The NP-completeness of quay crane scheduling problem

Ali Skaf, Samir Dawaliby, and Arezki Aberkane

Caplogy R&D, Paris, France

Abstract. This paper discusses the computational complexity of the quay crane scheduling problem (QCSP) in a maritime port. To prove that a problem is NP-complete, there should be no polynomial time algorithm for the exact solution, and only heuristic approaches are used to obtain near-optimal solutions but in reasonable time complexity. To address this, first we formulate the QCSP as a mixed integer linear programming to solve it to optimal, and next we theoretically prove that the examined problem is NP-complete.

Keywords: Quay crane, Container, Scheduling, Optimization, MILP, NP-complete.

1 Introduction

In any maritime port, several machines exist to unload/load containers, and quay cranes are allowed to unload containers from the vessels or load containers into the vessels. Each vessel is divided into several bays, and each bay contains several containers. (Fig. 1)

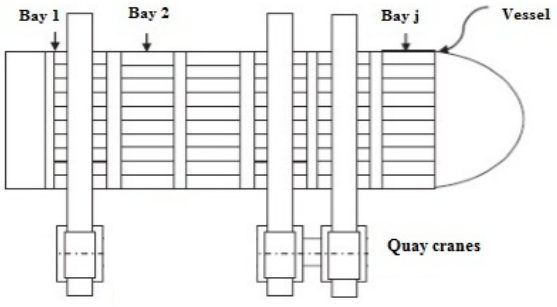

Fig. 1. Vessel-Quay cranes-Bays-Containers

In this paper, we consider the quay crane scheduling problem (QCSP) in the port of Tripoli-Lebanon. First, we present a novel mathematical Mixed Integer Linear Programming (MILP) model, and next we prove that the QCSP is NP-complete. A problem is NP-complete (Fig. 2) when it can be solved by a restraint class of

David C. Wyld et al. (Eds): AI, AIMLNET, BIOS, BINLP, CSTY, MaVaS, SIGI - 2022
pp. 17-27, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.121802

search algorithms and can be used to simulate any other problem with a similar algorithm. More precisely, each entry in the problem must be associated with a set of solutions of polynomial length, in which the validity can be tested quickly (in polynomial time for example) as the output for any input is "yes" if the solution set is not empty and "no" if it is empty. The complexity class of problems of this form is called NP, which stands for "non-deterministic polynomial time".
The theory of NP-completeness is designed by [1] HR Lewis (1983).
A problem is NP-complete if:

- it is in NP
- it is NP-hard

For several problems that have only NP solutions, there exist polynomial time algorithms that produce good approximations of the optimal solutions.
For a given NP problem, if there is a correct answer in polynomial time, then there is a correct answer in polynomial time for all the full NP problems.

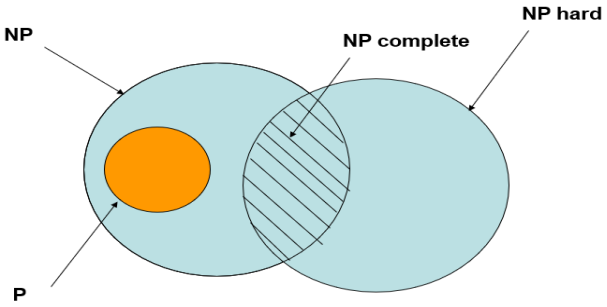

Fig. 2. NP-complete

Formally, we define the notion of polynomial reduction as follows: let A and B be two problems, with the aim of using the solution of A to solve B. (Fig. 3)

- A reduction of A to B is a polynomial time algorithm transforming any instance of B into an instance of A.
- Thus, if we have an algorithm to solve A, we also know how to solve B.
- A is therefore at least as difficult to solve as B.
- B is at most as difficult to solve as A.

  We also write $B \leq A$.

