Proceedings of the 2022 Winter Simulation Conference
B. Feng, G. Pedrielli, Y.Peng, S. Shashaani, E. Song, C.G. Corlu, L.H. Lee, E.P. Chew, T. Roeder, and P. Lendermann, eds.

SIMULATION-OPTIMIZATION APPROACH FOR INTEGRATED SCHEDULING AT WHARF APRON IN CONTAINER TERMINALS

Mengyu Zhu
Chenhao Zhou
Ada Che

School of Management
Northwestern Polytechnical University
127 West Youyi Road
Xi'an, Shaanxi 710072, P. R. China

ABSTRACT

As the bridge between land and sea transports, container terminals play a significant role in global trading activities. In order to improve the operational efficiency at the wharf apron, this paper introduced an integrated scheduling problem with the consideration of vehicle dispatching and routing, and developed a time-space network based on takt time. Given the highly uncertain nature of vehicle movement and equipment handshakes, a simulation-optimization approach was developed, which integrates an improved particle swarm optimization and a discrete event simulation model. To further reduce simulation run time, parallel computing was adopted in the algorithm. Numerical experiment shows that the proposed algorithm outperforms the genetic algorithm-based and strategy-based simulation-optimization approach.

1 INTRODUCTION

With the trend of economic globalization and the rapid development of global maritime trade, every government has admitted that the container terminal is vital to its economy (Hsu and Wang 2020). The wharf apron is at the seaside of the terminal and connects between the quayside activities and yard side activities. It consists of quay cranes, vehicles (i.e., automated guided vehicle, AGV), and space for the vehicle traveling back and forth. The space is composed of three parts, i.e., working lanes, passing lanes, and holding buffers. Vehicles interact with the quay crane on working lanes, which are beneath the quay crane arm. Passing lanes are for vehicles crossing the wharf at high speed. Once the vehicle approaches the corresponding quay crane, it will turn into the holding buffer and wait for being called.

For large vessels, thousands of containers will be discharged and loaded in a few hours (Chen et al. 2020). In order to depart the vessel in the shortest possible time, the terminal operating system (TOS) needs to manage a fleet of vehicles to handle all pending tasks at the same time. Due to the limited space, the coordination between quay cranes and vehicles is a technical difficulty. If the coordination fails, vehicles may face heavy traffic jams or conflicts on lanes, and even form a deadlock, which has to be avoided to maintain a smooth operation at the wharf apron.

Thus, in this paper, it is interested in determining an integrated scheduling plan at the wharf apron, including (1) vehicle dispatching, i.e., which vehicle is assigned to which task, and (2) vehicle routing, i.e., which lanes should each vehicle take for the vehicle to travel from its origin to destination. However, there are many uncertain factors in practice, such as vehicle arrival time at the wharf and the handling time of quay cranes and yard cranes, making it difficult to determine the plan. Those factors are also difficult to model in close mathematical form. Thus, it motivates us to use simulation to describe the complex operations at the wharf, and then integrate it with a heuristic algorithm to search for the optimized plan.

The contribution of this paper is to first propose a time-space network based on beat time to describe the moving process of vehicles at the wharf apron and then formulate a mixed integer programming model to describe the problem formally, and a particle swarm optimization algorithm is proposed to solve it. Considering the uncertainty, a simulation-optimization approach is proposed based on the simulation software FlexSim. The rest of the paper is structured as follows: Section 2 reviews the relevant literature. Section 3 establishes a mathematical model based on an time-space network based on takt time, followed by two heuristic algorithms proposed in Section 4. Section 5 develops the simulation-optimization approach, and corresponding numerical experiments can be found in Section 6. Finally, Section 7 draws the conclusion and suggests future research directions.

2 LITERATURE REVIEW

Container terminal scheduling has become a hot research field in the past ten years. Previous scholars only improve the efficiency of a single piece of equipment, but optimizing the scheduling of a kind of equipment may lead to a suboptimal solution. So, the scheduling problem of the container port has changed from the single equipment to the collaborative scheduling between multiple types of equipment.

For the quayside scheduling, Chang et al. (2010) considered the integrated scheduling problem of berth and quay cranes allocation. A multi-objective dynamic allocation model is formulated, and a genetic algorithm is designed to solve the problem. Similarly, Xiang and Liu (2021) studied the integrated problem of berth allocation and quay crane assignment. The study adopted the k-means clustering method to formulate a robust integrated scheduling model. Hsu and Chiang (2019) studied the dynamic and continuous berth allocation problem, established a mixed-integer programming (MIP) model, and solved the problem through a two-stage approach, which significantly outperforms benchmark algorithms. For the yard side scheduling, in order to decrease the carbon dioxide emission and cut down the maintenance cost of yard cranes, a two-stage stochastic programming model is proposed to plan yard template in Hu et al. (2021). An improved Benders decomposition algorithm is developed to solve large-scale scenarios. Niu et al. (2016) focused on two scheduling problems in container terminals, i.e., yard truck scheduling problem, and the integrated yard truck scheduling and container storage allocation problem. Chen et al. (2020) investigated an integrated scheduling problem of rail cranes and AGVs. A multi-commodity network flow model is established in the form of the space-time network, which is solved by the Lagrange relaxation method. Cao et al. (2020) considered the synchronous scheduling between yard cranes and yard trucks to shorten the turnover time of container vessels. Effective vehicle scheduling plays a crucial role in terminals as the vehicles directly interact with quay cranes and yard cranes. For vehicle dispatching and routing related problems, Xu et al. (2020) proposed a route planning model to resolve the challenge of low utilization caused by the no-load return of AGVs. The study established an AGV access route planning model and was solved by a simulated annealing algorithm. Luo et al. (2016) proposed an integrated problem of vehicle scheduling and container allocation, and designed a genetic algorithm as the solution approach. Luan et al. (2021) proposed a MIP model based on a time-space network and two bi-level algorithms to solve the equipment scheduling, lane allocation, and AGV conflict-free path planning problem simultaneously.

In summary, it can be found that the research on container terminals mainly focuses on the scheduling and allocation of single or multi resources. The integrated scheduling problem on vehicle dispatching and routing, especially on land and buffer allocation, was rarely discussed, and relevant topics with the consideration of uncertain factors were even less, not to mention using simulation, and simulation-optimization as solution approaches.

