A Data-Driven Genetic Programming Heuristic for Real-World Dynamic Seaport Container Terminal Truck Dispatching

1st Xinan Chen
School of Computer Science
The University of Nottingham Ningbo China
Ningbo, China
Xinan.Chen@nottingham.edu.cn

2nd Ruibin Bai
School of Computer Science
The University of Nottingham Ningbo China
Ningbo, China
Ruibin.Bai@nottingham.edu.cn

3rd Rong Qu
School of Computer Science
University of Nottingham
Nottingham, UK
Rong.Qu@nottingham.ac.uk

4th Haibo Dong
School of Computer Science
The University of Nottingham Ningbo China
Ningbo, China
Nick.Dong@nottingham.edu.cn

***Abstract*—International and domestic maritime trade has been expanding dramatically in last few decades, seaborne container transportation has become an indispensable part of maritime trade efficient and easy-to-use containers. As an important hub of container transport, container terminals use a range of metrics to measure their efficiency, among which the hourly container throughput (i.e., the number of twenty-foot equivalent unit containers, or TEUs) is the most important objective to improve. This paper proposes a genetic programming approach to build a dynamic truck dispatching system trained on real-world stochastic operations data. The experimental results demonstrated the superiority of this dynamic approach and the potential for practical applications.**

***Index Terms*—container terminal, genetic programming (GP), truck dispatching, dynamic**

I. Introduction and Problem Description

Global maritime trade expanded at a fast pace in the last few decades, and it comprises over 80% international trade now [1]. Therefore, as one of the major transportation hubs in maritime trade, container terminals are becoming more and more competitive, especially among terminals with similar geographical locations. This kind of competition is multi-dimensional, not only in absolute volume but also in work efficiency. Most shipping companies prefer to choose terminals with shorter average vessel stay time, which directly impacts their costs. In order to gain competitive advantages, terminals are extremely motivated to shorten the ship dock time.

Once a ship arrives, the container terminal unloads the containers to the storage yard and loads the export containers onto the ship from the storage yard. This operation time, plus any necessary waiting time, almost equals to ship stay time which most terminals want to reduce. In loading and unloading operation, QCs (Quay Cranes) are responsible for loading and unloading containers on the quayside; container trucks transport containers between quayside and yards as transport equipment; YCs (Yard Cranes) complete the loading and unloading of containers on yard areas (temporary storage space in the container terminal), thus forming a series of internal container transport operations including loading and unloading between the quayside and container yards. It is obvious that the more equipment devices participate in the operations, the less total operation time is required. However, with the influence of marginal decreasing and geographical limitation, such kind of strategy is difficult to take effect nowadays. Therefore, the container terminal must optimize the efficiency of existing resources to improve their productivity further [2].

Generally, there are three primary schedulable resources in container terminals, which are QCs, YCs, and trucks corresponding to three major types of container terminal optimization problems: QC assignment problem, YC routing problem, and truck dispatching problem. All of these there problems aiming to improve the operation efficiency of the container terminal through enhancing the utilization rate or working speed of corresponding equipment. However, different from the other two problems, because the operation of QCs and YCs requires the assistance of trucks, the solution of truck dispatch problems can also reduce the idle time of QCs and YCs and improve their operational efficiency.

Unfortunately, the traditional method that statically assigns trucks to the QCs (i.e. dedicated truck assignments to each QC) with some manual fine-tuning will enormously reduce operation efficiency since trucks are not able to take better-coordinated tasks available from other QCs. In other words, to finish tasks of binding QCs, trucks need to choose follow-up

This work is supported by the National Natural Science Foundation of China (Grant No. 71471092), Natural Science Foundation of Zhejiang Province (Grant No. LR17G010001) and Ningbo Municipal Bureau of Science and Technology (Grant No. 2017D10034) and Ningbo Port Co., Ltd.

tasks from these QCs instead of selecting more appropriate tasks of other QCs [3]. For instance, after transiting the container to binding QC, the truck must idle to the yard area to pick up next container of binding QC. While the better choice is moving to nearby QC and load another container to the yard area. Such strategy will not only increase trucks' no-load mileage but also cause QCs and YCs waiting and significantly affect their work efficiency. Because of this, to improve the utilization, boost operation efficiency of terminal equipment, reduce no-load truck moving distance, and shorten ship waiting time, this paper introduces a novel system that can dispatch trucks dynamically by real-time work instructions.

Work instruction (see an example in Table I) contains tasks to complete in the whole container terminal by sequence. Each line of work instruction is a task, and it includes the task identifier, the container identifier, the source crane, the destination crane, the operating type, the container size, and the container weight. Usually, tasks in a work instruction need to be completed in a fixed order because of various constraints related to the ship loading plans, container weights, etc. Also, as shown in the TEUs column in Table I, containers are divided into standard 20-foot containers and 40-foot large containers. Container trucks can take one large container or two standard containers at a time. Therefore, if a truck has loaded a small container, it can take another standard container before being dispatched to the destination crane to avoid the waste of truck resources.