The remainder of this paper is organized as follows: In section 2, we present a

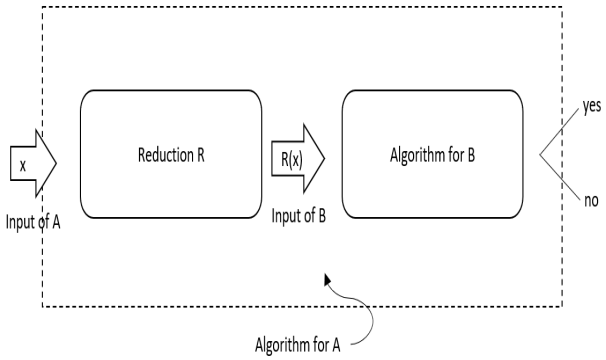

Fig. 3. Polynomial

literature review for the NP-complete problems, then in section 3, we describe in detail the new proposed mathematical model formulation for the QCSP problem. Next, we provide the NP-Completeness proof and discussion in section 4. Finally, some concluding remarks are made in section 5.

2 Literature review

[2] Plesn et al. (1979) proved that he Hamiltonian cycle problem is NP- complete even in the case of planar digraphs with out-degrees at most 2. Clearly, the bound two on degrees is the best possible. A Hamiltonian cycle in a graph or digraph is a cycle containing all the points. Thus any such cycle has p points as well asp lines (arcs) if the graph (digraph) has p points. No elegant characterization of the graphs or digraphs which possess Hamiltonian cycles exists, although the problem is at least one hundred years old. As the problem is a special case of the famous traveling salesman problem, it is interesting also from the computational viewpoint.
[3] Tovey (1984) proved that 3-SAT is NP-complete when restricted to instances where each variable appears in at most four clauses. When no variable appears in more than three clauses, 3-SAT is trivial and SAT is NP-complete. When no variable appears in more than two clauses, SAT may be solved in linear time.
[4] Ouyang et al. (1997) represent further evidence for the ability of DNA computing to solve NP-complete search problems. A pool of DNA molecules corresponding to the total ensemble of six-vertex cliques was built, followed by a series of selection processes. The algorithm is highly parallel and has satisfactory fidelity. This work represents further evidence for the ability of DNA computing to solve NP-complete search problems.
[5] Kaye (2000) demonstrates that a puzzle based on the Minesweeper game is NP-complete makes this important computer science topic accessible to high school students. The resource described here is a set of slides showing the detailed solution

of two introductory puzzles, following by the step-by-step simulation of digital circuit elements required for proving NP-completeness.
[6] Liu et al. (2002) show further evidence for the ability of DNA computing to solve NP-Complete problems. The graph-theoretic parameter that has probably received the most attention over the years is the chromatic number. As is well-known, the coloring problem is an NP-Complete problem. In their study, it has been solved by means of molecular biology techniques. The algorithm is highly parallel and has satisfactory fidelity. This work shows further evidence for the ability of DNA computing to solve NP-Complete problems.
[7] Kellerer et al. (2004) present an introduction to NP-Completeness of Knapsack Problems. The reader may have noticed that for all the considered variants of the knapsack problem, no polynomial time algorithm have been presented which solves the problem to optimality. Indeed all the algorithms described are based on some kind of search and prune methods, which in the worst case may take exponential time. It would be a satisfying result if we somehow could prove it is not possible to find an algorithm which runs in polynomial time, somehow having evidence that the presented methods are "as good as we can do". However, no proof has been found showing that the considered variants of the knapsack problem cannot be solved to optimality in polynomial time.
[8] Bresar et al. (2011) prove that the problem of determining is NP-complete for every path ≥ 2. For path =2 this equals to the vertex cover problem (shortly VCP) which is known to be NP-complete.
The importance of the TSP presented by [9] Hoffman et al. (2013) is that it is representative of a larger class of problems known as combinatorial optimization problems. The TSP problem belongs in the class of such problems known as NP-complete. Specifically, if one can find an efficient (i.e., polynomial-time) algorithm for the traveling salesman problem, then efficient algorithms could be found for all other problems in the NP-complete class.
In our previous papers, [11] Skaf et al. (2018) suggested two methods to solve the quay crane scheduling problem at port of Tripoli-Lebanon, to minimize the total completion time for all containers from the vessel to the storage location. Later [12] Skaf et al. (2019) proposed for the same problem a genetic algorithm to obtain near-optimal solutions in an acceptable CPU time. After that, [13] Skaf et al. (2021) detailed the model with results comparison with literature and benchmark.
In this study, we present a mixed-integer linear programming model to solve the quay crane scheduling problem and a proof that this problem is NP-complete.

