Dynamic Optimization of Quay Crane Assignment and Scheduling in Container Terminals: A Strategy-Sensitive Event-Driven MILP Approach

Essowiyaou Kassankogno[1,2], Sid Lamrous[1], Marie-Ange Manier[1], Yaogan Mensah[2]

***Abstract*— This study explores the dynamic quay crane assignment and scheduling problem in container terminals. The problem is modeled as a Mixed-Integer Linear Program (MILP) and solved using a rolling-horizon framework, guided by an event-driven orchestrator that recalculates the model upon the occurrence of specific events. These events include vessel arrivals, departures, and crane releases, depending on the chosen strategy. The compared strategies are defined by task continuity (or not) on the same crane, and optional reoptimization upon crane release, tested under four objectives structures based on vessel tardiness and stay time, including two lexicographic variants. Sixteen rescheduling policies are evaluated on twelve generated instances. The results show that the objective structure impacts the global trade-off, whereas the detailed analysis highlights the differentiated and potentially positive roles of task continuity on the same crane and reoptimization upon crane release.**

I. INTRODUCTION

The efficiency of container terminal operations is influenced by quay-side decisions, including berth allocation, crane assignment to vessels, and scheduling of loading and unloading tasks [1]–[3]. Quay cranes are central resources as their productivity affects vessel processing time and then stay time at port [4]–[6]. In terminals, cranes move on a common rail, which implies non-crossing and safety-distance constraints [7], [8]. The problem is therefore a combined assignment and scheduling problem, which explains the sustained interest in integrated formulations [4], [5], [8]–[10]. Among them, static formulations do not fully represent the operational evolution of a terminal. Vessel arrivals and departures modify the system state, crane availability changes over time, and some decisions become irreversible as soon as execution starts. This motivates dynamic, reactive, and rolling-horizon approaches for terminal scheduling problems [2], [11], [12]. Triggering policies can be classified into periodic, event-driven, and hybrid approaches [13]. In highly evolving environments, event-driven approaches are particularly relevant [13], while rolling-horizon methods repeatedly solve the problem over an updated horizon [11]. In this paper, we study a dynamic quay crane assignment and scheduling problem formulated as an MILP solved in a rolling-horizon framework with an event-driven rescheduling mechanism. A new solve may be triggered by vessel arrival or departure, and, depending on the strategy considered, crane release. Crane release is explicitly integrated because it changes the current decision state in an immediate and operationally meaningful way: when a resource becomes available again, a new assignment and sequencing decisions may become feasible. At the same time, the literature suggests that overly frequent schedule updates may reduce efficiency and induce additional changes in subsequent terminal operations [2], [11]. The objective is therefore to compare several rescheduling policies and to assess whether the responsiveness gained by crane-release-triggered reoptimization compensates for the cost of more frequent updates.

This paper makes two main contributions. First, it proposes an experimental framework for comparing sixteen rescheduling policies built from different scenarios of task preemption (also called task continuity on the same crane), conditional triggering, and objective structure within a single event-driven rolling-horizon MILP engine. Second, on twelve generated instances completed by a detailed same-instance reading, it highlights the respective roles of the objective structure and of the two rescheduling mechanisms studied here, namely continuity on the same crane and additional reoptimization triggered when a crane is released.

II. STATE OF THE ART

The literature on quay-side operations generally distinguishes several problems, including the *Berth Allocation Problem* (BAP), the *Quay Crane Assignment Problem* (QCAP), the *Quay Crane Scheduling Problem* (QCSP), and their integrated variants [1]–[3]. These decision levels are closely related: crane assignment determines vessel processing time, whereas scheduling feasibility depends directly on the retained assignment choices [4], [5], [9]. Recent reviews confirm that current research increasingly considers integrated models, uncertainty, automation, real-time decision-making, and sustainability-oriented objectives [2], [3], [6]. Recent contributions also extend integrated quay-side planning beyond the classical BAP, QCAP, and QCSP. Yu et al. [12] consider berth allocation and quay crane assignment/scheduling under vessel-arrival uncertainty, tide, berth deviation, and crane interference. Guo et al. [14]

*Project funded by the Agence Française de Développement (AFD) – through the Agence Nationale de la Recherche (ANR), under the Partnerships with African Higher Education program (PEA) / Project 21-PEA2-0007-01.