3 MATHEMATIC MODEL

3.1 Problem Description

This paper focuses on the integrated scheduling problem at the wharf apron, which optimizes vehicle dispatching and routing, crossing different types of lanes in the transportation network. There are three types of tasks, i.e., loading, discharging and empty. Figure 1 gives an illustration of how AGV operates at wharf apron: for a loading task, the vehicle will pick up a container at the yard side, travel through the passing lanes, hold short in one of the holding buffers, and then turn into working lane to interact with the quay crane. The discharging task will transport a container from the quayside to the yard side, and the vehicle operates in reverse of the loading task. The empty task is for a vehicle to move from the last known position to the starting position of the next task.

An example of the operation is also shown in Figure 1. Vehicle *V1* will wait for the quay crane to load the container on a working lane, and then it will pass through the holding buffer and the passing lanes to reach the yard side. The container will be discharged by the yard crane, and then the vehicle *V1* will return to the quay crane to perform the next transportation task. Note that in Figure 1, when vehicle *V1* is in the holding buffer, it happens that another vehicle *V2* attempts to pass through the same holding buffer, so the conflict between the two vehicles needs to resolve. In this case, the system may schedule the first-arrival vehicle to take the buffer first (e.g., *V1*), and let *V2* to wait in the passing lane. Of course, the system can direct *V2* to an empty buffer if the particular rule is applied.

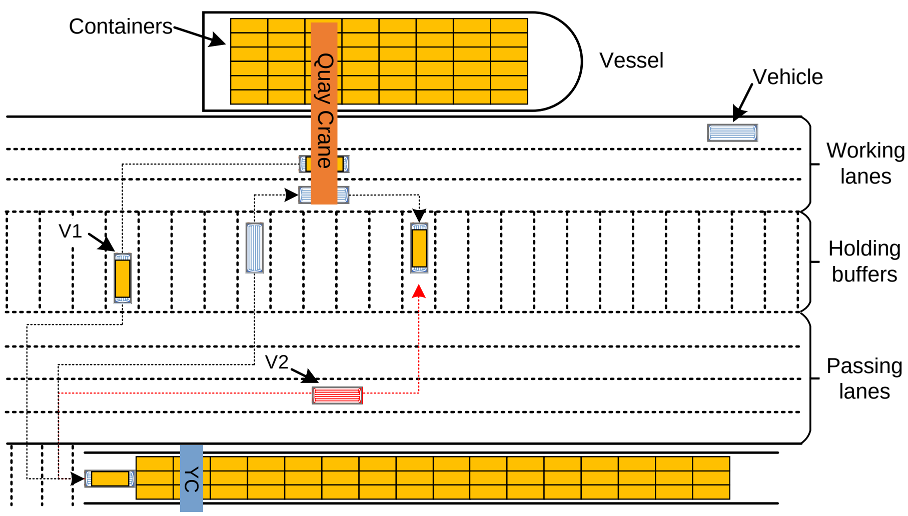

Figure 1: Problem illustration.

Specifically, the notations used in this paper are defined as follows. Task $i \in \boldsymbol{I}$ is carried out by a vehicle to transport. A task *i* can also be denoted as a container *i*. The vehicle $v \in \boldsymbol{V}$ parks on the working lane $k \in \boldsymbol{K}$ and wait for the task $i \in \boldsymbol{I}$ to be loaded. Once the quay crane loads the container onto the vehicle, the vehicle passes through the holding buffer $l \in \boldsymbol{L}$, and turns into the passing lane $p \in \boldsymbol{P}$ to get to the yard side. After the container is discharged, the vehicle immediately returns to the quayside to transport another container $j \in \boldsymbol{I}$. At the same time, once a vehicle *v* occupies the buffer *l*, other vehicles can not longer choose the buffer *l*. As the operation procedures of loading and discharging are symmetric, this paper only considers import containers. Assume quay crane and yard crane can only load or unload one container at a time.

3.2 Time-Space Network based on Takt Time

The time-space network is commonly used in the container terminals to schedule vehicles and tasks, such as Caprara et al. (2002); Shang et al. (2019) and Luan et al. (2021), which discretizes the time dimension by time period. However, the length of the time period will significantly affect the computation complexity. Three types of lanes require different ways of management. Vehicles need to interact with quay cranes for a few minutes in the working lanes, so the TOS does not need to check the status of the vehicle often. The holding buffers are used to temporally park the vehicles coming from the passing lanes. To avoid blocking the passing lanes, the TOS needs to frequently manage vehicles within, and once the working lane or holding buffer is available, the TOS will schedule the vehicle to its next location.

Therefore, an time-space network based on takt time is proposed according to the features of real-world operations as above. The idea is to define a minimum time unit, aka takt time, to schedule vehicles in holding buffers. Correspondingly, the time units of working lanes and passing lanes are 3 units and 2 units of takt time, which can be set according to the requirement of specific problems. As an example of integrated scheduling, Figure 2 shows the process of two vehicles transporting containers. Vehicle 1 selects lane 3 in the working lanes. Once the container is loaded, it departs to the holding buffer at $t = 2$, and takes 2 time unit to reach holding buffer 4, and then holds for 2 time units. Later, Vehicle 1 leaves the holding buffer and turns into passing lane 2 at $t = 8$. Then the yard crane may discharge the container at $t = 11$, and then Vehicle 1 departs from passing lane 2 immediately. At $t = 13$ Vehicle 1 arrive at the assigned holding buffer 4 taking 2 time units, and it leaves after 2 time units. Finally, Vehicle 1 gets to the working lane 3 at $t = 17$ to carry out the next task. The movement of Vehicle 2 is similar to Vehicle 1.

