Genetic procedure for the Single Straddle Carrier Routing Problem

Khaled MILI
Applied Economics and Simulation,
High institute of management
2000 Bardo, Tunisia

Faissal MILI
Applied Economics and Simulation,
Faculty of Management and Economic Sciences,
5100 Mahdia, Tunisia

***Abstract*— This paper concentrates on minimizing the total travel time of the Straddle Carrier during the loading operations of outbound containers in a seaport container terminal. Genetic Algorithm is well-known meta-heuristic approach inspired by the natural evolution of the living organisms. Heuristic procedure is developed to solve this problem by recourse to some genetic operators. A numerical experimentation is carried out in order to evaluate the performance and the efficiency of the solution.**

Keywords- Component; Container terminal optimization; routing Straddle Carrier; heuristic; Assignment strategy; Genetic Algorithm.

I. INTRODUCTION

The success of a seaport container terminal resides in a fast transshipment process with reduced costs and it is measured by many performance indicators such as the productivity and the customer satisfaction. Because containerships are highly capitalized and their operating costs are very high, it is very important that their turn-around time in container terminals is as short as possible. The turn-around time of a containership implies the sum of the queue time and service time which is the sum of time for berthing, unloading, loading and departing.

Since inbound containers are usually unloaded into a designated open space, the yard handling equipment (i.e. Straddle Carrier) does not have to travel much during the unloading operation. However, the time for loading depends on the loading sequence of containers as well as the number of loaded containers. The Single Straddle Carrier Routing Problem (SSCRP) has a great effect on the time service and the performance of the container terminal. In this paper, our objective is to minimize the total travel time of the Straddle Carrier (SC) for loading outbound containers.

The remainder of the paper is organized as follows: section 2 present the related works that solve the SSCRP. Section 3 will be reserved to detail the problem formulation. The two following sections will be reserved to present our Genetic Algorithm solution and our paper will be finished by numerical examples in order to prove the efficiency of our method. Some concluding remarks and perspectives to extend this work are finally discussed.

II. RELATED WORKS

Operational decision problems on seaport terminals have received increasing attention by researchers. Some of them evoked the SCRP as Kim and Kim ([1] [2] [3]); they present in their papers a mathematical formulation for the SSCRP where their objective is to minimize the distance between yard-bays in the storage area during the SC tour. They propose a solution procedure based on beam search heuristic method that its principle is to select the nearest yard-bay to visit by a SC.

However, they suggest that the times spend by the SC inside yard bays are constant. In reality, we note that these times vary and significantly affect the SC routing tour. In fact, we judge that the reshuffling and the unproductive movements inside a yard bay depend on the positions of required containers into stacks and levels. Hence, we will evaluate the time spent between initial position and all destinations yard bays in addition to the time cost inside each one and we select the least of them. Therefore, in our paper we solve the SSCRP by considering this time (intra-yard bay) as variable.

Little research has been done on this topic although the practical importance of this problem ([4] [5] [6] [7] [8]).

In 2003, V. Nunes Leal Franqueira [9] proposed Heuristic strategies Beam Search and Ant Colony Optimization to solve the single vehicle routing problem. These are tested comparatively. A new strategy for container collection is proposed as a substitute for the traditional greedy strategy of container collection. The proposed strategy increased the number of alternatives considered within the search space and turned out to improve the quality of solutions. It is proved that the Ant colony heuristic react well as beam search method in this strategy.

In 2005, E. Nishimura et al. [10] present in their paper a Genetic Algorithm heuristic to solve the trailer routing problem using a dynamic routing assignment method. They focus on the tours related to one cycle operation of the quay cranes. Experimental results demonstrate that the dynamic assignment is better than static one. The drawback to their solution procedure is the complexity of the trailer routing, which may increase the possibility of human error. Trailer drives may find difficult to follow the complicated itineraries assigned to them, resulting in mistakes in driving.

E. Nishimura et al. in 2001 [11] address in their paper the problem of a dynamic assignment ships to berths. They develop a heuristic procedure based on genetic algorithm where chromosomes are presented as character strings instead of bit binary string representation. The chromosome representation of the solution defines the set assignments of

ships to berths, giving ships the positions of service sequence for each berth.