TABLE I
EXAMPLE OF WORK INSTRUCTIONS

ID	ContainerID	Src	Dst	Type	TEUs	Ton
1731	FCIU3705890	CR12	J4	DSCH	2	17
1287	ECMU9249162	CR7	Q2	DSCH	1	23
137	NYKU2797417	Q5	CR7	LOAD	2	15
1514	TCLU5546292	F3	CR15	LOAD	1	19

According to the requirements and situations described above, a dispatching system flow chart is given in Figure1. When the system starts, it will read truck information to the vehicle pool, and then dynamically dispatches tasks to trucks according to the dispatching algorithm. After tasks are assigned to trucks, truck information in the vehicle pool will be updated. At the algorithm part of this dispatching system, as this is a dynamic dispatching system that needs to dispatch trucks in 2 seconds (based on the requirement of container terminal company), it is difficult to use the traditional static truck dispatch algorithm which needs several seconds to calculate a solution. Heuristic algorithms were introduced in this paper since they can obtain dispatching solutions immediately, provide better comprehensibility for operators, and tolerate real-world stochastic situations by abstracting problems and solving them in higher dimensions.

Unlike other research works which aim to find best priory route combination in specific tasks sequence and operate environment, the system proposed in this paper does not guarantee optimality. Instead, it provides a dispatching solution dynamically with proper consideration of all main equipment's work conditions within 2 seconds. Due to the impact of real-time tasks updating and various other emergencies in real container terminals, such kind of dynamic dispatching algorithms are more suitable to adjust unpredictable production environment through incorporating real-time environment parameters into heuristics. Moreover, heuristic-based dynamic dispatching system is also able to simulate human operation by converting operators' operational experience into heuristics [4].

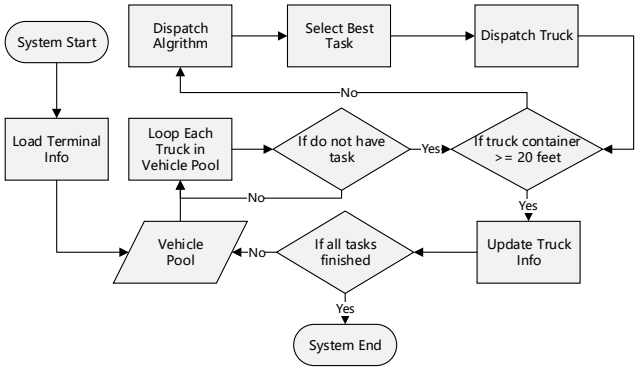

Fig. 1. Dispatching System Flow Chart

Owing to the above advantages of dynamic heuristic algorithms, this paper proposes two completely different types of heuristics to act as a dispatch algorithm in a dynamic dispatching system. One is a manually crafted heuristic by summarizing the accumulated experience and rules from container terminal operators. The other is a data-driven GP (Genetic Programming) heuristic trained by real operation data of a container terminal. We deduced that although operators' practical experience can be collected and transformed into heuristic to improve container terminal work efficiency, these experiences still have some limitations, especially in some uncommon conditions. In contrast, GP can obtain better heuristics through training individuals on numerous real-life data. The proposed GP trained heuristics can consider simultaneously equipment's working conditions and some extra features that may be omitted in operators' experience and adapt to particular conditions that operators' experience can not handle adequately. This hypothesis was verified though test results of different algorithms in Section V.

II. LITERATURE REVIEW

A typical container terminal has three major parts: berth area, yard area, and in & out area. It includes QCs, YCs, and trucks as three main schedulable resources work in those areas [5]. Research has been focusing primarily on improving container terminal efficiency by optimizing these three resources more rationally. Basic information and overview about container terminal optimization problem can be found in [6]–[14].

QCs are located in the berth area, once vessels arrived QCs will start to load or unload containers from or to trucks

under them. Generally, tracks will be laid under QCs to allow QCs to move laterally, but not to across berths nor other QCs. Because QCs are equipment that can directly operate containers on the ship, since 1985 Paul et al. [15] have tried to find an optimal combination of crane and berth to minimize QC makespan. Kim and Park [16] regarded QCs as the most crucial equipment in container terminal and proposed a branch and bound (B & B) method to schedule QCs by considering the nonsimultaneous constraints between adjacent berths as the safe distance of QCs. Though the comparison between the greedy randomized adaptive search procedure (GRASP) method and B & B method, B & B method shows better performance in small-scale questions. Kaveshgar et al. [17] further considered the time availability of the QC operation and introduced a new question named quay crane scheduling problem with time windows (QCSPTW). A GA (Genetic Algorithm) heuristic was applied to solve this new question with a broad set of numerical experiments using as benchmark instances. The outcome shows that the GA method is not only faster than other methods but also leads to an improvement in the solution quality in QCSPTW questions. Many other related studies have proposed different kinds of new algorithms to reduce the makespan of QCs [18]–[23], but most of these researches consider the scheduling of bridge crane and the actual tasks separately.

