Discrete-Event Control and Optimization of Container Terminal Operations

Tri Cahyono, Rully

DOI:
10.33612/diss.156020098

Publication date:
2021

Discrete-Event Control and Optimization of Container Terminal Operations

Rully Tri Cahyono

The research described in this dissertation has been carried out at the Engineering and Technology Institute (ENTEG), Faculty of Science and Engineering, University of Groningen, The Netherlands.

This dissertation has been completed in partial fulfillment of the requirements of the Dutch Institute of Systems and Control (DISC) for graduate study.

Printed by Ipskamp
Enschede, The Netherlands

Cover by Andi Yudha Asfandiyar

university of
groningen

Discrete-Event Control and Optimization of Container Terminal Operations

PhD thesis

to obtain the degree of PhD at the
University of Groningen
on the authority of the
Rector Magnificus Prof. C. Wijmenga
and in accordance with
the decision by the College of Deans.

This thesis will be defended in public on

12 February 2021 at 16.15 hours

by

Rully Tri Cahyono

born on 5 January 1986
in Jember, Indonesia

Supervisors
Prof. B. Jayawardhana
Prof. J.M.A. Scherpen

Assessment committee
Prof. I.F.A. Vis
Prof. D. Bauso
Prof. R.R. Negenborn

I am,
because we are

Dedicated to Intan and Kinan

Acknowledgments

When I arrived in Groningen in December 2012, I did not foresee that in a country where its climate was cold began a very warm PhD journey. The completion of the PhD is not possible without help and support of many people.

I would like to express my utmost gratitude to my first promoter, Prof. Bayu Jayawardhana. Bayu, your intelligence is unquestionably, and I have learned much from you on how to be capable in conducting academic research. But what I admire the most is your humility. You earn respect while still being humble, and you patiently devote a lot of time to develop your students. These qualities are something I very rarely see from top-level academia in Indonesia. Let me also sincerely thank to your family, where without their patience to let you supervise me during many summer nights, I doubt that I will ever make to this phase.

I would also like to convey my gratitude to my second promoter, Prof. Jacquelien Scherpen for her academic guidance, critical comments, and motivation.

I gratefully thank DIKTI who made this PhD possible with their scholarship. Some parts of the research are supported by Indonesia Port Corporation (IPC), and I thank Pak Dana Amin for the opportunity. I would like to express my appreciation to Prof. Iris Vis for the fruitful discussion on seaports. I thank to all my colleagues in KK SITE-ITB for their support during the second phase of this PhD in Bandung.

I and my wife received much help from the Indonesian community in Groningen, especially during the hard times after our son was born. I would like to convey warm gratitudes to all of you, and also the friends from the DTPA and SMS groups.

To all my family, I do not have a proper ode for their unconditional love and support. This thesis is dedicated to my lovely son and wife; Kinan, who pushes me to be a good example everyday (I am still trying hard!), and Intan, whose confidence in me, is the only sure thing in a world of dreadful uncertainty.

Bandung, January 2021
Rully Tri Cahyono

Contents

List of Abbreviations

AGV	Automated guided vehicle
ASC	Automated straddle carrier
BAP	Berth allocation problem
BB	Branch and bound
BCAP	Berth and crane allocation problem
CAP	Crane allocation problem
CY	Container yard
DBAP	Dynamic berth allocation problem
DBQA	Density-based quay crane allocation
DES	Discrete-event systems
DESDIS	Discrete-event systems with dynamic input sequence
ET	External truck
ETA	Expected arrival time
FCFS	First come first served
FSM	Finite state machine
GA	Genetic algorithm
GCR	Gross crane rate
HFLL	Heavy-first ight-last
HPSO	Hybrid particle swarm optimization
I-BCAP	Integrated berth and crane allocation problem
ILP	Integer linear programming
IT	Internal truck
LMI	Linear matrix inequalities
LP	Linear programming
MILP	Mixed integer linear programming
MPA	Model predictive allocation
MPC	Model predictive control
OR	Operations research

QC	Quay crane
RMGC	Rail-mounted gantry crane
RS	Reach staker
RTGC	Rubber-tyre gantry crane
TEU	Twenty foot equivalent unit
TOS	Terminal operating systems
VRP	Vehicle routing problems
VWT	Vessel waiting time
YC	Yard crane

Chapter 1

Introduction

Chapter 1

Introduction

The development of a standardized containerization system has marked the era of container terminals. Globalization of economic activities has promoted international trade where maritime transportation remains an important mean, as its cost to transport products from points of origin to points of destination is the cheapest among other modes of transportation. The simplicity and standardized operations of container cargoes have made them favorite in maritime transportation. Recently, more than 17.1% of sea-borne cargo is transported in containers, and the yearly growth is higher than other maritime transportation modes [83].

The importance of container terminal is shown with the annual growth of containerized international trade of 6.4% [83]. In 2018 only, it was estimated that 195 million TEUs was transported in the world; almost five times more than those two decades ago. Consequently, container terminals as one of the main actors in maritime transportation have also enjoyed the growth in global containerized trade. As presented in Table 1.1, the entire top ten terminals in the world have experienced significant increase in yearly TEUs handled [1]. We can see that most of recent top terminals lie in the East Asian region, which shows its outstanding economic activities in recent years.

The complex operations of a container terminal are costly and include million dollars investment in infrastructure and equipment [31]. The terminal operators

Table 1.1: *Top 10 world container terminal seaports. Volumes are in million TEU.*

Rank	*Port*	*Volume 2016*	*Volume 2015*	*Volume 2014*
1	Shanghai	37.13	36.54	35.29
2	Singapore	30.90	30.92	33.87
3	Shenzhen	23.97	24.20	24.03
4	Ningbo-Zhoushan	21.60	20.63	19.45
5	Busan	19.85	19.45	18.65
6	Hong Kong	19.81	20.07	22.23
7	Guangzhou	18.85	17.22	16.16
8	Qingdao	18.01	17.47	16.62
9	Jebel Ali	15.73	15.60	15.25
10	Tianjin	14.49	14.11	14.05

meet unrelenting demand from shipping liners to provide efficient operations, since the liners themselves are required to avoid any delay in the operations. A hold-up in a seaport will create multiple delaying effect in the liners' subsequent operations [82].

The importance of container terminals in the international trade, high cost incurred in the operation processes and stiff competition among terminals have pushed terminal operators to improve their services. Two options that are commonly looked at are 1) investment in additional equipment; and 2) improvement of the operation of the current process. The second option has motivated the emergence of new research in this field.

The works presented throughout this thesis focus on the modeling, control design strategies and optimization for container terminal operations. In the following, we will provide brief literature overview on topics that are related to our various contributions throughout the thesis.

1.1 Container terminal operations

A container terminal operations can be classified into three main areas i.e. seaside, storage, and transport [76]. The general layout of a container terminal is shown in Figure 1.1. It is shown that a number of ships can dock at various berth positions along the seaside and several quay cranes (QC) can be assigned to every berthed ship for loading and unloading containers. There are internal trucks (IT) waiting beneath the QC and they transport the containers to some specific destinations at a container yard (CY). The containers are then stored in the CY and several yard cranes (YC) re-allocate them internally within the CY or load/unload them to/from the external trucks (ET).

The seaside is a section where incoming ships arrive to the seaport and the the terminal operator allocates a berth positios and QC(s) to each vessel. This is known as berth and crane allocation problem (BCAP), where a detailed review is provided in [18]. A ship's loads is represented by its number of containers, where each box of container is measured as a twenty feet equivalent unit (TEU). The assigned QCs will unload a pre-determined number of boxes, known as the inbound/import containers, and they will finally be taken out by ET to the hinterland. Vice versely, there are also outbound/export containers. There is also the third container type called transshipment, in which a group of containers, after a temporary storage in the terminal, are transferred to other ships.

As QCs are the most expensive equipment in the terminal, the seaside operations is very important. Two measures are commonly used to evaluate the seaside's performance. The first is the gross crane rate (GCR) which is the average number of containers lifted per QC working hour. The second one is the vessel waiting

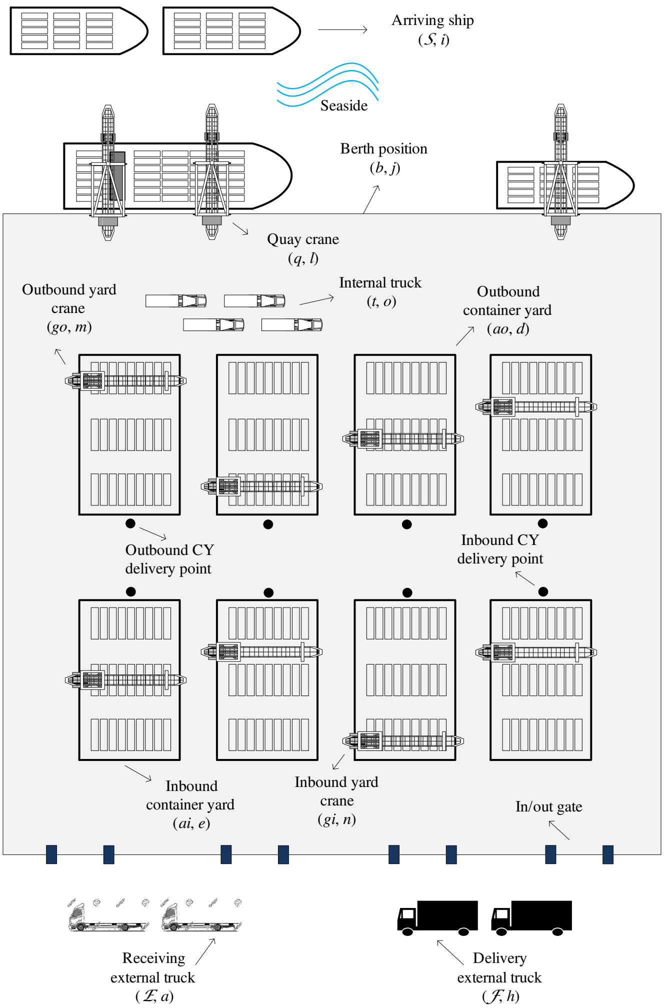

Figure 1.1: *Standard multi-level control configuration*

times (VWT) which is the total amount of time spent by ships to complete its load and unload operations [64].

The storage operations pertain to the management of containers in the CY, where the detail review is provided by [16]. A container position in the CY is defined by its row, bay, and tier. There are two decisions in these sub-operations, the positions a group of containers should be stored, and allocation of YC to handle them from/to IT. The container placement at the right positions in the CY is important. If an ET comes to the CY and the targeted container is not in the top tier, the terminal operators has to assigned YC to re-arrange the containers positions.

Sub-operations	Strategical decision	Tactical decision	Operational decision
Seaside	Quay lengths, Number of berths, Number of QCs	QC assignment to berths	QC and berth allocation and scheduling
Storage	CY capacity, Number of blocks, Number of YCs	YC assignment to blocks	YC allocation and scheduling
Transfer	Number of ITs, IT pooling area	IT assignment to blocks and berths	IT allocation and scheduling

Table 1.2: *Types of decision making level in container terminal operations.*

This process is known as housekeeping/marshaling, and highly avoided due to its unproductivity.

The seaside and storage sub-systems are connected with the transfer operations, which is discussed in [17]. In these sub-operations, the transporter known as the IT handles the container delivery between the QC and CY area. The terminal operators have to allocate number of working ITs in the most efficent ways.

The decision making in container terminal operations usually fall into three categories; strategical, tactical, and operational [76]. We divide the types of decision performed by terminal operators in an integrated terminal operations in Table 1.2.

The strategical level and operational level have the largest and the smallest planning horizon, respectively. This classification also applies in other areas such as supply chain management and production systems. For the decision making process in the tactical-level (i.e. weekly) or in the strategical-level (i.e. monthly), dynamical models have been developed to describe the dynamics in container terminal operations, see e.g. [2, 4, 11]. These dynamical models are subsequently used for resource allocation in container terminals using model predictive control. In particular, the models are used as predictive models of the process during the optimization step. In these papers, resource allocation is expressed as percentage of servers (equipments) capacity to transport containers to the subsequent server. As an alternative to [2, 4] where the percentage of servers is used as the decision variable, the control variables used in [11] are mainly the starting and finishing operation time of quay cranes, and the deployment of internal trucks and straddle carriers in berth and container yard.

The aim of the terminal operators is to operate the container terminal to meet the customers' demand in efficient manner in the least possible cost. As presented in Table 1.2, in this thesis we will focus on the tactical and operational levels of decision making in the container terminals. The current research efforts to achieve this aim will be discussed in the Chapter 1.3.

1.2 Discrete-event systems

Discrete-event systems (DES) are a class of systems where the state variables evolve according to discrete events that take place based on interactions among different (continuous- and/or discrete-) state variables in the systems [21]. A classical example of DES is a queuing system, in which, a new discrete-event is associated to the serving of new customer after the previous one from the previous discrete-event time has been served. We refer interested readers to [19] for an extensive discussion on the modeling and analysis of DES.

For the past few decades, DES framework has been used to model and to control a large class of physical and cyber-physical systems, which includes, the control of logistics systems, internet congestion control, manufacturing systems and many others that can be described by petri nets or finite-state machine/automata. With its wide application, the DES has attracted many researchers, including in the control community. Some examples of these works are discussed in [21, 27, 63, 68]. Fairly recent applications of DES in transportation and manufacturing systems are presented in [70] for general transportation and manufacturing systems.

Container terminal operations are highly suited with the evolving of event time framework in the DES. The state variables in the terminal operations are for instance the starting time, operations time, and finishing time of each equipment in the seaside, storage, and transfer sub-systems. Those state variables evolve every time an equipment finishes its operations. The asynchronous state-time among many equipment in the terminal add the complexity of the DES in the terminal operations.

When DES involve discrete-state with discrete input variables, the optimization/control of such DES leads to a combinatorial optimization problem which is NP-hard. The nature of DES as a class of NP-hard problems make it often deals intensively with combinatorial optimization. One can resort to a standard algorithm for solving combinatorial problems in DES which is the branch and bound (BB) method. As shown in [58], the BB method can converge to the global maxima/minima for some classes of DES optimization problems. Other well-known heuristic methods for solving combinatorial optimization problems with DES are genetic algorithm and particle swarm methods. In the paper, an analysis is presented and shows that from the Chebyshev inequality, the solutions obtained from the BB method converge to the global maxima/minima. It is also further shown that the discrepancy between the heuristic and global optimization is not significant.

Although the BB and other heuristic methods can be used to find a sub-optimal solution to the combinatorial problem for DES, the main drawback lies with the facts that the algorithms are limited only to the case where the problem can be recasted as a static optimization problem [25]. In this case, the static refers to constraining the dynamic problem by some terminal conditions and all possible control inputs are well-defined or known apriori within the given time interval (up to the terminal time).

This approach may no longer be feasible when the terminal conditions are free with infinite time horizon and when the input set changes dynamically and cannot be known apriori ahead of time. The latter case is commonly found in many DES application, such as transportation, scheduling, and logistics, where the actual incoming and outgoing goods always differ from the transmitted goods manifest and where the actual incoming and outgoing vehicles always differ from the precomputed plan. By static, we define that the input sets are known a priori before the optimization processes start. Therefore, a dynamic input set with possible changing input set cannot be handled by the BB method. The real time inputs are commonly found in many DES application, such as transportation, scheduling, and logistics.

Containers terminal operations are a class of logistics systems where semi-Markov models are commonly used, especially in inventory management [26, 73, 92]. Inventory in terminals can be seen as the containers, where the operators would like to manage it efficiently. While semi-Markov models can also capture the state evolution of the systems, the conditional probability that describes the transition from a state to another is usually known apriori [26, 73, 92]. Therefore, to handle real-time aspects in the terminal operations, some works in this area use the DES approach [2, 3, 4, 11, 89, 90, 91]. To model a system where its natural evolution is discrete (such as container terminals), [19] also recommends to use the DES modeling framework.

In [25], a dynamic DES model is developed for train scheduling problem where the frequent changes to the train operations (schedule, obstacle, rail availability) have limited the use of BB and similar algorithms. Instead of using BB, a greedy travel advance strategy is proposed in [25] on the basis of a dynamic DES model, which is able to find the sub-optimal control inputs of the train schedules with a framework similar to line search algorithm. The possible solutions in each iteration are limited to the group of trains in the same vicinity of direction and speed. Another related paper is [32], which studies a particular DES with dynamic input sets. The problem setting which includes complex systems in [32] falls into combinatorial problems. In this case, the events in DES are asynchronous where the states of each sub-system do not necessarily follow the same clock times and an LMI-based controller is proposed to solve such problem. By solving some linear matrix inequalities (LMI) that correspond to a desirable Lyapunov function, the controller are able to give sufficient results, where the cost function monotonically decreases.

A similar DES with asynchronous event transition can also be found in our previous works in [12, 13]. In these works, a model predictive allocation (MPA) method is proposed in conjunction with a pre-conditioning step. In particular, the DES model of container terminal operations is used to compute an optimal input sequence for a finite event horizon where the input sequence is heuristically pre-conditioned for accommodating the combinatorial optimization step. The proposed MPA method follows the same procedure as the model predictive control approach. The efficacy of our proposed method has been shown in both simulation as well as in real-life experiment. In this method, we have used the well-known first-come first-serve (FCFS) or the heavy-first light-last (HFLL) pre-conditioning step to the current input sequence and then truncate it, prior to computing the optimal solution in the model predictive step.

1.3 Container terminal optimization

The needs of more efficiency in complex container terminal operations have motivated researchers to put efforts in this field. The works mainly deal with the optimization of the terminal. As reviewed in [16, 17, 18], the typical operations that have been studied assume that the entire information is precisely known apriori, so that linear programming can be applied for solving the equipment allocation in the seaport. Examples of the information are for instance, ship arrivals and availability of storage positions both in container yard and vessel bays. However, this setting does not capture the operational-level decision making process where in fact terminal is a volatile environment and the set of inputs dynamically changes. The terminal operator knows about the ship arrivals or exact available storage positions only for a brief time horizon and the operations process itself is a dynamical process. Hence, the non-robustness and non-adaptiveness of the state-of-the-art approach to the dynamically changing environment has led to the wide adoption of a heuristic approach that is a combination of the first-come first-serve allocation strategy in the terminal.

When we talk about modeling in general container terminals, for the decision making process in the tactical-level (i.e. weekly) or in the strategical-level (i.e. monthly)[1], dynamical models have been developed to describe the dynamics in container terminal operations, see e.g. [2], [4], [84] and [42] where containers, trucks and ships are considered as a continuous flow, as opposed to considering them as discrete events. These dynamical models are subsequently used for resource allocation in container terminals using model predictive control. In particular, the

[1] There are three types of decision making in container terminal operations as in [76, 86] i.e. strategical, tactical, and operational level. The strategical level and operational level have the largest and the smallest planning horizon, respectively. This classification also applies in other areas such as supply chain management and production systems.

models are used as predictive models of the process during the optimization step. In these papers, resource allocation is expressed as percentage of servers (equipments) capacity to transport containers to the subsequent server. As an alternative to [2] and [4] where the percentage of servers is used as the decision variable, the control variables used in [84] are mainly the starting and finishing operation time of quay cranes, internal trucks and straddle carriers deployed in berth and container yard. These dynamical models are subsequently used for resource allocation in container terminals using model predictive control. The similar concept with more elaboration is found in [43], where not only the allocation of the cargo is considered, but also the cargo due date is optimized with a predictive horizon approach. In this regard, the setting in [43] also falls into tactical-level decision making process, since the operations time (i.e. the scheduling) of the equipment in the terminals are not considered.

In particular, the models are used as predictive models of the process during the optimization step. In these papers, resource allocation is expressed as percentage of servers (equipments) capacity to transport containers to the subsequent server. As an alternative to [2, 4] where the percentage of servers is used as the decision variable, the control variables used in [84] are mainly the starting and finishing operation time of quay cranes, internal trucks and straddle carriers deployed in berth and container yard.

The container terminal operations are dependent on each sub-system. For instance, the exact deployment of transporters are only known after QC work schedule and CY storage plan are definitive. Therefore, to make an optimal planning, the entire systems have to be considered when making decisions. Whereas in reality, the ongoing research tends to make limitations in each of the terminal's sub-system [16]. In practice, the terminal operators usually use traditional methods in creating terminal daily planning, for instance first-come-first served (FCFS) in making the berth and quay crane allocation (BCAP), ship stowage plan, and CY storage plan preferred are. Even in the commonly used Terminal Operating Systems (TOS), these methods are commonly found, whose its optimal performance cannot be guaranteed.

The aim of the terminal operators is to operate the container terminal efficiently in the least possible cost with minimal dissatisfaction level from its customers. The terminal operators execute a series of works to deliver the containers into: 1) CY, for the inbound ones, and 2) vessel, for the outbound ones. The storage configuration of the inbound containers in the CY and of the outbound containers in the vessels are known as the storage plan and stowage plan, respectively [24, 94].

1.4 Contributions

We discuss the contribution of this thesis in this sub-chapter. The contributions are five fold. The basic system observed in this research is the integrated berthing and crane allocation process. The systems are asynchronous and as our first contribution, we propose a dynamical modeling framework for such systems. Secondly, to guarantee effective solutions, we propose a MPA method. Thirdly, we analyse the optimal allocation problem using our proposed MPA method from the mathematical aspect. Lastly, we conduct experiments with large-scale hypothetical data and also field experiment in a real-life container terminal for the integrated berth and crane allocation problem. We also extend the dynamical modeling framework for two cases in maritime transportation network optimization and end-to-end integrated terminal operations. In those two cases, the MPA method that we have tested on large-scale simulation has shown the efficacy of the solution.

1.4.1 DES modeling for integrated berthing process

We propose in Chapter 3 a dynamical modeling framework using a DES formulation that describes both the real-time and continuously changing set of ship arrivals at any given time, as well as, the discrete-event dynamics during the berthing and loading/unloading process. A result in this endeavor has appeared in [13]. The proposed approach improves the one used in the DBAP as in [30] and [39]. To the best of our knowledge, the dynamic aspect of ship arrivals has not yet been discussed in the present research.

The DES formulation fits better to terminal operations than the usual periodic discrete-time systems description since there is aperiodicity in the ships' arrival time and the operations' time among different berthing positions is usually asynchronous. The book [19] provides an excellent exposition to the modeling and analysis of DES. In [70] such DES modeling framework is used to describe manufacturing and transportation systems. While in [11], the event is triggered every time uncertainty occurs in the terminal. Our proposed modeling framework fits well with the common practice in the terminal operations. Firstly, the berth planning is done based on the pro forma windows of incoming ships where the information may be incomplete and will change during the execution window. In the current state-of-the-art operations research (OR) modeling framework, such uncertainty is embedded in the constraints and introduces sub-optimality in the solution. In our framework, the dynamical modeling of ships' arrival allows for a real-time planning according to real-time factual information from the arriving ships. Secondly, the state equations (which are given by difference equations) capture the sequential process in seaside operations to a large extent and is also validated later in our real life experiment. Thirdly, the simple model allows us to not only gain insight to the sequential process, but also to deploy it in our real-time integrated allocation algorithm.

As reviewed in [18], the BAP (and container terminal operations problems in general) falls into NP-hard problems because of its complexity. The complexity itself is caused by the dimension of the problem i.e. the number of ships, the number of berth positions, and the number of quay cranes. One of the popular methods in solving the NP-hard problems, including the BAP, is genetic algorithm (GA) [18]. The GA allows flexibility for its users to solve the original problem through GA specific algorithm. The GA is employed in [20] and [40]. Another technique to solve the BAP is Tabu search as in [71] where the objective function is to minimize the housekeeping cost that is affected by the resulting ship schedule. While in [93], Lagrangian relaxation is used.

1.4.2 Model predictive allocation method for integrated berthing process

The non-robustness and non-adaptiveness of the above mentioned approaches to the dynamically changing environment have led to the wide adoption in real-life condition of a very simple heuristic approach, that is a combination of the first-come first-served strategy for the berthing allocation and of the density-based strategy for the quay crane allocation. In order to handle such dynamic environment (including the real-time information on arriving ships), we propose a novel real-time integrated berthing and crane allocation method in the present paper as our second contribution of this thesis in Chapter 3 which is based on model predictive control principle and rolling horizon implementation.

1.4.3 Mathematical analysis

We study an optimal input allocation problem for a class of discrete-event systems with dynamic input sequence (DESDIS) as presented in presented in Chapter 4. A similar DES with asynchronous event transition can also be found in Chapter 3. In these works, a model predictive allocation (MPA) method is proposed in conjunction with a pre-conditioning step. In particular, the DES model of container terminal operations is used to compute an optimal input sequence for a finite event horizon where the input sequence is heuristically pre-conditioned for accommodating the combinatorial optimization step. The proposed MPA method follows the same procedure as the model predictive control approach. The efficacy of our proposed method has been shown in both simulation as well as in real-life experiment. In this method, we have used the well-known first-come first-serve (FCFS) or the heavy-first light-last (HFLL) pre-conditioning step to the current input sequence and then truncate it, prior to computing the optimal solution in the model predictive step. While the re-ordering of the sequence (either using FCFS or HFLL) has played an important role in [12, 13], the mathematical analysis on the re-ordering of the input sequence in the pre-conditioning step is still missing.

In this case, the input space is defined by a finite sequence whose members will be removed from the sequence in the next event if they are used for the current event control input. Correspondingly, the sequence can be replenished with new members at every discrete-event time. The allocation problem for such systems describes many scheduling and allocation problems in logistics and manufacturing systems and leads to a combinatorial optimization problem. We show that for a linear DESDIS given by a Markov chain and for a particular cost function given by the sum of its state trajectories, the allocation problem is solved by re-ordering the input sequence at any given event time based on the potential contribution of the members in the current sequence to the present state of the system. In particular, the control input can be obtained by the minimization/maximization of the present input sequence only.

1.4.4 Field experiment and numerical analysis

The discrete-event modeling is also implemented for integrated container terminal operations and terminal network optimization in Chapter 5 and 6, respectively. In both cases, we evaluate the performance of our model predictive allocation strategy using: (i). extensive Monte-Carlo simulations using realistic datasets; (ii). real dataset from a container terminal in Tanjung Priuk port, Jakarta, Indonesia. The numerical experiments are provided to show the ability of the DES models in replicating the actual complex operations in the terminals. The large scale simulations also test the efficacy of the MPA algorithm which has been developed in Chapter 3. From the simulations, we understand that firstly, the DES models are able to mimic the complex operations in the terminal. This is further shown in the field experiment, where the state variables obtained from the models follows the data from the experiment. Secondly, from the large-scale simulations with hypothetical datasets, it shows the efficacy of the MPA compared to the meta-heuristic methods which are commonly used in this field of research. Even though, we have to note that the MPA needs bigger computation efforts than the existing methods. For the BCAP problem in Chapter 3, in addition to the two kind of aforementioned simulations, we also performed real life field experiment in the same container terminal. As has been mentioned in [86], the contribution on the real life field experiment provides an important insight to the performance of novel allocation method in reality which is typically not reported in the literature.

1.4.5 Extension of BCAP models to the seaport network and integrated terminal operations

We extend the implementation of the dynamcal models that has been developed in the BCAP to the cases in seaport network (Chapter 5) and integrated container terminal operations (Chapter 6). We study container terminal network performance under heterogeneous distributed operational BCAP policy. The operational optimization policy in each terminal is confined only to the seaside operations problem. Hence, only berth and quay crane allocation is considered in each seaport. We have two valid reasons for this research boundary. First, shipping liners which call to a seaport deal greatly with seaside operations. Second, an effective BCAP will contribute hugely to the whole performance of the terminal, since bottlenecks often occur in this sub-system, where the discussion is provided in [18, 40, 48, 66]. The bottlenecks in berthing process has even more important consideration, since QC is usually the most expensive equipment in the terminal [13, 48]. Therefore improvement in BCAP will greatly profit the overall operational performance of a single terminal, and it has been explained extensively in [53, 82, 87] that the terminal network's performance is heavily affected by the performances of each terminal in the network.

We analyze the container terminal network operations under heterogeneous distributed BCAP policy as the main bottleneck in terminal operations, and consequently affect greatly to the network performance. We study the improvement of the network operations from the perspective of terminal operators [48, 81]. Currently, as exemplified in [6] and [87], the common point of views of network operations improvement are from the shipping liners. In our case, we provide analysis and insights for the terminal operators on the optimization of network operations in a distributed way and with minimal effort. We investigate the performance of state-of-the-art MPA-based BCAP approach as proposed in [12] and [13] to improve network operations. The MPA gives better results compared to the state-of-the-art methods in BCAP [13]. We also propose methods for selecting important seaports in the network to which the MPA-based BCAP policies are applied.

In Chapter 6, we extend the modeling framework in [12, 13] to the integrated container terminal setting. In the state-of-the-art research, the integrated operations are commonly modeled with static OR-based approach as can be found in [33, 36, 44]. Subsequently, we propose a simultaneous allocation and scheduling of QC, YC, and IT in the operations planning as our second contribution. The approach is based on the model predictive algorithm (MPA) as presented in [12, 13] and its efficacy is demonstrated in a real experiment in Jakarta's main seaport, Tanjung Priok. The MPA is based on model predictive control (MPC) which is often use to find optimal solution of DES models [70]. Recently, a preliminary mathematical analysis of the MPA algorithm has been reported in [14]. This proposition is a prominent aspect that can not be completely achieved in [11, 89, 90], where only the lower-level controllers of equipments' scheduling are modeled dynamically while the allocation itself is done via a deterministic and static perspective with linear programming techniques.

1.5 Publications

Several peer-reviewed journal and conference papers contributing to this thesis are as follows.

Journal papers

- ”Discrete-event systems modeling and the model predictive allocation algorithm for integrated berth and quay crane allocation”, IEEE Transactions on Intelligent Transportation Systems, 2020, 21(3), pp. 1321-1331. (Chapter 3 of this thesis)
- ”Towards a competitive terminal network via heterogeneous and distributed dynamic optimization policies in the berth and quay crane allocation operations”, *under review*. (Chapter 5 of this thesis)
- ”Simultaneous allocation and scheduling of quay cranes, yard cranes, and trucks in dynamical integrated container terminal operations”, *under review*. (Chapter 6 of this thesis)

Peer-reviewed conference papers

- ”On the optimal input allocation of discrete-event systems with dynamic input sequence”, 58rd IEEE Conference on Decision and Control, December 11-13, 2019, Nice, France. (Chapter 4 of this thesis)
- ”Dynamic berth and quay crane allocation for multiple berth positions and quay cranes”, 14th European Control Conference, July 15-17, 2015, Linz, Austria. (Chapter 3 of this thesis)

Some materials on this thesis have been also partially presented at (local) scientific meetings as follows.

Conference abstracts

- ”Dynamic berth and quay crane allocation for complex berthing process in container terminals”, 34th Benelux Meeting on Systems and Control, March 24-26, 2015, Lommel, Belgium.
- ”Analysis of dynamic container terminal networks”, 35th Benelux Meeting on Systems and Control, March 22-24, 2016, Soesterberg, The Netherlands.

Poster

- ”Towards an integrated modeling of container terminal optimization”, ENTEG PhD Meeting, October 8, 2016, Groningen, The Netherlands.

1.6 Thesis outline

This thesis is organized as follows. Chapter 1 presents motivation and contribution of this thesis. Chapter 2 starts with preliminaries that provide necessary theoretical backgrounds for DES and MPC. The same chapter also provides the definition and examples of operations systems, especially their relation with DES modeling framework.

Chapter 3 discusses the DES modeling framework and MPA algorithm for integrated berth and quay crane allocation problem (BCAP), which will be the main foundation of this thesis. The framework in Chapter 3 is analysed mathematically in Chapter 4, which focuses on an optimal input allocation problem for a class of DES with dynamic input sequence (DESDIS). The modeling framework in Chapter 3 will later be used in Chapter 5 and 6 that discuss the extension for two applications. Firstly, the container network operations and secondly, the integrated end-to-end operations in container terminals. Finally, the conclusions and future works are given in Chapter 6.

Chapter 2

Preliminaries

Chapter 2

Preliminaries

We present in this chapter preliminaries of DES which later in Chapter 3 will serve as the modeling framework in this thesis. To bridge between the DES and operations systems, we also discuss the examples of DES in operations systems, especially related to the container terminal operations systems when available. Finally, we briefly present the concepts of model predictive control (MPC) as a popular method to solve the DES models.

2.1 Discrete-event systems

The discussion of DES is divided into three main parts. Firstly, we would like to remind the readers that DES is a class of systems, where the systems themselves can be modeled in several approaches, for instance state-space and operations research (OR). In this sub-chapter we focus to compare DES-based with OR-based modeling approach. The latter is currently ubiquitous and 'standard' to model operations systems, which is in fact the object of research in this thesis. Secondly, we discuss the characteristic of DES, and what kind of systems that suit the properties, especially in relation to operations systems. Thirdly, we will present one framework to graphically model DES problems with Petri nets.

2.1.1 Approaches in modeling

There are plenty of modeling techniques in science and engineering. For the clarity, in this thesis we only discuss the methods of systems modeling which have affinity with the utilization of quantitative approach. The term quantitative itself is closely related with mathematics. This to further distinguish with qualitative approach, which in some branches of sciences, conceptual framework design such as block diagrams is also called 'models' [22].

The input-output (I/O) modeling technique is briefly discussed in [46]. This method models the changes in outputs as the results from changes in inputs. We take the formulation of I/O modeling from [19]. The input and output variables are represented through two column vectors, $\mathbf{u}(t)$ and $\mathbf{y}(t)$, respectively

$$\mathbf{u}(t) = [u_1(t), .., u_p(t)]^T$$
$$\mathbf{y}(t) = [y_1(t), .., y_m(t)]^T.$$

We assume that there is some mathematical relationship between input and output, therefore there exists functions

$$y_1(t) = g_1(u_1(t), ..., u_p(t)), ..., y_m(t) = g_m(u_1(t), ..., u_p(t))$$

The I/O model is then defined in (2.1)

$$\mathbf{y} = \mathbf{g}(\mathbf{u}) = [g_1(u_1(t), .., u_p(t)), ..., g_m(u_1(t), .., u_p(t))]^T. \tag{2.1}$$

Another less common method in management science and operations communities is simulation [47], where a rule-based (if-then-else) approach is used to resemble the studied system. This is usually depicted in flowchart where each process or decision shows the sequence of events. The stochastic aspects of the systems are incorporated through random variates, which derived from statistical functions [47]. The common random variates is defined in (2.2)

$$X_i = f^{-1}(U_i), \quad U_i \in \{0, 1\}. \tag{2.2}$$

where X_i is the $i-$th random variate. The index i implies that the simulation is done step by step based on event/time-counters. U_i is the $i-$th random number whose value is distributed continuously uniform from zero to one, and $f^{-1}()$ is the inverse of statistical function considered in the system.

Others mathematical modeling methods which we will be the main focus in this thesis are state-space and operations research. One of the most common methods to model an observed system with a mathematical approach is state-space. This method emerged in the 1950s which uses differential equations to represent the systems [69]. A state-space involves the input $\mathbf{u}(t)$, the output $\mathbf{y}(t)$, and the state $\mathbf{x}(t)$. The complete equations is given in (2.3) and (2.4) [19]

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t), \quad \mathbf{x}(t_0) = x_0, \tag{2.3}$$
$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t), t). \tag{2.4}$$

The other method is operations research (OR), which emerged earlier during the 1940s. Interested readers in OR-based modeling can refer to [22, 35, 77]. The method mainly uses linear programming (LP) related technique to solve problems limited to several constraints. It has become the most popular technique in management science and operations management, hence the name [35, 77]. A

Table 2.1: *The difference between state-space and OR modeling*

Aspect of modeling	State-space	Operations research
Evolution of time	Dynamic	Static
Linearity	Linear and non-linear systems	Linear system by model's default
Feedback	Incorporated	Not-incorporated
Parameters	Deterministic and stochastic	Deterministic by model's default
Measurement of state over time	Incorporated	Not-incorporated

standard LP model is given in (2.5) [35, 77].

$$\begin{aligned} \text{max/min} \quad Z &= \mathbf{c}^T\mathbf{x} \\ \text{subject to} & \\ \mathbf{A}\mathbf{x} &= \mathbf{b} \\ \mathbf{x} &\in \mathbb{R}^+. \end{aligned} \tag{2.5}$$

The typical problem in LP is to select the best n decision variables which is represented by the vector $\mathbf{x}$. The evaluation of the decision variables is based on the cost function which is given by Z. We denote $\mathbf{c}$ as the column-vector $(n \times 1)$ of the cost-function coefficients. The decision variables have to satisfy constraints imposed by right-hand constraint of the column-vector $\mathbf{b}$ $(m \times 1)$, and $\mathbf{A}$ $(m \times n)$ is the matrix of technical coefficients.

Despite their common goals to accurately mimic systems into mathematical models, state-space has polar opposite approach to the OR counterparts. Based on summary from [35, 46, 69, 77] we show the difference between state-space and OR modeling in Table 2.1.

The differences naturally creates advantages of state-space models. The LP technique in OR has four main assumptions [35, 77]: 1) linearity, 2) certainty, 3) proportionality, 4) additivity. The two first assumptions make static-approach in OR-based modeling inevitable. The OR models usually assume that a set of parameters are known in the beginning of time-horizon. By default, the set can not change (in (2.5), there is no index of time in the LP model), therefore a real-time situation is not incorporated [77]. To make it 'real-time', OR-based models often use statistical functions, to reflect the possibility of changes in the parameters. The concept of real-time is important in this thesis, and will be discussed more thoroughly in Chapter 3.

An OR model receives sets of parameters, and tries to seek the best decision variables from the many possible combinations of such variables, such that the cost function is optimal. What happens in systems during optimization is not recorded. The information that is truly measured in the OR-model are what kind of effects (outputs) caused by the inputs. This black-box approach [22, 77] is quite similar with I/O based model as in [46].

We give an example with warehouse operations, which will later be discussed in Chapter 2.1.2. The goal of the operations is to seek optimum procurement policy, in which the observer determines the quantity of goods that the warehouse should acquire. In OR-model, the result will typically be firstly, the units of goods in each procuring period, and secondly, the interval between each period in the whole planning horizon [80]. The evolution of end inventory in each time-period t (or k) can not be tracked from OR-model's results. The deterministic parameters of OR-model as in Table 2.1 impose changes in OR-model's inputs. For instance, the inventory demand parameter is assumed fix for the whole planning horizon [80]. The changes in demand is facilitated through a specific statistical function, which is still not real-time. Those two drawbacks are not found in state-space modeling. The state in each time period (t or k) can always be traced. While the dynamic in state-space modeling incorporates the real-time aspect of the systems.

2.1.2 DES modeling

We have discussed in Chapter 2.1.2 the two superiority of state-space to OR modeling. Those are, 1) the inclusion of dynamic in the models, and 2) the recording of state changes in each time period during the whole planning horizon. In this sub-chapter we will present the discussion of DES modeling, to bridge into its application in the operations systems.

There are two types of states, continuous and discrete, where the combination of both are the hybrid state [19]. In accordance with the main topic in this thesis, we will discuss an example of a system with discrete states and later model the problems with DES as has been presented in [19]. In some cases, modeling a system of discrete-state approach is more natural and simpler to visualize than continuous-state [19, 69]. This is because the DES modeling is a series of logical statements. An example of the statements is, "if current state is x and something happens, then the next state is x', if those something else happens, then next state is $\hat{x}$". The DES application is found in many systems, no exception in the operations systems, such as queuing systems, manufacturing, transportation, and logistics.

