A TIMED PETRI NET MODEL FOR THE QUAY CRANE SCHEDULING PROBLEM

Roberto Trunfio
LabDoc - Dipartimento LISE
Università della Calabria
Rende (CS), Italy
roberto.trunfio@unical.it

KEYWORDS
Timed Petri Net; Simulation; Maritime Container Terminal; Scheduling; Quay Cranes

ABSTRACT

This paper deals with the problem of constructing the schedule for the operations of a group of quay cranes devoted to discharge/load a set of groups of containers from a vessel at a maritime container terminal. The schedule is constructed starting from the assignment of each individual group of containers to a quay crane under the goal of minimizing the overall vessel completion time, aka the *makespan*. The assignment is provided, e.g., by the search process of an optimization algorithm designed for solving the so called *quay crane scheduling problem*. In this paper, a novel Timed Petri Net model is proposed to construct the schedule from a given assignment. As a novelty, the proposed model considers the initial and final location of the quay cranes to ensure that some necessary physical constraints are satisfied during the idle periods. It also defines an easy-to-implement set of rules to construct the schedule such that the makespan is minimum.

INTRODUCTION

A maritime container terminal (MCT) is a facility located into a port where containers are stored and transshipped between land and ship transports for subsequent transportation. A large number of logistical processes arise in a MCT. In a holistic approach, these processes can be simulated all together by resorting to a high level simulation framework (Legato et al. 2008b), which clearly omits some details from the terminal activities to keep the computational tractability of the model. In fact, currently a detailed representation of all the features of the logistical processes can be obtained by developing a specialized simulation model focused on a single logistical process.

Since the core container transport mode is via ship, the operations focused on the vessels are of primary interest for the terminal managers focused on minimizing the lead time (Steenken et al. 2004). In fact, in literature one of the most studied logistical problems is related to the discharge/loading (D/L) operations of a vessel (Stahlbock and Voß 2008). Vessel D/L operations are performed by resorting to a pool of quay cranes (QCs) that travel on rails. Each vessel is divided into bays and each bay can be partitioned in two areas located below and above the deck, respectively. Each container is located within a bay and must be discharged or loaded at the port of call according to a predefined stowage plan. The problem of indentifying the optimal sequence of the container D/L operations under the objective that the overall vessel operations (i.e. the *makespan*) is minimized; it is known in literature as the *quay crane scheduling problem* (QCSP) (Daganzo 1989; Kim and Park 2004).

When an optimization algorithm is constructed to solve a specific QCSP formulation, a method to evaluate the schedule is required. A schedule is a sequence of activities and events (Pinedo 2002), thus an event-based simulation model can accomplish the task of evaluating the makespan of a schedule for the QCSP. For instance, this approach has been pursued in (Legato et al. 2012; Legato and Trunfio 2013; Trunfio 2014).

Timed Petri Net (TPN) models are an effective and powerful modeling tool to be used in this context (Zurawski 1994; Mejía and Montoya, 2010). In fact, TPN has no modeling limitation that would make the results from the simulation model deviate from the necessary model features. Moreover, in one of the latest formulation of the QCSP proposed by Legato et al. (2012), a TPN has been proposed to calculate the schedule makespan as well as the individual completion time of the specific D/L operations, which has been a plus with respect to all the known methodologies.

Recently, Chen et al. (2014) considered the QCSP by taking into account for additional, but necessary features. Therefore, a novel TPN model that can be used to capture these new features is presented in this paper. Moreover, the proposed model: *(i)* provides an easy-to-understand set of rules to ensure that the computed makespan is minimum with respect to the given assignment; and *(ii)* defines an easy-to-read description of the QC operations. The proposed TPN model has been validated by comparing the event-list obtained from other simulation methods from the literature on a set of instances from the literature.

The paper is organized as follows. The next section briefly introduces the necessary background on the QCSP. The subsequent section describes the proposed TPN. The second-last section provides a complete

Proceedings 28th European Conference on Modelling and Simulation ©ECMS Flaminio Squazzoni, Fabio Baronio, Claudia Archetti, Marco Castellani (Editors)
ISBN: 978-0-9564944-8-1 / ISBN: 978-0-9564944-9-8 (CD)

modeling example. The last section concludes the paper.

THE QUAY CRANE SCHEDULING PROBLEM

The QCSP is a scheduling problem that, according to the modern worldview, is tackled by considering *(i)* the containers from a bay as a unique indivisible group or *(ii)* the containers from a bay to be partitioned in groups. The first is known as *QCSP for complete bays* and the latter *QCSP for container groups*. In this paper we dealt with the QCSP for container groups, since it is clearly more challenging and realistic. Furthermore, the former is a special case of the latter.

