
PoD	port of destination
QC	quay crane
RaS	random stacking
RMGC	rail mounted gantry crane
SRMGC	single rail mounted gantry crane
TEU	twenty feet equivalent unit
TriRMGC	triple rail mounted gantry crane
TRMGC	twin rail mounted gantry crane
TTU	truck trailer unit
wshk	housekeeping job for a waterside departing container

wsin	waterside storage job
wsout	waterside retrieval job
wsshu	shuffle job caused by waterside departing container
XT	external truck

Chapter 1

Introduction

Within the framework of this work, a discrete-event-simulation model is described that is designed to reproduce the multi-objective, stochastic, real-time environment of automated RMGC (rail mounted gantry crane) systems at multiple-berth seaport container terminals. Basically, a single RMGC-operated yard block of the container storage yard is modelled in great detail along with the corresponding waterside and landside handover areas, while the up- and downstream processes are only considered in a very simplified way in order to keep the model at a manageable size in terms of parametrisation possibilities, simulation costs and interpretation effort.

The simulation model is implemented using the simulation model library Tecnomatix Plant Simulation 8.2. A detailed introduction to this model library is for instance given by Bangsow (2008). Here, neither the simulation software nor the source code of the created simulation model are presented in detail. Moreover, the functionalities and processes of the simulation model are explained in detail in order to allow for a profound evaluation of simulation results yielded with that model. With this in mind, some important processes are pictured in flowcharts in the following chapters in order to improve the understanding of the model's processes. A legend of the flow charts — that are created in the style of ISO 5807 (International Organization for Standardization) — is shown in figure 1.1. Process steps are represented as rectangles and decisions are pictured as parallelograms. All decisions are mapped as questions, with either yes or no as only possible answer. The horizontal field, where a process step is located in, indicates the area of accountability for the corresponding process step.

In the second chapter, the architecture of the simulation model is explained. Thereafter, the major modules of the simulation model — which are the scenario creation module, the drive control module, the administration module, the strategy module, the experimental control module and the statistics module — are successively introduced. This documentation is closed with a summarising overview on all inputs and outputs of the model, its limitations and features as well as on validation and verification issues.

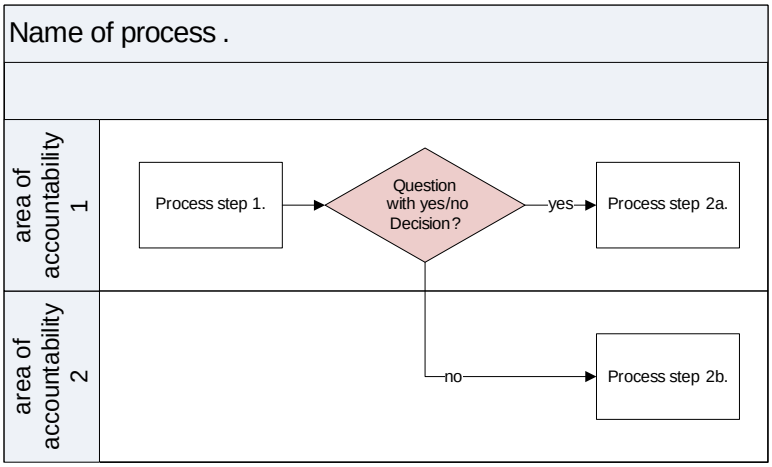

Figure 1.1: Legend of flowchart symbols.

Chapter 2

Architecture of Simulation Model

The simulation model comprises hundreds of variables, parameters, tables, methods, dialogues and other items — each of them connected with a certain module of the simulation model that is designed to fulfil a certain function within the entire model. The drive control module is for example responsible for the execution of all crane movements. In total, six modules and several submodules can be distinguished. An overview of the architecture of the implemented simulation model and the interactions between the modules is given in figure 2.1.

The main modules of the simulation model are depicted in the middle of figure 2.1. The scenario-creation module (see chapter 3) comprises all functions concerning the generation of reproducible container arrivals and departures for the modelled yard block. While individual containers as well as vessel and XT (external truck) arrivals are randomly generated by a parameter-based data generator, the time of container arrivals and departures at the modelled yard block are determined by a data preprocessing submodule which imitates the operations of QCs (quay crane) and waterside horizontal transport machines in a simplified way.

The administration module (see chapter 5) and the drive-control module (see chapter 4) are the core of the simulation model. While the latter one executes and controls all crane movements, the administration module has many functions. Firstly, the yard block capacities are managed. Secondly, the job-management submodule generates and manages transport jobs for the gantry cranes, initiates their scheduling and initiates the execution in the scheduled order. Thirdly, the handover-area management controls the occupancy of the handover-area capacities.

The strategy module (see chapter 6) is closely linked with the administration and drive-control modules as it contains exchangeable decision procedures for these modules. The stacking submodule decides on stacking locations for containers and returns its decision to the storage management submodule. The crane scheduling submodule schedules the crane assignment and sequencing of transport jobs and returns the next job for a calling crane to the job management submodule. The crane routing submodule decides on the routing of the RMGCs, with respect to granting the right of way and executing crane crossing manoeuvres and returns the decision to the collision- and deadlock-avoidance submodule.

The experimental control (see chapter 7) and the statistics modules (see chapter 8) have assisting cross-sectional functions for the main modules. By means of the experimental-control module different experiments can be automatically conducted by the simulation model. Statistical data that is needed to evaluate simulation experiments with different parameter settings is continuously gathered and processed by the statistics module.

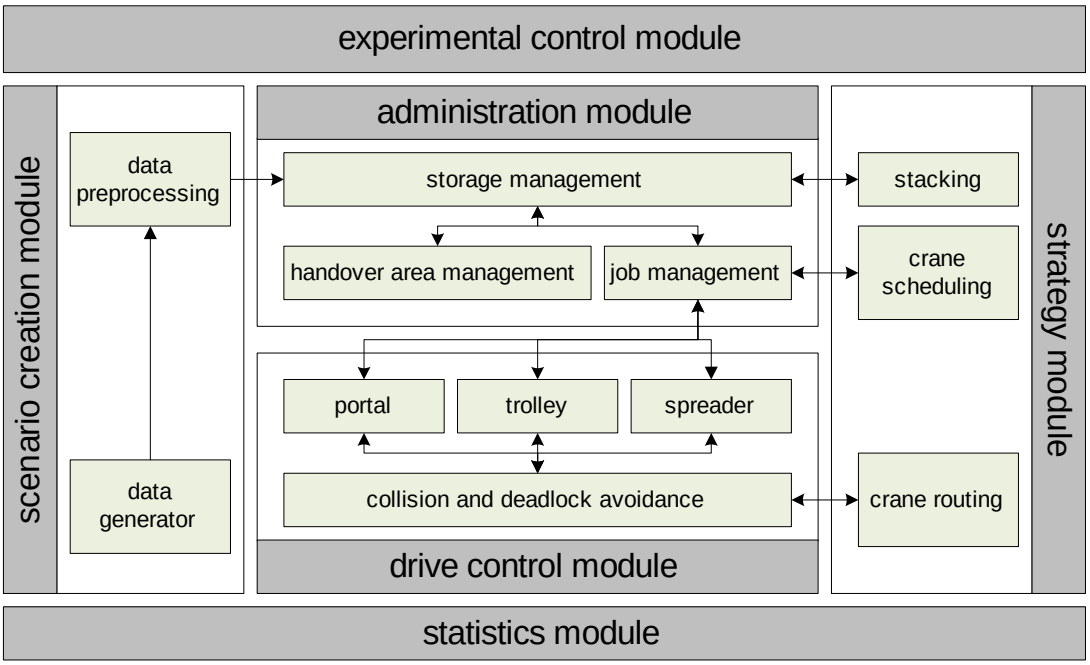

Figure 2.1: Architecture of simulation model.

Chapter 3

Scenario Creation Module

Meaningful results for practical terminal planning and operation do not just require a quite realistic simulation model, moreover a data generator is needed which creates close to reality scenarios. In fact, parameter-based data generation is a discipline on its own within the field of operations research — several works covering this problem in general (e.g. Hall and Posner, 2001) or for problem-specific settings (e.g. Kolisch and Sprecher, 1995; Hartmann, 2004) are available. The usage of parameters allows for systematic data generation according to a specific experimental setting. Within the framework of container storage yard simulations, data generation is also more than just generating containers with respect to a prescribed distribution function, since the mutual dependencies of real-world terminal functions are considerably complicated (see chapter 2 of Kemme, 2011). Means of transportation and containers have to be generated and each container has to be assigned to a mean of transportation for delivery and pick-up in a way that the predefined distributions on means of transportation, sizes of means of transportation, arrival times of means of transportation and dwell times are matched simultaneously (Hartmann, 2004).

The objective of a data generator for container terminal simulation and optimisation is to produce realistic scenarios that contain all required input data for the planned experiment. Usually, artificially generated scenarios are only the second-best choice, since real-world data is generally preferred (Hall and Posner, 2001). But this is not the case for container terminals to be built or to be expanded as well as for analysing future developments like larger vessels or changed vessel call patterns (Hartmann, 2004). Consequently, researchers tend to use artificially generated data. The literature on data generators for the special application of container terminals is not very heavyset. While some investigations make use of real-world data (e.g. Kang et al., 2006; Kozan and Preston, 1999), most references use artificially generated data (e.g. Petering et al., 2009; Kim et al., 2008). Since the data generation is usually not the primary focus of these references — this is an algorithm or a simulation study — only little insight into the data generation processes is given. So far, only the publications of Hartmann (2004) and Voogd et al. (1999) are directly devoted to data generation for container terminal simulation and optimisation. While Hartmann (2004) provides a detailed and generally applicable guidance for generating scenarios for seaport container terminals, Voogd et al. (1999) describes the data generation process for a case study which is dedicated to container stacking.

A central module of the RMGC simulation model described in this documentation is a two-phase scenario creation process. In the first phase, individual containers and means of transporta-

tion that will arrive at the terminal are generated according to user-defined parameter settings by the data generator submodule. Thereafter, in the second phase, the generated data on container and arrivals of means of transportation at the terminal are broken down to arrival times of containers and/or means of transportation at the modelled yard block based on simplifying assumptions for the operations of the QCs and waterside horizontal transport machines.

3.1 Data Generator

Like the simulation model itself, the data generator is a comparably complicated model. Therefore, its description is subdivided into several parts. Firstly, the output of the data generator that is needed by the simulation model is described. Next, the general structure of the applied data generator along with its main features and assumptions is explained, followed by a listing of the input parameters required by the generator programme. Afterwards, some necessary precalculations and the generation process for means of transportation as well as for containers is explained in detail. Finally, it is explained how container attributes as well as departing means of transportation and pick-up times for each individual container are generated.

3.1.1 Required Data Output for Simulation Model

Simplifying, the function of a yard block is to store containers which are delivered by internal or external vehicles and to retrieve the respective containers when they are needed by other vehicles. Thus, individual containers are required for the simulation, which means that each handled container should be identifiable by a unique number. The operational performance of yard block operations can be measured by the timely accuracy of storage and retrieval processes (i.e. the waiting times of vehicles in the transfer areas of the blocks). Therefore, it has to be known when a loaded vehicle arrives for delivery or an empty vehicle arrives for collection of a container. Hence, each container should be connected with a certain ingoing and outgoing date at the yard block. In addition, stacking strategies make use of the attributes of containers. Therefore, the size, the weight (-group), the PoD (port of destination) and the vessel a container is planned to depart with have to be known.

3.1.2 General Concept, Main Features and Assumptions

The model-specific data generator programme that is subsequently presented is based on the generators of Voogd et al. (1999) and Hartmann (2004). These generators are modified and extended in order to produce problem-specific data for simulating the operations of RMGC systems at seaport container terminals. In this subsection, the general concept of the data generator programme along with its main features and limitations is shortly introduced. The entire workflow for the creation of scenarios for an RMGC operated yard block is illustrated in figure 3.1.

Producing close to real-world distributions of container arrivals and departures necessitates the generation of individual means of transportation along with corresponding ETA (estimated time of arrival) and ETD (estimated time of departure). Here, the modes of transportation deep-sea vessel, feeder vessel and XT are distinguished. For reasons of simplicity it is assumed that transshipment containers are either going from feeder to deep-sea vessel or vice versa but

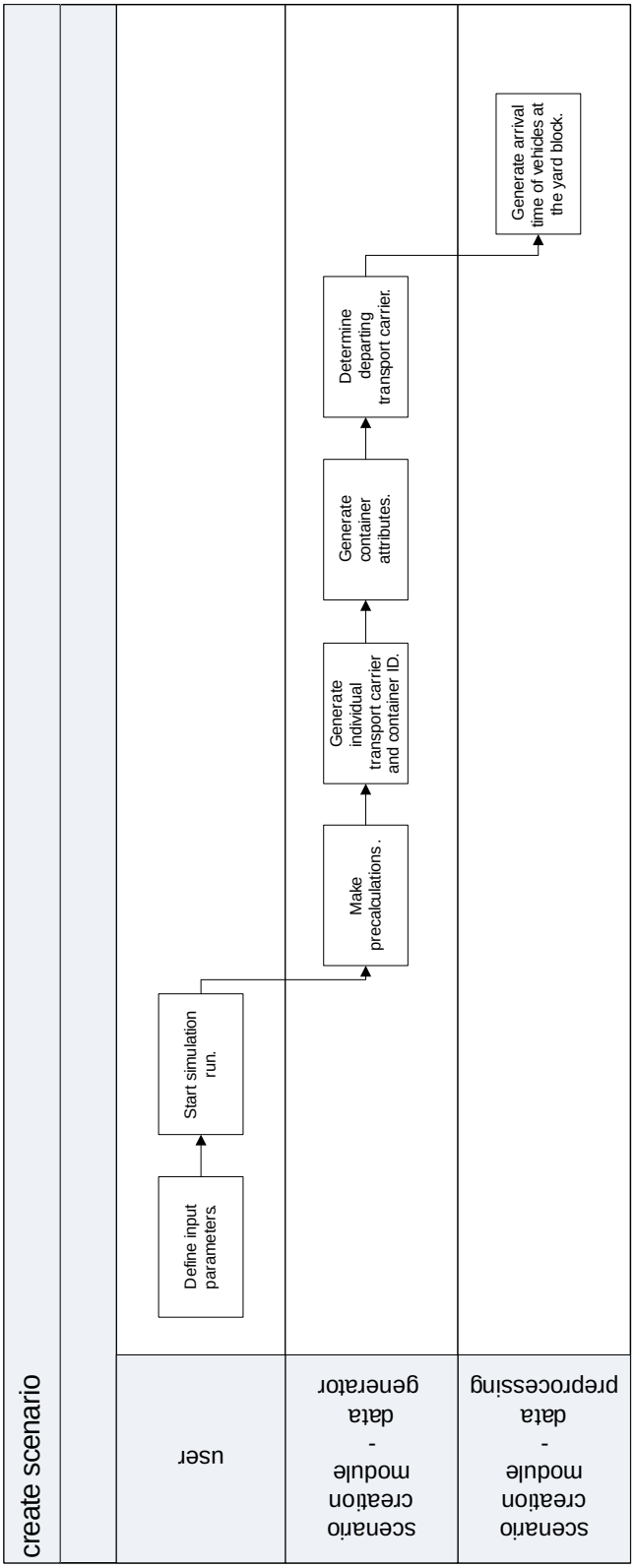

Figure 3.1: Process flow chart: 'create scenarios'.

never from deep-sea to deep-sea or feeder to feeder. Furthermore, it is assumed that the arriving mode of transshipment containers is equally distributed between deep-sea and feeder vessels. For import and export containers, it is assumed that import containers only arrive by and export containers only depart by deep-sea vessel and that the number of import containers equals the number of export containers. In addition, container flows from XT to XT are neglected. These assumptions allow for a simplified definition of the distributions between the modes of transportation and the corresponding container flows. The required calculations are explained in section 3.1.4.

The generator programme does not produce data for the whole terminal. Moreover, only the required data for the simulation of one RMGC operated yard block of any size is generated. Thus, after the input parameters have been defined, some precalculations are done in order to proportion some parameter settings — which have initially been made on the terminal level due to being commonly used figures in that field — to the yard block level. The container flows and the vessel sizes are for instance defined for the whole terminal level and on that score these values are proportioned to the block level. However, the proposed data generator can easily be adapted to produce close to real-world data for whole container terminal systems of any size and type.

On basis of the defined parameters and precalculations, individual means of transportation along with the related containers to be unloaded are generated. In this step of the generation process, the containers only have a unique ID, the other required data still has to be generated. In contrast to the data generation procedures of Voogd et al. (1999) and Hartmann (2004), here the length of the quay wall and therewith the berthing capacity of the terminal is explicitly considered. In the next step, the container-specific attributes size, weight and destination are generated according to the predefined distributions. The final step deals with the allocation of an outgoing mean of transportation and the related collection date to each initially generated container. Thereby, the defined dwell time distribution as well as the arrival time distributions for all modes are simultaneously taken into account. After the generation process, all produced means of transportation and containers are written to two separated files. The 'sailing list' contains in rows the information on all arriving deep-sea vessels, feeder vessels and XTs. The 'scenario data list' contains in rows all required information on the arriving containers in the simulation period. In order to ensure the reproducibility of all generated scenarios and thus the comparability of simulation results, all random and distribution-based decisions are instantiated by a controllable seed value.