We will discuss briefly an example of a simple warehouse system [19], which has been briefly presented before in Chapter 2.1.1. By using DES-modeling approach as in [19], we propose two input (control) variables namely $u_1(k)$ and $u_2(k)$ as the arriving and departure products, respectively. Those two inputs are defined as follows:

$$u_1(k) = \begin{cases} 1 & \text{if product arrives at the time } k \\ 0 & \text{otherwise} \end{cases} \quad (2.6)$$

$$u_2(k) = \begin{cases} 1 & \text{if product departs at the time } k \\ 0 & \text{otherwise} \end{cases} \quad (2.7)$$

The state of the system is $x(k)$ which is the end inventory in $k-$th time period. It is defined by the end inventory in the previous period ($x(k-1)$) and the two control variables. If a product arrives, and no one departs, there is an additional product at the warehouse. On the other hand, if no product arrives and a product is taken, the end inventory will be one less from the previous time period, provided there is at least one inventory left at the beginning of time period k, since negative inventory is not allowed. The DES model for the warehouse problem is given as follows in 2.8

$$x(k) = \begin{cases} x(k-1)+1 & \text{if } (u_1(k)=1, u_2(k)=0) \\ x(k-1)-1 & \text{if } (u_1(k)=0, u_2(k)=1, x(k)>0) \\ x(k-1) & \text{otherwise.} \end{cases} \quad (2.8)$$

We have seen the contrast between the warehouse modeling with DES and OR as has been explained in Sub-Chapter 2.1.1. With DES modeling, not only the inventory policy ($u_1(k)$ and $u_2(k)$) that can be optimized (with some techniques that will be later discussed in this thesis), but the state of systems ($x(t)$) can always be traced. We will later show that keeping the records will be useful to perform validation between the model we have built and the actual systems. In OR-based modeling, the validation is usually only performed as comparison between the cost functions from model, and the actual systems [22, 35, 77].

In Sub-Chapter 2.1.1, the depiction of a general continuous system is presented in (2.3)-(2.4). According to [19], the technique which are usually used to analyse continuous state-space is differential equation. As the opposite, in the discrete-state, the state-space X is discrete set. In this case, the discrete state-variables are only able to move from a discrete state-value to another at the discrete time step [19]. As a consequence from the integer numbers to describe the states, difference equations are usually used in DES modeling [19, 69].

There are two approaches in modeling DES, namely time-driven, and event-driven. The difference between these two approaches are what factor (time or event) triggers the movement of time k to $k+1$. In real worlds, it can also be seen when the observers review the systems' states. Most cases in DES are event-driven [19]. For instance, in the aforementioned warehouse systems modeling, the time counter of $k-1$ is moved to k if there are changes in $x(k)$, and it is not necessary for the two input variables $u_1(k)$ and $u_2(k)$ to exactly happen in exactly periodical

k (e.g. k evolves from 08:00, 08:15, 08:30, etc).

According to [19], an event in DES is described as a trigger from one state to another in the DES-system which happens instantaneously. In the inventory-warehouse example, the two events are $E = \{A, D\}$, where A denotes the product arrival, and D denotes the product delivery event. An event-driven DES can lead to asynchronous process, and it is more difficult to solve than the time-driven model [19, 69]. This is because, in event-driven DES the time counters for the parallel processes are not necessarily the same among each other. In a two parallel server queuing system, the k is moved to $k+1$ not based on time-counter, but possibly according to the ending of service time at any server. This lead to complexity, since at the other server, the service may still running. The asynchronous DES modeling will be the main focus of this thesis.

After discussing the example of DES and the approaches to model DES, the general representation of DES as cited from [19] is defined by

$$\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k), k), \quad \mathbf{x}(0) = 0, \tag{2.9}$$
$$\mathbf{y}(k) = \mathbf{g}(\mathbf{x}(k), \mathbf{u}(k), k). \tag{2.10}$$

The state space representation for the discrete state in 2.9-2.10 can be contrasted with the continuous state as in 2.3-2.4. For the case of linear DES is defined by

$$\mathbf{x}(k+1) = \mathbf{A}(k)\mathbf{x}(k) + \mathbf{B}(k)\mathbf{u}(k), \tag{2.11}$$
$$\mathbf{y}(k) = \mathbf{C}(k)\mathbf{x}(k) + \mathbf{D}(k)\mathbf{u}(k), \tag{2.12}$$

and the state-space for a DES in the time-invariant case is

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k), \tag{2.13}$$
$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k), \tag{2.14}$$

where $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, and $\mathbf{D}$ are all constant matrices of the systems parameters. The solution of a linear difference equation with initial condition $\mathbf{x}(0)$ and input $\mathbf{u}(0), .., \mathbf{u}(t)$ is taken from [69] as follows in (2.15)

$$\begin{aligned} \mathbf{x}(k) &= \mathbf{A}^k\mathbf{x}(0) + \sum_{j=0}^{k-1} \mathbf{A}^{k-j-1}\mathbf{B}\mathbf{u}(j), \\ \mathbf{y}(k) &= \mathbf{C}\mathbf{A}^k\mathbf{x}(0) + \sum_{j=0}^{k-1} \mathbf{C}\mathbf{A}^{k-j-1}\mathbf{B}\mathbf{u}(j) + \mathbf{D}\mathbf{u}(k). \end{aligned} \tag{2.15}$$

2.2 Operations systems

We have mentioned in the beginning of this sub-chapter that the object of this research (container terminal operations) is one kind of operations systems. The definition of operations is manufacturing and service processes that try to transform and create products and services in firms that will be used by customers [41]. As a consequence of the definition, transportation, physical distribution, and logistics also fall into the class of operations systems, because all of those subjects also deal with finding best efforts to deliver products and services to customers.

In this chapter, we will discuss the operations systems in relation with the application of DES in the systems. We begin with the common operations systems. Afterwards, we will focus with recent efforts to apply DES in container terminal operations.

2.2.1 DES in operations systems

As presented in Sub-Chapter 2.1 that OR is currently the most common modeling method in operations systems, there are some endeavors to use DES in operations systems modeling. We found in [21] an early application of DES in manufacturing systems. The study models production processes, where application can be found in flexible manufacturing systems and automated material handling systems. In this research, events are defined as the beginning and ending of a job in a particular machine. The state space of discrete production process in [21] is formulated as follows in (2.16) and (2.17)

$$X = XA \oplus UB \tag{2.16}$$

$$Y = XC \tag{2.17}$$

where $X = (x_1, ..., x_N)$ is the set of earliest starting times the production activities, and $U = (u_1, ..., u_R)$ is the control variables which show the sequence of parts processed in machines.

A is a weighted incidence matrix with dimension $N \times N$, where N is the number of all processing tasks for producing part in machine. B is the starting activities for all machines. Matrix C is a bipartite graph of last processing sequences in the corresponding machines. The goal is to order the jobs in the appropriate machines as given in (2.18)

$$Y(n) = Y(n-1)D \oplus \sum_{j \in J} (Y(n-q_j)D^j \oplus Y(n-q_j-1)\bar{D}_j) \tag{2.18}$$

where $D = KBA^*C$, and K is a feedback matrix. It is possible to order (2.18) so that

$$Y(n) = (\sum_{r=1}^{\triangle} Y(n-r)H^r)H^{0*} \tag{2.19}$$

if and only if $(H^0)^* = E \oplus H^0 \oplus (H^0)^2 \oplus ...$ exists. More detail explanation can be found in [21].

Another application of DES in operations systems can be found in [70]. In the research, the authors study DES with hard and soft constraints which later be solved by an MPC algorithm. In operations systems, hard constraints are defined as conditions that have to met to perform a task/job. On the other hand, soft constraints reflect pre-determined predecessors that can be violated with some penalties.

The systems studied are comprised of several operations and some cycles. The starting time of operation j in cycle k is defined in (2.20)

$$x_j(k) \geqslant d_j(k) \tag{2.20}$$

The cycle k can be seen as an event based system as have been explained in Sub-Chapter 2.1.2. Hard and soft synchronization constraints are defined

$$x_j(k) \geqslant x_i(k - \delta^*_{ij}(k)) + a_{ij}(k) \quad \forall i \in C_j^{\text{hard}}(k) \tag{2.21}$$

$$x_j(k) \geqslant x_i(k - \delta^*_{ij}(k)) + a_{ij}(k) - v_{ij}(k) \quad \forall i \in C_j^{\text{soft}}(k) \tag{2.22}$$

where delay in each cycle between operations is denoted by $\delta^*_{ij}(k)$ and in soft synchronization constraints, the synchronization can be broken by some penalties $v_{ij}(k)$. The soft synchronization constraints are later re-casted into MPC problems, whose solution is based on this following remark.

Remark 2.1. If $\hat{t}_{ij}^{\text{slack}}(k+l)$ is non-positive (or if there is another index i' such that $\hat{t}_{ij}^{\text{slack}}(k+l) > \hat{t}_{i'j}^{\text{slack}}(k+l)$), then $v_{ij}(k+l)$ does not influence the value of the objective function anymore. Therefore, the MPC cost function could be extnded with extra term

$$\rho \sum_{l=0}^{N_p-1} \sum_{j=1}^{n} \sum_{i \in C_j^{\text{soft}(k+l)}} v_{ij}(k+l) \tag{2.23}$$

with $\rho > 0$ a small number. In that way, the smallest possible values of the $v_{ij}(k+l)$ is obtained. This also determines which synchronizations are broken or not.

The modeling of MPC for DES with soft synchronization constrains in [70] is then applied to a production system. It is shown that the production problem can be solved by the proposition and online (real-time) inputs can be handled.

DES with partial synchronization is discussed in [23]. The DES is divided into two parts, the main system and the secondary system, and the interaction of two systems is governed by partial synchronizations. In this setting, the authors propose time-based DES, which differs from the event-based DES as cited in [21, 70].

The vectors defined in [23] are x, u, y which represents the vectors of counters associated with state, input, and output events. The index of the main and secondary system are 1 and 2, respectively. The behaviour of the system satisfy

$$\begin{aligned} &x_1(t) \geqslant A_{10}x_1(t) \oplus A_{11}x_1(t-1) \oplus B_1u_1(t) \\ &y_1(t) \geqslant C_1x_1(t), \end{aligned} \tag{2.24}$$

where the matrices A_{10}, A_{11}, B_1, C_1 are the parameters of synchronizations in the main system. The behaviour of the secondary system is given in (2.25)

$$\begin{aligned} &x_2(t) \geqslant A_{20}x_2(t) \oplus A_{21}x_2(t-1) \oplus B_2u_2(t) \\ &y_2(t) \geqslant C_2x_2(t) \\ &\quad \forall i, (\ni x_{1.j} \in \delta_i | x1.j(t-1)) \\ &\Longrightarrow x_{2.i} = x_{2.i}(t-1). \end{aligned} \tag{2.25}$$

The problems are then formulated in an MPC setting, and the solution of DES with partial synchronizations is given as follows.

Theorem 2.2. *Denote $\bar{y}_\epsilon$, the output induced by the input $\bar{u}_\epsilon$, defined by $\bar{u}_\epsilon(\tau) = \epsilon$ for $t+1 \leqslant \tau \leqslant t+T$ and assume that $r_j(\tau) \in \mathbb{N}_0$ for $t+1 \leqslant \tau \leqslant t+T$. Then, the unique solution of the MPC problem, denoted $\bar{u}_{opt}$, is given by*

$$\bar{u}_{opt}(\tau) = \bigwedge_{\tau \geqslant j \geqslant t} \tilde{v}(j) \quad \text{for } t+1 \leqslant \tau \leqslant t+T$$

The application of partial synchronizations can be found for instance in transportation and supply chain where several sub-systems are inter-connected into a single main system.

In this sub-chapter we have presented some preliminaries in DES application in operations systems. Most of the research formulate DES in event-based systems, and solve with MPC settings. The detail discussion of DES is later presented in Chapter 4.

2.2.2 DES in container terminal operations systems

We will discuss some efforts to use DES in container terminal operations systems in this sub-chapter. A DES framework is used in [11] to model rail operations in container terminal. The railway lines are the internal transporters among sections in the container terminals such as berth, CY, and gate. These sections are denoted as set of queues, where the state variables are number of containers or equipment (i.e. cranes) waiting in lines.

The subset of the dynamic of the transfer operations is as follow

$$q_i^{M_1}(t+1) = q_i^{M_1}(t) + [a_i^M(t) - u_i(t)] \triangle t, i = 1, ..., M, t = 0, ..., T-1 \quad (2.26)$$

$$q_i^{M_2}(t+1) = q_i^{M_2}(t) + [u_i(t) - u_{M+i}(t)] \triangle t, i = 1, ..., M, t = 0, ..., T-1 \quad (2.27)$$

$$q_i^{S}(t+1) = q_i^{S}(t) + [a_i^S(t) - u_{2M+i}(t)] \triangle t, i = 1, ..., S, t = 0, ..., T-1 \quad (2.28)$$

$$q^R(t+1) = q^R(t) + [\sum_{i=M+1}^{2M+S} u_i(t) - \sum_{j=1}^{I} u_j^R(t)] \triangle t, t = 0, .., T-1, \quad (2.29)$$

where $\triangle t$ is the sample time. The modeling in [11] is not entirely dynamic. The DES formulation is only to show the interrelation among sections in the terminal, for instance stacking area (M_1, M_2), crane (R), and trucks (I), where q represents the queues and a is the container input flows from berth.

The tactical planning itself is done through static modeling to determine the capacity of the queues, and later solved by MILP technique. The sub-optimal solutions of the DES model are found by an MPC-based algorithm.

The two way modeling system can also be found in [89, 90]. In [89], firstly, integer linear programming (ILP) models provide the scheduling of three state processes in the terminal, namely QC, AGVs and ASCs. The decision variables of the ILP models are x_{ij}, y_{ij}, z_{ij} which represent the sequences of flow shop of the three kinds of equipment. The scheduling then triggers the dynamics in the controllers as follows

$$\dot{\mathbf{r}}(t) = g(\mathbf{r}(t), u(t)) \quad (2.30)$$

$$\dot{r_1}(t) = r_2(t) \quad (2.31)$$

$$\dot{r_2}(t) = u(t), \quad r_2(t) \in [v_{\min}, v_{\max}], \quad u(t) \in [u_{\min}, u_{\max}] \quad (2.32)$$

where the controllers are the position ($r_1(t)$), velocity ($r_2(t)$), and acceleration ($u(t)$) of the equipments that have been scheduled through the higher level ILP models.

The DES controllers in the lower level is formulated as follows

$$\mathbf{r}(k+1) = \begin{bmatrix} 1 & \triangle T \\ 0 & 1 \end{bmatrix} \mathbf{r}(k) \begin{bmatrix} 0.5 \triangle T^2 \\ \triangle T \end{bmatrix} u(k) = \mathbf{A}\mathbf{r}(k) + \mathbf{B}u(k) \quad (2.33)$$

$$J = \sum_{k=1}^{N_s} 0.5m(r_2(k))^2 \quad (2.34)$$

where (2.34) describes the goal of to minimize the total energy used to operate the equipments in the terminal. The Hamiltonian function is solved through Pontryagin's Minimum Principle [89]. A similar approach is found in [90], where a predictive controller is used to allocate the number of containers handled by the equipment in the terminal, namely QC, automated guided vehicle (AGV), and automated straddle carrier (ASC). The scheduling of QC, AGV, and ASC itself are obtained with OR-based techniques.

A DES approach to model operations in container terminal is studied in [2, 4]. The studies are on tatical level decision-making where percentage of equipment capacities are determined. A similar approach can also be found in [3], where the case is in supply chain, and optimum capacities are sought in the each node of the chain, namely factories, warehouses and retailers. The subset of dynamics in [4] is as follows

$$x_i^z(t+1) = x_i^z(t) + a_i^z(t) - \triangle T \mu_i^z(t), i = 1, 2, ..., N_z, z = b, p, r \tag{2.35}$$

$$x_i^y(t+1) = x_i^y(t) + \triangle T (\sum_{j=1}^{N_z} \mu_j^z(t) u_j^z(t) - \mu_i^y(t) u_i^y(t)) \tag{2.36}$$

$$\begin{aligned} u_7^y(t) = \min\{ & \frac{x_7^y(t) + \triangle T \mu_4^y(t) u_4^y(t)(1-\beta_1(t))}{\triangle T \mu_7^y(t)} \\ & \gamma(t)(1-u_4^y(t), u_5^y(t), u_{10}^y(t), u_{11}^y(t))\} \end{aligned} \tag{2.37}$$

where x represents the vectors of queues of containers in each type of equipment, namely QC, rubber-tyre gantry crane (RTGC), reach staker (RS) and rail-mounted gantry crane (RMGC). The control variable is represented by u, which is the percentage of servers allocated for the operations. In the research, (2.37) represents the re-handling process or also known as housekeeping.

As similar with [2, 4], a DES approach to model the dynamics of container handling is also discussed in [91]. There are three state variables where $x_p^{\text{quay}}(k)$, $x_{pq}^{\text{yard}}(k)$, and $x_p^{\text{land}}(k)$ are the remaining quantity of cargo p to be unloaded at the quayside, the remaining quantity of cargo p to be loaded in the yard-slot q, and the remaining quantity of accumulated cargo p to be loaded and the hinterland at event-time k. The dynamics are as follows

$$x_p^{\text{quay}}(k+1) = x_p^{\text{quay}}(k) - \sum_{q \in Q} \delta_{pq}^{\text{in}}(k) u_{\text{in}} \triangle T \tag{2.38}$$

$$x_{pq}^{\text{yard}}(k+1) = x_{pq}^{\text{yard}}(k) + \delta_{pq}^{\text{in}}(k) u_{\text{in}} \triangle T - \delta_{pq}^{\text{out}}(k) u_{\text{out}} \triangle T \tag{2.39}$$

$$x_p^{\text{land}}(k+1) = x_p^{\text{land}}(k) + \sum_{q \in Q} \delta_{pq}^{\text{out}}(k) u_{\text{out}} \triangle T \tag{2.40}$$

where $\delta_{pq}^{\text{in}}(k), \delta_{pq}^{\text{out}}(k) \in \{0, 1\}$ are the binary control variables to determine if the stacker/reclaimer will or will not do the operations of inbound/outbound cargo p at slot-yard q from event time k to $k+1$. The other control variables are $u_{\text{in}} \triangle T$ and $u_{\text{out}} \triangle T$ as the unloading/loading rate of the cranes, respectively.

In this sub-chapter, we have shown the brief review of DES application in container terminals. The development of DES models for the specific settings in this thesis will be discussed in Chapter 3 and 6.

2.2.3 Petri nets

We present brief review of Petri nets in this sub-chapter which is taken from [19]. Petri nets is a graphical tool to represent interrelations of events in a DES model. There are two main parts of Petri nets, transitions and places. Events which are represented by transitions occur after several conditions are satistifed. The informaton of those conditions are stored in places.

Definition 1. *A Petri net graph is a weighted bipartite graph*

$$(P, T, A, w)$$

where $P = \{p_1, p_2, ..., p_n\}$ and $T = \{t_1, t_2, ..., t_m\}$ are the finite set of places and transitions, respectively. $A \subseteq (P \times T) \cup (T \times P)$ is the set of arcs from places to transitions and from transitions to places, where usually an arc is represented by (p_i, t_j) or (t_j, p_i). $w : A \to \{1, 2, 3, ...\}$ is positive integer weight function of the arcs. The set of input/output places to/from transition are repsented by $I(t_j) = \{p_i \in P : (p_i, t_j) \in A\}$ and $O(t_j) = \{p_i \in P : (t_j, p_i) \in A\}$, respectively.

Example 2.1. A Petri net graph as shown in Figure 3.1 is defined by

$$P = \{p_1, p_2\} \quad T = \{t_1\} \quad A = \{(p_1, t_1), (t_1, p_2)\} \quad w(p_1, t_1) = 2 \quad w(t_1, p_2) = 1$$

The input and output are $I(t_1) = \{p_1\}$ and $O(t_1) = \{p_2\}$, respectively. There are two places p_1 and p_2 in the Petri nets, which are represented by the two circles. The transition t_1 is indicated by the bar. The two input arcs from p_1 indicate the weight, thereby explains $w(p_1, t_1) = 2$.

One of the Petri nets' goals is to model the dynamic in the DES, therefore the it is able to capture the state transitions. To be enabled, the number of tokens in p_i has to be at least as large as the weight of the arc (p_i, t_j).

Definition 2. *A transition $t_j \in T$ i a Petri net is enabled if*

$$x(p_i) \geqslant w(p_i, t_j) \ \forall p_i \in I(t_j)$$

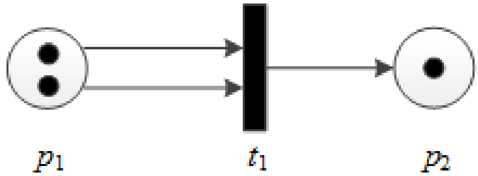

Figure 2.1: *Petri net graph for Example 2.1 where* $\mathbf{x} = [2, 1]$

The control input in a Petri net is defined by a vector $\mathbf{u} = [0, ..., 0, 1, 0, ..., 0]$ where 1 means that the $j-$transition is being performed, where $j \in \{1, ..., m\}$. An $m \times n$ incidence matrix $\mathbf{A}$ is defined from the net flows of input and output as its (j, i) entry as $a_{ji} = w(t_j, p_i) - w(p_i, t_j)$. Therefore, given an initial state $\mathbf{x}_0$, the sequences of the state is

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, t_k) = \mathbf{x}_k + \mathbf{u}_k \mathbf{A}.$$

We have briefly discussed the concepts of Petri nets in this sub-chapter. In Chapter 3 we will apply Petri nets to our case in BCAP.

2.3 Model predictive control

In Sub-Chapter 2.2 it is shown that MPC is a common approach to solve DES models [4, 11, 70, 89, 90, 91]. MPC is suitable because of its nature to seek the solution step per step, where the steps correspond to events in DES. The MPC in general consists of prediction model, objective function, and algorithm to obtain control law [15].

The prediction model has to represent the system that is being analysed, and for instance, the system can be represented as follow in a state-space form

$$x(t) = Mx(t-1) + Nu(t-1) \tag{2.41}$$

$$y(t) = Qx(t) \tag{2.42}$$

where x is the state, M, N, Q are the matrices of the system, and u is the control input, respectively. The prediction for the system's model as given in (2.41)-(2.42) is as follow

$$\begin{aligned}\hat{y}(t+k|t) &= Q\hat{x}(t+k|t)\\ &= Q[M^k x(t) + \sum_{i=1}^{k} M^{i-1} N u(t+k-i|t)].\end{aligned} \tag{2.43}$$

The prediction model is the fundamental in MPC which is used to predict the future outputs $\hat{y}(t+k|k)$ (value of variables at time $t+k$ calculated at time t) based on known predicted states $\hat{x}(t+k|k)$, and future control inputs $u(t+k|t), k = 0, ..., N-1$, where the prediction is performed for a prediction horizon N.

The objective function of the predicted model in (2.43) is a cost function used to evaluate the optimal control law. The objective function can be generalized as follows

$$J(N_{\text{min}}, N_{\text{max}}, N_u) = \sum_{j=N_1}^{N_2} \delta(j)[\hat{y}(t+j|k) - z(t+j)]^2 + \sum_{j=1}^{N_u} \lambda(j)[\triangle u(t+j-1)]^2. \tag{2.44}$$

The future output $\hat{y}(t+j|k)$, which is the value of the input at time $j+k$ calculated at time k should follow an external reference signal z and sample time $\triangle u$ which represents the control effort. The parameters of the objective/cost function are N_{min} and N_{max} which are the minimum and maximum cost horizons and N_u is the control horizon which is not necessarily has to be the same with N_{max}. The last parameters are $\delta(j)$ and $\lambda(j)$ which if necessary can be used to represent future behaviour. For instance, the exponential weight of $\delta(j)$ during the horizon is

$$\delta(j) = \alpha^{N_{\text{max}}-j} \tag{2.45}$$

where α is the exponential parameter whose value is between 0 and 1.

The control law is obtained by selecting $u(t+k|k)$ which minimizes the objective function in (2.44). The optimal control inputs are acquired by calculating the values of the predicted outputs $\hat{y}(t+k|k)$ where the current value of inputs and outputs, as well as the future inputs are considered. Some MPC algorithm discussed in [15] are dynamic matrix control, model algorithmic control, and predictive functional control. Some algorithms have been developed specifically to solve the prediction model of a DES. For an instance in [70], the control law is obtained by using some techniques from mixed-integer linear programming (MILP).

Chapter 3

DES modeling and model predictive algorithm for integrated BCAP

Chapter 3

DES modeling and model predictive algorithm for integrated BCAP

We study in this chapter the problem of integrated berth and quay crane allocation (I-BCAP) in general seaport container terminals and propose model predictive allocation (MPA) algorithm and preconditioning methods for solving I-BCAP. Firstly, we explain the generic dynamical model of berthing process for multiple berthing positions and multiple quay cranes. The proposed model predictive allocation algorithm and the corresponding pre-conditioning steps are presented are later discussed. Afterwards, we present the simulation setup and results. The field experimental setup and results are presented in the simulation sub-chapter. Finally, some concluding remarks are provided in the discussion.

3.1 Introduction

An important process in the terminal operations is the seaside operations-level decision making for the berth and quay cranes allocation. The assignment of berth positions and quay crane (QC) to incoming ships for handling their cargo plays an important role in minimizing the turnaround time. We refer interested readers to the papers [8] and [18] for an extensive review on the berth allocation problem and its current available solutions.

In the current literature, there are mainly two classes of berth allocation (BAP) problems which are based on the way they model the ship arrivals. These classes are the static BAP and the dynamic BAP, which is known as the DBAP as reviewed recently in [8] and [18]. In the former problem, as presented in [9], [72], [28], and [79], the entire arriving ships are assumed to have arrived at the port when the planning for the entire time interval is being made. Hence the berth allocation problem is solved based on a static set of ships' arrival time. First come first served (FCFS) rule, which is the most common method in allocating berth positions and cranes, is also based on the same assumption. It has been known that the FCFS method, which is simple and is easily adopted to the incoming ships, is not always efficient and applicable [39]. For instance, the FCFS can not be used if there is priority in seaport service where some ships can have higher priority than the others.

As opposed to the static set of ships' arrival time, DBAP takes into account the uncertainty in the arrival time of the ships within the planning horizon [30], [39], [40], [52], and [37]. Although the ships' arrival time is allowed to vary (hence the "dynamic" term is used), the berthing allocation method in these works is based on recasting the time-varying problem to a (mixed-integer) linear programming where the uncertainties is defined as stochastic constraint; similar to the ones used in [30]-[52]. As another example, in [39], the model of the berthing process comprises of a set of linear equations (without dynamics) and inequalities that describe physical bounds as well as uncertainty variables which represent the variation of the ships' arrival time. The model in [39] does not incorporate dynamical equations that describe the dynamics of the berthing operations and does not use the available real-time information of the arriving ships. Consequently the resulting allocation is conservative as it has to deal with the prescribed uncertainties on the ships' arrival and it does not feed back the real-time factual information of the arriving ships. For enabling a real-time allocation method that can handle a dynamically changing environment, a simple (yet useful) dynamical model is needed that can capture the essential elements in the terminal operations dynamics and be applicable for the development of optimal predictive allocation methods.

A closely related problem to BAP in the terminal operations is the crane allocation problem (CAP). While BAP is related to the allocation of incoming vessels to specific berth positions, CAP deals with the allocation and scheduling of QC to the already-assigned berthing ships. Until now, there have been works that focus on the integration of the BAP and CAP, such as in [20], [71], [40], [93], and [67] which is known as berth and crane allocation problem (BCAP). But, there are two main common limitations in the current literature that leads to sub-optimal solution in practice. The first limitation is related to their inability for handling real-time factual information that differs from the apriori information used for the planning, as discussed before. The second limitation is that they solve BAP and CAP in separate (but sequential) steps, due to complexity in the problem. The first step is solving the BAP without considering the CAP. The next step is allocating QC to the already-assigned ships.

As reviewed in [18], the BAP falls into NP-hard problems because of its complexity. The complexity itself is caused by the dimension of the problem i.e. the number of ships, the number of berth positions, and the number of quay cranes. One of the popular methods in solving the NP-hard problems, including the BAP, is genetic algorithm (GA) [18]. The GA allows flexibility for its users to solve the original problem through GA specific algorithm. The GA is employed in [20] and [40]. We will use a GA-based method as a benchmark for our optimization algorithm. Another technique to solve the BAP is Tabu search as in [71] where the objective function is to minimize the housekeeping cost that is affected by the resulting ship schedule. While in [93], Lagrangian relaxation is used.

For the CAP, the same techniques as mentioned above are not always used. This is due to the fact that the BAP and CAP are not solved simultaneously. Therefore, the method used to solve BCAP can vary. For instance, in [40], GA is used to solve the BAP, but the same method cannot be applied to solve the CAP because of GA limitation. Therefore, in [40] the CAP is modified into a maximum flow problem-based algorithm. While in the works of [20], [71], and [93], GA, a mixed-integer linear programming (MILP) and sub-gradient method are used, respectively.

A recent work on the integrated berth and quay-crane allocation is the paper [37]. In [37], the authors propose the use of Particle Swarm Optimization, which is another nature-inspired computational tools, to solve the integrated allocation problem. Similar to the setting in [40], as well as, other existing methods for the integrated BCAP as discussed before, the "dynamics" refers to the uncertainty in the parameters (which is the number of allocated QC in [37]) and is not suitable for dealing with real-time information. The inclusion of event-based berth plan which is able to incorporate the changing in ship arrivals is one of the main contribution of [37].

The non-robustness and non-adaptiveness of the above mentioned approaches to the dynamically changing environment has led to the wide adoption in real-life condition of a very simple heuristic approach, that is a combination of the first-come first-served strategy for the berthing allocation and of the density-based strategy for the quay crane allocation.

In order to handle such dynamic environment (including the real-time information on arriving ships), we propose a dynamical modeling framework using a discrete-event system (DES) formulation that describes both the real-time and continuously changing set of ship arrivals at any given time, as well as, the discrete-event dynamics during the berthing and loading/unloading process. The DES formulation fits better to terminal operations than the usual periodic discrete-time systems description since there is aperiodicity in the ships' arrival time and the operations' time among different berthing positions is usually asynchronous. In order to handle such dynamic environment (including the real-time information on arriving ships), we propose a novel real-time integrated berthing and crane allocation method which is based on model predictive control principle and rolling horizon implementation.

We also evaluate the performance of our model predictive allocation strategy using: (i). extensive Monte-Carlo simulations using realistic datasets; (ii). real dataset from a container terminal in Tanjung Priuk port, Jakarta, Indonesia; and (iii). real life field experiment in the aforementioned container terminal. To the best of our knowledge, the latter contribution on the real life field experiment provides an important insight to the performance of novel allocation method in reality which is typically not reported in literature.

3.2 Dynamical modeling of berthing process

We explain a generic dynamical model of berthing process in general seaport container terminals. First, in order to simplify the presentation, let us consider a simple berthing problem where there is only one berthing position and one QC. In this particular case, the decision variable is the berthing ship that is chosen from the set of ships-ready-to-be-berthed. In the generalization of this simple problem to the multiple berthing positions and multiple QC, we consider also the number of QC per berthing position as an additional decision variable. We summarize the notations in our modeling framework in Table 3.1.

3.2.1 The dynamic modeling of a simple berthing process

Before we start describing the berthing process dynamics, let us briefly recall the concept of DES which is a class of dynamical systems as expounded in [19]. Generically, DES are characterized by a *discrete* set of state space whose state transition is driven by (asynchronous discrete) events over time. Such class of systems encompasses systems described by automata and petri nets and it includes queueing systems, traffic systems, communication systems, etc. We refer interested readers to the book [19].

The DES is usually depicted with a Petri net [19]. The Petri net for the simple berthing process is provided in Figure 3.1. There are three kind of events which represent the infrastructure and equipment in berthing process, namely ships, berth positions and QCs. The events related to ship arrivals are S_b, S_r, S_d which represent conditions that a ship needs berth, a ship is ready for loading or unloading, and a ship is ready to depart, respectively. The events related to berth positions are B_s, B_f which are information telling that a berth position is ready and a berth operation has just finished, respectively. The events related to QCs are QC_s, QC_f which are the conditions that a QC is ready and a QC has departed, respectively. Two tokens which are available in the Petri net are u and v, the control variables of ship allocation to berth positions and number of QC allocated to a berthed ship, respectively. There are three transitions in the Petri net, $t_{\text{bst}}, t_{\text{qcs}}, t_{\text{fin}}$, which are the berth starting time, the QC starting time, and the berth/QC finishing time. Those three transitios will be the state variables in the next-discussed DES model for a simple berthing process.

Using such DES formalism, let us introduce the event time $k \in \mathbb{N}$ as a discrete sequence of events that corresponds to every initiation of a berthing process (at any berthing position). Thus, each event time k is related one-to-one to a unique time instance when a berthing process commences and such relation is denoted later by $t_{\text{bst}}(k)$. We denote $\mathcal{S}(k)$ as the set of arriving ships to seaport at event time $k \in \mathbb{N}$. Here, arriving ships refer to all ships that have already reported to the port on

Table 3.1: *List of mathematical notations*

Notation	*Description*
	Decision variables
$u(k)$	a control variable of the ship to be berthed chosen from the set $\mathcal{S}(k)$
$v_b(k)$	a control variable of the number of QC allocated to the $b-$th berth position
	Parameters
μ_a	Measure defined on elements of $\mathcal{S}(k)$ that gives the arrival time of the element(s)
μ_o	Measure defined on elements of $\mathcal{S}(k)$ that gives the operations time of the element(s)
	State variables
$t_{\text{bst}}(k)$	the state variable of berth starting time for the simple BCAP
$t_{\text{qcs}}(k)$	the state variable of QC operations time for the simple BCAP
$t_{\text{fin}}(k)$	the state variable of berth finishing time for the simple BCAP
$z_b(k)$	the state variable of berth starting time of the b-th berth position for the complex BCAP
$y_b(k)$	the state variable of remaining operations time of the b-th berth position for the complex BCAP
$x_b(k)$	the state variable of finishing time of the b-th berth position for the complex BCAP
	Sets and indices
k	event time
$\mathbb{N}$	the set of natural numbers
$\mathcal{S}(k)$	a dynamic set of arriving ships to seaport at event time k
$\mathcal{S}_i(k)$	The $i-$th element (ship) in the set $\mathcal{S}(k)$
$\mathcal{B}$	The discrete set of berthing positions
$\|\mathcal{B}\|$	The cardinality of the set $\mathcal{B}$
$\mathcal{Q}$	The discrete set of quay cranes
$\|\mathcal{Q}\|$	The cardinality of the set $\mathcal{Q}$
j	Index of the first earliest available berth position
b	Index of the other berth positions, where $b \neq j$
N	The planning horizon
M	The dimension of $\mathcal{S}(k)$ where $M > N$

their incoming. We denote the $i-$th element of $\mathcal{S}(k)$ by $\mathcal{S}_i(k)$. For instance, using the actual set of ships from our experimental dataset (Table XXX), the set $\mathcal{S}(k)$ can be $\mathcal{S}(k) = \{$"Berlian", "Fatima", "Meratus I"$\}$ and consequently $\mathcal{S}_2(k)$ refers to "Fatima".

Associated to $\mathcal{S}(k)$, we define two different measures, μ_a and μ_o which corre-

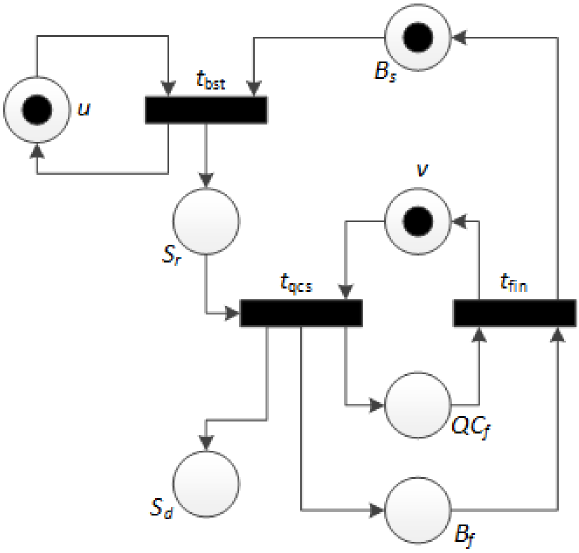

Figure 3.1: *A Petri net for a simple berthing process.*

spond to the arrival time and operations time, respectively. As an illustration using the above example of $\mathcal{S}(k)$, $\mu_a(\mathcal{S}_2(k))$ refers to the arrival time of the ship "Fatima", as it is the second element in the set. Similarly, $\mu_o(\mathcal{S}_1(k))$ and $\mu_o(\mathcal{S}(k))$ refer to the operation time of the ship "Berlian" and to the total operation time of all ships in $\mathcal{S}(k)$, respectively. Here, the measure $\mu_o(\mathcal{S}_i(k))$ refers to the i-th ship operations time for unloading and loading the entire containers by a single QC. This choice will be useful later when we take the number of QC as another decision variable. The total operations time itself depends on the number of containers in a ship, represented by twenty-feet equivalent unit (TEU), number of QC assigned to the ship, and quay crane capacity, that is usually in TEU per hour.

As one of the decision/input variables, we denote $u(k)$ as the ship to be berthed chosen from the set $\mathcal{S}(k)$. By defining $t_{\text{bst}}(k)$ as the berth starting time, $t_{\text{qcs}}(k)$ as the QC operation time and $t_{\text{fin}}(k)$ as the berth finishing time, the state space equation of the berthing process at a given event time k can be given by

$$t_{\text{bst}}(k) = \max\{\mu_a(u(k)), t_{\text{fin}}(k-1)\} \tag{3.1}$$
$$t_{\text{qcs}}(k) = \max\{t_{\text{bst}}(k), t_{\text{fin}}(k-1)\} \tag{3.2}$$
$$t_{\text{fin}}(k) = t_{\text{qcs}}(k) + \mu_o(u(k)) \tag{3.3}$$
$$\mathcal{S}(k) = \mathcal{S}(k-1)\backslash u(k) \cup \mathcal{U}(k).$$

The state space for $t_{\text{bst}}, t_{\text{qcs}}$ and t_{fin} is $\mathbb{N}$ and for $\mathcal{S}$ is the discrete set of all (admissible) arriving ships to the port. One can further simplify (3.1)-(3.3) into only two state equations as follows

$$x(k) = \max\{\mu_a(u(k)), x(k-1)\} + \mu_o(u(k)) \tag{3.4}$$
$$\mathcal{S}(k) = \mathcal{S}(k-1)\backslash u(k) \cup \mathcal{U}(k) \tag{3.5}$$

where the state $x(k)$ denotes the finishing time $t_{\text{fin}}(k)$ and $\mathcal{U}(k)$ is the set of new arriving ships that comes at the event time k. In (3.5), we have that the set of incoming ships at each time k is the same set from the previous time step modulo (c.f. the symbol $\backslash$) the ship that has been taken out from the set for berthing (e.g., the ship $u(k)$) and is added by (c.f. the symbol $\cup$) a (possible) set of incoming ship(s) $\mathcal{U}(k)$ arriving at time k.