The modern formulation of the QCSP for container groups (or for simplicity only QCSP from now on) has been proposed by Kim and Park (2004) by enriching a *multiple travelling salesman problem* formulation. Therefore, the Kim and Park formulation is $\mathcal{NP}$-hard. Several refinements have been proposed in the subsequent years; the major refinements are due to Bierwirth and Meisel (2009), who corrected the representation of some features related to the movement and locations of the QCs during the D/L operations. Moreover, (Chen et al. 2014) has pointed out the attention on the final an initial location of the QCs; the proposed formulation defines the optimal schedule under the assumption that the QC movements are unidirectional (Liu et al. 2006), but removes some modeling features useful for constructing the schedule, e.g. the completion time of the specific tasks.

In the QCSP, a group of containers identifies a task. The set of all the tasks is Ω and $|\Omega| = n$. Each task i has a processing time p_i. A task is located in a bay and the corresponding location is l_i. A task must be assigned to exactly one QC from the set Q, such that $|Q| = m$. The containers related to a bay are divided into groups according to the location of the containers in a bay (below or above the deck), the requested operation (discharge or loading) and the stowage plan (e.g. containers from the same bay area that must be discharged first to speed-up the loading operations of another vessel). Clearly, these facts generate precedence relations for the processing of the tasks: from the same bay, discharge tasks must precede loading tasks; discharge (loading) tasks located above the deck must precede (follow) discharge (loading) tasks located below the deck; other precedence relations may be defined if needed (e.g. due to the stowage plan), but only by ensuring that the previous rules hold true. For a given couple *(i, j)* of tasks *i* and *j*, the set of all the precedence relations is Φ. Figure 1 illustrates an example of a vessel to be handled by using two-QCs.

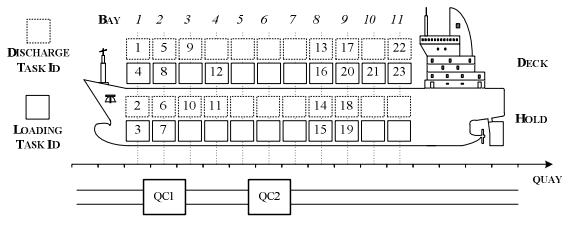

Figure 1: An example of feeder-vessel to D/L with 11 bays, two QCs and 23 tasks.

As shown in Figure 1, the quay side is discretized according to the length of a bay and, consequently, also the travel times of the QCs are calculated according to this space unit. To this extent, let $\hat{t}$ be the travel time required to cover the distance of a bay and $t_{l_i l_j}$ the travel time required to move from location l_i to l_j, i.e. $t_{l_i l_j} = |l_i - l_j| \cdot \hat{t}$.

The QCs serve along the time multiple vessels from the same berth according to what is defined by the *quay crane assignment and deployment problem* (Legato et al. 2008a; Bierwirth and Meisel 2010). Therefore, with respect to the vessel to be handled, each QC q must perform the assigned tasks within a time window defined by a ready time r_q and a due date d_q. Thus, a QC q has an initial location il_q and a final location fl_q, which are assumed by q at the ready time and when the D/L operations are completed, respectively. Observe that, while the initial location is generally given, the final location is calculated by the QCSP formulation according to the assignment of the tasks to the QCs. For convenience, the start time (i.e the *zero time*) of the QCSP schedule corresponds to the least ready time of the QCs.

Since the QCs travel on the same rails, then non-crossing constraints must be taken into account. Moreover, to avoid the collision of the QC booms, a safety distance δ must always be guaranteed between adjacent QCs. Clearly, assuming that the m QCs are numbered from 1 to m starting from left-to-right, then two QCs v and w (v < w) cannot work simultaneously on two tasks i and j, respectively, if holds true that $l_i > l_j - \delta_{vw}$, where $\delta_{vw} = (\delta + 1) \cdot |v - w|$ (observe that the case for v>w can be easily deduced). From this observation, Bierwirth and Meisel (2009) defined Δ_{ij}^{vw} as the time span to be elapsed between the completion of one between task i and task j and the remaining, under the assumption that i and j are assigned respectively to QCs v and w. Bierwirth and Meisel (2009) shown that only the 4-tuples *(i, j, v, w)* such that $\Delta_{ij}^{vw} > 0$ have to be considered to define non-crossing and safety distance constraints. Clearly, these constraints must be accounted for in order to guarantee that the schedule is correctly constructed. For convenience, the set of the aforementioned 4-tuples is defined as $\Theta = \{(i, j, v, w) \in \Omega^2 \times Q^2 | i < j \wedge \Delta_{ij}^{vw} > 0\}$.

THE TPN MODEL FOR THE QCSP

Petri Nets (PNs) are a powerful and formal modeling language introduced by Petri (1962) which can be represented as a bipartite graph, where nodes are *transitions* or *places* and a transition is connected to a place by an *arc* (and *vice versa*). In simulation, generally a transition represents an event, while a place represents a state. Model changes through the time occurs by using a *token*, a special entity that is put into a place to enable the transitions. A good lecture on modeling with PNs can be read here (Girault and Valk 2003).