YCs are located behind the berth area, at the yard area of the container terminal. Yard area is the most essential part of container terminal which can store a large number of containers temporarily until they are loaded on the vessel or transported out to container terminal by trucks. Although YCs will not transport containers to/from vessel directly, as most crucial transit center for loading and unloading containers, if YCs can not store or take out the container immediately, it will cause trucks jam in the yard area, which will lead to no truck supply for QCs and increase the makespan of whole loading and unloading operation. After Lai et al. [24] summarized the several yard crane schedule module, Zhang et al. [25] and Cheung et al. [26] have proposed static gantry crane scheduling models with tasks known in advance. They have demonstrated their model and gave a reasonable static solution of YC routing. While Kim et al. [27] simplified the optimization of YCs efficiency to find the minimum moving distance of YCs, and have built a mixed-integer optimization model to minimize the moving distance of YCs in the yard. Chen et al. [28] have formulated a YC scheduling problem as a hybrid flow shop scheduling problem with precedence and Blocking constraints (HFSS-B) and introduced a tabu search algorithm to solve it. Under the constraint that the YCs cannot interfere with each other, Javanshir et al. [29] have tried GA heuristics on this problem. Besides scheduling YCs to improve the efficiency of the yard, optimizing the storage location of containers can also reduce the time to access the containers and improve the turnover efficiency of the yard. Lee et al. [30] proposed a novel approach that integrates yard truck scheduling and the storage allocation two problems and tried a hybrid insertion algorithm to minimize total delay of requests and the total travel time of yard trucks.

Trucks are a kind of mobile resource without a fixed location, and they are responsible for the transportation of containers between QCs and YCs, including some inbound and outbound containers. Despite the fact that trucks do not work at a particular place, they usually gather at a stop station at the beginning or ending of a shift. Since the work of QCs and YCs needs the support of trucks, optimizing the dispatches of trucks is another popular research area. The first piece of relevant research is from Dantzig et al. [31] on the routing of trucks in a bulk terminal. However, modern container terminals generally have constant driving routes for trucks. Containers in container terminals are transported according to these routes. Consequently, research about trucks dispatching problems in modern container terminals are more concerned with truck dispatching strategies or policies than trucks routes. Lu et al. [32] formulated a min-max nonlinear integer programming model to dispatching yard trucks. A two-phase heuristic method was applied in their model and obtained better results than the closest position assignment principle which is better in solving larger-scale problems. Cao et al. [33] analyzed the problem of truck scheduling and configuration, and established an integer programming model with the objective of searching the shortest waiting time of trucks. They tried genetic algorithm and heuristic method to solve the optimization model. Their final results showed that the heuristic method is more suitable for the model. Bish et al. [34] targeted the minimization of the total time required to serve a ship and developed heuristic algorithms that are easy to implement. Through numerical experiments, they showed that these algorithms could get results close to the optimal solution. Beyond these traditional operation research methods, genetic algorithms have been applied to solve truck dispatching problems in a container terminal in [35]–[38]. Above research studies have proposed algorithms with good performance which can improve the operation efficiency of a terminal to a certain degree, and/or reduce the waiting time of ships. After all, they still have some limitations in their algorithms. For example, the existing GA methods are used to produce solutions for predefined particular scenarios but they are not able to generate solutions or solutions methods that can adapt to different problem solving environments.

Although many pieces of research works have reported improved the efficiency of container terminals from several different aspects, they focused more on the improvement of locally isolated problems. For example, few research studies on truck dispatching consider the handling capacity at QCs and YCs, which will determine whether the trucks are congested in QCs and YCs or not. Equipment in container terminals needs to cooperate when dealing with tasks in work instruction when each operation node at QCs and YCs is subject to capacity constraints. This means the conditions at other operation nodes should be considered when optimizing operation at a given operation node. Furthermore, most researches are based on the current state to give a static solution, each time the port situation changes the solution needs to be recalculated, which

make it hard to be adopted to application environments that are continuously changing.

To resolve the shortcomings of previous researches, this paper introduces a GP heuristic to calculate dispatching solutions base on the current situation of container terminals dynamically instead of generating a static dispatching scheme. To improve the coordination among different operation nodes in container terminals, we designed some innovative parameters in Section IV such as $powt, powd, remain\_tasks$, etc. to capture the working states of different devices in container terminals. Heuristics can generate dispatching schemes according to the real-time values of these parameters to ensure that the QCs and the YCs have sufficient truck supplies for interrupted working. Moreover, in GP heuristics, we abandon the traditional objective function, which is mainly based on the driving distance and no-load distance of trucks, and used the number of standard containers handled per hour (TEUs/hour) as the optimization objective. This enables trucks to choose some distant destinations with long no-load mileages but lacking truck supply. The description of the principles and structures of these heuristic algorithms and their performance in test sets will be given in the following section.

III. Problem Description and Formulation

The main goal of the problem under consideration is to improve the efficiency of container terminal operation by reducing the ships' waiting time and improving the turnover. The specific objective of this study is to minimize the overall finishing time of all tasks.