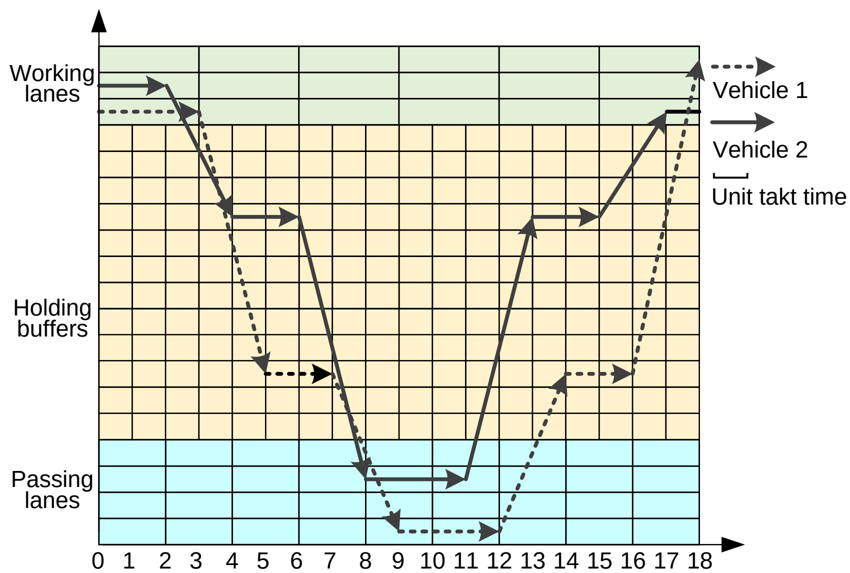

Figure 2: Time-space network based on takt time.

3.3 Modeling

Based on the time-space network concept, a mixed integer programming model is established, and the relevant notations are listed in Table 1, and decision variables in Table 2.

Table 1: Notations of sets and parameters.

Set	Description	Parameter	Description
$\boldsymbol{V}$	Set of vehicles	α	Takt time of working lanes
$\boldsymbol{I}$	Set of tasks	β	Takt time of holding buffers
$\boldsymbol{K}$	Set of working lanes	γ	Takt time of passing lanes
$\boldsymbol{L}$	Set of holding buffers	λ_{kl}	Turning time from lane k to buffer l
$\boldsymbol{P}$	Set of passing lanes	ω_{lp}	Turning time from buffer l to lane p
$\boldsymbol{N}^k$	Set of takt time unit of working lanes	θ	Handling time of quay crane loading
$\boldsymbol{N}^l$	Set of takt time unit of holding buffers	μ	Handling time of yard crane discharging
$\boldsymbol{N}^p$	Set of takt time unit of passing lanes	φ_i	Arrival time of task i

Table 2: Notations of decision variables.

Decision variables	Description
$x_{ikn}, y_{iln}, z_{ipn}$	=1, if task *i* selecting lane(buffer) *k*/*l*/*p* of *n*th takt time; =0, otherwise.
w_{vi}	=1, if vehicle *v* performs task *i*; =0, otherwise.
u_{vij}	=1, if vehicle *v* performs task *i* before task *j*; =0, otherwise.
s_i^q	Start time of quay crane loading task *i*.
s_i^y	Start time of yard crane discharging task *i*.
a_i^k	Time of task *i* arriving at working lanes.
a_i^l	Time of task *i* arriving at holding buffers.
a_i^p	Time of task *i* arriving at passing lanes.
d_i^k	Time of task *i* leaving working lanes.
d_i^l	Time of task *i* leaving holding buffers.
d_i^p	Time of task *i* leaving passing lanes.
T	Makespan

The full model is listed as follows:
Objective function:

$$\min T \tag{1}$$

Subject to:

$$a_i^p \le s_i^y \le \min\{d_i^p - \mu, T - \mu\}, \forall i \in \boldsymbol{I} \tag{2}$$

$$a_i^l + \beta \le d_i^l \le a_i^p - \omega_{lp} + M\left(2 - \sum_{n \in \boldsymbol{N}^l} y_{iln} - \sum_{n \in \boldsymbol{N}^p} z_{ipn}\right), \forall i \in \boldsymbol{I}, l \in \boldsymbol{L}, p \in \boldsymbol{P} \tag{3}$$

$$d_i^k + \lambda_{kl} \le a_i^l + M\left(2 - \sum_{n \in \boldsymbol{N}^k} x_{ikn} - \sum_{n \in \boldsymbol{N}^l} y_{iln}\right), \forall i \in \boldsymbol{I}, k \in \boldsymbol{K}, l \in \boldsymbol{L} \tag{4}$$

$$\max\{a_i^k, \varphi_i\} \le s_i^q \le d_i^k - \theta, \forall i \in \boldsymbol{I} \tag{5}$$

$$d_i^p + a_i^p - d_i^k \le a_j^k + M(1 - u_{vij}), \forall v \in \boldsymbol{V}, i, j \in \boldsymbol{I}, i \ne j \tag{6}$$

$$(n-1)\alpha - M(1 - x_{ikn}) \le a_i^k \le n\alpha + M(1 - x_{ikn}), \forall i \in \boldsymbol{I}, k \in \boldsymbol{K}, n \in \boldsymbol{N}^k \tag{7}$$

$$(n-1)\beta - M(1 - y_{iln}) \le a_i^l \le n\beta + M(1 - y_{iln}), \forall i \in \boldsymbol{I}, l \in \boldsymbol{L}, n \in \boldsymbol{N}^l \tag{8}$$

$$(n-1)\gamma - M(1 - z_{ipn}) \le a_i^p \le n\gamma + M(1 - z_{ipn}), \forall i \in \boldsymbol{I}, p \in \boldsymbol{P}, n \in \boldsymbol{N}^p \tag{9}$$

$$\sum_{k \in \boldsymbol{K}} \sum_{n \in \boldsymbol{N}^k} x_{ikn} = 1, \sum_{l \in \boldsymbol{L}} \sum_{n \in \boldsymbol{N}^l} y_{iln} = 1, \sum_{p \in \boldsymbol{P}} \sum_{n \in \boldsymbol{N}^p} z_{ipn} = 1, \sum_{v \in \boldsymbol{V}} w_{vi} = 1, \forall i \in \boldsymbol{I} \tag{10}$$

$$u_{vij} + u_{vji} \le \min\{w_{vi} + w_{vj}, 1\}, \forall v \in \boldsymbol{V}, i, j \in \boldsymbol{I}, i \ne j \tag{11}$$

$$u_{vij} + u_{vji} \ge w_{vi} + w_{vj} - 1, \forall v \in \boldsymbol{V}, i, j \in \boldsymbol{I}, i \ne j \tag{12}$$

$$\sum_{i \in \boldsymbol{I}, i \ne j} u_{vij} \le \sum_{i' \in \boldsymbol{I}} w_{vi'} - 1, \forall v \in \boldsymbol{V}, j \in \boldsymbol{I} \tag{13}$$

$$\sum_{j \in \boldsymbol{I}, j \ne i} u_{vij} \le \sum_{i' \in \boldsymbol{I}} w_{vi'} - 1, \forall v \in \boldsymbol{V}, i \in \boldsymbol{I} \tag{14}$$