There are several interesting extensions of the original language formulated by Petri and TPNs are one of

them. In the literature for the QCSP, a TPN has been proposed in (Legato et al. 2008c) by focusing on the D/L of each single container from a bay and modeling as a plus the operations performed by shuttle vehicles in the quay side; there, however, some important features were not considered (e.g. QC safety distance). Later, another TPN has been used by Legato et al. (2012) to construct the schedule of a given set of task-to-QC assignments. In that TPN model, the deterministic times are associated to transitions and the rules for computing the least cost makespan are regulated by "complicated" equations. We remedy these issues by resorting to a different approach to time modeling. As a matter of fact, another way to introduce time into a PN model is to associate the time with the arcs. Hence, this is the modeling approach pursued in the novel TPN proposed here for the QCSP.

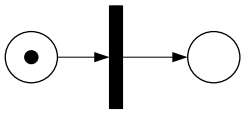

Figure 2: An example of a simple Petri Net: the large circles are places, the vertical black bar is a transition, arcs connect places and transitions and, finally, the small black circle is a token.

We represent the whole process of handling the containers from a vessel as a single-source single-sink TPN. A single token must be put into the source place in order to start the D/L operations. The basic schema of the TPN model is depicted in Figure 3. As shown in the figure, once the schedule evaluation/construction starts (from the "start" transition on), a TPN must be defined for each QC to model a sequence of operations. The QCs have to interact at specific interaction points, hidden from Figure 3, due to the constraints described in the previous section.

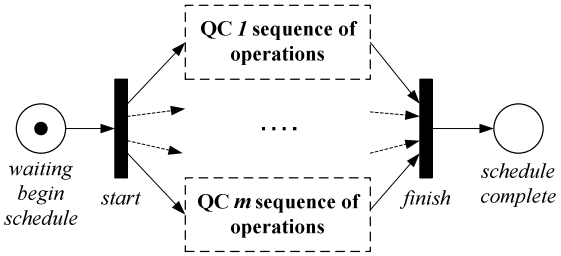

Figure 3: The skeleton of the model.

As suggested by van der Aalst (1996), we represent an *operation* performed by a QC as depicted in Figure 4. As shown in this figure, once that a token leaves the begin place, exactly t unit of times must be elapsed to fire the "finish" transition.

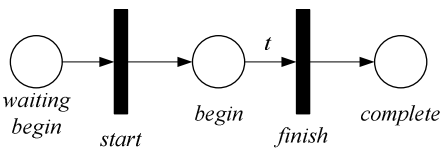

Figure 4: An operation of a QC.

Since the QCs perform the operations in sequence, when two operations are put one before the other in a QC sequence, the "complete" place of the first operation is merged with the "waiting begin" place of the second operation.

We identify four types of operations conducted by a QC during its life cycle: *setup, task execution, travel* and *completion*. Clearly, for a QC $q \in Q$ there is only one "setup operation" and one "completion operation", which are put, respectively, at the beginning and at the end of the sequence of operations of q; in the middle there is the sequence of "task execution operations", which is defined according to the ordered list S_q of tasks assigned to q. A "travel operation" is put between two "task execution operations", or a "setup operation" and a "task execution operation", or a "task execution operation" and a "completion operation", when needed (i.e. when the travel time is greater than zero). The deterministic time t depicted on the arc that connects the place "begin" with the transition "finish" assumes the value of the ready time, task processing time and travel time for the "setup operation", a "task execution operation" and a "travel operation", respectively. The "completion operation" has t=0 (therefore it is omitted from the arc). A remark is required for the travel time. When the "setup operation" of QC q and the "task processing operation" for task i are one before the other, then $t = t_{il_q l_i}$; if the "task processing operations" for two tasks i and j are one after the other, then $t = t_{l_i l_j}$; finally, if the "task processing operation" of task i precedes the "complete operation" of QC q, then $t = t_{l_i f l_q}$.

Given this premise, we have to model the following features: *(i)* precedence relations between tasks; *(ii)* QC non-crossing and safety distance for ready QC; *(iii)* QC location before the ready period; *(iv)* QC location after the completion of the last task; *(v)* QC due date.

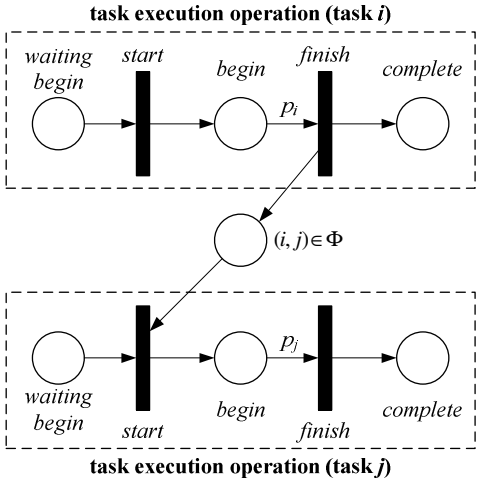

Figure 5: Representation of a precedence relation between two tasks.