3.1.3 Input Parameters

The input parameters can be distinguished into general parameters, terminal parameters, yard parameters, parameters for means of transportation and parameters for container attributes.

The general parameters are the length T of the simulation period and the length T^s of the warm-up period. Both values are measured in the number of weeks, since it is assumed that the deep-sea vessels follow a periodically weekly-repeated vessel call pattern (see section 3.1.5). The warm-up period is required, as the yard is not initially filled with containers when the simulation period begins and investigations are generally concerning the regular yard operations, not the setup process. Hence, the statistical data collection won't start until the warm-up period has finished and the yard block system has reached a steady state. Consequently, the simulation

results are based on $T - T^s$ weeks of regular yard block operation.

Required terminal parameters are the annual terminal throughput π^{through}, the length π^{length} of the quay wall, the fraction π^{ts} of transshipment containers and the number π_i^{qcm} of QC moves per hour for vessels of mode i. The annual terminal throughput π^{through} is measured in the number of containers handled by the QCs per year (Saanen, 2004). While the hinterland turnover is not considered for the terminal throughput, transhipment containers are counted twice, as they are unloaded and loaded by the QCs. The length π^{length} of the quay wall, which limits the number of vessel that can be served simultaneously, is measured in metres. The berthing time of a vessel clearly depends on the unloading and loading speed of the QCs. Here, a fixed number π_i^{qcm} of QC moves (i.e. containers per hour) for vessels of mode i is assumed. The fraction π^{ts} of transhipment containers defines the share of the throughput π^{through} which is induced by containers that are ingoing and outgoing by vessel. This parameter is used to compute the number C_i of containers delivered by mode i and the fraction c_{ij} of containers delivered by mode i and picked-up by mode j ($i, j \in \{\text{deep-sea, feeder, xt}\}$).

Yard parameters are the size of the RMGC yard blocks, the average filling rate of the yard block π^{fill}, the dwell time distribution and the block-factor π^{bf}. The size of the yard blocks is measured in the number of TEU-sized slots (twenty feet equivalent unit), which is given by the number n^{x} of bays, the number of n^{y} lanes and the number n^{z} of tiers per yard block. It is assumed that all yard blocks of the container terminal are equally sized. The average filling rate of the yard block describes the ratio between the net number of occupied slots and the theoretically available slots over the simulation period. The dwell time is the number of days a container spends in the yard. Depending on the type of dwell time distribution, different parameters are required. However, here the mean container dwell time $\overline{\delta}$ is introduced. The block-factor π^{bf} is a parameter which is required to adjust the data generation process from the terminal to the yard block size. With this parameter, the yard block distribution of containers associated with a certain vessel is controlled. In case of an uniform distribution, all blocks receive the same number of containers from/for a specific vessel, regardless of the distance to the respective berthing place. As this might be unrealistic, the uniformly distributed number of containers is multiplied with $\pi^{\text{bf}} \geq 1$. The greater π^{bf}, on the one hand, the more containers are received for/from a specific vessel, and on the other hand, the less vessels are related to a specific yard block.

For a realistic generation of the means of transportation some classes should be defined for the modes $i \in \{\text{deep-sea, feeder}\}$, since transport capacities and vessel lengths can — even for the same mode — vary up to a few thousand TEU and more than hundred metres, respectively. Thus, the number n_i^{class} of classes of transport mode i has to be defined. For XTs different classes are not considered and it is assumed that each XT either delivers a single container or picks up a single container. For each class k the fraction ρ_{ik}^{class} for the respective mode has to be defined with $\sum_k \rho_{ik}^{\text{class}} = 1$ for each mode i. In addition, for each class k an interval for the moves per call and the ship lengths has to be defined. The interval of the moves per call is given by the minimum number $\underline{S}_{ik}$ of moves per call in class k of mode i and the maximum number $\overline{S}_{ik}$ of moves per call. The interval of the ship length is given by the minimum length $\underline{L}_{ik}$ of a ship in class k of mode i and the maximum length $\overline{L}_{ik}$.

Finally, the arrival time distributions for each mode i has to be defined. While the arrivals of deep-sea vessels are mainly determined by a user-defined, weekly repeated vessel call pattern and

only the exact arrival times randomly vary by a few hours from week to week, the arrival times of XTs and feeder vessels are generated completely at random and there is no weekly repetition. Thus, for mode $i \in \{\text{deep-sea}\}$ a vessel call pattern has to be specified by the user, while for modes $i \in \{\text{xt, feeder}\}$ distributions on the weekly and daily arrival probabilities have to be specified. Differences in the arrival frequencies are possible for different days of the week and different time slots of the day. For the sake of simplicity, systematic differences in daily arrival distributions are neglected here. Thus, only the number n_i^{timeslot} of daily time slots for mode $i \in \{\text{xt, feeder}\}$, the first $\underline{H}_{it}$ and the last $\overline{H}_{it}$ hour of slot t and the fraction $\rho_{it}^{\text{timeslot}}$ of arrivals in time slot t of mode i have to be defined.

For the weekly repeated vessel call pattern, the user has to specify the number of weekly repeated vessel tours n^{vcp}. In addition, the user has to specify the deep-sea class k of tour l, the hourly planned arrival and departure times over the week of tour l as well as information on the concernment of the modelled yard block (i.e. if any container of tour l will be stored in the modelled block). Of course, each vessel of tour l has the same attributes. The vessel call pattern has to be specified by the user with respect to the quay wall and yard block capacities, which means that the definitions of π^{length}, $\underline{L}_{ik}$, $\overline{L}_{ik}$, the planned arrival and departure times, as well as the definitions of the yard block capacity, $\underline{S}_{ik}$, $\overline{S}_{ik}$, n^{vcp} and the concernment of the modelled yard block have to be well-matched. To make sure that the specified vessel call pattern will be feasible even after the concretisation of the vessel-individual moves per call, in the first instance the time interval between the planned arrival and departure times of tour l may be based on the maximum number of moves per call for vessels of the corresponding class. Thus, a quay wall capacity is granted to each vessel, that will be sufficient independently on the later chosen concrete number of moves per call for each individual deep-sea vessel.

The container attributes are the size, the weight and the destination. The data generator programme distinguishes 20' and 40' containers, 45' containers are not considered. The average size over all containers handled at a seaport container terminal in terms of TEU is given by the TEU-factor π^{teu}. Parameters defining the weight distribution of containers are the number n^{weight} of weight classes and the fraction ρ_w^{weight} of containers in weight class w. Parameters defining the destinations are the number n_i^{dest} of destinations for containers departing with mode i and the fraction ρ_{ig}^{dest} of containers with destination g among all containers departing with mode i.

3.1.4 Precalculations

The precalculations can be distinguished into two types. The first type is required due to only generating data for a single yard block and deals with the proportional adaption from the terminal size to the block size. The second type of precalculations deals with the definition of the container flows.

In order to proportion some parameters to the block level, the relation between the size of the single block and the size of the total yard has to be computed. Therefore, the fraction ρ^{block} of yard slots located in the simulated yard block is computed by

$$\rho^{\text{block}} = \frac{n^{\text{x}} \cdot n^{\text{x}} \cdot n^{\text{z}} \cdot \pi^{\text{fill}}}{\left\lceil \pi^{\text{through}} \cdot \pi^{\text{teu}} \cdot \left(1 - \frac{\pi^{\text{ts}}}{2}\right) \cdot \frac{\bar{\delta}}{365} \right\rceil}. \tag{3.1}$$

The numerator gives the number of averagely used slots in the regarded yard block, as the the number of theoretically available yard slots is multiplied with the average filling rate. The denominator gives the number of averagely used slots in the whole storage yard of the terminal by multiplying the number of containers stored in the yard per year with the fraction of a year that an average container stays in the yard. Since each transshipment container is handled twice by the QCs but only stored once in the yard, the throughput π^{through} of the terminal per year has to be reduced by these double handlings in order to get the number of containers stored in the yard per year.

For the definition of the container flows, the number C_i of containers delivered by mode i and the fraction c_{ij} of containers delivered by mode i and picked-up by mode j have to be computed. Based on the assumptions made in section 3.1.2, the number of containers delivered by deep-sea vessel, feeder vessel and XT are given by

$$C_{\text{deep-sea}} = \frac{n^{\text{x}} \cdot n^{\text{x}} \cdot n^{\text{z}} \cdot \pi^{\text{fill}} \cdot T \cdot 7}{\overline{\delta} \cdot \pi^{\text{teu}}} \cdot \frac{\frac{1-\pi^{\text{ts}}}{2} + \frac{\pi^{\text{ts}}}{4}}{1 - \frac{\pi^{\text{ts}}}{2}}, \tag{3.2}$$

$$C_{\text{feeder}} = \frac{n^{\text{x}} \cdot n^{\text{x}} \cdot n^{\text{z}} \cdot \pi^{\text{fill}} \cdot T \cdot 7}{\overline{\delta} \cdot \pi^{\text{teu}}} \cdot \frac{\frac{\pi^{\text{ts}}}{4}}{1 - \frac{\pi^{\text{ts}}}{2}} \text{ and} \tag{3.3}$$

$$C_{\text{xt}} = \frac{n^{\text{x}} \cdot n^{\text{x}} \cdot n^{\text{z}} \cdot \pi^{\text{fill}} \cdot T \cdot 7}{\overline{\delta} \cdot \pi^{\text{teu}}} \cdot \frac{\frac{1-\pi^{\text{ts}}}{2}}{1 - \frac{\pi^{\text{ts}}}{2}}, \tag{3.4}$$

respectively. The first terms of the equations give the total number of to be stored containers in the regarded block during the simulation period, while the second terms of equations 3.2, 3.3 and 3.3 give the fractions of containers delivered by deep-sea vessel, feeder vessel and XT, respectively. The numerator of the first terms give the number of used slot-days during the simulation period, as $n^{\text{x}} \cdot n^{\text{x}} \cdot n^{\text{z}} \cdot \pi^{\text{fill}}$ give the number of averagely used slots, and $T \cdot 7$ give the number of days in the simulation period. Since 40' containers require two slots and containers normally stay a few days in the block, the number of used slot-days is divided by the average dwell time and the TEU-factor, yielding the total number of to be stored containers in the regarded block during the simulation period.

Considering the previously explained assumptions, the second terms give the fractions of containers delivered by the considered mode of transportation only on basis of the fraction π^{ts} of transshipment containers. Clearly, this implies $\frac{\frac{1-\pi^{\text{ts}}}{2}+\frac{\pi^{\text{ts}}}{4}}{1-\frac{\pi^{\text{ts}}}{2}} + \frac{\frac{\pi^{\text{ts}}}{4}}{1-\frac{\pi^{\text{ts}}}{2}} + \frac{\frac{1-\pi^{\text{ts}}}{2}}{1-\frac{\pi^{\text{ts}}}{2}} = 1$. The idea behind these fractions can best be explained by a small numerical example. In case $\pi^{\text{ts}} = 0.4$ and 100 containers are handled by the QCs, $100 \cdot \left(1 - \frac{\pi^{\text{ts}}}{2}\right) = 80$ containers arrive in the yard, thereof $100 \cdot \frac{\pi^{\text{ts}}}{2} = 80 \cdot \frac{\frac{\pi^{\text{ts}}}{2}}{1-\frac{\pi^{\text{ts}}}{2}} = 20$ are transshipment containers and $100 \cdot \frac{\pi^{\text{ts}}}{4} = 80 \cdot \frac{\frac{\pi^{\text{ts}}}{4}}{1-\frac{\pi^{\text{ts}}}{2}} = 10$ arrive by deep-sea and feeder vessel, respectively. Consequently, $80 \cdot \left(1 - \frac{\frac{\pi^{\text{ts}}}{2}}{1-\frac{\pi^{\text{ts}}}{2}}\right) = 80 \cdot \left(\frac{1-\pi^{\text{ts}}}{1-\frac{\pi^{\text{ts}}}{2}}\right) = 60$ are either import or export. Since half of these 60 containers arrive by XT and deep-sea vessel respectively, altogether $80 \cdot \frac{\frac{1-\pi^{\text{ts}}}{2}+\frac{\pi^{\text{ts}}}{4}}{1-\frac{\pi^{\text{ts}}}{2}} = 40$ container arrive by deep-sea vessel, $80 \cdot \frac{\frac{\pi^{\text{ts}}}{4}}{1-\frac{\pi^{\text{ts}}}{2}} = 10$ container arrive by feeder vessel and $80 \cdot \frac{\frac{1-\pi^{\text{ts}}}{2}}{1-\frac{\pi^{\text{ts}}}{2}} = 30$ container arrive by XT. The relation between the transshipment factor π^{ts} and the fraction of containers arriving by deep-sea vessel, feeder vessel and XT is depicted in figure 3.2. The fraction of container arriving by deep-sea vessel

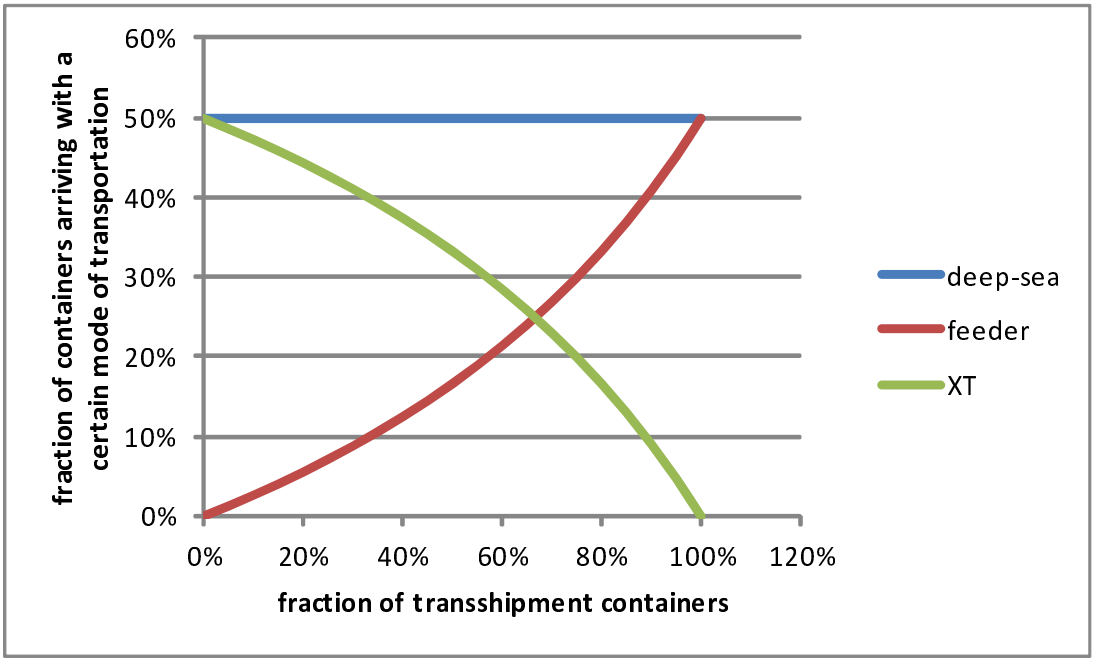

Figure 3.2: Effects of the fraction of transshipment containers.

equals independently of the fraction of transshipment containers 50%, while the fraction of XT containers decreases and the fraction of feeder containers increases with increasing transshipment rate.

Due to the previously made assumptions on the container flow, it follows that $c_{\text{deep-sea,deep-sea}} = c_{\text{feeder,feeder}} = c_{\text{feeder,xt}} = c_{\text{xt,feeder}} = c_{\text{xt,xt}} = 0$. Additionally, all containers arriving by XT or feeder vessel will be picked-up by deep-sea vessel (i.e. $c_{\text{feeder,deep-sea}} = c_{\text{xt,deep-sea}} = 1$). From the assumption also follows that all transshipment containers arriving by deep-sea vessel will be picked-up by a feeder vessel and that the remaining containers arriving by deep-sea vessel will be picked-up by XT. Hence, these fractions are computed by

$$c_{\text{deep-sea,feeder}} = \frac{\frac{\pi^{\text{ts}}}{4}}{\frac{1-\pi^{\text{ts}}}{2} + \frac{\pi^{\text{ts}}}{4}}, \tag{3.5}$$

$$c_{\text{deep-sea,xt}} = \frac{\frac{1-\pi^{\text{ts}}}{2}}{\frac{1-\pi^{\text{ts}}}{2} + \frac{\pi^{\text{ts}}}{4}} \tag{3.6}$$

with the transshipment factor being the only influence parameter. Reviving the previous example with $\pi^{\text{ts}} = 0.4$ leads to $c_{\text{deep-sea,feeder}} = \frac{1}{4}$ and $c_{\text{deep-sea,xt}} = \frac{3}{4}$.