Remark 3.1. Compared to the existing literature ([2], [4],), our dynamical modeling framework for the berthing problem has resulted into state equations involving set dynamics (c.f. (3.5)). The analysis of such dynamics in the context of seaports interconnection is not trivial and we will not treat this issue in this chapter. However, we still take into account the set dynamics in our optimization problem later.

3.2.2 Generalization to the multiple berthing positions and multiple QC

In this sub-chapter, we will extend the dynamical modeling of a simple berthing process in (3.4)-(3.5) into multiple berth positions and multiple QC. For defining the domain of our decision variables, we denote the set of discrete berthing positions by $\mathcal{B}$ where $|\mathcal{B}|$ is the total number of positions and we denote $\mathcal{Q}$ as the set of QC where $|\mathcal{Q}|$ defines the total number of available QC. Note that we consider only discrete berthing positions in this chapter. We denote $x_b(k)$ as the finishing time of the b-th berth position at the event time k, for all $b = 1, .., |\mathcal{B}|$.

In this setting, every time an assigned QC has finished an operation at a particular berth position, a new berthing process will commence where a new ship needs to be allocated and berthed. This means that this is an event-based dynamical model, since k is triggered from a completed event from previous $k-1$. As before, the finishing time for the b-th position will be denoted by $x_b(k)$.

To capture the complexity, in contrast to the dynamical modeling for the simple berthing process, we need at least three state variables for every berthing position that record the starting time t_{bst}, the estimated finishing time t_{fin} and the remaining operations time. For every $b-$ th berth position, we denote new state variables $z_b(k)$ as the berth starting time and $y_b(k)$ as the remaining operations time. We define an additional control variable $v_b(k) \in \mathbb{N}$ which is the number of QC allocated to the b-th berth position at step k and we assume that the total number of QC is constant during the entire operations. For every event time k, the dynamics is given as follows. By letting

$$j = \arg\min_{b}[x_b(k-1)] \tag{3.6}$$

the dynamics of the j-th berth position is given by

$$z_j(k) = \max\{x_j(k-1), \mu_a(u(k))\} \tag{3.7}$$

$$y_j(k) = \mu_o(u(k)) \tag{3.8}$$

$$x_j(k) = z_j(k) + \frac{y_j(k)}{v_j(k)} \tag{3.9}$$

and the dynamics of the other berth positions $b \neq j$ is given by

$$z_b(k) = \begin{cases} x_j(k-1) & \text{if } x_j(k-1) > z_b(k-1) \\ z_b(k-1) & \text{otherwise} \end{cases} \tag{3.10}$$

$$\begin{aligned} y_b(k) = {} & y_b(k-1) \\ & - [z_b(k) - z_b(k-1)]v_b(k-1) \end{aligned} \tag{3.11}$$

$$x_b(k) = z_b(k) + \frac{y_b(k)}{v_b(k)} \tag{3.12}$$

$$\sum_{b=1}^{|\mathcal{B}|} v_b(k) = |\mathcal{Q}| \tag{3.13}$$

$$\mathcal{S}(k) = \mathcal{S}(k-1) \backslash u(k) \cup \mathcal{U}(k). \tag{3.14}$$

Equation (3.6) refers to the earliest available berth position (denoted by j) based on the finishing time of each berth position at the previous event time $k-1$. The state variable z in (3.7) and (3.10) defines the berthing time for every berth position at every event time k. The state variable y in (3.8) and (3.11) is the remaining operations time at every berth positions. Finally, as in the simple berthing process, the state variable x describes the estimated finishing time for every berth position based on the allocated QC given by the input variable v. The equation (3.13) ensures that the total number of QC assigned to all berth positions is the same as the total number of available QC.

An illustration of the complex berthing process is provided in Figure 3.2. In this figure, two berthing positions are considered namely, the $j-$ th and $b-$ th berth positions. As discussed above, at the event time k, the $j-$ th and the previously allocated QC become available since the berthing operations has been completed. Hence, a new allocation process is started where a new ship $u(k)$ is allocated to

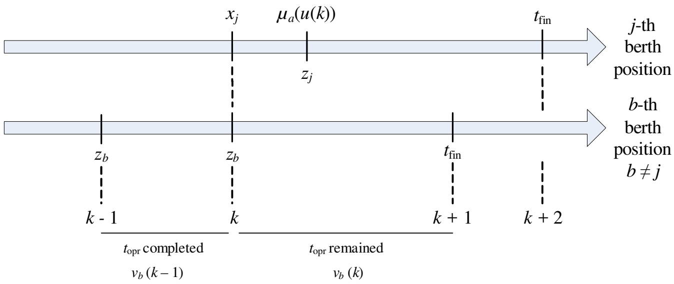

Figure 3.2: *Illustration of the discrete-event systems in the berthing process with multiple berthing positions and multiple QC. It shows the dynamical relationship between two berthing positions at each event time k. The variables x_j, z_j, x_b and z_b are as defined in* (3.7)-(3.12).

the vacant berth position. Simultaneously, the QC can be redistributed during the event time k which can result in the in the changes to the remaining operation time for the other berthing positions (e.g. the $b-$th berth position in this figure). We can notice from this figure that the berthing process is an asynchronous process where we cannot define a periodic time sampling as commonly used for modeling dynamical systems. Instead we use the event time k.

From the state equation (3.9) and (3.12), one can deduce that the domain of the state space is

$$\{(x,y,z) \in \mathbb{R}_+^{|\mathcal{B}|} \times \mathbb{R}_+^{|\mathcal{B}|} \times \mathbb{R}_+^{|\mathcal{B}|} \mid x_i > z_i, i = 1, \ldots, |\mathcal{B}|\}.$$

It can also be seen from (3.7) and (3.10) that the state trajectories of the berthing time z is monotone non-decreasing.

3.3 Model predictive allocation strategy

In this sub-chapter, we propose a model predictive allocation (MPA) strategy for solving the I-BCAP where we modify the standard model predictive control approach to our discrete-event systems formulation presented in the previous sub-chapter. In our proposed approach, the DES model of berthing process as presented in Sub-Chapter 3.2, is used to optimize the berthing control input u and quay cranes control input v for a finite events horizon in the future and subsequently, the solution for the current event is implemented and the horizon is rolled by one event time further.

In the following, we discuss first the cost functions that will be optimized by our proposed model predictive allocation algorithm, as well as, be used for comparing our methods with existing approaches. Subsequently, we present our proposed algorithm and discuss two preconditioning steps for tractably solving the optimization problem.

We explain the structure of the problem. The models given in the previous sub-chapters are nonlinear with nonlinear dynamics. The nonlinearity of the dynamics is resulted from the asynchronous dynamics (operations) among berth positions b at each time step k. Since the problem is nonlinear, the MPA is not necessarily convex [74]. The problem's state variables are continuous which represent the starting time, remaining operations time, and finishing time $(z_b(k), y_b(k), x_b(k))$. The control variables of the problem are discrete, first the ship allocation to berth $(u(k))$ and second, the number of QC allocated to each ship $(v_b(k))$. As we have discussed in the previous sub chapters, the difference between our approach (DES-based model) with the OR-based model is the inclusion of the dynamics of the input sets. Therefore, the setting of the inputs is more incline to real-time (such as real-time arrival of the ships $(\mathcal{S}(k))$) than the deterministic/probabilistic aspects of the sets, which in the state-of-the-art research is usually facilitated by some statistical function [37, 39, 40].

3.3.1 Objective functions

The cost function of the berthing process, whether for the simplest one or for the multiple berth positions and QC, is based on both the operations cost and waiting cost. These two cost components are closely related to the time that the ships spent at the assigned berth positions for completing their berthing process. The cost function which will be described in this sub-chapter is nonlinear and this add the complexity of the MPA.

The operations cost is the cost of operating the QC that are allocated to a particular ship. Let us denote the operating cost of a QC unit (Euro/hour) by C_o. The operational cost between the step $k-1$ to step k is then defined by C_o multiplied by the time needed for unloading/loading containers from/to a ship. In other words, it is given by $C_o \mu_o(u(k))$ and

$$C_o\Big([x_j(k-1) - z_j(k-1)]v_j(k-1) \\ + \sum_{b \neq j} [z_b(k) - z_b(k-1)]v_b(k-1)\Big), \quad (3.15)$$

for a single QC and multiple QC case, respectively, where j is as in (3.6).

On the other hand, the waiting cost is associated to the total time that a particular ship spends at seaport, i.e. from the time it arrives until it leaves after the assigned QC have completed the operations. It may happen that the particular ship has to wait after its arrival, since all berth positions are occupied. We denote

C_w the waiting cost of a ship unit (Euro/hour).

For simplicity of notation, for the multiple berth positions and multiple QC, we denote the earliest available berth position at event time k by $w(k)$ which is defined by

$$w(k) = \arg\min_b [x_b(k-1)]. \tag{3.16}$$

Based on this description, the cost functions (defined from the step k with horizon N) for the simple berthing process and for that with multiple berth positions and multiple QC are given by

$$J(x,u) = \sum_{n=k}^{k+N} \mu_o(u(n))C_o + [x(n) - \mu_a(u(n))]C_w \tag{3.17}$$

and

$$\begin{aligned} &J(x,y,z,u,v) \\ &= \sum_{n=k}^{k+N} \Big([x_{w(n)}(n-1) - z_{w(n)}(n-1)]v_{w(n)}(n-1)C_o \\ &\quad + \max\{x_{w(n)}(n-1) - \mu_a(u(n)), 0\}C_w \\ &\quad + \sum_{b \neq w(n)} [z_b(n) - z_b(n-1)](v_b(n-1)C_o + C_w) \Big) \end{aligned} \tag{3.18}$$

respectively.

In this formulation, we have explicitly defined the cost function as a function of state variables x, y and z and of input variables u and v that satisfy (3.6)–(3.14). Note that since $x > z$ (as remarked after (3.14)), the cost function J in (3.18) is positive definite.

3.3.2 Model predictive allocation algorithm

Let us now describe our model predictive allocation (MPA) algorithm. For every event time k, we denote $\hat{z}(l)$, $\hat{y}(l)$, and $\hat{x}(l)$, with the integer $l \geqslant 0$, as the predicted state variables at event time $k+l$ based on known/measured state variables at current event time k. Using this notation, $\hat{z}(0) = z(k), \hat{y}(0) = y(k)$ and $\hat{x}(0) = x(k)$. Also, $\hat{z}(-1) = z(k-1), \hat{y}(-1) = y(k-1)$ and $\hat{x}(-1) = x(k-1)$. Similar to (3.16), we define $\hat{w}(l) = \arg\min_b \hat{x}_b(l-1)$. Using these notations, the MPA algorithm for the berth and quay cranes is given as follows where we use the event horizon $\{0, 1, 2, \ldots N\}$ with $N > 0$ for the predictive state variables $(\hat{z}, \hat{y}, \hat{x})$, which is equivalent to the rolling event horizon $\{k, k+1, k+2, \dots, k+N\}$ for the state variables (z, y, x). For generality, we will describe the algorithm for the multiple berthing positions and multiple QC case. It is straightforward to adapt the algorithm for the simple berthing process.

MPA Algorithm

1. For a new event time k, we update the current state variables as in (3.6)–(3.14).

2. Solve the following optimization problem

$$\min_{\hat{u},\hat{v}} J(\hat{x}, \hat{y}, \hat{z}, \hat{u}, \hat{v})$$

subject to

$$\hat{z}_{\hat{w}(l)}(l) = \max\{\hat{x}_{\hat{w}(l)}(l-1), \mu_a(\hat{u}(l))\} \tag{3.19}$$
$$\hat{y}_{\hat{w}(l)}(l) = \mu_o(\hat{u}(l)) \tag{3.20}$$
$$\hat{x}_{\hat{w}(l)}(l) = \hat{z}_{\hat{w}(l)}(l) + \frac{\hat{y}_{\hat{w}(l)}(l)}{v_{\hat{w}(l)}(l)} \tag{3.21}$$

and for every $b \neq \hat{w}(l)$

$$\hat{z}_b(l) = \begin{cases} \hat{x}_{\hat{w}(l)}(l-1) & \text{if } x_{\hat{w}(l)}(l-1) > z_b(l-1) \\ z_b(l-1) & \text{otherwise} \end{cases} \tag{3.22}$$

$$\begin{aligned} \hat{y}_b(l) &= \hat{y}_b(l-1) \\ &\quad - [\hat{z}_b(l) - \hat{z}_b(l-1)]\hat{v}_b(l-1) \end{aligned} \tag{3.23}$$
$$\hat{x}_b(l) = \hat{z}_b(l) + \frac{\hat{y}_b(l)}{\hat{v}_b(l)} \tag{3.24}$$
$$\sum_{b=1}^{|\mathcal{B}|} \hat{v}_b(l) = |\mathcal{Q}| \tag{3.25}$$
$$\hat{\mathcal{S}}(l) = \hat{\mathcal{S}}(l-1) \backslash \hat{u}(l), \tag{3.26}$$

where $l = 0, 1, N$, $\hat{u}$ and $\hat{v}$ are the predicted control input within the horizon.

3. Implement the berthing control input $u(k) = \hat{u}(0)$ and the quay crane control input $v(k) = \hat{v}(0)$.

4. Increment the event time k by one and return to 1).

3.3.3 Preconditioning steps

For solving the optimization problem in Step 2 of the MPA algorithm above in the event horizon $\{0, 1, \ldots N\}$, we need to find an optimal berthing control input $\hat{u}(0), \ldots, \hat{u}(N)$ from the space of $\mathcal{S}(k)$ and an optimal quay cranes control input $\hat{v}(0), \ldots, \hat{v}(N)$ from the available number of quay cranes $|\mathcal{Q}|$. If $|\mathcal{S}(k)|$ and $|\mathcal{Q}|$ are small, one can easily solve the optimization problem by using an exhaustive search. When they are very large (with, for instance, $M := \dim \mathcal{S}(k) \gg N$), solving such combinatoric problem is NP-hard and one can use a heuristic method, such as, the Genetic Algorithm (a popular nature-inspired computational method and is used for solving I-BCAP in [40]), for finding a (sub)-optimal sequence of N ships that minimizes the cost function.

As an alternative to GA and HPSO for solving the aforementioned optimization problem, we propose a preconditioning step, called N*-level* FCFS, which is a quasi-exhaustive search that combines FCFS and exhaustive search approaches. The approach is detailed as follows.

N**-level First-Come First-Served** (N-level FCFS)

1. Order the set of ships-to-be-berthed at step k, $\mathcal{S}(k)$ based on the measure of the arrival time μ_a such that

$$\mu_a(\mathcal{S}_1(k)) \leqslant \mu_a(\mathcal{S}_2(k)) \leqslant \ldots \leqslant \mu_a(\mathcal{S}_M(k))$$

   holds where M is the dimension of $\mathcal{S}(k)$ and is assumed to be larger than the horizon N.

2. Pick the first N ships from the ordered set and denote such subset of ships as $\mathcal{D}(k)$.

3. Perform exhaustive search of optimal berthing control input $\hat{u}$ from $\mathcal{D}(k)$ and $\hat{v}$ from the available number of quay cranes that solves the optimization problem in the given event horizon as above.

The above N-level FCFS replaces the optimization step 2 in the MPA algorithm as given before. This combination will be referred to as MPA-FCFS method throughout this research. One can also propose another pre-conditioning step based on the measure of operations time μ_o that is closely related to the size or container density of the incoming ships. This method is called N**-level Heavy-First Light-Last** N-level HFLL. It has the same procedure with the (N-level FCFS), but, the ship ordering in the step 1 is now based on the measure of the operational time μ_o such that $\mu_o(\mathcal{S}_1(k)) \geqslant \mu_o(\mathcal{S}_2(k)) \geqslant \ldots \geqslant \mu_o(\mathcal{S}_M(k))$. The ordering ensures that the first N ships that is defined as $\mathcal{D}(k)$ in Step 2 will consist of the N heaviest ships from the current set of ships at the port $\mathcal{S}(k)$. The combination of N-level HFLL and the MPA algorithm will be referred to as MPA-HFLL method.

3.4 Simulation

To evaluate the efficacy of our proposed method, as well as, for comparing it with standard and state-of-the-art approaches, we firstly conduct numerical simulations as presented in this sub-chapter. Subsequently, we present experimental validation in the next sub-chapter, i.e., Sub-Chapter 3.5. The simulations comprise of two different setups. In the first simulation, we use an actual dataset obtained from a real container terminal. In the second one, we use large-scale realistically generated datasets to test the robustness of our proposed method through Monte-Carlo simulations.

3.4.1 Simulation setup

Let us describe the simulation setup for both types of datasets where we define how the data are gathered (or generated) and discuss the underlying assumptions. For the computer simulations, we use a standard personal computer with a 1.60 GHz Intel Core i7-720QM processor. The experimental data are stored in microsoft Excel and processed using VBA 2016 and Matlab version 2016a. All other data processing is done using Matlab version 2016a.

In both settings, we will compare our proposed approach with two different benchmark methods which are commonly used in practice. The first benchmark method is the combination of FCFS and density-based QC allocation (DBQA) which are also the current policies in the terminal considered in the field experiment. The DBQA allocates the QCs according to the container density of the entire ships that are currently berthed in the seaport. The second benchmark method is the GA-based solution as recently proposed in [40]. We remark that Hybrid Particle Swarm Optimization (HPSO) algorithm, which is another nature-inspired computational algorithm, has been used to solve the I-BCAP in [37]. In [37], it is reported that the performance of HPSO-based solution is comparable to the GA-based one. Therefore we compare our results with the original GA-based solution as in [40] and the HPSO-based method as in [37].

For the first method, FCFS allocates ships based on their arrival times. The first earliest arriving ship will be allocated before the second earliest and this process is recursed until the entire ships are allocated. While for the DBQA, it allocates the number of QC based on the density of the already-berthed ships. The density of a ship is defined as the proportion of the particular ship's load to the total load of the

Table 3.2: *A subset of an actual dataset from an Indonesian seaport in Jakarta used in the first simulation setup.*

No	*Ship's name*	*TEU*	*Arrival time*	*Operations time (QC min)*
1	"Berlian"	665	20-01-14 23:59	1,995
2	"Fatima"	713	20-01-14 23:59	2,139
3	"Meratus I"	750	21-01-14 2:20	2,250
4	"Sejahtera"	463	21-01-14 3:00	1,389
5	"Vertikal"	894	22-01-14 8:00	2,682
6	"T. Rejeki"	429	22-01-14 8:15	1,287
7	"Meratus II"	318	23-01-14 8:30	954
8	"Meratus III"	392	23-01-14 16:00	1,176
9	"Perintis"	368	23-01-14 23:00	1,104
10	"Meratus IV"	807	24-01-14 23:59	2,421

entire ships which berth during the same period of a event time k. The number of QC allocated to each ship is proportional to its density.

For the second method, the proposed method in [40] consists of two separate procedures for solving I-BCAP. The berth allocation part is solved using genetic algorithm (GA) following the same procedure as outlined in [57]. For details on both the GA-based and the HPSO-based I-BCAP methods, we refer interested readers to the Supplemental Material in [13] and the paper [37].

We use the terminal standard for the C_o, i.e. the cost spent by the terminal operator to operate QC for loading and unloading containers and it is approximately given by Euro 1,250 per hour. For the waiting cost of C_w, since there is no data available, we use information from the terminal, i.e. the estimated hourly cost spent by every ship to wait for the completed operation in the seaport, which is in our case is Euro 7,500 per hour [61]. As a benchmark, we refer into [71] which states that the delay cost for a 15,000 TEU ship is about a million Euro per day. Since the ships considered in our simulations are at most below 7,000 TEU, our estimation of C_w is appropriate.

Let us now discuss the two datasets that will be used in our simulation below.

First simulation setup

For the first simulation setup, we used an actual dataset from a seaport in Tanjung Priuk, Jakarta. The data is obtained from the smallest terminal in the seaport which consists of 2 berth positions and 7 QC with the same technical specification. The data is collected from the primary source of Pelindo II from which we get the permission to use the data for an academic purpose. The time period of the dataset is from 20 January 2014 to 31 January 2014. There are 28 incoming ships to the terminal whose loads range from 327 to 2,156 TEU.

Table 3.3: *The setting of simulation scenario.*

Scenario	*Mean*	*Std. dev.*	*Lower bound load (TEU)*	*Upper bound load (TEU)*
Light load	6.0	0.6	1,000	3,000
Normal load	5.5	0.5	3,000	5,000
Heavy load	5.0	0.4	5,000	10,000

A subset of the data for the first 10 ships is shown in Table 3.2. The measure μ_a and μ_o that we define in Sub-Chapter 3.2.1, can be obtained from the arrival time and from the operations time (in QC min.), respectively.

Second simulation setup

For the second simulation setup, we generate large-scale realistic datasets for evaluating the efficacy of our method via Monte-Carlo simulations. In total, there are 300 different datasets, representing a combination of various different terminal settings, as well as, ship loads which are generated as follows.

For each dataset, we generate a set of 50 arriving ships according to one of the following scenarios: light load, normal load, and heavy load. The first scenario is the case where incoming ships arrive to the seaport in sparse inter-arrival times, and the loads are not high. On the other hand, the heavy load scenario is the opposite situation of the light load scenario.

The ship arrivals data is generated based on log-normal distribution to avoid negative value of inter-arrival times. We obtain the parameters from real data as described in Sub-Chapter 3.4.3, where from the 29 ship arrivals, the log-normal mean and standard deviation are 5.96 and 0.63 hours, respectively. Based on interviews with the operators who work at the seaport of Tanjung Priuk from which the data were taken, the arrivals can be categorized as a light one [61]. We also generate the ships loads (TEUs) based on the uniform distribution. The basis to categorize ships loads is derived from common container vessels classifications. The feeder ships'capacity are usually up to 3,000 TEUs. The Panamax ships are up to 5,000 TEUs. While the Post-Panamax is a generation of ships that are able to carry 10,000 TEUs. The important assumption is each of berth positon can handle all kind of ships regardless of their sizes. Hence, we generate our parameters as follows in Table 3.3.

For each different ship load setting, we test our method with four different terminal settings. The number of berth positions in each setting are 2, 3, 4, 5, respectively, while the QC numbers are 5, 7, 9, and 11, respectively. In all of these types of terminals, the technical specifications of the berth and QC are all the same. Table 3.4 – 3.6 are subsets of realistically generated datasets that are used in the Monte Carlo simulation

Table 3.4: *A subset of a dataset of the first 10 of 50 arriving ships for the light load scenario.*

Ship index	*Arrival date*	*Arrival time*	*Load (TEU)*
1	1 March 2016	05:50 AM	1,829
2	1 March 2016	12:04 PM	1,952
3	1 March 2016	14:08 PM	2,891
4	1 March 2016	18:25 PM	1,750
5	1 March 2016	22:29 PM	1,674
6	2 March 2016	07:38 AM	1,149
7	2 March 2016	13:50 PM	1,709
8	2 March 2016	20:07 PM	2,079
9	2 March 2016	23:57 PM	1,697
10	3 March 2016	05:29 AM	1,478

Table 3.5: *A subset of a dataset of the first 10 of 50 arriving ships for the normal load scenario.*

Ship index	*Arrival date*	*Arrival time*	*Load (TEU)*
1	1 March 2016	02:26 AM	3,211
2	1 March 2016	07:17 AM	4,137
3	1 March 2016	09:17 AM	4,372
4	1 March 2016	14:08 PM	3,288
5	1 March 2016	18:59 PM	3,089
6	2 March 2016	02:29 AM	4,383
7	2 March 2016	06:16 AM	4,363
8	2 March 2016	11:58 AM	3,403
9	2 March 2016	15:04 PM	3,786
10	2 March 2016	21:47 PM	4,539

3.4.2 Benchmarking methods

We use two benchmarking methods from [37] and [40] as comparison to our proposed MPA algorithm. We modify the HPSO algorithm used in [37] to:

1. Estimate the working duration of a ship which includes the QC setup time and handling time. In the simulation, we assume the setup time is fixed while the handling time follows the formulation of operations time (μ_o).

2. Set general parameters values of n, m, q, i, C_1, C_2, C_3, ρ, α, β, t_s, t_m as in [18]. Based on those parameters, we initialize a population of particles and seed of random positions (X_i) which is used in the iteration.

3. Determine speeds (V_i).

4. Determine the positions of the swarm particles that represent the berth position and number of QC allocated.

Table 3.6: *A subset of a dataset of the first 10 of 50 arriving ships for the heavy load scenario.*

Ship index	*Arrival date*	*Arrival time*	*Load (TEU)*
1	1 March 2016	01:47 AM	9,607
2	1 March 2016	04:16 AM	9,362
3	1 March 2016	06:25 AM	6,127
4	1 March 2016	08:55 AM	5,300
5	1 March 2016	10:26 AM	5,540
6	1 March 2016	14:17 PM	8,713
7	1 March 2016	16:19 PM	8,929
8	1 March 2016	18:51 PM	5,712
9	1 March 2016	22:11 PM	5,881
10	2 March 2016	00:40 AM	5,818

5. Identify events, their event times, owners (ships), and types which is enumerated for the entire possible combinations.

6. Find the next event time $t(k)$.

7. Transform the swarm particles into rank sets that follow dynamic rank order values (DROVs).

8. Compare the fitness value (FV) of the events. Select the events with the highest FVs.

The second benchmark method is the GA-based I-BCAP algorithm from [40]. We modify the original algorithm to suit our problem settings as follows:

1. Set the number of chromosomes where each chromosome is represented as character strings and the length/digits of the strings is

$$|\mathcal{S}| + |\mathcal{B}| - 1 \tag{3.27}$$

   which is number of ready-to-be-berthed ships plus number of berth positions minus one. The $|\mathcal{B}| - 1$ character strings are used as separator so that the remaining character strings (associated to ships) can be grouped into $|\mathcal{B}|$ berth positions. For each group of ships, the order of the berthing is determined by the position of the character string in that particular group.

2. For initialization, randomize the sequence in each chromosome.

3. Following the formulation in (3.6)-(3.5), calculate the current objective

function which is given by

$$g = \sum_{k=1}^{N}([x_j(k-1) - z_j(k-1)]C_o. \tag{3.28}$$
$$+ \max\{x_j(k-1) - \mu_a(u(k)), 0\}C_w$$
$$+ \sum_{b\neq j}[z_b(k) - z_b(k-1)]C_o + C_w).$$

4. Transform the objective function g to a fitness value defined by

$$f = 1/(1 + \text{exp}(g/10,000)) \tag{3.29}$$

where f has a value range from 0 to 0.5.

5. Reproduction. This step is to copy each of individual chromosome with the probability that is proportional to the fitness values. Thus, a chromosome with a higher fitness value will have more copies at the next generation.

6. Crossover. After reproduced chromosomes constitute a new population, this step is performed to introduce new chromosomes (or children) by recombining current strings. The crossover operator is the 2-point crossover [40].

7. Mutation. We perform this step to introduce random changes to the chromosomes.

8. Repeat the steps 3-7 for a predetermined number of generations.

Based on the above ship sequencing result, a maximum flow algorithm is used to allocate the QC following the procedure in [40]. As in a standard maximum flow algorithm, a network is evaluated in order to find an optimum flow between two nodes in each edge. The network here is the ship schedule obtained from the BAP where the nodes/vertices represent the ships and the edges represent transfer link between two ships. There are two additional nodes which are the sink and the source nodes which serve as the start and end nodes. After solving the algorithm, the prior ship schedule from the BAP algorithm may be changed due to infeasibility of the crane allocation.

3.4.3 Simulation results

In this sub-chapter, we provide simulation results for the multiple berthing positions and multiple QC case.

Table 3.7: *Simulation result of the berth and quay crane allocation for multiple berth positions and multiple QC using our proposed MPA with N-level FCFS pre-conditioning step.*

N	*Allocation Strategy*	*Total cost MPA-FCFS*	*Total cost MPA-HFLL*
1	FCFS & DBQA	3,921,923	3,921,923
2	MPA-FCFS	3,828,642	3,883,197
3	MPA-FCFS	3,642,352	3,775,005
4	MPA-FCFS	3,547,157	3,619,357
5	MPA-FCFS	3,364,535	3,586,977
6	MPA-FCFS	3,230,941	3,434,669
7	MPA-FCFS	3,162,912	3,370,241
8	MPA-FCFS	3,115,403	3,308,427
	GA-based method	3,445,444	3,445,444
	HPSO-based method	3,246,945	3,246,945

First simulation results

Using the first simulation setup as presented before, we apply N-level FCFS and N-level HFLL pre-conditioning step to our proposed MPA method and the simulation results are shown in Table 3.7 where the event horizon length N is taken between 1 and 8. In these tables we provide also results using the two benchmark methods as described in Sub-Chapter 3.4.1.

As shown in Table 3.7, the total cost using our proposed MPA method monotonically decreases as the event horizon length N increases. The MPA with N-level FCFS and N-level HFLL with a horizon of 8 can already decrease the total cost of 20.56% and 15.64%, respectively, compared with the traditional FCFS and DBQA methods. In comparison to the benchmark from [40], the MPA method with 8-level FCFS and 8-level HFLL pre-conditioning result in cost reduction of 9.58% and 3.98%, respectively.

To show the effect of these various allocation methods, we present in Figure 3.3(a)-(d) the allocation results using the FCFS & DBQA, our proposed MPA-FCFS methos with $N = 8$, the HPSO-based I-BCAP method as in [37], and the GA-based I-BCAP method as proposed in [40], respectively.

The small box in the right-upper part of every ship's schedule box represents number of QC assigned to the particular schedule. We can observe from these figures that the different methods can produce different berthing and QC allocation. For instance, it can be seen from Figure 3.3(a) which shows the allocation result using FCFS & DBQA, that ”Meratus I” is berthed prior to ”Sejahtera” which conforms to the arrival data as given in Table 3.2. On the other hand, our proposed method, which gives up to 20% cost reduction as shown in Table 3.7, gives priority to ”Sejahtera” over ”Meratus I”. Surprisingly, the GA-based method yields a similar allocation strategy as the FCFS & DBQA, as shown in Figure 3.3(d), although there

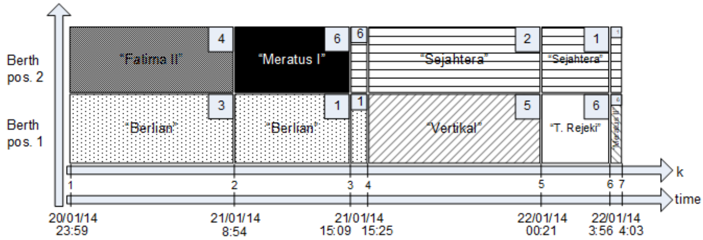

Figure 3.3: *The resulting berth and quay crane allocation for the first 7 event time using: (a). the de-facto FCFS and DBQA method; (b). our proposed MPA-FCFS method; (c). HPSO-based I-BCAP method as in [37]; and (d). GA-based I-BCAP method as proposed in [40]. The berth positions are shown in the vertical axis and the actual real time (and the event time k) is shown side-by-side in the horizontal axis.Each box represents allocated ship at different berth position where the label describes the ship's name and the number on the top-right corner of every box gives the allocated QC.*

are differences for the higher event time k not shown in the figures.

One can notice from Table 3.7 that as the event horizon length N increases, the complexity has also increased and resulted in longer calculation time per event time. The complexity will also increase as the dimension of the problem increases, i.e. the number of berth positions, quay cranes, and incoming ships. But, we can see that with N of 8 the numerical optimization only needs calculation time of 156 sec which is relatively fast in comparison to the average operational time for loading and unloading ships. It can be seen from Figure 3.3(a) and Figure 3.3(b) that the completed ship operation time is around four hours at the lowest.

Second simulation results

We simulate each of 300 datasets according to discrete-event system as in (3.6) – (3.14). For each dataset, we evaluate the performance of the FCFS & DBQA, MPA with N-level FCFS, as well as, with N-level HFLL pre-conditioning steps (with $N = 8$), the GA-based and HPSO-based I-BCAP methods.

The average BCAP cost reductions are presented in Figure 3.4, 3.5, and 3.6, where we present a summary of cost reduction for each berth and quay crane configuration from 100 simulations each with respect to the *de-facto* method of FCFS & DBQA (in percentage). The standard deviation is also given for each method and each scenario in these figures. The horizontal and the vertical axis are the seaport configuration and the average reduction cost, respectively. The cost reduction is calculated as a percentage from the traditional FCFS & DBQA method. Vertical line at each average cost point gives the standard deviation. We also present the average calculation time for each method and for each scenario in Figure 3.7.

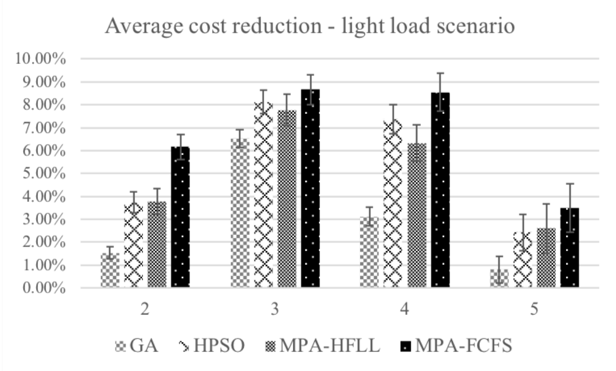

Figure 3.4: *Average BCAP cost reduction from 100 datasets for each berth and QC configuration with the light load scenario.*

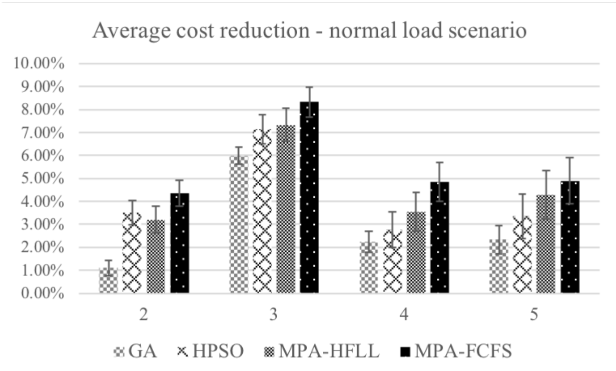

Figure 3.5: *Average BCAP cost reduction from 100 datasets for each berth and QC configuration with the normal load scenario.*

From these results we can see that for the entire scenarios, our proposed MPA with both pre-conditioning steps outperform the traditional method, as well as, when compared to the benchmark method as in [40]. Note that in all of these simulations, the ships in each scenario are not necessarily the same, because the datasets for each simulation run is different.

Table 3.8 – 3.11 give the simulation results of I-BCAP that are solved using our proposed methods (MPA-FCFS and MPA-HFLL), the de-facto FCFS & DBQA combined method and the GA-based I-BCAP method.

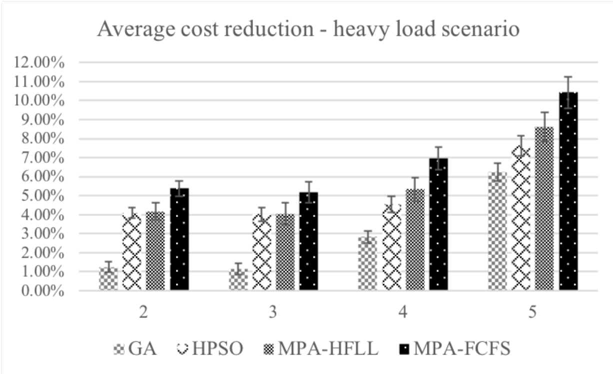

Figure 3.6: *Average BCAP cost reduction from 100 datasets for each berth and QC configuration with the heavy load scenario.*

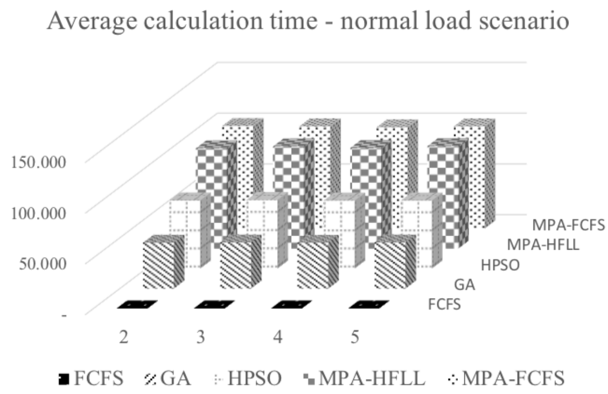

Figure 3.7: *The average calculation per time-step of each method for normal load scenario. The horizontal and vertical axis are seaport configuration and calculation time in seconds, respectively.*

3.5 Field Experiment

In this sub-chapter, we present field experimental results using our proposed method to a real container terminal in Tanjung Priuk, Jakarta. The main purpose of the experiment is to validate our approach in a real-life field experiment. We first discuss the experimental setup in Sub-Chapter 3.5.1 and then present the experimental result in Sub-Chapter 3.5.2.

Table 3.8: *Simulation result of I-BCAP for a terminal with two berth positions and five QC using our proposed methods (MPA-FCFS and MPA-HFLL) which are compared with the standard method of FCFS & DBQA and state-of-the-art method.*

Allocation Strategy	*Average Total cost (Euro)*	*Ave. Calc. time per step (s)*
Sc. 1: Light load		
FCFS & DBQA	7,824,079	0.097
MPA-FCFS	7,342,260	99.739
MPA-HFLL	7,529,231	99.145
HPSO	7,530,542	68.185
GA-based approach	7,704,715	45.244
Sc. 2: Normal load		
FCFS & DBQA	15,765,822	0.097
MPA-FCFS	15,080,885	101.066
MPA-HFLL	15,261,459	98.701
HPSO	15,212,767	67.120
GA-based approach	15,591,721	45.137
Sc. 3: Heavy load		
FCFS & DBQA	31,696,101	0.102
MPA-FCFS	29,988,507	99.699
MPA-HFLL	30,374,911	104.284
HPSO	30,400,539	68.344
GA-based approach	31,302,216	44.962

3.5.1 Experimental setup

The experiment was conducted in Port of Tanjung Priuk, Jakarta for one week in 2016. Indonesia Port Corporation (IPC) as the port owner as well as the operator in the terminal that we study, has given us permission to do the real life field experiment for an academic purpose. In this work, to protect the identity of shipping liners, we anonymize the ship names, as well as, the time information where we initialize the time according to the first arriving ship and maintain the information on the inter-arrival time and on the load information. The original data are available upon request.

We study the arrivals of ships to a terminal which has two berth positions and seven QC, the same terminal from which the data in Sub-Chapter 3.4.1 is obtained. There were, in total, eleven ships which came to the terminal as anonymously shown in Table 3.12.

We can see in Table 3.12 that the arrival time is separated into two different data. The planned arrival time, known as the expected arrival time (ETA) reflects the schedule sent by each shipping liner to the terminal operators as an integrated part of container manifests and is one of the main information sources used by the terminal planner. Considering the very dynamics situation of maritime

Table 3.9: *Simulation result of I-BCAP for a terminal with three berth positions and seven QC using our proposed methods (MPA-FCFS and MPA-HFLL) which are compared with the standard method of FCFS & DBQA and state-of-the-art method.*

Allocation Strategy	*Average Total cost (Euro)*	*Ave. Calc. time per step (s)*
Sc. 1: Light load		
FCFS & DBQA	6,535,782	0.098
MPA-FCFS	5,969,846	100.541
MPA-HFLL	6,028,286	99.792
HPSO	6,004,895	67.453
GA-based approach	6,109,702	45.964
Sc. 2: Normal load		
FCFS & DBQA	13,095,712	0.099
MPA-FCFS	12,004,630	100.697
MPA-HFLL	12,136,182	100.649
HPSO	12,158,785	67.569
GA-based approach	12,311,713	45.345
Sc. 3: Heavy load		
FCFS & DBQA	25,116,485	0.099
MPA-FCFS	23,816,582	100.042
MPA-HFLL	24,100,122	108.388
HPSO	24,106,448	68.392
GA-based approach	24,831,863	45.685

transportation, the planned schedule may alter due to many factors. The two most known factors causing delay are weather and delay from previous seaports. As a result, the actual arrival time may differ from the planned one, as given in the fourth column of the table. Ship index 5, for instance, arrived five hours late. Due to its real-time optimization capability, our proposed method can use the actual arrival time information. We note that there is other source of uncertainties that is not taken into account in our modeling framework explicitly which has influenced the performance of our algorithm. It is the delay caused by the movement of quay cranes from one berthing position to another. Despite this, as shown in the next sub-chapter, our algorithm can still perform very well.