III. PROBLEM FORMULATION

A. Straddle Carrier definition

By a "subtour" of a SC, we mean a visiting sequence of yard-bays which a SC visits to pick up all the containers which will be loaded onto a cluster of cells in the ship. An overview of a container terminal is presented in Figure 1.

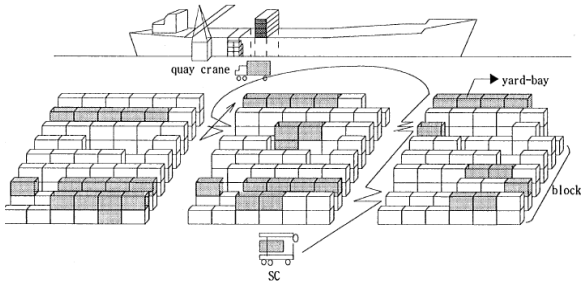

Figure 1. An overview of a container terminal

B. Optimization model

An optimization model will be developed to display the container arrivals and yard locations and the actual and optimized assignment of straddles to containers.

The main part of this modeling is to develop and evaluate the algorithms for assigning straddles to containers.

The discussion from the previous section illustrates the fact that the manner in which the straddles are assigned container jobs impacts the cost and service quality of operation.

In general, the problem of assigning straddles to containers can be formulated as the assignment problem, a mathematical programming problem presented by L.N. Spasovic et al. in 1999 [12].

$$Min\ Z = \sum_i \sum_j c_{ij} x_{ij} \qquad (1)$$

s.t

$$\sum_{i=1}^{n} x_{ij} = 1 \quad for \quad j = 1, 2, ..., n \qquad (2)$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad for \quad i = 1, 2, ..., n \qquad (3)$$

Where

i, j= indices

n = number of containers.

$$x_{ij} = \begin{cases} 1 \text{ if the feasible assignment of container "i" to straddle "j" is selected} \\ 0 \text{ otherwise} \end{cases}$$

$$c_{ij} = \cos t \text{ of the } (i, j) \text{ assignment}$$

Equation (1) is an objective function that minimizes costs. Constraints (2) and (3) are typical assignment problem restrictions that ensure that a straddle can be assigned to only one container and vice versa.

IV. GENETIC ALGORITHMS (GA)

A. Motivation

Because of existing the several equality constraints in our model, the implementation of the GA in order to quick and facilitate achieve to the feasible solutions is the objective of our work.

Our reasons for choosing GA as a solution approach are as follows:

We need an approach to search the feasible route. The most of the model's constraints are as equality form and, therefore, obtaining of the feasible solutions is a hard task. In this case, the probability of reaching infeasible solutions is more than feasible solutions and therefore we need a population-based approach such as GA to better exploration of the solution routing.

GA is a well-known meta-heuristic that its efficiency is verified for many problems in the literature.

GAs work on a population of the solutions simultaneously. They combine the concept of survival of the fittest with structured, yet randomized, information exchange to form robust exploration and exploitation of the solution routing [13].

B. Probleme outlines

The problem can be summarized as follow:

- Each feasible solution of the problem is treated as a chromosome in the GA.
- Every container must be picked up once and exactly once at any route.
- The handling time of each container is dependent on its positions in the storage yard.

The set of chromosomes construct a generation where steps of GA are applied:

Step1: A fitness function is applied to extract the fittest value of the generation

The SCRP is a minimization problem; thus the smaller the objective function is, the higher the fitness value must be.

Step2: The GA selects the fittest chromosomes and applied a crossover operator to give rise to better solution.

The crossover scheme should be capable of creating new feasible solution (child) by combining good characteristics of both parents.

Step3: These solutions are treated by a mutation operator which introduces random changes to the chromosomes to create new generation

C. Our representation

In our GA application, a candidate solution to an instance of the SCRP specify the number of required containers, the possible visited yard bays, the partition of the demands and

also the delivery order for each route. Each chromosome represents a feasible solution.

We will apply genetic operators to generate new individuals. Each of them defines a possible route of genes that represent the required containers.

For our problem a chromosome is a set of containers that can be visited by a SC to perform a quay crane (QC) work schedule. For example a QC demands r_t containers type A to load them into a containership, SC has to move toward the yard bays that include this type of containers, and transports them to the QC. Each container has a transportation cost depends on its position inside the storage yard and especially in its chromosome's order. The number of genes in a chromosome is bounded by r_t required containers. And the cost of the SC's tour will be the sum of transportation costs of the picked up containers.