[1]Essowiyaou Kassankogno, Sid Lamrous, and Marie-Ange Manier are with Université Marie et Louis Pasteur, UTBM, CNRS, institut FEMTO-ST, F-90000 Belfort, France essowiyaou.kassankogno@utt.fr, sid.lamrous@utbm.fr, marie-ange.manier@utbm.fr

[2]Essowiyaou Kassankogno and Yaogan Mensah are with Université de Lomé, laboratoire LAMMA, UL, 01 BP 1515, Lomé, Togo ymensah@univ-lome.tg

integrate berth allocation, quay crane assignment, and yard assignment, while Xiao et al. [15] integrate berth allocation, quay crane assignment, and pilotage scheduling. These works confirm the interest of integrated and uncertainty-aware models, whereas this paper focuses on event-driven rescheduling policies for quay crane assignment and scheduling. Integrated approaches have also gained significant attention in the literature. Boysen, Briskorn and Meisel [7] propose a classification of crane scheduling problems in the presence of interference. Agra and Oliveira [8] investigated a BACASP-type problem allowing for crane reassignment over time while adhering to non-crossing and minimum safety distance constraints. Regarding integrated QCAP-QCSP formulations, Diabat and Theodorou [4], Theodorou and Diabat [5], and Fu, Diabat and Tsai [9] show that integration improves crane utilization compared with sequential approaches. Other formulations include the integrated berth allocation, quay crane assignment and scheduling problem of Abou Kasm, Diabat and Cheng [16], and combined quay crane assignment and quay crane scheduling models with crane inter-vessel movement and non-interference constraints [10]. Despite this, many integrated formulations still rely on a fixed or baseline planning state. In changing environments, several works underline the relevance of reactive or rolling-horizon approaches. Chang, Fang and Fan [11] propose a dynamic rolling strategy for multi-vessel quay crane scheduling triggered by vessel arrival or vessel departure, and they also discuss rescheduling when a job is finished, depending on crane idleness. Rodrigues and Agra [2] classify uncertainty-aware berth and crane approaches into proactive, reactive, and proactive/reactive families. More cautiously, in the literature reviewed in this paper, we did not identify a formulation where quay crane release is isolated and compared as a dedicated rescheduling trigger within a strategy-sensitive rolling-horizon framework. The literature also points out that overly frequent updates may be undesirable. Chang, Fang and Fan [11] note that repeated triggering may reduce efficiency and increase crane travel distances, while Rodrigues and Agra [2] emphasize that frequent readjustments may degrade global terminal efficiency and propagate changes to subsequent operations. Related notions of continuity are mainly addressed through non-preemptive execution and static versus dynamic crane-allocation policies. Some works impose that once a quay crane starts a task, it must complete it before leaving, while others compare static policies, in which cranes remain until workload completion, with dynamic policies, in which cranes may join or leave before final completion [10], [16]. However, in the literature we reviewed, we did not find any approach explicitly based on events that ensures the continuity of an already in progress task at the time of rescheduling. Regarding objective functions, the reviewed literature mainly considers time- or cost-based criteria such as service completion time, operation time at port, waiting time, departure delay, tardiness, recovery cost, or berth utilization [2], [8], [10], [11]. However, we did not identify an explicit comparison between total tardiness and vessel stay time through both mono-objective and lexicographic formulations in an event-driven rolling-horizon quay crane rescheduling framework. Our work lies at the intersection of these strands. We rely on integrated QCASP or BACASP-type formulations with rail-mounted cranes, non-crossing constraints, reassignment decisions, and safety-distance requirements [4], [8]–[10], [16]. We adopt an event-driven rescheduling logic in a rolling-horizon framework [11], [13]. More specifically, we compare several rescheduling policies within a single MILP engine, with particular attention to three dimensions that remain only partially connected in the literature: task continuity on the same crane versus possible preemption and reassignment, optional reoptimization upon crane release, and the choice between tardiness or stay-time based objective structures, including lexicographic variants.

III. EVENT-DRIVEN DYNAMIC RESCHEDULING IN A ROLLING-HORIZON FRAMEWORK

We aim at solving a dynamic integrated Quay Crane Assignment and Scheduling Problem (QCASP). It consists both in assigning quay cranes to vessels whose containers must be unloaded (or loaded), and in scheduling those handling tasks constraints such as time constraints (precedence, time windows) and spatial ones (security distance between cranes). In its dynamic variant, each time an event is identified (vessel arrival or departure and eventually crane release), it triggers the scheduling of unfinished tasks, including tasks in progress and new tasks. Then we define an event-driven dynamic rescheduling in a rolling-horizon framework. It consists in solving a mixed-integer linear programming model described below, at each event time t^{ev}, until all tasks are performed. We do not consider BAP as we consider that a given location is already assigned to each arriving vessel.

A. A MODEL FOR THE QCASP

We consider vessels which dynamically arrive at a port and whose containers must be unloaded within a given time window. In a vessel, the containers are arranged in rows called bays. The unloading/loading of containers in a given bay is called a task. There is at most one task for each bay. We assume that the quay is represented by a one-dimensional line. All quay cranes move on the same rail and must respect a minimum safety distance Δ_S. At time t^{ev}, completed tasks are removed, ongoing tasks remain in the state, and decisions concern only the unfinished part of the system.

Indices and input data.: Index $i \in V$ denotes a vessel, n a task of vessel i, and $j \in G$ a quay crane. For each vessel i, ts_i denotes its arrival time and d_i its due date. For each task (i, n), P_i^n denotes the processing time used in the rebuilt problem and l_i^n its quay position. For each crane j, x_j^0 denotes its position at time t^{ev} and v_j its travel speed. Δ_S denotes the minimum safety distance between cranes. M is a global time bound used to define the rebuilt decision window. Tasks are first separated into engaged tasks T^{eng}, whose current execution is constrained by the inherited state, and non-engaged tasks T^{neng}, whose assignment, start time, and position in the crane sequence are decided by the MILP.