3 Mathematical formulation

In this section, we present the mathematical formulation of the QCSP with the different variables, parameters and the mathematical model.

3.1 Assumptions

We consider in this study a single vessel to transport containers. It is divided into several bays which in turn contains several containers. We also consider multi-quay cranes to unload/load container from/to the vessel with interference constraints between them (each bay is handled by at most one quay crane at a time). Each quay crane can unload/load one container at a time and the most important constraint here is that a quay crane should complete its unloading/loading in a current bay before passing to another one. Finally, we ignore the travel time between bays, and we do not consider any idle quay crane in this study.

3.2 Data

The notation that was used in the proposed formulation of the problem is shown below:

- Q: set of quay cranes
- $|Q|$: number of quay cranes
- i : index of quay cranes i ($\forall i \in Q$)
- B: set of bays
- $|B|$: number of bays
- j, j' : index of bays j and j' ($\forall j, j' \in B$)
- C_j: number of containers in bay j.
- Tc: time required for a quay crane to unload a container and place it in the storage location. It is not the same for all containers in the different bays
- M: large integer number

3.3 Decision variables

The decision variables that was used in the proposed formulation of the problem is shown below:

- $x_{(j,i)} = \begin{cases} 1 & \text{if quay crane i unload container from bay j} \\ 0 & \text{otherwise} \end{cases}$

- $z_{(j,j')} = \begin{cases} 1 & \text{if the containers unloading in bay j finishes before the starting} \\ & \text{of the containers unloading in bay j'} \\ 0 & \text{otherwise} \end{cases}$

- t_j : completion time of bay j

- C_{max} : makespan

3.4 Mixed integer linear programming

The mixed integer linear formulation (MILP) was founded by [10] Leonid Kantorovich (1939). It is a technique for optimizing an objective function, subject to linear inequality and equality constraints.

The MILP we propose is as follows:

Objective

$$Minimize \quad C_{max} \tag{1}$$

Equation (1) is the objective function : minimize the makespan (completion time of all bays).

Subject to

$$\sum_{i=1}^{|Q|} x_{(j,i)} = 1 \quad \forall j \in B \tag{2}$$

Constraint (2) : each bay must be handled only by one quay crane.

$$t_j \geq (Tc \cdot C_j) \quad \forall j \in B \tag{3}$$

Constraint (3) : the completion time is bigger than the working time in each bay (working time = number of containers $\times$ time needed to unload a container by a quay crane and store it into the storage location).

$$t_j - t_{j'} + (Tc \cdot C_{j'}) + z_{(j,j')} \cdot M > 0 \quad \forall j, j' \in B \tag{4}$$

Constraint (4) : if $z_{(j,j')} = 1$, then bay j finishes before bay j' starts.

$$t_j - t_{j'} + (Tc \cdot C_{j'}) - (1 - z_{(j,j')}) \cdot M \leq 0 \quad \forall j, j' \in B \tag{5}$$

Constraint (5) : if $z_{(j,j')} = 0$, bay j finishes after bay j' starts.