The position is defined by the number of yard bay, stack and level. Every type or group is presented by a matrix. Each element of the matrix will be equal to 1 if the group of the current container is the same as the required type and 0 otherwise. Therefore we begin our procedure by this set of required containers from which we construct the initial generation of n chromosomes. The GA will choose randomly a set of chromosomes that contains r_t genes. The gene represents a required container.

Let $C^{i}_{s,l}$ designs the container inside yard bay *i* in stack *s* in level *l*; i.e. $C^{1}_{8,3}$ represents a container inside the first yard bay, in the third level of the 8th stack.

V. GENETIC OPERATORS

A. Fitness fuction

For our solution procedure we will take under consideration the sigmoid function as defined in E. Nishimura et al. in 2001 [11] ; where z(y) denotes the objective function value.

$$f(y) = 1/1 + \exp(z(y)/10.000)) \qquad 0 \le f(y) \le 0.5 \quad (4)$$

For the feasible solutions, our GA calculates the cost of each route that satisfies the objective function of the SSCRP. Then it compares between these costs and selects the smallest amount one.

B. Reproduction

It is a process in which chromosomes are copied according to their scaled fitness function values, i.e., chromosomes with a higher fitness value would have more of their copies at the next generation. This can be done by randomly selecting and copying chromosomes with probabilities that are proportional to the fitness values (costs of routes presented in each chromosome).

1) Initial Situation: We have ***n*** genes representing the number of the available required containers.

Example: for n=9

$c^1_{1,1}$	$c^1_{6,1}$	$c^1_{9,1}$	$c^2_{3,1}$	$c^2_{4,2}$	$c^4_{4,3}$	$c^4_{9,3}$	$c^5_{4,1}$	$c^5_{5,2}$

Figure 2. list of required containers

2) Initial generation: GA chooses randomly r_t genes from this table, constructs generations of chromosomes and selects the two fittest ones from each generation.

For our example, let chromosomes ***A*** and ***B*** be the selected parents.

A:	$c^2_{4,2}$	$c^4_{4,3}$	$c^4_{9,3}$	$c^1_{6,1}$	$c^1_{9,1}$	$c^5_{4,1}$	$c^2_{3,1}$
B:	$c^4_{9,6}$	$c^5_{4,1}$	$c^1_{6,1}$	$c^1_{9,1}$	$c^2_{3,1}$	$c^2_{4,2}$	$c^4_{4,3}$

Figure 3. Parent's chromosomes

C. Crossover

We use the 2-point crossover to introduce new chromosomes (or children) by recombining current genes. In this crossover, two cut points are randomly chosen on the parent chromosomes. In order to create an offspring, the strings between these two cut points in both chromosomes are interchanged.

A crossover may generate infeasible children in terms of constraint, i.e., a child chromosome may have container to be picked up twice. In order to keep the feasibility, the crossover operation is performed in the following manner.

First, substrings to be interchanged are given by two crossover points that are randomly determined. After interchanging the substrings between chromosomes ***A*** and ***B***, we have chromosomes ***A′*** and ***B′*** as temporary children.

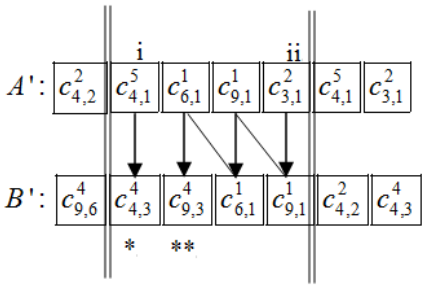

Figure 4. Temporary children

They are infeasible, because for instance, chromosome ***A′*** has containers $C^{5}_{4,1}$ and $C^{2}_{3,1}$ to be picked up twice and lose containers $C^{4}_{4,3}$ and $C^{4}_{9,3}$. Next, additional interchanges described below are carried out to make them feasible. Letting the interchanged string be the substring that was interchanged so far in each chromosome, we examine genes from left to right in the interchanged string of ***A′***. Note that we never interchange any genes in the interchanged string again in the following process:

As the most left gene is container $C^{5}_{4,1}$ which is scheduled to be picked up twice in ***A′***, obtain the container type (which is $C^{4}_{4,3}$) in ***B′*** at the corresponding position of the gene (hereafter referred to as a cell).