3.1.5 Generating Means of Transportation and Containers

After the precalculations are completed, the generator programme starts the generation process of individual means of transportation and containers. In general, the generation process is similar for all modes of transport. Firstly, the individual means of transportation along with the numbers of containers delivered by the related means of transportation for the regarded yard block are generated. Thereafter, the berthing times of the means of transportation are computed, followed by the generation of individual containers along with the related arrival data and the departing

mode. However, differences in the generation process still exist. Therefore, the generation process for deep-sea vessels is explained in detail and for feeder vessels and XTs the variations are shortly described.

The generation of deep-sea vessels is based on the user-defined vessel call pattern. Firstly, T individual vessels of deep-sea tour l are generated, if the modelled yard block is concerned with that tour l. The classes of these vessels as well as their arrival times over the week are more or less given by the user-defined tour l. Only the exact arrival time a_v^{vessel} of each individual vessel v varies randomly within an equally-distributed time interval around the user-defined planned arrival time for vessels of tour l, in order to realistically reproduce some stochastic differences of the realised arrivals compared to the vessel call pattern. Next, the number S_v of containers that is delivered by vessel v is generated in two steps: Firstly, each vessel v is assigned a class-dependent, equal-sized, minimum number of containers $S_v^{\min}$ to be unloaded by vessel v, in order to ensure that container volume of each generated vessel is connected with the modelled yard block. Secondly, some additional containers are randomly distributed over all generated vessels, in order to reproduce stochastic variations of the container deliveries between different vessels. In the first step, the minimum number of containers $S_v^{\min} = \underline{S}_{\text{deep-sea},k} \cdot 0.5 \cdot \rho^{\text{block}} \cdot \pi^{\text{bf}} - \epsilon$ is computed by a repeated procedure, that enlarges the correction parameter ϵ, until $\sum_v S_v^{\min} < C_{\text{deep-sea}}$. Initially, ϵ is set to be zero. In the second step, the final number of to be delivered containers $S_v = S_v^{\min} + S_v^{\text{add}}$ is generated for each individual vessel on basis of $S_v^{\min}$ and a random additive S_v^{add}, which is an equal-distributed number chosen from $\left[0, \left(\overline{S}_{\text{deep-sea},k} - \underline{S}_{\text{deep-sea},k}\right) \cdot 0.5 \cdot \rho^{\text{block}} \cdot \pi^{\text{bf}}\right]$, until $\sum_v S_v = C_{\text{deep-sea}}$. Next, the departure times d_v^{vessel} of all individual vessels are determined. In order to consider differences in the QC productivity, the berthing time of vessel v is a normally distributed variable $b_v^{\text{vessel}} \sim N\left(\frac{2 \cdot S_v}{\pi_{\text{deep-sea}}^{\text{qcm}}}, 0.1 \cdot \frac{2 \cdot S_v}{\pi_{\text{deep-sea}}^{\text{qcm}}}\right)$ which depends on the time needed to unload and load all containers from/on vessel v, which roughly is $\frac{2 \cdot S_v}{\pi_{\text{deep-sea}}^{\text{qcm}}}$. The departure time of vessel v is then computed by $d_v^{\text{vessel}} = a_v^{\text{vessel}} + b_v^{\text{vessel}}$.

As for all containers that arrive by feeder vessel and XT, the outgoing deep-sea vessel has to be known (see section 3.1.1) — even containers that arrive shortly before T and that are planned for departure after T — additional deep-sea vessels and the corresponding arrival dates are created for some weeks after T. Thus, a concrete deep-sea vessel can be determined for each container that arrives by feeder vessel or XT(see section 3.1.7).

Finally the individual containers that are delivered by each deep-sea vessel are generated. Besides the unique ID of the containers, the related arrival date and the departing mode are also generated. The arrival date $a_c^{\text{container}}$ of an individual container c is generated by drawing a random time out of the interval $\left[a_v^{\text{vessel}}, a_v^{\text{vessel}} + 0.5 \cdot b_v^{\text{vessel}}\right]$, which roughly represents the unloading time of the related vessel v. The pick-up mode of container c is drawn in the style of the optimised procedure of Hartmann (2004), which means that the pick-up mode for the first vessel is randomly chosen between feeder vessel and XT, while for each following vessel the pick-up mode with the currently greatest difference between the user-defined fractions of $c_{\text{deep-sea,feeder}}$ and $c_{\text{deep-sea,xt}}$ and the actually realised fractions of these container flows is selected, thus ensuring that the fractions $c_{\text{deep-sea,feeder}}$ and $c_{\text{deep-sea,xt}}$ are roughly realised in the end.

For feeder vessels and XTs, the generation steps are slightly different, as no user-defined vessel call pattern exists. For the generation of a feeder vessel, a class k out of the number $n_{\text{feeder}}^{\text{class}}$ is drawn in the style of the optimised procedure of Hartmann (2004). Next, the number S_v of con-

tainers that are delivered by vessel v is randomly chosen from $\left[\underline{S}_{\text{feeder},k} \cdot 0.5 \cdot \rho^{\text{block}} \cdot \pi^{\text{bf}}, \overline{S}_{\text{feeder},k} \cdot 0.5 \cdot \rho^{\text{block}} \cdot \pi^{\text{bf}}\right]$. For all XTs the number S_v is simply set to 1, which means that each generated truck only delivers one container. The generation of individual feeder vessels and trucks is repeated until the total number of delivered containers equals C_{feeder} and C_{xt}, respectively. The arrival date of a feeder vessel is generated by simply drawing a random time out of the interval $[0, T]$. Unlike deep-sea vessels, it is assumed that feeder vessels do not arrive according to a periodical vessel call pattern. However, the berthing capacity has to be respected, therefore for each feeder vessel an arrival time is drawn until an appropriate one is found. According to the previously made assumptions, the departing mode of a container that arrives by feeder vessel always is a deep-sea vessel. The arrival dates of individual XTs are generated by first drawing a random arrival day out of $[0, T]$. Thereafter, a time slot t out of the number $n_{\text{xt}}^{\text{timeslot}}$of time slots with probability $\rho_{\text{xt},t}^{\text{timeslot}}$ is randomly drawn. Finally, an arrival time in the interval $\left[\underline{H}_{\text{xt},t}; \overline{H}_{\text{xt},t}\right]$ is randomly chosen. The departing mode of a container that arrives by XT always is a deep-sea vessel. The container generation process for containers that are delivered by XT and/or feeder is the same as for deep-sea vessels.

At the end of this part of the generation programme, individual means of transportation and individual containers are generated. In Addition, the arrival dates and the pick-up mode of each container are known.

3.1.6 Generating Container Attributes

In the next part of the data generation process, the required container attributes of size, weight and destination are generated for all containers that have been produced in the previous part. Firstly, the weight of a container is determined by drawing a weight group w with probability ρ_w^{weight} out of the defined number n^{weight} of weight groups. Based on the user-defined TEU-factor π^{teu}, the size of each container is determined with the optimised procedure of Hartmann (2004). The destination of an individual container is generated by randomly drawing a destination g with probability ρ_{ig}^{dest} out of the number n_i^{dest} of possible destinations for containers that are planned to depart with mode i.

At the end of this part of the generation programme, the container attributes of size, weight, and destination are known for each individual container.

3.1.7 Generating Departing Means of Transportation and Pick-Up Time

The only missing information for each individual container c are the concrete mean of transportation that is planned to collect container c and the departure time $d_c^{\text{container}}$ of container c with the collecting mean of transportation. Both information is generated in the final part of the generator programme.

The determination of a collecting mean of transportation is quite similar for containers that are planned to depart by deep-sea or feeder vessel. In the first step, the individual dwell time δ_c of container c is randomly drawn based on the shifted exponential probability distribution function

$$f(\delta_c) = \begin{cases} \frac{1}{\bar{\delta}-1} e^{\frac{-\delta_c}{\bar{\delta}-1}} & \delta_c \geq 1 \\ 0 & \delta_c < 1 \end{cases}, \tag{3.7}$$

that is specified by the user-defined mean container dwell time $\overline{\delta}$. In this way, both a minimum dwell time of one day for each container c (i.e., $\delta_c \geq 1.0$ days, $\forall c \in C$) and an expectancy value for the container dwell time of $\overline{\delta}$ days are ensured. Owing to possible differences between the dwell time distributions of transshipment containers and import-export containers (Drewry, 1998), a mean container dwell time $\overline{\delta}^{\text{ts}}$ for transshipment containers and a mean container dwell time $\overline{\delta}^{\text{ie}}$ for import export containers can be separately specified, with $\overline{\delta} = \frac{\pi^{\text{ts}}}{2} \cdot \overline{\delta}^{\text{ts}} + \left(1 - \frac{\pi^{\text{ts}}}{2}\right) \cdot \overline{\delta}^{\text{ie}}$ defining the mean container dwell time over all types of containers. In the next step, the sailing list is checked for an appropriate vessel v, which is characterised by being of the required mode of transportation, having sufficient loading capacity left and having a loading interval $\left[a_v^{\text{vessel}} + \frac{d_v^{\text{vessel}} - a_v^{\text{vessel}}}{2}, d_v^{\text{vessel}}\right]$ that covers the time the container c has to be picked-up according to the drawn dwell time. This procedure is repeated until a dwell time δ_c has been drawn for which an appropriate vessel v exists. For containers with an XT as departing mode the generation process is different. A new XT arrival is generated for each container to be collected. Hence, no appropriate XT has to be searched for, but the XT arrivals have to be generated according to the predefined number $n_{\text{xt}}^{\text{timeslot}}$ of time slots and the related fractions $\rho_{\text{xt},t}^{\text{timeslot}}$ of arriving XTs within that time slot.

The generation of the actual departure date $d_c^{\text{container}}$ for an individual container c is quite different for all three modes. In case the departing mode is an XT, the departure date is just the arrival date of the previously generated collecting XT. For feeder vessels as departing mode, a time is randomly chosen out of the loading interval $\left[a_v^{\text{vessel}} + \frac{d_v^{\text{vessel}} - a_v^{\text{vessel}}}{2}, d_v^{\text{vessel}}\right]$ of the collecting vessel v that is assigned to collect container c. In case of a deep-sea vessel two possibilities exist: Firstly, like for feeder vessels, a time can be randomly chosen out of the loading interval of the collecting vessel. Secondly, if online stowage planning is applied, no concrete pick-up date is generated for the container. Moreover, the departing date will be generated just a few hours before the loading process of the assigned vessel starts. In addition, containers of the same category are exchangeable for each departing date (see chapter 5 of Kemme, 2011). Here, online stowage planning is implemented in three steps. Firstly, during the data generation process for each deep-sea vessel a date is determined for the creation of the stowage plan of the corresponding vessel. This date is a user-defined time t^{os} before the loading process of that vessel starts. Hence, the creation date of a stowage plan for vessel v is defined by $a_v^{\text{vessel}} + \frac{d_v^{\text{vessel}} - a_v^{\text{vessel}}}{2} - t^{\text{os}}$. Secondly, when the simulation time reaches the creation date of a stowage plan, a pick-up date for each container that has to be loaded by the relevant vessel is randomly chosen out of the loading interval $\left[a_v^{\text{vessel}} + \frac{d_v^{\text{vessel}} - a_v^{\text{vessel}}}{2}, d_v^{\text{vessel}}\right]$ of the vessel. Thirdly, the pick-up dates of containers that belong to the same category are exchanged during the loading process such that containers that require fewer shuffle moves are loaded first.

At the end of this part of the scenario generation process all required data (cf. 3.1.1) for the simulation of a single RMGC yard block is available. But so far, the generated dates $a_c^{\text{container}}$ and $d_c^{\text{container}}$ represent the container arrival and departure times at/from the terminal interfaces, respectively, but not at/from the modelled yard block. Thus, the generated arrival and departure dates have to be adjusted to the yard block level, which is done by the data preprocessing submodule, that is explained in the next section.

3.2 Data Preprocessing

Like the presented data generator, the data preprocessing submodule is a part of the scenario creation module (cf. figure 2.1). But here no new data is generated, moreover the previously generated data is adjusted to the yard block level by simplifying taking into account the waterside processes of the QCs and the waterside horizontal transport equipment (i.e. SC or AGVs). This is required due to a timely discrepancy between the generated data and the data required for the simulation model. So far the arrival date $a_c^{\text{container}}$ and the departure date $d_c^{\text{container}}$of an individual container c are based on the arrivals of the relevant vessels and XTs at the terminal's waterside and landside interfaces. But the interfaces of the simulation model are the waterside and landside handover areas of the modelled yard block. For the dates on landside container arrivals and departures, it is simplifying assumed that all XTs arrive directly at the yard block, so that the generated XT arrival times actually represent the needed container arrival and departure times at the yard block. Of course, vessels cannot be assumed to arrive directly at the yard block. Therefore, the time between the vessel and the yard block — which is the loading and unloading time of QCs as well as the transportation time of horizontal transport machines — has to be taken into account for the arrival and departure dates of containers at the waterside handover area of the modelled yard block.

So far, for containers that arrive by vessel, the date $a_c^{\text{container}}$ is the time container c is picked-up by a QC for unloading from the arriving vessel. Similarly, for containers that depart by vessel, the date $d_c^{\text{container}}$ is the time container c is dropped by a QC on the vessel that container is planned to depart with. The waterside processing time $t^{\text{wsprocess}}$, which is the time between starting/finishing the pick-up/drop operation of a container on the vessel and the arrival/departure at the modelled yard block depends on several factors like driver skills, traffic jams, weather conditions and the degree of coordination between QC and horizontal transport operations. Thus, the waterside processing time $t^{\text{wsprocess}}$ may be considered as a stochastic component. However, here a user-defined fixed value is assumed for the waterside processing time, in order to exclude any effects of the QC and waterside horizontal transport operations for the investigation of RMGC systems as aimed at with this simulation model.

Finally, the arrival and departure times at the yard block are yielded by adding and subtracting the waterside processing time $t^{\text{wsprocess}}$, respectively. For waterside arrivals, the actual arrival of a container at the yard block is $t^{\text{wsprocess}}$ after the pick-up by the QC on the vessel (i.e. $a_c^{\text{container}} + t^{\text{wsprocess}}$). For waterside departures, the actual departure of a container from the yard block is $t^{\text{wsprocess}}$ before the container is placed on the vessel by the QC (i.e., $d_c^{\text{container}} - t^{\text{wsprocess}}$).

At the end of this process step (see figure 3.1), all required data for the simulation of a RMGC operated yard block is generated and adjusted to the yard block level.

Chapter 4

Drive Control Module

The drive control module consists of four submodules — namely the drive control for all crane movements (i.e. portal, trolley and spreader) as well as the collision and deadlock avoidance submodule (see figure 2.1). While the first three submodules are needed for each crane system, the latter one is only needed for multi-crane RMGC systems in order to avoid collisions and/or deadlocks between the cranes of TRMGC, DRMGC and TriRMGC systems. Table 4.1 shows the events of the simulation model that are linked with the drive control module. There are two types of events: main and interface. Main events describe changes in the state of the crane, while interface events fulfil an interface function to other modules and submodules of the simulation model. The events 'job received' and 'job finished' are the connections to the administration module and the events 'shunting initiated' and 'shunting finished' are the interface to the collision and deadlock avoidance submodule.

This section is organised as follows: Firstly, the execution and the control of all crane movements are explained, which include the submodules for portal, trolley and spreader movements. Thereafter, a description of the collision and deadlock avoidance submodule is given.

4.1 Crane Movements

The movements of gantry cranes are caused by two reasons. Firstly, transport jobs require the crane to move to the pick-up and drop positions of the corresponding job. Such a crane movement

main events	interface events
portal/trolley/spreader starts acceleration	job received
portal/trolley/spreader reaches top speed	job finished
portal/trolley/spreader starts deceleration	shunting initiated
portal/trolley/spreader stops	shunting finished
portal/trolley/spreader reaches pick-up position	
portal/trolley/spreader reaches drop position	
portal/trolley/spreader reaches break point x/y/z	
portal reaches shunting position	
trolley reaches crossing position	
spreader reaches driving position	
spreader finishes fine positioning	

Table 4.1: Discrete events relevant to the drive control module.

is initiated by the job management submodule, which sends the driving coordinates for a job to the drive control module and induces the event 'job received'. Secondly, within multi-crane systems, a crane may be required to drive to a shunting position due to collision avoidance and right of way rules. These crane movements are initiated (event: 'shunting initiated') by the collision and deadlock avoidance submodule, which decides on the right of way in close cooperation with the crane routing submodule and sends the coordinates of the shunting position to the drive control module.