Denote Q and Y the sets of all QCs and YCs, respectively, in the container terminal, and let $C = Q \cup Y$. V represents the set of m trucks available for assignment in the container terminal, and their loading/unloading positions shall be selected from C. Function $t(x, y)$ calculates the travel time between two operation points x and y according to the actual terminal road network. Let d be the depot of all trucks. At the beginning of the operation, d is generally the default position that trucks start from, and after all tasks are completed, trucks will return to d. Loading, unloading, and moving of containers are three different types of operations. Trucks transport containers between YCs and QCs, and in some cases, between YCs to YCs. After receiving its task, each truck will go to either a QC or a YC depend on the task type (ship loading or unloading). Upon completion of the task, the truck may wait or execute a next task. The work instruction set includes all n tasks represented as $W = \{w_1, w_2, w_3, \ldots, w_n\}$. Source and destination locations for each task i are denoted by a_i and b_i respectively, and $a_i, b_i \in C$. Denote s_i the start time that task i is serviced at its source node and e_i the completion time at its destination node. $S = \{s_1, s_2, s_3, \ldots, s_n\}$ and $E = \{e_1, e_2, e_3, \ldots, e_n\}$. Because at both the start and end of a task, a crane is needed to either load or unload the container, two functions $sct(w_i), ect(w_i)$ are used to query the operation time of task i at these two positions. The operation times at the source and destination nodes are stochastic and are drawn from the given probability distributions. Whether a task w_i is bound to crane c_j is denoted by a binary indicator given in (1) which is given as inputs of the problem.

$$\gamma(w_i, c_j) = \begin{cases} 1 & a_i = c_j \\ 0 & a_i \neq c_j \end{cases} \tag{1}$$

The assignment relationship between tasks and trucks is represented by (2).

$$\alpha(v_i, w_j) = \begin{cases} 1 & w_j \text{ is assigned to } v_i \\ 0 & w_j \text{ is not assigned to } v_i \end{cases} \tag{2}$$

Equation (3) shows if w_k is serviced immediately after task w_j by v_i.

$$\beta(v_i, w_j, w_k) = \begin{cases} 1 & w_k \text{ is next to } w_j \\ 0 & w_k \text{ is not next to } w_j \end{cases} \tag{3}$$

The truck dispatching optimization is to allocate all tasks in W to different trucks while maximizing the throughput (i.e. TEUs/hour) of the container terminal. Combined with the above definitions, the truck dispatching model can be expressed:

$$\max \frac{n}{\max E - \min S} \tag{4}$$

$$\sum_{i=1}^{m} \alpha(v_i, w_j) = 1 \quad \forall w_j \in W \tag{5}$$

$$\sum_{i=1}^{m} \sum_{k=1}^{n} \beta(v_i, w_j, w_k) \leq 1 \quad \forall w_j \in W, w_j \neq w_k \tag{6}$$

$$t(x_i, y_j) > 0 \quad \forall x_i \neq y_j \in (C \cup d) \tag{7}$$

$$s_i = \max \begin{cases} \sum_{j=1}^{n} \sum_{k=1}^{m} \beta(v_k, w_j, w_i) \cdot (t(b_j, a_i) + e_j) \\ t(d, a_i) \cdot (1 - \sum_{k=1}^{m} \beta(v_k, w_j, w_i)) \end{cases} \tag{8}$$

$$e_i = s_i + t(a_i, b_i) + sct(w_i) + ect(w_i) \quad \forall\ i \in [1, n] \tag{9}$$

$$\gamma(w_i, c_j) \cdot (s_i + sct(w_i)) \leq \gamma(w_{i+k}, c_j) \cdot (s_{i+k}) \quad \forall k \geq 1, \forall w_i, w_{i+k} \in W, \forall c_j \in C \tag{10}$$

Formula (4) is the objective function to maximum TEUs/hour by minimizing the difference between the end time of the last completed task and the starting time of the first started task among fixed number of tasks n. Constraint (5) ensures that each task is assigned exactly to one truck. Constraint (6) ensures that each task is followed by maximum one other task or nothing if it is the last task. Constraint (7) that the travel time on the map include the stop station d is greater than 0. Formulas (8) and (9) calculate task start time and end time. For each crane, due to the container terminal transportation rules, constraint (10) make sure that tasks start until previous tasks are completed.

IV. METHODOLOGIES

This section describes the design of the heuristic-based dynamic dispatching system as well as heuristic truck dispatching algorithms. As mentioned above, most of the researches about container terminal truck dispatching problem have excellent performance in experimental environment, but they are normally deterministic algorithms, which assume a deterministic problem environment. Such solutions did not fit real-life container terminals working environment well since container terminals' environment are changing frequently. These changes include changes in tasks' destination requirements and sequence, cranes' operation time, and trucks' travel time as well as trucks' availability. For these deterministic methods, if the environment has changed, the solution must be re-computed to guarantee the feasibility. Otherwise, old solutions generated based on the previous environment may cause unreasonable decisions, and result in truck congestion as well as long crane waiting time. Additionally, time required to re-compute a new solution usually exceeds the computing time restriction in container terminal companies.

A. Proposed Dynamic Dispatching System

Consequently, we propose to use dynamic algorithms to dispatch trucks base on the actual situation of the container terminal in real-time. As it has been shown in Figure 1, a dynamic dispatching system has been developed. This system will automatically loop all available trucks in the container terminal vehicle pool then send instructions to idle trucks. Trucks will be dispatched to most appropriate cranes measured by a utility function (which include heuristics, see details in later subsections) and the first available task of the dispatched crane is executed to ensure that tasks associated with cranes can be completed in the predefined sequence. The core algorithm of the dynamic dispatching system is shown in Algorithm 1.