Therefore the container $C^{5}_{4,1}$ placed in the 7th bit in the chromosome ***A′*** will be replaced by $C^{4}_{4,3}$.

The second double container in chromosome ***A'*** is $C^{2}_{3,1}$.

When we look at the corresponding gene in chromosome ***B'*** we find the container $C^{1}_{9,1}$ which already exists in ***A'***, so we follow the interchange arrows until we reach its end. In our example we stop in the container $C^{4}_{9,3}$ which will replace $C^{2}_{3,1}$ in chromosome ***A'***.

We look at chromosome ***B'*** and we follow the same way as in ***A'***.

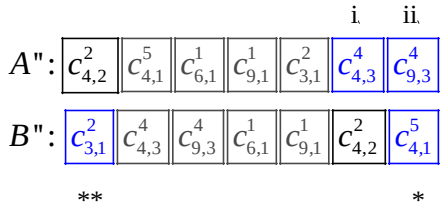

Figure 5. children after crossover

D. Mutation

This genetic operator introduces random changes to the chromosomes by altering the value to a gene with a user-specified probability called mutation rate. As an example we choose to apply the swap operator to each gene. This operator consists in randomly selecting two genes and exchanging them.

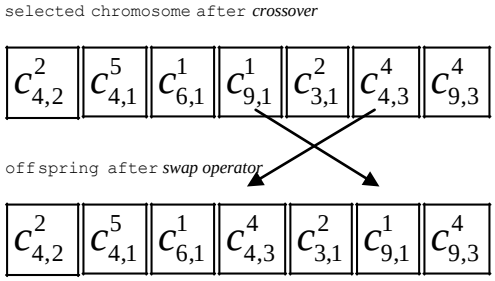

Figure 6. Swap operator

VI. Solution Heuristic Procedure

At really quay crane work schedule, many cases can be found. In this paper the 'single tour' and the 'subtours' cases are chosen.

Case 1: single tour (where containers of the same type are required)

1. We have initial situation with all available required containers
2. Repeat

- Create a generation in which all chromosomes have r genes where r is equal to the number of required containers.
- Calculate the objective function for each individual
- Transform it to a fitness value
- Select best-ranking individuals to reproduce
- let individuals having better fitness be new parents
- Create offspring through crossover operator
- Evaluate the individual fitness of the offspring
- Select the chromosome having the biggest fitness value.
- Apply mutation operator on the selected chromosome.
- Select the chromosome having the best solution

Until the number of required generations is reached

3. Select the best solution from all resulting chromosomes

Case 2: subtours (many subtours to perform)

Let ***t*** design a number of subtour {***t = 1...T*** }

Example:

(***t=1***) first subtour to transport containers type ***A***

(***t=2***) second subtour; transport containers type ***B***.

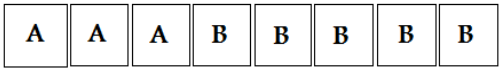

Figure 7. chromosome of two subtours

The genetic procedure for case2 is presented as follow:

Step1**:** For ***t = 1*** (first subtour)

- let r_1 containers (genes) from a type ***h***
- apply genetic algorithm developed in case 1

Step2: For subtours *t = {2 ...T}*

- start from the last straddle carrier's position of the resulting chromosome in step1
- Apply genetic algorithm developed in case 1

VII. Numerical Examples

The resulting solutions of two strategies (Greedy collection strategy and Random collection strategy) are used as references for our experiments. The Greedy strategy is the collection of the maximum quantity of containers available at a determined location. The Random strategy is the collection of a random quantity of containers without leaving behind containers needed to complete a work schedule subtour. A MATLAB program was used to solve the above mixed-integer programming for an example problem. The MATLAB is a programming environment for algorithm development, data analysis, visualization, and numerical computation. All experiments were performed on a 3GHz Pentium 4 computer.

In the following tables the presented solutions of the GA procedure are the best ones between 12 generations for every iteration.

Case **1:** in this case containers of the same group (A) are required. Two instances are generated in order to evaluate the GA procedure.

