REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie

Département d'informatique

N° d'ordre : …IA06…../M2/2021

Mémoire

Présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : **Intelligence Artificielle (IA)**

Résolution du problème de stockage de conteneurs dans un port par un algorithme d'optimisation du coronavirus

---

Par :

MEHENNI SAID

Soutenu le 27/06/2022 devant le jury composé de :

TELLI Abdelmoutia	MCA	Président
ZOUAI Meftah	MAA	Examinateur
BERGHIDA Meryem	MCB	Encadrant

Année universitaire 2021-2022

Acknowledgments

I would like to express my gratitude to my supervisor Dr Berghida Meryem who supported and guided me throughout my Thesis .

I would like also to thank all my teachers and my colleagues.

Contents

List of Figures

List of Tables

General Introduction

Container terminals considered as the main part in the sea transportation, they offer facilities to move containers from to vessels and to vessels.

Containers are the critical logistics nodes and have a big Importance in national and international economies, in a container terminal they stacked on the storage yard temporary before they retrieved , the limited space of the storage yard and the big number of containers arrived that leads to a typical problem called “Container Relocation Problem”, it’s a result of stacking containers above each other, when the target container needs to be retrieved is blocked by containers above it, for that it is necessary to relocate them to achieve the target container, the relocations of the blocking containers are unproductive , for that should minimized.

the complexity of the problem gives an indication on the hardness of the problem, but it is important to consider the size, Small instances may be solved by an exact approach, and vise versa easy problems with high size can be solved with Metaheuristics because there is no exact approaches can solve CRP and find optimal solutions in a proper time, for that Caserta et al. [23] propose two formulations of the problem and show that the CRP is NP-hard ,

for that we propose *nature inspired approach,* the Corona Virus Optimization (CVO) Algorithm to solve the Container relocation problem because of the Corona virus spreading behavior it has aim to cover the search solutions space.

Our work organized on 3 chapters:

Chapter 1: this chapter related to the Combinatorial Optimization in general (we will take different aspects of Optimization , types of optimization Problems and different solving approaches).

Chapter 2: this chapter related to the Container relocation Problem(we will describe the Container relocation problem with different classifications and also the related works on it)

Chapter 3 : in this chapter we will set our contributions to solve the CRP , and the results get by testing instances , comparing to other heuristics proposed in that field).

CHAPTER 1

COMBINATORIAL OPTIMIZATION

1-1 INTRODUCTION

Combinatorial optimization is an area of research at the intersection of applied mathematics, computer science, and operational research. it overlaps with many other areas such as computation complexity, computational biology, VLSI design, communication networks, and management science. It includes complexity analysis and algorithm design for combinatorial optimization problems, numerical experiments and problem discovery with applications in science and engineering [7].

the authors in [5] describe combinatorial optimization problems as "problems where the decision space is finite but possibly too big to be enumerated.

1-2 BASICS

1-2-1 Diversification

Diversification means to generate diverse solutions so as to explore the search space on the global scale [1],

1-2-2 Intensification

Intensification strategies are based on modifying choice rules to encourage move combinations and solution features historically found good. They may also initiate a return to attractive regions to search them more thoroughly. Since elite solutions must be recorded in order to examine their immediate neighborhoods [2].

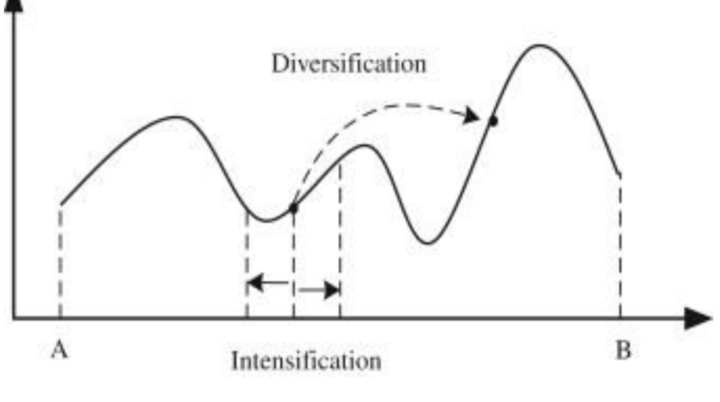


Fig. 1.1 Intensification and Diversification [22]

1-2-3 Neighborhood.

A neighborhood function N is a mapping $N : S \rightarrow 2^S$ that assigns to each solution s of S a set of solutions $N(s) \subset S$.

A solution s' in the neighborhood of s ($s' \in N(S)$) is called a neighbor of s. A neighbor is generated by the application of a move operator m that performs a small perturbation to the solution s.

The structure of the neighborhood depends on the target optimization problem. It has been first defined in continuous optimization. The neighborhood $N(s)$ of a solution s in a continuous space is the ball with center s and radius equal to ϵ with $\epsilon > 0$ [3]

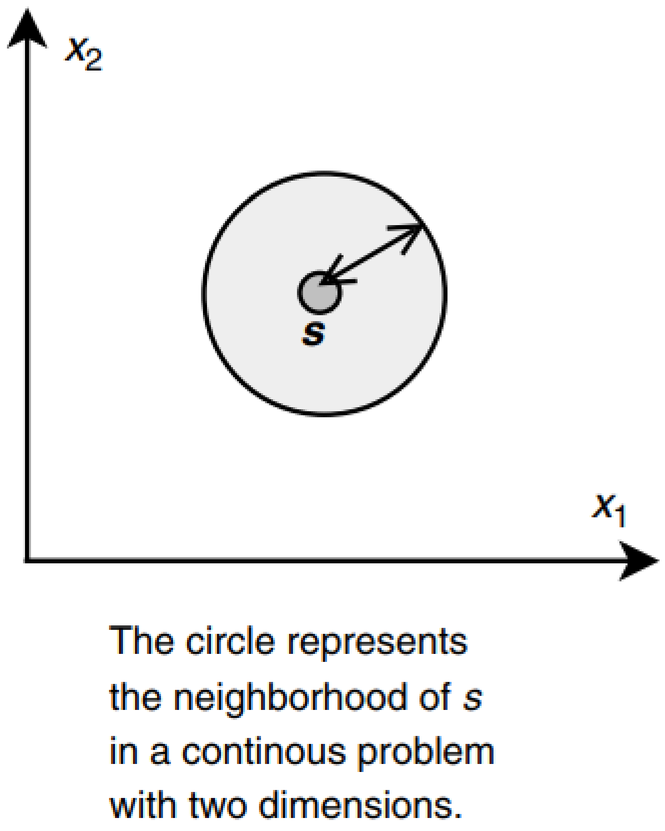

x_2
s
x_1
The circle represents the neighborhood of *s* in a continous problem with two dimensions.

Fig 1.2. Neighborhoods for a continuous problem [3]

1.2.4 Local Optimum and Global Optimum

1.2.4.1 Local optimum

Relatively to a given neighboring function N, a solution $s \in S$ is a local optimum if it has a better quality than all its neighbors; that is, $f(s) \leq f(s')^2$ for all $s' \in N(s)$ (Fig. 1.3).

For the same optimization problem, a local optimum for a neighborhood N1 may not be a local optimum for a different neighborhood N2.

1.2.4.1 Global optimum

A solution s* ∈ S is a global optimum if it has a better objective function than all solutions of the search space, that is, ∀s ∈ S, f(s *) ≤ f(s).