B. Manual crafted Heuristic based on Experience

Currently, most container terminals bind trucks to cranes statically to complete corresponding tasks and let operators optimize these dispatching schemes. Even if this most basic truck dispatching method is adopted, under the optimization of experienced operators, many container terminals can still maintain high operation efficiency, Which shows that the rules summarized by these operators base on their historical work experience in container terminal can effectively improve the container terminal operation efficiency. Through the communications and discussions with these experienced operators, we transformed the rules summarized by operators into a heuristic algorithm. Moreover, this algorithm is inserted into the dynamic truck dispatching system mentioned above to simulate solutions made by operators and work as a control group. The structure of manual heuristic is shown in the Algorithm 2.

In this manual heuristic algorithm, some user parameters are used, such as $desired\_trucks$, which means the most suitable number of trucks for all working cranes, and $powt\_limit$ means the maximum number of trucks assigned to any work crane. These user parameters are based on the actual needs of the container terminal and generated by the statistics result of the operators' records. Other variables are calculated in real-time like $crane.powa$, which indicates the available number of trucks that have already arrived at the crane, as well as another parameter $crane.powt$ which is the total number of trucks working for the crane. On account of the needs of the operation in the real container terminal, each crane will be set a corresponding priority $crane.priority$, and the algorithm will give priority to the crane with a higher $crane.priority$ value. The function t can calculate the travel time that a truck takes from its current position to the crane. When the algorithm is running, real-time parameters will be used to calculate a specific score for each crane, and then the dispatching system will judge whether to dispatch trucks to corresponding crane by its score.

Algorithm 1 Dynamic Truck Dispatching

Require: Truck $trucks$, Crane $cranes$, Heuristic $heuristic$
  function $dispatch(truck)$
    $min\_score \leftarrow +\infty, dispatch\_task \leftarrow NULL$
    for cr in $cranes$ **do**
      $cr.score \leftarrow heuristic(cr, truck)$
      if $cr.score < min\_score$ and $cr.taskNum > 0$ **then**
        $min\_score \leftarrow cr.score$
        $dispatch\_task \leftarrow cr.tasks[0]$
      end if
    end for
    return $dispatch\_task$
  end function
  while $tasks.number \geq 0$ **do**
    for tr in $trucks$ **do**
      if $tr.tasks = NULL$ **then**
        while $tr.teu < 2$ **do**
          $tr.tasks \leftarrow dispatch(tr)$
        end while
      end if
    end for
  end while

Algorithm 2 Manual Heuristic Algorithm

Require: Parameters $parameter$, Travel Time t
  function $heuristic(cr, truck)$
    if $crane.powa < parameter.desired\_trucks$ **then**
      $score \leftarrow t(truck, cr) * (cr.powa - cr.priority)$
    else
      $score \leftarrow t(truck, cr) * parameter.desired\_trucks$
    end if
    if $crane.powt \geq parameter.powt\_limit$ **then**
      $score \leftarrow score + 200000$
    end if
    return $score$
  end function

C. Genetic Programming

Genetic programming is a technique for generating tree-based computer programs through evolution [39]. In this paper, the individual fitness of GP is assessed by TUE/hour. Furthermore, genetic operations such as crossover, mutation, selection are used as evolution methods between generations of individuals. Parameters used in GP are shown in Table II.

TABLE II
THE PROPOSED GP TERMINALS AND OPERATORS

Name	Label	Description
+	+	Add operation
-	-	Minus operation
*	*	Multiplication operation
/	/	division operation
$>(\geq)$	$>(>=)$	Greater than (Greater than or equal to)
$<(\leq)$	$<(<=)$	Less than (Less than or equal to)
&	&&	Logic and
|	||	Logic or
powa	powa_table_i	Arrived truck number of crane$_i$
powt	powt_table_i	Total truck number of crane$_i$
powd	powd_table_i	Total truck moving distance of crane$_i$ (expressed in second)
Desired Trucks	desired_wq_trucks	Desired trucks number of a crane
Travel Time	tr_fp2src_time	Travel time form truck to crane$_i$
powt limit	powt_limit	Total truck number limit of a crane
Remain tasks	remain_task_num	Remain tasks number of crane$_i$
Constant Number	constant	Random constant number

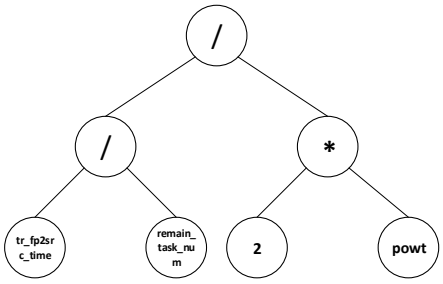

Fig. 2. Instance of GP Heuristic Tree

$$tr\_fp2src\_time/remain\_task\_num/2 * powt \quad (11)$$

As Figure 2 illustrated, individuals in GP populations are formed like trees, and leaves of trees will be replaced or removed in mutation and crossover. These GP individual trees can be transformed into expressions that are easier to comprehend. For example, the tree in Figure 2 can be decoded into Expression (11). If we give parameters values as $tr\_fp2src\_time = 400, remain\_task\_num = 20, powt = 5$, the expression 10 evaluates to 50, which can be regarded as scores in Algorithm 1. Eventually, when all tasks are finished by the GP heuristic dispatch method, TEUs/hour will be calculated as the fitness. Individuals will be sorted by their fitness values and then the next generation of evolution process starts.