TABLE I. Instances (case1).

	Instance 1	**Instance 2**
Type of required containers	***A***	***A***
Number of required containers	10	70

Number of available containers	66	215
Number of yard bays	6	20
Number of 3 level's stacks	10	15

TABLE II. GENETIC ALGORITHM PROCEDURE (GA) WITH THE GREEDY COLLECTION STRATEGY (GCS) AND RANDOM COLLECTION STRATEGY (RCS): COSTS AND PROCESSING TIMES (IN CASE1).

	Iterations		1	2	3	4	5
Instance1	Costs (sec)	GCS	1138.7				
		RCS	1138.7	1180.0	1179.3	1138.7	1180.0
		GA	694.4	747.6	672.8	667.0	695.1
	Processing times (sec)		0.85	0.94	1.08	0.52	0.78
Instance2	Costs (sec)	GCS	8269.8				
		RCS	7860	8503.1	8980.4	8503.8	7649
		GA	7835.8	7937.8	7646.3	8819.1	8891.1
	Processing times (sec)		32.4	31.3	33.67	32.13	34.63
	Iterations		6	7	8	9	10
Instance1	Costs (sec)	GCS	1138.7				
		RCS	967.8	1179.3	1140.9	1179.3	1138.7
		GA	684.4	798.0	680.7	721.8	741.8
	Processing times (sec)		0.85	0.55	0.75	1.33	1.34
Instance2	Costs (sec)	GCS	8269.8				
		RCS	8912.6	7845.6	8579.7	8093.1	8473.2
		GA	8265.9	8254.6	8293.1	8959.8	8263.5
	Processing times (sec)		32.4	32.15	33.48	31.54	32.44

Case 2: in this case containers of the different groups are required. Two instances are generated in order to evaluate the GA procedure.

TABLE III. INSTANCES (CASE2).

	Instance 3		**Instance 4**	
Type of required containers	***A***	***B***	***A***	***B***
Number of required containers	10	8	75	60
Number of available containers	66	68	215	240
Number of yard bays	6		20	
Number of 3 level's stacks	10		15	

TABLE IV. GENETIC ALGORITHM PROCEDURE (GA) WITH THE GREEDY COLLECTION STRATEGY (GCS) AND RANDOM COLLECTION STRATEGY (RCS): COSTS AND PROCESSING TIMES (IN CASE2).

	Iterations		1	2	3	4	5
Instance1	Costs (sec)	GCS	1659.9				
		RCS	2168.3	1733.3	2202.4	1944.8	1999.3
		GA	1770.1	1398.9	1486.6	1698.1	1633.4
	Processing times (sec)		0.85	2.5	2.42	0.39	0.45
Instance2	Costs (sec)	GCS	12916				
		RCS	14175	14115	13396	15249	14691
		GA	13931	17668	17051	14391	16931
	Processing times (sec)		32.4	75.0	69.5	59.8	72.1
	Iterations		6	7	8	9	10
Instance1	Costs (sec)	GCS	1659.9				
		RCS	1944.8	1684.8	1946.3	1724.6	1800.6
		GA	1839.9	1786.0	1417.6	1734.9	1703.8
	Processing times (sec)		0.85	0.5	1.51	2.30	1.95
Instance2	Costs (sec)	GCS	12916				
		RCS	15421	13886	15190	15335	14968
		GA	15003	14372	14817	14834	15173
	Processing times (sec)		32.4	62.9	68.7	70.4	75.1

By analyzing the results in table II and table IV the following remarks can be made:

- In instance 1, Genetic algorithm demonstrated to return solutions of better quality than the Random collection strategy and the Greedy collection approach, within the same range of the processing time.
- Due to the random nature of Genetic algorithm heuristic method, increasing the size of the problem in instance 2 to look for solutions does not necessarily guarantee that a better solution will be reached. As observed in iterations 5 and 7, Random collection strategy found a better quality solution.
- By looking at iterations 4 and 8 in instance 2, we notice that Greedy collection strategy worked better since it found the minimum costs. It can be explained by the fact that the genetic heuristic selects the next location based on container's positions, i.e. both the yard-bay and the position of containers (stacks and levels) are taken into account, while the Greedy collection strategy selects the next location based on the yard-bays only.
- It seems that increasing the number of subtours has the same effect as increasing the number of the required containers.
- In big size problem, genetic procedure is less performed, this is can be explained by the fact that at each generation built by genetic procedure, a number of selected containers is fixed and the genetic operators are used only in this set of containers in order to choose the best cost solution between them.
- Genetic algorithm demonstrates better solution sin small sizes problem.
- The processing time in all iterations is quite reasonable.