Hence, the main goal in solving an optimization problem is to find a global optimal solution s*. Many global optimal solutions may exist for a given problem. Hence, to get more alternatives, the problem may also be defined as finding all global optimal solutions. [3]

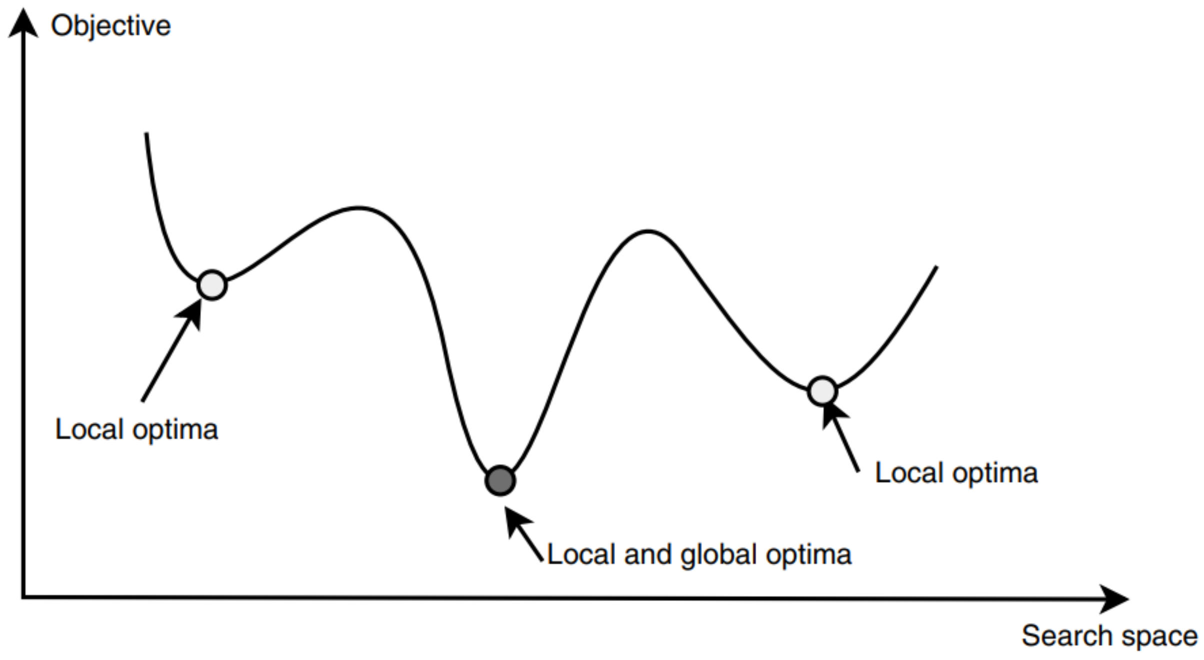


Fig 1.3. Local optimum and global optimum in a search space. A problem may have many global optimal solutions.

1.3 Complexity

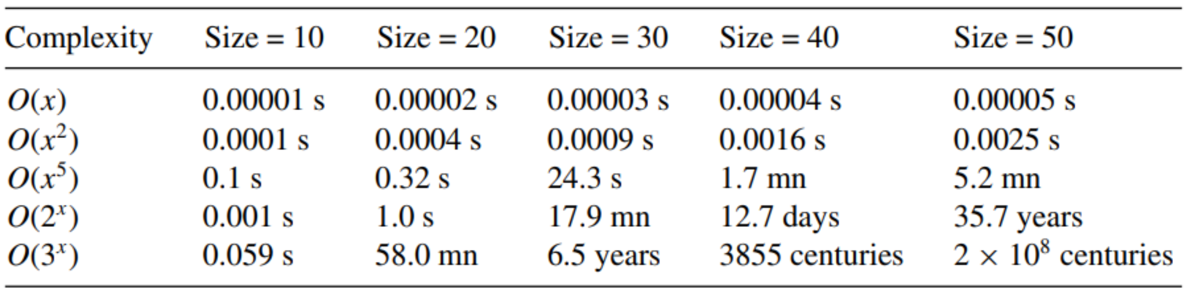
The complexity of an optimization problem divides on two parts, the complexity of the problem that we want to solve, and the complexity of the Algorithm we use to solve that problem.

1.3.1 Complexity of Algorithms

An algorithm needs two important resources to solve a problem: time and space. The time complexity of an algorithm is the number of steps required to solve a problem of size n.

The complexity is generally defined in terms of the worst-case analysis. The goal in the determination of the computational complexity of an algorithm is not to obtain an exact step count but an asymptotic bound on the step count.

Table 1.1 Search Time of an Algorithm as a Function of the Problem Size Using Different Complexities (from [4])



Complexity	Size = 10	Size = 20	Size = 30	Size = 40	Size = 50
$O(x)$	0.00001 s	0.00002 s	0.00003 s	0.00004 s	0.00005 s
$O(x^2)$	0.0001 s	0.0004 s	0.0009 s	0.0016 s	0.0025 s
$O(x^5)$	0.1 s	0.32 s	24.3 s	1.7 mn	5.2 mn
$O(2^x)$	0.001 s	1.0 s	17.9 mn	12.7 days	35.7 years
$O(3^x)$	0.059 s	58.0 mn	6.5 years	3855 centuries	2×10^8 centuries

1.3.2 Complexity of Problems

The complexity of a problem is equivalent to the complexity of the best algorithm solving that problem. A problem is tractable (or easy) if there exists a polynomial-time algorithm to solve it.

A problem is intractable (or difficult) if no polynomial-time algorithm exists to solve the problem. The complexity theory of problems deals with decision problems. A decision problem always has a yes or no answer [3].

1.4 Methods of solving optimization problems

Following the complexity of the problem, it may be solved by an exact method or an approximate method (Fig. 1.4) [3]. Exact methods obtain optimal solutions and guarantee their optimality.

For NP-complete problems, exact algorithms are non-polynomial time algorithms (unless P = NP). Approximate (or heuristic) methods generate high quality solutions in a reasonable time for practical use, but there is no guarantee of finding a global optimal solution. [3]

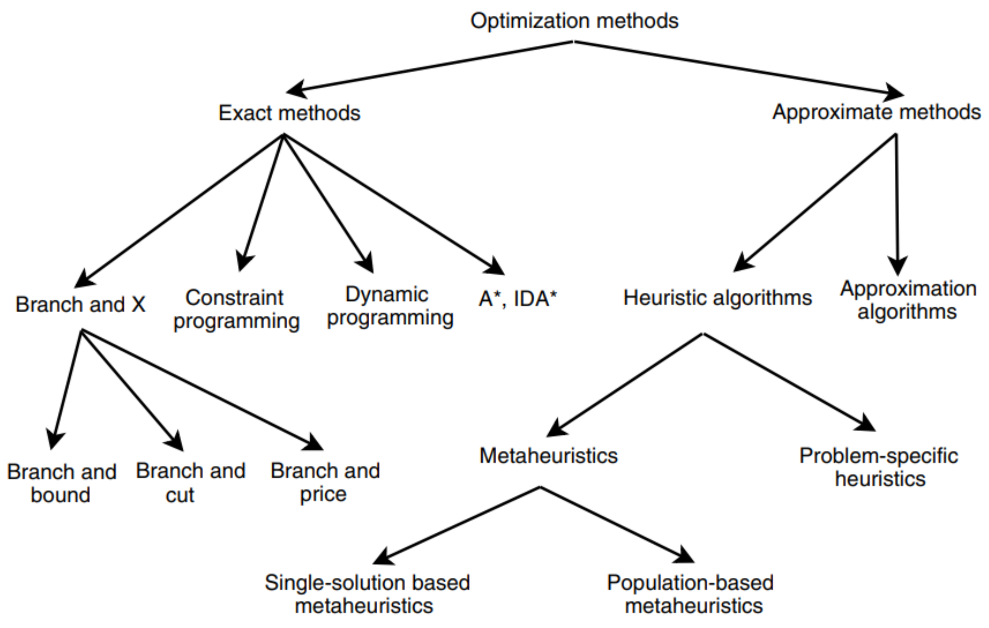


Fig 1.4 Classical optimization methods

1.4.1 Exact Methods

In the class of exact methods one can find the following classical algorithms: dynamic programming, branch and X family of algorithms (branch and bound, branch and cut, branch and price) developed in the operations research community, constraint programming, and A∗ family of search algorithms (A* , IDA*—iterative deepening algorithms) developed in the artificial intelligence community . Those enumerative methods may be viewed as tree search algorithms. The search is carried out over the whole interesting search space, and the problem is solved by subdividing it into simpler problems.[3]

1.4.2 Heuristics

The term heuristic derives from the Greek verb heuriskein ðtqirjimÞ that means ‘‘to find’’. In fact, heuristics are basic approximate algorithms that search the solution space to find a good solution. There are mainly two types of heuristics: constructive algorithms and local search algorithms. Constructive algorithms build a solution by joining together ‘‘pieces’’, or components, of a solution, that are added one after the other until a solution is complete. Local search algorithms start from a pre-existent solution (called a current solution) and try to improve it by modifying some of its components (such a modification is called a move)[5].