3.5.2 Model validation and experimental results

As we have a limited amount of time (one week) and as the incoming ships always vary (in terms of loads and arrival time), it is not possible to conduct field experiments using different algorithms under exact condition. Therefore, we used the whole week operations for evaluating our proposed MPA-FCFS method where the parameters in our predictive model as in (3.6) – (3.14) are based on apriori information from the bill of lading and from the real-time information. Note that

Table 3.10: *Simulation result of I-BCAP for a terminal with four berth positions and nine QC using our proposed methods (MPA-FCFS and MPA-HFLL) which are compared with the standard method of FCFS & DBQA and state-of-the-art method.*

Allocation Strategy	*Average Total cost (Euro)*	*Ave. Calc. time per step (s)*
Sc. 1: Light load		
FCFS & DBQA	5,500,804	0.100
MPA-FCFS	5,030,628	99.245
MPA-HFLL	5,153,078	99.517
HPSO	5,095,430	68.982
GA-based approach	5,329,191	45.561
Sc. 2: Normal load		
FCFS & DBQA	10,726,893	0.098
MPA-FCFS	10,207,620	99.803
MPA-HFLL	10,346,651	99.078
HPSO	10,428,365	66.981
GA-based approach	10,488,030	45.073
Sc. 3: Heavy load		
FCFS & DBQA	21,479,989	0.099
MPA-FCFS	19,982,082	99.665
MPA-HFLL	20,331,890	99.387
HPSO	20,503,879	68.548
GA-based approach	20,873,761	44.874

the implementation of MPA-FCFS policy to the incoming ships is based on the simulation results in Sub-Chapter 3.4.3 where it is shown that MPA-FCFS method performs consistently better than MPA-HFLL policy. Based on the resulting schedule using our MPA-FCFS, we then validate our predictive model (3.6) – (3.14) against the information from the actual berthing process. Figure 3.8 shows the evolution of one of the state variables (namely, the state of finishing time for the first berth position, denoted by $x_1(k)$) based on our predictive model using the optimized ships schedule in comparison to that obtained from the actual one. In this figure, we can see clearly that our modeling framework in (3.6) – (3.14) captures well the dynamic behaviour of berthing process. The small discrepancy that is observed in this figure is mainly due to the extra dynamics introduced by the QC's movement.

Using the validated predictive model (without incorporating the delay caused by the movement of QC), we compare the performance of our MPA-FCFS with the *de-facto* FCFS & DBQA combined method, with the GA-based I-BCAP method and with the HPSO-based I-BCAP method. The results are summarized in Table 3.13. We calculate the cost incurred according to formula as described in Sub-Chapter 3.2.1, where we used the same unit costs as provided in Sub-Chapter 3.4.1. From this table, we see that our proposed MPA-FCFS method has a lower cost of 6%, 3% and 1.8% than the standard method, the GA-based method [40] and the HPSO-

Table 3.11: *Simulation result of I-BCAP for a terminal with five berth positions and eleven QC using our proposed methods (MPA-FCFS and MPA-HFLL) which are compared with the standard method of FCFS & DBQA and state-of-the-art method.*

Allocation Strategy	*Average Total cost (Euro)*	*Ave. Calc. time per step (s)*
Sc. 1: Light load		
FCFS & DBQA	4,155,288	0.098
MPA-FCFS	3,910,259	100.714
MPA-HFLL	4,046,987	100.377
HPSO	4,054,390	67.103
GA-based approach	4,122,165	45.069
Sc. 2: Normal load		
FCFS & DBQA	8,495,331	0.099
MPA-FCFS	7,879,732	100.661
MPA-HFLL	8,130,301	100.950
HPSO	8,210,458	67.010
GA-based approach	8,297,125	45.361
Sc. 3: Heavy load		
FCFS & DBQA	17,320,403	0.099
MPA-FCFS	15,512,533	100.788
MPA-HFLL	15,828,258	100.638
HPSO	16,000,541	67.946
GA-based approach	16,240,233	45.334

Table 3.12: *Anonymized dataset of arriving ships in the field experiment.*

Ship index	*Arrival date*	*Planned arrival time*	*Actual arrival time*	*Load (TEU)*
1	Day 1	00:00	06:43	7,379
2	Day 1	17:00	17:12	5,428
3	Day 2	01:00	01:00	5,639
4	Day 3	12:00	12:00	6,988
5	Day 4	08:00	13:00	6,523
6	Day 4	11:35	11:20	4,625
7	Day 5	09:40	09:40	4,028
8	Day 5	13:00	13:50	6,853
9	Day 6	07:10	08:00	7,219
10	Day 6	17:00	17:00	7,629
11	Day 7	10:40	11:00	8,725

based method [37], respectively. This agrees qualitatively with our simulation results discussed in the previous sub-chapter. When we take into account the extra cost incurred due to the delay in moving the assets (quay cranes), it provides an explanation to the lower performance than that expected from the simulations

Table 3.13: *Comparison of berth and quay crane allocation performance using the proposed MPA-FCFS method, using the standard FCFS & DBQA combined method and using the GA-based I-BCAP method as in [40]. The MPA-FCFS method is based on a field experiment in Tanjung Priok, Jakarta, while the others are based on a validated predictive model that is used in the field experiment.*

Allocation Strategy	*Average Total cost*
FCFS&DBQA	3,921,923
MPA-FCFS	3,687,532
HPSO-based approach as in [37]	3,754,765
GA-based approach as in [40]	3,812,345

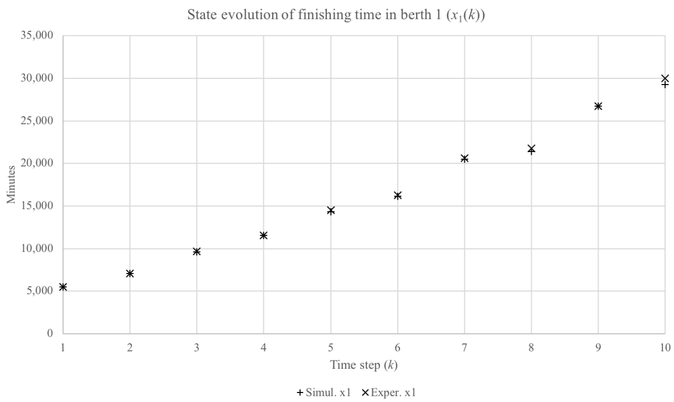

Figure 3.8: *The plot of trajectory of state variable $x_1(k)$ that describes the finishing time of the 1^{st} berth position. The trajectory that is based on the predictive model (3.6) – (3.14) is shown in $+$ while that obtained from the actual field experiment is shown in $\times$. The abscissa is the event-time k and the ordinate is the value of the state variable at each event-time.*

where we can potentially gain up to 8%-9% of cost reduction (c.f. Figure 3.4, 3.5, 3.6). It is therefore foreseen that the inclusion of additional DES models that describe the dynamics of various assets in the berthing process can increase the predictive capability of such integrated models and subsequently improve the real-time optimization algorithm in our MPA method.

3.6 Discussion

We have formulated a dynamical modeling framework of berthing process for general seaport container terminals. The framework can capture the discrete-event dynamics of multiple berthing positions and multiple quay cranes, as well as, asynchronous berthing time for different berthing positions. The discrete-event model has been used in the development of a real-time model predictive allocation strategy for solving the I-BCAP. We have also presented several pre-conditioning steps to tractably solve the optimization problem. Simulation results have shown the efficacy of our proposed method where Monte Carlo simulations have been performed using large-scale realistically-generated datasets.

We have also conducted a field experiment in an actual operational container terminal where our proposed method was applied. The results show that our proposed method still outperforms both the standard and the state-of-the-art methods (based on the validated model).

The BCAP dynamical models and the MPA algorithm that have been developed in this chapter will be the main foundation in this thesis. The modeling framework will later be extended to two different implementation. In Chapter 5, an analysis of container terminal network with implementation of MPA-based BCAP in the selected seaports. In Chapter Chapter 4, the modeling framework is extended to the integrated container terminal operations, in which, not only the seaside, but the storage and transfer operations are also considered. The mathematical analysis of the BCAP modeling framework and MPA algorithm are provided in Chapter 4.

Chapter 4

On the optimal input allocation of DES with dynamic input sequence

Chapter 4

On the optimal input allocation of DES with dynamic input sequence

We present an analysis to the re-ordering of the input sequence to the combinatorial optimization problem in this section. The mathematical analysis in this chapter is based on the modeling framework of the discrete dynamical and asynchronous berth and quay crane allocation problem in Chapter 3. We will firstly explain the systems' descriptions of our discrete-event time systems with dynamic input sequence (DESDIS) and the associated combinatorial optimization problem. Later on, we present the problem of combinatorial optimization where the input sequence changes dynamically with possible new additional sequence and we propose a simple control law that solves the optimization problem. In particular, we show that the resulting control law is based on a particular re-ordering of the input sequence.

4.1 Introduction

Discrete-event systems (DES) are a class of systems where the state variables evolve according to discrete events that take place based on interactions among different (continuous- and/or discrete-) state variables in the systems [21]. A classical example of DES is a queuing system, in which, a new discrete-event is associated to the serving of new customer after the previous one from the previous discrete-event time has been served.,We refer interested readers to [19] for an extensive discussion on the modeling and analysis of DES.

For the past few decades, DES framework has been used to model and to control a large class of physical and cyber-physical systems, which includes, the control of logistics systems, internet congestion control, manufacturing systems and many others that can be described by petri nets or finite-state machine/automata. With its wide application, the DES has attracted many researchers, including in the control community. The research efforts in DES from the control systems perspective can be found in [21], [27], [63] and [68]. Fairly recent applications of DES in transportation and manufacturing systems are presented in [12] and [13] for container terminal operations and in [70] for general transportation and manufacturing systems.

When DES involve discrete-state with discrete input variables, the optimization/control of such DES leads to a combinatorial optimization problem which is NP-hard. One can resort to a standard algorithm for solving combinatorial problems in DES which is the branch and bound (BB) method. As shown in [58], the BB method can converge to the global maxima/minima for some classes of DES optimization problem. Other well-known heuristic methods for solving combinatorial optimization problems with DES are genetic algorithm and particle swarm methods. In [58], an analysis is presented and shows that from the Chebyshev inequality, the solutions obtained from the BB method converge to the global maxima/minima. It is also further shown that the discrepancy between the heuristic and global optimization is not significant.

Although the BB and other heuristic methods can be used to find a sub-optimal solution to the combinatorial problem for DES, the main drawback lies with the facts that the algorithms are limited only to the case where the problem can be recasted as a static optimization problem [25]. In this case, the static refers to constraining the dynamic problem by some terminal conditions and all possible control input are well-defined or known apriori within the given time interval (up to the terminal time).

This approach may no longer be feasible when the terminal conditions are free with infinite time horizon and when the input set changes dynamically and cannot be known apriori ahead of time. The latter case is commonly found in many DES application, such as transportation, scheduling, and logistics, where the actual incoming and outgoing goods always differ from the transmitted goods manifest and where the actual incoming and outgoing vehicles always differ from the precomputed plan. By static, we define that the input sets are known a priori before the optimization processes start. Therefore, a dynamic input set with possible changing input set cannot be handled by the BB method. The real time inputs are commonly found in many DES application, such as transportation, scheduling, and logistics.

In [25], a dynamic DES model is developed for train scheduling problem where the frequent changes to the train operations (schedule, obstacle, rail availability) have limited the use of BB and similar algorithms. Instead of using BB, a greedy travel advance strategy is proposed in [25] on the basis of a dynamic DES model, which is able to find the sub-optimal control inputs of the train schedules with a framework similar to line search algorithm. The possible solutions in each iteration are limited to the group of trains in the same vicinity of direction and speed. Another related paper is [32] where stabilization problem for a particular DES with dynamic input set is considered. In this case, the events in DES are asynchronous where the states of each sub-system do not necessarily follow the same clock times and an LMI-based controller is proposed to solve such problem. By solving a Lyapunov function and linear matrix inequalities (LMI), the controller are able to give sufficient results, where the cost function monotonically decreases.

A similar DES with asynchronous event transition can also be found in our previous works as in [12, 13]. In these works, a model predictive allocation (MPA) method is proposed in conjunction with a pre-conditioning step. In particular, the DES model of container terminal operations is used to compute an optimal input sequence for a finite event horizon where the input sequence is heuristically pre-conditioned for accommodating the combinatorial optimization step. The proposed MPA method follows the same procedure as the model predictive control approach. The efficacy of our proposed method has been shown in both simulation as well as in real-life experiment. In this method, we have used the well-known first-come first-serve (FCFS) or the heavy-first light-last (HFLL) pre-conditioning step to the current input sequence and then truncate it, prior to computing the optimal solution in the model predictive step. While the re-ordering of the sequence (either using FCFS or HFLL) has played an important role in [12, 13], the mathematical analysis on the re-ordering of the input sequence in the pre-conditioning step is still missing.

In this research, we present an analysis to the re-ordering of the input sequence to the combinatorial optimization problem. The systems' descriptions of our discrete-event time systems with dynamic input sequence (DESDIS) and the associated combinatorial optimization problem are presented in Sub-Chapter 4.2. In Sub-Chapter 4.3, we present the problem of combinatorial optimization where the input sequence changes dynamically with possible new additional sequence and we propose a simple control law that solves the optimization problem. In particular, we show that the resulting control law is based on a particular re-ordering of the input sequence. Finally, conclusions and future works are presented in Sub-Chapter 4.4.

4.2 Preliminaries and Optimal Input Allocation Problem

Notations. We denote the vector of all ones by $\mathbb{1}$. A matrix $A \in \mathbb{R}^{n\times n}$ is called a stochastic matrix if $A_{ij} \geqslant 0$ for all i, j and $\sum_j A_{ij} = 1$ for all i where A_{ij} is the (i, j) element of A. Let us consider an undirected graph $\mathcal{G}$ given by the (V, E) where V is the set of vertices and $E \subset V \times V$ is the set of edges.

Such graph $\mathcal{G}$ can be represented by the stochastic matrix A where the element A_{ij} shows the communication weight from the j-th vertex to the i-th vertex. The graph $\mathcal{G}$ is called connected if for every pair vertices there is a path on the graph that connects these vertices. Equivalently, it is connected if the kernel of A has rank one and is given by $\mathbb{1}$.

Let us consider the following generic model of discrete event system Σ whose input is taken from a time-varying sequence:

$$\Sigma : x(k+1) = Ax(k) + Bf(x(k), u(k)), \tag{4.1}$$

where $x(k) \in \mathbb{R}_+^n$ denotes the state variables (such as, the berth starting time, berth operations time, berth finishing time, etc.), k is the discrete-event time variable, $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times m}$ is assumed to be *non-negative matrix*, i.e., $a_{ij} \geqslant 0$ and $b_{ij} \geqslant 0$. This system description is relevant for describing the dynamic behaviour of planning and scheduling in logistics systems, particulary in the container terminal operations, which will be shown later in our example.

We remark that if we consider the case where $f = 0$ then our assumption on A implies that (4.1) describes a positive discrete-event system. In this case, $x(0) > 0$ (where the inequality is interpreted element-wise) implies that $x(k) > 0$ for all $k > 0$. Each element in the function $f : \mathbb{R}_+^n \times \mathbb{R}_+^m \to \mathbb{R}_+^m$ is assumed to be positive definite function, i.e., $f_i(x, u) > 0$ for all $(x, u) \neq (0, 0)$ and for all i, which is monotonically increasing with respect to the second argument and is uniformly with respect to the first argument. More precisely, f satisfies

$$(u_2 - u_1)^T \left[f(x, u_2) - f(x, u_1) \right] > 0 \qquad \forall u_1 \neq u_2, \forall x.$$

The proof of the lemma follows trivially from the property of a contracting non-negative matrix A. A transition matrix whose eigenvalues are outside the open unit circle (i.e. $\sigma(A) \in \mathbb{C} \backslash \mathbb{B}_1$ with $\mathbb{B}_1$ be an open ball with radius of one and centered at the origin).

Let us describe again our general system in (4.1). In fact, for logistics systems example as given below, we may consider the case where $\rho(A) = 1$, in which case, the state x grows exponentially. This is relevant for the situation where x describes the time variables of the underlying process, such as, loading time, finishing time, or the variables that describe the cumulative number of the processed goods.

The input variable $u(k) \in \mathbb{R}^m$ is the decision/input variable that is taken from a (possibly, infinite) sequence $\mathcal{U}_k = \{v_i\}_{i \in \{1,2,\ldots,N\}}$. We further assume that the evolution of $u(k)$, $x(k)$ and $\mathcal{U}_k$ follow the following rule.

Discrete-Event Systems with Dynamic Input Sequence (DESDIS):

(A1). (*Initialization.*) Let the initial sequence $\mathcal{U}_0$ be given by $\mathcal{U}_0 := \{v_i\}_{i \in \{1,2,\ldots N\}}$ with $v_i \in \mathbb{R}_+^m$ for all $i = 1, \ldots N$ with N be the dimension of initial sequence and let $k = 0$. For every step $k = 0, 1, \ldots, N$ the following operation is executed:

(A2). (*Decision making step and state evolution.*) If $\mathcal{U}_k \neq \varnothing$ then let the input $u(k)$ be given by an element from $\mathcal{U}_k$, i.e., $u(k) \in \mathcal{U}_k$ (which can be based on a particular input allocation/control law which will be discussed later) and the state is updated according to (4.1). Otherwise, $u(k) = 0$.

(A3). (*Update of the input sequence.*) If $\mathcal{U}_k \neq \varnothing$ then the decision sequence is updated according to $\mathcal{U}_{k+1} = \mathcal{U}_k \backslash u(k)^\frown V_k$ where $\backslash$ is the element removal operation from the sequence, $^\frown$ is a concatenation operator of two sequences and V_k is the possible new additional sequence at step k. In other words

$$\mathcal{U}_{k+1} = \{\mathcal{U}_k \backslash u(k), V_k\}.$$

Otherwise $\mathcal{U}_{k+1} = V_k$.

(A4). (*Update of the event time.*) Let $k = k+1$ and return to **(A2)**.

Note that for the particular case of $V_k = u(k)$ (i.e., $\mathcal{U}_{k+1} = \mathcal{U}_k$ or the element that is taken from $\mathcal{U}(k)$ is directly replenished by an identical element $\{u(k)\}$), we have the usual description of linear discrete-time systems with atomic input set $\mathcal{U}$. For the control of such systems with a fixed atomic input set $\mathcal{U}$, we can refer interested readers to the literature on discrete-event control systems, on hybrid control systems and on finite-state automata.

When A is a stochastic matrix and $f = 0$, it is well-known that the state trajectories of the autonomous system will reach consensus (see, for instance, [10, 34]).

Lemma 4.1. *Consider the system* (4.1) *with $f = 0$ and A be stochastic matrix. Then for every $x(0) \in \mathbb{R}^n_+$, there exists $x^* \in \mathbb{R}^n_+$ such that $\lim_{k\to\infty} x(k) = x^*$, i.e., the state trajectory x converges to a positive constant vector. Moreover if the graph associated to A is strongly connected then $x^* = a\mathbb{1}$ for some constant $a > 0$.*

Example 4.1. As a simple example to the system described in (4.1), we can consider the following system

$$x(k+1) = x(k) + F(u(k)) \tag{4.2}$$

where $x(k), u(k)$ are scalar and the function F is a positive definite function. This simple dynamics may represent many logistics systems where the state x represents a particular operations time. For example, a simple berthing process [12] can be described by (4.2) where $x(k)$ defines the berthing time for the k-th event and F defines the operational time of loading/unloading the ship whose size is associated to $u(k)$. In this case, each element in the sequence $\mathcal{U}_k$ represents the set of ship's size that are waiting to be berthed. When a ship $u(k)$ has been assigned for berthing, then this ship will be no longer in the set $\mathcal{U}_k$ and at the same event, a new set of arriving ships V_k can call to the port for berthing. After the berthing process of the ship $u(k)$ is completed, the ship will continue its journey to its next destination and therefore it will also not appear in the next input sequence $\mathcal{U}_{k+1}$ and V_k will be added to this new sequence.

Other example that can be described by (4.2) is the discrete production process where $x(k)$ describes the finishing production time for the k-th event, $u(k)$ represents the set of number of products to be produced in the line and F is the operational production time which is a (nonlinear) function of the product order.

For a higher-order example, we can consider Markov chain describing Markov processes where A is a stochastic matrix. △

We remark that the assumption on A and the positive definiteness of f ensures that $\|x(\ell)\| > \|x(m)\|$ for all $\ell > m$ when $u = 0$. This property is shown in the following lemma.

Lemma 4.2. *Consider the system* (4.1) *as above with* $u = 0$ *and positive definite function* f *satisfying* $\|f(\xi, 0)\|^2 \geqslant L\|\xi\|^2$ *where* $L > 0$ *satisfies*

$$L > \frac{1 - \lambda_{\min}(A^T A)}{\lambda_{\min}(B^T B)} \tag{4.3}$$

with $\lambda_{\min}$ *denotes the minimum eigenvalue. Then* $\|x(\ell)\| > \|x(m)\|$ *for all* $\ell > m$.

Proof : By taking $\ell = m + 1$, we have that

$$\begin{aligned}
&\|x(\ell)\|^2 = \|Ax(m) + Bf(x(m), 0)\|^2 \\
&= x^T(m)A^T Ax(m) + 2x^T(m)A^T Bf(x(m), 0) \\
&\quad + f^T(x(m), 0)B^T Bf(x(m), 0) \\
&\geqslant \lambda_{\min}(A^T A)\|x(m)\|^2 + \lambda_{\min}(B^T B)\|f(x(m), 0)\|^2 \\
&> \|x(m)\|^2
\end{aligned}$$

where the first inequality is due to the non-negativity of $x(m)$, A and B, and last inequality is due to the bound on $\|f(\xi, 0)\|^2$ in (4.3). □

The above results show that under the hypothesis of Lemma 4.2, we have the following ordering of the norm of state trajectories.

$$\|x(0)\| < \|x(1)\| < \ldots < \|x(k)\| < \|x(k+1)\| < \ldots$$

Furthermore, if we restrict to the classes of system where the diagonal elements of A are greater than or equal to 1 then we have ordering of every element of the state trajectories.

Lemma 4.3. *Suppose that* (4.3) *holds and* $a_{ii} \geqslant 1$ *for all* i*. Then we have*

$$x_i(0) < x_i(1) < \ldots < x_i(k) < x_i(k+1) < \ldots$$

for all $i = 1, \ldots n$*.*

Proof : The proof of the lemma follows immediately from the dynamics of each state variable x_i that is given by

$$x_i(k+1) = a_{ii}x_i(k) + \sum_{j \neq i} a_{ij}x_j(k) + b_i f(x(k), u(k)).$$

Since $x_j(k) > 0$ for all $j \neq i$ and f is a positive definite function, the claim follows trivially. □

In the above results, we have shown nice ordering properties in terms of the state trajectories of (4.1). In particular, when the system is driven by integrators (such as the one in Example 4.1) each state trajectory is a monotonically increasing signal.

Let us now introduce how the decision process in the input allocation can influence the state dynamical behaviour by using the simple example as in Example 4.1.

Example 4.2. Consider the simple system as in (4.2) with $F(s) = s$. For simplicity, let us consider $\mathcal{U}_0 = \{1, 5, 4\}$ and the state is initialized at the origin, i.e., $x(0) = 0$. Since the cardinality of $\mathcal{U}_0$ is 3 ($\dim(\mathcal{U}_0) = 3$) and we do not consider replenishment to the input sequence (e.g., $V_k = \varnothing$ for all k), then the state evolution following the **DESDIS** rule reaches steady state in a finite-time.

First-Come First-Serve (FCFS) rule. If the input is assigned according to the first-come first-serve rule (where the order in the sequence $\mathcal{U}_0$ determines the assignment timing, i.e., the first element is used first and the last element will be applied lastly) then the input is given by $u(0) = 1, u(1) = 5, u(2) = 4$. In this case, the trajectory of x is given by

$$\begin{aligned}
x(0) &= 0 \\
x(1) &= x(0) + u(0) = 1 \\
x(2) &= x(1) + u(1) = 6 \\
x(3) &= x(2) + u(2) = 10 \\
x(4) &= x(3) \\
&\vdots \\
x(k+1) &= x(k)
\end{aligned}$$

and the evolution of the input sequence is

$$\begin{aligned}
\mathcal{U}_1 &= \mathcal{U}_0 \backslash u(0)^\frown \varnothing = \{5,4\} \\
\mathcal{U}_2 &= \mathcal{U}_1 \backslash u(1)^\frown \varnothing = \{4\} \\
\mathcal{U}_3 &= \mathcal{U}_2 \backslash u(2)^\frown \varnothing = \varnothing \\
\mathcal{U}_k &= \varnothing \qquad\qquad\qquad \forall k > 3.
\end{aligned}$$

Reordering the input sequence. Instead of assigning the input using FCFS rule as above, we can also set the input $u(k)$ according to a certain control law/allocation mechanism based on the available elements in the current input sequence $\mathcal{U}_k$. One particular instance for this is by firstly rearranging the input sequence $\mathcal{U}_0$ in the *ascending order* and then apply the FCFS rule. Note that this control law is equivalent to taking the minimum over $\mathcal{U}_k$, i.e., $u(k) = \min\{\mathcal{U}_k\}$. For the above example, we can rearrange $\mathcal{U}_0$ into $\{1,4,5\}$ and then we apply the FCFS rule. This gives us $u(0) = 1, u(1) = 4, u(2) = 5$, $u(k) = 0$ for all $k > 2$ and subsequently, $x(0) = 0, x(1) = 1, x(2) = 5, x(3) = 10 = x(k)$ for all $k > 3$. The input sequence evolution is given by $\mathcal{U}_1 = \{4,5\}, \mathcal{U}_2 = \{5\}, \mathcal{U}_3 = \varnothing = \mathcal{U}_k$ for all $k > 3$. We remark that since the cardinality of $\mathcal{U}_0$ is 3 then there are 6 combinations for the reordering of the input sequence $\mathcal{U}_0$. In the rest of the chapter, we will discuss such allocation/control problem for finding an optimal reordering of the input sequence that minimizes a given cost function. △

In Example 4.2, two possible state trajectories have been shown based on two different ways the input is allocated from the initial input sequence. As remarked in Example 4.2, there are in total six possible state trajectories where all of them will reach the same steady-state value of 10. In fact, the property of steady state value that is invariant to all possible combination of input allocation can be extended to the system (4.1), where f is only a function of u, as given in the following proposition.

Proposition 4.4. *Consider the system* (4.1) *with* $f : (x, u) \mapsto F(u)$ *where* F *is a function of* u *and* A *is a doubly stochastic matrix. If* $V_k = \varnothing$ *for all* k *then for all initial input sequence* $\mathcal{U}_0$ *with card*$\{U_0\} =: N_0$*, the systems with dynamic input sequence described by* **(A1)-(A4)** *satisfies*

$$\mathbb{1}^T x(k) = \sum_{n=0}^{N_0-1} \mathbb{1}^T BF(u(n)) \qquad \forall k \geqslant N_0,$$

i.e., the sum of the state $x(k)$ *is constant for all* $k \geqslant N_0$*.*

Proof : As the input sequence is not replenished for every input allocation, it follows from the **DESDIS** rule **(A3)** that $\mathcal{U}_k = \varnothing$ for all $k \geqslant N_0$. This means that for all $k \geqslant N_0$, $u(k) = 0$ and therefore,

$$\begin{aligned}
\mathbb{1}^T x(k+1) &= \mathbb{1}^T Ax(k) = \mathbb{1}^T x(k) \qquad \forall k \geqslant N_0 \\
&= \mathbb{1}^T Ax(N_0 - 1) + \mathbb{1}^T BF(u(N_0 - 1)) \\
&= \mathbb{1}^T x(N_0 - 1) + \mathbb{1}^T BF(u(N_0 - 1)) \\
&= \mathbb{1}^T Ax(N_0 - 2) + \sum_{n=N_0-1}^{N_0-2} \mathbb{1}^T BF(u(n)) \\
&\ \ \vdots \\
&= \sum_{n=0}^{N_0-1} \mathbb{1}^T BF(u(n)).
\end{aligned}$$

This equality holds for arbitrary choice of input allocation $u(k), k = 0, \ldots, N_0 - 1$ from $\mathcal{U}_0$. $\square$

When we furnish the current input sequence $\mathcal{U}_k$ with an additional new sequence V_k at every time step k (as in the step **(A3)** of the **DESDIS** update rule), the number of possible state trajectories can increase dramatically.

For the system in (4.1), suppose that we can define a cost function $J(x, u)$ that must be minimized by optimally allocating the input $u(k)$ from the available finite sequence $\mathcal{U}_k$. Using such J, we can define our optimal allocation/control problem as follows.

Optimal input allocation problem: For a given discrete-event system (4.1) with dynamic input sequence satisfying **(A1)-(A4)**, with given expansion input sequences V_k, $k \in \mathbb{N}$ and cost function $J(x,u) : l^\infty(\mathbb{N},\mathbb{R}^n) \times l^\infty(\mathbb{N},\mathbb{R}^m) \to \mathbb{R}_+$, determine the optimal input allocation $u^*(k) \in \mathcal{U}_k$ for all k such that

$$u^* = \operatorname*{argmin}_u J(x,u)$$

where the state trajectory x satisfies (4.1) and the **DESDIS** rule in **(A1)-(A4)**.

In the following proposition, it is shown that for the particular case when $V_k = \varnothing$ for all k, the solution to the above optimal input allocation problem corresponds to a particular ordering of $\mathcal{U}_0$ that depends on B and the nonlinear function F.

Proposition 4.5. *Consider the system as in Proposition 4.4 and suppose that the cost function is given by*

$$J(k) = \sum_{n=0}^{k} \mathbb{1}^T x(n). \tag{4.4}$$

Then the optimal input allocation u^* satisfies

$$\mathbb{1}^T BF(u^*(0)) \leqslant \mathbb{1}^T BF(u^*(1)) \leqslant \cdots \leqslant \mathbb{1}^T BF(u^*(N_0)) \tag{4.5}$$

where $N_0 :=$ card$(\mathcal{U}_0)$ and $u^*(k) = 0$ elsewhere.

Proof : Following the computation in the proof of Proposition 4.4, we can expand (4.4) as follows.

$$\begin{aligned}
J(k) &= \mathbb{1}^T x(k) + \mathbb{1}^T x(k-1) + \cdots + \mathbb{1}^T x(0) \\
&= \sum_{n=0}^{k-1} \mathbb{1}^T BF(u(n)) + \sum_{n=0}^{k-2} \mathbb{1}^T BF(u(n)) + \cdots \\
&\qquad + \mathbb{1}^T BF(u(0)) + k\mathbb{1}^T x(0) \\
&= k\mathbb{1}^T BF(u(0)) + (k-1)\mathbb{1}^T BF(u(1)) + \cdots \\
&\qquad + \mathbb{1}^T BF(u(k)) + k\mathbb{1}^T x(0).
\end{aligned}$$

It follows immediately from this equality and since $\mathcal{U}_0$ is finite that the minimum of J is reached if and only if (4.5) holds which is independent of the initial condition $x(0)$. □

We want to recall the readers that as the **DESDIS** fulfills **(A1) - (A4)**, the optimal input sequence satisfies

$$(u^*(0), u^*(1), \ldots, u^*(N_0)) \in \mathrm{P}_{N_0}(\mathcal{U}_0)$$

where $\mathrm{P}_{N_0}(\mathcal{U}_0)$ denotes the set of N_0-permutations of $\mathcal{U}_0$.

As defined in (4.4), the cost function in the above proposition is given by the sum of all state values at all time. Thus the minimization of the cost function means that at any given time the state values must be made as small as possible by allocating proper input sequence. If we refer to Example 1, this cost function can be interpreted as the sum of all berthing time and the minimization of this function implies that the optimal input allocation will ensure that the berthing time is kept as small as possible. Consequently the berth position will be made vacant as earlier as possible.

We have shown in Proposition 4.5 that in the absence of input sequence expansion V_k, the solution to the optimal input allocation problem is given by ordering the sequence $(u^*(0), u^*(1), \ldots, u^*(N_0))$ such that (4.5) holds. In particular, if $x(k)$ is a scalar, i.e., $B = 1$, and F is positive definite and non-decreasing then (4.5) becomes

$$u^*(0) \leqslant u^*(1) \leqslant \cdots \leqslant u^*(N_0),$$

i.e., the scalar input $u^*(k)$ is ordered from the lowest to the largest. On the other hand, if F is non-increasing then we have that $u^*(0) \geqslant u^*(1) \geqslant \cdots \geqslant u^*(N_0)$. Such an ordering property in the optimal input sequence is closely related to the reordering input sequence approach as discussed in Example 2. In this example, instead of allocating the input according to the sequence $\mathcal{U}_0$ (i.e., $u(0) = 1, u(1) = 5$ and $u(2) = 4$), we can firstly reorder $\mathcal{U}_0$ and then assign the input according to the re-ordered sequence.

In general, such ordering property in the optimal input sequence is not true for arbitrary cost functions J. This can again be exemplified by Example 4.2. In this example, if the cost function is given by $J(3) = \sum_{n=0}^{3} x(n)$ then the optimal control input is $u(0) = 1, u(1) = 4, u(2) = 5, u(k) = 0$ for all $k > 2$. However, when the cost function is modified into $J(3) = \sum_{n=0}^{3} x(n) + \sum_{n=0}^{3} 2u(n)n$, where we penalize the allocation of high control input at a later time, the optimal control input becomes $u(0) = 5, u(1) = 4, u(2) = 1, u(k) = 0$ for all $k > 2$. In this case the order of optimal input allocation is reversed despite the fact that F is an identity (contrary to what has been described before on non-decreasing F).

In the following sub-chapter, we consider the aforementioned optimal input allocation problem when $V_k \neq 0$.

4.3 Optimal Input Allocation with $V_k \neq 0$

Previously, we have seen that when the input sequence is not dynamically expanded by V_k as defined in step **(A3)**, the optimal input allocation is based on a particular order of the initial input sequence $\mathcal{U}_0$. In fact, it can be described by

$$u(k) = \operatorname*{argmin}_{\xi \in \mathcal{U}_k} \mathbb{1}^T BF(\xi)$$

where $\mathcal{U}_k$ is always updated according to **(A3)**. We will now study whether the above control law solves also the optimal input allocation problem for the case of $V_k \neq 0$.

Proposition 4.6. *Consider the system* (4.1) *with* $f : (x, u) \mapsto F(u)$ *where* F *is a function of* u*,* A *is a doubly stochastic matrix and the system satisfies* **(A1)**-**(A4)**. *Let* V_k*,* $k \in \mathbb{N}$*, be the expansion sequences for the input sequence* $\mathcal{U}_k$ *that is updated according to* **(A3)**. *Suppose that the cost function is given by* (4.4). *Then for all initial input sequence* $\mathcal{U}_0$*, the control law defined by*

$$u(k) = \operatorname*{argmin}_{\xi \in \mathcal{U}_k} \mathbb{1}^T BF(\xi) \qquad \forall \text{ event time } k \in \mathbb{N} \tag{4.6}$$

solves the optimal input allocation problem.

Proof : We prove the proposition by induction. When we want to minimize $J(1)$, it is straightforward to see that the optimal input $u^*(0)$ is given by (4.6) with $k = 0$. When $k = 2$, we need to show that the optimal cost function $J^*(2)$ is obtained by taking

$$u(0) = \operatorname*{argmin}_{\xi \in \mathcal{U}_0} \mathbb{1}^T BF(\xi)$$

and

$$u(1) = \operatorname*{argmin}_{\xi \in \mathcal{U}_0 \backslash u(0) ^\frown V_0} \mathbb{1}^T BF(\xi),$$

where we have applied **(A3)** to get $\mathcal{U}_1 = \mathcal{U}_0 \backslash u(0) ^\frown V_0$. Indeed, by direct computation, we have that

$$\begin{aligned} J^*(2) &= \min_u J(k) \\ &= \min_{\substack{u(0) \in \mathcal{U}_0 \\ u(1) \in \mathcal{U}_1}} \left(2\mathbb{1}^T BF(u(0)) + \mathbb{1}^T BF(u(1))\right) + 2\mathbb{1}^T x(0). \end{aligned} \tag{4.7}$$

It remains to show that the above equality can be written as

$$= \min_{u(0)\in\mathcal{U}_0} 2\mathbb{1}^T BF(u(0)) + \min_{u(1)\in\mathcal{U}_1} \mathbb{1}^T BF(u(1)) + 2\mathbb{1}^T x(0),$$

in which case, the claim of the proposition for the minimization of $J^*(2)$ holds. We prove it by contradiction. Suppose that there exists $u(0) \in \mathcal{U}_0$ and $u(1) \in \mathcal{U}_1$ such that (4.7) holds while the above equality is not satisfied. In this case, we have two cases: (i). both $u(0), u(1) \in \mathcal{U}_0$; or (ii). $u(0) \in \mathcal{U}_0$ and $u(1) \in V_0$. For the first case, we arrive at a similar situation as in Proposition 4.5 which also implies that the above equality holds, a contradiction. On the other hand, for the second case, it follows trivially that the above equality holds; again a contradiction.

For the last part of proof by induction, we will now show that given the optimal input allocation $U_k^* := (u^*(0), u^*(1), \ldots, u^*(k))$, which is calculated recursively as in (4.6) and minimizes $J(k+1)$, the minimizer of $J(k+2)$ is given by $(U_k^*, u^*(k+1))$ where $u^*(k+1)$ is computed as in (4.6). Similar as before, we have that

$$\begin{aligned} J^*(k+2) &= \min_{\substack{u(0)\in\mathcal{U}_0 \\ \vdots \\ u(k+1)\in\mathcal{U}_{k+1}}} \Big((k+1)\mathbb{1}^T BF(u(0)) + k\mathbb{1}^T BF(u(1)) + \cdots \\ &\quad \cdots + 2\mathbb{1}^T BF(u(k)) + \mathbb{1}^T BF(u(k+1))\Big) \\ &\quad + (k+1)\mathbb{1}^T x(0). \end{aligned}$$

Using the same arguments as before, we can prove by contradiction that the above equality is equivalent to

$$\begin{aligned} J^*(k+2) &= \min_{\substack{u(0)\in\mathcal{U}_0 \\ \vdots \\ u(k)\in\mathcal{U}_{k}}} \Big((k+1)\mathbb{1}^T BF(u(0)) + k\mathbb{1}^T BF(u(1)) + \cdots \\ &\quad \cdots + 3\mathbb{1}^T BF(u(k-1)) + 2\mathbb{1}^T BF(u(k))\Big) \\ &\quad + \min_{u(k+1)\in\mathcal{U}_{k+1}} \mathbb{1}^T BF(u(k+1)) + (k+1)\mathbb{1}^T x(0). \end{aligned}$$

The solution to the first term of the above equality is the same as the minimization of $J(k+1)$, which is U_k^*. For the second term, it is the solution $u^*(k+1)$ to (4.6) for $k+1$. Therefore, we have that $U_{k+1}^* = \big(U_k^*, u^*(k+1)\big)$ as claimed.