$$\sum_{i=1}^{|Q|} i \cdot x_{(j,i)} + 1 \leq \sum_{i'=1}^{|Q|} i' \cdot x_{(j',i')} + (z_{(j,j')} + z_{(j',j)}) \cdot M$$

$$\forall j, j' \in B, j < j' \tag{6}$$

Constraint (6) avoid the interference between the quay cranes. If bays j and j' are performed simultaneously, this means that $(z_{(j,j')} + z_{(j',j)}) = 0$.
If quay crane i and i' work in bay j and j' respectively, then i+1 $\leq$ i'.

$$C_{max} = \max_{j} t_j \tag{7}$$

Constraint (7) defines the C_{max} value.

$$x_{(j,i)} = \{0, 1\} \quad \forall j \in B, \forall i \in Q \tag{8}$$

$$z_{(j,j')} = \{0, 1\} \quad \forall j, j' \in B, j < j' \tag{9}$$

Constraints (8) and (9) define the property of the decision variables.

4 NP-completeness

In this section we discuss the complexity of the quay crane scheduling problem.

Before getting in details, we need some parameters in this discussion. Let's suppose that X is a set of positive integer, the working time $(Tc \cdot C_j) \in X$ $(1 \leq j \leq |B|)$ and Y is a given number which $Y \in X$.
The objective is to find all the schedules of the $|Q|$ quay cranes which unload containers from $|B|$ bays without interfering with each other and whose total completion time is less or equal to Y.

In the following, we present 4 phases to proof that the quay crane scheduling problem is NP-complete:

Phase 1: Indicate that the QCSP is in NP.
The feasibility of a given quay crane schedule for the QCSP problem can be analyzed in polynomial time, taking into account that the schedule must satisfy the non-interference constraints and thus can be realized in $O(|B|^2)$ time.
After that, we check that the completion time (makespan) $\leq Y$ which can be done in $O(|B|)$ time. Therefore, the QCSP is in NP.

Phase 2: Choosing an NP-complete problem.
The partition problem is a very known NP-complete problem ([1] HR Lewis (1983)).

The steps of this problem are defined as follows:
There are $|B|$ elements in a fixed set $L = \{l_1, l_2, \ldots, l_{|B|}\}$. For each element $l_j \in L, l_j \in X$ and the sum of all elements $\sum_{l_j \in L} l_j = Z$.
$\rightarrow$ Can the set L be divided into two separate sub-sets L_1 and L_2 such that $\sum_{l_j \in L_1} l_j = \sum_{l_j \in L_2} l_j = Z/2$?

Problem explanation with a numerical example: We suppose that the set $L = \{80, 117, 57, 100, 58, 110, 174\}$ and the sum of all elements $\sum_{l_j \in L} l_j = 696$. The answer to the previous question is Yes because the set L can be divided into two separate sub-sets $L_1 = \{80, 110, 58, 100\}$ and $L_2 = \{57, 117, 174\}$ such that $\sum_{l_j \in L_1} l_j = \sum_{l_j \in L_2} l_j = Z/2 = 348$. (See Fig. 4.)

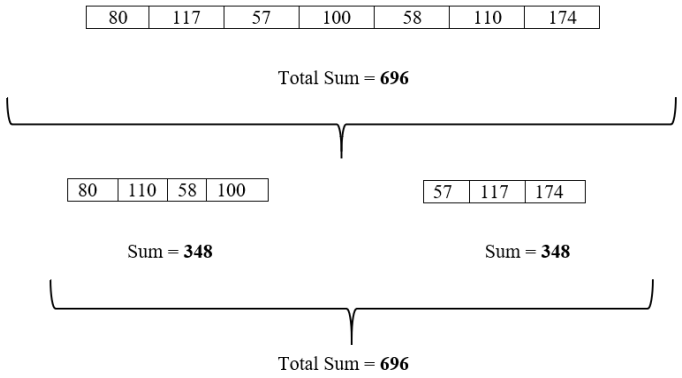

Fig. 4. Two subsets

The partition is obtained using the function below (developed in JAVA):

```
static int SplitPoint(int array[], int n){

        int lSum = 0;
        for (int i = 0; i < n; i++) {
                lSum += array[i];
                int rSum = 0;
                for (int j = i+1 ; j < n ; j++ )
                        rSum += array[j];
                if (lSum == rSum)
                        return i+1;
        }
        return -1;
}
```