1.4.3 Metaheurstics

The Greek suffix ‘‘meta’’ used in the word metaheurstic means ‘‘beyond, in an upper level’’. Thus, metaheuristics are algorithms that combine heuristics (that are usually very problem-specific) in a more general framework.

According to Blum and Roli (2003) [5] **Metaheuristics** are applied to “I know it when I see it problems”. They’re algorithms used to find answers to problems when you have very little to help you: you don’t know beforehand what the optimal solution looks like, you don’t know how to go about finding it in a principled way, you have very little heuristic information to go on, and brute-force search is out of the question because the space is too large. But if you’re given a candidate solution to your problem, you can test it and assess how good it is. That is, you know a good one when you see it. For example: imagine if you’re trying to find an optimal set of robot behaviors for a soccer goalie robot. You have a simulator for the robot and can test any given robot behavior set and assign it a quality (you know a good one when you see it). And you’ve come up with a definition for what robot behavior sets look like in general. But you have no idea what the optimal behavior set is, nor even how to go about finding it.[6]

Many classification criteria may be used for metaheuristics:

• ***Nature inspired versus non-nature inspired***: Many metaheuristics are inspired by natural processes: evolutionary algorithms and artificial immune systems from biology; ants, bees colonies, and particle swarm optimization from swarm intelligence into different species (social sciences); and simulated annealing from physics.

• ***Memory usage versus memoryless methods:*** Some metaheuristic algorithms are memoryless; that is, no information extracted dynamically is used during the search. Some representatives of this class are local search, GRASP, and simulated annealing. While other metaheuristics use a memory that contains some information extracted online during the search. For instance, short-term and long-term memories in tabu search.

• ***Deterministic versus stochastic:*** A deterministic metaheuristic solves an optimization problem by making deterministic decisions (e.g., local search, tabu search). In stochastic metaheuristics, some random rules are applied during the search (e.g., simulated annealing, evolutionary algorithms). In deterministic algorithms, using the same initial solution will lead to the same final solution, whereas in stochastic metaheuristics, different final solutions may be obtained from the same initial solution. This characteristic must be taken into account in the performance evaluation of metaheuristic algorithms.

• ***Population-based search versus single-solution based search:*** Single-solution based algorithms (e.g., local search, simulated annealing) manipulate and transform a single solution during the search while in population-based algorithms (e.g., particle swarm, evolutionary algorithms) a whole population of solutions is evolved. These two families have complementary characteristics: single-solution based metaheuristics are exploitation oriented; they have the power to intensify the search in local regions. Population-based metaheuristics are exploration oriented; they allow a better diversification in the whole search space. In the next chapters of this book, we have mainly used this classification. In fact, the algorithms belonging to each family of metaheuristics share many search mechanisms.

• ***Iterative versus greedy:*** In iterative algorithms, we start with a complete solution (or population of solutions) and transform it at each iteration using some search operators. Greedy algorithms start from an empty solution, and at each step a decision variable of the problem is assigned until a complete solution is obtained. Most of the metaheuristics are iterative algorithms [3]

1-5 Coronavirus Optimization Algorithm (CVO):

1-5-1 Corona virus biology and ecology:

Since COVID-19 was first reported on December 31, 2019 in Wuhan, China, it has rapidly spread. The virus has infected people around the world and can, therefore, be considered as a **pandemic**.

As COVID-19 continues its global proliferation, governments have implemented regulations and instructed their citizens to adopt certain measures to combat the virus. Curfews, city quarantines, fines, and business closures have been deemed necessary by governments, while individuals are advised of preventive practices, such as regular hand washing, wearing masks, staying at home, and observing social distancing. Health organizations, including the WHO, have also taken steps to curb and reduce the prevalence of COVID-19. [8]

One of the most effective measures for controlling any contagious disease is its mathematical modeling. A well-designed model of an infectious disease can help predict the disease's behavior and so facilitate the planning of defensive strategies.

One of the most important parameters in infectious disease modeling is the basic reproductive number (denoted as **R0**), which is a function of the contact rate between individuals, symptom transition probability, and the duration of infectiousness.

This number denotes the average number of persons infected by a single individual:

- If $R0 < 1$, then each infected person can only infect less than one other person; therefore, the growth of the disease is expected to stop.
- If $R0 = 1$, then each infected person can infect one other person on average, thus resulting in stable disease growth[8]. This condition is called endemic, when the number of infected people does not increase or decrease.
- However, if $R0 > 1$, then each person can infect more than one other person on average and, as a result, the disease, if left untreated, is expected to grow exponentially and consequently lead to an epidemic or pandemic.

Since COVID-19 is pandemic, the R0 rate is certainly greater than 1. According to research, the R0 rate for COVID-19 depends on a number of factors that include climate, race, population density, age, percentage of public transportation use, and even average income [8].

Accordingly, this number varies among different countries and even in different regions of a country. The average range for the COVID-19 R0 rate is estimated between 2 and 12 for different parts of the world.

Based on R0, Fig. 1.5 shows how the COVID-19 virus has been transmitted and spread since its discovery. As seen in Fig. 1.5 [8], the degree of attention paid to COVID19 protocols by the public has so far prevented the entire world population from contracting the virus. As a result, only a part of the world's population has become infected, which is perhaps due to a lack of care in following protocols or a weakened immune system.

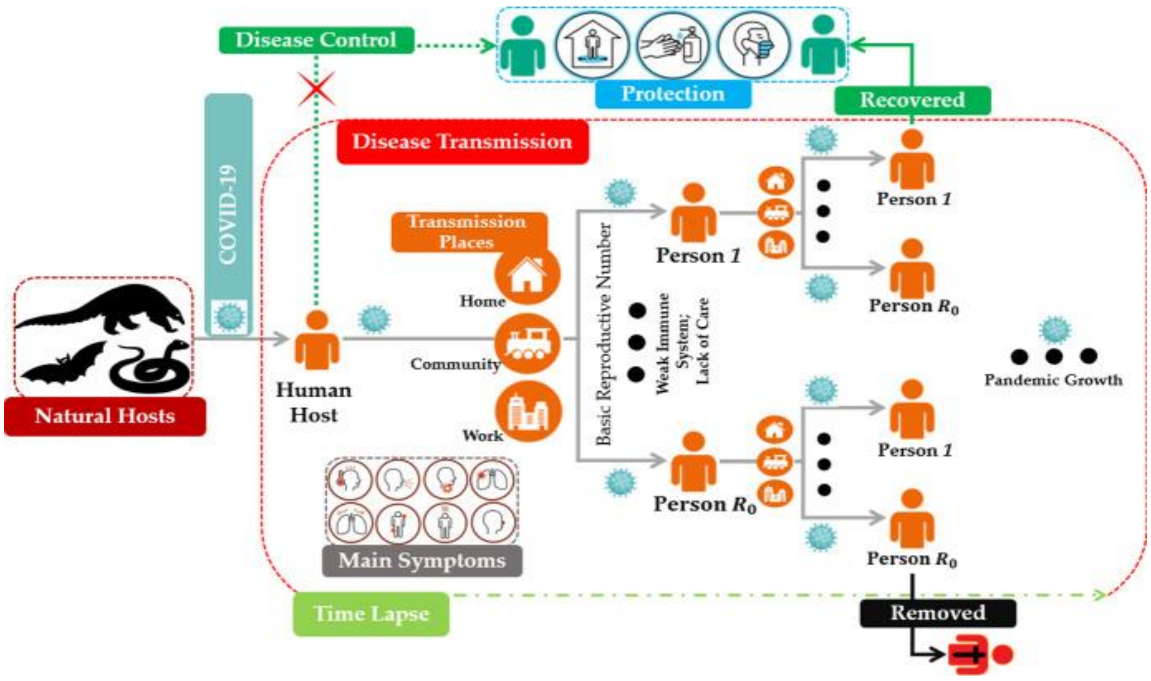


Fig 1.5 How the COVID-19 virus has been transmitted and spread since its discovery, based on the value of R0

Mathematical models of the disease can simulate the transmission process at different levels. Some types of models show cell interactions in a single patient, while others present the prevalence of the virus among different communities that are geographically dispersed.