State-dependent inherited data.: For an engaged task $(i,n) \in T^{\mathrm{eng}}$, $g_{i,n}^{\mathrm{eng}}$ denotes the crane currently processing that task. When policy conservation is activated, $g_{i,n}^{\mathrm{prev}}$ denotes the inherited crane assignment. When continuity is activated, $g_{i,n}^{\mathrm{ref}}$ denotes the crane that must be kept for the corresponding task. If an inherited starting time is imposed, it is denoted by $D_{i,n}^{0}$.

Rebuilt-instance sets and parameters.: Using the notation introduced above, the rebuilt instance is specified by the following additional sets and parameters. For each vessel i, $T_i^{\mathrm{neng}} = \{\, n : (i,n) \in T^{\mathrm{neng}} \,\}$ is the set of non-engaged tasks, $T_i^{\mathrm{eng}} = \{\, n : (i,n) \in T^{\mathrm{eng}} \,\}$ is the set of engaged tasks, and $T_i = T_i^{\mathrm{neng}} \cup T_i^{\mathrm{eng}}$ is the set of unfinished tasks. The set $\phi_i \subseteq G$ contains the cranes admissible for the tasks of vessel i. For each crane $j \in G$, $\mathcal{T}_j$ is the set of unfinished tasks that crane j may process, and $\mathcal{T}_j^{\mathrm{neng}} \subseteq \mathcal{T}_j$ is the set of non-engaged tasks that crane j may process. The crane set G is partitioned into G^{free}, G^{mov}, and G^{eng}, which respectively contain the cranes free at t^{ev}, already assigned but not yet in position, and currently processing a task. For each crane $j \in G^{\mathrm{eng}}$, $(i_j, n_j) \in T^{\mathrm{eng}}$ denotes the task currently processed by crane j. The parameter a_j is the reference position of crane j: its current position if $j \in G^{\mathrm{free}} \cup G^{\mathrm{mov}}$, and the position of its current task if $j \in G^{\mathrm{eng}}$. The parameter r_j is the release time of crane j: the current time if $j \in G^{\mathrm{free}} \cup G^{\mathrm{mov}}$, and the completion time of its current task if $j \in G^{\mathrm{eng}}$. For each task $(i,n) \in T^{\mathrm{neng}}$, $\underline{D}_i^n$ is the lower bound of its rebuilt start-time window, and $\overline{D}$ is the global upper bound of this window. Finally, $j \prec j'$ means that crane j lies to the left of crane j' on the rail, and A^{noneng} contains the pairs of adjacent cranes in $G^{\mathrm{free}} \cup G^{\mathrm{mov}}$.

Rebuilt MILP.: Given this rebuilt instance, the rebuilt optimization problem is the following MILP.

Decision variables.: For all admissible vessels, tasks, and cranes, the main variables are:

$$\begin{aligned} &D_i^n \in \mathbb{R}_+, \qquad te_i \in \mathbb{R}_+, \qquad L_i \in \mathbb{R}_+, \\ &Y_{ij}^n \in \{0,1\}, \qquad Z_{i,n,i',n',j,j'} \in \{0,1\}. \end{aligned} \tag{1}$$

Here, D_i^n is the starting time of task (i,n), te_i is the completion time of all tasks of vessel i, L_i is the tardiness of vessel i, and $Y_{ij}^n = 1$ if crane j is assigned to task (i,n). The binary variable $Z_{i,n,i',n',j,j'}$ is used for ordering disjunctions between distinct tasks $(i,n) \neq (i',n')$ and admissible cranes $j \in \phi_i$, $j' \in \phi_{i'}$.

For each crane $j \in G^{\mathrm{free}} \cup G^{\mathrm{mov}}$, the first-movement variables are:

$$\begin{aligned} &U_j \in \{0,1\}, \qquad X_j \in \mathbb{R}, \qquad T_j \in \mathbb{R}_+, \qquad S_j \in \mathbb{R}, \\ &F_{ij}^n \in \{0,1\}, \qquad \forall (i,n) \in \mathcal{T}_j^{\mathrm{neng}}. \end{aligned} \tag{2}$$

Here, $U_j = 1$ if crane j is assigned to at least one task in $\mathcal{T}_j^{\mathrm{neng}}$, $F_{ij}^n = 1$ if (i,n) is its first assigned task, X_j is the position of that first task, S_j is the movement starting time, and T_j is the starting time of the first task.