Firstly, as $\mathcal{U}_0$ is not influenced by V_0, it follows from (4.7) that

$$J^*(2) = \min_{u(0)\in\mathcal{U}_0} 2\mathbb{1}^T BF(u(0)) + \min_{\substack{u(0)\in\mathcal{U}_0 \\ u(1)\in\mathcal{U}_1}} \mathbb{1}^T BF(u(1)) + 2\mathbb{1}^T x(0)$$

$\square$

In this proposition, we have shown that the optimal input allocation can be computed recursively. In particular, at each event time k, $u^*(k)$ is obtained based only on the current input sequence $\mathcal{U}_k$ and is independent on the possible expansion sequence in any future event time. In this regards, the optimization of $J(k+1)$ requires only card$(\mathcal{U}_k) =: N_k$ operations instead of $N_k!$ (or even larger when we take into account all possible permutation with the inclusion of future V_n, $n > k$).

Example 4.3. Let us consider again the system as in Example 2. Suppose that $V_0 = \{3, 2\}$, $V_1 = \{1, 3, 5\}$ and $V_k = \varnothing$ for all $k > 2$. Using the input allocation law as in Proposition 4.6, we can recursively compute the optimal input allocation $u(k)$ as follows:

$$\begin{aligned}
u(0) &= \operatorname*{argmin}_{\xi\in\{1,5,4\}} \xi = 1, & \mathcal{U}_1 &= \{5,4\} \cup \{3,2\} \\
u(1) &= \operatorname*{argmin}_{\xi\in\{5,4,3,2\}} \xi = 2, & \mathcal{U}_2 &= \{5,4,3\} \cup \{1,3,5\} \\
u(2) &= \operatorname*{argmin}_{\xi\in\{5,4,3,1,3,5\}} \xi = 1, & \mathcal{U}_3 &= \{5,4,3,3,5\} \\
u(3) &= \operatorname*{argmin}_{\xi\in\{5,4,3,3,5\}} \xi = 3, & \mathcal{U}_4 &= \{5,4,3,5\} \\
u(4) &= \operatorname*{argmin}_{\xi\in\{5,4,3,5\}} \xi = 3, & \mathcal{U}_5 &= \{5,4,5\} \\
u(5) &= \operatorname*{argmin}_{\xi\in\{5,4,5\}} \xi = 4, & \mathcal{U}_6 &= \{5,5\} \\
u(6) &= \operatorname*{argmin}_{\xi\in\{5,5\}} \xi = 5, & \mathcal{U}_7 &= \{5\} \\
u(7) &= 5 \qquad\qquad \text{and} & u(k) &= 0 \qquad \forall k > 7.
\end{aligned}$$

Thus, the maximum number of operations is on the computation of $u(2)$. We can compare this with the exhaustive search of $(u^*(0), u^*(1), u^*(2), \ldots, u^*(7))$ where we evaluate all permutation of the combined input sequence $(\mathcal{U}_0, V_0, V_1, V_2) = \{1, 5, 4, 3, 2, 1, 3, 5\}$ and evaluate the state evolution of **DESDIS** for each of possible permutation of input sequence according to **(A1)–(A4)**. $\triangle$

4.4 Discussion

Based on the DES modeling framework and MPA algorithm from Chapter 3, we have shown that for a particular case of allocation problem in DESDIS, re-ordering of the input sequence at every discrete-event time is needed for determining the optimal sequence. The DES is a class of dynamic system with wide applications, such as in transportation, scheduling and logistics. The DES often interrelates with combinatorial optimization and the current algorithm, such as the branch and bound (BB) is only suitable for the static problem setting. Although some analyses of the optimality condition in DES have been taken into account, a special condition of DES with input sequence has not yet studied rigorously. The analysis provides a good basis for the development of model predictive allocation methods for DESDIS as pursued, for instance, in [12, 13]. The results show that the DESDIS produce optimal allocation when the controller follow some of the remarks that we imposed, namely the preconditing of the sequence. It is further shown that the same precondition in a model predictive control (MPC) problem setting will yield a monotonically decreasing cost function as the MPC time horizon grows. Further works are needed on the re-ordering of input sequence $\mathcal{U}_k$ (and the subsequent expansion sequences V_k for a finite event horizon) when a general cost function is considered.

Chapter 5

An analysis of competitive terminal network via BCAP optimization policies

Chapter 5

An analysis of competitive terminal network via BCAP optimization policies

We present in this chapter an analysis of the deployment of a distributed dynamic optimization algorithm to a sub-network of the container terminals. We extend the I-BCAP dynamical DES model which has been developed in Chapter 3 to the operations of network of container terminals. In each network, we select the selected seaports to implement the MPA-based I-BCAP. We discuss the framework of container terminal modeling. We later review the BCAP model and the MPA algorithm proposed in Chapter 3. Afterwards, simulation setup, data set, and results are discussed. Finally, some findings will be discussed.

5.1 Introduction

Container terminals as one of the main actor in maritime transportation has also enjoyed the growth in global containerized trade. This mutual benefit is more likely shown in container terminal networks [56]. We define the network as a group of specific container terminals, which a number of ships frequently call at. A network can be managed by the same terminal operator.

A cost-optimal container terminal network will hugely profit the terminal operators ([82], [6], [60]). It is discussed in [45, 62, 88] that a group of operators always seek novel ways to improve the terminal networks' performances. At the same time from the shipping liners' point of view, it is also an asset [6], where the main interest of the liners is to optimize their operations in a network. As another important stakeholder, terminal operators are looking for ways to optimize their overall performance in the network with minimal implementation cost while maintaining the autonomy of each terminal in regulating their own operations and keeping the independency of the liners that they serve [6, 87]. As emphasized in [82] most works in container terminal network are in optimization of shipping liners, but there is also important interest from the terminal operators to improve the performance of terminal networks' operational performance [81]. In the later explanation, we will discuss the importance of optimization from the terminal operators' perspective.

Consequently, in this chapter, we investigate the terminal network optimization through the deployment of distributed (e.g. locally implemented) optimization methods for container terminal operations. In particular, our main research questions are how the heterogeneity in the local decision making process of every terminal affects the whole terminal network performance and how we can effectively deploy distributed optimization methods in the network. We implement the decision in one section of terminal, namely berthing process, since the literature has shown that BCAP is the major bottleneck in container terminal [18, 40, 48, 66]. We also will discuss more detail the relations of optimization in BCAP and the improvement of container terminal networks overall operational performance.

As a particular example, we consider the maritime transportation sector in Indonesia, which is an archipelago across an area of 5,271 km from West to East and 2,210 km from North to South and where the network of islands are only served by four operators, namely ”Pelindo I”, ”Pelindo II”, ”Pelindo III”, and ”Pelindo IV”. In contrast to the customers of the break-bulk and general cargo terminals which can be classified as tramper, the main customers in container terminal in Indonesia are the shipping liners [85]. As opposed to the shipping liners, ships engaged in tramp do not have fixed schedule of ports to call.

With the aforementioned example, an operational optimization policy applied in one node in the ”Pelindo” network can greatly affect the network performances in Indonesia. This statement is even more highlighted considering the fact that container throughput in Indonesia has been growing rapidly for the past decade. For realizing its potential, The Government of Indonesia has launched a program to enhance its maritime transportation network, namely ”The Indonesian sea-highways”. The aim of this project is to improve the network performance comprising of a selected main seaports that cover Indonesia trade area from West to East.

The importance of container terminal networks has motivated some research activities in this topic. In [82] the authors evaluate a number of decisions involved in a network, namely the container routing, fleet management and network design. Container routing is an optimization problem where every single container movement in a network needs to be determined optimally. Many other works focus on balancing ship assignments such as the minimization of empty containers in the return journey [38], [65]. Fleet management deals mainly with scheduling and speed optimization of shipping liners in the networks. Network design is usually related to selecting combination of container terminals that will serve the shipping liners.

Recently, the optimization of a container network is usually studied from the perspective of shipping liners. Hence, liners’ performance criterion is used primarily in the optimizations’ cost function. One important criterion is the ship speed, as used in [38] where the vessel speed is used as the decision variable for determining the shipping liners schedule. In another work [65], a different criteria is considered for determining optimal vessel schedules where fuel emissions are used in the cost function. From a more tactical level, a number of vessel voyage in a network is also a decisive determinant in network efficiency. Large players in shipping industry usually receive multiple demands from freight forwarders (who represent consignees). Accommodating the entire demand can be inefficient, since some of the demands are in small numbers; therefore, economic of scale will not be met. Selecting only important voyages is thus beneficial for shipping liners, as discussed in [53].

While many papers have discussed how a stakeholder should develop a competitive network (see [6] and [87] for examples), less attention has been put on the optimal operations of a network. In fact, the operational performance (i.e. efficient cost) of container terminal network is one of main factors for freight forwarders to route cargo through the chain of seaports, hence port operators always seek creative ways to decrease the cost charged to the customers [81]. In [49], the general movement of cargo within a terminal and between terminals are modeled into a integer linear programming (ILP) to minimize the network operational cost.

So far, the network operations optimization is driven by the shipping liners. As discussed above, an optimal network operations involve not only the liners but also the operators [59]. As one of the contributions of this chapter, we study container terminal network performance under heterogeneous distributed operational BCAP policy. The operational optimization policy in each terminal is confined only to the seaside operations problem. Hence, only berth and quay crane allocation is considered in each seaport. We have two valid reasons for this research boundary. First, shipping liners which call to a seaport deal greatly with seaside operations. Second, an effective BCAP will contribute hugely to the whole performance of the terminal, since bottlenecks often occur in this sub-system, where the discussion is provided in [18, 40, 48, 66]. The bottlenecks in berthing process has even more important consideration, since QC is usually the most expensive equipment in the terminal [13, 48]. Therefore improvement in BCAP will greatly profit the overall operational performance of a single terminal, and it has been explained extensively in [53, 82, 87] that terminal network's performance is heavily affected by the performances of each terminal in the network.

5.2 Preliminaries on BCAP modeling & optimization

As have been explained in Sub-Chapter 5.1, we seek optimum BCAP strategy in each terminal in the same seaport network. The strategy uses our recent work in [13] where DES models are developed for BCAP. The DES modeling includes dynamics in the berthing process, for instance the real-time changing of ship arrivals. The dynamics are not incorporated in the state-of-the-arts BCAP models in [37, 40, 52], and to include ship arrival variations, some probabilistic parameters are instead used while in fact the probems are later re-casted into static mixed-integer-linear programming (MILP) models.

The ability to capture the dynamics are essential, because in container terminal networks, a scheduled vessel can arrive later due to delay in the previous seaport. In [13], we also did a real-life experiment and the arrival of some ships indeed deviated from the original plan. In static problems ([37, 40, 52]), the MILP needs to be re-run with new set of inputs, while in [13] the set will dynamically alter. To solve the DES model, we propose an MPA to seek the optimal berth and QC allocation. In particular, we compare the MPA with operations research (OR)-based approach in [37, 40] where the former method shows improvement, as well as cost reduction in the real-life experiment. We also mathematically analyze the novel allocation method in [14] where the control inputs can be obtained by our proposition.

We will discuss the description of the berthing and loading/unloading process model which is based on the works in [12] and [13]. We will briefly discuss this model as well as the MPA-based BCAP policy in this sub-chapter. Interested readers are referred to [13] for detail of the model and approaches. In the following, we will provide an intuitive explanation of the model, as well as, the dynamic optimization method.

5.2.1 Dynamical discrete-event system models

At every discrete-event time step, we determine two control variables, namely the ship allocation to the selected berth position and the number of QCs allocated to every berth position.

The models are event-triggered, where the event-time step is incremented every time a ship finishes loading/unloading containers. The allocated ship is taken from a dynamic set of ready to-be-berthed ships where a number of ships may arrive or leave from the set during the computation process. Every ship in this set has its own arrival and operations time.

In the discrete-event systems (DES) model, we employ three state variables, namely berth starting time, berth operations time, and berth finishing time. The BCAP model has been validated in real-life field environment as in [13], where the state variable of berth finishing time is used for validation.

5.2.2 Cost functions

The cost functions used to evaluate the decisions are based on the operations cost and waiting cost. The operation cost is obtained based on the operation cost per unit of time multiplied by the total operations time, which is defined by the time spent by ships for completing their loading and unloading operations. The waiting cost is given by the waiting cost per unit of time multiplied by the total waiting time, which is the total idle time spent by every ship before it berths.

5.2.3 BCAP strategies

Using the aforementioned dynamics and cost functions, two strategies are available and will be considered in this chapter. They are the traditional FCFS (first-come-first-served) & DBQA (density based quay crane allocation)- based BCAP and the MPA-based BCAP method.

FCFS & DBQA-based BCAP strategy

This strategy is the most common method used in container terminals because of its easiness and fairness to the ship. At every time step we allocate a ship with the earliest arrival time; hence the name FCFS is originated from. Afterward, the number of QCs are allocated proportionally to the densities of containers in the berthed ships. The density is defined by a proportion of the number of containers in a particular ship to the number of containers in the entire berth positions.

MPA-based BCAP strategy

As opposed to the traditional method, the MPA-based strategy looks into the future where we compute an optimal allocation for a given discrete event time horizon. At each event time, the method calculates optimal decisions/allocation for a given event time horizon and subsequently, we implement only the decisions/allocation for the current event time step. The process is recursed in a rolling horizon manner until all ships from the set have already been scheduled, or the computation reaches the termination time. In [13], it has been shown, both numerically and experimentally that the performance of MPA-based BCAP policy always outperforms the genetic algorithm (GA) and the hybrid particle swarm optimization (HPSO) based BCAP methods.

5.3 Container terminal network

Container terminal network problems are presented in this sub-chapter, and firstly we would like to discuss the modeling setting used in this research as shown in Figure 5.1. A seaport network consists of many terminals, where each terminal owns equipment (berth positions and QCs). We rank the important seaports with models from [54], which generally based on transportation cost between two connected terminals and will later be explained in Sub-Chapter 5.3.3.

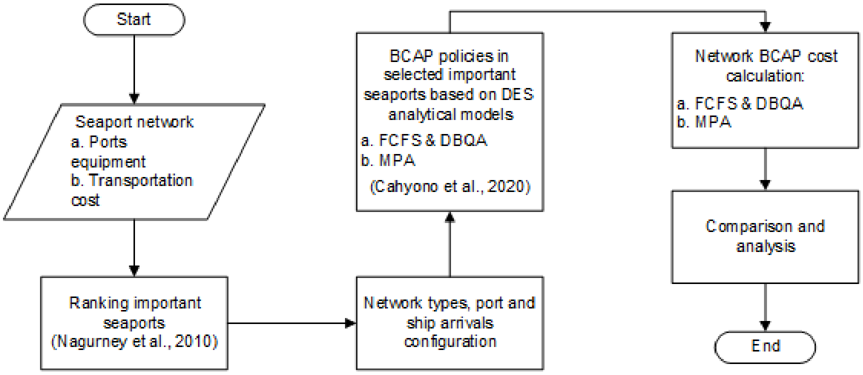

Figure 5.1: *Modeling seaport network methodology*

In each container terminal, we determine the BCAP policy, namely the berth positions and QCs based on DES analytical models from our recent works in [13]. The reasoning of this has been intensively discussed in Sub-Chapter 5.1. The DES applied in this research is not to be confused with discrete-event simulation, which are popular in logistics and based on random variate generation from some probabilistic functions (see [78]). We use instead DES framework as in [19] which is based on analytical models from series of discrete-events in the operations systems.

We then find the solution (berth and QC allocation) from the DES-based BCAP models. We compare the traditional FCFS & DBQA with our novel MPA method in [13]. To further differentiate with discrete-event simulation in [78] whose solutions are obtained from logical rule-based algorithms, the MPA is based from model predictive control (MPC) whose preliminaries mathematical efficacy is shown in [14].

The BCAP models are provided in [13]. In summary, the decision variables in each discrete step k are $u(k)$ and $v_b(k)$ which are the berthing position and number of QC allocated to an incoming ship, respectively. The incoming ship is from a dynamic set of arrivals $\mathcal{S}(k)$. One of the main differences with the BCAP-static models in [37, 40, 52] are the ability of the DES model in [13] to record the state variables in each k, namely $z_b(k)$, $y_b(k)$, and $x_b(k)$ which are the starting time, operations time, and finishing time, respectively. The allocation are evaluated with a cost function $J(x, u)$, which consists of operational and waiting cost.

To give clearer understanding, let us first consider a simple network as presented in Figure 5.2. In this figure, the network consists of only two seaports, namely "Port A" and "Port B". Each seaport has infrastructure (berth positions) and equipments (QC), whose numbers can be different. In Figure 5.2, container vessels travel back and forth between the two seaports. The transporting vessels are members of

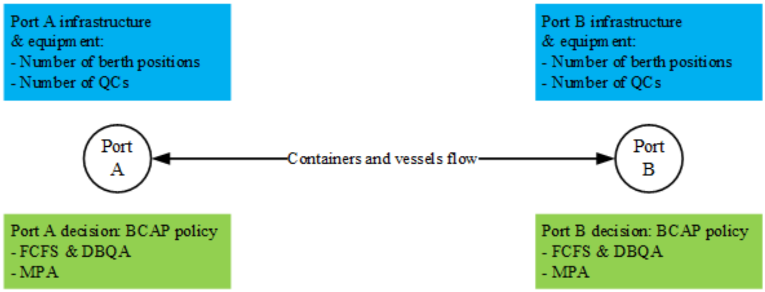

Figure 5.2: *A simple network with two seaports. Each seaport has its own number of berth positions and QC, where both seaports have to decide which local policy that needs to be implemented.*

dynamic sets, which means that they are real time and not pre-determined. A vessel may arrive at the seaport later than the scheduled time because of delayed operations in the previous terminal.

Given these settings, each container terminal has to decide what BCAP policy it should use. Two possible options are available, i.e. the traditional FCFS and DBQA or the optimal MPA-based policy as explained before in Sub-Chapter 5.2. In this work, we also assume that the ships use the optimal traveling speed as assigned by shipping liners. Hence, speed optimization is not considered in this chapter.

One of our hypotheses is that by implementing the optimal policy to a selected number of seaports, the whole network cost will decrease significantly compare to implementing optimal policy at arbitrary seaports.

In general, container terminals that use our algorithm will have efficient operations according to the simulation and experimental results as reported in [13]. Both of the simulation and experiment show that the MPA performs better than the traditional FCFS & DBQA method. Therefore, waiting times in those selected terminals can be reduced. This will further affect the other terminals, whether using MPA or traditional FCFS-DBQA methods. The low waiting times can minimize delay in ships' arrival times, which, in turn, reduces the total operations' time of the whole network.

5.3.1 Network types

We consider three standard network topologies in our investigation. They are ring/loop network, hub & spoke network and mesh network as illustrated in Figure 5.3[55]. In our simulation setup, we vary the number of nodes in each topology. Each seaport is represented by a node which has to determine what type of BCAP policy it will apply.

A ring/loop is typically used in a network in which shipping liners deploy its

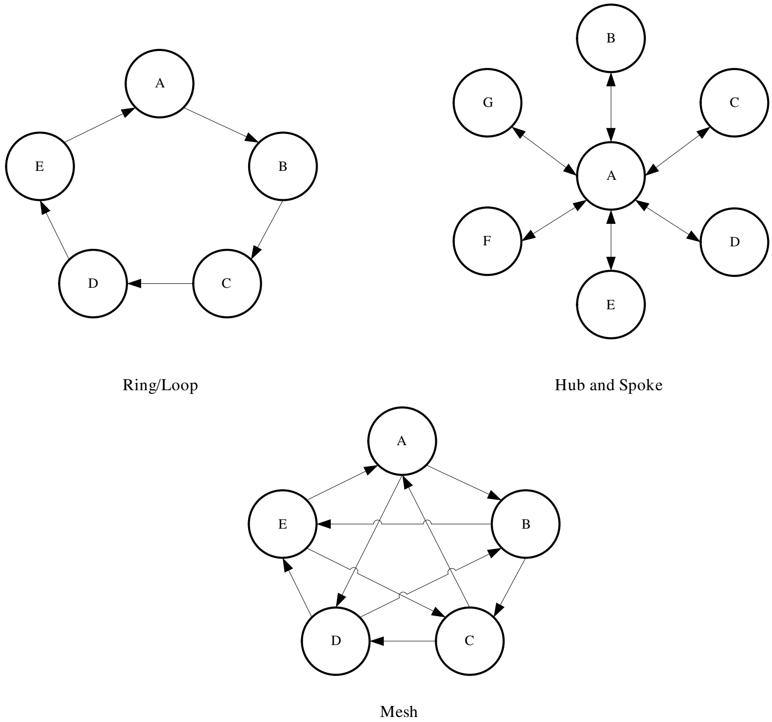

Figure 5.3: *An illustration of three network types used in this work. The arrows indicate the direction of the containers and vessels flow. The number of container terminals used later in the simulation does not necessarily be the same as in this figure.*

vessels to travel circularly from the port of origin, traversing the intermediate and end ports, and then back to the origin. This network type can be found when a terminal operator is the main stakeholder whose prime interest is to regulate such network. Although arguably the rarest type in maritime transportation, a circular network is a good example that represents the typical behavior of a maritime network, where the number of containers travel from origin to end likely stays the same. In addition, our specific case in Indonesia which will later be explained in the next sub-chapter also uses this type of network.

The hub & spoke is probably the most well-known network type, as it can be traced in some cases in South East Asia. One leading example is the network where Port of Singapore is the corresponding hub. In this case, big-size vessels from all parts of the world call to this hub. Small vessels from surrounding regions also call regularly to the Port of Singapore. They act as feeders for the hub seaport. An example of the partners are Port of Tanjung Priuk in Indonesia, Port of Klang in Malaysia, and Port of Laem Chabang in Thailand.

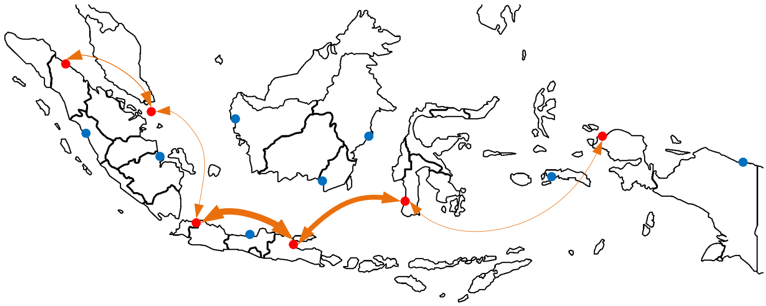

Figure 5.4: *The maritime transportation in Indonesia. Red nodes and orange arrows indicate that particular seaports and networks are part of the "Indonesian sea highways" project. The weight of the arcs is an exact visualization of demand volumes between two seaports in the particular link. Blue nodes mean that particular seaports are among Indonesia's main container terminals, but not parts of the project.*

For the last network type, the mesh network represents network when no seaport serves as the "formal" hub. Therefore, the traffics look like scattered. This network type reflects almost all container flows in Indonesian maritime industry. For an example, although Port of Tanjung Priuk is the largest seaport owned and partially operated by "Pelindo II", it is not the focal point of trade like in Port of Singapore. Considerable and balanced amounts of container are transported among other seaports, where they are ran by different terminal operators.

5.3.2 The "Indonesian sea highways" case

For a real case study, we present the case of "Indonesian sea highways" which is initiated by the Government of Indonesia. The map of Indonesian maritime industry is presented in Figure 5.4. Since every edge in this network is bidirectional, it is a particular case of the ring/loop network.

Indonesia is an archipelago where its vast area is not yet covered with a reliable sea transportation [85]. Therefore, from almost 50 container terminals spread in the country, the government have selected six prominent seaports whose geographical positions are deemed strategic to serve the national trade flow. Those seaports are Port of Kuala Tanjung in the almost western most province namely North Sumatera, Port of Batam which is less than 10 nautical miles to Port of Singapore, the busiest Port of Tanjung Priuk in the country's capital in Jakarta, the second busiest Port of Tanjung Perak in East Java, the Makassar seaport in South Sulawesi and the last one the Port of Sorong in West Papua.

It can be seen in Figure 5.4 that maritime trade in Indonesia is not balanced. A significant volume of the trade lies in the center, where Java island, the center of economic activities is located. The weight of the arcs itself is an exact visualization of demand volumes between two seaports in the particular link. Since the data of the demand is not available, we use approximation from volume of all types of cargo transportation accordingly. The volume represents the trade flow in which each particular seaport lies. This assumption is reasonable since the profile of container transport volume will not differ too much. The aforementioned demand data is shown in Table 5.1. In the simulation setting, it will further be explained that the raw demand is transformed into number of ship arrivals, where every arriving ship has its own load (TEU).

Table 5.1: *Transportation demand between each region which represents each seaport [85].*

Origin-Destination	*Transportation value (ton/yr.)*
North Sumatera - Riau Islands	4,225,156
Riau Islands - North Sumatera	4,521,140
Riau Islands - DKI Jakarta	1,166,743
DKI Jakarta - Riau Islands	1,096,483
DKI Jakarta - East Java	33,779,731
East Java - DKI Jakarta	32,284,745
East Java - South Sulawesi	37,503,830
South Sulawesi - East Java	13,274,245
South Sulawesi - West Papua	949,343
West Papua - South Sulawesi	631,214

The more detail illustration of trade flows in the ”sea highways” is shown in Figure 5.5, where now there are two trade flows in each sub-network between two seaports. The conclusion from this figure is the same as that from the previous picture, but we can now see clearly that the ”Tanjung Perak-Makassar” link is heavily unbalanced. It is also important to note that each container terminal has different size, meaning that different number of berth positions and QCs are owned.

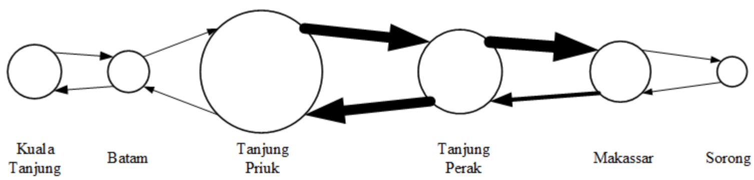

Figure 5.5: *Six seaports of the ”Indonesian sea highway”. The diameter of each circle represents the size of the seaport. The directed arcs show the direction of container flows among seaports. The weight of each arc represents the volume of container flow.*

The ultimate goal of the government is to enhance the performance of the ”sea highways”, which is still an open measure. Several interesting considerations are efficiency of the network, which is highly related to operational cost; and how to promote trade balance especially to the eastern part of Indonesia. Speaking of balance of trade, it is of course an aspect that the government cannot fully control. As Indonesia adheres to the free trade policy, hence, the maritime industry cannot be forced to ship containers more to the east than their current operations. This brings us to the research motivation as given in the Introduction where terminal operators are interested to optimize the network performance with minimal cost while maintaining the autonomy of every terminal and every liner.

By imposing a certain policy to a particular seaport, the government can further analyze what the effects to the network are. It is also useful to analyze what priority should be given if only a limited number of seaports can use the optimal BCAP policy, but significant improvement from the whole network is still desired. This is an interesting decision since each terminal has different specifications and each terminal has different constraints and willingness to apply such an optimal BCAP strategy.

5.3.3 Determining important seaports

Instead of applying the MPA-based BCAP strategy to arbitrary seaports, we can implement it to a number of important seaports. By this approach we are looking for methods to identify such sub-network so that the network's cost decreases significantly with only implementing the optimal BCAP policy to the terminals in this sub-network.

Each terminal's importance is determined using a modified ranking method as proposed in [54]. In the original literature, the importance is not directly defined on seaports, but instead on arcs connecting seaports. For a given network, the entire arcs' importance are normalized, so their summation is equal to one.

In each arc, the importance is defined by the function of container flows, transportation cost, and capacity associated to this arc. The more containers flow and its associated transportation cost in a particular arc, the more important the connecting arc is.

For simplicity, in the basic network types, we assume that the transportation cost is the same for every arc. While in the Indonesian sea highways case, different transportation cost is obtained from historical trade data among provinces.

We present the model from [54] as follows. The cost function for link/arc i is given by

$$c_i(f_i) = t_i \left(1 + k \frac{f_i}{u_i}^{\beta} \right) \tag{5.1}$$

where the amount of flow, transportation cost and capacity in the link i are denoted by f_i, t_i, and u_i, respectively. We take the model parameters of k and β from the paper in [54] as $k = 0.15$ and $\beta = 4$.

The total cost of the i-th link is given by

$$\begin{aligned} \hat{c}_i &= c_i(f_i)f_i && (5.2) \\ &= t_i\left(1 + k\frac{f_i}{u_i}^{\beta}\right)f_i && (5.3) \end{aligned}$$

Using $\hat{c}_i$, we can define the importance/weight of each link i by

$$\alpha_i = \frac{\hat{c}_i}{G}, \tag{5.4}$$

where G is the total cost for the whole links in the network and is defined by $G = \sum_i \hat{c}_i$.

Based on α_i, we can rank the terminals importance as follows: Let us denote $\mathcal{A}_j$ as the set of all links/arcs where the j-th terminal/node is at their heads and tails. Correspondingly, the links that are taken into account are both of the incoming and outgoing ones. Then, each terminal's importance r_j is given by

$$r_j = \sum_{i \in \mathcal{A}_j} \alpha_i \tag{5.5}$$

and subsequently we can order or rank the terminals based on r_j where terminal with the highest r_j is ranked first.

5.4 Simulation setup

5.4.1 Container terminal configuration

We use hypothetical data for the three basic network types as explained in the last sub-chapter. In each network type, the distance between any two adjacent seaports is set randomly according to uniform distribution whose the lower and upper bound is 200 and 400 km, to reflect the actual distance in Indonesian seaport networks. All of the vessels are assumed to constantly follow the designated speed for container vessels, which is 24 knots.

For ring/loop and mesh network types, the number of nodes is fixed at 50 seaports. While for the hub and spoke, to reflect real practice, we limit the number of seaports only to 20, where one seaport serves as a hub and the rest as feeder seaports. In a mesh network, every single seaport may access multiple seaports, i.e., an adjacent seaport that is connected with a circular arc and other seaports with non-circular arcs as connectors. As one can see in Fig. 5.3, there are loops in the mesh network type, such as a sub-network that consists of nodes 1, 4, and 5 and the route is 1-4-5-1. Other possible loops are 5-2-4-5 and 1-2-3-1.

Table 5.2: *An example of a subset of the first 8 of 50 seaports configuration for the ring/loop and mesh network types.*

Seaport index	*Seaport type*	*Num. of berth positions*	*Num. of QCs*
1	Large	11	19
2	Medium	8	13
3	Medium	5	9
4	Small	3	6
5	Large	9	14
6	Large	11	12
7	Small	3	5
8	Medium	7	12

Every seaport has different number of berth positions and QCs. For the ring/loop and mesh network types, we divide 50 seaports proportionally into three categories:

1. Small. Number of berth positions is between 2 and 4. Number of QCs is between 3 and 8.

2. Medium. Number of berth positions is between 5 and 8. Number of QCs is between 6 and 16.

3. Large. Number of berth positions is between 9 and 12. Number of QCs is between 10 and 24.

When a seaport already falls into a particular size, its number of berth positions and QCs are determined randomly with a uniform distribution. A subset of seaports configuration for these two network types is presented in Table 5.2.

As for the hub and spoke type, the first seaport is the hub, where its number of berth positions and QCs are randomized between 24 to 30, and 45 to 50, respectively. This estimation is based on benchmarking from hub terminals such as Port of Singapore and Port of Rotterdam.

The remaining 19 feeder seaports follow the aforementioned procedure for the other two network types, in which the seaports are balanced into small, medium, and large categories. The subset of the first 8 of 20 seaports configuration is presented in Table 5.3.

We present the setup of ship arrivals used in the simulation. The set of ship arrivals for the three network types is divided into three load scenarios: reduced load, normal load, and heavy load. The first scenario is a set where the incoming ships arrive to the seaport in sparse inter-arrival times, and the loads are not high.

Table 5.3: *An example of a subset of the first 8 of 20 seaports configuration for the hub and spoke network type.*

Seaport index	*Seaport type*	*Num. of berth positions*	*Num. of QCs*
1	Hub	26	49
2	Medium	5	8
3	Medium	6	7
4	Large	12	21
5	Medium	5	8
6	Large	12	16
7	Small	2	4
8	Small	2	3

Table 5.4: *A subset of a dataset of the first 10 of 50 arriving ships for the reduced load scenario.*

Seaport index	*Arrival date*	*Arrival time*	*Load (TEU)*
1	1 March 2020	05:50 AM	1,829
2	1 March 2020	12:04 PM	1,952
3	1 March 2020	14:08 PM	2,891
4	1 March 2020	18:25 PM	1,750
5	1 March 2020	22:29 PM	1,674
6	2 March 2020	07:38 AM	1,149
7	2 March 2020	13:50 PM	1,709
8	2 March 2020	20:07 PM	2,079
9	2 March 2020	23:57 PM	1,697
10	3 March 2020	05:29 AM	1,478

On the other side, the heavy load scenario is set completely to the opposite of the reduced one.

In each dataset, we generate 50 ships arrivals. The entire ships arrive at the first seaport in the network. For the ring/loop network type, after the BCAP operations are completed in the first seaport, the ship starts its journey to the next adjacent seaport. While for the mesh type, a ship that has just finished its operations follow the following procedure:

1. The first ship goes to the second adjacent seaport that is connected with the current seaport through a circular arc.

2. The second ship goes to another seaport that is connected with a non circular arc. Since there are multiple possibilities, the destination seaport is selected randomly.

Table 5.5: *A subset of a dataset of the first 10 of 50 arriving ships for the normal load scenario.*

Seaport index	*Arrival date*	*Arrival time*	*Load (TEU)*
1	1 March 2020	02:26 AM	3,211
2	1 March 2020	07:17 AM	4,137
3	1 March 2020	09:17 AM	4,372
4	1 March 2020	14:08 PM	3,288
5	1 March 2020	18:59 PM	3,089
6	2 March 2020	02:29 AM	4,383
7	2 March 2020	06:16 AM	4,363
8	2 March 2020	11:58 AM	3,403
9	2 March 2020	15:04 PM	3,786
10	2 March 2020	21:47 PM	4,539

3. The subsequent odd numbered ships follow the procedure for the first one while the rest follows that for the second one.

For the hub and spoke type, every ship starts from the hub and then it is assigned to a random spoke for its next destination.

All simulation ends after 60 days and we assume that all seaports in the networks work 24 hours per day and 7 days a week.

We generate 100 datasets for each load scenario. Therefore, there are 300 datasets available for each network. The ship arrivals data is generated based on log-normal distribution to avoid negative value of inter-arrival times. We obtain the parameters from a real-data as in [12], where from the 29 ship arrivals, the mean and standard deviation are 5.96 and 0.63, respectively. Based on interviews with the operators who work at the seaport of Tanjung Priuk from which the data was taken, the arrivals can be categorized as the reduced one. Hence, we generated our parameters as follows.

- The reduced load scenario has mean of 6.0 and standard deviation of 0.6.
- The normal load scenario has mean of 5.5 and standard deviation of 0.5.
- The heavy load scenario has mean of 5.0 and standard deviation of 0.4.

We also generate the ships loads (TEUs) based on the uniform distribution as follows.

- The reduced load scenario has a lower bound of 1,000 TEUs and an upper bound of 3,000 TEUs.
- The normal load scenario has a lower bound of 3,000 TEUs and an upper bound of 5,000 TEUs.

Table 5.6: *A subset of a dataset of the first 10 of 50 arriving ships for the heavy load scenario.*

Seaport index	*Arrival date*	*Arrival time*	*Load (TEU)*
1	1 March 2020	01:47 AM	9,607
2	1 March 2020	04:16 AM	9,362
3	1 March 2020	06:25 AM	6,127
4	1 March 2020	08:55 AM	5,300
5	1 March 2020	10:26 AM	5,540
6	1 March 2020	14:17 PM	8,713
7	1 March 2020	16:19 PM	8,929
8	1 March 2020	18:51 PM	5,712
9	1 March 2020	22:11 PM	5,881
10	2 March 2020	00:40 AM	5,818

- The heavy load scenario has a lower bound of 5,000 TEUs and an upper bound of 10,000 TEUs.

The basis to categorize ships loads is derived from common container vessels classifications. The feeder ships'capacity are usually up to 3,000 TEUs. The Panamax ships are up to 5,000 TEUs. While the Post-Panamax is a generation of ships that are able to carry 10,000 TEUs. There are in fact Super-Panamax and Mega-Panamax ships. But since the load range is too wide, we do not consider them in this research.

5.4.2 The sets of ships arrivals

We set the first ship to arrive on 1 March 2020. The datasets for each load scenario are the same for the three network types.

To give an illustration, we provide one dataset example for each load scenario. The subset of the datasets is presented in Table 5.4, 5.5, and 5.6 in the Appendix, respectively for the reduced, normal, and heavy load scenarios. The probability density functions of these datasets are depicted in Figure 5.6. Based on the 100 datasets in each scenario, the average hours of inter-arrival times are 18.41, 15.69, and 4.86 for the reduced, normal, and heavy load scenarios, respectively.

It is important to note that the ship's loads as in Table 5.4, 5.5, and 5.6 in the Appendix are only for the first seaport. In the subsequent seaports, the ships also have container loads that we define randomly. In our simulation, we have prepared sets of ships load in each subsequent seaport.

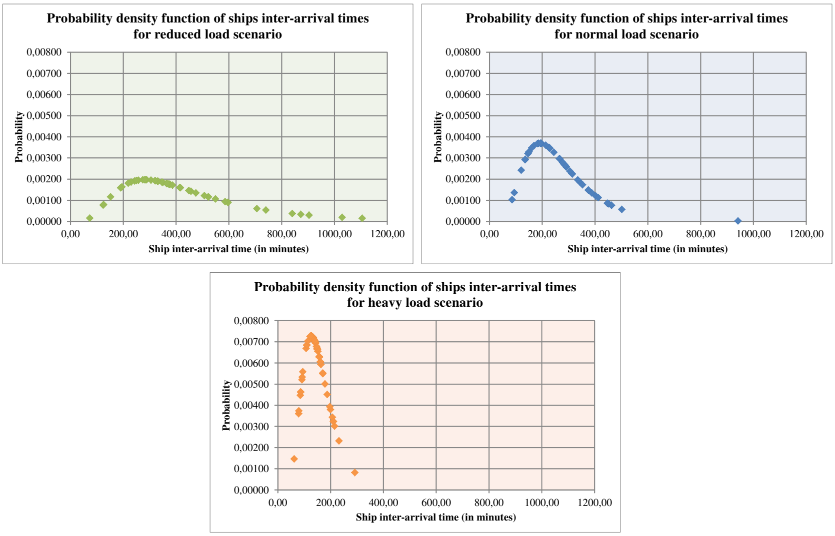

Figure 5.6: *The probability density functions for the three ship arrivals scenarios for the ring/loop, mesh, and hub & spoke network types. The horizontal axis shows the ship inter-arrival time (in minutes) and the vertical axis shows the probability (non-cumulative).*

5.5 Simulation results

5.5.1 Simulation results of basic network types

Since there are three network types and three load scenarios, where each scenario has 100 datasets, we have generated 900 cases. In every case, we simulate two different approaches in the deployment of MPA-based BCAP. In the first one, we randomly select a number of seaports at which the MPA policy are implemented. In the second one, our selections are based on the seaports' importance as discussed before in Sub-Chapter 5.3.3.