All the mathematical models proposed for COVID-19 have been of the latter type. Since the beginning of the COVID-19 outbreak, a number of mathematical models have been introduced. The basis of most of these models is SIR, which is a standard compartmental disease model that consists of three compartments: susceptible, infectious, and recovered/removed. One of these compartments is assigned to each member of the community.

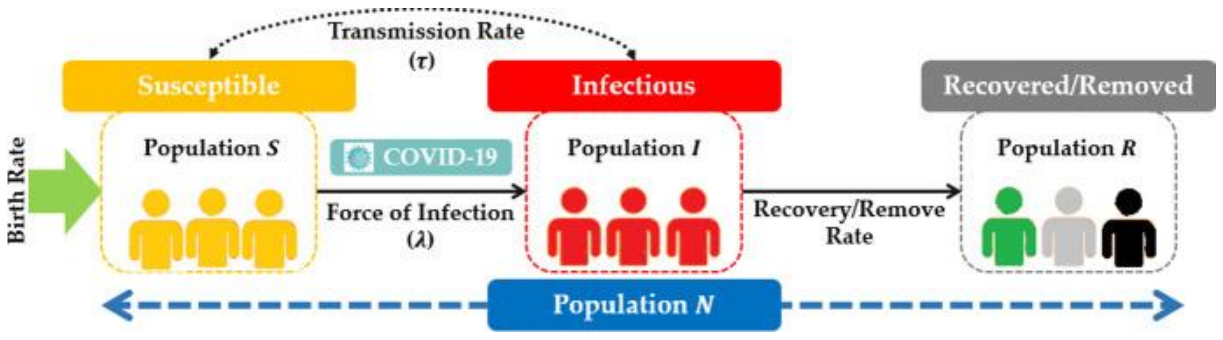


Fig. 1.6. The interactions in the simplest SIR model among the three COVID19 compartments: susceptible, infectious, and recovered/removed

1-5-2 Corona virus optimization algorithm

The Corona virus optimization CVO algorithm is a new optimization algorithm. It was proposed by Alireza Salehan and Arash Deldari [8].

Table 1 presents the most important parameters and variables used in the CVO algorithm. The table lists the concepts inspired by COVID-19 and describes the parameters and variables considered in the optimization problem space.

For example, the difference between the LocalBest and the GlobalBest is that the LocalBest variable stores the best solution in each iteration of the algorithm, while the GlobalBest variable stores the best solution in all iterations.

In general, the CVO approach considers that the disease transmission process begins in a limited population and continues based on pandemic iterations. In each iteration and after the transmission of symptoms from infected to susceptible individuals, the susceptible individuals with weaker immune systems become ill and are added to the infectious population. It is assumed that individuals with stronger immune systems are transferred to the recovered/removed population. At the end of each iteration, if the value of the weakest immune system, which is stored in the LocalBest variable, is less than the GlobalBest, then the GlobalBest variable is updated.

Table 1 Some of the most important parameters and variables for CVO algorithm implementation

Name	**Concept based on COVID-19**	**Description in Problem Space**
Iters	The number of recurrences of the COVID-19 pandemic	The number of iterations
basePop	The number of initial infectious populations	Initial solutions
nPop	The maximum number of populations becoming infected	The maximum number of solutions in all iterations
nVar	The number of symptoms	The variables of optimization problem
LB	Lower bound of symptoms	The minimum value of each variable
UB	Upper bound of symptoms	The maximum value of each variable
GlobalBest	Patient with the weakest immune system during the pandemic	The best solution in all iterations
LocalBest	Patient with the weakest immune system during each recurrence	The best solution in each iteration
CVO	Severity of COVID-19 infection	Fitness function
R0	Basic reproductive number	The number of new solutions created for each current solution
pop	All patients in all recurrences	The set of the solutions with less fitness function values in all iterations
newPop	New patients in each recurrence	New solutions in each iteration

Algorithm 1 [8] represents the mechanism of the proposed CVO method. The input parameters of this algorithm are Iters, basePop, nPop, nVar, LB, UB, and R0. The output of the algorithm is the best solution which is returned by the GlobalBest variable.

This variable consists of the symptoms (the optimization problem variables) and the value of the weakest immune system (the optimization problem solution) throughout the whole algorithm's iterations. Lines 1 to 16 initialize the GlobalBest variable and the initial population. In line 3 of the algorithm, according to the force of transmission of COVID-19 and the possibility of its transmission even from the first contact, the value of the contact rate is equal to 1. [8]

If the optimization problem is continuous, in lines 5 to 7 for each basePop member of the population, a set of random numbers in the interval [LB,UB] are generated and stored in pop as the initial symptoms of the current patient. However, if the optimization problem is discrete, in lines 8 to 10 for each member of the initial population, a random permutation of the problem variables is assigned as the initial symptoms. In line 11, the value of the fitness function is calculated for each member of the pop population and, if it is less than the GlobalBest value, it is replaced in lines 12 and 13. [8]

The iterations of the CVO algorithm are performed in lines 17 to 53. In each iteration:

- first a population of new patients is created for the current iteration (line 18) and then the value of the LocalBest variable is initialized (lines 19 and 20).
- Lines 21 to 40 check the possibility of transmitting the disease from infected to susceptible individuals during a single iteration.
- Line 22 shows that any infected person can infect R0 susceptible individuals. If the type of problem is continuous, the new potential symptoms are determined based on Eqs. 1, 2, 3 in lines 23 to 27, the value of which must be in the interval [LB, UB].
- In line 25 of the algorithm, |newPop| denotes the number of new patient populations in the current iteration. However, if the problem is discrete, lines 28 to 30 produce new susceptible symptoms by randomly replacing a permutation of symptoms of the previous patient.
- In line 31 of the algorithm, the immunity level of the new susceptible person is determined based on the symptoms and using the optimization fitness function.
- If the level of immunity for the susceptible person is less than the immunity level of the infected person transmitting the disease, the susceptible person will be added to the new patient population in lines 32–34.
- Moreover, if the immunity level of the susceptible person is less than the immunity level of the entire patient population during one iteration, the symptoms and immunity

level of the susceptible person will replace LocalBest in lines 35 to 38 of the algorithm.[8]

- It can be assumed that susceptible individuals with a higher immunity level are those who have followed health protocols and are, therefore, not added to the new patient population.
- After the new patient population is determined during one iteration, the GlobalBest values and the members of the entire patient population are updated for this iteration in lines 41–52.
- In lines 41–46, if the value obtained for the LocalBest is less than the GlobalBest, the GlobalBest value is replaced by the LocalBest and all members of the new patient population in this iteration will replace the total population of pop patients.
- Otherwise, in line 45, new patients will be added to the pop patient set.
- In line 47 of the algorithm, the number of the initial population's members is updated for the next iteration which is equal to the number of pop members. [8]
- If this initial population exceeds the total population (which is one of the algorithm's input parameters), the pop population is sorted in ascending order by the value of the immune system in lines 48–52. Therefore, the smaller nPop number is selected as the pop population and the value of the initial population variable is updated.