Objective functions.: Depending on the selected mode, the rebuilt problem uses:

$$\begin{aligned} D=1: &\quad \operatorname{lex\,min}\left(\textstyle\sum_i L_i,\ \sum_i (te_i - ts_i)\right), \\ D=2: &\quad \min \textstyle\sum_i L_i, \\ D=3: &\quad \min \textstyle\sum_i (te_i - ts_i), \\ D=4: &\quad \operatorname{lex\,min}\left(\textstyle\sum_i (te_i - ts_i),\ \sum_i L_i\right). \end{aligned} \tag{3}$$

Core constraints.:

Bounds and fixed decisions.: For each non-engaged task and each engaged task, the rebuilt start times satisfy:

$$\begin{aligned} &\underline{D}_i^n \le D_i^n \le \overline{D}, \quad \forall (i,n) \in T^{\mathrm{neng}}, \\ &D_i^n = D_{i,n}^0, \qquad \forall (i,n) \in T^{\mathrm{eng}}. \end{aligned} \tag{4}$$

For each engaged task, the inherited crane assignment is fixed:

$$\begin{aligned} &Y_{i,g_{i,n}^{\mathrm{eng}}}^n = 1, \\ &Y_{ij}^n = 0, \qquad \forall j \in \phi_i \setminus \{g_{i,n}^{\mathrm{eng}}\}. \end{aligned} \tag{5}$$

When policy conservation is enforced, the inherited crane assignment is maintained:

$$\begin{aligned} &Y_{i,g_{i,n}^{\mathrm{prev}}}^n = 1, \\ &Y_{ij}^n = 0, \qquad \forall j \in \phi_i \setminus \{g_{i,n}^{\mathrm{prev}}\}. \end{aligned} \tag{6}$$

When continuity is enforced, the reference crane assignment is maintained:

$$\begin{aligned} &Y_{i,g_{i,n}^{\mathrm{ref}}}^n = 1, \\ &Y_{ij}^n = 0, \qquad \forall j \in \phi_i \setminus \{g_{i,n}^{\mathrm{ref}}\}. \end{aligned} \tag{7}$$

Unique assignment, completion time, and lateness.: Each non-engaged task is assigned to exactly one admissible crane, and vessel completion and lateness are defined by:

$$\begin{aligned} &\sum_{j \in \phi_i} Y_{ij}^n = 1, && \forall (i,n) \in T^{\mathrm{neng}}, \\ &L_i \ge 0, \qquad L_i \ge te_i - d_i, && \forall i \in V, \\ &te_i \ge D_i^n + P_i^n, && \forall i \in V,\ \forall n \in T_i. \end{aligned} \tag{8}$$

Release constraints.: For each $(i,n) \in T^{\mathrm{neng}}$ and $j \in \phi_i$, the start time must be compatible with crane release and travel:

$$D_i^n \ge r_j + |a_j - l_i^n|/v_j - M(1 - Y_{ij}^n). \tag{9}$$

For each engaged task $(i',n') \in T^{\mathrm{eng}}$ processed by crane $j = g_{i',n'}^{\mathrm{eng}}$, and each $(i,n) \in \mathcal{T}_j^{\mathrm{neng}}$, we impose:

$$\begin{aligned} D_i^n \ge D_{i',n'}^0 + P_{i'}^{n'} + |l_{i'}^{n'} - l_i^n|/v_j + \varepsilon \\ - M(1 - Y_{ij}^n). \end{aligned} \tag{10}$$

Here, ε is a small constant.

Sequencing on the same crane.: For each crane $j \in G$ and each pair of distinct tasks $(i,n),(i',n') \in \mathcal{T}_j$, the two possible orders are modeled by:

$$\begin{aligned} D_{i'}^{n'} \ge\ & D_i^n + P_i^n + |l_i^n - l_{i'}^{n'}|/v_j + \varepsilon \\ & - M(1 - Z_{i,n,i',n',j,j}) - M(2 - Y_{ij}^n - Y_{i'j}^{n'}), \\ D_i^n \ge\ & D_{i'}^{n'} + P_{i'}^{n'} + |l_{i'}^{n'} - l_i^n|/v_j + \varepsilon \\ & - M Z_{i,n,i',n',j,j} - M(2 - Y_{ij}^n - Y_{i'j}^{n'}). \end{aligned} \tag{11}$$

Safety-distance disjunction.: For each pair of distinct tasks $(i,n) \neq (i',n')$, and each pair of cranes $j \in \phi_i$, $j' \in \phi_{i'}$ such that $j \neq j'$ and $|l_i^n - l_{i'}^{n'}| < \Delta_S$, define:

$$\begin{aligned}\delta_{i,n,i',n',j,j'} = \max\{&(\Delta_S - |l_i^n - l_{i'}^{n'}|)/v_j,\\ &(\Delta_S - |l_i^n - l_{i'}^{n'}|)/v_{j'}\}.\end{aligned} \tag{12}$$