Precedence relations between tasks

A precedence relation $(i,j) \in \Phi$, as proposed in (van der Aalst 1996), is modeled as shown in Figure 5. So, to model a precedence relation it is required an ad-

ditional place to be connected between the "finish" transition of the "task execution operation" of task i and the "start" transition of the "task execution operation" of task j, as illustrated in Figure 5. Clearly, if both tasks i and j are assigned to the same QC and i precedes j in the list S_q, then the precedence relations is implicitly accounted for and therefore should not be depicted; otherwise, since j precedes i, the schedule is unfeasible.

QC non-crossing and safety distance

QC non-crossing and safety distance, as explained in the previous section, must be accounted for any 4-tuple $(i,j,v,w) \in \Theta$. Therefore, given two "task execution operations" related to the couple of tasks i and j ($i<j$), which are assigned respectively to QCs v and w and $\Delta_{ij}^{vw} > 0$, the aforementioned constraints can be modeled as provided in Figure 6.

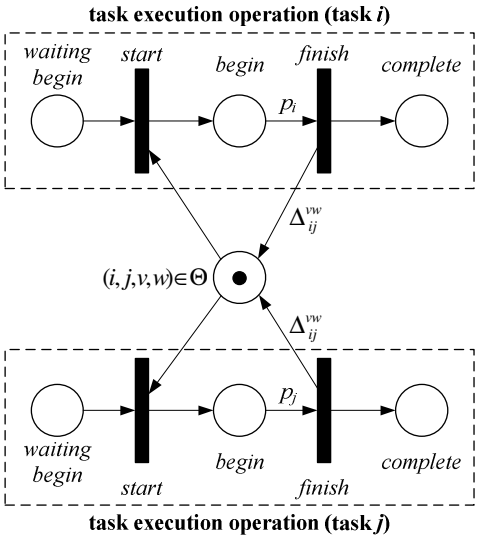

Figure 6: Representation of non-crossing and safety distance constraints.

QC location before the ready period

Since there is no guarantee that the initial location of a QC (i.e. the location assumed from time zero to the ready time) is the same of the location of the first task, then the QC location before the ready period must be accounted in order to avoid QC crossing and to guarantee the safety distance. To model this new feature, we define the set Ψ^I of 3-tuples (i,v,w) such that task i is assigned to QC v and v must wait the ready time of QC w to start working on i. Formally, the new set is defined as $\Psi^I = \{(i,v,w) \in \Omega \times Q^2 | i \in S_v \wedge (l_i > il_w - \delta_{vw} \vee l_i < il_w + \delta_{vw})\}$. To ensure that after the ready time of QC w there is no violation of non-crossing and safety distance requirements, the following time span is introduced:

$$\Delta_{il}^{vw} = \begin{cases} (l_i - il_w + \delta_{vw})\hat{t} & \text{if } v < w \text{ and } l_i > il_w - \delta_{vw}; \\ (il_w - l_i + \delta_{vw})\hat{t} & \text{else if } v > q \text{ and } l_i < il_w + \delta_{vw}; \\ 0 & \text{otherwise.} \end{cases}$$

This stated, Figure 7 illustrates how to model this case for any 3-tuple in set Ψ^I.

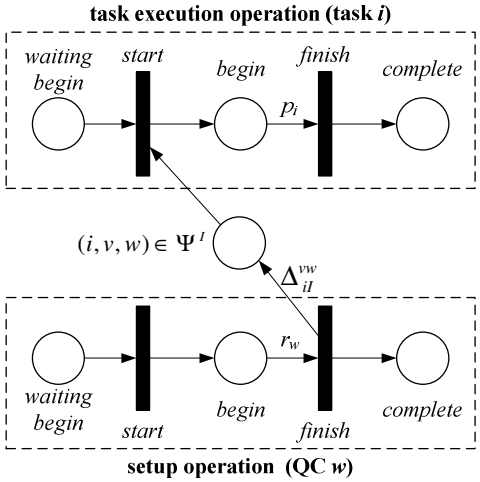

Figure 7: Representation of non-crossing and safety distance constraints for a QC not yet ready.

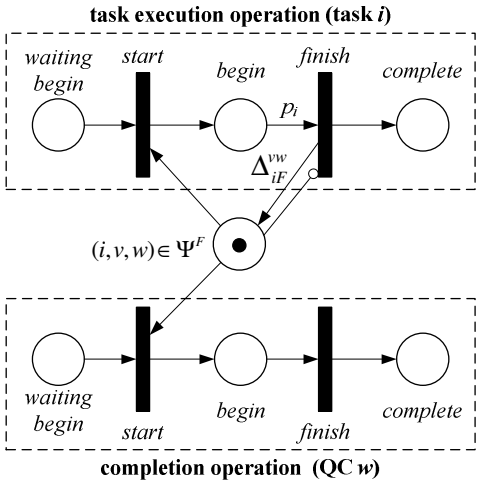

Figure 8: Representation of non-crossing and safety distance constraints for a QC that completed its schedule.