The proposed approach reduces the population of infected people as a result of their recovery or death, a policy inspired by COVID-19 behavior.[8]

Algorithm 1 The CVO Mechanism

Inputs: The values of **Iters, basePop, nPop, nVar, LB, UB**, and R_0

```
// Initialization of GlobalBest and Initial Infectious Population
GlobalBest.Symptoms = NULL ;
GlobalBest.ImmuneSystem = +∞ ;
$\rho = 1$ ;
for (i = 1 ; i ≤ basePop ; i ++)
  if (Problem type == Continues)
    $pop_i$.Symptoms = LB ≤ nVar numbers of random values ≤ UB ;
  endif
  if (Problem type == Discrete)
    $pop_i$.Symptoms = Random_Permutation (nVar) ;
  endif
  $pop_i$.ImmuneSystem = CVO ($pop_i$.Symptoms) ;
  if ($pop_i$.ImmuneSystem < GlobalBest.ImmuneSystem)
    GlobalBest.Symptoms = $pop_i$.Symptoms ;
    GlobalBest.ImmuneSystem = $pop_i$.ImmuneSystem ;
  endif
endfor_i
// Iterations of COVID-19 Pandemic
for (iter = 1 ; iter ≤ Iters ; iter ++)
  newPop = ∅ ;
  LocalBest.Symptoms = NULL ;
  LocalBest.ImmuneSystem = +∞ ;
  for (i = 1 ; i ≤ basePop ; i ++)
    for (j = 1 ; j ≤ $R_0$; j ++)
      if (Problem type == Continues)
        $\varphi$ = NormalRandom(0,1) ; // Equation 3
        $\lambda = \rho \times \varphi \times \frac{basePop + |newPop|}{nPop}$ ; // Equations 1 and 2
        newSusceptible.Symptoms = LB ≤ $pop_i$.Symptoms + $\lambda$ ≤ UB ;
      endif
      if (Problem type == Discrete)
        newSusceptible.Symptoms = Random_Permutation ($pop_i$.Symptoms) ;
      endif
      newSusceptible.ImmuneSystem = CVO (newSusceptible.Symptoms) ;
      if (newSusceptible.ImmuneSystem ≤ $pop_i$.ImmuneSystem)
        Add newSusceptible to newPop ;
      endif
      if (newSusceptible.ImmuneSystem < LocalBest.ImmuneSystem)
        LocalBest.Symptoms = newSusceptible.Symptoms ;
        LocalBest.ImmuneSystem = newSusceptible.ImmuneSystem ;
      endif
    endfor_j
  endfor_i
  if (LocalBest.ImmuneSystem < GlobalBest.ImmuneSystem)
    Replace GlobalBest with LocalBest ;
    Replace all pop members with newPop members ;
  else
    Insert all newPop members to pop ;
  endif
  basePop = |pop| ;
  if (basePop > nPop)
    Sort pop members ASCENDING based on ImmuneSystem;
    Select the first nPop members from pop ;
    basePop = nPop ;
  endif
endfor_iter
```

Objective output: The best solution **GlobalBest**

CHAPTER 2

CONTAINER RELOCATION PROBLEM (CRP)

2-1 Containerization

Containerization is the use of containers as a tools of transferring goods and merchandise. This concept appeared only in the 20th century, but since then it has become an indispensable element in the field of transport. Many elements have contributed to its success, among which we can cite its multi-modal character which makes it possible to transition between different modes of transport. This advantage, combined with the possibility of geo-localization of cargo, With the computer systems, quickly won over exporters and at the same time contributed to the globalization of trade. Thus becoming an international tool, the container is then standardized with the agreements between transport companies. Improvements and specifications have been subsequently carried out in order to make the containers more compatible with certain types of cargo. However, the container is not necessarily an infallible element, because even while it has many advantages, it has made manual checks nearly impossible. By Therefore, additional e orts have been required to create tools of technical control **[14]**

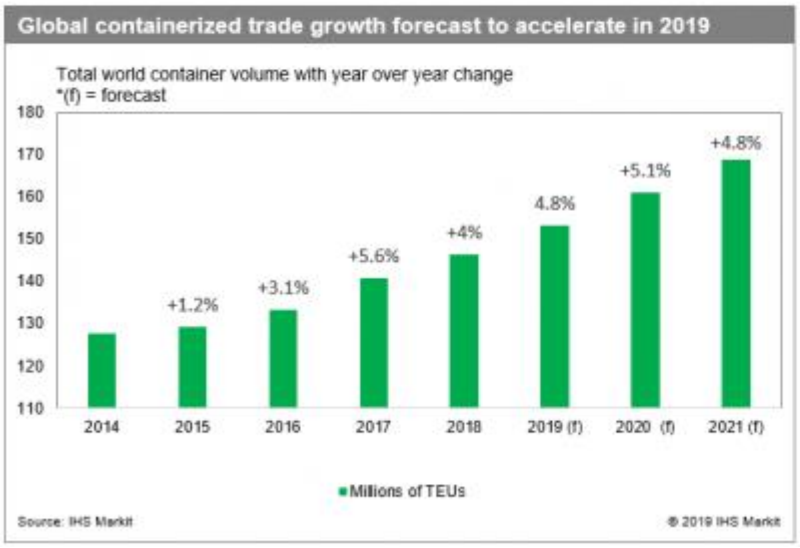


Fig 2.1. Global container growth forecast to rebound

2.2 Terminal structure and handling equipment:

In general terms, container terminals can be described as open systems of material flow with two external interfaces. These interfaces are the quayside with loading and unloading of ships, and the landside where containers are loaded and unloaded on/off trucks and trains.

Containers are stored in stacks thus facilitating the decoupling of quayside and landside operation. After arrival at the port, a container vessel is assigned to a berth equipped with cranes to load and unload containers. Unloaded import containers are transported to yard positions near to the place where they will be transshipped next. Containers arriving by road or railway at the terminal are handled within the truck and train operation areas. They are picked up by the internal equipment and distributed to the respective stocks in the yard. Additional moves are performed if sheds and/or empty depots exist within a terminal; these moves encompass the transports between empty stock, packing center, and import and export container stocks [15] (Fig. 2.2)

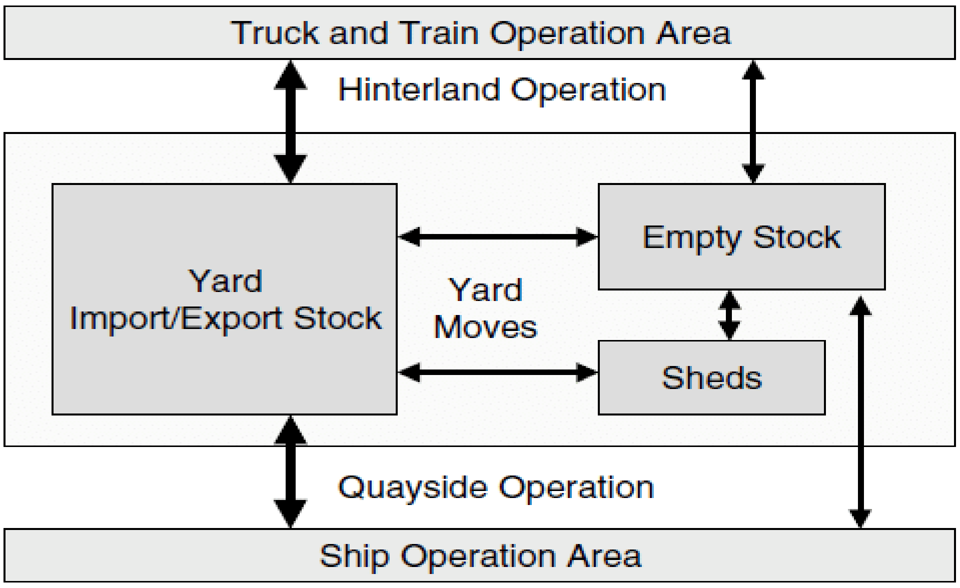


Fig. 2.2. Operation areas of a seaport container terminal and flow of transports

2.3. Problem description

To decouple seaside and landside operations, incoming containers are not immediately loaded on an outgoing vehicle, but stored in the yard for up to several days. Due to limited space, terminals stack containers. Consequently, only the topmost container of each stack can be accessed directly. If another container has to be retrieved, containers above have to be relocated. These unproductive relocations (also called reshuffles or rehandles) should be avoided since they increase the retrieval time and hence the overall performance of the terminal. However, relocations cannot be avoided completely as little information about future retrievals is known when a container has to be stored.