The results for the normal load scenario for the three network types are presented in Figure 5.7 which provide a summary of network cost for each network type and load scenario plan from 100 simulations each. Note that in all of these simulations, the most important seaports in each scenario are not necessarily the same terminals, because the datasets for each simulation run are different.

For both different deployment methods, the results are intuitive. The network cost in both cases decreases monotonically as more seaports use the MPA-based BCAP methods. More importantly, it is shown that by implementing the optimal policy only to the important seaports, we obtain more efficient cost reduction than the arbitrary-chosen seaports. For instance, in the ring/loop network, the costs from applying MPA to 10 and 20 important seaports are comparable to randomly

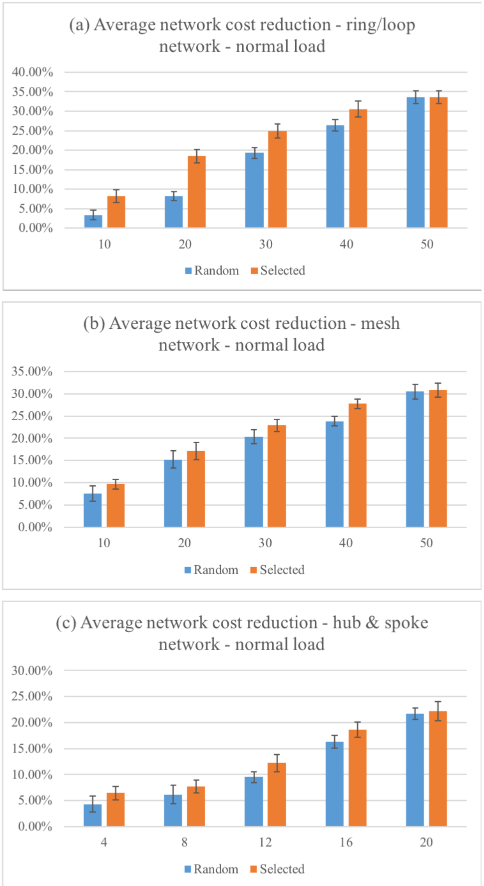

Figure 5.7: *Average network cost reduction from 100 datasets with normal load scenario for the (a). ring/loop network; (b) mesh network; (c) hub and spoke network. The horizontal and the vertical axis are the number of seaports using MPA and the average network cost reduction in thousand Euro, respectively. The reduction is calculated from a basic network where no seaport uses MPA. Vertical line at each average cost point is the standard deviation error bars.*

implementing the optimal policy to 20 and 35 seaports. Although not as salient, we also see the similar behavior in the other two network types.

From the practical perspective, applying MPA-based BCAP policy only to important seaports is an advantage to the terminal operators as well. With the MPA-based BCAP, the order of the ships' arrival is not necessarily the same as the order of the berthing. Although applying MPA to more seaports is beneficial to the whole network operations, it may not be the case for the shipping liners. Therefore, limiting the optimal BCAP policy only to few seaports reduces dissatisfaction among liners.

5.5.2 Simulation results of the Indonesian sea highway case

We repeat the same procedure for the Indonesian sea highway case, in which we determine important seaports as given before in Sub-Chapter 5.3.3. The weight of all links in the network are presented in Table 5.7.

Table 5.7: *Weight value of each link in the network as described in (4) in Subsection 3.3.*

Link no (i)	*Link*	*Weight value* (α_i)
1	Kuala Tanjung-Batam	0.039
2	Batam-Kuala Tanjung	0.040
3	Batam-Tg. Priuk	0.010
4	Tg. Priuk-Batam	0.011
5	Tg. Priuk-Tg. Perak	0.261
6	Tg. Perak-Tg. Priuk	0.256
7	Tg. Perak-Makassar	0.250
8	Makassar-Tg. Perak	0.129
9	Makassar-Sorong	0.003
10	Sorong-Makassar	0.001

We then rank the importance of each terminal as given in the Table 5.8. It is obvious that the two most important seaports are Tanjung Priuk and Tanjung Perak, since trade flows are heavily concentrated in network arc involving these two seaports. But contrary to the generally accepted stance, the results in Table 5.8 shows that the most important seaport is Tanjung Perak, instead of Tanjung Priuk in the capital city of Indonesia, since it connects the heaviest container transportation in Indonesia. Based on the importance in Table 5.8, the simulation result can be seen in Table 5.9.

Table 5.8: *Importance value of each terminal in the network as described in (5) in Subsection 3.3.*

Node no. (j)	*Terminal*	*Importance value* (r_j)	*Rank of importance*
1	Tg. Perak	0.896	1
2	Tg. Priuk	0.528	2
3	Makassar	0.382	3
4	Kuala Tanjung	0.079	4
5	Batam	0.061	5
6	Sorong	0.004	6

By applying MPA only to seaports in rank 1 and 2 (Tanjung Priuk, Tanjung Perak, Makassar), the network's cost reduction does not much differ when we apply MPA

Table 5.9: *Network cost of each different strategy.*

Scenario	*Cost (Mill. Euro)*
All seaports use FCFS	23,742
All seaports use MPA	20,292
Rank 1 (Tg. Perak) & 2 (Tg. Priuk) use MPA, the rest use FCFS	20,695
Rank 1 (Tg. Perak) use MPA, the rest use FCFS	21,245
Rank 2 (Tg. Priuk) use MPA, the rest use FCFS	21,791
Rank 3 (Makassar) use MPA, the rest use FCFS	23,432

to the all six seaports. Even by only applying MPA to seaports in rank 2 (Tanjung Perak and Makassar), the cost reduction is still quite significant. We observe that the optimal strategy can be applied only to number of selected seaports.

5.6 Discussion

Based on the simulation results with three network topologies (ring/loop, mesh, and hub & spoke), we consistently observed that the more seaports implementing the optimal MPA-based BCAP policy, the lower network cost is. We have analyzed three different network types with different ships routing rules where similar results are observed. It shows that local actions (distributed optimization of arbitrary number of container terminals) lead to an improved global performance (a competitive network with reduced network's cost). It means that for a large container terminal network, it is always a benefit to apply the optimal BCAP method to as many terminals as possible.

To deal with this dilemma of ships' dissatisfaction if they are assigned to the berth position later than their arrival time (with the MPA method), we also propose how to implement the optimal BCAP policy only to some important seaports. From the simulation results, we obtain the second conclusion where significant cost reduction is still obtained with fewer number of seaports than those seaports chosen arbitrarily.

In practice, this method is logical. From the simulation results, we see that the important seaports are the large ones, where the terminal operators usually have strong authority to impose such policy. This proposition is further verified with a study case from the Indonesian container terminal network, where we observe that even by applying MPA policy only to the two most important seaports, we still get significant cost reduction.

Chapter 6

Modeling and optimization of integrated container terminal operations

Chapter 6

Modeling and optimization of integrated container terminal operations

We present in this chapter a dynamical modeling of integrated (end-to-end) container terminal operations using finite state machine (FSM) framework where each state machine is represented by a discrete-event system (DES) framework, which is the extension of I-BCAP modeling framework in Chapter 3. The asynchronous DES operations in the I-BCAP is applied not only to the seaside, but to the whole sections in container terminals, which are the operations of quay cranes (QC), internal trucks (IT), and yard cranes (YC) and also the selection of storage positions in container yard (CY) and vessel bays. The QC and YC are connected by the IT in our models. After research motivation is presented in the first chapter, the explanation of container terminal operations is provided, which will serve as the foundation of the dynamical mathematical models. Afterwards, the allocation strategy of the models will be given. Subsequently, we also describe the simulation set-up and results. The simulations use the MPA method from our previous research and the benchmarking methods from the state-of-the-art literature. Finally findings are provided in the discussion.

6.1 Introduction

Container terminals have been important nodes in global maritime transportation network for the past six decades. The standardization and low cost of container boxes have made them the foremost choice of transportation means in the international trade [64]. The trend of containerization growth has been twice the growth of the total world and maritime trade for the past decade [76]. The increasing demand in container trade has made the terminal operators to put efforts to optimize and streamline their operations in order to guarantee an efficient service to the shipping liners, as well as the inland shippers and consignees as the terminal's main customers [75].

In general container terminal, a number of ships can dock at various berth positions along the seaside and several quay cranes (QC) can be assigned to every berthed ship for loading and unloading containers. There are internal trucks (IT) waiting beneath the QC and they transport the containers to some specific destinations at container yard (CY). On the other way around, IT also deliver containers from CY to QC, which will load them to the pre-determined stacking point at the vessels. The CY is divided into two parts. The section which is closer to the berth is dedicated to the export (hence, outbound) containers, and the other part is for the import/inbound ones. The containers are stored in the CY and several yard cranes (YC) re-allocate them internally within the CY (known as housekeeping/re-handling) or load/unload them to/from external trucks (ET), which finally deliver the containers to their owners (consignees) in the factories or warehouses.

Container terminal operations are typically divided into three main areas, namely seaside, storage, and transfer [75, 76]. The seaside is a section where incoming ships arrive at the seaport and the the terminal operator allocates berth positions and QC(s) to each vessel. This is known as the integrated berth and crane allocation problem (I-BCAP), where a detailed review is provided in [18]. A ship's load is represented by its number of containers, where each box of container is measured as a twenty feet equivalent unit (TEU), approximately six meters long, while the longer container is forty feet (FEU). The typical decisions in BCAP are allocation of berth positions and QCs to the incoming ships [18]. In more detailed levels, the terminal planners determine the exact positions of outbound containers should be at the vessels, which usually identified by bays and tiers [18].

The storage operations is the management of containers in the CY and we refer to [16] for a review on this specific operations. A container position in the CY is defined by its row, bay, and tier, which is comparable to $x - y - z$ axis in the Cartesian systems. There are three typical decisions in this sub-operations. Firstly, the positions where a group of containers should be stored. Secondly, the allocation of YC to handle them from/to IT. The container placement at the right positions in the CY is important. Thirdly, if an ET comes to the CY for pick-up operations, and the targeted container is not in the top tier, the terminal operators has to assign YC to re-arrange the containers positions. This situation therefore leads to the third process, which is known as housekeeping/marshalling. Due to its cost-inefficiency, marshalling is highly avoided in terminal operations [16].

The seaside and storage sub-systems are connected with the transfer operations, whose review is discussed in [17]. In this sub-operations, transporters handle the container delivery between QC and CY area. Common transporters in container terminals are rail-mounted gantry crane (RMGC), rubber-tyre gantry crane (RTGC), reach stakers (RS) and internal trucks (IT). In this chapter, we will focus on the IT. The decisions in this sub-sytems are the allocation of IT to serve as the link between QC and YC, and vice versa. The scheduling of IT to QC or YC is also important, therefore we found significant works of vehicle routing problems (VRP) in the container terminals. The most common method for scheduling is currently performed in daily basis, where the schedule of IT is created at the beginning of each day, based on ships' arrivals, outbound containers' stowage plans, and inbound containers' external delivery. In this setting, variations of these three inputs are often neglected [17].

In accordance with the complex seaport operations, the aim of the terminal operators is to operate the container terminal efficiently in the least possible cost with minimal dissatisfaction level from its customers [33, 36, 64, 76]. The purpose of the terminal operators can be summarized into container delivery whose destinations are to: 1) CY, for the inbound boxes, and 2) vessel, for the outbound boxes [33, 36, 44]. The storage configuration of the inbound containers in the CY and of the outbound containers in the vessels are known as the storage plan and stowage plan, respectively [24, 94].

The complexity of container terminal operations has been studied extensively in literature and some literature reviews in this topic are presented in [75, 76]. The container terminal operations in the three sub-systems as above are dependent to each other. For instance, the exact deployment of IT can only be executed after the QC and YC allocation are definitive. For allocating cost-effective QC and YC themselves, the detailed knowledge on the schedule is required.

To make an optimal operations planning, the entire sub-systems in the terminal have to be considered [33, 36, 44]. However, in practice, the complexity of the operations makes the state-of-the-art research in container terminal operations limited only to each sub-system [75, 76]. Excellent reviews for the seaside, storage, and transfer operations are provided in [16, 17, 18]. To the best of authors knowledge, there is no literature review dedicated for the container terminal integrated operations.

In the practical level, the terminal operators almost exclusively rely on the non-integrated decision making process to produce planning in each sub-system of the terminal operations. The QC, YC, and IT are allocated according to the first-come-first-served (FCFS) criteria, which does not guarantee the optimality of the planning [16, 17, 18, 64].

There are indeed several works on the integrated terminal operations such as in [2, 4, 11, 33, 36, 44, 89, 90, 91]. Although the end-to-end operations process is modeled in [2, 4], the problems are more in the tactical level, which relate to resource allocation. In these papers, resource allocation is expressed as percentage of servers (equipment) capacity to transport containers to the subsequent server. A similar tactical approach is found in [91] where dynamical models are used to depict the integrated operations in dry terminals. In this research, two control variables are considered, the in and out stacking rate of of the stacking carriers in the terminals. In [36], a genetic algorithm (GA)-based pseudo code is used to create the planning. The non-existence of mathematical models makes the test-ability of the problems can not be guaranteed.

One noticeable drawback of the state-the-art models is the static approach, in which the inputs are known apriori. In the state-the-art (static), operations research is the main technique used in the container terminal operations modeling [16, 17, 18, 75, 76], and linear programming (LP) can be applied for solving the equipment allocation in the seaport. One assumption of LP is the inputs have to be deterministic, which implies that the changing of inputs during solution searching is not permitted

For instance, in [33], the IT are pre-determined before the schedule of QC, IT and YC are solved through linear programming technique. In fact, during terminal operations, there are chances of disruption of the equipment conditions [17, 64]. This dynamic behavior is not yet represented in [33]. The discussion of modeling approach in seaport operations is heavily discussed in [13], which concludes that the dynamical approach is more suitable to capture the changing environment. A discrete-event system (DES) model is developed in [13] and it is important to note that the DES here is not the same terminology that commonly used in operations systems, where some probabilistic functions are employed to represent the random behavior of systems. The latter approach is know as discrete-event simulation as exemplified in [78].

Some works in terminal operations have tried to incorporate dynamical modeling as studied in [11, 89, 90, 91]. In these two works ([11, 89]), a partially dynamical aspect in the lower-level controllers is included, which is the detail movement of QC, YC and IT/rail in terminal. However, prior to this step, the allocation of the three equipment to berth and CY are solved from a static model so-called the higher-level controllers, where a similar concept of LP is used to find the solutions. In [11, 89] the mixed-integer linear programming model (MILP) technique is used to find the optimal allocation of QC, YC, ASC, and rail.

This setting does not completely capture the real equipment allocation problems in the terminal. In the beginning of each planning period, the terminal operators allocate the QC, YC, and IT based on available information in the terminal, namely vessel arrivals and CY storage status. But later on, the equipment detail scheduling in [11, 89] is handled via a linear programming technique which in reality is static. The changing in berth and CY configuration will be seen by the terminal operators as new possible storage/stowage plan, and will subsequently change the entire previous allocation and scheduling of QC, YC, and IT. This dynamics behaviour in container terminal operations is not considered in the modeling framework in [11, 33, 89].

As opposed to the static modeling, we employ a dynamical model based on discrete-event systems (DES) in this chapter. The DES framework is suitable for describing the terminal operations problems, since each job completed by either QC, YC, or IT can be seen as a discrete-event time step [19]. For terminal operations, the DES modeling framework has been successfully applied to a sub-system of terminal operations, namely the berth and and quay crane allocation [12, 13].

As stated in [12, 13], the generalization of the work to the complete terminal operations remains open. The lack of dynamical models in container terminal operations as mentioned before has motivated us to study dynamical modeling in integrated terminal operations. In particular, we also use finite state machine (FSM) framework, where the DES formulation is represented in each of the state machine formulation. As discussed in [51], FSM framwork incorporates a set of several discrete variables. In this regard, the FSM suits our problems where the complex systems of terminal operations can be represented by discrete variables.

The rest of this chapter is organized as follows. After research motivation is presented in the first sub-chapter, Sub-Chapter 6.2 is devoted for the explanation of container terminal operations, which will serve as the foundation of the dynamical mathematical models presented in the Sub-Chapter 6.3. The allocation strategy of the models is given in Sub-Chapter 6.4. Subsequently, we describe the simulation set-up and results in Sub-Chapter 6.5. The simulations use the MPA method from our previous research and the benchmarking methods from the state-of-the-art literature. Finally concluding remarks and possible future works are discussed in Sub-Chapter 6.6.

6.2 Container terminal operations

We present the generalization of integrated container terminal operations framework in this sub-chapter. The framework will serve as the basis for the dynamical models development in the Sub-Chapter 6.3.

6.2.1 General assumptions

Based on the previous studies in [2, 33, 36, 89], we summarize the integrated container terminal operations which is defined as the sequential series of processes to unload inbound containers from ships to CY, and correspondingly, the set of processes to load outbound containers from CY to vessels, where in both types of operations, the vessels are already allocated berth positions in the terminals. The schematic diagram of an end-to-end container terminal operations is depicted in Figure 6.1.

This chapter discusses the loading and unloading and loading processes in container terminals which will be focused for the inbound containers. The reasoning for the omission of outbound containers is presented in Sub-Chapter 6.3. The receiving and delivery operations, which are performed by the ET, are neglected in this research. This limitations operations framework can also be found in [11, 33, 89] due to complexity of ET operations, which includes random aspects of time to pick and deliver containers to/from hinterlands [50].

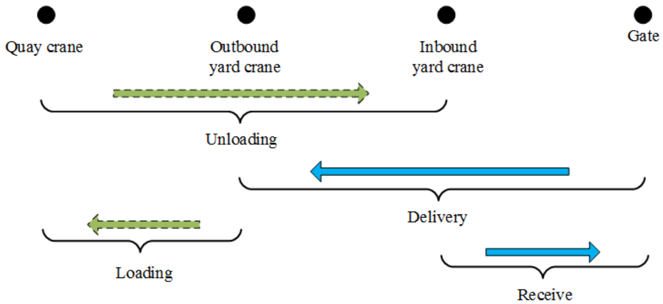

Figure 6.1: *An illustration of an integrated container terminal operations. The bracket refers to the two (begin and end) parties which are involved in every process. The arrows show the direction of container flow. The green-dashed arrow refers to transportation process of a container by an IT. The blue-solid arrow represents the transportation of a container by an ET, which is not considered in this work.*

Regarding the handling operations of containers, the goals of the terminal operators are to, firstly, locate the inbound containers into import sub-blocks in CY, which is known as the unloading process, and secondly, place the outbound containers into vessels, which is reversely the loading process. The output the first and second goals are the CY's storage plan and vessel's stowage plan, respectively. Examples of operations and modeling for CY's storage and vessel's stowage plan are discussed in [24, 94], and the illustrative example is given in Figure 6.2.

In a ship, the smallest unit to store containers is the sub-bay, whose capacity is more than 5 TEUs [94]. A group of several sub-bays is the bay. The containers in the seaside are handled by QCs. Some QCs work on several berthed-ships, and in practice the QCs do not have some specific working areas, as long as their movement do not interfere among each other [18]. In this research, we assume that a bay at a vessel is allocated by a QC which handles the container from the beginning of unloading/loading until finished.

A container yard consists the areas for the outbound and inbound containers. The smallest unit to place containers in the CY is the sub-block, whose capacity is more than 20 TEUs [94]. A group of some sub-blocks in the same ordinate is defined as a block, where a set of blocks in physically marked region is the preferred area, which is usually designated for specific customers (shipping liners). A yard crane is assigned to some specific specific preferred areas. An inbound YC cannot move to the outbound CY preferred areas, and vice versa. We assume in this research that a YC is allocated to a specific block in the CY. Figure 6.2 shows an illustration of a CY configuration.

A storage plan is the set of decisions that the terminal operators know to which CY's sub-blocks the inbound containers will be allocated. On the other

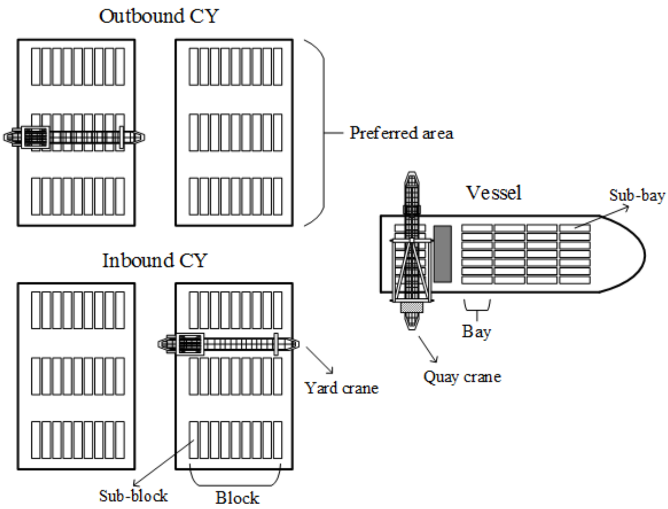

Figure 6.2: *A top view layout of the CY (left) and vessel (right). Both of CY and vessel serve as temporary storage in a container terminal. CY is usually divided into inbound (import) and outbound (export) sections, where the storage positions are identified by blocks. Vessels carry both of import and export containers where the storage positions are identified by bays. CY and vessels are served by YCs, and QCs, respectively.*

Table 6.1: *The subset of inbound containers handling sequence from a vessel's bays. The sequence are created by the terminal operators and based on this information, the stevedore pick the containers from the vessel and transport them into appropriate CY blocks with ITs.*

Sequence	*Container ID*	*Vessel bay*	*Row*	*Tier*
1	MAL-110	53	1	3
2	MAL-113	53	1	2
3	MAL-109	53	1	1
4	GOT-580	52	3	3
5	GOT-582	52	3	2

hand, a stowage plan is the information on vessel's sub-bays where the outbound containers will be allocated. The inputs to create those two plans for the inbound and outbound containers are the handling sequences, which are illustrated in Tables 6.1 and 6.2, respectively.

The direction of handling sequence in Tables 6.1 and 6.2 are from and to vessels, respectively. In Table 6.1 the terminal operators have to create CY storage plan, while in Table 6.2, the vessel stowage plan need to be devised. The range of container numbers that can be handled each ship is 1,000 to 10,000 TEU per ship [13]. The two examples in Tables 6.1 and 6.2 do not necessarily belong to the same vessel. The alphabetical characters in the container ID in Tables 6.1 and 6.2

Table 6.2: *The subset of outbound containers handling sequence from several blocks in the outbound CY. The sequence are created by the terminal operators and based on this information, the stevedore pick the containers from the CY's blocks and transport them into appropriate vessel's bays with ITs.*

Sequence	*Container ID*	*CY block*	*Row*	*Tier*
1	HUT-904	12	5	3
2	HUT-907	12	5	2
3	HUT-910	12	5	1
4	MAE-881	11	2	3
5	MAE-880	11	2	2

usually refer to the customers/owners of the containers. As have been explained in Sub-Chapter 6.1, the container handling sequences may dynamically change. Therefore, the information given to load or unload containers has to be updated regularly based on latest condition in the field.

For modeling purpose, we assume that the X, Y, and Z coordinate (position) of each inbound container in the CY blocks and each outbound container in the vessel bays are not stipulated. Instead, we determine the CY's sub-block and the vessel's sub-bay to which the containers will be located, and the exact placement of containers in CY's blocks and in vessel's bays are assumed to be properly managed. This limitation is also found in [4, 11]. We believe that the dynamical models of integrated terminal operations which still in the initial phase in this research will be too complicated if this setup is considered. For detail treatment in the modeling of CY's storage plan and ship's stowage plan, we refer interested readers to [24, 94].

6.2.2 Job definition

As explained in the previous sub-chapter, the container handling sequence which is explained in Sub-Chapter 6.2.1 is performed by three main equipments in the terminal, namely QC, YC, and IT. Correspondingly, we define a job as an operation/work that is either 1) to unload each of inbound container from the vessel to the inbound CY; or 2) to load each of outbound container from the outbound CY to the vessel. For the former one, Table 6.1 presents a subset of jobs for the inbound container handling sequence, and similarly, for the latter one, Table 6.2 shows a subset of jobs for the outbound container handling sequence. In both cases, a job is performed by the terminal operators by pairing a QC and a YC, which is connected by an IT, as exemplified in Figure 6.3. In each job, the terminal operators select a QC, if a loading job is about to executed, and in reverse, a YC is selected if an unloading job is being considered. In both kind of jobs, the IT has to be selected from the group of available ITs to perform the operations.

In this chapter, we focus mainly on the modeling and optimization of the former

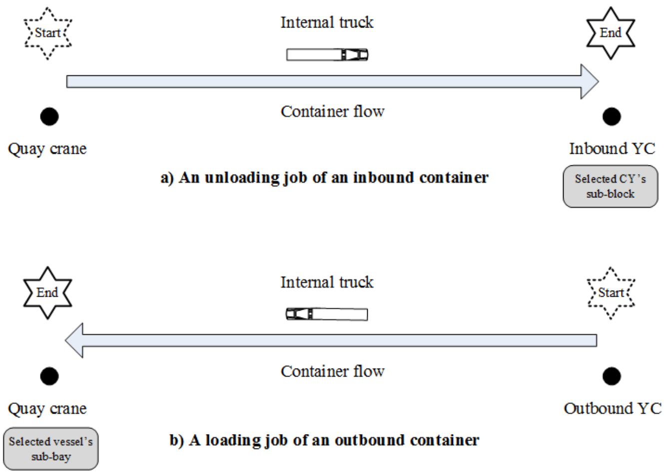

Figure 6.3: *An illustration of two kinds of job available in the integrated container terminal operations. An unloading job refers to the terminal operations of an import/inbound container with the start and end operations at QC (vessel's sub-bay) and YC (CY), respectively. A loading job refers to the handling of an export/outbound container with the start and end operations at YC (CY) and QC (vessel's sub-bay), respectively. In both types of jobs, internal trucks need to be allocated to transport the containers between the two cranes.*

one, e.g., the unloading processes of inbound (import) containers. Hence we incorporate detailed model for the inbound container sequence while the outbound one is simplified by a lumped model.

Figure 6.3 illustrates the container handling sequence in both types of job. As shown in Figure 6.3a, *an unloading job* for the inbound container is initiated by the unloading of an assigned container with the prescribed QC from the vessel to the empty IT chassis, which is mostly located beneath the QC. Subsequently, the loaded IT brings the inbound container to a pre-determined inbound CY's sub-block location before it is picked up by the allocated YC. This particular job is completed when the YC has successfully placed the container into the allocated sub-block in the inbound CY.

The reverse process is applied for the outbound container as shown in Figure 6.3b. *A loading job* starts when a YC at the export CY picks up an already assigned outbound container from the outbound CY and places it on an empty IT's chassis, which is normally situated under the YC. The loaded IT will then transport the outbound container to the allocated QC. When the QC is ready, it takes the container from the IT and brings it to a prescribed sub-bay location in the vessel. Once the outbound container is positioned at the right location in the vessel, the loading job is completed. It is important to note that the YC in the unloading job is different with the YC used in the loading job, while a QC can perform operations both for unloading and loading jobs.

The same job definition is also used in the works of [11, 33, 89], where IT act as connectors between QC and YC for handling the containers. As have been discussed in Sub-Chapter 6.1, the allocation and scheduling are done separately in all these works. Particularly, the dynamics of the integrated container terminal operations problems is only used for the scheduling. In this research, we perform simultaneous allocation and scheduling where the process' dynamics play an important role in both types of decision.

We also assume in this research that the detail movement of the QC, YC, and IT is not included in the modeling. For instances, in the operations of seaside cranes, the hoist and release operations are not considered. Instead, we will later assume that each QC and YC operations requires a constant operational time. The speed of each IT when transporting is also assumed to be constant.

6.3 Dynamical modeling of integrated container terminal operations

In this sub-chapter we present the dynamical models based on the integrated terminal operations framework presented in Sub-Chapter 6.2. As have been introduced in Sub-Chapter 6.1, we follow the modeling framework in [13] using DES for describing the operations of the cranes and trucks and it is combined with a finite-state machine (FSM) for distinguishing between the loading and unloading jobs. The DES model in our present work uses a discrete event time $k \in \mathbb{N}$ which corresponds to the start or initiation of an unloading or a loading job as explained in Sub-Chapter 6.2.2.

As briefly mentioned before, we focus the DES modeling effort on the inbound handling sequence in the present work while for the outbound sequence, we simplify the DES modeling of it by simply assuming a block of area in CY instead of detail sub-blocks of area as in the inbound case.

The involvement of external parties (e.g. the external trucks (ET)) adds to the complexity of the DES modelling. Due to the schedule constraints of the departing vessels, the terminal operators usually apply stricter schedule for ET to deliver outbound containers than for ET to bring the inbound containers to the hinterland [50]. Some traditional terminals, as later shown in our case in Sub-Chapter 6.5, do not have the truck appointment systems for notifying IT that the import containers are ready for clearance from the CY. The random aspect on hinterland container transport by trucks introduces complexity for the inbound operations [50]. Consequently, we focus on a detailed modelling of the inbound CY in this work that represents closely to the integrated operations of container terminals in practice.

In the following sub-chapters, we will firstly described the general setup of the DES and FSM model. It is followed by the DES-FSM model development of the integrated terminal operations. Lastly, we present the predictive model that is used for the development of model predictive control along with the associated cost function.

6.3.1 General DES & FSM setup

Throughout the chapter, we will use various mathematical notations in our modelling and methods that are summarized in Table 6.3. The discrete-event time is denoted by k that corresponds typically to the start of a discrete event in the operations, such as, the start operation of a QC, IT or YC.

Let us define the variables of sets involved in such integrated operations based on the three main operating areas in the terminal, namely, the seaside (berth), the storage (CY), and the transporter (IT). For the seaside operations, $\mathcal{S}(k)$ denotes the set of available sub-bays in ships at an event time k. The set of all sub-bays in all berthed vessels is denoted by $\mathcal{S}_{\text{tot}}(k)$. As before, it follows that $\mathcal{S}(k) \subset \mathcal{S}_{\text{tot}}(k)$ for all discrete-event time k. We emphasize here that both $\mathcal{S}(k)$ and $\mathcal{S}_{\text{tot}}(k)$ accumulate sub-bays information from all berthed vessels at any given discrete-event time k.

As discussed in the Introduction, the seaside and the storage operations are connected by the transporters. While there are different forms of transporter, we restrict ourselves to the use of internal trucks (IT) in this work since it is still the dominant mode of internal transportation in many terminals worldwide, particularly, those in the developing countries [17, 50]. We denote the set of indices of internal trucks by $\mathcal{T}_{\text{tot}} = \{1, 2, \ldots, L\}$.

It has been mentioned before that there are two jobs for the import/inbound and export/ outbound containers and we will associate each job with a state in the FSM. Correspondingly, we denote $\mathcal{J} = \{l, u\}$ as the state space of the FSM where l refers to the state of loading job and u corresponds to the state of unloading job. As described before, we will focus on the detailed modeling for the unload jobs for the inbound containers. The set $J(k) \subset \mathcal{J}$ denotes the state of the FSM at event time k. The guard condition for the FSM will be given later in Sub-Chapter 6.3.2 which is based on the state variables of the DES.

At each event-time k, when a job is assigned (e.g., loading or unloading), all assigned internal trucks proceed to their next position. As described before, we will focus on the detailed modeling for the unload jobs for the inbound containers. Therefore, we assume that the inbound and outbound CY are located in different area of the terminal, as commonly found in practice [16]. We also assume that there is only one block of outbound CY with its own dedicated crane. At any given event time, we have the set $\mathcal{T}_{\text{out}}$ which is a set of trucks that are positioned and readily available at the outbound CY. Hence, in this work, the outbound job

Table 6.3: *List of mathematical notations used in the dynamical models of integrated container terminal operations*

Notation	*Description*
	Decision variables
$u(k)$	Control variable for assigning job from the set $\mathcal{J}(k)$
$t(k)$	Control variable for assigning an inbound truck from the set of IT $\mathcal{T}_{\text{in}}(k)$
$\bar{t}(k)$	Control variable for assigning an outbound truck from the set of outbound IT $\mathcal{T}_{\text{out}}(k)$
$s(k)$	Control variable for assigning a vessel's sub-bay from/to which an inbound/an outbound container is sent/delivered
$n(k)$	Control variable for assigning a CY sub-block from the set $\mathcal{Y}(k)$ for inbound operations
	Parameters
L	The number of inbound internal trucks
M	The number of positions of QCs
N	The number of positions of inbound CYs
v	Average speed of an IT
$\mathrm{d}(a, b)$	Distance between point a and b in the terminal
α	Average time needed by a QC to handle a container
β	Average time needed by a YC to handle a container
	State variables
$x_t^i(k)$	The state variable of the position of the i-th IT
$x_q^i(k)$	The state variable of finishing time of the i-th QC
$x_c^i(k)$	The state variable of finishing time of the i-th QC
	Sets
$\mathcal{S}_{\text{tot}}(k)$	Set of all sub-bays in ships at event time k
$\mathcal{S}(k)$	Set of available sub-bays in ships at event time k
$\mathcal{S}_{\text{arr,tot}}(k)$	Set of new sub-bays from newly berthed vessels
$\mathcal{S}_{\text{arr,empty}}(k)$	Set of available sub-bays from newly berthed vessels
$\mathcal{S}_{\text{dep,tot}}(k)$	Set of new sub-bays from recently departed vessels
$\mathcal{S}_{\text{dep,empty}}(k)$	Set of available sub-bays from recently departed vessels
$\mathcal{T}_{\text{tot}}$	Set of all internal trucks
$\mathcal{T}_{\text{in}}(k)$	Set of all available IT assigned for unloading jobs at event time k
$\mathcal{T}_{\text{out}}(k)$	Set of all available IT assigned for loading jobs at event time k
$\mathcal{J}$	State space of the FSM for jobs: loading job (l) and unloading job (μ)
$J(k)$	The state of the machine/job at event time k
	Indices
k	Event time
K	Planning horizon time
j	Index of the first earliest available QC
l	Index for a loading job
u	Index for an unloading job

depends only on the availability of cranes in QC and of sub-bays in the vessel. When a loading job is executed, we can replenish the truck in $\mathcal{T}_{\text{out}}$ to replace the assigned outbound truck. However, for the set of inbound IT $\mathcal{T}_{\text{in}}$, they are not necessarily available at all cranes of QC. Thus when an unloading job is executed, we need to allocate a truck from $\mathcal{T}_{\text{in}}$ that will move from its current position in the terminal to the assigned QC crane. Note that at any given event time k, the sets of IT satisfies the following relations

$$\mathcal{T}_{\text{tot}} = \mathcal{T}_{\text{in}}(k) \cup \mathcal{T}_{\text{out}}(k) \tag{6.1}$$

with $\mathcal{T}_{\text{in}}(k) \cap \mathcal{T}_{\text{out}}(k) = \varnothing$. Due to this conservative relation, for the rest of this chapter, we will only describe the dynamics of $\mathcal{T}_{\text{out}}(k)$ while the state of $\mathcal{T}_{\text{in}}(k)$ can be deduced directly from (6.1).

For each truck index $i = 1, ..., L$, we denote

$$x_t^i(k) \in \{0, 1, 2, ..., M, M+1, M+N\} \tag{6.2}$$

as the position state of $i-$th truck where 0 refers to the outbound CY position, $1, ..., M$ refer to each of M cranes of QC and the rest represent each of N cranes in the inbound CY. For example, $x_t^i(k) = 2$ corresponds to the state of $i-$th truck at event time k which is located at the 2^{nd} crane in QC while $x_t^i(k) = M+2$ means that the $i-$th truck is located at the 2^{nd} yard crane in the inbound CY at event time k. The position of each truck x_t^t is initialized at $x_{t,0}^i \in \{1, 2, ..., M, M+1, M+N\}$ for all $i \in \mathcal{T}_{\text{in}}(0)$ and $x_{t,0}^i = 0$ otherwise.

Following the modeling framework for berth and quay-crane allocation in [13], the state variables will be given by the finishing time of the two equipment (the cranes in both QC and YC) and the position of all IT at each event-time k. Namely, $x_t^i(k)$ describes the position state of the $i-$th IT at event-time k, $x_q^i(k)$ refers to the finishing time of the $i-$th QC at event-time k and $x_c^i(k)$ denotes the finishing time of the $i-$th yard crane in the inbound CY at event time k.

6.3.2 DES-FSM of integrated container terminal operations

Based on the description of variables and sets in the previous sub-chapter, we can now present the DES-FSM modeling of integrated container terminal operations. Firstly, a new event time k is triggered whenever a QC has finished unloading/loading job from/to an assigned IT from the previous event time $k-1$. Thus the actual time associated to the new event time k is given by

$$j = \arg\min_i [x_q^i(k-1)]. \tag{6.3}$$

Simultaneously, a guard condition is used to determine the transition of state machine at the new event time k. Before defining the guard condition, we denote $\mathrm{d}(a, b)$ as the distance between the two points a and b in the container terminal, and the route from a to b has to follow pre-defined possible paths in the terminal. We also assume that the operations time needed by a QC to handle a 20-feet container at the sub-bay $s \in \mathcal{S}$ (loading or unloading) is described by the function $\mu : \mathcal{S}_{\text{tot}} \to \mathbb{R}_+$. It is dependent on the location of the sub-bays in the vessel, e.g., the Cartesian coordinate of the sub-bays on the vessel. Using these notations and given j as in (6.3), the guard condition and the transition of state machine are as follows.

Guard: If there exist $n \in \{1, ..., N\}$, $s \in \mathcal{S}_{\text{tot}} \backslash \mathcal{S}(k-1)$ and $\ell \in \mathcal{T}_{\text{tot}} \backslash \mathcal{T}_{\text{out}}(k-1)$ such that

$$\begin{aligned} x_c^n(k-1) < x_q^j(k-1) &+ \mu(s) \\ &+ \frac{1}{v}[\mathrm{d}(x_t^\ell(k-1), j\text{-th QC}) \\ &+ \mathrm{d}(j\text{-th QC}, n\text{-th YC})] \end{aligned} \tag{6.4}$$

then

$$J(k) = u, \tag{6.5}$$

or otherwise

$$J(k) = l, \tag{6.6}$$

where v is the constant velocity of an IT.

Roughly speaking, the inequality (6.4) in the guard condition means that there will be an available yard crane in the inbound CY (the n-th CY crane) at the next event time when we allocate the ℓ-th truck (which is not currently located in the outbound CY) to unload a container at the s-th sub-bay in the vessel from their current position $x_t^\ell(k-1)$ to the final destination of the n-th YC. As we have explained in Sub-Chapter 6.2.2, an inbound container can be processed in the inbound CY only when both the YC and the IT are ready. When these conditions hold then the empty IT moves from its current position to the QC, holds its position at QC until it has received the container from the crane, and subsequently the loaded IT proceeds on to the designated YC.