QC location after the completion of the last task

In a similar fashion, also the final location of a QC must be considered to guarantee the non-crossing and safety distance requirements. To this extent, it is introduced a new set, say Ψ^F, that, similarly to the definition of Ψ^I, we define $\Psi^F = \{(i,v,w) \in \Omega \times Q^2 | i \in S_v \wedge (l_i > fl_w - \delta_{vw} \vee l_i < fl_w + \delta_{vw})\}$. In this case the time span is defined as follows:

$$\Delta_{iF}^{vw} = \begin{cases} (l_i - fl_w + \delta_{vw})\hat{t} & \text{if } v < w \text{ and } l_i > fl_w - \delta_{vw}; \\ (fl_w - l_i + \delta_{vw})\hat{t} & \text{else if } v > q \text{ and } l_i < fl_w + \delta_{vw}; \\ 0 & \text{otherwise.} \end{cases}$$

We assume that once a QC w completes its operations it becomes definitely idle and cannot be moved along the rails until the completion of the vessel D/L operations. Thus, any assignment of a task i to a QC v such that $(i,v,w) \in \Psi^F$ must be completed before that w repositions to its final location fl_w; otherwise, the schedule is unfeasible. This new feature disables any "task operation" of a task i assigned to a QC v once that QC w repositions to its final location if and only if

$(i, v, w) \in \Psi^F$. As shown in Figure 8, this new feature is modeled by resorting to an inhibitor arc.

QC due date

For a given QC $q \in Q$, it must be guaranteed that q repositions to the final location fl_q before the due date d_q. So, whenever $d_q < \infty$, we model this requirement as shown in Figure 9. As shown in the figure, after $d_q - r_q$ time units from the ready time of QC q, an inhibitor arc is used to avoid that the schedule is completed. Clearly, from the time d_q the inhibitor arc starts to inhibit the completion of QC q and thus the sink is never reached. As a result, the given schedule is unfeasible.

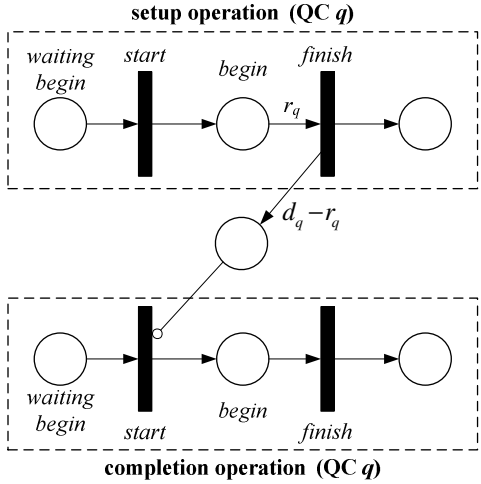

Figure 9: Representation of the due date for a QC.

As one may observe, the previous relation can be established between the "setup operation" and all the following operations: for simplicity, here we depict this requirement only between the "setup operation" and the "completion operation" of each QC.

Time handling and rules for firing

Since the problem at hand requires constructing the schedule such that the makespan is minimum, we introduce some assumptions to define the mechanisms used to handle and to update the simulated time. In particular, we assume that each token in the TPN brings through the graph some necessary information: *(i)* a variable to memorize the time, say $clock$; and *(ii)* a variable to save the index of the QC associated to the latest visited QC operation, say $crane$. At time zero, for each token in the TPN is set $clock = 0$ and $crane = null$. A remark for the $clock$ variable is due. Every time a token traverses an arc with a weight t, then for this token it is set $clock = clock + t$. Thus, we can observe that each token has its own $clock$ value. Whenever a transition fires, a new token is returned for each outgoing arc. For each of these generated tokens, if the traversed transition is from an operation of a QC $q \in Q$, then it is set $crane = q$; otherwise, it is set $crane = null$. Moreover, the $clock$ value is set equal to the largest time calculated overall the incoming arcs, where the time is obtained for each arc as the sum of the $clock$ values of the token consumed from the incoming place plus the time from the incoming arc.

An example of this approach is provided in Figure 10. For illustrative purposes only, all the tokens are named (e.g. tok_1, tok_2, etc). Figure 10 is divided into two parts: the former (Figure 10(a)) and the latter (Figure 10(b)) show the net before and after the firing of the sole transition, respectively. The transition from the figure has three incoming tokens, say tok_1, tok_2 and tok_3, from three different places. Since the transition is enabled, it may fire, but has to wait until 20 unit of times are elapsed due to the weight of the incoming arc in the middle. Once that the transition fires, two generated tokens, say *tok* (clearly, they are identical) are put in the outgoing places. The $clock$ of each token *tok* is set such that $clock = max\{10, 5 + 20, 15\} = 25$. The $crane$ variable is set to q, since the transition belongs from an operation of QC q.