The objective is to minimize the total makespan *T*. Constraints (2)-(6) are time constraints which need to be satisfied while the task is being performed, where *M* is a large positive number. Constraints (7)-(9)

denote the relationship between arriving time of each area and the selected takt time. For example, for constraint (8), if $y_{iln}=1$, then the time a_i^l arriving at the holding buffers must be within the interval $[(n-1)\beta, n\beta]$. Constraint (10) indicates that one task can only select one takt time of one lane in each area, and one task can only be performed by one vehicle. For Constraint (11)-(12), for arbitrary tasks *i* and *j*, $u_{vij}+u_{vji}\le 1$ must hold. If $u_{vij}+u_{vji}=1$ which indicates both tasks *i* and *j* are performed by vehicle *v*, then $w_{vi}+w_{vj}=2$; if $u_{vij}+u_{vji}=0$ which indicates vehicle *v* either performs only one of task *i* and *j* or neither, then $w_{vi}+w_{vj}\le 1$. Thus Constraint (11) and (12) hold. In Constraint (13), $\sum u_{vij}$ represent the total number of tasks performed by vehicle *v* before task *j*, and $\sum w_{vi}$ is the total number of tasks performed by vehicle *v*. If task *j* is the last task performed by vehicle *v*, then $\sum u_{vij}=\sum w_{vi}-1$; otherwise, $\sum u_{vij}<\sum w_{vi}-1$.

4 SOLUTION APPROACH

Despite the conventional solution approaches commonly used to solve integrated scheduling problems, the uncertainties during the operation always interfere with the schedule. For example, the miss-match between equipment handshakes will cause one equipment to wait for the other one, and the schedule will be disrupted with errors accumulating. Therefore, a solution that has good performance under uncertainties is more preferred than an optimal solution in deterministic settings.

Simulation-optimization has been proven to be useful to find good solutions in complex systems with high uncertainties, especially when the system's performance cannot be formulated precisely. Its applications in maritime logistics can be found in Zhou et al. (2021). The integrated scheduling problem described in this study is in line with the characteristics of the simulation-optimization problem, where the search algorithm, specifically the particle swarm optimization (PSO), is used to sample feasible vehicle dispatching and routing decision, while the simulation is used to evaluate the real performance of the solutions. The framework of the proposed simulation-optimization approach is illustrated in Figure 3.

The PSO was firstly proposed by Kennedy and Eberhart (1995), which has been widely adopted to solve maritime problems because of its simplicity and efficiency, such as quay crane scheduling problem(Malekahmadi et al. 2020), yard truck scheduling problem (Hsu et al. 2021), yard crane scheduling problem (He et al. 2015). In a recent development, a PSO variation, local PSO (LPSO) proposed by Niu et al. (2016), has shown to be effective in solving the integrated problem of yard truck scheduling and storage allocation. In this study, the LPSO is further modified to be part of the simulation-optimization framework.

4.1 Local Particle Swarm Optimization

Before going into details, the solution of the PSO algorithm needs to be defined: the solution has 5*I* elements, i.e., $\boldsymbol{X}_n=\{x_{ni},...,x_{n(I+i)},...,x_{n(2I+i)},...,x_{n(3I+i)},...,x_{n(4I+i)}\}$ for $i=1,2,...,I, n=1,2,...,N$, and x_{ni} is a real number. $x_{ni}\in[0,V]$ indicates that the vehicle $\lceil x_{ni}\rceil$ performing task *i*, where $\lceil x\rceil$ takes the minimal integer which is not less than x. If one or several tasks are handled by the same vehicle, $x_{n(I+1)}\in[0,I]$ will determine the handling sequence of those task(s). For instance, if $\lceil x_{ni}\rceil=\lceil x_{nj}\rceil$, $x_{n(I+i)}\le x_{n(I+j)}$ indicates that the vehicle performs task *i* before task *j*. $x_{n(2I+1)}\in[0,K]$ denotes that the task *i* selects working lane $\lceil x_{n(2I+i)}\rceil$. The definitions of $x_{n(3I+i)}\in[0,L]$ and $x_{n(4I+1)}\in[0,P]$ are similar to $x_{n(2I+1)}$, which represent holding buffers and passing lanes, respectively. An example of the solution representation with $I=10, V=3, K=3, L=20$, and $P=6$ is given in Figure 4. In the row with x_{ni}, the remark denotes that each task is handled by one of the vehicles. In the row with $x_{n(I+i)}$, each cell has a value with a unique integer number, which represents the order of vehicles performing tasks.

Motivated by Clerc and Kennedy (2002), the formulations for updating particle position and velocity are $\boldsymbol{X}_n^{t+1}=\boldsymbol{X}_n^t+\boldsymbol{V}_n^{t+1}$ and $\boldsymbol{V}_n^{t+1}=\chi(\boldsymbol{V}_n^t+c_1R_1(\boldsymbol{P}_n^t-\boldsymbol{X}_n^t)+c_2R_2(\boldsymbol{G}^t-\boldsymbol{X}_n^t))$ respectively, where $\boldsymbol{V}_n^t$ and $\boldsymbol{X}_n^t$ is the current velocity and position of particle *n*, $\boldsymbol{P}_n^t$ and $\boldsymbol{G}^t$ is the current best and global best positions of particle *n*, R_1 and R_2 are a random number of $[0,1]$, c_1 and c_2 are acceleration coefficients, and χ is called constriction factor. Note that c_1 and c_2 can be computed by $c_{1,2}=c_{\min}+(c_{\max}-c_{\min})\times t/T$, where T is the maximum number of iterations. χ is used to prevent the infinite growth of velocity and ensure the convergence of the algorithm, and it is computed by $\chi=2/|2-c-(c^2-4c)^{1/2}|$, where $c=c_1+c_2, c>4$.