The number of relocations increases with the stacking height of containers and is therefore a bigger issue at terminals using stacking cranes for storage operations. The yard of such a terminal is illustrated in Figure 2.3. The yard is divided into different blocks. Each block consists of several bays, each bay of several stacks and each stack of several tiers. Thanks to new technologies, the terminal knows exactly at which position (block, bay, stack, tier) each container is stored and which positions are empty. [9]

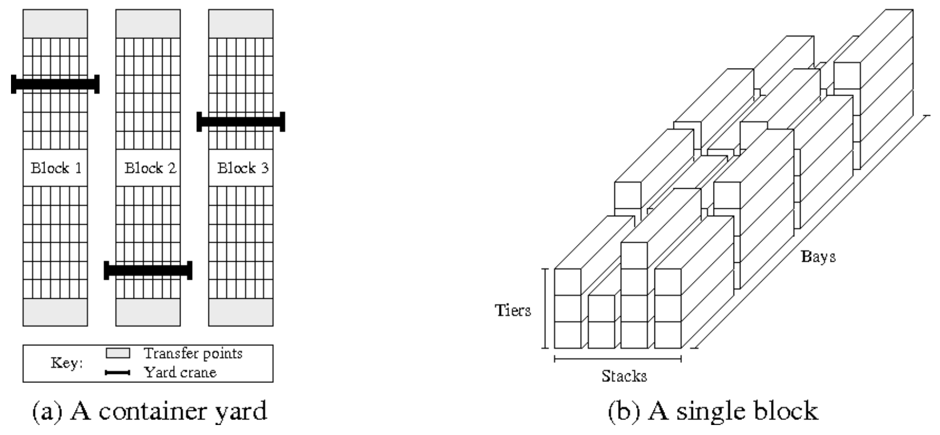


Fig 2.3. Blocks, bays, stacks and tiers

Decisions where to place containers are taken, when containers enter the terminal or when they have to be relocated. Different academic problems have been extracted for yard optimization: the storage space allocation problem to determine storage locations (a block or a single position) for incoming containers; the remarshalling / premarshalling problem to reorganize a block / a bay in less busy periods as new information becomes available in order to reduce the number of relocations during the retrieval process;

the container relocation problem to retrieve all containers from a bay in a given sequence with a minimum number of relocations. We deal with the container relocation problem. In this case, the stowage plan of vessels and the service order of trucks are known and impose the retrieval order of containers. Generally, the storage layout does not match the retrieval order and containers have to be relocated. Figure 2.4 illustrates the problem. The objective is to retrieve all containers in the given sequence with a minimum number of relocations.

Two variants of the problem exist: all containers may be relocated or only containers above the current target container may be relocated

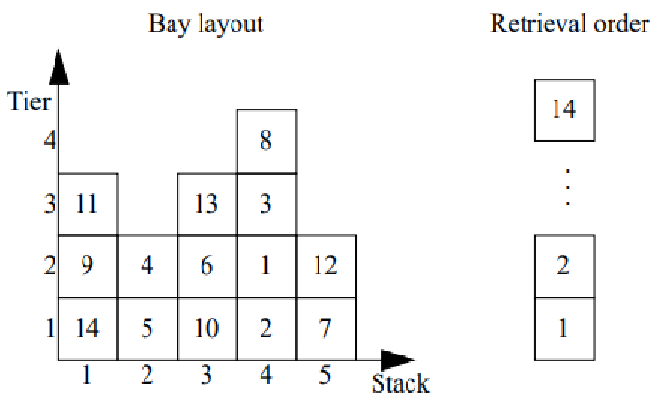


Fig 2.4. Container relocation problem

2.4. CRP Classifications

Static / Dynamic CRP:

If there are no new containers during the retrieval process to be stacked on the bay, the problem is called static CRP, otherwise it's called dynamic CRP. Since the crane needs to serve the whole block, the dynamic CRP within one bay is usually not under consideration [16].

Restricted / Unrestricted CRP: CRP is restricted, if relocations are only allowed for the blocking containers above the container with highest priority. Otherwise, it's unrestricted, which means the unrestricted CRP is the super-set of restricted CRP. Generally, it has lower relocation rate and its corresponding algorithm is more complex than restricted CRP. [16]

Stochastic CRP: If the retrieval sequence is not fully known, for instance, several containers shall be stacked on a train, then the retrieval sequence is not important as long as the corresponding containers are stacked on the correct position. [16]

CRP in block: In reality, relocations could happen in whole or part of container yard, which is defined as a block (Figure 1). In this scenario, the relocation rate could not be the single judgment of the problem; instead, several new judgments were introduced, like average operation time of container and average waiting time of truck Furthermore, the above-described types could be combined in this scenario. [16]

2.5.Variables

To represent the container relocation problem [10]:

- A bay consists of W stacks and H tiers. Each slot within the bay is addressed with coordinates (i, j) where i ∈ {1, . . . , W} and j ∈ {1, . . . , H}.
- The initial configuration contains N containers, labeled 1, . . . , N.
- Containers have to be retrieved in ascending order, e.g. container 1 is the first one to be retrieved and container N the last one.
- At each time period t (t = 1, . . . , T), container n = t is retrieved and any blocking containers are relocated

2.6. Constraints

The problem definition relies on assumptions A1 to A7[9]:

1- A1: The initial bay layout and precedence constraints among single containers or groups of containers are known in advance.
2- A2: No new containers arrive during the retrieval process.
3- A3: Only the topmost container of a stack can be picked up. A relocated container can only be put on the top of another stack or on the ground.
4- A4: Containers are only relocated within the bay since relocations between bays are very time consuming.
5- A5: The bay size is limited by the maximum numbers of stacks and tiers.
6- A6: Containers in the same bay have the same size and can be piled up in any order.
7- A7: The distance traveled within one bay (horizontally and vertically) has little impact on the time to relocate or to retrieve containers.
8- A8: Only blocking containers located above the current target container may be relocated.

2.7. Related work

in this section we mention the different works done by researchers to solve the problems related to the Container Relocation Problem in ascending schedule in the last years as table 2.1 shows below:

Year	Related work
2016	Ndèye Fatma Ndiaye in her these entitled with “Algorithmes d’optimisation pour la résolution du problème de stockage de conteneurs dans un terminal portuaire” [14] she propose meta-heuristic algorithms, namely: a bee colony algorithm, a genetic algorithm, an ant colony algorithm, and a simulated annealing algorithm. These algorithms are approximate solution methods, i.e. the optimality of the solutions they provide is not of the solutions they provide is not guaranteed,. In order to combine their performances and to avoid the fast convergence towards a local optimum, hybridizations between these hybrids between these different meta-heuristic algorithms have been proposed. These hybrid algorithms have proved their efficiency by the good quality of their solutions. The first hybridization is The first hybridization is a reinforcement of the ant colony algorithm by a genetic algorithm, while the other two represent local searches at each iteration performed by the simulated annealing algorithm. The first hybridization is a reinforcement of the ant colony