Thus stated, the general rule to be applied when multiple transitions are enabled to fire is the "least delay rule", which implies that, given two transitions, the $clock$ value for an outgoing token is temporarily computed, but only the QC with the smaller value of $clock$ is selected for firing. If the two $clock$ values are equals and there is only one incoming arc for both the transitions, then both the transitions are enabled to fire; otherwise, the transition enabled to fire is selected at random (see for instance the case that models the "QC non-crossing and safety distance", which implies non-simultaneity between two "task operations" and so, that only one-out-of-two enabled transitions can fire).

Finally, the last firing rule follows here. Given a QC $q \in Q$ such that $d_q < \infty$, if at time d_q there is a token in the "waiting begin" place of the "completion operation", then the "start" transition of the same operation is enabled to fire; otherwise, the "start" transition becomes inhibited and therefore the schedule becomes unfeasible.

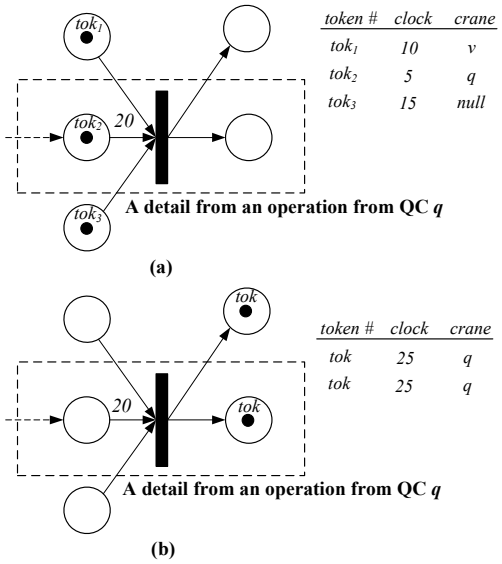

Figure 10: Example of time update. Figure 10(a) and Figure 10(b) show, respectively, the *clock* of the tokens before and after that the transition fires.