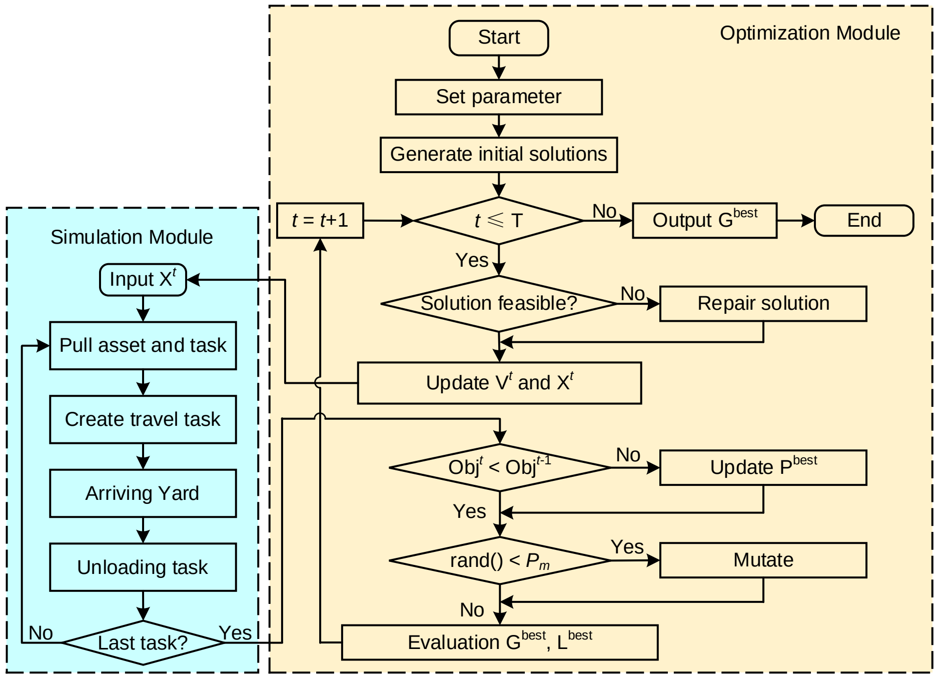

Figure 3: Framework of simulation-optimization approach.

task	1	2	3	4	5	6	7	8	9	10
x_{ni}	2.39(v_3)	0.39(v_1)	1.36(v_2)	2.28(v_3)	1.77(v_2)	2.82(v_3)	0.68(v_1)	2.97(v_3)	0.52(v_1)	1.28(v_2)
$x_{n(I+i)}$	5.98	6.26	4.73	9.23	0.04	3.79	2.12	8.47	1.42	3.62
$x_{n(2I+i)}$	2.27	1.96	1.64	1.27	2.24	0.63	1.22	2.84	0.57	1.04
$x_{n(3I+i)}$	4.52	16.02	9.09	4.15	10.36	8.57	1.69	13.46	19.51	17.19
$x_{n(4I+i)}$	4.28	2.26	3.22	2.24	1.47	5.94	0.23	4.45	3.31	0.78

Figure 4: Solution representation.

Generally, particles share information through the global best position $\boldsymbol{G}^t$, which means that each particle interacts with other particles in the whole population and may lead to local optimal. As an improvement, Eberhart and Kennedy (2002) and Marinakis et al. (2013) proposed a new concept, i.e., LPSO, which constructs a neighborhood topology for each particle. The neighborhood topology of a particle, which can usually be represented by a graph, is the set of particles connected to it. As shown in Figure 5, the global best position $\boldsymbol{G}^t$ in the whole population is replaced by the local best position $\boldsymbol{L}_n^t$ in the particle neighborhood topology, and ring topology is one of the most commonly used structures.

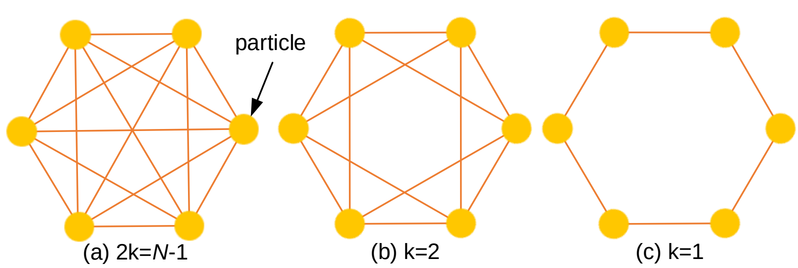

Figure 5: Ring topology.

In the ring topology, each particle is only connected with 2k particles (the global PSO is equivalent to that each particle is connected with N-1 particles, as shown in Figure 5(a)), and the indexes of these particles are sequential. For example, the particles connected with particle i are $i-k,...,i,...,i+k$. As shown in Figure 5(c), when k =1, the ring topology has the simplest structure, and each particle is only connected with two adjacent particles. When k takes a small value, the ring topology can delay the propagation of information in the whole population and reduce the convergence speed. Let $\aleph_n$ denote the neighborhood topology of particle n, then the local best position is $\boldsymbol{L}_n^t = \{\boldsymbol{X}_m \mid f(\boldsymbol{X}_m) \le f(\boldsymbol{X}), \boldsymbol{X}_m \in \aleph_n\}$ and velocity formulation is $\boldsymbol{V}_n^{t+1} = \chi(\boldsymbol{V}_n^t + c_1 R_1(\boldsymbol{P}_n^t - \boldsymbol{X}_n^t) + c_2 R_2(\boldsymbol{L}_n^t - \boldsymbol{X}_n^t))$.

There are two strategies to generate the initial solution: random strategy and equilibrium strategy. The random strategy is to generate a solution following the uniform distribution. The equilibrium strategy is to balance the workload between vehicles. Following this idea, first, some solutions are generated randomly and then retain solutions whose number of tasks performed by vehicles is within the range of ± 20% of the average value I/V . Those solutions beyond the range are regarded as inferior solutions and can be discarded. The process is repeated until a good solution is found.