TABLE III
GP INITIALIZATION AND PARAMETERS

Population Size	10000
Max Generation	1000
Crossover Probability	0.45
Mutation Probability	0.15
Reproduction Probability	0.4
Tree Initialization Method	Ramped half-and-half
Selection Method	Tournament selection, size 9
Depth Restriction	12

With initial parameters in Table III, individuals can be produced through Algorithm 3. According to the steps in the algorithm 3, first initial individuals by initialization parameters in the GP population. Moreover, to prevent evolved GP heuristics being too long, the individuals' fitnesses are punished base on their length. Then evolve the population by default parameters till reach max generation.

Algorithm 3 Proposed GP Algorithm for Dynamic Truck Dispatch

Require: Initial Parameters $initial$
  $p \leftarrow Population$
  $p.initial\_individuals(initial.population\_size)$
  $generation \leftarrow 0$
  for $generation < initial.max\_generation$ **do**
    $p.calculate\_fitness()$
    $p.penalize\_long\_individuals()$
    $next\_generation \leftarrow Population$
    while $next\_generation.size() < p.size()$ **do**
      Insert an $individual$ to $next\_generation$ by Crossover, Mutation, or Reproduction in p
    end while
    $p \leftarrow next\_generation$
    $generation \leftarrow generation + 1$
  end for

GP heuristics are assumed to consider more hidden factors and perform better than manual heuristic since humans have experimental limitations [40], which leads them to pay too much attention to common phenomena and parameters when summarizing the experience. The manual heuristic will ignore some exclusive solutions that seem to be abnormal in short term but beget excellent performance while GP heuristics are trained on actual operation data and are more likely to better handle all different scenarios. The comparison results of these heuristics are given in the next section.

V. PROBLEM INSTANCES AND EXPERIMENTS RESULTS

In all experiments, although the dynamic dispatching algorithm can adapt to a different number of trucks and QCs, to

TABLE IV
TEST RESULTS IN 40 TRUCKS 7000 TASKS (TEUS/HOUR)

Test Set	Original	Manual	GP Heuristics				
			1	*2*	*3*	*4*	*5*
TestSet1_7000_average	130.61	145.04	157.17	157.11	157.21	157.21	157.21
TestSet2_7000_average	129.75	145.92	155.85	155.85	155.94	155.93	155.93
TestSet3_7000_average	126.71	149.04	158.23	158.23	157.86	157.84	157.84
Average Improvement Base Original	0.00%	13.67%	21.75%	21.73%	21.68%	21.68%	21.68%
Average Gap	-100.00%	0.00%	59.03%	58.92%	58.57%	58.53%	58.53%

ensure the consistency of data as well as facilitate comparison and analysis, the number of QCs is set to 8, and the number of trucks is set to 40. For simulating the stochastic changes of real container terminal environments, training and testing sets will have 10 subsets of different load/unload time of cranes and travel time of trucks. GP fitness and ultimate output will be represented through the average value of these 10 subsets. The five best GP heuristics are as follows:

1) (((powd_table_i≤remain_task_num)*powt_table_i)* (tr_fp2src_time‖remain_task_num))-remain_task_num
2) ((7≤remain_task_num)+3)-remain_task_num+ desired_wq_trucks
3) (remain_task_num&&(powa_table_i)/9≤tr_fp2src_time <desired_wq_trucks*powt_limit/tr_fp2src_time/ powt_table_i)-remain_task_num
4) ((((powt_limit≤powt_table_i≤desired_wq_trucks/6))/ desired_wq_trucks≤desired_wq_trucks≥powt_limit))- (powt_limit&&powa_table_i)+remain_task_num
5) ((((powt_limit+powd_table_i))/desired_wq_trucks ≤desired_wq_trucks≥powt_limit))-(powt_limit&& powa_table_i)+remain_task_num

All the above GP heuristics were trained on 10 training sets with 500 tasks each, and the fitness of these results is the TEUs/hour in finishing all tasks. To prevent possible over-fitting, during the process of GP evolution, a training set will be selected randomly every 20 generations from 10 candidate training sets to simulate real container terminals, in which tasks do change over time. Test sets have 7000 tasks, the original static algorithm, manual algorithm, GP generated heuristic algorithms are compared for the same test sets. Test results are shown in Table IV.

Based on the test outcome, it can be concluded that in these three test sets (include 30 subsets in different conditions), the average TEUs/hour of manual heuristic and GP heuristics are both higher than original algorithm which binds trucks to cranes. The manual heuristic can improve about 14% TEUs/hour from the original algorithm used in practice. While these GP heuristics performed better and gained an improvement of around 22%. In the last row of Table IV, to describe the difference between manual heuristic and GP heuristics more clearly, the performance gap is introduced base on Formula (12) which tries to compare the improvement of GP heuristics and manual heuristics base on the original algorithm. As it is shown in Table IV, GP heuristics' performance about 59% better than manual heuristic.

$$Gap = (\frac{Result - Original}{Manual - Original} - 1) \cdot 100\% \qquad (12)$$

The manual heuristic algorithm has been practiced in Ningbo port over years, base on the statistical data, work efficiency has improved 8.14% and ship stay time has reduced 2.24%. If using the saving time to operate more ships, according to previous annual profits, manual algorithm could improve profits by about $2,400,000 per year. Although GP heuristics have not applied in real port yet, it can be inferred from manual heuristic algorithm's performance that GP heuristic algorithm can make around $1,320,000 extra profits each year.