Immediately after receiving the driving coordinates of a job or a shunting position, the crane is prompted by the drive control module to start the required movements to the received coordinates — if not already standing there. The crane movement consists of portal, trolley and spreader movements, which are executed in certain sequences depending on the respective crane system. First of all, the spreader is only moved if both portal and trolley have stopped and have already reached the respective driving coordinates. Conversely, portal and trolley are only allowed to start a new movement if the spreader has returned to its driving position. To be in the driving position, the spreader has to be completely hoisted, so that no collisions with containers in the top layer of the storage block can occur during portal and trolley movements. For crane systems with crossing capabilities (i.e. DRMGC and TriRMGC systems), additional sequence restrictions for portal and trolley movements of the outer large crane have to be regarded, since collisions with the inner small crane(s) have to be avoided. Regularly, the trolley of the outer large crane has to be positioned in the special crossing position, which is at the side of the portal, beyond the profile of the inner small crane. The trolley of the outer large crane is only allowed to leave this crossing position after permission by the collision and deadlock avoidance submodule. For the same-sized (inner small) cranes in the TRMGC and TriRMGC systems, simultaneous portal and trolley movements are allowed without any restrictions.

The driving coordinates of each crane movement are given in metres alongside the x-axis (bays), y-axis (lanes) and z-axis (tiers) of the yard block. Each driving coordinate is a multiple of the metrics of a storage slot, which can be freely scaled in the simulation model. Here, according to the metrics of a 20' container and some safety margin between neighbouring bays and lanes, the size of a storage slot is by default set to 6.4 m long, 2.8 m wide and 2.6 m high. Each driving coordinate is located in the middle of the x-y-metrics and on top of the z-metric of a storage slot. Thus, a 20' container is picked alongside the middle of its x-axis, but as a 40' container occupies two slots it can not be picked in middle. Moreover, it is assumed that the driving coordinates for picking-up or dropping a 40' container is the slot which is closer to the waterside handover area. This is just a small simplification and the effects can be neglected since differences in the total driving distances are balanced out due to longer distances to/from the landside handover area and shorter distances to/from the waterside handover area.

The crane only has to stop its movements at potential driving coordinates. Thus, breakpoints for portal, trolley and spreader movements are implemented alongside the x-, y- and z-axis for all possible driving coordinates. The driving coordinate and breakpoint issues are illustrated for the x-y-dimensions by a top view of the modelled yard block in figure 4.1. It is shown that alongside the x-axis $n^{\mathrm{x}}+4$ breakpoints for the portal movements with distances of 6.4 m are implemented, whereof the first two and the last two breakpoints represent the waterside and landside handover areas, respectively. Alongside the y-axis, n^{y} breakpoints with distances of 2.8 m are implemented

and in case of RMGC systems with crossing capabilities, an additional breakpoint — representing the crossing lane — is implemented 5.6 m apart from the last breakpoint. Alongside the z-axis, $n^z + 1$ breakpoints are implemented with distances of 2.6 m between the first n^z breakpoints and a distance of 3.0 m to the last breakpoint, which is the driving position of the spreader. The breakpoints are used by the drive control module for the execution of certain crane movements. Each time a breakpoint is reached by portal, trolley and spreader (event: 'portal/trolley/spreader reaches break point x/y/z'), it is checked if the respective movement has to be stopped because either the destination is reached or a collision has to be avoided. The driving coordinates for a container that is stored in bay 10, lane 5 and tier 3 are for example (76.8 m, 12.6 m, 7.8 m).

Furthermore, the millimetre-exact positions of portal, trolley and spreader are controlled by the drive control module for each thousandth part of a second. Therefore, acceleration and deceleration of all crane movements can be precisely modelled. Different values for velocity, acceleration and deceleration can be defined for portal, trolley and spreader movements of each crane, depending on whether the crane is laden or not.

So far, all functions of the drive control module are deterministic. However, in practise, even for an automated RMGC system, some stochasticity is observed. Containers cannot always be placed exactly in the center of the respective storage slots as wind and other influence factors have to be regarded. Consequently, some sensor-controlled fine positioning of the spreader may be required that cannot be predicted in advance. Here, the period of time for the fine positioning of the spreader along with the time for twist-lock handling is assumed to be a random gamma-distributed variable. Considering that container pick-up and drop operations inside the yard block and in the waterside and landside handover areas are to different degrees affected by stochastic influences, thus inducing different distributions of the fine positioning times, the simulation model allows for the parametrisation of different gamma distributions for container handling in the storage block, in the waterside handover area and in the landside handover area.

4.2 Collision and Deadlock Avoidance

An important topic within multi-crane RMGC systems is the avoidance of collisions and deadlocks. A collision means an accident of two cranes (i.e. one crane drives against the other crane), while a deadlock in general occurs if two or more processes have stopped and they are waiting one of the other processes to fulfil a certain criterion. But since the observed processes are all stopped, the awaited criterion will never be fulfilled — the processes are trapped in a deadlock situation. Within the TRMGC system such a situation is easily imaginable: The waterside crane is in bay 8 and has to drive to bay 20, while the landside crane is in bay 16 and has to drive to bay 4. A collision avoidance mechanism would prevent them from colliding somewhere around bay 12, but then both cranes would have stopped, each of them awaiting the other crane to clear the way to its destination. Consequently, it has to be decided on basis of some criteria which crane has to clear the way and which one is allowed to drive first. Altogether, within RMGC systems collision and deadlock avoidance are two coherent issues that have to be treated simultaneously. A description of a collision and deadlock avoidance mechanism for TRMGC and DRMGC systems is for example given in Valkengoed (2004).

Within this simulation model, a claiming-based collision and deadlock avoidance mechanism

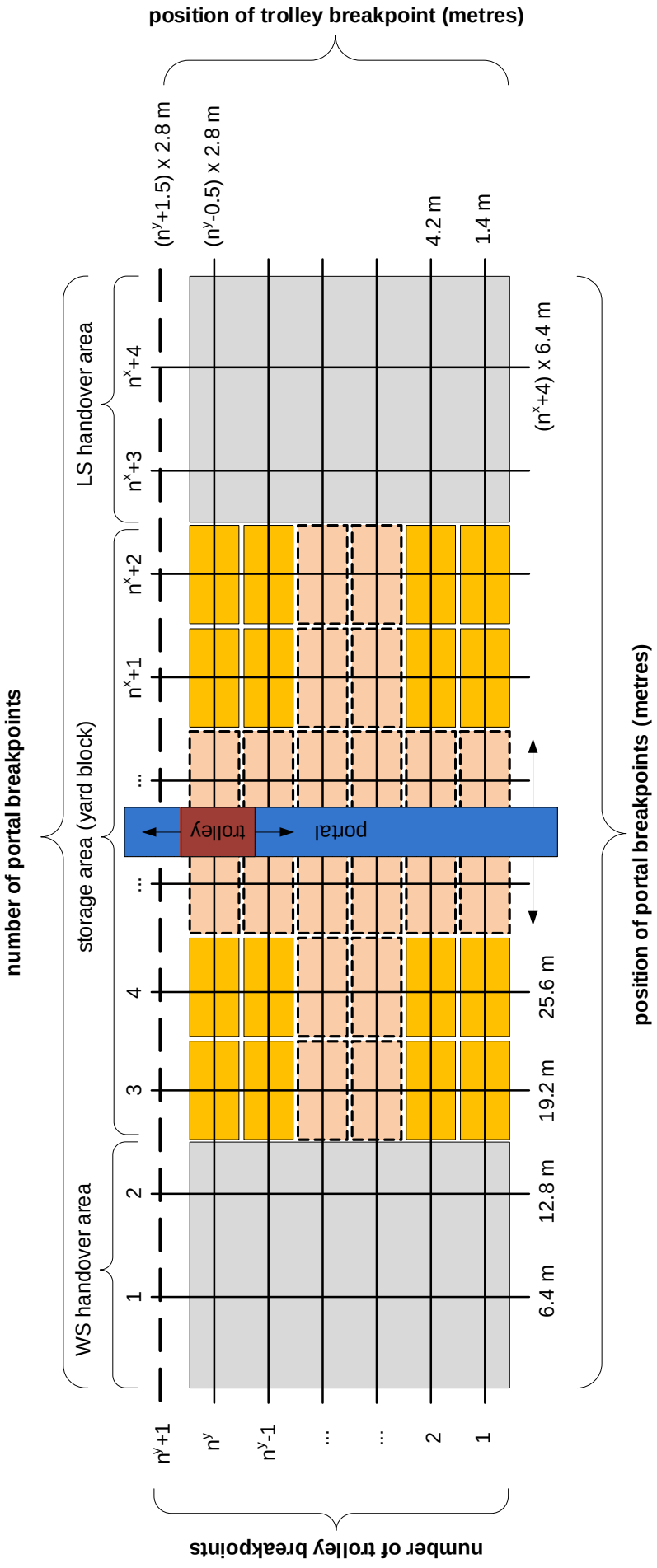

Figure 4.1: Schematic top view of modelled yard block.

is applied. Basically, the x-axis is divided into several sections — so-called claims — and a crane is only allowed to drive into such a section if it has previously been reserved for the respective crane (i.e. it has been claimed). Otherwise, the crane is not allowed to start driving or it has to stop. However, here a section only needs to be claimed by a crane, when the corresponding yard bay is blocked by that crane for storage and/or retrieval operations of other cranes. While a small crane is always blocking the yard bays it is located in, an outer large crane is only blocking these bays when its trolley is not located in the crossing position. Otherwise, containers below the portal of the outer large crane are still accessible for a small crane. The sections are zoned according to the breakpoints along the x-axis of the storage block. Each section covers 3.2 m to the left and 3.2 m to the right of a certain breakpoint; which means a section is equivalent to the length of a storage slot (see figure 4.1).

Before a certain section is claimed by a crane, several tests and decisions are required by the collision and deadlock avoidance submodule. Most tests and decisions depend on the used crane system, but some rules apply for all multi-crane systems. Firstly, no overlap is possible between claims of different cranes. Secondly, a section can only be claimed by a certain crane if it has not already been claimed by another crane. However, this does not mean a strict application of the FCFS-principle (first-come-first-served) — a claim for a certain crane can be canceled in order to allow claiming for another crane. Thirdly, in order to ensure a safety distance to other cranes, a minimum of three sections is always claimed by each crane. Besides the section the crane is located in, also the sections to the right and to the left are claimed by that crane. Hence, on both sides of the central point of the crane 9.6 m are claimed, which is a reasonable safety distance (Valkengoed, 2004).

The collision and deadlock avoidance mechanism for the TRMGC system is probably the simplest one, since crane conflicts are only possible in certain situations. When the waterside crane is running in the waterside direction, no conflicts can occur, since the landside crane can never be positioned closer to the waterside than the waterside crane itself. Thus, claiming is always possible up to its target position for the waterside crane when moving to the waterside. The same applies vice versa for the landside crane when running in the landside direction.

A schematic representation of the collision and deadlock avoidance mechanism for the TRMGC system is given in figure 4.2. Each time a crane receives new driving coordinates or it passes a breakpoint, this process is started. Firstly, it is checked whether a predecessor of the crane's job has to be picked first at its destination. In case a container has to be shuffled by the other crane before the regarded crane can pick its container, it is checked next if the other crane can reach its destination without conflicts. If this is possible, the sections up to the target position of the other crane are granted to the regarded crane. If not, claiming is denied and a shunting position is determined so that the other crane can reach its target position in order to shuffle the container.

In case no predecessors have to be picked first, it is checked for claiming conflicts with the other crane. When no conflicts arise, the whole distance to the crane's destination is claimed. Otherwise, it is firstly checked if the two cranes are running towards each other, which means that the target positions are located in a way that not both of them can be reached simultaneously. In that case it is checked next, if at least the other crane can reach its target destination without the need for shunting operations of the regarded crane. If this applies, the sections are claimed

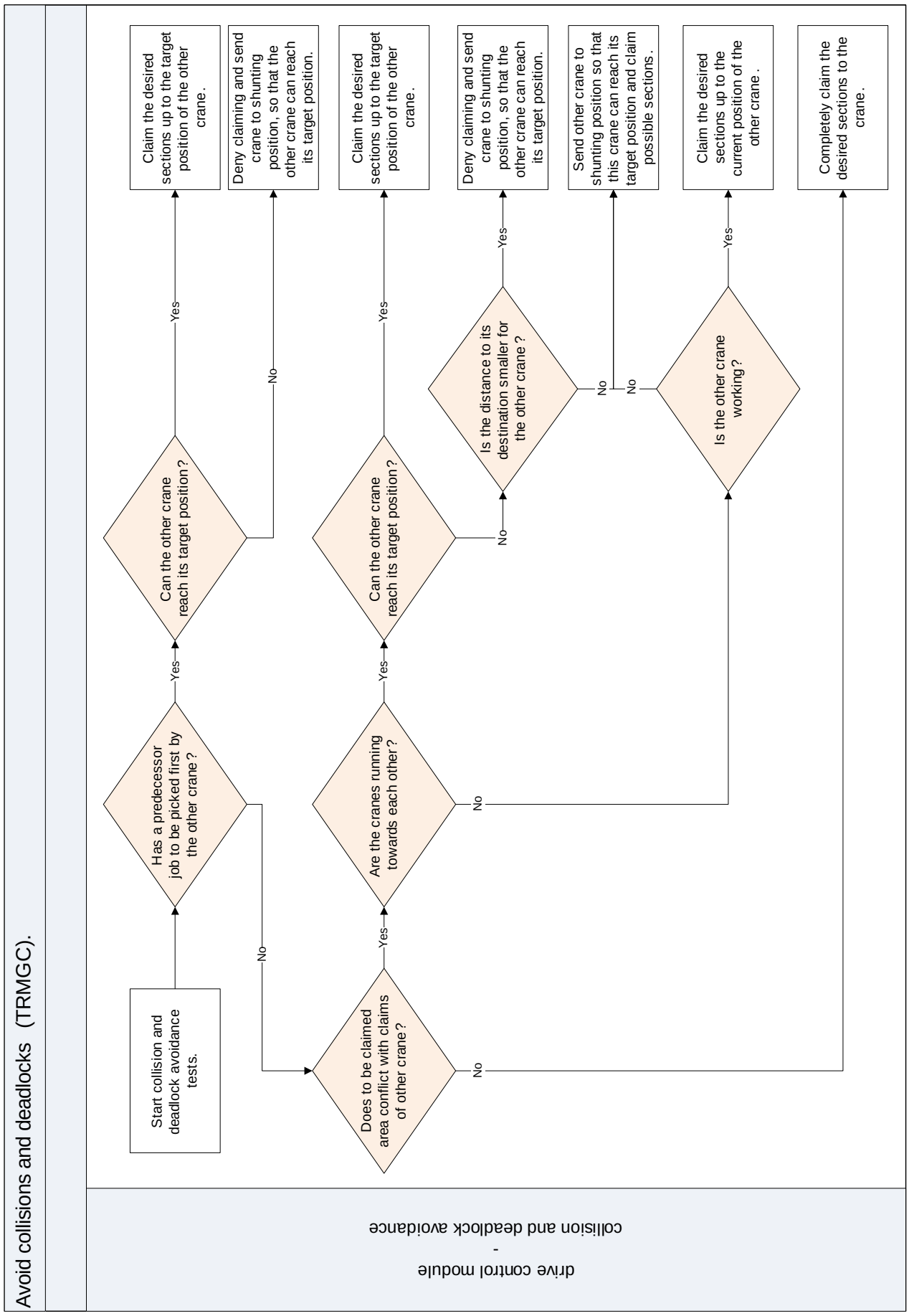

Figure 4.2: Process flow chart: 'Avoid collisions and deadlocks (TRMGC).'

up to the target position of the other crane. But if also the other crane cannot reach its target position, it has to be decided which of the cranes is moved to a shunting position. Here, the minimum remaining distance to the target position is chosen as criterion. But one can also think of other criteria like for example the minimum handover area due date of the cranes' jobs. However, if this distance is smaller for the other crane, claiming for the regarded crane is denied, instead of a shunting position is determined which allows the other crane to reach its target position. In case the distance is smaller for the regarded crane, a shunting position for the other crane is determined and the sections up to the current position of the other crane are claimed. The same applies for situations in which the cranes are not running towards each other and the other crane is not working. But when the other crane is currently executing a certain job, the desired sections are claimed up to the current position of the other crane.

Due to the crossing capability, a lot more flexibility is involved with the DRMGC system than with the TRMGC system, but at the same time — contrary to the TRMGC systems — collisions can occur for all crane driving directions within the DRMGC system. Thus, on the one hand the number of different collision situations increases, while on the other hand a collision is only possible if the trolley of the outer large crane is not located in the crossing position. A schematic representation of the collision and deadlock avoidance mechanism for the DRMGC system is given in figure 4.3. Due to the different characteristics of the outer large and inner small cranes, two separate processes are depicted. On a first eyes view, both processes seem to be similar and simpler than the TRMGC process. This is possibly true for the inner small crane, but not for the outer large crane which contains two process steps that are executed by the strategy submodule for crane routing decisions. These process steps — which greatly influence the outcomes of the collision and deadlock avoidance mechanism — are described in section 6.3. In summary, the extent the trolley of the outer large crane is not located in the crossing position and along with it the number of to be claimed sections for the outer large crane and the possibility for crane interferences depend on the applied crane routing strategy.