After the new event time and the associated state machine have been updated, we proceed to the decision making process. Based on the guard condition as before, if $J(k) = l$ then three decision variables for the outbound process have to be made, namely, the outbound internal truck $\bar{t}(k)$ taken from $\mathcal{T}_{\text{out}}(k-1)$, the internal truck $t(k)$ taken from and $\mathcal{T}_{\text{tot}} \backslash \mathcal{T}_{\text{out}}(k-1)$ for marshalling trucks in $\mathcal{T}_{\text{out}}(k)$ and the vacant sub-bay in the vessel $s(k)$ taken from $\mathcal{S}(k-1)$.

Otherwise, when $J(k) = u$ then three decisions for the inbound process have to be made. They are the inbound internal truck $t(k)$ taken from $\mathcal{T}_{\text{tot}} \backslash \mathcal{T}_{\text{out}}(k-1)$, the available yard crane $n(k)$ that satisfies (6.4) and the inbound container sub-bay $s(k)$ in the vessel that belongs to $\mathcal{S}_{\text{tot}} \backslash \mathcal{S}(k-1)$. We note that as there can be more than one solution of n and ℓ that satisfy (6.4), a combinatorial optimization on the truck and the crane may be required to optimize the operations. In the next sub-chapter, we will return back to the allocation strategy.

Subsequently, after these decision variables (depending on the particular job) have been taken, the state variables x_t^i, x_q^i, x_c^i and the dynamic sets $\mathcal{S}$, $\mathcal{T}_{\text{in}}$ and $\mathcal{T}_{\text{out}}$ are updated as follows. On the one hand, when the system is in the loading mode with $J(k) = l$, we have the following update rule:

$$\begin{aligned} x_q^j(k) = x_q^j(k-1) + \mu(s(k)) \\ + \frac{1}{v}\text{d}(j\text{-th quay, outbound CY crane}) \end{aligned} \tag{6.7}$$

$$x_t^{\bar{t}(k)}(k) = j, \qquad x_t^{t(k)}(k) = 0 \tag{6.8}$$

where j is as in (6.3), $\mu(s(k))$ denotes the crane operations time for loading the container to the sub-bay $s(k)$ and

$$x_q^i(k) = x_q^i(k-1) \qquad \forall i \neq j \tag{6.9}$$

$$x_c^i(k) = x_c^i(k-1) \qquad \forall i \tag{6.10}$$

$$x_t^i(k) = x_t^i(k-1) \qquad \forall i \neq \bar{t}(k) \text{ or } t(k) \tag{6.11}$$

$$\mathcal{T}_{\text{out}}(k) = \mathcal{T}_{\text{out}}(k-1) \cup t(k) \backslash \bar{t}(k) \tag{6.12}$$

$$\mathcal{S}(k) = \mathcal{S}(k-1) \backslash s(k) \cup \mathcal{S}_{\text{arr,empty}}(k) \backslash \mathcal{S}_{\text{dep,empty}}(k) \tag{6.13}$$

$$\mathcal{S}_{\text{tot}}(k) = \mathcal{S}_{\text{tot}}(k-1) \cup \mathcal{S}_{\text{arr,tot}}(k) \backslash \mathcal{S}_{\text{dep,tot}}(k), \tag{6.14}$$

where $\mathcal{S}_{\text{arr,tot}}(k)$ and $\mathcal{S}_{\text{arr,empty}}(k)$ is the set of new sub-bays and available sub-bays from the newly berthed vessel(s), respectively, and correspondingly, $\mathcal{S}_{\text{dep,tot}}(k)$ and $\mathcal{S}_{\text{dep,empty}}(k)$ are those from the recently departed vessel(s).

On the other hand, when the unloading job occurs (e.g., $J(k) = u$), these variables are updated according to

$$
\begin{aligned}
x_q^j(k) &= x_q^j(k-1) + \mu(s(k)) \\
&\qquad + \frac{1}{v}\mathrm{d}(x_t^{t(k)}(k-1), j\text{-th quay}) && (6.15)\\
x_c^{n(k)}(k) &= x_q^j(k-1) \\
&\quad + \frac{1}{v}\mathrm{d}(x_t^{t(k)}(k-1), j\text{-th quay}) + \mu(s(k)) \\
&\quad + \frac{1}{v}\mathrm{d}(j\text{-th quay}, n(k)\text{-th CY crane}) + \beta && (6.16)\\
x_t^{t(k)}(k) &= n(k) + M && (6.17)
\end{aligned}
$$

where j is as in (6.3), $\mu(s(k))$ gives the crane operations time for unloading the container from the sub-bay $s(k)$ and

$$
\begin{aligned}
x_q^i(k) &= x_q^i(k-1) \qquad \forall i \neq j && (6.18)\\
x_c^i(k) &= x_c^i(k-1) \qquad \forall i \neq n(k) && (6.19)\\
x_t^i(k) &= x_t^i(k-1) \qquad \forall i \neq t(k) && (6.20)\\
\mathcal{T}_{\text{out}}(k) &= \mathcal{T}_{\text{out}}(k-1) && (6.21)\\
\mathcal{S}(k) &= \mathcal{S}(k-1) \cup s(k) \cup \mathcal{S}_{\text{arr,empty}}(k) \backslash \mathcal{S}_{\text{dep,empty}}(k) && (6.22)\\
\mathcal{S}_{\text{tot}}(k) &= \mathcal{S}_{\text{tot}}(k-1) \cup \mathcal{S}_{\text{arr,tot}}(k) \backslash \mathcal{S}_{\text{dep,tot}}(k). && (6.23)
\end{aligned}
$$

In contrast to the quay crane operations, the operations time for the yard crane is approximately constant and is given by β (c.f. (6.16)). It is assumed here that the yard cranes are well-placed in the container yard such that the operations time for unloading any container to the yard is relatively constant.

Roughly speaking, the dynamics of the state variables and the sets of IT and sub-bays in (6.7)-(6.23) can be described qualitatively as follows. During the loading mode, Eq. (6.7) describes the finishing time of loading process at the j-th quay crane that comprises of the standard crane operations time and the travel time of the internal truck $\bar{t}(k)$ from outbound CY to the crane. The latter is described in (6.8) along with the marshalling of an IT truck $t(k)$ to the outbound CY. The rest of the state variables of the quay cranes, yard cranes and internal trucks are the same as the previous event time as presented in (6.9)-(6.11). Due to the use and marshalling of trucks from and to the container yard, respectively, the set of available outbound trucks $\mathcal{T}_{\text{out}}$ is updated accordingly as in (6.21). Lastly, the used sub-bay is removed from the set of available sub-bays in the vessel $\mathcal{S}(k)$ in (6.22).

Similarly, for the unloading mode, Eqns. (6.15)-(6.17) describe the unloading process. In (6.15), the finishing time of j-th quay crane x_q^j is updated according to the unloading operations time $\mu(s(k))$ and the travel time from the current

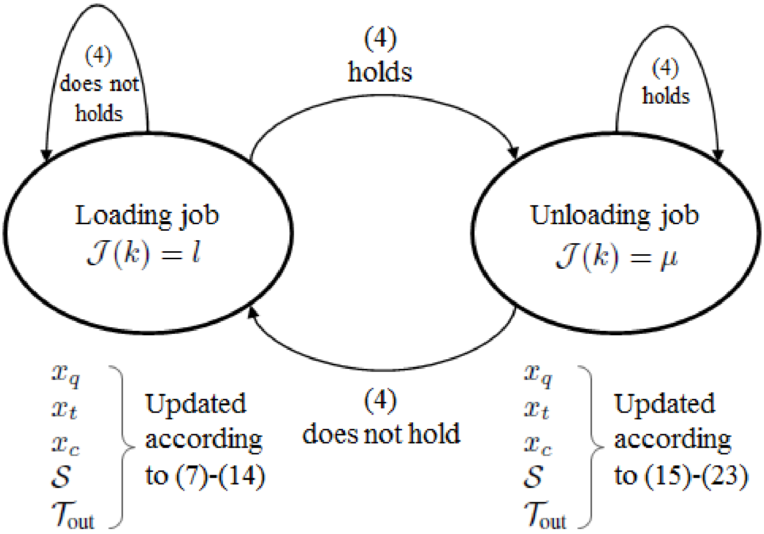

Figure 6.4: *An illustration of the state machine transition for the two types of jobs and the update of state variables after each transition.*

position of $t(k)$-th truck to the crane. The finishing time of the $n(k)$-th yard crane is computed in (6.16) based on the accumulation of total quay crane operations time, the travel time from the j-th quay crane to the yard crane and the yard crane operations time β. The final position of $t(k)$-th truck position will be at the $n(k)$-th yard crane as given in (6.17). The rest of equations (6.18)-(6.22) can be understood similarly to those for the loading job as before.

The transition of this finite state machine is shown in Figure 6.4. The illustration explains the guard condition and the state machine transition as given in (6.4).

Similar to our previous work in [13], the model considers the set dynamic of available trucks and of ship's sub-bays at each event time k. The dynamic aspect of these sets is not yet considered in the state-of-the-art approaches, as found for example in [33, 36]. In these works, it is commonly assumed that the entire information is known and can directly be used in its entirety for the whole planning horizon. Whereas in practice, the availability of vessels' bays and trucks in a terminal is very dynamic which can be due to disruptions, such as, the equipment breakdowns or the non-existence of human operators [5, 24, 48] or due to an incomplete/incorrect container manifest in the vessel. The integration of the above model with the following model predictive allocation strategy will allow us to monitor the operations and to adjust the planning in real-time in dealing with such dynamically changing environment.

6.4 Model predictive allocation method for integrated terminal operations

In this sub-chapter, we will use the integrated terminal model from the previous sub-chapter as a predictive model for optimizing the allocation of trucks, cranes and sub-bays. We will present the adaptation of model predictive allocation strategy as presented in [13] to such integrated terminal operations.

In general, the model predictive allocation strategy can be described as follows. As shown in the previous sub-chapter, at each event time k, we have to make a decision for a number of operational variables according to the admissible jobs at the time and based on which, the state will transition to the next state.

Instead of making decision to these variables based only on the information at each event time k, we can use the model to predict the outcome of the future states within a finite horizon of event time when a given sequence of decisions is being evaluated. Subsequently, the first action from the optimal sequence of decisions can be implemented in the terminal operations for the current event time. This allocation strategy is recursively done for all subsequent events. We note that the update of the states based on the available information at any given time ensures that the model will always be up-to-date with the current terminal operations.

We will firstly describe the objective functions that will be optimized by our proposed model predictive allocation algorithm. We will then describe the algorithm and the preconditioning steps to solve the recursive optimization problem. As there are two sets of decision variables that correspond to two possible jobs, the predictive model will take into account all possible future machine states that depend on the outcome of a particular sequence of decisions within a finite horizon of event time.

6.4.1 Cost function

The cost function used to evaluate policies in our dynamical models is related to the operations time. The use of operations time in the cost function for the optimization has been used extensively in literature, see for instance, [36]. Other types of cost function, such as, the length of queue [11] and the energy expenditure [33, 89], can be considered as well in our setting as the dynamical model described in the previous sub-chapter can include information on the queue of IT and fuel consumption of IT by tracking the distance travelled.

We recall from the previous sub-chapter that $n(k)$ is one of the decision variables on the yard crane unit that will be assigned for processing an inbound container in CY. Let us denote $w(k)$ the earliest available QC at the next event time k for unloading/loading job based on the available information/predicted state at $k-1$:

$$w(k) = \arg\min_i [x_q^i(k-1)] \tag{6.24}$$

As we are interested with the total operations time, the objective function for the optimization of terminal operations can be the total time to unload the entire import containers and to load all the export containers starting from the initial event time $k=0$ up to the event time equal to the total number of inbound and outbound containers. By Bellman's principle of optimality, the optimization of this operations is equivalent to the optimization of the cost-to-go or Bellman value function at every event time k, which will be the total operations time from the event time k until the end of operations. We note that, the latter involves combinatorial optimization that includes future machine states $\mathcal{J}(k)$ whose number of states can grow exponentially with the future event time. By adopting the receding horizon control (known also as the model predictive control (MPC)) approach, instead of using the aforementioned cost-to-go, we can consider a (shorter) finite horizon of event time for the cost function.

Consequently, at every event time k, we consider the following receding horizon-based cost function

$$Z(k) = \sum_{m=k}^{K} z(m) \tag{6.25}$$

where K is the length of horizon, $z(m)$ in (6.25) is the total time spent in the container terminal to allocate a single inbound/outbound container that is defined by

$$z(m) = \begin{cases} x_q^{w(m)}(k) - x_q^{w(m)}(k-1) & \text{if } \mathcal{J}(k) = l \\ x_c^{n(m)}(k) - x_c^{n(m)}(k-1) & \text{if } \mathcal{J}(k) = u, \end{cases} \tag{6.26}$$

where the future state and decision variables follow the model as in (6.7)–(6.22). From (6.26), we have that the total operational time for loading an outbound container is given by the difference of finishing time of QC at k and $k-1$. Similarly, the time for unloading an inbound container is based on the difference of finishing time of the assigned yard crane at k and $k-1$.

6.4.2 Allocation algorithm and pre-conditioning steps

We follow the framework of model predictive allocation (MPA) as provided in [13]. At every event time k, we denote $\hat{x}_t^i(h)$, $\hat{x}_q^i(h)$, $\hat{x}_c^i(h)$ and $\hat{w}(h)$ with integer $h \geqslant 0$ as the predicted state variables and the predicted available quay crane, respectively, at event time $k+h$ computed using a copy of the model. Using these notations,

the MPA algorithm is given as follows.
MPA algorithm (for integrated terminal operations):

1. For a new event time k, identify the available quay crane j according to (6.3) and evaluate the **Guard** condition for determining the current job $\mathcal{J}(k)$.

2. Based on the previous information of the state variables, j and $\mathcal{J}(k)$, solve the following receding horizon optimization problem

$$\min_{\hat{n},\hat{t},\hat{\bar{t}},\hat{s}} Z(k)$$

subject to the following state equations for all $h = 0, 1, \ldots K$ with K be the length of the horizon

$$\left.\begin{aligned}
\hat{x}_q^{\hat{w}(h)} &= \hat{x}_q^{\hat{w}(h-1)} + \mu(\hat{s}(h)) \\
\hat{x}_{t,p}^{\hat{\bar{t}}(h)}(h) &= \hat{w}(h) \\
\hat{x}_{t,p}^{\hat{t}(h)}(h) &= 0 \\
\hat{x}_q^i(h) &= \hat{x}_q^i(h-1), \forall i \neq \hat{w}(h) \\
\hat{x}_c^i(h) &= \hat{x}_c^i(h-1), \forall i \\
\hat{\mathcal{T}}_{\text{out}}(h) &= \hat{\mathcal{T}}_{\text{out}}(h-1) \cup \hat{t}(h) \\
&\quad \backslash \hat{\bar{t}}(h) \\
\hat{\mathcal{S}}(h) &= \hat{\mathcal{S}}(h-1) \backslash \hat{s}(h) \\
&\quad \cup \hat{\mathcal{S}}_{\text{arr,empty}}(h) \backslash \hat{\mathcal{S}}_{\text{dep,empty}}(h) \\
\hat{\mathcal{S}}_{\text{tot}}(h) &= \hat{\mathcal{S}}_{\text{tot}}(h-1) \\
&\quad \cup \hat{\mathcal{S}}_{\text{arr,tot}}(h) \backslash \hat{\mathcal{S}}_{\text{dep,tot}}(h) \\
\hat{n}(h) &= \varnothing
\end{aligned}\right\} \text{if } \hat{\mathcal{J}}(h) = l$$

or else (whenever $\hat{\mathcal{J}}(h) = \mu$)

$$\begin{aligned}
\hat{x}_q^{\hat{w}(h)} &= \hat{x}_q^{\hat{w}(h-1)} + \mu(\hat{s}(h)) \\
\hat{x}_c^{\hat{n}(h)}(h) &= \hat{x}_q^{\hat{w}(h-1)} \\
&\quad + \tfrac{1}{v}\mathrm{d}(\hat{x}_{t,p}^{\hat{t}(h)}, \hat{w}(h)\text{-th quay}) + \mu(\hat{s}(h)) \\
&\quad + \tfrac{1}{v}\mathrm{d}(\hat{w}(h)\text{-th quay}, \hat{n}(h)\text{-th CY crane}) \\
&\quad + \beta \\
\hat{\mathcal{T}}_{\text{out}}(h) &= \hat{\mathcal{T}}_{\text{out}}(h-1) \\
\hat{\mathcal{S}}(h) &= \hat{\mathcal{S}}(h-1) \cup \hat{s}(h) \cup \hat{\mathcal{S}}_{\text{arr,empty}}(h) \\
&\quad \backslash \hat{\mathcal{S}}_{\text{dep,empty}}(h) \\
\hat{\mathcal{S}}_{\text{tot}}(h) &= \hat{\mathcal{S}}_{\text{tot}}(h-1) \\
&\quad \cup \hat{\mathcal{S}}_{\text{arr,tot}}(h) \backslash \hat{\mathcal{S}}_{\text{dep,tot}}(h) \\
\hat{x}_q^i(h) &= \hat{x}_q^i(h-1), \forall i = \hat{w}(h) \\
\hat{x}_c^i(h) &= \hat{x}_c^i(h-1), \forall i \neq \hat{n}(h) \\
\hat{\bar{t}}(h) &= \varnothing
\end{aligned}$$

3. Using the optimal solution in 2), assign the inbound yard crane $n(k) = \hat{n}(0)$, the outbound vessel's bay $s(k) = \hat{s}(0)$, the inbound internal truck $t(k) = \hat{t}(0)$ and the outbound internal truck $\bar{t}(k) = \hat{\bar{t}}(0)$. When $\hat{n}(0)$ or $\hat{\bar{t}}(0)$ is an empty set, it means that there is no assignment of yard crane or outbound internal truck, respectively.

4. Increment the event time k by one and return to 1).

For solving the optimization problem in Step 2 of the above-mentioned MPA algorithm with the event horizon $h = \{0, 1, \ldots K\}$, we need to compute the optimal inbound yard cranes $\hat{n}(0), \ldots, \hat{n}(K) \in \{1, \ldots, N\}$, the vessel's sub-bays $\hat{s}(0), \ldots, \hat{s}(K) \in \mathcal{S}_{\text{tot}}(k)$, the inbound internal trucks $\hat{t}(0), \ldots, \hat{t}(K) \in \mathcal{T}_{\text{tot}} \backslash \mathcal{T}_{\text{out}}(k)$ and the outbound internal trucks $\hat{\bar{t}}(0), \ldots, \hat{\bar{t}}(K) \in \mathcal{T}_{\text{out}}(k)$. Solving this combinatorial optimization for a finite length of horizon will still be non-trivial when the horizon length K is large. In order to facilitate this, we introduce preconditioning, similar to that used in [13]. The preconditioning is based on the ordering of the elements in the discrete sets of the decision variables (according to some measures), and followed by a truncation of the ordered sets. The optimization is then performed based on the truncated sets. In particular, we consider the following optimization steps using the above mentioned preconditioning:

1. Let $\mathcal{S}_{\text{ordered}}^{\text{unload}}(k) \subset \mathcal{S}_{\text{tot}}(k)$ be the ordered set of sub-bays containing the remaining containers to be unloaded at time k, which are ordered based on the container handling sequence predetermined by the terminal operator (as exemplified in Table 6.1 and 6.2). Set $\mathcal{A}(k)$ as the first K elements of $\mathcal{S}_{\text{ordered}}^{\text{unload}}(k)$.

2. Similarly, we define $\mathcal{S}^{\text{load}}_{\text{ordered}}(k) = \{s_1, s_2, \ldots\} \subset \mathcal{S}(k)$ as the ordered set of the available sub-bays at time k, which is ordered based on the quay-crane operations time such that
$$\mu(s_1) \leqslant \mu(s_2) \leqslant \mu(s_3) \leqslant \ldots.$$
Set $\mathcal{B}(k)$ as the first K elements of $\mathcal{S}^{\text{load}}_{\text{ordered}}(k)$.

3. Let $\mathcal{Y}(k) = \{y_1, y_2, \ldots, y_N\}$ be the ordered set of inbound yard cranes at time k such that
$$x_c^{y_1}(k) \leqslant x_c^{y_2}(k) \leqslant \ldots \leqslant x_c^{y_N}(k)$$
holds where N is the number of yard cranes. Accordingly, set the truncated ordered set $\mathcal{C}(k)$ as the first K elements of $\mathcal{Y}(k)$.

4. Define $\mathcal{T}_{\text{ordered}}(k) = \{t_1, t_2, \ldots\}$ as the ordered set of internal trucks at time k based on the distance to the $w(k)$-th quay crane where $w(k)$ is as in (6.24), e.g.,
$$\mathrm{d}\big(x^{t_1}_{t,p}, w(k)\big) \leqslant \mathrm{d}\big(x^{t_2}_{t,p}, w(k)\big) \leqslant \ldots$$
holds. Based on this ordered set, set $\mathcal{D}(k)$ as the first K elements of $\mathcal{T}_{\text{ordered}}(k)$.

5. Using the truncated ordered sets $\mathcal{A}(k), \mathcal{B}(k), \mathcal{C}(k), \mathcal{D}(k)$, compute the optimal sequence of sub-bays $\hat{s} \in \mathcal{A}(k)$ (for unloading) or $\mathcal{B}(k)$ (for loading), optimal sequence of inbound yard cranes $\hat{n} \in \mathcal{C}(k)$ and optimal sequence of inbound trucks $\hat{t} \in \mathcal{D}(k)$ that solve the receding horizon optimization problem in step 2) of the MPA algorithm.

The above optimization with pre-conditioning algorithm replaces then step 2) of the MPA algorithm as given before. The preconditioning step that is described above is similar to the one used in [13]. In particular, the model predictive allocation algorithm in [13] uses also the truncated ordered set of berthed ships prior to finding an optimal sequence of ships that solves the receding horizon optimization problem for allocating berth and quay cranes. Instead of dealing with a truncated ordered set, the proposed algorithm above involves four truncated ordered sets, which makes it harder to solve the problem. Yet this preconditioning step facilitates significantly the search of optimal sequences, in comparison to solving the combinatorial optimization using the whole sets of sub-bays, internal trucks and yard cranes. In the following sub-chapter, we will compare the performance of our proposed algorithm above with the state-of-the-art genetic algorithm and particle swarm optimization.

Table 6.4: *Simulation parameters used to test the dynamical models of integrated container terminal operations. The parameters are empirically obtained from observation in Port of Tanjung Priuk, Jakarta, Indonesia. In each of the equipment, we measured 100 container handling operations and took the average operations time.*

Parameter	Value	Unit
QC 1 operations time	180.03	seconds/container
QC 2 operations time	171.37	seconds/container
QC 3 operations time	154.85	seconds/container
YC export operations time	128.53	seconds/container
YC import 1 operations time	115.78	seconds/container
YC import 2 operations time	114.94	seconds/container
Average IT speed	21.02	km/hour

6.5 Simulation

In this sub-chapter, we present the simulation use to evaluate the effectiveness of the model-based allocation algorithm proposed in Sub-Chapter 6.4, which is developed based on the model presented in Sub-Chapter 6.3. We present two kind of simulations. Firstly, we simulate the dynamical models based on real-data collected from a real container terminal and compare the results from the MPA algorithm with the results obtained from the existing policies in the terminal. Secondly, we use Monte Carlo simulation based on large datasets to test the efficacy of the algorithm.

6.5.1 Simulation set-up

The parameters which are used in the simulation are collected from a field observation in international container terminal of Port of Tanjung Priuk, Jakarta, Indonesia. The terminal consists of two berth positions, three quay cranes, two yard cranes in the import side of container yard for the inbound containers, one yard crane in the export side of container yard for the outbound containers, and ten internal trucks. The parameters are shown in Table 6.4.

The parameters are obtained from one hundred observations of operations time for each equipment. As an example, for obtaining the quay crane operations time, we assume that

$$\mu(s) = \alpha + \frac{1}{v_{\mathrm{QC}}}\mathrm{d}(s,0)$$

where α is the average time for a QC to unload/load a container, v_{QC} is the average travel speed of the crane and $\mathrm{d}(s,0)$ is the distance between the sub-bay s to the crane. For the first QC, we measured the time needed by the QC to unload/load 100 set of containers, and by taking the average, we obtain the parameter $\alpha = 180.03$

seconds/container (c.f. Table 6.4). The corresponding speed v_{QC} is obtained by literature studies from [7, 29] with value of 90 meter/minute. This value is also confirmed by the terminal. The operational time variances among cranes are caused by the difference in specifications or equipment's ages. To obtain the parameters of IT speed, we collected one hundred observations for each of the ten trucks and subsequently, based on the average, we set $v = 21.02$ km/hour.

The parameters will be used to simulate the dynamical models of the integrated container terminal operations. To compare with the state-of-the-art methods in this topic, we select two benchmark methods from [33] and [36]. In [33], linear programming (LP) problems are defined for determining the allocation and scheduling of QC, YC, and IT. The problems are solved through genetic algorithm (GA) and particle swarm optimization (PSO) approach. We present the summary of the GA and PSO below, and for the completeness the readers can refer to [33].

1. Select randomly q initial routes of job to handle containers to a QC where $\theta_j^q = 1$ as in [33]. Calculate the insertion cost as in [33] to obtain one of the decision variables, which is the IT selected to perform the operations for the QC, which is represented by $C_{12}(j, u, j') = aq_{j'u} - aqj'$. Calculate the best insertion task, and select the set of jobs for the QC. Repeat the process for all the QC.

2. Select randomly y initial routes of job to handle containers to a YC where $\psi_j^y = 1$ as in [33]. Calculate the insertion cost as in [33] to obtain one of the decision variables, which is the IT selected to perform the operations for the YC, which is represented by $C_{12}(j, u, j') = aq_{j'u} - aqj'$. Calculate the best insertion task, and select the set of jobs for the YC. Repeat the process for all the YC.

3. Select randomly j initial routes of job to handle containers to an IT. Calculate the insertion cost as in [33] to obtain two of the decision variables, which are the QC and the YC to which the IT will operate, which is represented by $C_{12}(j, u, j') = S_{j'u} - S_{j'}$, and $S_j = (1 - \lambda_y).ay_j + \lambda_y.aq_j$. Calculate the best insertion task, and select the set of jobs for the IT. Repeat the process for all the IT.

4. The sets of jobs selected for the QCs, YCs, and ITs in Step (1), (2), (3) by the GA algorithm will be the initial inputs for the PSO. Select the S best individuals from the GA as a particle. Calculate and update the particle to find the best position according to $x_{pj}(t+1) = x_{pj}(t) + v_{pj}(t+1)$ as in [33]. Evaluate the fitness of the sets of the particles of PSO with the same procedure as GA, by the equation $F = 1/(CP.\sum_{i \in X} f_{1i} + CE.f_2)$, and select the best jobs.

The second benchmark is from [36]. We slightly modify the problem setting in [36], which considers the operations of automated container terminals, while we consider in this work a modeling framework for generic terminals. The solution in [36] is obtained through GA, and the procedure is summarized as follows:

1. Select initial populations of the tasks for the cranes (QC and YC), the vehicle (IT), and the storage (CY position) for the sets of inbound and outbound containers.
2. Consider the precedence of tasks/operations and select a random string of numbers whose dimension is $N = \sum_{k=1}^{K} \sum_{i=1}^{Q_k} T_{ki} = \sum_{m=1}^{M} \sum_{n=1}^{N_m} O_{mn}$ as in [36].
3. Evaluate the chromosomes in Step (2) with fitness criterion from objective functions in [36], perform mutation and crossovers and repeat until no task with better fitness is obtained.

6.5.2 Simulation results and validation

In this sub-chapter we will present the simulation results based on data which has been collected from the international container terminal of Port of Tanjung Priuk, Jakarta. The terminal is the smallest in the seaport and the regular vessels that call to the terminal historically range from 300 to 1,000 TEU. To comply with the settings in the dynamical models that we have developed in Sub-Chapter 6.3.1, export and import containers refer to the outbound and inbound containers, respectively.

The terminal operators currently employ density-based quay crane allocation (DBQA) method to allocate QC to the vessels. With this method, the QCs are allocated proportionally with the container density along the quay/berth, or in other words, the number of container per meter berth. The detail explanation of DBQA can be found in [13]. For allocating the YC, the terminal operators use first-come-first-served (FCFS) policy where a job to handle a container is assigned to the earliest available YC. The existing allocation method for IT is also based on FCFS, where a container to be handled is assigned to the earliest and nearest IT. The latter criterion is observed since the ITs always move between the QCs and the YCs.

We use a dataset which is collected from a week observations at the terminal. During that period, four vessels arrived, where the specifications are provided in Table 6.5. The entire containers in Table 6.5 are simulated with 1) terminal's existing policy, 2) MPA algorithm as explained in Sub-Chapter 6.4, 3) two benchmark methods as explained in Sub-Chapter 6.5.1. The simulation results are presented in Table 6.6.

Table 6.5: *Dataset collected from observation at the Port of Tanjung Priuk, Jakarta, Indonesia. The observations are conducted for a week period, where four vessels arrived with each its loads of inbound and export containers.*

Vessel	Total load (TEU)	Import load (TEU)	Export load (TEU)
A	473 TEU	283 TEU	190 TEU
B	312 TEU	209 TEU	103 TEU
C	527 TEU	358 TEU	169 TEU
D	323 TEU	171 TEU	152 TEU

Table 6.6: *Simulation results based on dynamical models of integrated container terminal operations in* (6.3)-(6.19) *with parameters and dataset in Table 6.4 and 6.5. The MPA is performed until $K = 8$, and the objective functions of total operations time is compared with the existing method in the terminal, and two benchmarking methods from [33] and [36].*

K	Allocation strategy	Total operations time (minutes)
-	FCFS & DBQA (existing)	2,572.16
1	MPA	2,713.64
2	MPA	2,667.82
3	MPA	2,614.92
4	MPA	2,605.51
5	MPA	2,541.84
6	MPA	2,535.33
7	MPA	2,469.03
8	MPA	2,405.78
-	GA as in [36]	2,502.06
-	GA & PSO as in [33]	2,435.15

We can see in Table 6.6 that the MPA algorithm with $K \geqslant 5$ improves the performance of the existing method by 1.69%. With $K = 8$, MPA's result is 6.48% better than the existing methods used by the terminal operators. The two benchmarking methods from [36] and [33] also have better performances than the FCFS & DBQA methods. With $K = 8$, our MPA method is 3.21% and 1.03% better than the GA in [36] and the GA and PSO in [33], respectively.

To validate the dynamical models, we compare the one of the state variables, which is the finishing time of the first QC ($x_q^1(k)$). The state variables in each time k are obtained from the observation and the outputs from the simulation as provided in Figure 6.5.2.

From the one week observation, we recorded the realization of QC, YC, and IT allocation in the terminal, where the total operations time needed to handle 1,635 TEU in Table 6.5 was 2,624.84 minutes. From the same dataset, we then find the

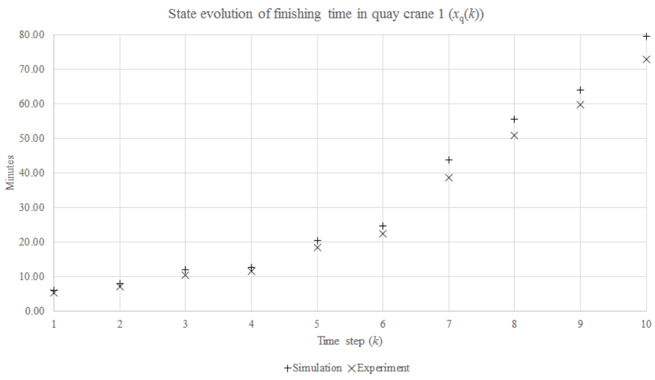

Figure 6.5: *The plot of trajectory of the state variable $x_1^1(k)$ which describes the finishing time of the first quay crane. The horizontal axis is the discrete time step (k). The vertical axis is the time in minutes. In each k, two $x_c^1(k)$s are plotted, where the crosses ($\times$) refer to the actual data recorded from the observation in the terminal, and the plusses ($+$) show the evolution of corresponding state variable from the simulation using* (6.3)-(6.19) *with the same dataset as the former observation.*

optimal control inputs according to dynamical models in (6.3)-(6.19) and the total operations time for the existing FCFS & DBQA methods is 2,576.28 minutes as presented in Table 6.6. The evolution of state variables from both of the observation and simulation of the existing allocation methods were recorded, and the evolution of the state variable of the finishing time of the first internal truck ($x_t^1(k)$) for the first ten discrete time steps (k) is presented in Figure 6.5.2.

It can be seen that the dynamical models are able to mimic the dynamic in the container terminal operations. There are indeed discrepancies between those two state variables. This mainly caused by variations in container handling by QCs and YCs. From one operations to another, the time needed by a QC or a YC to handle a container varies slightly, where we use constant parameters as in Table 6.4. The variations are rooted from the detail operations of the cranes which are not modeled yet in our dynamical models of integrated container terminal operations.

An example of container handling sequence by the three cranes using the MPA algorithm is presented in Table 6.7. It can be seen from the the subset of of the results that the job sequence do not necessarily follow the FCFS rule as now being applied by the terminal operators in the observed seaport.

6.5.3 Simulation results using generated data

For evaluating further the performance of dynamical models in (6.3)-(6.19) and MPA algorithm that has been developed, in this sub-chapter we present the simulation results using realistically generated terminal data inputs. We generate

Table 6.7: *The subset of yard cranes allocation results using MPA algorithm, where the sequence of container handling is shown in each working cranes. The number shows the index of each container.*

Crane	Container handling sequence
YC 1 import	01 - 02 - 03 - 04 - 12 13 - 14 - 15 - 16 - 20
YC 2 import	05 - 06 - 07 - 08 - 18 19 - 25 - 26 - 30 - 31
YC export	284 - 285 - 286 - 287 - 288 303 - 304 - 305 - 310 - 311

Table 6.8: *The setting of simulation scenario using realistically generated datasets. This table presents vessels' loads configuration for each scenario.*

Scenario	Lower bound ship load (TEU)	Upper bound ship load (TEU)
Light load	300	800
Normal load	800	1,500
Heavy load	1,500	3,000

three scenarios with a total of 150 datasets of container operations as presented in Table 6.8 and 6.9. The scenario is reflected from the common terminal operations configuration. For instance, the terminal observed in this chapter can be classified into a terminal with light loads.

In each scenario, 50 datasets are generated, with 100 vessels' loads in each dataset. The examples of the subset of a dataset for each scenario is presented in Table 6.10, 6.11, 6.12. The total loads in every vessel are randomized with uniformly distributed numbers whose lower and upper bounds parameters are presented in Table 6.8. The lower and upper bound parameters of import and export loads percentage are determined from observations and discussion with the terminal operators. For the import load percentage, the lower and upper bound are 40% and 70%, respectively, and the parameters for the export load percentage are 25% and 40%, respectively. In each load, the percentages for the import and export loads are randomized based on the bounds and weighted so the summation

Table 6.9: *The setting of simulation scenario using realistically generated datasets. This table presents number of equipment configuration in the terminal for each scenario.*

Scenario	Number of QC	Number of YC	Number of IT
Light load	3	3	10
Normal load	6	6	20
Heavy load	10	10	30

Table 6.10: *A subset of dataset of light load scenario, where the loads of the first 10 vessels are presented.*

Vessel	Total load (TEU)	Import load (TEU)	Export load (TEU)
1	342	209	133
2	677	417	260
3	306	178	128
4	537	275	262
5	650	389	261
6	507	358	149
7	526	344	182
8	334	197	137
9	658	406	252
10	708	456	252

Table 6.11: *A subset of dataset of normal load scenario, where the loads of the first 10 vessels are presented.*

Vessel	Total load (TEU)	Import load (TEU)	Export load (TEU)
1	1,228	734	494
2	1,013	741	272
3	1,313	854	459
4	881	581	300
5	1,211	829	382
6	1,287	659	628
7	1,140	755	385
8	1,072	580	492
9	1,297	865	432
10	927	643	284

of both of the loads percentages are 100%.

We use constant parameters for the QC and YC operations time, with the time to handle a container for both of the two types of cranes are 180 and 170 seconds, respectively. This parameters are obtained from the standard (manufacturing) specifications of the cranes. The summary of the Monte Carlo simulation results with the large datasets are presented in Table 6.13. The average of total operations time in each scenario shows that MPA always outperform the existing FCFS and DBQA methods, as well as the two benchmarking methods from [33] and [36], although it can be seen that the difference between MPA and GA & PSO method is slight.

The graphical representations of the simulation results are provided in Figure 6.6 and 6.7. With $K = 8$, the average cost reduction from the existing FCFS & DBQA methods of MPA are greater than GA and GA & PSO. The MPA indeed has

Table 6.12: *A subset of dataset of heavy load scenario, where the loads of the first 10 vessels are presented.*

Vessel	Total load (TEU)	Import load (TEU)	Export load (TEU)
1	2,418	1,500	918
2	2,299	1,505	794
3	1,922	1,343	579
4	2,130	1,273	857
5	1,601	1,061	540
6	2,326	1,456	870
7	1,909	1,285	684
8	1,548	951	597
9	1,766	957	809
10	2,111	1,438	673

Table 6.13: *Simulation result of dynamical models in* (6.3)-(6.19) *using the generated datasets with our proposed MPA methods which are compared with the existing method of FCFS & DBQA and two state-of-the-art methods.*

Allocation Strategy	*Average total opr. time (min.)*	*Ave. calc. time per step (s)*
Sc. 1: Light load		
FCFS & DBQA	841.85 $\pm$ 6.18	0.111
GA as in [36]	824.95 $\pm$4.18	51.876
GA & PSO as in [33]	813.84 $\pm$4.14	53.784
MPA (with $K = 8$)	799.36 $\pm$4.66	128.875
Sc. 2: Normal load		
FCFS & DBQA	1,754.94 $\pm$9.45	0.120
GA as in [36]	1,713.59 $\pm$8.62	52.095
GA & PSO as in [33]	1,699.22 $\pm$8.99	53.284
MPA (with $K = 8$)	1,658.61 $\pm$10.08	135.899
Sc. 3: Heavy load		
FCFS & DBQA	3,570.89 $\pm$15.75	0.123
GA as in [36]	3,508.87 $\pm$14.58	51.996
GA & PSO as in [33]	3,415.64 $\pm$14.88	53.853
MPA (with $K = 8$)	3,395.64 $\pm$15.32	137.088

obvious setback, where the calculation time is much greater than the other three methods. This due the problems complexity, in which five control variables (job, YC, vessel's bay, and IT) have to be solved simultaneously. In comparison to the total operations time of a single container, which takes more than fourteen minutes (see Table XIII), the computational time of our proposed algorithm (which is slightly more than two minutes) is still acceptable.