VI. CONCLUSION AND FUTURE WORK

In this paper, a dynamic heuristic algorithm scheme is proposed to solve the container terminal truck dispatching problem. In this scheme, trucks in container terminals dispatched dynamically guided by scores generated by heuristics. Different from the traditional port truck dispatching system, such a heuristic-based dynamic dispatching system no longer needs the operators to assign trucks to different cranes frequently.

Two different types of heuristics (manual and GP evolved) are used as the core dispatching algorithms in the dynamic dispatching system, and compared with the original algorithm used in practice for test data sets. Both manual heuristic and GP heuristics make improvements over the original dispatching algorithm, which indicates that the heuristic dispatching approach can effectively improve the operations efficiency for container terminals. Furthermore, according to the test results, the improvements from the best five GP heuristics are about 59% compared with the method in practice. It can be concluded that GP can learn some hidden factors omitted in manual heuristic during the evolution process in real operation data. For instance, all of the best five GP heuristics use $remain\_task\_num$ as a key factor to adjust dispatching for the remaining tasks, while manual heuristic does not use it at all.

In conclusion, dynamic GP heuristic dispatching approach adapt well to complex real-world container terminal environments, dispatch trucks accurately and quickly. GP heuristics show an ability to make up flaws in the manual crafted heuristic, and they significantly improve the efficiency of container terminals, shorten ship dock time, and boost the benefits of container terminals. However, there are still spaces for further improvement, such as weak generality. Performance

of GP can be affected by uncertain environments, and the heuristics is not easy to read. In future research, we plan to use GP to generate Hyper-Heuristics to overcome these defects and get better dispatching solutions.

REFERENCES

[1] S. N. Sirimanne, J. Hoffman, W. Juan, R. Asariotis, M. Assaf, G. Ayala, H. Benamara, D. Chantrel, J. Hoffmann, A. Premti, L. RodrÃƒÂ©Ã‚Âguez, and F. Youssef, "Review of maritime transport, 2019," tech. rep., 2019.

[2] I. Vacca, M. Bierlaire, and M. Salani, "Optimization at container terminals: status, trends and perspectives," in *Swiss Transport Research Conference*, no. CONF, 2007.

[3] Q. Yong-xing, "Truck process system of container terminals," *Port & Waterway Engineering*, vol. 5, 2005.

[4] A. Holzinger, "From machine learning to explainable ai," in *2018 World Symposium on Digital Intelligence for Systems and Machines (DISA)*, pp. 55–66, IEEE, 2018.

[5] J. Chen, R. Bai, H. Dong, R. Qu, and G. Kendall, "A dynamic truck dispatching problem in marine container terminal," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–8, IEEE, 2016.

[6] I. F. Vis and R. De Koster, "Transshipment of containers at a container terminal: An overview," *European journal of operational research*, vol. 147, no. 1, pp. 1–16, 2003.

[7] T. E. Notteboom, "Container shipping and ports: an overview," *Review of network economics*, vol. 3, no. 2, 2004.

[8] H. J. Carlo, I. F. Vis, and K. J. Roodbergen, "Storage yard operations in container terminals: Literature overview, trends, and research directions," *European journal of operational research*, vol. 235, no. 2, pp. 412–430, 2014.

[9] B. Brinkmann, "Operations systems of container terminals: a compendious overview," in *Handbook of terminal planning*, pp. 25–39, Springer, 2011.

[10] B. Dragović, E. Tzannatos, and N. K. Park, "Simulation modelling in ports and container terminals: literature overview and analysis by research field, application area and tool," *Flexible Services and Manufacturing Journal*, vol. 29, no. 1, pp. 4–34, 2017.

[11] J. W. Böse, *Handbook of terminal planning*, vol. 49. Springer Science & Business Media, 2011.

[12] H.-O. Günther and K.-H. Kim, "Container terminals and terminal operations," 2006.

[13] T. Bektas and T. Crainic, *A brief overview of intermodal transportation*. CIRRELT, 2007.

[14] D. Steenken, S. Voß, and R. Stahlbock, "Container terminal operation and operations research-a classification and literature review," *OR spectrum*, vol. 26, no. 1, pp. 3–49, 2004.

[15] P. Schonfeld and O. Sharafeldien, "Optimal berth and crane combinations in containerports," *Journal of waterway, port, coastal, and ocean engineering*, vol. 111, no. 6, pp. 1060–1072, 1985.

[16] K. H. Kim and Y.-M. Park, "A crane scheduling method for port container terminals," *European Journal of operational research*, vol. 156, no. 3, pp. 752–768, 2004.

[17] N. Kaveshgar and N. Huynh, "A genetic algorithm heuristic for solving the quay crane scheduling problem with time windows," *Maritime Economics & Logistics*, vol. 17, no. 4, pp. 515–537, 2015.