The above function performs the following features:

- Traversing the array elements
- Adding the current element to a variable
- Finding the sum of the rest of the array elements
- Splitting the point index
- Returning -1 if it is not possible to split the array into two parts

Phase 3: Making a conversion from the partition problem to the quay crane scheduling problem.
The partition problem is switched to the QCSP as follows: An instance of the QCSP corresponding to an instance of the partition problem, has Q quay cranes and $|B| + |Q|$ bays. The given positive number X is set as Z, then the following equations (10, 11 and 12) show the processing time of each bay, in another meaning the processing time of bay 1 and bay $|B| + 2$ is set as $Z/2$, the processing time of bay 2 to bay $|B| + 1$ is set as l_1 to $l_{|B|}$, respectively, and the processing time of bay $|B| + 3$ to bay $|B| + |Q|$ is set as Z.

$$Tc \cdot C_1 = Tc \cdot C_{|B|+2} = Z/2 \tag{10}$$

$$Tc \cdot C_{j+1} = l_j \quad 1 \leq j \leq |B| \tag{11}$$

$$Tc \cdot C_j = Z \quad \forall |B| + 3 \leq j \leq |B| + |Q| \tag{12}$$

Table 1 shows this conversion. It indicates $|Q|$ quay cranes, $|B| + |Q|$ bays and the processing time of each bay.
After that, we must prove that the set L can be divided into two separate sub-sets L_1 and L_2 such that $\sum_{l_j \in L_1} l_j = \sum_{l_j \in L_2} l_j = Z/2$ **if** all the $|B| + |Q|$ bays can be completed by $|Q|$ quay cranes in Z time without interference between all quay cranes.
Firstly, we suppose that the set L can be divided into two separate sub-sets L_1 and L_2 such that $\sum_{l_j \in L_1} l_j = \sum_{l_j \in L_2} l_j = Z/2$.

Then, the scheduling of the $|Q|$ quay cranes without interference is as follows:

- Quay Crane 1 unloads containers from all the bays $j + 1$, where $l_j \in L_1$ and then bay 1.
- Quay Crane 2 unloads containers from bay $|B| + 2$, and then all the bays $j + 1$, where $l_j \in L_2$.
- Quay Cranes 3 unloads containers from bay $|B| + 3$.
- ...
- Quay Crane $|Q|$ unloads containers from bay $|B| + |Q|$.

Clearly, in this schedule the latest completion time between all bays is Z without any interference. Therefore, if the set L can be divided into two separate sub-sets L_1 and L_2 such that $\sum_{l_j \in L_1} l_j = \sum_{l_j \in L_2} l_j = Z/2$, all the $|B| + |Q|$ bays can be completely handled by $|Q|$ quay cranes in Z time without interference between all quay cranes.

Table 1. Phase 3 : Transformation

Quay crane	**Bay Number**	**Bay Processing time**
1	1	$D/2$
	2	l_1
	3	l_2
	...	...
	$\|B\|$	$l_{\|B\|-1}$
	$\|B\|+1$	$l_{\|B\|}$
2	$\|B\|+2$	$D/2$
3	$\|B\|+3$	D
4	$\|B\|+4$	D
...	...	...
$\|Q\|$	$\|B\|+\|Q\|$	D