Due to the rapid growth of velocity, the PSO may fall into local optimum easily. Therefore, inspired by Zeng et al. (2017), a mutation mechanism is proposed. Given the mutation rate P_m and two random numbers r_1 and r_2, if $r_1 < P_m$, then perform the mutation operation. If mutation is performed and the random number r_2 is larger than 0.5, $\boldsymbol{X}_n = \boldsymbol{X}_n + R_1(1-t/T)^{\theta}(\boldsymbol{P}_n - \boldsymbol{X}_n)$, otherwise, $\boldsymbol{X}_n = \boldsymbol{X}_n + R_1(1-t/T)^{\theta}(\boldsymbol{G} - \boldsymbol{X}_n)$. Meanwhile, a large mutation rate is set at early stage of the iteration to increase the search range, and a low mutation rate is set at a later stage to search for the optimal solution. Thus, $P_m = P_m^{\max} - (P_m^{\max} - P_m^{\min}) \times t/G$. In addition, a repairing procedure is implemented: for two tasks i and j performed on the same vehicle, if $\varphi_i \le \varphi_j$, $x_{n(I+i)} \le x_{n(I+j)}$ holds; otherwise, swap the values of $x_{n(I+i)}$ and $x_{n(I+j)}$.

4.2 Simulation Modeling

The simulation model developed in this study is based on commercial software FlexSim, which supports modeling as discrete event simulation. Our model is modeled in the combination of 3D objects and process flows. The 3D objects are to simulate equipment interactions, especially vehicle movement, which represent quay cranes and yard cranes. The process flow model controls the simulation logic of the 3D objects.

To manage resources such as AGVs, tasks, quay cranes, and yard cranes, the *List* of the process flow is adopted. Corresponding *Lists*, i.e., *AGVList*, *TaskList*, *QCList*, and *YCList*, is used to hold resources until it is called to serve other activities by triggering a *Pull Activity*. The resource will be returned back to the *List* via a *Push Activity*. The logic of the simulation model process is as follows:

Simulation Logic	
Step1	Convert the solution into a suitable form for the simulation model;
Step2	Pull a task from the *TaskList*, and pull an AGV from *AGVList*;
Step3	Pull a quay crane from *QCList*;
Step4	Quay crane loads the task onto AGV;
Step5	Push QC into *QCList*;
Step6	After leaving the working lane, arrive at the holding buffer, and then reach passing lane(s);
Step7	Pull a yard crane from *YCList*;
Step8	Yard crane discharges the task from the AGV;
Step9	Push YC into *YCList*;
Step10	Task enters the *Sink*, and the AGV returns to the quayside;
Step11	Push AGV into *AGVList*;
Step12	If it is the last task, go to *Step 2*; Otherwise, go to *Step 13*;
Step13	Output results.

5 NUMERICAL EXPERIMENT

In this section, under the simulation-optimization framework, the performances of the LPSO and benchmark heuristics, including the genetic algorithm (GA) and first-come-first-serve-based algorithm (FCFS), are evaluated. As a reference, the lower bound of the problem is obtained using the commercial solver Gurobi. The algorithms are developed in C++ using Visual Studio 2019. The experiments are conducted on a workstation with Intel(R) Xeon(R) W-1290P CPU (10 Cores) @ 3.70GHz and 64GB RAM. The version of FlexSim software is 20.1.1 Education version.

5.1 Experiment Setup

The vehicle speed, and the handling times of the quay crane and yard crane are assumed to follow a normal distribution with the mean value of 4.5 meters per second, 120 seconds, and 90 seconds, and the standard deviation of 0.167, 1.667, 1.667, respectively. According to the layout of Xiamen Yuanhai Port, the length of the wharf apron is 400 meters, and the width is 104 meters. The working lane has 4 lanes while the passing lane has 6. In between, there are 50 holding buffers that are perpendicular to working and passing lanes. The parameters of the LPSO, including *T*, *N*, θ, δ, $P_m^{\max}$, $P_m^{\min}$, $c_{\max}$, and $c_{\min}$, are set to be 20, 20, 4, 0.7,0.1, 0.01, 5, and 2, respectively.

The size of the problem can be defined by the number of tasks and the number of vehicles, which is denoted as $(I \times V)$. For demonstration purposes, four scenarios in different problem sizes are introduced, i.e., (20×6), (40×6), (60×8) and (80×8). Five replications will be conducted for each scenario, and the average value is taken as the outcome. In order to obtain statistical results, 100 simulation runs will be conducted for each particle, and the simulation is conducted in non-parallel way.

5.2 Result Analysis

The experiment results are shown in Table 3, where *LB* is the lower bound of the problem obtained from Gurobi, in which the input data is fixed, so the LB is the same for the same scenario. *OBJ* is the objective value of the corresponding algorithm, *Time* represents CPU time, *Gap* is the gap between *OBJ* and *LB*, where $Gap = (OBJ - LB) / LB \times 100\%$. Note that due to the randomness in the simulation, the results have randomness, so it is reasonable that gap is negative in Table 3.

In Table 3, it can be found that from the view of problem size, the LPSO outperforms the GA and is far better than the FCFS. Preliminary results also show that the average gap decreases when the problem size increases. It is possible that when the ratio between the numbers of vehicles and tasks is appropriate, vehicles can be fully utilized with the minor waiting time. If there are too many vehicles or too few tasks, the vehicle may become idle frequently.

In terms of computational efficiency, the average computing time of all approaches increases with the increase of problem size. This reason is that the larger problem size, the more vehicles and tasks, so the time for a single simulation run increases accordingly. In addition, the convergence of LPSO and GA algorithm is shown in Figure 6. It can be founded that LPSO algorithm decreases rapidly in first few iterations and becomes stable after a donze of iterations. However, the GA-based approach takes longer time to get a significant improvement. Note that it is normal that the starting points of LPSO and GA are different because the two approaches find different optimal solutions in the first generation.

Note that, in Table 3, as single thread is used for simulation experiment, it takes three hours or more to achieve the results of LPSO and GA. It is mainly due to the huge number of simulation runs needs to be executed during the whole search. Thus, we further adopts the parallel computing technique to accelerate the simulation run for scenario (20×6) at each iteration of LPSO and GA algorithms. As the CPU is equipped with 10 cores, multiple threads (nearly 20) can be used to run simulation experiment concurrently. As shown in Table 4, the computation time of the LPSO and GA are much less, 1248 seconds verse 10957 seconds, than the result presented in Tabel 3, which is acceptable in real world application.

Table 3: Algorithm comparison.