According to figure 4.3, the collision and deadlock avoidance process for both cranes begins with a check on predecessors. For the inner small crane the checks and outcomes concerning predecessors are the same like in the TRMGC system. The trolley of the outer large crane is moved into the crossing position — if not already there — in case a container has to be shuffled by the inner small crane before the outer large crane can pick its container. In case no predecessors have to be picked first, it is checked for claiming conflicts with the other crane. While the desired sections are completely claimed by the inner small crane if no conflict occurs, the desired sections are only claimed up to the conflict start position with the outer large crane in case of a conflict. For an outer large crane, the outcome of the conflict check is greatly dependent on the used crane routing strategy and the number of to be claimed sections. However, if no conflict is expected, the desired sections are completely granted to the outer large crane. Whereas in case of a conflict, the actions depend as well on the decision of the routing strategy.

The TriRMGC system contains parts of the TRMGC and DRMGC systems. Consequently, parts of both collision and deadlock avoidance mechanisms are used for the TriRMGC system. While the collision and deadlock avoidance between the two same-sized inner small cranes works like the TRMGC mechanism, the collision and deadlock avoidance between the outer large crane and the two inner small cranes works similar to the DRMGC mechanism. Like for the DRMGC

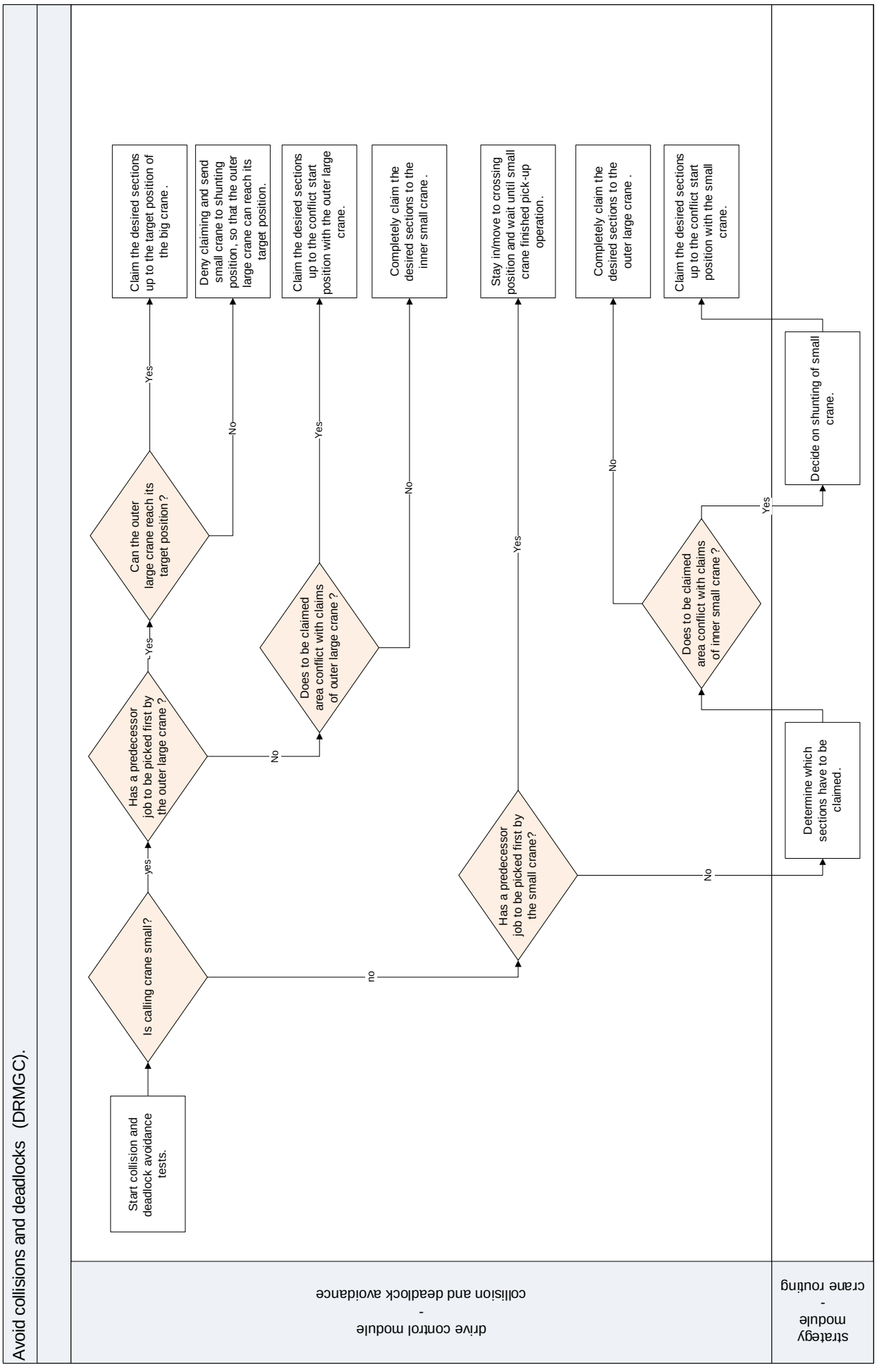

Figure 4.3: Process flow chart: 'Avoid collisions and deadlocks (DRMGC).'

system, the checks and outcomes for the outer large crane greatly depend on the used crane routing strategy.

Chapter 5

Administration Module

The administration module is probably the nerve centre of the simulation model, as it has interfaces to all other modules and is responsible for manifold functions: Firstly, information on arriving and departing containers — that are provided by the scenario creation module — are processed by the administration module. Secondly, there is interaction with the strategy module as the decision making process for container stacking and crane scheduling is initiated and the results are processed by the administration module. Thirdly, the drive control module is provided with driving coordinates of to be executed jobs by the administration module. Finally, parameters of the administration module are automatically controlled by the experimental control module and the statistics module obtains most information from the administration module.

According to its main functions, the administration module is divided into three submodules: The stacking capacities of the modelled yard block are managed by the storage management submodule. The job management submodule generates and manages transport jobs for the gantry cranes. The handover area management controls the occupancy of the handover area capacities.

5.1 Storage Management

The storage management submodule comprises all administrative functions concerning the stacking capacities of the simulated yard block and the storage positions of individual containers. While three processes are directly controlled by the storage management submodule, many processes of other modules work with the information that is provided by this submodule.

In detail, four information sources are managed by the storage management submodule. Firstly, the *slot occupancy table*, which has dimensions $n^{\mathrm{x}} \times (n^{\mathrm{y}} \cdot n^{\mathrm{z}})$, contains information on the current occupancy of each slot in the yard block as well as the ID of the corresponding container. In case container c is stacked in bay p_c^{x}, lane p_c^{y} and tier p_c^{z}, cell $(p_c^{\mathrm{x}}, p_c^{\mathrm{y}} + (p_c^{\mathrm{z}} - 1)\, n^{\mathrm{y}})$ of the slot occupancy table is filled with 'c'. The representative cell of an empty slot is filled with '0'. Secondly, the filling rate of the yard block is then computed by counting the cells in the occupancy table with entries $\neq 0$ and dividing this number by the number of theoretically available slots. Thirdly, the *stack occupancy table*, which has dimensions $n^{\mathrm{bay}} \times n^{\mathrm{lane}}$, contains information on the prospective occupancy of each stack in the modelled yard block. In contrast to the slot occupancy table, besides the currently stored containers also the slots are taken into account that are only virtually occupied (i.e. a container is already planned for storage

in the corresponding stack but has not yet arrived). Finally, the *relocation table*, which also has dimensions $n^x \times n^y$, contains information on the number of planned but not yet executed container pick-ups for each stack in the yard block. In order to avoid unnecessary shuffle moves, it is harmful to store new containers in a stack with planned pick-ups. Therefore, jobs with drop positions in stacks with planned pick-ups are avoided and/or postponed until all pick-ups of that stack are executed.

The first process that is controlled by the storage management submodule is 'planning admission into yard block'. The respective flow chart of this process is shown in figure 5.1. It is assumed that the storage management submodule is informed about the future arrival of container c at the modelled yard block a random triangular-distributed time in advance of the actual arrival time $a_c^{\text{container}}$ of container c, which means there is a stochastic look-ahead time for prospective container arrivals at the modelled yard block. Owing to the fact that the up- and downstream process at the waterside and landside block ends of the modelled yard block are to different degrees predictable, different probability distributions for the look-ahead times can be specified by the user for vehicle/container arrivals in the waterside and landside handover areas. Once the information on a prospective container arrival is received by the storage management submodule, two points are then immediately checked: Firstly, it is checked whether there is some capacity left or if the user-defined maximum filling rate is already reached. Secondly, the module checks for available storage capacity in stacks that have the required size for that container (i.e. 20' or 40'). In case there is no capacity available in the yard in general or in the required stacks, the container is rejected by the modelled yard block and redirected to another yard block. Otherwise, the handover lane management submodule (cf. section 5.3) is informed about the container arrival, which then tries to allocate a position in the handover area to the respective container arrival. If no position is available in the handover area, stacking of that container is also rejected. Otherwise, a storage location for this individual container is determined in real-time by the stacking submodule of the strategy module. Afterwards, for the corresponding stack of the planned storage location, the virtual stacking height is increased (i.e. the relevant cell(s) in the *stack occupancy table* is (are) increased by one). Finally, a transport job from the planned position in the handover area to the planned storage position is generated.

Another process that is controlled by this submodule is 'Planning relocation from yard block' which is illustrated in figure 5.2. The process starts with the announcement — according to a randomly drawn look-ahead time for a vehicle arrival at the yard block, just as explained before — of the prospective departure time $d_c^{\text{container}}$ of an individual container c. In case containers are stacked above the required one, shuffle moves have to be planned, which is done according to the process 'Planning shuffle move' that is illustrated in figure 5.3. However, this process continues with the determination of a concrete drop position in the relevant handover area (cf. section 5.3). Finally, a transport job from the current storage position to the chosen position in the handover area is generated.

The last process that is mainly controlled by the storage management submodule is 'Planning shuffle moves' which is illustrated in figure 5.3. As described before, the process is mostly initiated by the process 'Planning relocation from the yard block'. After receiving information on the container that has to be retrieved from that stack, a container is identified that has to be relocated to another stack in that block. Next it is checked, if there is capacity available in

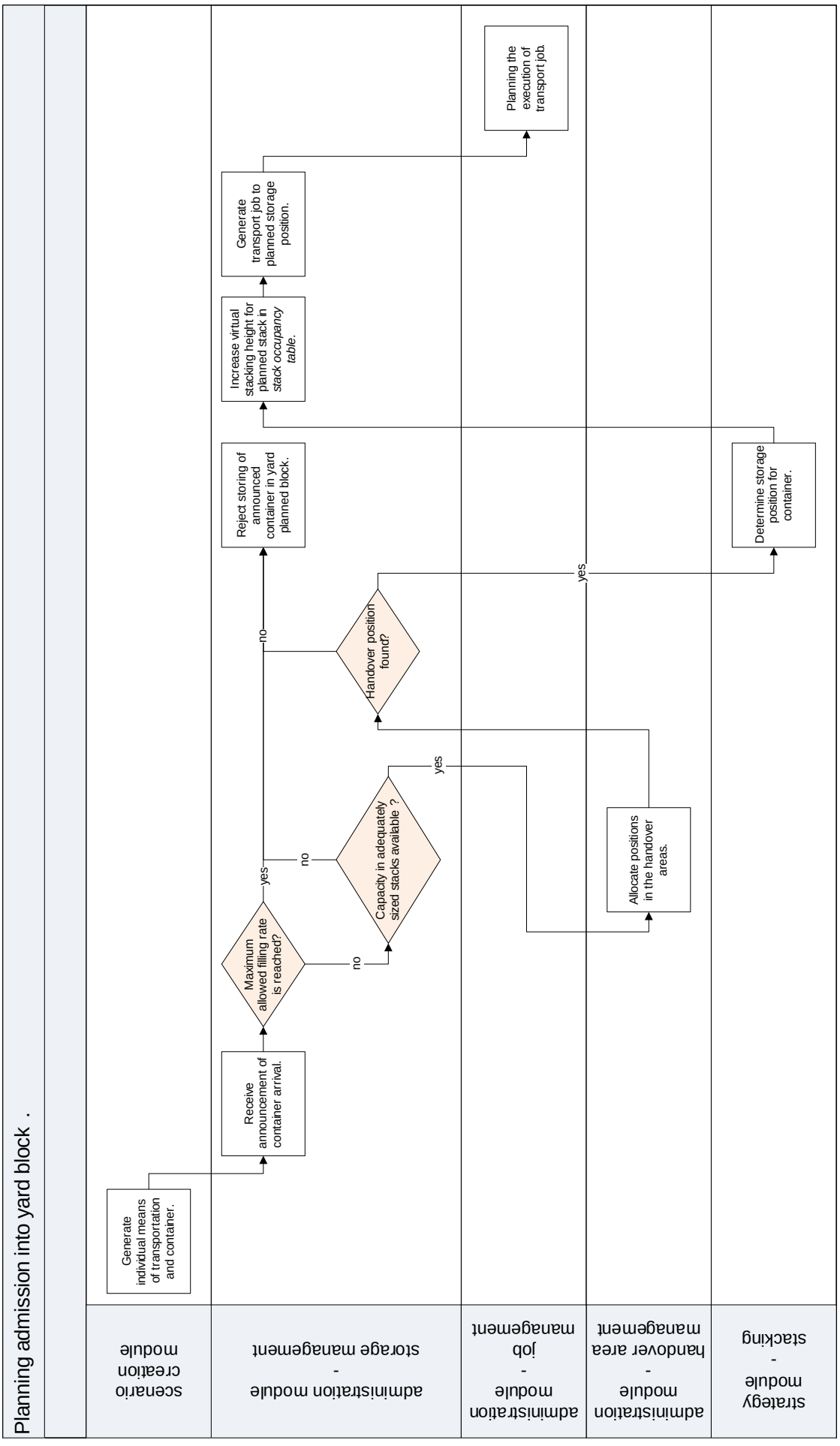

Figure 5.1: Process flow chart: 'Planning admission into yard block.'

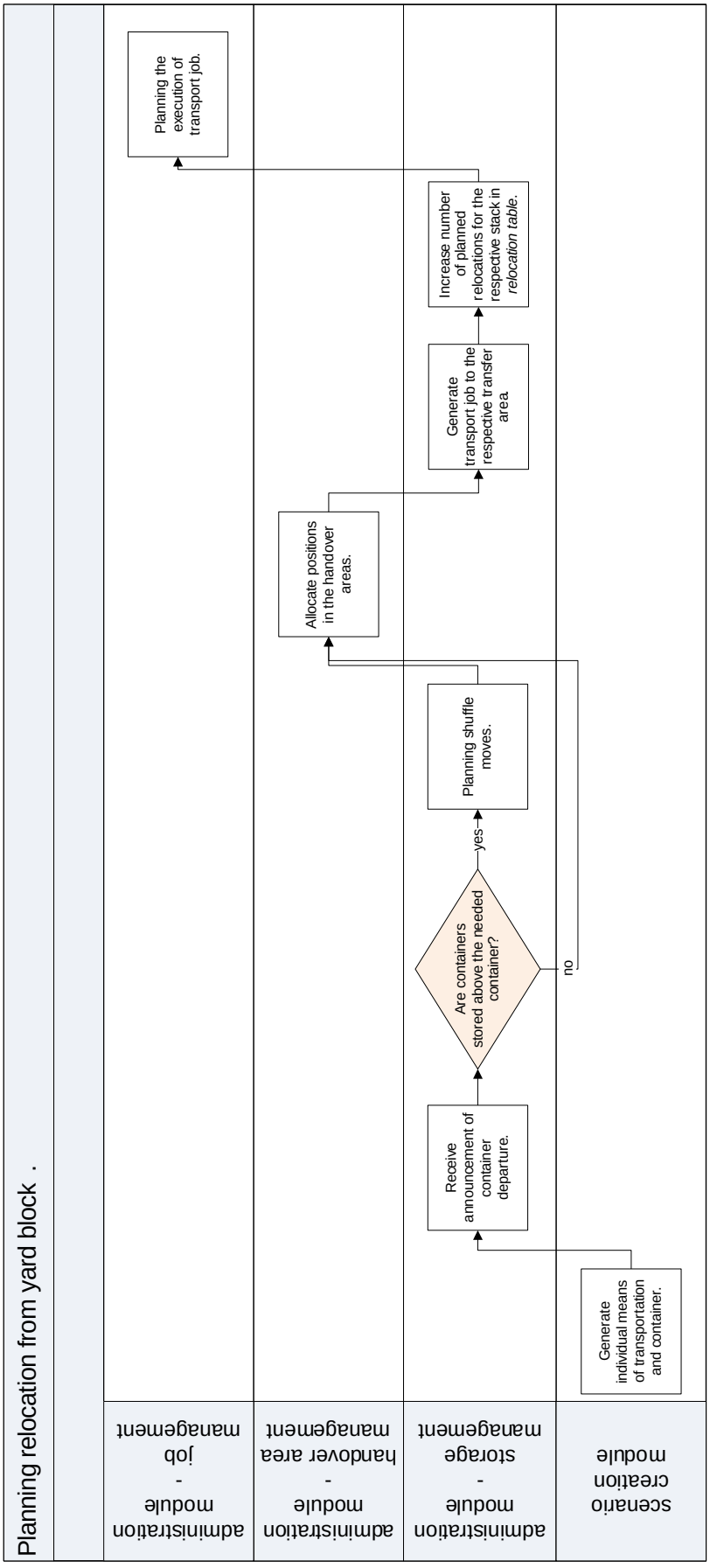

Figure 5.2: Process flow chart: 'Planning relocation from yard block.'

adequately-sized stacks of that block. In case no adequate stack is found, the identified container is planned to be retrieved at the waterside block end, in order to be transported by horizontal transport vehicles to another yard block with adequate stacks available. Therefore, a position in the waterside handover area is determined, a transport job to the waterside handover area is generated and the *relocation table* is modified accordingly. If, however, at least one adequate stack is available, a storage location for the to be shuffled container is determined and the corresponding transport job is created. Thereafter, for the current and planned stack of that container, the relevant cells of the *relocation table* and *stack occupancy table* are increased by one. This process is repeated until all containers on top of the to be retrieved one are either planned for a shuffle move or a waterside retrieval move.