In addition, suppose all the $|B| + |Q|$ bays can be completely handled by $|Q|$ quay cranes in Z time without any interference between them, then all the Q quay cranes are fully used as the sum of the processing time of all the bays is $|Q| \cdot |B|$.
Then, the completion time (makespan) of each quay crane must be equal to Z. Moreover, there is no interference in the previous quay crane schedule. Accordingly, the sum of the processing time of all bays, excepting bay 1 handled by Quay Crane 1 should be $Z/2$ and the sum of the processing time of all bays, excepting bay $|B|+2$ handled by Quay Crane 2 should be $Z/2$, which means that the set L can be

divided into two separate sub-sets L_1 and L_2 such that $\sum_{l_j \in L_1} l_j = \sum_{l_j \in L_2} l_j = Z/2$. Subsequently, if all the $|B|+|Q|$ bays can be completely handled by $|Q|$ quay cranes in Z time without interference between them, the set L can be divided into two separate sub-sets L_1 and L_2 such that $\sum_{l_j \in L_1} l_j = \sum_{l_j \in L_2} l_j = Z/2$.

Phase 4: Proof that the above conversion is polynomial. The above conversion can be done in $O(|B| + |Q|)$ time. Therefore, PARTITION PROBLEM $\propto QCSP$, and finally the theory is proved.

5 Conclusion

In this paper, we have investigated the problem of quay crane scheduling in a maritime port. The objective is how to orchestrate the planning of quay cranes that unload containers from bays without interfering with each other so that the total completion time is minimized. To this end, first we proposed a novel mixed integer linear programming model to solve the studied problem to optimilaty, and then we provided a theoritical analysis to prove that QCSP is NP-complete.

References

1. H. R. Lewis, "Michael r. πgarey and david s. johnson. computers and intractability. a guide to the theory of np-completeness. wh freeman and company, san francisco1979, x+ 338 pp.," *The Journal of Symbolic Logic*, vol. 48, no. 2, pp. 498–500, 1983.
2. J. Plesn *et al.*, "The np-completeness of the hamiltonian cycle problem in planar diagraphs with degree bound two," *Information Processing Letters*, vol. 8, no. 4, pp. 199–201, 1979.
3. C. A. Tovey, "A simplified np-complete satisfiability problem," *Discrete applied mathematics*, vol. 8, no. 1, pp. 85–89, 1984.
4. Q. Ouyang, P. D. Kaplan, S. Liu, and A. Libchaber, "Dna solution of the maximal clique problem," *Science*, vol. 278, no. 5337, pp. 446–449, 1997.
5. R. Kaye, "Minesweeper is np-complete," *The Mathematical Intelligencer*, vol. 22, no. 2, pp. 9–15, 2000.
6. Y. Liu, J. Xu, L. Pan, and S. Wang, "Dna solution of a graph coloring problem," *Journal of chemical information and computer sciences*, vol. 42, no. 3, pp. 524–528, 2002.
7. H. Kellerer, U. Pferschy, and D. Pisinger, "Introduction to np-completeness of knapsack problems," pp. 483–493, 2004.
8. B. Brešar, F. Kardoš, J. Katrenič, and G. Semanišin, "Minimum k-path vertex cover," *Discrete Applied Mathematics*, vol. 159, no. 12, pp. 1189–1195, 2011.
9. K. L. Hoffman, M. Padberg, G. Rinaldi, *et al.*, "Traveling salesman problem," *Encyclopedia of operations research and management science*, vol. 1, pp. 1573–1578, 2013.
10. L. V. Kantorovich, "Mathematical methods of organizing and planning production," *Management science*, vol. 6, no. 4, pp. 366–422, 1960.
11. A. Skaf, S. Lamrous, Z. Hammoudan, and M.-A. Manier, "Exact method for single vessel and multiple quay cranes to solve scheduling problem at port of tripoli-lebanon," pp. 457–461, 2018.
12. A. Skaf, S. Lamrous, Z. Hammoudan, and M.-A. Manier, "Genetic algorithm to optimize unloading of large containers vessel in port of tripoli-lebanon," pp. 569–574, 2019.
13. A. Skaf, S. Lamrous, Z. Hammoudan, and M.-A. Manier, "Solving methods for the quay crane scheduling problem at port of tripoli-lebanon," *RAIRO-Operations Research*, vol. 55, no. 1, pp. 115–133, 2021.