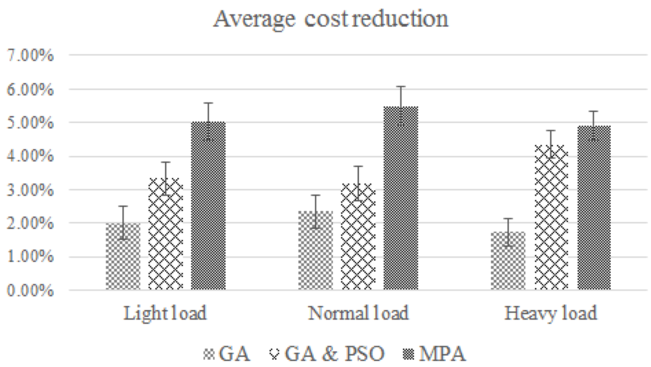

Figure 6.6: *Average cost reduction of GA, GA & PSO and MPA methods when compared to the existing FCFS & DBQA method. The vertical axes in each bar are the error bars.*

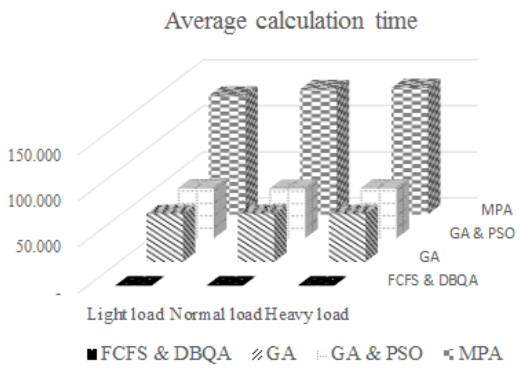

Figure 6.7: *Average calculation time per step (in minutes) for each method in each scenario.*

6.6 Discussion

We have formulated dynamical models of integrated container terminal operations based on DES modeling framework. The operations is an end-to-end processes that include the seaside, storage, and transfer sub-systems, which are usually analyzed independently in the state-of-the-art literature. The difficulty in the optimization caused by the asynchronous operations among quay cranes, yard cranes, and internal trucks is overcome in this research.

The proposed model predictive allocation method allows us to plan the terminal operations integratively and simultaneously: the allocation and scheduling of QC, YC, and IT, as well as, the placement of the boxes in the CY and ship, based on ship's and CY's unloading plan for the inbound and outbound containers, respectively.

We have also conducted data collection from a real container terminal. The simulation shows that given the same inputs, the state variables obtained from the dynamical model, can closely follow the actual state variables collected from the realization of equipment allocation in the seaport by the terminal planner. This implies that the modeling framework can be used to describe any general integrated container terminal operations. Moreover, we solved the optimization problem using our proposed model predictive allocation algorithm with preconditioning. We have shown that the proposed approach performed better than: 1) the existing FCFS & DBQA methods used in the studied terminal, 2) the GA-based method from literature, 3) the GA & PSO-based method from the literature, in which the former two methods use commonly static modeling approach using operations research. Based on the Monte Carlo simulation with large datasets, the MPA outperforms those three methods, although the high computational time of the MPA needs to be taken into account in the trade-off with the cost reduction of the operational time.

Chapter 7

Conclusion and outlook

Chapter 7

Conclusion and outlook

We have presented in this thesis a body of works that are relevant with the optimization and design of control for container terminal operations systems. The topics include various operations in a terminal, which comprise of seaside, integrated, and network operations among several connected terminal. Moreover, we discuss mathematical analysis on the discrete-event systems framework and optimization approach that is pertinent to the terminal operations. In this chapter, we provide a reflection on the results that have been presented in Chapter 3-6 and provide some suggestions for the future research avenue in this topics.

7.1 Conclusion

This thesis deals with the complexity of container terminal operations systems which have mostly been studied in literature using methods from operations research (OR) discipline. In particular, most of the recent works are limited by the use of static optimization that does not handle well the dynamic environment and obstructions that are commonly encountered by operators in actual container terminals. Accordingly, this research focuses on development of discrete-event systems modeling framework and model predictive allocation strategy that are suitable for the operations of general container terminal systems.

In Chapter 3, we study the problem of integrated berth and quay crane allocation (I-BCAP) in general seaport container terminals and propose model predictive allocation (MPA) algorithm and preconditioning methods for solving I-BCAP. Firstly, we propose a dynamical modeling framework based on discrete-event systems (DES) that describes the operation of berthing process with multiple discrete berthing positions and multiple quay cranes. Secondly, based on the discrete-event model, we propose a MPA algorithm for solving I-BCAP using model predictive control (MPC) principle with a rolling event horizon. The validation and performance evaluation of the proposed modeling framework and allocation method are done using: (i). extensive Monte-Carlo simulations with realistically-generated datasets; (ii). real dataset from a container terminal in Tanjung Priuk port, located in Jakarta, Indonesia; and (iii). real life field experiment at the aforementioned container terminal. The numerical simulation results show that our proposed MPA algorithm can improve the efficiency of the process with the reduction of the total handling and waiting cost in comparison to the commonly adapted method of first-come first-served (FCFS) (for the berthing process) combined with the density-based quay cranes allocation (DBQA) strategy. Moreover, the proposed method outperforms the state-of-the-art HPSO-based and GA-based method proposed in recent literature. The real life field experiment also shows an improvement.

The dynamical models developed in Chapter 3 serves as the modeling framework for the remaining of the research. The I-BCAP models also show that in complex container terminal operations sytems, instead of well-known static modeling with operations research, we can also use another approach, with dynamical discrete-event systems (DES) modeling. In particular, the DES are able to incorporate the asynchronous processes in the terminal operations.

In Chapter 4, we provide the mathematical analysis for the dynamical models and MPA. A study an optimal input allocation problem for a class of discrete-event systems with dynamic input sequence (DESDIS) is provided. In this case, the input space is defined by a finite sequence whose members will be removed from the sequence in the next event if they are used for the current event control input. Correspondingly, the sequence can be replenished with new members at every discrete-event time. The allocation problem for such systems describes many scheduling and allocation problems in logistics and manufacturing systems and leads to a combinatorial optimization problem. We show that for a linear DESDIS given by a Markov chain and for a particular cost function given by the sum of its state trajectories, the allocation problem is solved by re-ordering the input sequence at any given event time based on the potential contribution of the members in the current sequence to the present state of the system. In particular, the control input can be obtained by the minimization/maximization of the present input sequence only.

In Chapter 5, we analyze the deployment of a distributed dynamic optimization algorithm to a sub-network of the container terminals. With the increasing globalization of trade and consolidation of container terminal networks worldwide, a competitive network has become an important determinant in attracting shipping liners to call at a number of terminals in the network. For realizing such competitive network with minimal effort, in particular, we focus on the dynamic optimization of berth and quay crane allocation (BCAP) operations, since it has been extensively discussed in the literature that BCAP is usually the bottleneck in the terminal. Therefore, an improvement in one section of terminal operations will significantly improve the overall terminal's performance. Because seaport network is heavily inter-related among each other, an improvement in one section of terminal operations (BCAP) also improves the network operational performance. We use our recently proposed model predictive allocation (MPA) strategy to solve BCAP which has been shown to outperform state-of-the-art methods for a single terminal operations. Afterwards, we present a ranking method to determine significant terminal nodes which can improve the overall network performance significantly when the MPA-based BCAP strategy is implemented to this sub-network. Finally, using both simulations and real data from the Indonesian terminal network, we show the efficacy of our method.

In Chapter 6, we present a dynamical modeling of integrated (end-to-end) container terminal operations using discrete-event system (DES) framework that incorporates the operations of quay cranes (QC), internal trucks (IT), and yard cranes (YC) and also the selection of storage positions in container yard (CY) and vessel bays. The QC and YC are connected by the IT in our models. As opposed to the commonly adapted modeling in container terminal operations, in which the entire information/inputs to the system are known for a defined planning horizon, in this research we use real-time trucks, crane, and container storage operations information, which are always updated as the time evolves. The dynamical model shows that the predicted state variables closely follow the actual field data from a container terminal in Tanjung Priuk, Jakarta, Indonesia. Subsequently, using the integrated container terminal DES model, we proposed a model predictive algorithm (MPA) to obtain the near-optimal solution of the integrated terminal operations problem, namely the simultaneous allocation and scheduling of QC, IT, and YC, as well as selecting the storage location for the inbound and outbound containers in the CY and vessel. The numerical experiment based on the extensive Monte Carlo simulation and real dataset show that the MPA outperforms both of the policies currently implemented by the terminal operator and the state-of-the-art method from the current literature.

7.2 Outlook

In this thesis we have presented some finding in modeling, design of control, and optimization of container terminal operations. We have imposed some assumptions in the research, and in this chapter we discuss some improvement that can be accommodate in the future research to alleviate the drawbacks.

In Chapter 3 and 6, the dynamical models do not yet consider the detail movement of the terminal operations. For instance, in handling containers, the hoisting or lowering cranes operations are not yet modeled. The movement of QC, YC, and IT, which is not incorporated yet in the current models, is an interesting research avenue for the future works. The consideration of the movement will make the models more realistic. There are plenty research on the detail operations, and they also commonly use static approach. We would like to further investigate the application of the dynamical modeling framework in this research into those kind of problem settings.

We also would like to observe the obstructions that occur in the integrated container terminal operations in the simulation. The obstructions mainly come from the changing in CY's storage levels from time to time, due to some random containers pick-up and delivery from ET, and will subsequently change the original CY's storage plan and ship's stowage plan. This setting attests the propositions of the models presented in this work, which should be able to accommodate such dynamic and changing inputs. A guarantee in the model to incorporate double-cycle operations in QC and YC operations will also be relevant future work.

In Chapter 3 and 5, the MPA algorithm makes a ship that arrives earlier at the terminal do not necessarily receives priority in berthing over other ships that arrive later. Although the MPA results in lower total cost from the terminal operators perspective, it has been observed during the field experiment period that some of the arriving ships had shown their dissatisfaction due to a possible uncommon schedule, i.e. a ship may be berthed after another ship which actually arrives later after the former vessel. Therefore, taking into account the shipping liners' dissatisfaction in the cost function can be an interesting and relevant future work.

Applying the optimal BCAP policy in Chapter 5 is not an easy decision to a terminal operator, even if it owns the entire terminals in the same network. The network cost, which is a combination of berth operations cost, ships' waiting costs in terminal, and ships' transportation costs among terminals, is defined from the perspective of the terminal operator, and not yet an integrated cost borne by the shipping liners as well. Therefore, an improved network performance is not always a prominent interest of the shipping liners, even though it obviously also benefits them, since lower waiting time in a terminal means reducing delay in the subsequent terminals. Such terminal operators' difficulty is rooted from a fact that the MPA-based BCAP policy does not allocate berth positions to the incoming ships in a time orderly manner as the traditional BCAP & DBQA methods. In spite of the fact that the optimal policy reduces average ships waiting times, some ships can have dissatisfaction if it is assigned to the berth position later than their arrival time. This can be another research avenue in this topic.

We did mathematical analysis to show efficacy of the MPA algorithm in Chapter 4. In the analysis, the cost function is limited into positive definite one. Further works are needed on the re-ordering of input sequence $\mathcal{U}_k$ (and the subsequent expansion sequences V_k for a finite event horizon) when a general cost function is considered.

Bibliography

[1] Top 50 world container ports. http://www.worldshipping.org. accesed on 10 October 2018.

[2] A. Alessandri, C. Cervellera, M. Cuneo, M. Gaggero, and G. Soncin. Modeling and feedback control for resource allocation and performance analysis in container terminals. *IEEE Transactions on Intelligent Transportation Systems*, 9 (4):601–614, Dec 2008. ISSN 1524-9050. doi: 10.1109/TITS.2008.2006737.

[3] A. Alessandri, M. Gaggero, and F. Tonelli. Min-max and predictive control for the management of distribution in supply chains. *IEEE Transactions on Control Systems Technology*, 19(5):1075–1089, Dec 2011. ISSN 1063-6536. doi: 10.1109/TCST.2010.2076283.

[4] A. Alessandri, C. Cervellera, and M. Gaggero. Predictive control of container flows in maritime intermodal terminals. *IEEE Transactions on Control Systems Technology*, 21(4):601–614, Jul 2013. ISSN 1063-6536. doi: 10.1109/TCST.2012.2200680.

[5] D. Ambrosino, A. Sciomachen, and E. Tanfani. Stowing a containership: the master bay plan problem. *Transportation Research Part A: Policy and Practice*, 38(2):81–99, Feb 2004. ISSN 0965-8564. doi: 10.1016/j.tra.2003.09.002.

[6] N. Asgari, R. Z Farahani, and M. Goh. Network design approach for hub ports-shipping companies competition and cooperation. *Transportation Research Part A: Policy and Practice*, 48:1–18, Feb 2013. ISSN 0965-8564. doi: 10.1016/j.tra.2012.10.020.

[7] A. Bartošek and O. Marek. Quay cranes in container terminals. *Transaction on Transport Science*, 6(1):9–18, Jan 2013. ISSN 1802–9876. doi: 10.2478/v10158-012-0027-y.

[8] C. Bierwirth and F. Meisel. A survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research*, 202(3):615–627, May 2010. doi: 10.1016/j.ejor.2009.05.031.

[9] J. Blazewicz, T. C. E. Chang, M. Machowiak, and C. Oguz. Berth and quay crane allocation: a moldable task scheduling model. *Journal of the Operational Research Society*, 62(7):1189–1197, Jul 2011. ISSN 0160-5682. doi: 10.1057/jors.2010.54.

[10] V.D. Blondel, J.M. Hendrick, A. Olshevsky, and J.N. Tsitsiklis. Convergence in multiagent coordination, consensus, and flocking. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 0191–2216, Dec 2005. ISBN 0-7803-9567-0. doi: 10.1109/CDC.2005.1582620.

[11] C. Caballini, C. Pasquale, S. Sacone, and S. Siri. An event-triggered receding-horizon scheme for planning rail operations in maritime terminals. *IEEE Transactions on Intelligent Transportation Systems*, 15(1):365–375, Feb 2014. ISSN 1524-9050. doi: 10.1109/TITS.2013.2280493.

[12] R. T. Cahyono, E. J. Flonk, and B. Jayawardhana. Dynamic berth and quay crane allocation for multiple berth positions and quay cranes. In *Proceedings of the 14th IEEE Conference on Decision and Control*, Jul 2015. ISBN 978-3-9524-2693-7. doi: 10.1109/ECC.2015.7331037.

[13] R. T. Cahyono, E. J. Flonk, and B. Jayawardhana. Discrete-event systems modeling and model predictive allocation algorithm for integrated berth and quay crane allocation. *IEEE Transactions on Intelligent Transportation Systems*, 21(3):1321–1331, Mar 2020. ISSN 1524-9050. doi: 10.1109/TITS.2019.2910283.

[14] R.T. Cahyono and B. Jayawardhana. On the optimal input allocation of discrete-event systems with dynamic input sequence. In *Proceedings of the 58th IEEE Conference on Decision and Control*, pages 1062–1067, Dec 2019. doi: 10.1109/CDC40024.2019.9029828.

[15] Eduardo F. Camacho and Carlos Bordons. *Model Predictive Control*. Springer-Verlag London Limited, 1999. ISBN 3-540-76241-8.

[16] H. J. Carlo, I. F. A. Vis, and K. J. Roodbergen. Storage yard operations in container terminals: Literature overview, trends, and research directions. *European Journal of Operational Research*, 235(2):412–430, Jun 2014. doi: 10.1016/j.ejor.2013.10.054.

[17] H. J. Carlo, I. F. A. Vis, and K. J. Roodbergen. Transport operations in container terminals: Literature overview, trends, research directions and

classification scheme. *European Journal of Operational Research*, 236(1):1–13, Jul 2014. doi: 10.1016/j.ejor.2013.11.023.

[18] H. J. Carlo, I. F. A. Vis, and K. J. Roodbergen. Seaside operations in container terminals: literature overview, trends, and research directions. *Flexible Services and Manufacturing Journal*, 27(2-3):224–263, Jun 2015. ISSN 1936-6582. doi: 10.1007/s10696-013-9178-3.

[19] Christos Cassandras and Stéphane Lafortune. *Introduction to Discrete Event Systems*. Springer Publishing Company, Incorporated, 2008. ISBN 3642xxx, 9780387686127.

[20] D. Chang, Z. Jiang, W. Yan, and J. He. Integrating berth allocation and quay crane assignments. *Transportation Research Part E: Logistics and Transportation Review*, 46(6):975–990, Nov 2010. ISSN 1366-5545. doi: 10.1016/j.tre.2010.05.008.

[21] G. Cohen, D. Dubois, J. Quardat, and M. Viot. A linear-system-theoretic view of discrete-event processes and its use for performance evaluation in manufacturing. *IEEE Transactions on Automatic Control*, 30(3):210–220, Mar 1985. ISSN 0018-9286. doi: 10.1109/TAC.1985.1103925.

[22] Hans G. Daellenbach and Donald C. McNickle. *Management Science Decision Making Through Systems Thinking*. Palgrave Macmillan, 2005. ISBN 1-4039-4174-2.

[23] X. David-Henriet, L. Hardouin, J. Raisch, and B. Cottenceau. Model predictive control for discrete event systems with partial synchronization. *Automatica*, 70:9–13, Apr 2016. doi: 10.1016/j.automatica.2015.12.006.

[24] R. Delgado, R. M. Jensen, and K. Janstrup. A constraint programming model for fast optimal stowage of container vessel bays. *European Journal of Operational Research*, 220(1):251–261, Jul 2012. ISSN 0377-2217. doi: 10.1016/j.ejor.2012.01.028.

[25] M. Dorfman and J. V. Medanic. Scheduling trains on a railway network using a discrete event model of railway traffic. *Transportation Research Part B: Methodological*, 38(1):81–98, Jan 2004. doi: 10.1016/S0191-2615(03)00006-7.

[26] I. Giannoccaro and P. Pontrandolfo. Inventory management in supply chains: a reinforcement learning approach. *International Journal of Production Economics*, 78(2):153–161, Jul 2002. ISSN 0925-5273. doi: 10.1016/S0925-5273(00)00156-0.

[27] G. Goodwin, P. Ramadge, and P. Caines. Discrete-time multivariable adaptive control. *IEEE Transactions on Automatic Control*, 25(3):449–456, Jun 1980. ISSN 0018-9286. doi: 10.1109/TAC.1980.1102363.

[28] Y. Guan, W. Xiao, R.K. Cheung, , and C. Li. A multiprocessor task scheduling model for berth allocation: heuristic and worst-case analysis. *Operations Research Letters*, 30(5):343–350, Oct 2010. ISSN 0167-6377. doi: 10.1016/S0167-6377(02)00147-5.

[29] E. U. Guldogan. Simulation-based analysis for hierarchical storage assignment policies in a container terminal. *Simulation: Transactions The Society for Modeling and Simulation International*, 87(6):523–537, May 2010. ISSN 0037–5497. doi: 10.1177/0037549710369812.

[30] P. Hansen, C. Oğuz, and N. Mladenović. Variable neighborhood search for minimum cost berth allocation. *European Journal of Operational Research*, 191 (3):636–649, Dec 2008. ISSN 0377-2217. doi: 10.1016/j.ejor.2006.12.057.

[31] H. E. Haralambides, P. Cariou, and M. Benacchio. Costs, benefits and pricing of dedicated container terminals. *International Journal of Maritime Economics*, 4(1):21–34, Apr 2002. ISSN 1388-1973. doi: 10.1057/palgrave.ijme.9100031.

[32] A. Hassibi, S. P. Boyd, and J. P. How. Control of asynchronous dynamical systems with rate constraints on events. In *Proceedings of the 38th IEEE Conference on Decision and Control*, Dec 1999. doi: 10.1109/CDC.1999.830133.

[33] J. He, Y. Huang, W. Yan, and S. Wang. Integrated internal truck, yard crane and quay crane scheduling in a container terminal considering energy consumption. *Expert Systems with Applications*, 42(5):2464–2487, Apr 2015. ISSN 0957-4174. doi: 10.1016/j.eswa.2014.11.016.

[34] J.M. Hendrickx, G. Shi, and K.H. Johansson. Finite-time consensus using stochastic matrices with positive diagonals. *IEEE Transactions on Automatic Control*, 60(4):1070–1073, Apr 2015. ISSN 0018-9286. doi: 10.1109/TAC.2014.2352691.

[35] F.S. Hillier, G.J. Lieberman, and B. Nag. *Introduction to Operations Research*. Prentice Hall, 2010. ISBN 978-0-07-133346-7.

[36] S. M. Homayouni, S. H. Tang, and O. Motlagh. A genetic algorithm for optimization of integrated scheduling of cranes, vehicles, and storage platforms at automated container terminals. *Journal of Computational and*

Applied Mathematics, 270:545–556, Nov 2014. ISSN 0377-0427. doi: 10.1016/j.cam.2013.11.021.

[37] H-P. Hsu. A hpso for solving dynamic and discrete berth allocation problem and dynamic quay crane assignment problem simultaneously. *Swarm and Evolutionary Computation*, 27:156–168, Apr 2016. ISSN 2210-6502. doi: 10.1016/j.swevo.2015.11.002.

[38] L. Hvattum, I. Norstad., K. Fagerholt., and G. Laporte. Analysis of an exact algorithm for the vessel speed optimization problem. *Networks*, 62(2):132–135, Sep 2013. doi: 10.1002/net.21503.

[39] A. Imai, E. Nishimura, and S. Papadimitriou. The dynamic berth allocation problem for a container port. *Transportation Research Part B: Methodological*, 5(4):401–417, May 2001. ISSN 0191-2615. doi: 10.1016/S0191-2615(99) 00057-0.

[40] A. Imai, H.C. Chien, E. Nishimura, and S. Papadimitriou. The simultaneous berth and quay crane allocation problem. *Transportation Research Part E: Logistics and Transportation Review*, 44(5):900–920, Sep 2008. ISSN 1366-5545. doi: 10.1016/j.tre.2007.03.003.

[41] F.R. Jacobs and R.B. Chase. *Operations and Supply Chain Management, fourteenth edition*. McGraw Hill Education, 2014. ISBN 978-1-259-12322-1.

[42] R.R. Negenborn J.L. Nabais and M.A. Botto. Hierarchical model predictive control for optimizing intermodal container terminal operations. In *Proceedings of the 16th IEEE Conference on Intelligent Transportation Systems*, Jan 2013. doi: 10.1109/ITSC.2013.6728314.

[43] R.R. Negenborn J.L. Nabais, R.B. Carmona Benítez, and M.A. Botto. Achieving transport modal split targets at intermodal freight hubs using a model predictive approach. *Transportation Research Part C*, 60:278–297, Nov 2015. ISSN 0968-090X. doi: 10.1016/j.trc.2015.09.001.

[44] T. Jonker, M.B. Duinkerken, N. Yorke-Smith, A. de Waal, and R.R. Negenborn. Coordinated optimization of equipment operations in a container terminal. *Flexible Services and Manufacturing Journal*, Jul 2019. doi: 10.1007/s10696-019-09366-3.

[45] C. Kadaifci, U. Asan, S. Serdarasan, and U. Arican. A new rule-based integrated decision making approach to container transshipment terminal selection. *Maritime Policy & Management*, 46(2):237–256, Jun 2018. doi: 10.1080/03088839.2018.1489149.

[46] Bohdan T. Kulakowski, John F. Gardner, and J. Lowen Shearer. *Dynamics Modeling and Control of Engineering Systems third edition*. Cambridge University Press, 2007. ISBN 978-1-107-65044-2.

[47] A.M. Law and W.D. Kelton. *Simulation Modeling and Analysis third edition*. Prentice Hall, 2000. ISBN 0-07-059292-6.

[48] D-H. Lee, J.X. Cao, Q. Shi, and J.H. Chen. A heuristic algorithm for yard truck scheduling and storage allocation problems. *Transportation Research Part E: Logistics and Transportation Review*, 45:810–820, 2009. ISSN 1366-5545. doi: 10.1016/j.tre.2009.04.008.

[49] D-H. Lee, J.G. Jin, and J.H. Chen. Terminal and yard allocation problem for a container transshipment hub with multiple terminals. *Transportation Research Part E: Logistics and Transportation Review*, 48:516–528, 2012. ISSN 1366-5545. doi: 10.1016/j.tre.2011.09.004.

[50] N. Li, G. Chen, K. Govindan, and Z. Jin. Disruption management for truck appointment system at a container terminal: A green initiative. *Transportation Research Part D: Transport and Environment*, 61:261–273, Jun 2018. ISSN 1361-9209. doi: 10.1016/j.trd.2015.12.014.

[51] J. Lygeros, K. H. Johansson, S. N. Simic, J. Zhang, and S. S. Sastry. Dynamical properties of hybrid automata. *IEEE Transactions on Automatic Control*, 48 (1):2–17, Jan 2003. ISSN 0018–9286. doi: 10.1109/TAC.2002.806650.

[52] F. Meisel and C. Bierwirth. Heuristics for the integration of crane productivity in the berth allocation. *Transportation Research Part E: Logistics and Transportation Review*, 45(1):196–209, Jan 2009. ISSN 1366-5545. doi: 10.1016/j.tre.2008.03.001.

[53] M. Mourao, M. Pato, and A. Paixao. Ship assignment with hub and spoke constraints. *Maritime Policy & Management*, 29(2):135–150, Apr 2002. ISSN 0308-8839. doi: 10.1080/03088830110078364.

[54] A. Nagurney, Q. Qiang, and L. Nagurney. Environmental impact assessment of transportation networks with degradable links in an era of climate change. *International Journal of Sustainable Transportation*, 4(3):154–171, Jul 2010. doi: 10.1080/15568310802627328.

[55] R. Negenborn, B. De Schutter, and J. Hellendoorn. Multi-agent predictive control for transportation networks: serial versus parallel schemes. *Engineering Applications of Artificial Intelligence*, 21(3):353–366, Apr 2008. ISSN 0952-1976. doi: 10.1016/j.engappai.2007.08.005.

[56] N. Nehme, M. Awad, and I.A. Kaysi. An epsilon bargaining game-theoretic formulation between carrier and container terminal operators for servicing vessels during unloading operations. *Journal of Intelligent Transporation Systems*, 20(3):270–281, Sep 2015. doi: 10.1080/15472450.2015.1065740.

[57] E. Nishimura, A. Imai, and S. Papadimitriou. Berth allocation planning in the public berth system by genetic algorithms. *European Journal of Operational Research*, 131(2):282–292, Jun 2001. ISSN 0377-2217. doi: 10.1016/S0377-2217(00)00128-4.

[58] V. I. Norkin, G. C. Pflug, and A. Ruszczyński. A branch and bound method for stochastic global optimization. *Mathematical Programming*, 83(1-3):425–450, Jul 1998. ISSN 0025-5610. doi: 10.1007/BF02680569.

[59] T. Notteboom. A carrier's perspective on container network configuration at sea and on land. *Journal of International Logistics and Trade*, 1(2):65–87, Jun 2004. doi: 10.24006/jilt.2004.1.2.65.

[60] T. Notteboom and J-P. Rodrigue. The corporate geography of global container terminal operators. *Maritime Policy & Management*, 39(3):249–279, Apr 2012. ISSN 14645254. doi: 10.1080/03088839.2012.671970.

[61] A. I. La Ode. Private communication. Technical report, Indonesia Port Corporation, 2016.

[62] F. Parola, G. Satta, and S. Caschili. Unveiling co-operative networks and hidden families in the container port industry. *Maritime Policy & Management*, 41(4):384–404, May 2014. ISSN 14645254. doi: 10.1080/03088839.2013.782442.

[63] K.M. Passino and P.J. Antsaklis. On the optimal control of discrete event systems. In *Proceedings of the 28th IEEE Conference on Decision and Control*, Dec 1989. doi: 10.1109/CDC.1989.70672.

[64] M. E. H. Petering and K. G. Murty. Effect of block length and yard crane deployment systems on overall performance at a seaport container transshipment terminal. *Computers & operations Research*, 36(5):1711–1725, May 2009. doi: 10.1016/j.cor.2008.04.007.

[65] X. Qi and D. Song. Minimizing fuel emissions by optimizing vessel schedules in liner shipping with uncertain port times. *Transportation Research Part E: Logistics and Transportation Review*, 48(4):863–880, Jul 2012. ISSN 1366-5545. doi: 10.1016/j.tre.2012.02.001.

[66] T. Qin, Y. Du, and M. Sha. Evaluating the solution performance of ip and cp for berth allocation with time-varying water depth. *Transportation Research Par E*, 87:167–185, Mar 2016. ISSN 1366-5545. doi: 10.1016/j.tre.2016.01.007.

[67] B. Raa, W. Dullaert, and R. van Schaeren. An enriched model for the integrated berth allocation and quay crane assignment problem. *Expert Systems with Applications*, 38(11):14136–14147, Oct 2011. ISSN 0957-4174. doi: 10.1016/j.eswa.2011.04.224.

[68] P. J. G. Ramadge and W. M. Wonham. The control of discrete event systems. In *Proceedings of the IEEE*, pages 81–98, Jan 1989. doi: 10.1109/5.21072.

[69] Karl Johan Åstrom and Richard M. Murray. *Feedback Systems An Introduction for Scientists and Engineers*. Princeton University Press, 2008. ISBN 978-0-691-13576-2.

[70] B. D. Schutter and T. J. J. van den Boom. Model predictive control for discrete-event systems with soft and hard synchronization constraints. *International Journal of Control*, 76(1):82–94, Nov 2003. doi: 10.1080/00207170210000049188.

[71] B. D. Schutter and T. J. J. van den Boom. Modeling and solving the tactical berth allocation problem. *Transportation Research Part B: Methodological*, 44 (2):232–245, Feb 2010. ISSN 0191-2615. doi: 10.1016/j.trb.2009.07.003.

[72] B. D. Schutter and T. J. J. van den Boom. A lamda-optimal based heuristic for the berth scheduling problem. *Transportation Research Part C: Emerging Technologies*, 18(5):794–806, Oct 2010. ISSN 0968-090X. doi: 10.1016/j.trc.2009.07.001.

[73] A. Seidscher and S. Minner. A semi-markov decision problem for proactive and reactive transshipments between multiple warehouses. *European Journal of Operational Research*, 230(1):42–52, Oct 2013. ISSN 0377-2217. doi: 10.1016/j.ejor.2013.03.041.

[74] D. Simon, J. Löfberg, and T. Glad. Nonlinear model predictive control using feedback linearization and local inner convex constraint approximations. In *Proceedings of the 12th European Control Conference*, pages 2056–2061, Dec 2013. ISBN 978-3-033-03962-9. doi: 10.23919/ECC.2013.6669575.

[75] R. Stahlbock and S. Voß. Operations research at container terminals: a literature update. *OR Spectrum*, 30(1):1–52, Oct 2007. ISSN 1436-6304. doi: doi.org/10.1007/s00291-007-0100-9.

[76] D. Steenken, S. Voß, and R. Stahlbock. Container terminal operation and operations research - a classification and literature review. *OR Spectrum*, 26 (1):3–49, Jan 2004. ISSN 1436-6304. doi: 10.1007/s00291-003-0157-z.

[77] Hamdy A. Taha. *Operations Research An Introduction ninth edition*. Prentice Hall, 2011. ISBN 978-0-13-139199-4.

[78] A.A. Tako and S. Robinson. The application of discrete event simulation and system dynamics in the logistics and supply chain context. *Decision Support Systems*, 52(4):802–815, Mar 2012. ISSN 0167-9236. doi: 10.1016/j.dss.2011.11.015.

[79] R. Tavakkoli-Moghaddam, A. Makui, S. Salahi, M. Bazzazi, and F. Taheri. An efficient algorithm for solving a new mathematical model for a quay crane scheduling problem in container ports. *Computers & Industrial Engineering*, 56(1):241–248, Feb 2009. ISSN 0360-8352. doi: doi.org/10.1016/j.cie.2008.05.011.

[80] R.J. Tersine. *Principles of Inventory and Materials Management*. Prentice Hall, 1994. ISBN 0-13-126484-2.

[81] J.L. Tongzon. Port choice and freight forwarders. *Transportation Research Part E: Logistics and Transportation Review*, 45:186–195, 2009. ISSN 1366-5545. doi: 10.1016/j.tre.2011.09.004.

[82] N. K. Tran and H-D. Haasis. Literature survey of network optimization in container liner shipping. *Flexible Services and Manufacturing Journal*, 27(2–3): 139–179, Jun 2015. ISSN 1936-6582. doi: 10.1007/s10696-013-9179-2.

[83] UNCTAD. Review of maritime transport 2018. Technical report, United Nations Conference on Trade and Development, 2018.

[84] F.B. van Boetzelaer, T.J.J. van den Boom, and R.R. Negenborn. Model predictive scheduling for container terminals. *IFAC Proceedings Volumes*, 47(3): 5091–5096, Aug 2014. ISSN 2589-3653. doi: 10.3182/20140824-6-ZA-1003.00134.

[85] C. van der Baan, A. Dipo, D. Wignall, D. Takola, H. Fahmiasari, V. Hinssen, H. Sandee, and R. Meeuws. State of logistics indonesia 2015. Technical report, The World Bank, 2015.

[86] R.J.H.A. van Zijverden and R.R. Negenborn. Survey of approaches for integrated control of intermodal container terminals. In *Proceedings of the 9th IEEE International Conference on Networking, Sensing and Control*, pages 67–72, Apr 2012. doi: 10.1109/ICNSC.2012.6204893.

[87] Y. Wang and K. Cullinane. Traffic consolidation in east asian container ports: A network flow analysis. *Transportation Research Part A: Policy and Practice*, 61:152–163, Mar 2014. ISSN 0965-8564. doi: 10.1016/j.tra.2014.01.007.

[88] B. Wiegmans, A. van der Hoest, and T. Notteboom. Port and terminal selection by deep-sea container operators. *Maritime Policy & Management*, 35(6):517–534, Nov 2008. ISSN 1464-5254. doi: doi.org/10.1080/03088830802469329.

[89] J. Xin, R. R. Negenborn, and G. Lodewijks. Energy-aware control for automated container terminals using integrated flow shop scheduling and optimal control. *Transportation Research Part C: Emerging Technologies*, 44:214–230, Jul 2014. ISSN 0968-090X. doi: doi.org/10.1016/j.trc.2014.03.014.

[90] J. Xin, R. R. Negenborn, and G. Lodewijks. Event-driven receding horizon control for energy-efficient container handling. *Control Engineering Practice*, 39:45–55, 2015. ISSN 0967-0661. doi: doi.org/10.1016/j.conengprac.2015.01.005.

[91] J. Xin, R. R. Negenborn, and T. van Vianen. A hybrid dynamical approach for allocating materials in a dry bulk terminal. *IEEE Transactions on Automation Science and Engineering*, 15(3):1326–1336, Jul 2018. ISSN 1558-3783. doi: 10.1109/TASE.2017.2784483.

[92] Y. Yang, S. Pan, and E. Ballot. Innovative vendor-managed inventory strategy exploiting interconnected logistics services in the physical internet. *International Journal of Production Research*, 55(9):2685–2702, 1366-588X 2017. ISSN 0925-5273. doi: 10.1080/00207543.2016.1275871.

[93] C. Zhang, L. Zheng, Z. Zhang, L. Shi, and A. J. Amstrong. The allocation of berths and quay cranes by using a sub-gradient optimization technique. *Computers & Industrial Engineering*, 58(1):40–50, Feb 2010. ISSN 0968-090X. doi: doi.org/10.1016/j.cie.2009.08.002.

[94] L. Zhen. Modeling of yard congestion and optimization of yard template in container ports. *Transportation Research Part B: Methodological*, 90:83–104, Aug 2016. ISSN 0191-2615. doi: doi.org/10.1016/j.trb.2016.04.011.

Summary

This thesis discusses the dynamical modeling of complex container terminal operations. The operations are usually divided into three main parts namely seaside, storage, and transfer. In the current literature, the systems are usually modeled in static way using linear programming techniques. This setting does not completely capture the dynamic aspects in the operations, where information about external factors such as ships and trucks arrivals or departures and also the availability of terminal's equipment (QC, YC, IT) can always change.

We propose a dynamical modeling of container terminal operations using discrete-event systems (DES) modeling framework. The basic framework in this thesis is the DES modeling for berth and quay crane allocation problem (BCAP) where the systems are not only dynamic, but also asynchronous. In order to handle such dynamic environment, we propose a novel berth and QC allocation method, namely the model predictive allocation (MPA) which is based on model predictive control principle and rolling horizon implementation.

The DES models with asynchronous event transition are mathematically analyzed to show the efficacy of our method. We study an optimal input allocation problem for a class of discrete-event systems with dynamic input sequence (DESDIS). We show that in particular, the control input can be obtained by the minimization/maximization of the present input sequence only.

The DES modeling framework is extended to the cases in seaport network and integrated container terminal operations. We study container terminal network performance under heterogeneous distributed operational BCAP policy. We investigate the performance MPA-based BCAP to improve network operations. We also propose methods for selecting important seaports in the network to which the MPA-based BCAP policies are applied. The MPA give better results compared to the state-of-the-art methods in BCAP. In the second case, we propose a dynamical modeling of integrated (end-to-end) container terminal operations using finite state machine (FSM) framework where each state machine is represented by a

discrete-event system (DES) framework. The proposed MPA method allows us to plan the terminal operations integratively and simultaneously. We have shown that the proposed approach performed better than the existing method used in the studied terminal and state-of-the-art methods in the literature.

Samenvatting

Dit proefschrift gaat over het dynamisch modelleren van complexe containerterminal activiteiten. De activiteiten worden doorgaans verdeeld in drie onderdelen, namelijk kust, opslag, en overslag. In de huidige literatuur worden deze systemen doorgaans gemodelleerd op een statische wijze door middel van lineaire programmeringstechnieken. In deze setting worden dynamische aspecten in de activiteiten niet volledig meegenomen, omdat informatie over externe factoren, zoals de aankomst of het vertrek van schepen en vrachtwagens en de beschikbaarheid van voorzieningen van de terminal (kadekranen, werfkranen, IT), continu aan verandering onderhevig zijn.

In dit proefschrift wordt het modelleren van containerterminal activiteiten verricht volgens het discrete-event systemen (DES) modelleringskader. Het basiskader hiervoor is het DES modelleren van de ligplaats en kadekraan toewijzingsprobleem (BCAP), waarbij de systemen een dynamisch én asynchroon karakter bezitten. Deze eigenschappen in acht nemend, stellen we een innovatieve ligplaats en kadekraan toewijzingsmethode voor, namelijk een model predictive allocatie (MPA), welke berust op model predictive control en een rollende horizon implementatie.

Om de doeltreffendheid van onze methode aan te tonen zijn de DES modellen met asynchrone event-overgangen op wiskundige wijze geanalyseerd. We bestuderen een optimaal invoertoewijzingsprobleem voor een klasse van discrete-event systemen met een dynamische invoerreeks (DESDIS).In het bijzonder laten we zien dat de invoerregeling verkregen kan worden door slechts de huidige invoerreeks te minimaliseren/maximaliseren.

Het DES modelleringskader is vervolgens uitgebreid naar de situaties van zeehaven netwerken en geïntegreerde containerterminal activiteiten. We bestuderen de prestatie van een containerterminal netwerk volgens het beleid van heterogene gedistribueerde BCAP activiteiten. We onderzoeken ook de prestatie van het op MPA-gebaseerde BCAP beleid voor het verbeteren van netwerk activiteiten. Daarnaast bieden we methoden aan voor het selecteren van belangrijke zeehavens

binnen het netwerk waarbij het op MPA-gebaseerde BCAP beleid wordt toegepast. Het op MPA-gebaseerde beleid geeft betere resultaten in vergelijking met de allernieuwste methoden in BCAP. In het tweede geval bieden we een ontwerp aan voor het dynamisch modelleren van geïntegreerde (eind-tot-eind) containerterminal activiteiten middels het eindigetoestandsautomaat (FSM) raamwerk. Hierbij wordt elke toestandsautomaat vertegenwoordigd door een discrete-event systeem (DES). De voorgestelde MPA methode staat ons toe om de terminal activiteiten integraal en simultaan te plannen. We hebben laten zien dat de voorgestelde aanpak beter presteert dan de bestaande methode, terwijl deze methode volop gebruikt wordt en tot de allernieuwste methoden uit de literatuur behoort.