No.	Gurobi	LPSO			GA			FCFS		
	LB	*OBJ*	*Time*	*Gap*(%)	*OBJ*	*Time*	*Gap*(%)	*OBJ*	*Time*	*Gap*(%)
Scenario (20×6)										
1		2452.67	11071.19	-1.93	2763.24	11073.66	10.49	2939.67	499.52	17.54
2		2469.47	10937.48	-1.26	2608.53	11014.60	4.30	2771.37	492.81	10.81
3	2501	2603.97	9808.28	4.12	2574.41	11214.41	2.94	3019.91	446.66	20.75
4		2590.30	10171.25	3.57	2653.41	10848.61	6.09	2951.02	462.06	17.99
5		2578.73	11799.74	3.11	2690.56	11220.20	7.58	2985.34	440.28	19.37
Avg.	2501	2539.03	10757.59	1.52	2658.03	11074.30	6.28	2933.46	468.27	17.29
Scenario (40×6)										
1		4655.38	10830.60	6.46	5005.80	11834.11	14.47	5143.90	472.17	17.63
2		4737.21	10884.58	8.33	5076.57	10536.00	16.09	5531.14	532.83	26.48
3	4373	4722.57	11851.58	7.99	4868.22	11700.53	11.32	5218.62	519.61	19.34
4		4814.28	10547.87	10.09	4759.32	12173.13	8.83	5489.74	505.80	25.54
5		4761.78	11887.95	8.89	4815.05	10763.33	10.11	5161.83	479.80	18.04
Avg.	4373	4738.24	11200.52	8.35	4904.99	11401.42	12.17	5309.05	502.04	21.41
Scenario (60×8)										
1		6900.29	12394.24	0.63	7049.04	11841.02	2.80	8268.30	584.89	20.58
2		6859.56	13120.62	0.04	6910.03	11744.10	0.77	8040.07	518.30	17.25
3	6857	6876.98	13052.03	0.29	7017.32	13371.63	2.34	8088.42	585.39	17.96
4		6903.04	12348.55	0.67	7006.06	12145.37	2.17	7945.78	498.09	15.88
5		6934.87	11921.56	1.14	6848.21	13232.70	-0.13	7704.36	523.03	12.36
Avg.	6857	6894.95	12567.40	0.55	6966.13	12466.96	1.59	8009.38	541.94	16.81
Scenario (80×8)										
1		9027.37	13918.46	-0.15	9092.01	13983.34	0.56	10348.90	629.11	14.47
2		8989.28	13417.02	-0.57	9033.40	13604.86	-0.08	11005.69	610.08	21.73
3	9041	9023.38	14699.26	-0.19	9017.62	13984.87	-0.26	10107.18	665.47	11.79
4		9118.99	13990.51	0.86	9070.83	14187.59	0.33	10728.39	562.81	18.66
5		9172.65	13164.81	1.46	9164.23	13372.74	1.36	10689.42	626.31	18.23
Avg.	9041	9066.33	13838.01	0.28	9075.62	13826.68	0.38	10575.92	618.76	16.98

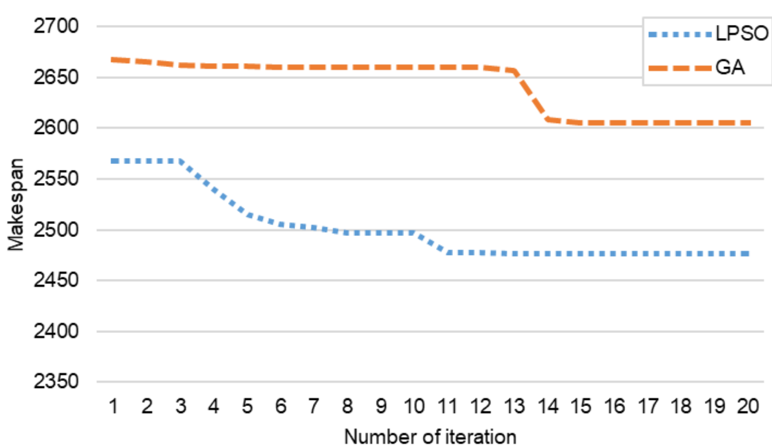

Figure 6: Convergence of LPSO and GA for scenario (20 × 6).

Table 4: Algorithm comparison.

No.	Gurobi	LPSO			GA			FCFS		
	LB	*OBJ*	*Time*	*Gap*(%)	*OBJ*	*Time*	*Gap*(%)	*OBJ*	*Time*	*Gap*(%)
Scenario (20×6)										
1		2476.17	1250.91	-0.99	2605.02	1406.22	4.16	2939.67	499.52	27.26
2		2413.38	1237.95	-3.50	2603.70	1495.98	4.11	2771.37	492.81	19.97
3	2501	2443.68	1292.09	-2.29	2609.92	1505.62	4.36	3019.91	446.66	30.73
4		2420.82	1234.81	-3.21	2549.88	1504.17	1.95	2951.02	462.06	27.75
5		2447.66	1227.30	-2.13	2592.17	1415.97	3.65	2985.34	440.28	29.24
Avg.	2501	2440.34	1248.61	-2.43	2592.14	1465.59	3.64	2933.46	468.27	26.99

6 CONCLUSION

This paper studies the integrated scheduling problem of vehicle dispatching and routing with the consideration of potential conflicts within working lanes, passing lanes, and holding buffers at the wharf apron. The problem was first developed into a mixed integer programming model using time-space network based on takt time, which allows different equipment to have unequal time units, i.e., takt time. To handle the operation uncertainty and obtain a good feasible solution, a PSO-based simulation-optimization approach was developed, which integrated the LPSO with certain mutation and repairing techniques. Through four groups of numerical studies, the result of the objective value showed that the LPSO outperformed other benchmark heuristic and strategy based algorithms from 0.1% to 6%. Further more, when the parallel computing is implemented for simulation run, the computation time can be significantly reduced by 88.6 % on average.

As a preliminary work, this paper still has room for improvements. For example, (1) Better benchmark algorithms can be compared with the proposed one; (2) The number of iterations can be dynamically adjusted according to the trend of outcomes using the multi-fidelity simulation-optimization approach. The approach may use a simplified simulation model or a queueing model to quickly evaluate solutions in the large range. Once the promising solutions are identified, a detailed simulation model will be used to search for the optimized solution; (3) In addition, by controlling the total computing budget, the number of simulation runs for each particle, and the number of iterations can be optimized based on the optimal computing budget allocation algorithm.