All three processes that are described in this section are continued by the process 'Planning the execution of transport jobs' which is headed by the job management submodule and explained in the following section.

5.2 Job Management

The job management submodule manages all tasks and information connected with crane transport jobs: New jobs are processed, jobs are checked for feasibility and scheduling as well as the execution of transport jobs are initiated. In detail, two information sources are managed by this submodule and one process is controlled that is of central importance for the simulation model.

The information sources managed by this submodule are the *job information table* and the *joblist*. The *job information table* is probably the most important information source as it contains extensive data of all individual jobs that are already executed, that are currently executed and that have just been announced and are waiting for execution. Once available, the *job information table* comprises the following data of each job: Job ID, ID of the handled container, pick-up coordinates, drop coordinates, job type (wsin, wsout, lsin, lsout, wsshu, lsshu, wshk, lshk), due date for arrival at the handover area, realised arrival at the handover area, due date for container pick-up, realised container pick-up, job status (waiting, assigned, started, picked, finished, deleted), assignable crane(s) (according to the preselection method, see section 6.2), assigned crane, required predecessor job, realised crane waiting time due to crane's earliness, realised waiting time for other transport modes (XTs, AGVs, SCs) due to crane's lateness, total job execution time, crane empty driving time, crane interference time. The *joblist* contains the ID of all jobs with status 'waiting' (i.e. all jobs that have not yet been assigned to a crane). Each time a new job is created, it is recorded in the *job information table* and the *joblist*. After the status of a job changes from 'waiting' to 'assigned' or 'deleted' the job is deleted from the *joblist*.

The process that is controlled by the job management submodule is 'Planning the execution of a transport job'. The flow chart of this process is shown in figure 5.4. This process is always running during a simulation run as it continuously observes the *joblist* and the status of the crane(s). The execution of the process is then activated, each time a new job is recorded in the *joblist* or a crane becomes available. Thereafter, the first process step checks for required sequence relations of the jobs that are listed in the *joblist*. Sequence relations are generally required if a shuffle move has to be executed before another container is accessible. The job ID of the shuffle

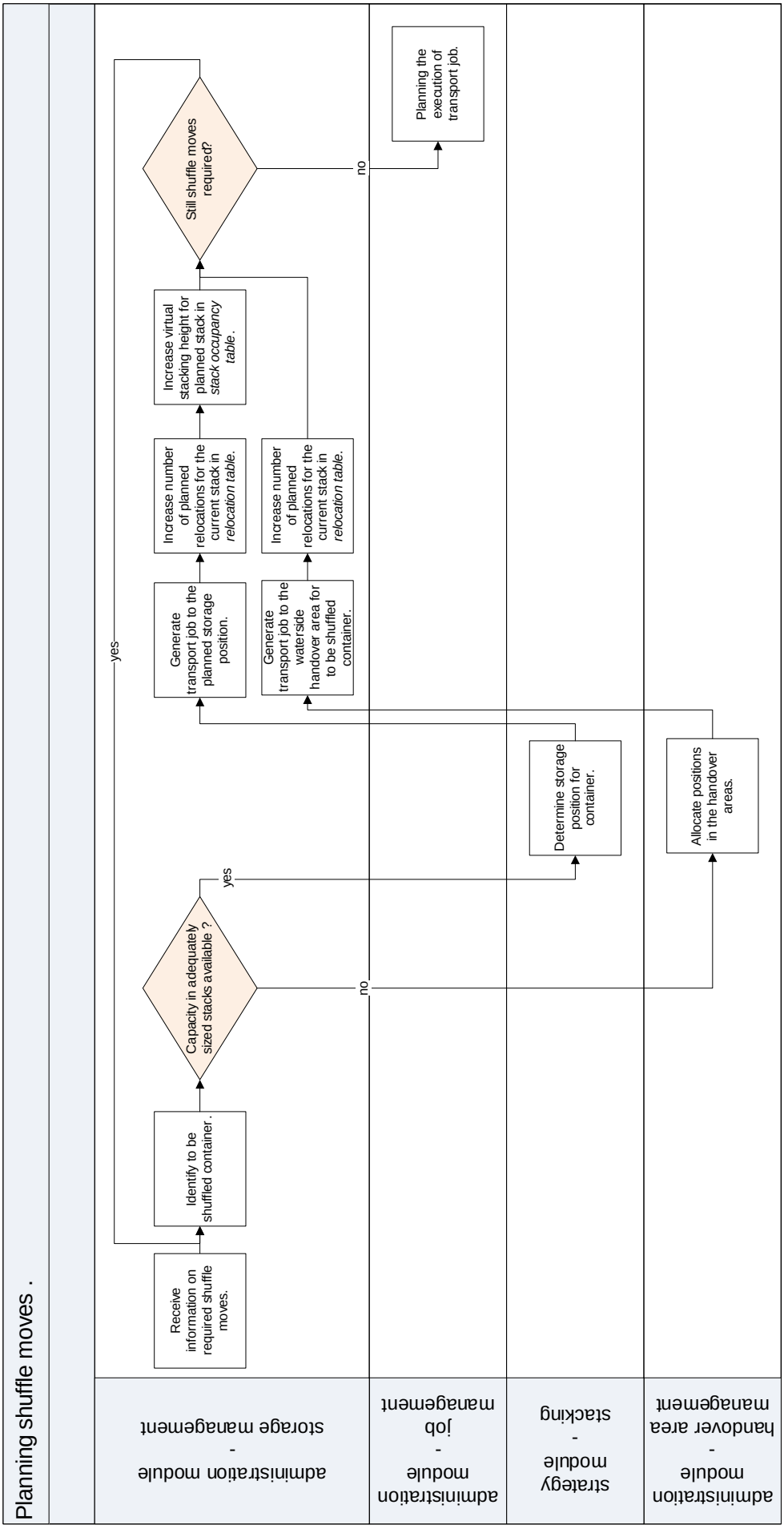

Figure 5.3: Process flow chart: 'Planning shuffle moves.'

move is recorded as predecessor of the job that is stored directly below the shuffle container. In the next step, the assignable crane is determined for each job in the *joblist* according to the chosen preselection method (see section 6.2). Based on the type and driving coordinates of a job, it is determined which crane(s) are allowed to execute the job — of course, for SRMGC the decision is very simple and therefore this process step can be skipped. Afterwards, it is checked if valid assignments of jobs to cranes are at all possible. The process only continues to the next step if at least one crane is available for which at least one job is determined to be assignable. Otherwise, the process is reset to the beginning and the status of the cranes and the *joblist* are again observed for changes. However, in case valid assignments of jobs to cranes are available, depending on the selected crane scheduling strategy (see section 6.2), one or even more jobs are assigned to the cranes and sequenced. After that, as a precaution, the job that is scheduled next for execution is checked for feasibility. Within the framework of the feasibility check, the corresponding container of this job is tested for availability, storability and retrievability, which means it is tested if the relevant container is still stored at the pick-up coordinates of the job, if there is still enough capacity at the drop coordinates and if the container can be picked without preceding shuffle moves, respectively. If required, the job is either postponed, deleted or changed. In case the job is deleted, the process is reset to the determination of required precedence relations and it is tried to plan another job for execution. Otherwise, the execution of the job is started by the drive control module and the process is also reset to the determination of required precedence relations.

5.3 Handover Area Management

The handover areas do not really belong to the yard block system, moreover they are interfaces to the waterside and landside horizontal transport systems. Hence, the handover areas can partly be ascribed to the container storage yard subsystem and the horizontal transport subsystem of seaport container terminal systems. However, as the primary focus of this simulation model is the container storage yard subsystem and the horizontal transport subsystem is not explicitly modelled, it is decided to model the handover areas only in a simplified way.

Several parametrisations are possible for the waterside and landside handover areas. In fact, for both handover areas, the user has to specify the capacities and the probability distributions for vehicle residence times in the handover area, for look-ahead times of vehicle arrivals in the handover areas and for spreader fine positioning times in the handover areas. Firstly, the capacity of the landside handover area represents the number of XTs that can be simultaneously located in and/or assigned to the handover area of a yard block, while the meaning of the waterside handover area capacity depends on the underlying horizontal transport system. In case AGVs are used as waterside horizontal transport vehicle, the meaning is similar as for the landside capacity. But for SCs as horizontal transport vehicle, the capacity represents the number of containers that can be simultaneously placed in and/or assigned to the waterside handover area of a yard block.

Secondly, the residence time of a transport vehicle in the handover area is the period of time a vehicle stays in the handover area after the loading or unloading operation of that vehicle is finished. Of course, the occupied position in the handover is blocked for other vehicles until the

strategy module - crane deployment
administration module - job management
administration module – storage management
drive control module
Determine job assignment and sequence.
yes
Is valid crane - job assignment available ?
no
Availability check for valid crane - job combination.
Determine assignable crane for each job in *joblist* according to preselection method.
Check for required precedence relations of jobs.
Wait for changes in joblist and crane status.
Planning relocation from yard block.
Planning admission into yard block.
Planning shuffle moves.
Execute transport job (change in crane status to „available“).

relevant vehicle has left the handover area. Here, it is assumed that the residence time is an exponentially-distributed random variable. Usually, the expected residence time is longer for the landside than for the waterside handover area, as XTs are manually controlled and the driver has to leave his cab during the handover operation, while waterside vehicles might be automated and a driver would not be required to leave his cab during handover. Thirdly, the look-ahead time of vehicle arrivals defines the period of time the arrival of a vehicle in the handover area is certainly known in advance of its actual arrival. The longer the look-ahead time for an arrival, the better the corresponding transport job can be scheduled according to a user-defined objective. Since waterside horizontal transport machines are controlled by the terminal, longer look-ahead times may be expected for the waterside arrivals than for XT arrivals at the landside. Here, it is assumed that the look-ahead times of an individual arrival is a random triangular-distributed variable. Finally, the time required for spreader fine positioning in both handover areas has to be specified (see also 4.1). Here, it is assumed that the required spreader fine positioning time for an individual container pick-up or delivery is a random gamma-distributed variable.

The only information sources that are managed by this submodule are the *handover occupancy tables* of the waterside and landside handover areas. The cells of these tables represent the positions in the handover areas and are either filled with 0 for an unoccupied position or the ID of an container that is planned to be picked-up or dropped at the corresponding position. Thus, the number of zeros in the handover occupancy tables indicate the number of available positions in both areas. Here, it is assumed that a handover position is occupied with a certain container along with the creation of the respective job. In addition, it is assumed that all positions in the handover areas are directly accessible for the cranes, which means containers are positioned on the ground or on chassis but never stacked one upon the other.

The containers are allocated to a certain handover position within the process 'Allocate positions in the handover area' which is illustrated in figure 5.5. This process is embedded in the processes 'Planning admission into yard block', 'Planning relocation from yard block' and 'Planning shuffle moves' (see figures 5.1, 5.2 and 5.3). After the basic information on a new job has been received by the handover area management submodule, it is checked whether it is a storage (job types: wsin, lsin) or a retrieval job (job types: wsout, lsout). In both cases, it is then checked for available capacity. If it is a storage job and there is no position available in the handover area, this information is send to the storage management submodule which then rejects the storage of the relevant container. If there are some unoccupied positions in the relevant handover area, for both retrieval and storage jobs, a position is randomly drawn and the allocated position is send to the storage management submodule. In case of a retrieval job and no available positions in the handover area, the system waits until a position becomes available. Thus, a retrieval job is not created until a position in the handover area has been found. Altogether, the maximum number of main jobs (i.e. storage and retrieval job) that are simultaneously plannable is specified by the user-defined joint capacity of the waterside and landside handover areas.

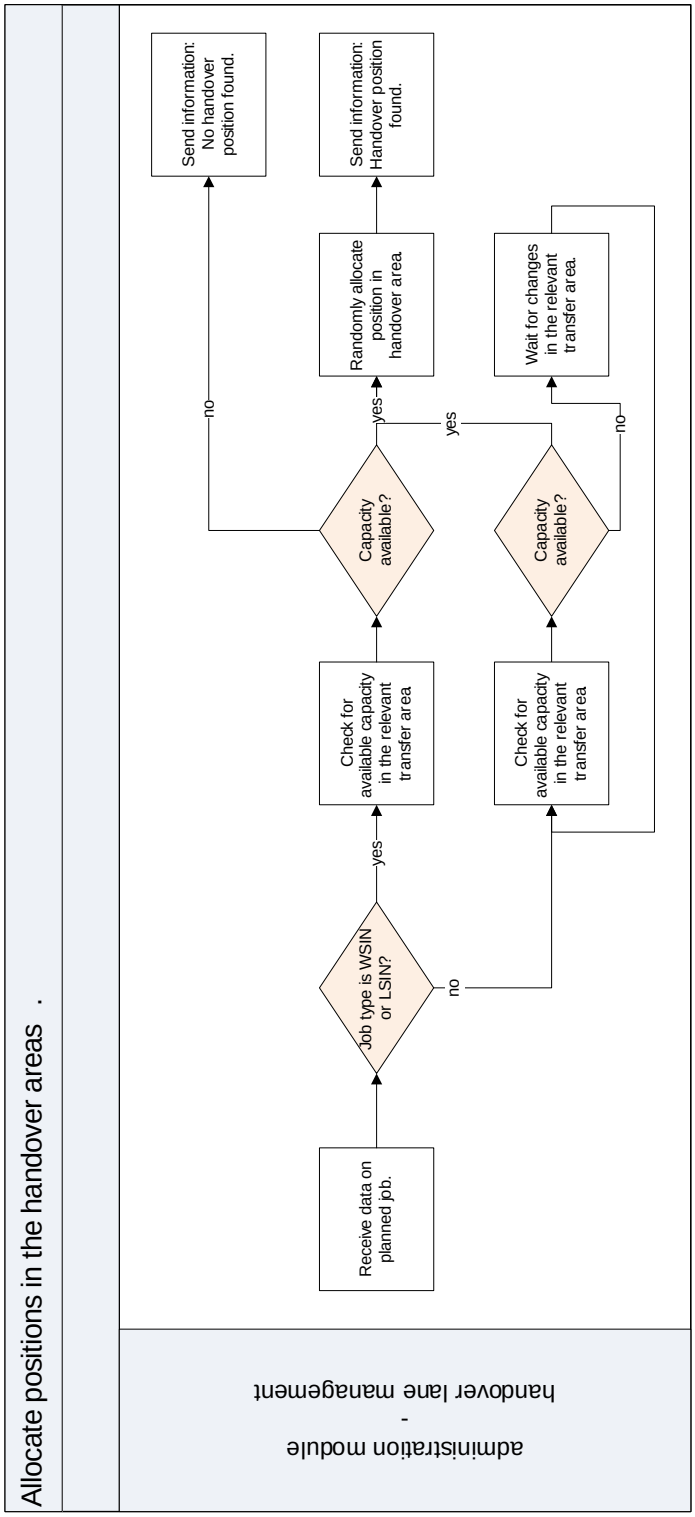

Figure 5.5: Process flow chart: 'Allocate positions in the handover areas.'

Chapter 6

Strategy Module

Within the framework of the strategy module, all operational planning decisions are made, which include decisions on the storage positions of individual containers, the crane assignment and sequencing of transport jobs as well as the routing of the cranes. Therefore, three submodules are implemented: The stacking submodule, the crane scheduling submodule and the crane routing submodule. All these submodules are subsequently explained, but the details and the ideas of the different strategies are not described here — instead of, the general technical implementation of these strategies is briefly introduced. A detailed description of all implemented operational strategies is given in chapter 5 of Kemme (2011).