[18] N. Al-Dhaheri, A. Jebali, and A. Diabat, "The quay crane scheduling problem with nonzero crane repositioning time and vessel stability constraints," *Computers & Industrial Engineering*, vol. 94, pp. 230–244, 2016.

[19] L. Cruz-Reyes, A. L. Alvarez, M. Quiroz-Castellanos, C. Gómez, N. R. Valdez, G. C. Valdez, and J. G. Barbosa, "A hybrid metaheuristic algorithm for the quay crane scheduling problem," in *Handbook of Research on Military, Aeronautical, and Maritime Logistics and Operations*, pp. 238–256, IGI Global, 2016.

[20] O. A. Kasm, A. Diabat, and T. Cheng, "The integrated berth allocation, quay crane assignment and scheduling problem: mathematical formulations and a case study," *Annals of Operations Research*, pp. 1–27, 2019.

[21] L. Moccia, J.-F. Cordeau, M. Gaudioso, and G. Laporte, "A branch-and-cut algorithm for the quay crane scheduling problem in a container terminal," *Naval Research Logistics (NRL)*, vol. 53, no. 1, pp. 45–59, 2006.

[22] J. Liu, Y.-w. Wan, and L. Wang, "Quay crane scheduling at container terminals to minimize the maximum relative tardiness of vessel departures," *Naval Research Logistics (NRL)*, vol. 53, no. 1, pp. 60–74, 2006.

[23] N. Al-Dhaheri, A. Jebali, and A. Diabat, "A simulation-based genetic algorithm approach for the quay crane scheduling under uncertainty," *Simulation Modelling Practice and Theory*, vol. 66, pp. 122–138, 2016.

[24] K. Lai and K. Lam, "A study of container yard equipment allocation strategy in hong kong," *International Journal of Modelling and Simulation*, vol. 14, no. 3, pp. 134–135, 1994.

[25] C. Zhang, J. Liu, Y.-w. Wan, K. G. Murty, and R. J. Linn, "Storage space allocation in container terminals," *Transportation Research Part B: Methodological*, vol. 37, no. 10, pp. 883–903, 2003.

[26] R. K. Cheung, C.-L. Li, and W. Lin, "Interblock crane deployment in container terminals," *Transportation Science*, vol. 36, no. 1, pp. 79–93, 2002.

[27] K. H. Kim and K. Y. Kim, "An optimal routing algorithm for a transfer crane in port container terminals," *Transportation science*, vol. 33, no. 1, pp. 17–33, 1999.

[28] L. Chen, N. Bostel, P. Dejax, J. Cai, and L. Xi, "A tabu search algorithm for the integrated scheduling problem of container handling systems in a maritime terminal," *European Journal of Operational Research*, vol. 181, no. 1, pp. 40–58, 2007.

[29] H. Javanshir and A. G. S. SEYED, "Yard crane scheduling in port container terminals using genetic algorithm," 2010.

[30] D.-H. Lee, J. X. Cao, Q. Shi, and J. H. Chen, "A heuristic algorithm for yard truck scheduling and storage allocation problems," *Transportation Research Part E: Logistics and Transportation Review*, vol. 45, no. 5, pp. 810–820, 2009.

[31] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management science*, vol. 6, no. 1, pp. 80–91, 1959.

[32] H.-A. Lu and J.-Y. Jeng, "Modeling and solution for yard truck dispatch planning at container terminal," in *Operations Research Proceedings 2005*, pp. 117–122, Springer, 2006.

[33] J. Cao, Q. Shi, and D.-H. Lee, "A decision support method for truck scheduling and storage allocation problem at container," *Tsinghua Science & Technology*, vol. 13, pp. 211–216, 2008.

[34] E. K. Bish, F. Y. Chen, Y. T. Leong, B. L. Nelson, J. W. C. Ng, and D. Simchi-Levi, "Dispatching vehicles in a mega container terminal," in *Container terminals and cargo systems*, pp. 179–194, Springer, 2007.

[35] J. Bose, T. Reiners, D. Steenken, and S. Voß, "Vehicle dispatching at seaport container terminals using evolutionary algorithms," in *Proceedings of the 33rd annual Hawaii international conference on system sciences*, pp. 10–pp, IEEE, 2000.

[36] E. Nishimura, A. Imai, and S. Papadimitriou, "Yard trailer routing at a maritime container terminal," *Transportation Research Part E: Logistics and Transportation Review*, vol. 41, no. 1, pp. 53–76, 2005.

[37] L. H. Lee, E. P. Chew, K. C. Tan, and Y. Wang, "Vehicle dispatching algorithms for container transshipment hubs," *OR spectrum*, vol. 32, no. 3, pp. 663–685, 2010.

[38] J. He, W. Zhang, Y. Huang, and W. Yan, "A simulation optimization method for internal trucks sharing assignment among multiple container terminals," *Advanced Engineering Informatics*, vol. 27, no. 4, pp. 598–614, 2013.

[39] J. R. Koza, "Genetic programming ii: Automatic discovery of reusable subprograms," *Cambridge, MA, USA*, vol. 13, no. 8, p. 32, 1994.

[40] U. Hahn, "Experiential limitation in judgment and decision," *Topics in cognitive science*, vol. 6, no. 2, pp. 229–244, 2014.