Hindawi Publishing Corporation
Abstract and Applied Analysis
Volume 2013, Article ID 634812, 17 pages
http://dx.doi.org/10.1155/2013/634812

Research Article

Optimal Scheduling for Retrieval Jobs in Double-Deep AS/RS by Evolutionary Algorithms

Kuo-Yang Wu,[1] Sendren Sheng-Dong Xu,[2] and Tzong-Chen Wu[1,3]

[1] *Department of Information Management, National Taiwan University of Science and Technology, Taipei 106, Taiwan*
[2] *Graduate Institute of Automation and Control, National Taiwan University of Science and Technology, Taipei 106, Taiwan*
[3] *Taiwan Information Security Center, National Taiwan University of Science and Technology, Taipei 106, Taiwan*

Correspondence should be addressed to Sendren Sheng-Dong Xu; sdxu@mail.ntust.edu.tw

Received 21 January 2013; Revised 7 May 2013; Accepted 27 May 2013

Academic Editor: Jein-Shan Chen

We investigate the optimal scheduling of retrieval jobs for double-deep type Automated Storage and Retrieval Systems (AS/RS) in the Flexible Manufacturing System (FMS) used in modern industrial production. Three types of evolutionary algorithms, the Genetic Algorithm (GA), the Immune Genetic Algorithm (IGA), and the Particle Swarm Optimization (PSO) algorithm, are implemented to obtain the optimal assignments. The objective is to minimize the working distance, that is, the shortest retrieval time travelled by the Storage and Retrieval (S/R) machine. Simulation results and comparisons show the advantages and feasibility of the proposed methods.

1. Introduction

Optimization has shown itself to be one of the most important problems of engineering design and scientific analysis (e.g., see [1–12]) and has been widely studied in the manufacturing industry [1–8]. Since Automated Storage and Retrieval Systems (AS/RS) have been used as a part of manufacturing systems [1–8, 13–19] and function as important links in the supply chain, many researchers have discussed the applications of optimization to such systems. The AS/RS have great flexibility in interfacing with other components of Flexible Manufacturing Systems (FMS) to maintain quick responses to demands in manufacturing, warehousing, and distribution applications [19]. The AS/RS are a class of industrial automatic systems composed of high-level stereoscopic shelves, raising machines, forklift trucks, stock movement systems, unmanned carriers, control systems, peripheral apparatuses, and so forth. The system can fully utilize minimum stock space and realize online control of the apparatus through computer. It can rapidly process data, deal accurately with stock control, and thus rationally improve management efficiency. However, the high efficiency of AS/RS is based on storage and retrieval locations and scheduling management [8]. Scheduling can be a difficult task due to a variety of reasons, including different goals, priorities, and the complexity of the computing systems. In recent years, there has been growing interest in exploring the scheduling and performance of AS/RS. For example, Chetty and Reddy [1] proposed the Genetic Algorithm (GA) for the AS/RS integrated with machines. Wang et al. [2] discussed the selection of AS/RS scheduling rules based on GA. Asokan et al. [3] discussed the optimal scheduling of the AS/RS in FMS.

Traditional AS/RS are almost structured in the single-deep configuration for the storage rack, such that the Storage and Retrieval (S/R) machine has movement control in the middle of the pathway. However, in a range-limited factory, it will store significantly more goods if we adopt the double-deep AS/RS, a double-deep rack with the pallets being stored and retrieved by an automated S/R machine (e.g., see Figures 1 and 2, [19]). Due to more goods being preserved in the double-deep AS/RS, it is even more important to precisely manage the fetching and placing of goods, such that possible interference and/or deadlock can be avoided. Therefore, in order to make the double-deep type AS/RS more effective, we have to optimize the storage location and the job scheduling for storage and retrieval. A survey of the literature indicates

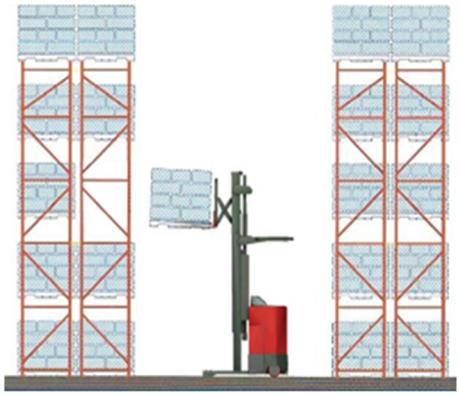


Figure 1: Configuration of a double-deep rack.

Figure 2: An actual photo of a double-deep rack.

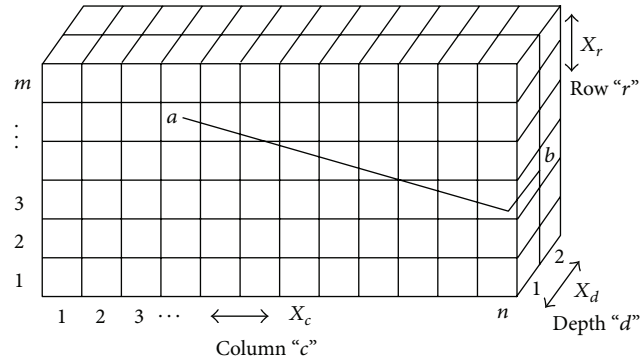


Figure 3: Description of the AS/RS structure model.

that optimal scheduling for double-deep type Automated Storage and Retrieval Systems (AS/RS) has not yet been studied. In this paper, we systematically propose a formula for the AS/RS traveled distance calculation and are the first to use evolutionary algorithms to optimize retrieval jobs scheduling problems for the double-deep type AS/RS. Based on the study, scheduling with evolutionary algorithms is shown to perform more effectively than random retrieval without evolutionary algorithms. Similar applications can thus be extended to more complex systems and job scheduling.