6.1 Stacking

All decisions on storage positions for individual containers are made within the stacking submodule. Thus, it is frequently called by the storage management submodule (see section 5.1). In detail, the stacking submodule executes the process step 'Determine storage position for container' which is embedded in the processes 'Planning admission into yard block' and 'Planning shuffle moves'. Within this process step, a feasible storage position is determined for a container that is transmitted by the administration module. The storage position is chosen out of the set of all adequately sized stacks with remaining capacity which is previously determined by the storage management submodule. Which allowed storage position is actually selected depends on the applied container stacking strategy. Basically, every possible stacking strategy can be applied, but initially only a random stacking strategy (RaS) and a greatly parametrisable combined cost function stacking strategy (CCFS), which is based on the ideas of the category, the retrieval time and the positional stacking concepts, are implemented. The main idea and the functionality of both strategies are described in great detail in section 5.2 of Kemme, 2011. Considering the fact that the performance of the stacking strategies in terms of the number of shuffle moves does not only depend on the applied strategy itself, but also on the exchangeability of containers during the vessel loading process, the user is able to decide on whether online stowage planning is applied or not (see section 3.1.7).

6.2 Crane Scheduling

The crane assignment of transport jobs as well as their execution sequence is determined by the crane scheduling submodule. In figure 5.4 it is shown that this submodule is called two times by the job management submodule during the process 'Planning the execution of a transport job'. Firstly, the crane scheduling submodule is responsible for the process step 'Determine assignable crane for each job in *joblist* according to preselection method'. Here, for each job that is waiting for execution and that has not yet been assigned to a crane (i.e. having status 'waiting'), this submodule determines on basis of user-defined rules which crane(s) is/are allowed to execute which job. These rules are based on the type and the driving coordinates of a job. A set of rules, that defines the assignable crane for each job, is called preselection method. For each job, the results of the preselection method — which are the assignable cranes — are recorded in the *job information table*. Of course, for the SRMGC crane the one and only crane is allowed to perform each job, but for multi-crane systems a crane may be restricted to perform a currently waiting job for technical reasons (i.e. not possible to access the relevant handover area) and/or performance reasons (i.e. in order to reduce interferences between the cranes). Detailed considerations on the purpose of preselection methods and different design variants of preselection methods are presented in section 5.3 of Kemme (2011).

Secondly, also the process step 'Determine job assignment and sequence' is executed by the crane scheduling submodule. Here, based on the assignable crane(s) for each job in the *joblist*, a job is assigned to a currently idle crane of the modelled yard block. Which crane-job assignment is actually realised depends on the applied combination of online policy and solution method for the crane scheduling problem. In section 5.3 of Kemme (2011) four online policies and seven solution methods are introduced that are all implemented in the RMGC simulation model and can be combined in different ways. In case the greedy or FIFO online policy is applied, only the next job for an idle crane is determined, whereas the crane assignments are based on sequences of several jobs for all cranes of a yard block if the replan or ignore online policy is applied (Grötschel et al., 2001). The FIFO online policy simply assigns the jobs in the order of appearance, while the greedy online policy evaluates all allowed (by the preselection method) crane-job assignments with respect to a user-defined scheduling criterion or cost function and selects the assignment with the lowest costs for realisation. By applying the replan online policy, new sequences of jobs are created with the used solution method each time the process step 'Determine job assignment and sequence' is called by the job management module. All previously generated and only partly realised sequences of jobs for the cranes are rejected. In contrast, by applying the ignore online policy, new sequences of jobs are only determined if a job sequence for one of the cranes has already been completely realised. As long as the previously generated job sequence of the currently idle crane has not been completely realised, no new schedule is created for all cranes and the next not yet performed job of the previously planned job sequence for the idle crane is assigned for realisation.

6.3 Crane Routing

The crane routing submodule of the simulation model is closely linked with the collision and deadlock avoidance submodule, as it comprises all decision making processes on crane routing

in terms of crane crossing manoeuvres and granting the right of way in conflicting situations. While all conflicts between the inner small cranes are simply resolved as described in section 4.2, exchangeable routing procedures for the DRMGC and TriRMGC systems — in particular for the outer large cranes — are implemented in the simulation model. Therefore, the crane routing submodule is called twice during the process 'Avoid collisions and deadlocks (DRMGC)' (see figure 4.3). If this process is executed for the outer large crane, the process steps 'Determine which sections have to be claimed' and 'Decide on shunting of inner small crane' are performed by the crane routing submodule.

The first process step is related to the question, to which extent is the movement of the portal of the outer large crane allowed if its trolley is not located in the crossing position. In contrast to the collision and deadlock avoidance mechanism that is presented in Valkengoed (2004), here it is assumed that the trolley of the outer large crane is by default located in the crossing position during portal movements, if not differently specified by the applied crane routing strategy. Consequently, within the simplest crane routing strategy, the trolley is only allowed to leave its crossing position if the portal has reached its target position, and before the portal starts driving again, the trolley has to be moved back to the crossing position — independently of the distance(s) to the inner small cranes. Therefore, only three sections need to be claimed for each trolley movement. However, other strategies allow portal movements without the trolley being located in the crossing position, but then some additional sections have to be claimed. Here, the sections that have to be claimed are determined with respect to the selected routing strategy and the expected driving times for the trolley and the portal of the outer large crane.

The second process step is only relevant in case sections have to be claimed due to portal movements without the trolley being located in the crossing position and a conflict with the inner small crane is detected by the collision and deadlock avoidance submodule. Then it is determined on basis of the selected routing strategy as well as the status of the cranes and their driving coordinates which crane is granted the right of way. The outcomes may either be that the outer large crane cannot start its planned move (i.e. the inner small crane is given the right of way) or the inner small crane is stopped and/or moved to a shunting position (i.e. the outer large crane is given the right of way). In section 5.4 of Kemme (2011) several design variants of crane crossing processes and other enhancing features for routing of RMGC multi-crane systems with crossing capabilities are introduced, which are all implemented in this simulation model.

Chapter 7

Experimental Control Module

The experimental control module may not be regarded as an immanent part of the simulation model since it is not responsible for any function of the real-world yard block system. Moreover, the purpose of the experimental control module are reliefs and time savings for conducting simulation experiments. A simulation experiment is defined as a certain number of simulation runs with different seed-initialised data realisations but with the same parameter settings for each of these runs. The experimental control module facilitates automatic changes of data realisations and parameter settings according to a user-defined pattern. This pattern is specified by the number of different data realisations for each simulation experiment, the to be modified parameters, the extent of modification for each parameter and the order in which parameters are modified. Since a major purpose of this simulation model is the investigation of layout changes, a predefined pattern concerning changes of the numbers of bays, lanes and tiers is already included in the model. In addition, methods and tables can be specified by the user that enable each pattern of parameter changes.

The results of each experiment and each single simulation run are automatically recorded and exported into Excel files. For each experiment, a new Excel file is created that contains the results of each simulation run for that experiment and another file is created after all user-defined experiments have been conducted which contains the average results of the simulation runs for each experiment. Both types of Excel files are similarly structured. The results of the simulation runs and experiments are shown in rows and each row contains in columns the values of these figures for the corresponding simulation run or experiment.

Chapter 8

Statistics Module

Like the experimental control module, also the statistics module is not an intrinsic part of the simulation model. It is responsible for data collection and automated computing of meaningful figures which allow for a differentiated evaluation and interpretation of the conducted experiments. The figures of each simulation run are based on the data collection from the end of the warm-up period T^s to the end of the simulation horizon T. The data for some figures is collected in special tables while other figures are computed on basis of administrative data components like the *job information table* or the *slot occupancy table*. Most figures are separately computed for each crane and for each job type. Consequently, depending on the applied crane system, several dozens of statistical figures are available for each simulation run. In fact, 60, 173, 173 and 228 figures are computed for each simulation run and/or experiment with the SRMGC, TRMGC, DRMGC and TriRMGC system, respectively. These figures include:

- the number of jobs performed by all cranes and by each individual crane,
- the fraction of all job types performed by all cranes and by each individual crane,
- the final filling rate of the yard block,
- the average filling rate of the yard block,
- the number of redirected containers to other yard blocks,
- the number of relocated containers to other yard blocks,
- the average realised container dwell time,
- the operating crane hours for all cranes and each individual crane,
- the total productivity (jobs/h) for all cranes and each individual crane, in terms of performed jobs over the statistically relevant simulation horizon ($T - T^s$),
- the operative productivity (jobs/h) for all cranes and each individual crane, in terms of performed jobs over the operating crane hours,
- the productive productivity (jobs/h) for all cranes and each individual crane, in terms of performed main jobs over the operating crane hours,
- the maximum total productivity (jobs/h) for all cranes and each individual crane,

- the maximum productive productivity (jobs/h) for all cranes and each individual crane,
- the mean execution time for jobs performed by all cranes and each individual crane,
- the mean EDT (empty driving time) for jobs performed by all cranes and each individual crane,
- the mean crane interference time during the execution of jobs performed by all cranes and each individual crane,
- the number of crane waiting times due to early arrivals in the handover areas for all cranes, all types of jobs, each individual crane and each job type,
- the mean crane waiting time due to early arrivals in the handover areas for all cranes, all types of jobs, each individual crane and each job type,
- the standard deviation of the mean crane waiting time due to early arrivals in the handover areas for all cranes, all types of jobs, each individual crane and each job type,
- the maximum crane waiting time due to early arrivals in the handover areas for all cranes, all types of jobs, each individual crane and each job type,
- the number of vehicle waiting times due to late crane arrivals in the handover areas for all cranes, all types of jobs, each individual crane and each job type,
- the mean vehicle waiting times due to late crane arrivals in the handover areas for all cranes, all types of jobs, each individual crane and each job type,
- the standard deviation of the mean vehicle waiting times due to late crane arrivals in the handover areas for all cranes, all types of jobs, each individual crane and each job type as well as
- the maximum vehicle waiting times due to late crane arrivals in the handover areas for all cranes, all types of jobs, each individual crane and each job type.

In case of a single simulation run, the resulting figures are displayed on the console and exported to an Excel file. But, the results of each simulation run are not displayed and exported, if several runs and experiments are conducted automatically by usage of the experimental control module. Moreover, the results of different simulation runs for each experiment are collected and exported to a joint Excel file and average values of each experiment are exported to another Excel file (see chapter 7). Finally, the temporal development of some important figures (e.g. average filling rate of the yard block, the mean EDT, the mean vehicle waiting time) can be displayed in charts that dynamically change with the progress in simulation time.

Chapter 9

Summarising Evaluation

In order to give an impression and to allow for an evaluation on the quality and the practical use of the RMGC simulation model, that is comprehensively described with all its modules in the preceding chapters, some additional information on the main features and limitations of the model as well as its validity are provided in this chapter. It is started with a summarising overview on the extensive parametrisation possibilities and the data and animation outputs of the model, which is followed by a discussion on the model's main features and limitations compared to other simulation models in the field of seaport container terminals. Finally, the validation process of this simulation model is explained, in order to create trust in the reliability and the practical validity of the model outputs.

9.1 Model Inputs and Outputs

Within the previous sections, several inputs and outputs of the simulation model are mentioned and described in the context of their associated modules. A summary of all these inputs and outputs is provided in this section, in order to give the user an overview of the huge parametrisation possibilities and the extensive outputs which allow for a reliable and meaningful interpretation of the simulation experiments. Firstly, the inputs of the simulation model are summarised and thereafter an overview of the different types of model outputs is given.

The model inputs consist of parameters that specify the scenario creation, the technical crane data, the yard block dimensions, the handover areas, the automatic experiment execution and the yard operations. Exemplary screen-shots of the menu windows for specifying the corresponding input parameters are shown in A.1. The inputs for the scenario creation comprise all parameters that define the distribution of container arrivals and container attributes. A detailed description of these parameters is provided in section 3.1.1. The acceleration, deceleration and velocity of portal, trolley and spreader can be individually specified by the user for each loaded and unloaded crane of the yard block (see section 4.1). The yard block dimensions are specified by parameters on the number of bays, lanes and tiers as well as on the length, width and height of a single storage slot. For both handover areas, the capacities can be defined by the user and distributions for the fine positioning of the spreader, the look-ahead times for vehicle arrivals and the residence time of vehicles in the handover area can be parametrised. In addition, it can be selected whether the waterside transport vehicles are either AGVs or SCs. For the automatic execution of experiments, the to be changed parameters and the corresponding change pattern

as well as the number of different data realisations for the experiments have to be specified by the user. Finally, dozens of parameters for the yard operations have to be specified by the user. Most of theses parameters are connected with the stacking and crane scheduling strategies, which may be individually parametrised for each crane and for each job type. Besides these strategy parameters, the user also has to decide on the usage of housekeeping moves, the application of online stowage planning, the settings of the crane routing strategy and various other parameter settings. Altogether, more than 200 inputs may be specified for a single simulation run.

The outputs of the simulation model are twofold. Firstly, several figures are recorded and computed for each simulation run. Most of theses figures are distinguished according to the relevant crane and job type. A detailed listing and description of the available statistical output is provided in chapter 8. Secondly, during each simulation run, all crane movements can be graphically displayed in a two dimensional model in any desired simulation speed. Thus, the driving behaviour of the cranes can easily be observed, which simplifies tracing of errors and the interpretation of performance figures. Some screenshots on the animation of crane movements in this RMGC simulation model are shown in appendix A.2.

9.2 Features and Limitations

So far, an extensive description of an elaborated yard block simulation model is given, which contains several noteworthy features but as well some limitations. Within this section, an overview of the most important features and limitations of the presented simulation model is provided. The features and limitations are discussed and compared to other simulation models in order to allow for a profound evaluation of the presented simulation model by the user.

Starting with the model limitations — the most important one is that QCs and SCs/AGVs are not explicitly modelled. It is assumed that the related processes are deterministic and that no waiting times for the RMGCs are induced due to late arrivals of SCs/AGVs at the yard block (see section 3.2). In addition, only one yard block is modelled. Therefore, the interdependencies between processes of the whole storage yard, the SCs/AGVs and QCs are neglected and the GCR (gross crane rate) cannot be used as a performance indicator. However, in Petering et al. (2009), it is shown that the waiting times of SCs/AGVs at the waterside handover areas have significant effects for the GCR. Consequently, the effects of the yard block for the performance of the whole terminal system can be reliably evaluated without detailed modelling of the whole terminal system.

In addition, several assumptions (not necessarily real limitations) have to be made for the scenario creation. Firstly, trains and land-land movements are ignored by the scenario creation module. The former assumption can be made without loss of generality since the transport between a rail yard and the landside handover area of the yard block would be performed by TTUs (truck trailer unit). Hence, the processes are similar to that for XTs, only the look-ahead time for arrivals of TTUs may be longer since the transport is controlled by the terminal. Also, no noteworthy limitation is involved with the latter assumption, since containers arriving and departing by XTs are usually not desired by the terminals and only make up for very small fractions of the overall cargo volume. Secondly, some simplifying assumptions on the relations of import, export and transshipment containers are made for the creation of container flows between

all considered modes of transportation in order to avoid the need for explicit definitions of all these container flows (see section 3.1.2). Thirdly, 24/7 operations are assumed for all terminal processes. Concerning the waterside processes, this may be a reasonable assumption for most terminals, but for the landside processes, the relevant legislative provisions of the country where the terminal is located have to be regarded (e.g. truck driving ban on weekends).

Finally, only 20' and 40' standard dry containers are created. Containers of other sizes (45' long, high cubes, foldable, etc.) and boxes for special goods (refrigerated goods, liquids, dangerous goods, etc.), which are accountable for about 15% of the throughput of a container terminal (Petering et al., 2009), are neglected. In addition, it is assumed that only a single container is either delivered or collected by each arriving vehicle in the waterside and landside handover areas. Technical errors and machine breakdowns are ignored for the RMGCs.

The most remarkable features of the simulation model are the previously unavailable parametrisation options (see section 9.1) and the detailed reproduction of the highly dynamic, stochastic, real-time environment of an RMGC storage yard system at a multiple-berth container terminal (see chapters 4 and 5). Most other simulation models that are described in the relevant literature name far less parametrisation options and make rather simplifying assumptions on the driving behaviour of the gantry cranes (cf. e.g. Petering et al. 2009, Dekker et al. 2006, Choe et al. 2007). In this respect, the realistic crane movements — including load-dependent acceleration and deceleration of portal, trolley and spreader — along with the sophisticated collision and deadlock avoidance mechanisms can be regarded as unique characteristics compared to other simulation models.