The corresponding safety-distance disjunction is:

$$\begin{aligned}D_{i'}^{n'} \geq\ & D_i^n + P_i^n + \delta_{i,n,i',n',j,j'} + \varepsilon\\ &- M(1 - Z_{i,n,i',n',j,j'}) - M(2 - Y_{ij}^n - Y_{i'j'}^{n'}),\\ D_i^n \geq\ & D_{i'}^{n'} + P_{i'}^{n'} + \delta_{i,n,i',n',j,j'} + \varepsilon\\ &- MZ_{i,n,i',n',j,j'} - M(2 - Y_{ij}^n - Y_{i'j'}^{n'}).\end{aligned} \tag{13}$$

First-movement constraints.: For each crane $j \in G^{\text{free}} \cup G^{\text{mov}}$, the first assigned task is identified by:

$$\begin{aligned}&F_{ij}^n \leq Y_{ij}^n, && \forall (i,n) \in \mathcal{T}_j^{\text{neng}},\\ &\sum_{(i,n)\in\mathcal{T}_j^{\text{neng}}} F_{ij}^n = U_j,\\ &U_j \geq Y_{ij}^n, && \forall (i,n) \in \mathcal{T}_j^{\text{neng}},\\ &U_j \leq \sum_{(i,n)\in\mathcal{T}_j^{\text{neng}}} Y_{ij}^n.\end{aligned} \tag{14}$$

For each $(i,n) \in \mathcal{T}_j^{\text{neng}}$ and each $(i',n') \in \mathcal{T}_j^{\text{neng}} \setminus \{(i,n)\}$, if (i,n) is the first assigned task of crane j, no other task assigned to j can start earlier:

$$D_{i'}^{n'} \geq D_i^n - M(1 - F_{ij}^n) - M(1 - Y_{i'j}^{n'}). \tag{15}$$

For each $j \in G^{\text{free}} \cup G^{\text{mov}}$ and $(i,n) \in \mathcal{T}_j^{\text{neng}}$, the position and timing of the first assigned task are imposed by:

$$\begin{aligned}&X_j - l_i^n \leq M(1 - F_{ij}^n), \qquad l_i^n - X_j \leq M(1 - F_{ij}^n),\\ &T_j - D_i^n \leq M(1 - F_{ij}^n), \qquad D_i^n - T_j \leq M(1 - F_{ij}^n),\\ &S_j - \big(D_i^n - |a_j - l_i^n|/v_j\big) \leq M(1 - F_{ij}^n),\\ &\big(D_i^n - |a_j - l_i^n|/v_j\big) - S_j \leq M(1 - F_{ij}^n),\\ &X_j - a_j \leq MU_j, \qquad a_j - X_j \leq MU_j.\end{aligned} \tag{16}$$

Define

$$\widetilde{X}_j = \begin{cases} X_j, & \text{if } j \in G^{\text{free}} \cup G^{\text{mov}},\\ a_j, & \text{if } j \in G^{\text{eng}}.\end{cases} \tag{17}$$

The initial order of cranes on the rail and the adjacent first movements are enforced by:

$$\begin{aligned}&\widetilde{X}_j + \Delta_S \leq \widetilde{X}_{j'}, \qquad \forall j,j' \in G,\ j \prec j',\\ &S_{j'} \geq T_j + \varepsilon - M(1 - Z_{j,j'}) - M(2 - U_j - U_{j'}),\\ &S_j \geq T_{j'} + \varepsilon - MZ_{j,j'} - M(2 - U_j - U_{j'}),\\ &\forall (j,j') \in A^{\text{noneng}}.\end{aligned} \tag{18}$$

where $Z_{j,j'}$ is equal to 1 if crane j starts its first task before crane j' starts its first task.

IV. EXPERIMENTAL SETUP AND OBSERVED INDICATORS

The experimental analysis is based on 12 generated instances. These instances contain between 2 and 5 vessels and between 4 and 6 available quay cranes. Depending on the instance, each vessel is associated with between 2 and 5 tasks. For each vessel, the data specify an arrival time, a due date, and task processing times. The instances also provide task positions, initial crane positions, admissible crane sets, and vessel spans. In all instances, crane speed is fixed to 1.0. The test set was generated specifically for the present study. It therefore does not directly correspond to an industrial case study. It is used to compare the behaviour of the tested rescheduling policies under controlled configurations. In the reported campaign, all 192 runs execute the full theoretical workload. The largest solved case in this study is instance 12, with 5 vessels, 6 cranes and 20 tasks. All experiments were carried out on an Intel Core i7-13700H machine with 16 GB of RAM. The models were implemented in Python with PuLP and solved with CBC MILP Solver 2.10.3 through `PULP_CBC_CMD`. No explicit time limit, MIP gap, or thread parameter was imposed. Since all runs use the same MILP formulation, rolling-horizon mechanism, and solver interface, the comparison focuses on the effects of the rescheduling policy parameters B, C, and D, rather than on differences between algorithms.

The experimental protocol relies on an event-driven simulation with successive rescheduling steps. For each *instance–strategy* pair, execution is progressively constructed through several optimizations run until all tasks are completed. At each detected event, the system state is updated and, according to the considered strategy, a new MILP is rebuilt and solved on the unfinished part of the system.