	algorithm by a genetic algorithm, while the other two represent local searches at each iteration performed by the simulated annealing algorithm in the ant colony algorithm on the one hand, and the genetic algorithm on the other. These hybridizations have allowed to accelerate the convergence of the of the algorithms and sometimes to improve the quality of the best solutions obtained.
2018	a paper titled with "The Stochastic Container Relocation Problem" [16] , the authors worked on a specific type of CRP which is Stochastic Container Relocation Problem (SCRP) which relaxes the CRP assumption of knowing the full retrieval order of containers and consider it particularly unrealistic in real operations.
2018	Consuelo Parre~no-Torres*, Ramon Alvarez-Valdes they worked with "the pre-marshalling problem" [17] which consists in rearranging the containers placed in a bay in the order in which they will be required later, looking for a sequence with the minimum number of moves. With sorted bays, loading/unloading operations are significantly faster, as there is no longer a need to make unproductive moves in the bays once ships are berthed. we address the pre-marshalling problem by developing and testing integer linear programming models.
2019	MAGLIC, Marko GULIC, Lovro MAGLIC [18].they worked with the CRP problem by finding out if Genetic Algorithm (GA) can give new insights in the problem of solving the CRP. In this paper we focus on the two-dimensional, static, offline and restricted CRP of real-world yard container bays. Four rules are proposed for determining the position of relocated containers. they applied GA to find the best sequence of container retrievals according to these four rules in order to minimize the number of relocations within the bay .
2021	Tiecheng Jiang solved the Container Relocation Problem via Reinforcement Learning based on a concept proposed called "blocking degree", then choose the stack with the lowest count to relocate a container above. [19]
2021	Lei Wei and Fuyin Wie in [13] solved the Container Relocation Problem via Reinforcement Learning based on a new concept proposed called "blocking count", itself is an adjust of the blocking degree proposed by Tiecheng Jiang .[19]
2021	Omayma EL Majdoubi used the Corona Virus Algorithm to solve the Travelling salesman problem. The main properties of CVOA are as follows: The probabilities and parameters are defined also updated by scientific community, the exploration of search space is handled as long as the infected population is not null and the high rate of expansion ensures better use of search space leading to the intensification of the resolution.[20]

CHAPTER 3

CONCEPTION

EXPERIMENTS AND RESULTS

3.1. Contributions

3.1.1. Related to the Corona Virus Algorithm

Adapting the original Algorithm to the Container Relocation Problem:

To adapt heuristics and metaheuristics to a particular optimization problem, a good representation of the considered problem is needed [10]

firstly the solution of a CRP is just a sequence of a relocations with a given order of retrievals, in the CVOA a single solution is Pop_i from a set of solutions pop , pop_i consist of a set of symptoms , in our case with CRP the set of symptoms is the sequence of retrievals , the intensity of each symptom is the number of Relocations to retrieve a Container.

in the original Algorithm of CVOA to during the infection and the transmission of the virus from individual to a new infected one there is a permutation of the symptoms order, but in our case that permutation is impossible because we have a given order of retrievals, for that we change the Algorithm to adapt it to CRP.

Because the order of retrievals of the Container is a mandatory, and it's unchangeable, for that we cannot use the exact Algorithm for the reason that in the CVOA the generation of a new infected individuals is based on the permutation of the symptoms, and in CRP case we can't do that permutation because the order is respectable, for that we propose an adjustment for the CVOA in Algorithm 2.1:

Algorithm 2.1 : CVOA adapted to CRP

Inputs : The values of Iters, basePop, nPop, nVar, LB, UB, and R_0

```
// Initialization of GlobalBest and Initial Infection Population
GlobalBest.Symptoms = NULL;
GlobalBest.ImmuneSystem = +∞;
for(i = 1 ; i ≤ basePop ; i++ )
    // determine set of symptoms or find a solution by using Heuristic HC
    //popi.Symptoms = Random_Permutation(nVar) ;
    popi.Symptoms =nVar measure_symptom();
    popi.ImmuneSystem = CVO (popi.Symptoms);
    if(popi.ImmuneSystem≤ GlobalBest.ImmuneSystem)
        GlobalBest.Symptoms = popi.Symptoms;
        GlobalBest.ImmuneSystem = popi.ImmuneSystem;
    endif
endfor_i
```

```
  //Iteration of COVID-19 Pandemic
 for(iter = 1 ; iter ≤ Iters ; iter++ )
 newPop = Ø;
 LocalBest.Symptoms = NULL;
     LocalBest.ImmuneSystem = +∞;
 for(i = 1 ; i ≤ basePop ; i++ )
         for(j = 1 ; j ≤ R0 ; j++ )
         //newSusceptible.Symptoms =Random_Permutation (popi. Symptoms)
                 M = Random(min,max)  // M :  Corona Virus Mutation Probability
                 newSusceptible.(nVar-M) Symptoms = pop_i. .(nVar-M)Symptoms
                 newSusceptible.M Symptoms = pop_i.M measure_symptom();
                 newSusceptible.ImmuneSystem = CVO (newSusceptible. Symptoms);
                 if(newSusceptible.ImmuneSystem≤  pop_i.ImmuneSystem)
                 LocalBest.Symptoms= newSusceptible. Symptoms;
                 LocalBest.ImmuneSystem = newSusceptible.ImmuneSystem;
                 endif
         endfor_ j
     endfor_ i
 if(LocalBest.ImmuneSystem≤  GlobalBest.ImmuneSystem)
         ReplaceGlobalBest withLocalBest
         Replaceall pop memberswith newPop members;
    else
 insert all newPop members to pop
 endif
 basePop = |pop|;
 if (basePop > nPop)
 Sort pop membes ASCENDING based on ImmuneSystem
    select the first nPop members from pop
 basepop = nPop
endfor_ iter
```

3.1.2. Related to the heuristic used to choose the best stack for the relocated container

It is based on the score of the stack, the container relocated to the stack of the best score (we consider the best score is the minimum) determined in the Equations below:

Equation 2.1 $s(i) = \sum j * tier(j)$ //$j \in W$, tier(j) = number of tiers for the j container j

Equation 2.2$i^* \leftarrow {}_{\min}\sum j * tier(j)$ //$j \in W$

Algorithm 2.2 Heuristic HC for the container relocation problem[9]

Input: a bay layout

Output: a solution for the container relocation problem

```
nb_relocations ← 0
for i = 1 to W do
        determine s(i) using Equation (1)
end for
for t = 1 to T do
        determine W
        while ∃container above target container do
                n ← topmost relocation container
                determine $i^*$ for n using Equation (2.1)
                relocate container n to stack $i^*$
                nb_relocations ← nb_relocations +1
                //$s(i^*) \leftarrow \min\{s(i^*), n\}$
                $s(i^*) \leftarrow s(i^*) + n * ind(n)$
                update W
        end while
        retrieve target container from s′
        determine s(i′) using Equation (2.2)
end for
        return nb_relocations and executed retrievals and relocations
```

To better understand the HC adjusted how it works, we take an example , consider a small bay with three stacks and four tiers and the given initial configuration illustrated in Table 3.1. Each container number determines its scheduled retrieve time.

During the retrieval process, the relocations number is the total of the relocations number to retrieve all containers .

Initial bay configuration

0	0	0
6	7	3
9	1	2
8	5	4

t = 1

0	0	0
6	7	3
9	1	2
8	5	4

relocated 7

score(S0) = 44
score(S2) = 17
selected stack with
minimum score : S2

$\longrightarrow$

t = 2

0	0	7
6	0	3
9	1	2
8	5	4

retrieved 1

t = 3

0	0	7
6	0	3
9	0	2
8	5	4

relocated 7

t = 4

0	0	0
6	0	3
9	7	2
8	5	4

relocated 3

t = 5

0	0	0
6	3	0
9	7	2
8	5	4

retrieved 2

t = 6

0	0	0
6	3	0
9	7	0
8	5	4

retrieved 3

t = 7

0	0	0
6	0	0
9	7	0
8	5	4

retrieved 4

t = 8

0	0	0
6	0	0
9	7	0
8	5	0

relocated 7

t = 9

0	0	0
6	0	0
9	0	0
8	5	7

retrieved 5

t = 10

0	0	0
6	0	0
9	0	0
8	0	7

retrieved 6

t =11

0	0	0
0	0	0
9	0	0
8	0	7

retrieved 7

t = 12

0	0	0
0	0	0
9	0	0
8	0	0

relocated 9

t = 13

0	0	0
0	0	0
0	0	0
8	9	0

retrieved 8

t = 14

0	0	0
0	0	0
0	0	0
0	9	0

retrieved 9

t = 15

0	0	0
0	0	0
0	0	0
0	0	0

t = 6

Relocations number = 5

3.2. Results

The proposed Algorithm is coded in Python. Experiments are carried out on the instances introduced by W Zhu [21].