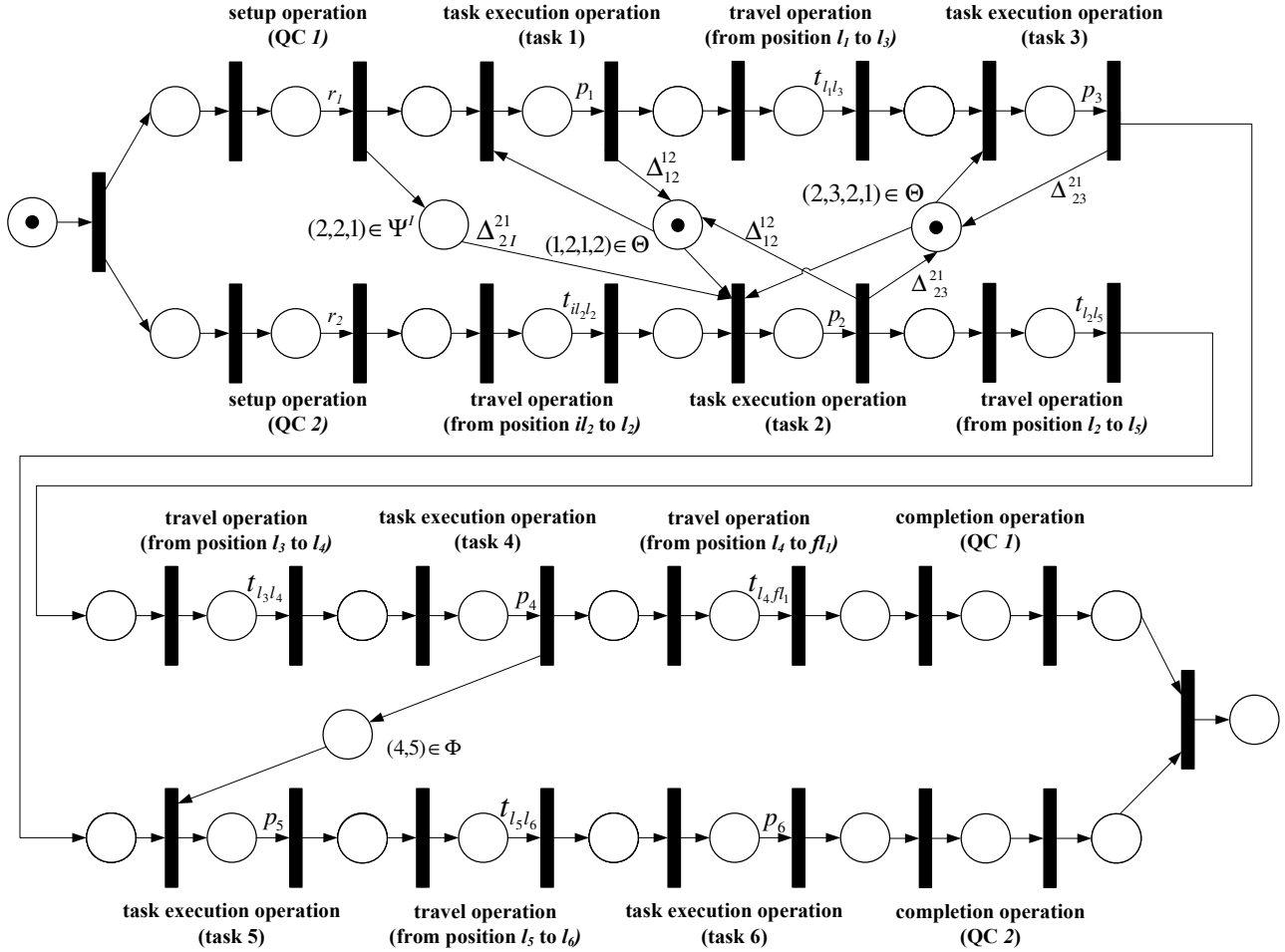

Figure 11: An example of a TPN model for the evaluation of a schedule in the QCSP.

AN EXAMPLE OF THE TPN FOR THE QCSP

An example of a TPN is provided here by starting from the instance illustrated in Figure 8 from Section 3.4 from (Legato et al. 2012). The QCSP instance is defined by the set of tasks Ω ={1, 2, 3, 4 , 5, 6}, the set of QCs Q={1, 2}. The pair of tasks (4, 5) and (5, 6) are, respectively, from the same bay. Furthermore we define the set of precedence relations Φ ={(4, 5), (5, 6)}. Moreover, we assume that task locations and the safety margin are given such that Θ = {(1, 2, 1, 2), (2, 3, 1, 2), (3, 5, 1, 2), (3, 6, 1, 2), (4, 5, 1, 2), (4, 6, 1, 2)}. In addition, we assume that the lists of tasks assigned to the QCs are as follows: S_1={1, 3, 4} and S_2={2, 5, 6}. The initial location of the QC 1 is in front of the bay of task 1 (which from the 4-tuple (1, 2, 1, 2) $\in \Theta$ implies that (2, 2, 1)$\in \Psi^I$, while the initial location of QC 2 is in front of the bay of task 6. For what concerns the final location, we assume that both the QCs comes back to the initial location (i.e. initial and final location coincides): therefore, from {(3, 6, 1, 2), (4, 6, 1, 2)} $\in \Theta$ (or alternatively from {(3, 5, 1, 2), (4, 5, 1, 2)} $\in \Theta$) we have that {(3, 1, 2), (4, 1, 2)}$\in \Psi^F$. Finally, no due date is set for both the QCs.

Thus, in Figure 11 we report the model that describes this instance. Clearly, processing times as well as travel times are provided as generic values. In the proposed example, no due date is considered, so the relative modeling features are not reported. Moreover, from S_2 comes that the precedence relation (5, 6) is implicitly accounted and thus the relative constraint is not modeled. Finally, the two 3-tuples in Ψ^F are not considered as well, since, according to the precedence relation (4, 5), both tasks 3 and 4 must be completed by QC 1 before that QC 2 can reposition to its final location.

Observations

Some observations on the proposed TPN model are listed in the following. First, the travel time could be considered in a more concise way, but this would remove some interesting features of the model related to representation of the QC movements along the quay. Moreover, without a doubt, the size of the proposed model can be further reduced, e.g. by removing a lot of couples place-transition that are merely represented to model the *start-finish* events of each operation and are not involved in any kind of constraints (i.e., precedence relations, non-crossing, safety distance). Finally, observe that if the given task lists per QC or the initial/final location of the QCs are not compliant to a well-formed mathematical formulation, then the sink cannot be reached and therefore the corresponding schedule is assumed to be unfeasible. A mathematical

formulation that tackles all these features will be provided in a companion paper.

CONCLUSIONS

We have proposed a methodology for the construction of the schedule for the vessel discharge and loading operations of groups of containers by means of a pool of quay cranes in a maritime container terminal. The methodology is aimed to be used as the makespan evaluation method for the well known *Quay Crane Scheduling Problem*. The methodology consists of a Timed Petri Net model. The proposed model provides a clear representation of the events that occur during the vessel discharge/loading operations. Moreover, it provides a simple set of rules to be used to construct the schedule under the objective of makespan minimization. The proposed TPN model can be implemented and analyzed through any of the several available tools, like *GreatSPN* (Ajmone Marsan 1995; Baarir et al. 2009). Finally, a black-box Java implementation of the simulation model is freely available upon request to the author or at the author's website.

ACKNOWLEDGEMENTS

The author thanks *Prof. Pasquale Legato* (DIMES, Università della Calabria, Italy) for his teaching and mentoring effort that have been a springboard for the writing of this paper. Moreover, the author would like to thank the four anonymous reviewers for their valuable comments and suggestions to improve the paper.