For each instance, the 16 strategies combining the parameters (B,C,D) were evaluated, which results in 192 runs. Parameter B controls task continuity on the same crane, with $B0$=False if a task in progress at t^{ev} can be preempted and reassigned to another crane, and $B1$=True if preemption is not allowed. For parameter C, $C1$=True allows rescheduling upon crane release, contrary to $C0$=False. Parameter D denotes the selected objective structure, with $D \in \{1,2,3,4\}$. The performance of the strategies evaluated through three operational KPIs (total tardiness L, total stay time T_s, and makespan $C_{\max}$), together with CPU time for a given run as the main computational indicator. The detailed same instance reading on instance 8 complements this global view by comparing pairs of strategies that differ only by B or only by C, through vessel-level KPIs and crane schedules.

V. EXPERIMENTAL RESULTS

Objective structure D: Table I summarizes the mean effect of the 4 objective structures over all B,C combinations and the 12 instances. For aggregated averages, D=1 delivers the best results for L, T_s and $C_{\max}$, while D=2 yields the lowest average CPU time. At the aggregated level, the two lexicographic approaches (D=1 and D=4) outperform the single-objective solutions on these criteria, while being more

TABLE I

AGGREGATION BY OBJECTIVE STRUCTURE D OVER THE 192 RUNS.

D	Mean L	Mean T_s	Mean $C_{\max}$	Mean CPU (s)
1	24.06	87.23	55.56	159.59
2	25.25	89.48	57.21	33.56
3	26.25	89.48	58.75	48.42
4	24.98	88.29	57.06	60.64

TABLE II

MEAN RESULTS FOR THE 16 STRATEGIES OVER THE 12 INSTANCES.

Strategy	Mean L	Mean T_s	Mean $C_{\max}$	Mean CPU (s)
B0C0D1	23.83	87.00	55.50	274.92
B0C1D1	24.25	87.92	55.92	299.12
B1C0D1	24.17	87.08	55.42	25.56
B1C1D1	24.00	86.92	55.42	38.74
B0C0D2	25.58	89.75	57.75	30.30
B0C1D2	25.75	90.33	58.92	72.67
B1C0D2	24.83	88.92	56.08	12.78
B1C1D2	24.83	88.92	56.08	18.51
B0C0D3	25.75	88.92	59.50	82.69
B0C1D3	25.08	88.50	58.83	94.28
B1C0D3	27.08	90.25	58.33	8.22
B1C1D3	27.08	90.25	58.33	8.51
B0C0D4	25.08	88.58	58.42	81.00
B0C1D4	24.67	88.42	57.83	124.56
B1C0D4	25.08	88.08	56.00	18.78
B1C1D4	25.08	88.08	56.00	18.24

time-consuming, especially D=1 which is on average about 3 to 5 times longer to solve than the single-objective modes.

Evaluation of the 16 strategies: Table II reports the 16 strategies on L, T_s, $C_{\max}$, and CPU time, and provides the global synthesis over the 12 instances. Table III reports the same indicators for reference instance 8. Tables IV and V then isolate the effects of B and C, respectively.

Table II shows that, for D=1, (B=1, C=1) gives the best mean stay time T_s and makespan $C_{\max}$, whereas (B=0, C=0) gives the smallest mean tardiness L. It means that the best results may be obtained when integrating either the crane release as a triggering event, or the task preemption possibility. Nevertheless the results are not the same depending on the chosen mode D. They also depend on the characteristics of the instance solved. For example, with Table III related to instance 8, for any objective structure except D=2, adding only the rescheduling strategy based on crane release (C=0 changed into C=1) does not change the criteria values, whereas allowing task preemption and reassignment (from B=1 to B=0) may change the solution obtained (in a better way 5 times out of 8 for the tardiness). The interest of rescheduling upon crane release and/or allowing task preemption clearly appears on some instances in a rather contextual way. This confirms that the purpose of the comparison is not to select a universally dominant strategy, but to identify when and how each mechanism changes the behaviour of the schedule, as shown in in Tables IV and V.

Table IV compares two strategies for instance 8 with only one parameter changed: $B0$ allows preemption and

TABLE III

KPI SYNTHESIS FOR INSTANCE 8.

Strategy	L	T_s	$C_{\max}$	CPU time (s)
B0C0D1	26	101	51	2675.82
B0C1D1	26	101	51	2765.48
B1C0D1	28	100	51	8.62
B1C1D1	28	100	51	9.36
B0C0D2	38	113	53	73.34
B0C1D2	46	120	69	104.76
B1C0D2	38	113	53	14.96
B1C1D2	38	113	53	14.71
B0C0D3	33	108	67	386.75
B0C1D3	33	108	67	442.13
B1C0D3	38	113	53	21.94
B1C1D3	38	113	53	23.78
B0C0D4	33	108	67	437.28
B0C1D4	33	108	67	458.63
B1C0D4	28	100	51	18.46
B1C1D4	28	100	51	17.58

TABLE IV

EFFECT OF CHANGING ONLY B FOR INSTANCE 8 ($C = 0$, $D = 1$).