VIII. CONCLUSION

In this paper, we have presented the routing problem of the SC to load outbound containers. We use Genetic Algorithm as an approach to solve the SSCRP, we defined its utility, and we developed our Genetic heuristic procedure and its operators such as fitness function, reproduction, crossover and mutation in order to get best solution to choose a route for the SC which has the least cost. Numerical examples are carried out to prove the performance and the efficiency of our method comparing with greedy and random collection strategies.

IX. FUITURE WORK

In a real container terminal there is more than one quay crane, with their respective work schedules, and many Straddle Carriers (SCs). Several SCs must complete their routing, by sharing the same container yard-map. This new scenario introduces a few complications to the single SC routing problem, increasing significantly the routing problem

complexity. Our future work aims to solve the multiple straddle carrier routing problem (MSCRP) in a container terminal. In Other metaheuristics can be applied in the future, such Ant colony and particle swarm optimization, and comparative study can be done.

REFERENCES

[1] K. H. Kim, K. Y. Kim, A routing algorithm for a single transfer crane to load export containers onto a containership , Computers & Industrial Engineering 33 (1997) 673- 676.

[2] K. H. Kim, K. Y. Kim, Routing straddle carriers for the loading operation of containers using a beam search algorithm, Computers & Industrial Engineering 36 (1999) 109-136.

[3] K. H. Kim, K. Y. Kim, A routing algorithm for a single straddle carrier to load export containers onto a containership, Int. J. Production Economics 59 (1999) 425- 433.

[4] DW. Cho, Development of a methodology for containership load planning. Ph.D. thesis, Oregon State University, 1982.

[5] YG. Chung, SU. Randhawa, ED. McDowell, *A simulation analysis for a transtainer-based container handling facility*, Computers Ind Eng 14(2) (1988) 113±25. [

[6] F. Soriguera, D. Espinet and F.Robuste, A simulation model for straddle carrier operational assessment in a marine container terminal, Journal of Maritime Research 2006.

[7] K. M. Van Hee, R.J. Wijbrands, *Decision support system for container terminal planning*, European Journal of Operational Research 34 (1988) 262–272.

[8] C. Zhang, J. Liu, Y.-W. Wan, K.G. Murty, R.J. Linn, Storage space allocation in container terminals, Transportation Research. Part B: Methodological 37 (2003) 883– 903.

[9] V. Nunes Leal Franqueira, Single Vehicle Routing in Port Container Terminals, Master thesis, Universidade Federal do Espirito Santo, Brazil. August 2003.

[10] E. Nishimura, A. Imai, S. Papadimitriou, *Yard trailer routing at a maritime container terminal*, Transportation Research Part E: Logistics and Transportation Review, Volume 41, Issue 1, (2005) 53-76.

[11] E. Nishimura, A. Imai, S. Papadimitriou, Berth allocation in the public berth system by genetic algorithms, European Journal of Operational Research 131 (2001) 282– 292.

[12] L N. Spasovic, et all, "Increasing productivity and service quality of the straddle carrier opérations at container port terminal", Journal of Advanced Transportation October 20, 1999.

[13] M. Bazzazi, , N. Safaei, , N. Javadian, "A genetic algorithm to solve the storage space allocation problem in a container terminal," Computers & Industrial Engineering, vol. 56, pp. 44-52, April 2009.

AUTHORS PROFILE

Khaled MILI

Ph. Doctor in quantitative methods,

A member of Loboratory of operational research (LARODEC),

High institute of management

E-Mail: khaledmili@yahoo.fr

Phone: 00216 52 181 176

Faissal MILI

Ph. Doctor in quantitative methods,

A member of Applied Economics and Simulation laboratory,

Faculty of Management and Economic Sciences, 5100 Mahdia, Tunisia

E-Mail: milisoft@yahoo.fr

Phone: 0021620545728