In spite of the various assumptions that are necessary for the implemented data generator, it offers some noteworthy features in comparison to the generators that are applied within other simulation studies. The data generator produces individual means of transportation and containers and assigns each container to a mean of transportation for delivery and pick-up in such a way that the predefined distributions on transport modes, transport mode sizes, transport mode arrival times and dwell times are matched simultaneously. Furthermore, pure transshipment terminals, import-export terminals and hybrid terminals can be simulated, since the programme provides data generation for variable fractions of transshipment containers. Unlike other known data generators, the presented one also considers the user-defined length of the quay wall and the berthing times of the vessels for the random generation of the vessel arrivals (e.g. Hartmann et al. 2007; Voogd et al. 1999). According to Petering et al. (2009), the major limitations of most simulation models on container terminals are the consideration of only one vessel at the same time and rather short simulation horizons, often of only one day. Here, containers and vessels are produced by the presented data generator for a user-defined simulation horizon in such a way that multiple vessels may be served simultaneously.

The simulation model contains several parametrisable stochastic components. Besides the data generator, which produces random container arrivals and attributes, also the look-ahead times for individual XT and SC/AGV arrivals, the residence time of XTs and AGVs/SCs in the handover areas and the time required for fine positioning and twist-lock-handling of the spreader at the pick-up and drop locations are user-defined stochastic components. Another noteworthy model feature is the recording and computation of several dozens of figures, which allow for a meaningful and differentiated interpretation of the simulation experiments (see chapter 8). Fi-

nally, all crane movements can be graphically observed in any desired simulation speed, which facilitates the understanding of the crane driving behaviour and the interpretation of the simulation results (see section 9.1). In summary, the presented simulation model is based on several assumptions, but most of them are reasonable and cannot be regarded as model limitations. Moreover, it is shown that the model contains several unique characteristics in comparison to known simulation models.

9.3 Validation and Verification

In order to use the results of a simulation model for supporting real-world planning decisions on the design and the operation of RMGC systems at seaport container terminals, it is of utmost importance that the model is without errors and valid for that purpose. This is in particular true for the simulation model presented here, as it is developed from scratch, without any reused simulation modules that have already been verified and/or validated before.

In general, a simulation model can be validated by applying statistical and expert validation to the graphical and numerical outputs (i.e. animation and figures) of that model (Kleijnen, 1999). In the context of automated container terminals, the usage of both statistical and expert validation is recommended by Saanen (2004). However, the new simulation model developed for the purpose of this work is not designed to represent actually existing container terminals, but to support the planning of new constructions of RMGC systems. Therefore, it would be improper for validating this model to only compare the simulation results of a certain real-world model configuration with the actual performance figures of the corresponding real system. Instead of, the developed model is iteratively validated by a multi-stage expert analysis of the graphical and numerical model outputs. The expert analysis is mainly based on the authors professional experience on design planning and operation of international seaport container terminals. In addition, several discussions with operational terminal staff, terminal managers and professional terminal simulation people are used for the expert validation of this model.

The multi-stage expert validation of the developed simulation model is organised in several iteratively repeated steps. It is started with an inspection of the numerical simulation results of pilot runs with different model configurations. Whenever relatively extreme performance measurements are found (based on the expert evaluation), the relevant model configuration is rerun and the animation is carefully inspected for errors and/or inaccuracies of the modelled and/or implemented RMGC operations until the cause(s) for the unexpected measurements are identified. If necessary, the model is revised, the pilot runs are repeated and the expert validation of these runs is started again from the beginning. This validation process is repeated until no longer any unimagined performance figures, errors and/or operational inaccuracies are found. Finally, the model resulting from the expert validation process is validated against both real-world performance figures of existing RMGC systems and simulation results of other simulation models on RMGC systems. It is found, that the developed model is able to reproduce the results of these systems with only negligible differences of the compared figures (i.e. only very few seconds for the mean vehicle waiting time in the handover area).

Appendix

A.1 Screen-Shots of Menu Windows

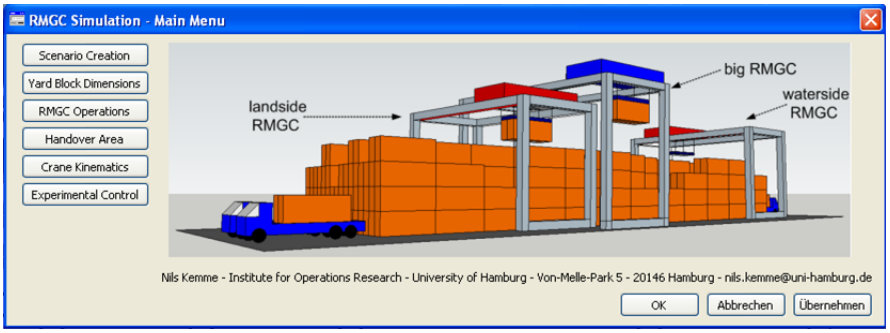

Figure A.1: Screenshot of RMGC simulation model — main menu (TriRMGC).

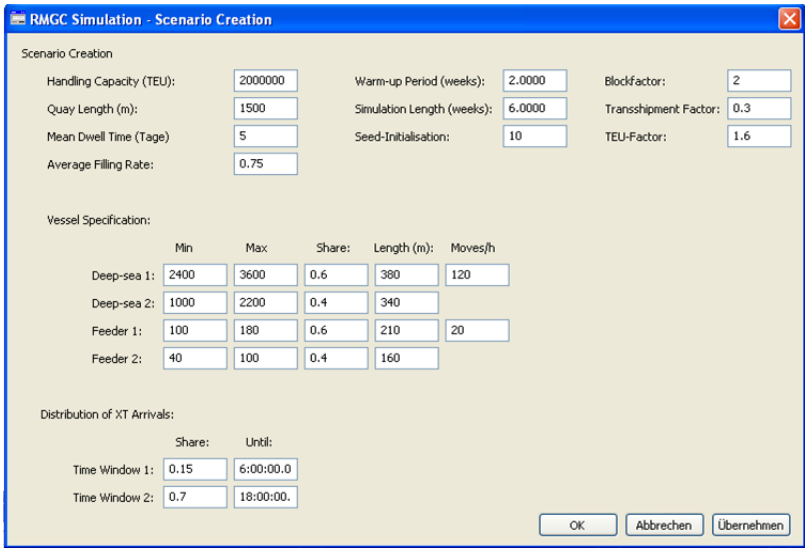

Figure A.2: Screenshot of RMGC simulation model — scenario creation menu.

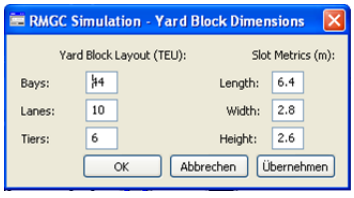

Figure A.3: Screenshot of RMGC simulation model — yard block dimensions menu.

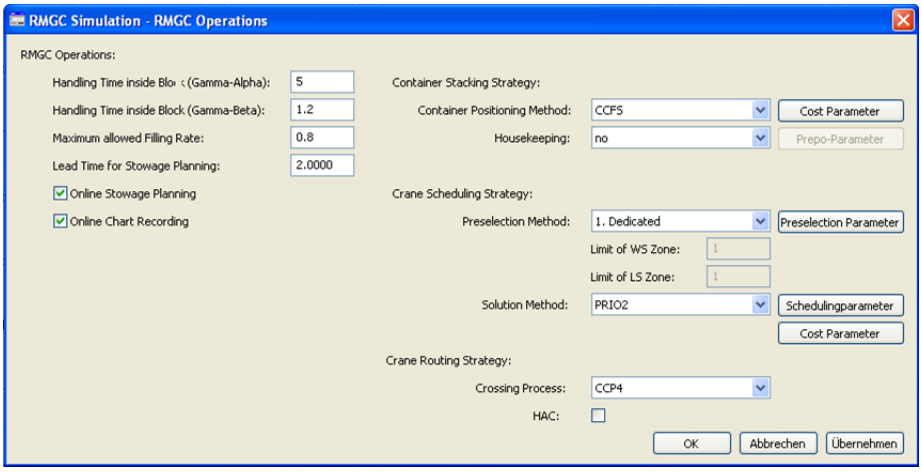

Figure A.4: Screenshot of RMGC simulation model — RMGC operations menu (TriRMGC).

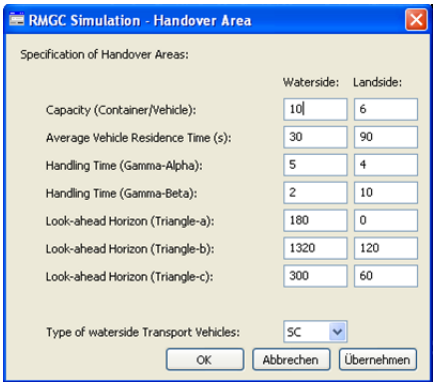

Figure A.5: Screenshot of RMGC simulation model — handover areas menu.

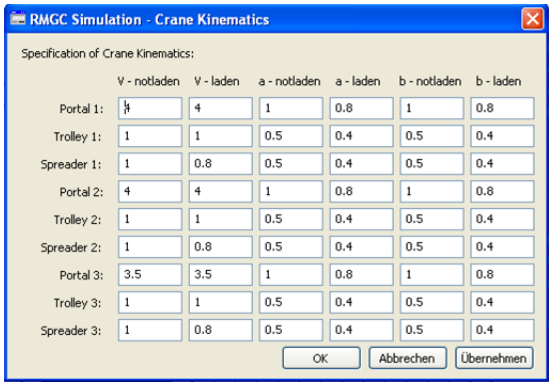

Figure A.6: Screenshot of RMGC simulation model — crane kinematics menu (TriRMGC).

A.2 Screen-Shots of Crane Animation

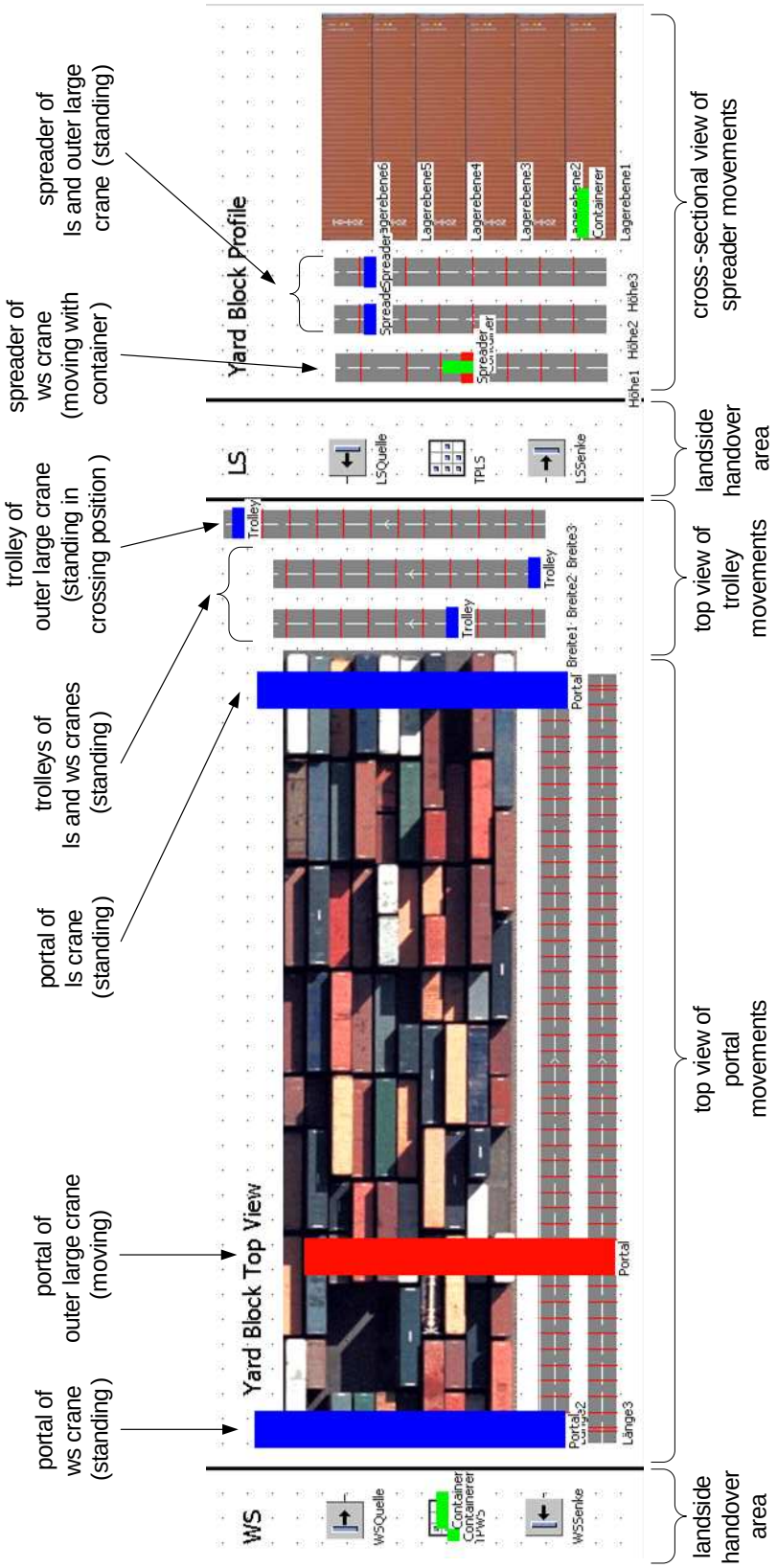

Figure A.7: Screenshot of RMGC simulation model — exemplary visualisation of RMGC operations (TriRMGC).

Bibliography

Bangsow, S. (2008). *Fertigungssimulation mit Plant Simulation und SimTalk.* Munich: Carl Hansner Verlag.

Choe, R., T. Park, M. O. Seung, and R. R. Kwang (2007). Real-time Scheduling for Non-Crossing Stacking Cranes in an Automated Container Terminal. In M. A. Orgun and J. Thornton (Eds.), *AI 2007: Advances in Artificial Intelligence*, Volume 4830 of *Lecture Notes in Computer Science*, pp. 625 – 631. Berlin et al.: Springer.

Dekker, R., P. Voogd, and E. van Asperen (2006). Advanced Methods for Container Stacking. *OR Spectrum 28*(4), pp. 563 – 586.

Drewry (1998). *World Container Terminals: Global Growth and Privat Proftits.* London: Drewry Shipping Consultants.

Grötschel, M., S. O. Krumke, J. Rambau, T. Winter, and U. Zimmermann (2001). Combinatorial Online Optimization in Real Time. In M. Grötschel, S. O. Krumke, and J. Rambau (Eds.), *Online Optimization of Large Scale Systems*, pp. 679 – 704. Berlin et al.: Springer.

Hall, N. G. and M. E. Posner (2001). Generating Experimental Data for Computational Testing with Machine Scheduling Applications. *Operations Research 49*(6), pp. 854 – 865.

Hartmann, S. (2004). Generating Scenarios for Simulation and Optimization of Container Terminal Logistics. *OR Spectrum 26*(2), pp. 171 – 192.

Hartmann, S., D. Briskorn, and N. Kemme (2007). Simulation und Optimierung fahrerloser Transportsysteme. *Industrie Management 23*(4), pp. 37 – 40.

Kang, J., K. R. Ryu, and K. H. Kim (2006). Deriving Stacking Strategies for Export Containers with uncertain Weight Information. *Journal of Intelligent Manufacturing 17*(4), pp. 399 – 410.

Kemme, N. (2011). *Design and Operation of RMGC Systems — A Simulation Study on the Performance Effects of alternative Designs and Planning Strategies for automated Rail-Mounted-Gantry-Crane Systems at Seaport Container Terminals.* Ph.D. Thesis, University of Hamburg. Work in progress.

Kim, K. H., Y. M. Park, and M.-J. Jin (2008). An Optimal Layout of Container Yards. *OR Spectrum 30*(4), pp. 675 – 695.

Kleijnen, J. P. C. (1999). Validation of Models: Statistical Techniques and Data Availability. In P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans (Eds.), *Proceedings of the 1999 Winter Simulation Conference*, Phoenix, AZ, pp. 647 – 654.

Kolisch, R. and A. Sprecher (1995). Characterization and Generation of a General Class of Resource-Constrained Project Scheduling Problems. *Management Science 41*(10), pp. 1693 – 1704.

Kozan, E. and P. Preston (1999). Genetic Algorithms to Schedule Container Transfers at Multi-Modal Terminals. *International Transactions in Operational Research 6*(3), pp. 311 – 329.

Petering, M. E. H., Y. Wu, W. Li, M. Goh, and R. de Souza (2009). Development and Simulation Analysis of Real-Time Yard Crane Control Systems for Seaport Container Transshipment Terminals. *OR Spectrum 31*(4), pp. 801 – 835.

Saanen, Y. A. (2004). *An Approach for Designing Robotized Maritime Container Terminals.* Ph.D. Thesis, Technical University of Delft, Rotterdam.

Valkengoed, M. P. J. v. (2004). How passing Cranes influence Stack Operations in a Container Terminal: A Simulation Study. Diploma Thesis, University of Amsterdam.

Voogd, P., R. Dekker, and P. J. M. Meersmans (1999). FAMAS-Newcon: A Generator Program for Stacking in the Reference Case. Econometric Institute Report EI 9943-/A, Erasmus University of Rotterdam.