	B0C0D1	B1C0D1
L	26	28
T_s	101	100
$C_{\max}$	51	51
CPU (s)	2675.82	8.62
$V1$**: Stay/tardiness**	11 / 0	11 / 0
$V2$**: Stay/tardiness**	32 / 1	28 / 0
$V3$**: Stay/tardiness**	31 / 10	34 / 13
$V4$**: Stay/tardiness**	27 / 15	27 / 15
Crane schedules, format: task [start time, end time]. Bold entries denote all segments of preempted tasks.		
QC1	• V1.T1 [2,8] • V2.T1 [33,40]	• V1.T1 [2,8]
QC2	• V1.T2 [4,11] • **V2.T3_R1** [25,27] • V2.T2 [31,39]	• V1.T2 [4,11] • V2.T1 [17,24] • V2.T2 [28,36]
QC3	• **V2.T3** [10,16] • V2.T4 [20,29] • **V2.T5_R1** [33,40] • **V3.T1_R1** [44,46]	• V2.T3 [10,18] • V2.T4 [22,31] • V3.T1 [39,45]
QC4	• **V2.T5** [14,16] • **V3.T1** [20,24] • V3.T2 [28,35] • V3.T3 [39,47]	• V2.T5 [14,23] • V3.T2 [31,38] • V3.T3 [42,50]
QC5	• V3.T4 [18,26] • V4.T1 [32,39] • V4.T2 [43,51]	• V3.T4 [18,26] • V4.T1 [32,39] • V4.T2 [43,51]

reassignment of an interrupted task, whereas $B1$ enforces task continuity on the same crane. With B0C0D1, we can observe that the makespan remains unchanged, with a small increase of the total stay time (1%), while the total tardiness decreases by about 7.14%. This gain of 2 t.u. on tardiness results from a 1 t.u. increase for vessel $V2$ with two preempted tasks, and a 3 t.u. decrease for vessel $V3$, with one preempted task. For example task $T3$ on vessel $V2$ (labelled $V2.T3$) is assigned to crane $QC3$ and carried out without interruption with B1C0D1, whereas it is first carried out in part by $QC3$ ($V2.T3$) then completed later

TABLE V

EFFECT OF CHANGING ONLY C FOR INSTANCE 6 ($B = 0$, $D = 2$).

	B0C0D2	B0C1D2
L	17	10
T_s	75	70
$C_{\max}$	48	48
CPU (s)	41.50	27.50
$V1$**: Stay/tardiness**	17 / 0	19 / 0
$V2$**: Stay/tardiness**	28 / 8	21 / 1
$V3$**: Stay/tardiness**	30 / 9	30 / 9
Crane schedules, format: task [start time, end time]. Bold entries denote all segments of preempted tasks.		
QC1	• **V1.T2** [5,9] • V1.T1 [12,17]	• V1.T2 [5,11] • V1.T1 [14,19]
QC2	• V1.T3 [2,8] • **V1.T2_R1** [15,17]	• V1.T3 [2,8] • V1.T4 [11,18] • **V2.T1_R1** [28,30]
QC3	• V1.T4 [9,16] • V2.T1 [23,28] • V2.T2 [31,37]	• **V2.T1** [15,18] • V2.T2 [21,27]
QC4	• **V2.T4** [12,17] • V2.T3 [22,28] • **V2.T4_R1** [31,34] • V3.T1 [41,46]	• V2.T4 [12,20] • V2.T3 [23,29] • V3.T1 [39,44]
QC5	• V3.T4 [21,29] • V3.T3 [32,39] • V3.T2 [42,48]	• V3.T3 [18,25] • V3.T2 [28,34] • V3.T4 [40,48]

by $QC2$ ($V2.T3\_R1$) in the B0C0D1 strategy. This strategy modifies the assignment and scheduling of tasks for almost all cranes except $QC5$. Table V compares two strategies for instance 6, that differ only by C, with B=0 (task preemption allowed) and D=2 fixed. Strategy B0C1D2 allows rescheduling when a crane is released, whereas B0C0D2 does not. While the makespan remains unchanged, B0C1D2 reduces total tardiness by 41.18% and total stay time by 6.67%. CPU time also decreases. The gain is mainly associated with vessel $V2$. As in Table IV, the change in strategy modifies task assignments, time intervals, and sequences on several cranes. Preempted tasks appear in both schedules. This example shows that crane-release reoptimization can be useful in some configurations, even if it is not the case for all instances.

VI. CONCLUSION

This paper proposed a MILP to model a dynamic 2-objectives Quay Crane Assignment and Scheduling Problem. Then 16 rescheduling policies were evaluated on 12 generated instances that we solved in a rolling-horizon framework. The analysis on various instances shows that changing the continuity policy or/and reoptimization upon crane release may substantially reshape both vessel departures and crane schedules. Nevertheless the improvement due to these strategies strongly depends on the configuration of the instance which includes its size but also the simultaneous availability of resources within the horizon and the structural characteristics of the instance. Finally the global results also indicate that the way to consider the objective functions impacts on the solutions. This will guide our future research towards the development of approximate methods to solve this multi-objective problem, in which we could also add new criteria, like minimizing the distance travelled by each crane.