REFERENCES

S. Baarir, M. Beccuti, D. Cerotti, M. De Pierro, S. Donatelli, G. Franceschinis, "The GreatSPN tool: recent enhancements". *Performance Evaluation Review,* vol. 36 (4), pp. 4–9, 2009.

C. Bierwirth, F. Meisel, "A fast heuristic for quay crane scheduling with interference constraints". *Journal of Scheduling*, vol. 12, pp. 345–360, 2009.

C. Bierwirth, F. Meisel, "A survey of berth allocation and quay crane scheduling problems in container terminals". *European Journal of Operational Research*, vol. 202, pp. 615–627, 2010.

J. H. Chen, D. H. Lee, M. Goh, "An effective mathematical formulation for the unidirectional cluster-based quay crane scheduling problem". *European Journal of Operational Research*, vol. 232(1), pp. 198–208, 2014.

C. Girault, R. Valk, "Petri Nets for systems engineering: a guide to modeling, verification, and applications". Springer, 2003.

C. Daganzo, "The crane scheduling problem". *Transportation Research Part B*, vol. 23(3), pp.159–175, 1989.

K. Kim, Y. Park, "A crane scheduling method for port container terminals". *European Journal of Operations Research*, vol. 156, pp. 752–768, 2004.

P. Legato, D. Gullì, R. Trunfio, "The quay crane deployment problem at a maritime container terminal". Proc. of the *22th European Conference on Modelling and Simulation* (ECMS 2008), pp 53–59. Nicosia (Cyprus), June 3-6, 2008a. DOI: 10.7148/2008-0053.

P. Legato, D. Gullì, R. Trunfio, R. Simino, "Simulation at a maritime container terminal: models and computational frameworks". Proc. of the *22th European Conference on Modelling and Simulation* (ECMS 2008), pp 261–269. Nicosia (Cyprus), June 3-6, 2008b. DOI: 10.7148/2008-0261.

P. Legato, D. Gullì, R. Trunfio, "Modelling, simulation and optimization of logistic systems". Proc. of the *20th European Modeling and Simulation Symposium (Simulation in Industry)* (EMSS 2008), ISBN: 978-88-903724-0-7, pp. 569–578. Amantea (Italy), September 17-19, 2008c.

P. Legato, R. Trunfio, F. Meisel, "Modeling and solving rich quay crane scheduling problems". *Computers and Operations Research*, vol. 39(9), pp. 2063–2078, 2012. DOI: 10.1016/j.cor.2011.09.025.

P. Legato, R. Trunfio, "A local branching-based algorithm for the quay crane scheduling problem under unidirectional schedules". *4OR - A Quarterly Journal of Operations Research*, 2013. DOI: 10.1007/s10288-013-0235-2.

J. Liu, Y.W. Wan, L. Wang, "Quay crane scheduling at container terminals to minimize the maximum relative tardiness of vessel departures". *Naval Research Logistics*, vol. 53(1), pp. 60–74, 2006.

M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, G. Franceschinis. Modelling with Generalized Stochastic Petri Nets. J. Wiley, 1995.

G. Mejía, C. Montoya, "Applications of resource assignment and scheduling with Petri Nets and heuristic search". *Annals of Operations Research*, vol. 181(1), pp. 795–812, 2010.

C. A. Petri, "Kommunikation mit automaten". PhD thesis. University of Bonn, 1962.

M. Pinedo, "Scheduling theory, algorithms, and systems". Prentice Hall, 2002.

D. Steenken, S. Voß, R. Stahlbock, "Container terminal operation and operations research - a classification and literature review", in OR Spectrum, vol. 26, pp. 3-49, 2004.

R. Stahlbock, S. Voß, "Operations research at container terminals - a literature update", OR Spectrum, vol. 30, pp. 1–52, 2008.

R. Trunfio, "A note on: A modified generalized extremal optimization algorithm for the quay crane scheduling problem with interference constraints", Engineering Optimization, *in press*, 2014.

W. M. P. van der Aalst, "Petri net based scheduling". *OR Spektrum*, vol. 18(4), pp. 219–229, 1996. DOI: 10.1007/BF01540160.

R. Zurawski, "Petri nets and industrial applications: A tutorial". *IEEE Transactions on Industrial Electronics*, vol. 41(6), pp.567–583, 1994. DOI: 10.1109/41.334574.

AUTHOR BIOGRAPHY

ROBERTO TRUNFIO gained a Ph.D. in Operations Research at the Department of Electronics, Informatics and Systems (DEIS), University of Calabria, Italy, in 2009. He has been senior engineer at NEC where he has worked on optimization and simulation techniques applied to logistics. He currently holds a PostDoc at LabDoc (University of Calabria). His research interests include decision support systems, discrete-event simulation models, simulation-based optimisation, optimization algorithms, semantic web, ontologies, text mining, natural language processing. His home page is at www.roberto.trunfio.it.