Each instance H-W-N describes the initial configuration of the H tiers and W

stacks and N containers, 50 instances taken for each one, in total 150. The maximum bay height is limited to H +1.. Results of this comparison are shown in Table 1. Column 1 identifies the instances. Column 2 shows Number of containers , columns 3 to 6 shows the results of different Heuristics applied in CRP problem, columns 3 shows the results of ‘’ Corona Virus Optimization Algorithm “ Proposed .

Average relocations						
Tiers * Stacks	**No of Containers**	**Zhang (2000) – TLP**	**Caserta et (2009)– min-max**	**ovanovic, (2014) –chain**	**Livia MAGLIĆ*, Marko GULIĆ, Lovro MAGLIĆ (2019)**	**Coronavirus optimization Algorithms Proposed**
3 * 6	16	8.95	7.72	7.72	**7.88**	8.0
3 * 7	19	11.50	9.02	9.05	9.15	9.48
6 * 4	19	23.20	17.15	17.05	16.93	18.60

The results illustrate that when the number of containers increased , the gap of the proposed optimization Algorithm compared to the Optimal result (Livia MAGLIĆ*, Marko GULIĆ, Lovro MAGLIĆ (2019)) increased also, but in reality the performance of the Algorithm doesn't decrease because the rate of the gap compared to the number of containers almost still the same even with the change of container's number.

3.3. Conclusion

in this chapter we worked with the static case of the Container relocation problem in a terminal, we proposed an adjustment for the Corona Virus Optimization Algorithm and proposed a Heuristic used in the Algorithm to minimize the total number of relocations to retrieve a given sequence of containers .and comparing the testing results with other Heuristic results applied on the CRP.

General Conclusion

In this work we proposed to solve Container Relocation Problem with Corona virus Optimization Algorithm, it divided on three chapters:

First chapter: it covered different aspects of Combinatorial optimization like Diversification and intensification, and complexity of optimization problems and optimization Algorithms. different methods of Solving optimization problems (exact methods, approximate methods), under approximate methods, Coronavirus Algorithm was taken.

Second chapter: we took the Container relocation problem CRP(Terminal structure and handling equipment, Problem description , CRP classifications, and finally Related work in ascending schedule)

Third chapter : we proposed contributions aim to adapting Coronavirus optimization Algorithm for solving the static version of container relocation problem(adjusting the Coronavirus algorithm, proposed an Heuristic used for selected the stack to relocate a container)

Because the big number of publications in the last years proves the importance of the solving methods performance, the main reason is that if the performance increases causes improving of economy of companies , and even countries and the decreasing of that performance leads the whole world to big problems , for that an extra work is needed , in our case we can work more on the ‘Select Stack ’ Heuristic proposed combined with updated studies of Coronavirus optimization Algorithm, we can also visit real Container terminals to get more clear conception of the problem.

References

[1]. Xin-She Yang1 , Suash Deb, Simon Fong. Metaheuristic Algorithms: Optimal Balance of Intensification and Diversification. Natural Sciences Publishing Cor , 2014

[2]. Fred Glover, Manuel Laguna Tabu Search. Springer New York, NY, 1997

[3] TALBI, El-Ghazali. Metaheuristics: from design to implementation. John Wiley & Sons, 2009.

[4]. Michael, Garey. David , Johnson. Computers and Intractability: A Guide to the Theory on NP-Completeness. W. H. Freeman and Co. Publishers, New York, 1979.

[5] Bianchi, Leonora; Dorigo, Marco; Gambardella, Luca Maria; Gutjahr, Walter J. A survey on metaheuristics for stochastic combinatorial optimization.Natural Computing . 239-287 (Jun 2009).

[6] Sean Luke, Essentials of Metaheuristics, Lulu, second edition, available for free at http://cs.gmu.edu/~sean/book/metaheuristics/ 2013

[7]. Thai My T. Journal of Combinatorial Optimization. Springer ISSN:1382-6905 2020

[8]. Alireza Salehan and Arash Deldari, Corona virus optimization (CVO): a novel optimization algorithm inspired from the Corona virus pandemic . J Supercomput. 2022

[9] Elisabeth Zehendner. Operations management at container terminals using advanced information technologies. Other. Ecole Nationale Supérieure des Mines de Saint-Etienne, 2013.

[10] Marco Caserta, Silvia Schwarze, Stefan Voss .A New Binary Description of the Blocks RelocationProblem and Benefits in a Look Ahead Heuristic. Conference Paper · April 2009

[11] V. Galle, V. H. Manshadi, S. Borjian Boroujeni, C. Barnhart, P. Jaillet .The Stochastic Container Relocation Problem. Transportation Science 52(5):1035-1058.https://doi.org/10.1287/trsc.2018.0828 .2018

[12] Andresson Silva Firmino, Valéria Cesário Times. chapter . A Coronavirus Optimization Algorithm for Solving the Container Retrieval Problem, Frontiers in Nature-Inspired Industrial Optimization. Springer Nature . 2022 ISBN : 978-981-16-3127-6

[13] Lei Wei Fuyin Wie Sandra Schmitz Kunal Kunal Bernd Noche. Optimization of Container Relocation Problem via Reinforcement Learning. Logistics Journal: Proceedings – ISSN 2192-9084 2021

[14] Ndèye Fatma Ndiaye. Algorithmes d'optimisation pour la résolution du problème de stockage de conteneurs dans un terminal portuaire. Mathématiques générales [math.GM]. Université du Havre, 2015

[15] Dirk Steenken, Stefan Voß, and Robert Stahlbock. Container terminal operation and operations research - a classification and literature review. OR Spectrum 26, 3–49. https://doi.org/10.1007/s00291-003-0157-z (2004).

[16] V. Galle; V. H. Manshadi; S. Borjian Boroujeni; C. Barnhart and P. Jaillet, The Stochastic Container Relocation Problem, Transportation Science, 52, (5), 1035-1058 . (2018)

[17] Consuelo Parreño-Torres, Ramón Alvarez-Valdés, Rubén Ruiz.Integer programming models for the pre-marshalling problem. Eur. J. Oper. Res. 274(1). 142-154 (2019)

[18] Maglic, Livia & Gulić, Marko & Maglic, Lovro OPTIMIZATION OF CONTAINER RELOCATION OPERATIONS IN PORT CONTAINER TERMINALS. Transport. 35. 1-11. 10.3846/transport.2019.11628. . (2019).

[19] Tiecheng Jiang et al A New Heuristic Reinforcement Learning for Container Relocation Problem. J. Phys.: Conf. Ser. 1873 012050. 2021

[20] El Majdoubi, Omayma & Abdoun, Farah & Otman, Abdoun. A New Optimized Approach to Resolve a Combinatorial Problem: CoronaVirus Optimization Algorithm and Self-organizing Maps. 10.1007/978-3-030-73882-2_86. (2021).

[21] Bo Jin, Andrew Lim , Wenbin Zhu . A Greedy Look-Ahead Heuristic for the Container Relocation Problem. In: Ali, M., Bosse, T., Hindriks, K.V., Hoogendoorn, M., Jonker, C.M., Treur, J. (eds) Recent Trends in Applied Artificial Intelligence. IEA/AIE 2013. Lecture Notes in Computer Science(), vol 7906. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-38577-3_19 . (2013)

[22] Hindriyanto Dwi Purnomo , Hui-Ming Wee . Soccer game optimization with substitute players . Journal of Computational and Applied Mathematics Volume 283 Issue C. https://doi.org/10.1016/j.cam.2015.01.008 . August 2015

[23] Marco Caserta, Silvia Schwarze, Stefan Voß,A mathematical formulation and complexity considerations for the blocks relocation problem,European Journal of Operational Research,Volume 219, Issue 1,2012