REFERENCES

[1] C. Bierwirth and F. Meisel, "A follow-up survey of berth allocation and quay crane scheduling problems in container terminals," *European Journal of Operational Research*, vol. 244, no. 3, pp. 675–689, 2015, doi: 10.1016/j.ejor.2014.12.030.

[2] F. Rodrigues and A. Agra, "Berth allocation and quay crane assignment/scheduling problem under uncertainty: A survey," *European Journal of Operational Research*, vol. 303, no. 2, pp. 501–524, 2022, doi: 10.1016/j.ejor.2021.12.040.

[3] N. Makhado, T. Paepae, M. Sejeso, and C. Harley, "Berth allocation and quay crane scheduling in port operations: A systematic review," *Journal of Marine Science and Engineering*, vol. 13, no. 7, Art. no. 1339, 2025, doi: 10.3390/jmse13071339.

[4] A. Diabat and E. Theodorou, "An integrated quay crane assignment and scheduling problem," *Computers & Industrial Engineering*, vol. 73, pp. 115–123, 2014, doi: 10.1016/j.cie.2013.12.012.

[5] E. Theodorou and A. Diabat, "A joint quay crane assignment and scheduling problem: Formulation, solution algorithm and computational results," *Optimization Letters*, vol. 9, no. 4, pp. 799–817, 2015, doi: 10.1007/s11590-014-0787-x.

[6] B. Dragović, A. Dragović, and M. A. Dulebenets, "The quay crane operation problem at marine container terminals: Bibliometric analysis, emerging trends, and future research opportunities," *Transportation Research Part E: Logistics and Transportation Review*, vol. 201, Art. no. 104266, 2025, doi: 10.1016/j.tre.2025.104266.

[7] N. Boysen, D. Briskorn, and F. Meisel, "A generalized classification scheme for crane scheduling with interference," *European Journal of Operational Research*, vol. 258, no. 2, pp. 343–357, 2017, doi: 10.1016/j.ejor.2016.08.041.

[8] A. Agra and M. Oliveira, "MIP approaches for the integrated berth allocation and quay crane assignment and scheduling problem," *European Journal of Operational Research*, vol. 264, no. 1, pp. 138–148, 2018, doi: 10.1016/j.ejor.2017.05.040.

[9] Y.-M. Fu, A. Diabat, and I.-T. Tsai, "A multi-vessel quay crane assignment and scheduling problem: Formulation and heuristic solution approach," *Expert Systems with Applications*, vol. 41, no. 16, pp. 6959–6965, 2014, doi: 10.1016/j.eswa.2014.05.002.

[10] G. Alsoufi, X. Yang, and A. Salhi, "Combined quay crane assignment and quay crane scheduling with crane inter-vessel movement and non-interference constraints," *Journal of the Operational Research Society*, vol. 69, no. 3, pp. 372–383, 2018, doi: 10.1057/s41274-017-0226-3.

[11] D. Chang, T. Fang, and Y. Fan, "Dynamic rolling strategy for multi-vessel quay crane scheduling," *Advanced Engineering Informatics*, vol. 34, pp. 60–69, 2017, doi: 10.1016/j.aei.2017.09.001.

[12] M. Yu, X. Liu, X. Ji, Y. Ren, and W. Guo, "Integrated berth allocation and quay crane assignment and scheduling problem under the influence of various factors," *IET Collaborative Intelligent Manufacturing*, vol. 6, no. 4, Art. no. e70001, 2024, doi: 10.1049/cim2.70001.

[13] A. Hamzadayi and G. Yildiz, "Event driven strategy based complete rescheduling approaches for dynamic m identical parallel machines scheduling problem with a common server," *Computers & Industrial Engineering*, vol. 91, pp. 66–84, 2016, doi: 10.1016/j.cie.2015.11.005.

[14] L. Guo, J.-F. Zheng, J. Du, Z. Gao, and K. Fagerholt, "Integrated planning of berth allocation, quay crane assignment and yard assignment in multiple cooperative terminals," *Transportation Research Part E: Logistics and Transportation Review*, vol. 183, Art. no. 103456, 2024, doi: 10.1016/j.tre.2024.103456.

[15] L. Xiao, G. Laporte, P. Sun, and W. Xie, "The integrated planning of berth allocation, quay crane assignment, and pilotage scheduling," *Computers & Operations Research*, vol. 184, Art. no. 107201, 2025, doi: 10.1016/j.cor.2025.107201.

[16] O. Abou Kasm, A. Diabat, and T. C. E. Cheng, "The integrated berth allocation, quay crane assignment and scheduling problem: Mathematical formulations and a case study," *Annals of Operations Research*, vol. 291, pp. 435–461, 2020, doi: 10.1007/s10479-018-3125-3.