ACKNOWLEDGMENT

This research is supported by the National Natural Science Foundation of China [72101203], Shaanxi Provincial Key R&D Program, China [2022KW-02].

REFERENCES

Cao, P., G. Jiang, S. Huang, and L. Ma. 2020. "Integrated Simulation and Optimisation of Scheduling Yard Crane and Yard Truck in Loading Operation". *International Journal of Shipping and Transport Logistics* 12(3):230–250.

Caprara, A., M. Fischetti, and P. Toth. 2002. "Modeling and Solving the Train Timetabling Problem". *Operations Research* 50(5):851–861.

Chang, D., Z. Jiang, Y. Wei, and J. He. 2010. "Integrating Berth Allocation and Quay Crane Assignments". *Transportation Research Part E: Logistics and Transportation Review* 46(6):975–990.

Chen, X., S. He, Y. Zhang, L.C. Tong, P. Shang, and X. Zhou. 2020. "Yard Crane and AGV Scheduling in Automated Container Terminal: A Multi-Robot Task Allocation Framework". *Transportation Research Part C: Emerging Technologies* 114:241–271.

Clerc, M., and J. Kennedy. 2002. "The Particle Swarm - Explosion, Stability, and Convergence in a Multidimensional Complex Space". *IEEE Transactions on Evolutionary Computation* 6(1):58–73.

Eberhart, R., and J. Kennedy. 2002. "A New Optimizer Using Particle Swarm Theory". In *Proceedings of the Sixth International*

Symposium on Micro Machine and Human Science, edited by S. Spurgeon, 39–43. Nagoya, Japan: Institute of Electrical and Electronics Engineers, Inc.

He, J.L., Y.F. Huang, and W. Yan. 2015. "Yard crane scheduling in a container terminal for the trade-off between efficiency and energy consumption". *Advanced Engineering Informatics* 29(1):59–75.

Hsu, H., and C. Wang. 2020. "Resources Planning for Container Terminal in a Maritime Supply Chain Using Multiple Particle Swarms Optimization". *Mathematics* 8(5):764.

Hsu, H., and T. Chiang. 2019. "An Improved Shuffled Frog-Leaping Algorithm for Solving the Dynamic and Continuous Berth Allocation Problem". *Applied Sciences-Basel* 9(21):4682.

Hu, H., J. Mo, and L. Zhen. 2021. "Improved Benders Decomposition for Stochastic Yard Template Planning in Container Terminals". *Transportation Research Part C: Emerging technologies* 132:103365.

Kennedy, J., and R. Eberhart. 1995. "Particle Swarm Optimization". In *Proceedings of ICNN'95 - International Conference on Neural Networks*, edited by S. Spurgeon, 1942–1948. Perth, WA, Australia: Institute of Electrical and Electronics Engineers, Inc.

Luan, D., M. Zhao, Q. Zhao, and N. Wang. 2021. "Modelling of Integrated Scheduling Problem of Capacitated Equipment Systems with a Multi-Lane Road Network". *Plos One* 16(6):251875.

Luo, J., Y. Wu, and A.B. Mendes. 2016. "Modelling of Integrated Vehicle Scheduling and Container Storage Problems in Unloading Process at an Automated Container Terminal". *Computers & Industrial Engineering* 94:32–44.

Malekahmadi, A., M. Alinaghian, S.R. Hejazi, and M.A.A. Saidipour. 2020. "Integrated Continuous Berth Allocation and Quay Crane Assignment and Scheduling Problem with Time-Dependent Physical Constraints in Container Terminals". *Computers & Industrial Engineering* 147:106672.

Marinakis, Y., G. Iordanidou, and M. Marinaki. 2013. "Particle Swarm Optimization for the Vehicle Routing Problem with Stochastic Demands". *Applied Soft Computing* 13(4):1693–1704.

Niu, B., T. Xie, L. Tan, Y. Bi, and Z. Wang. 2016. "Swarm Intelligence Algorithms for Yard Truck Scheduling and Storage Allocation Problems". *Neurocomputing* 188:284–293.

Shang, P., R. Li, J. Guo, K. Xian, and X. Zhou. 2019. "Integrating Lagrangian and Eulerian Observations for Passenger Flow State Estimation in an Urban Rail Transit Network: A Space-Time-State Hyper Network-based Assignment Approach". *Transportation Research Part B: Methodological* 121:135–167.

Xiang, X., and C. Liu. 2021. "An Almost Robust Optimization Model for Integrated Berth Allocation and Quay Crane Assignment Problem". *Omega* 104:102455.

Xu, Y., L. Qi, W. Luan, X. Guo, and H. Ma. 2020. "Load-in-Load-out AGV Route Planning in Automatic Container Terminal". *IEEE Access* 8:157081–157088.

Zeng, M., W. Cheng, P. Guo, and J.R. Torregrosa. 2017. "Modelling and Metaheuristic for Gantry Crane Scheduling and Storage Space Allocation Problem in Railway Container Terminals". *Discrete Dynamics in Nature and Society* 2017:9025482.

Zhou, C., N. Ma, X. Cao, L.H. Lee, and E.P. Chew. 2021. "Classification and Literature Review on the Integration of Simulation and Optimization in Maritime Logistics Studies". *IISE Transactions* 53(10):1157–1176.

AUTHOR BIOGRAPHIES

MENGYU ZHU is a graduate student in the Department of Management Science and Engineering of Northwestern Polytechnical University, Xi’an. He holds a B.Sc. in Industrial Engineering from China University of Mining and Technology. His research interests are integrated scheduling of automated container terminals, focusing on mathematical modeling and simulation-optimization algorithms. His email address is zhumy@mail.nwpu.edu.cn.

ADA CHE is a Professor in the Department of Management Science and Engineering of Northwestern Polytechnical University, Xi’an. He has authored or coauthored over 50 articles in international journals. His current research interests include transportation planning and optimisation, production scheduling, and operations research. His email address is ache@nwpu.edu.cn.

CHENHAO ZHOU is a Professor with the School of Management, Northwestern Polytechnical University, Xi’an. His research interests are transportation systems and maritime logistics using simulation and optimization methods. His email address is zhouchenhao@nwpu.edu.cn.