The Genetic Algorithm (GA) [20–25] was first proposed by Holland 1975 [20] based on the genetic scheme and bioevolution. Being a global search algorithm, it is capable of producing better solutions in complex situations through chromosome representation, reproduction, crossover, and mutation. The concept of Artificial Immune System (AIS) was improved from GA and was originally proposed by Jerne [26]; it has since been widely studied (e.g., see [26–32]). In Immune Genetic Algorithm (IGA) approaches, we calculate the affinity between any pairs of strings. If the affinity is close, the string will be selected with a lower possibility to avoid local minimum. Due to the advantage that a variety of feasible spaces can be better assured while solving the optimization problems to more likely achieve optimal solutions, IGA is thus applied in this study. Particle Swarm Optimization (PSO) [33–38] was first proposed by Kennedy and Eberhart 1995 [33] based on swarm intelligence. All three algorithms share many advantages, such as fast convergence speed, easy implementation, and limited parameters to be tuned. Unlike GA and IGA, however, PSO does not have evolution operators, such as crossover and mutation. PSO is not only easy to implement, since there are fewer parameters to adjust, but it is also able to combine local and global search methods [33]. In this study, GA, IGA, and PSO are applied to optimal retrieval scheduling in the AS/RS. The algorithms and objective functions can minimize the working distance, that is, the shortest retrieval time, travelled by the automated S/R machine, so that the AS/RS will be more effective. Finally, simulation results and comparisons show the advantages and feasibility of the proposed methods.

This remainder of this paper is organized as follows. Section 2 presents the problem statement. Section 3 describes the implementation of evolutionary algorithms, including GA, IGA, and PSO. The simulation results are illustrated and discussed in Section 4. Finally, Section 5 presents the conclusion.

2. AS/RS Structure Model and Problem Statement

Figure 3 shows the structure of the double-deep type Automated Storage and Retrieval Systems (AS/RS) considered in this work. The AS/RS structure is rectangular with m rows, n columns, and 2 depths, where m is set to be 20 and n is set to be 30. Each cell is capable of holding one cargo in any type. In our study, we assume that there are 4 types of cargos and 10 for each type. The center distance between any two adjacent cells in a row is X_r, in a column is X_c, and in a depth is X_d, respectively. The numbers 1, 2, 3 given beside the cell indicate the address of the storage location. There is one shuttle of the crane type, which is capable of moving vertically, horizontally, and thoroughly. The maximum load capacity of the forklift truck is 10 goods each time; one cargo can be taken at each time. The goods stored in the outer-layer of the storage rack will stop the retrieval of the inner storage unit. In this case, the inner-layer cargo can be accessed only

if the outer-layer cargo is moved first. We assume that both of the Pickup and Deposit (P&D) stations are at the same points at lower left-hand corner of the aisle. The movement, that is, the travelled distance, of the Storage and Retrieval (S/R) machine is a major decision parameter in the operation and control of the AS/RS. The distance travelled by the S/R machine, from address point a to point b to complete one activity, is calculated by the following two steps.

Step 1 (identification of row, column, and depth numbers). A simple table can be used to represent the row, column, and depth numbers of the addresses, that is, the storage locations of cargo (see Tables 1–5).

Step 2 (distance calculation for the movement from point a to point b). The distance between two storage location points a and b can be calculated using the general formula

$$\begin{aligned}\text{distance} = {} & P + E + M_a + 2 \cdot X_d \cdot a_d \\ & + \left[\sqrt{\left(X_r \cdot (a_r - b_r)\right)^2 + \left(X_c \cdot (a_c - b_c)\right)^2} \right. \\ & \left. + M_b + 2 \cdot X_d \cdot b_d \right],\end{aligned} \tag{1}$$

where P is distance from the P&D station to the first storage cell; M_a and M_b are the required time to move the outer-layer cargo when the outer-layer cargo stops the retrieval of the inner-layer cargo; X_r, X_c, and X_d are center distances between two adjacent cells in a row, column, and depth, respectively; a_r and b_r are distances between the locations "a" and "b" row-wise; a_c and b_c are distances between the locations "a" and "b" column-wise; a_d and b_d are distances between the locations "a" and "b" depth wise; and E is distance from the P&D station to last storage cell.

We set the P&D station to be at point $(0, 0, 0)$, and the formula for the P&D can be calculated as

$$\begin{aligned} P &= \sqrt{\left(a_{1r} \cdot X_r\right)^2 + \left(a_{1c} \cdot X_c\right)^2}, \\ E &= \sqrt{\left(a_{wr} \cdot X_r\right)^2 + \left(a_{wc} \cdot X_c\right)^2}, \end{aligned} \tag{2}$$

where X_r and X_c are center distances between two adjacent cells in a row, column, respectively, a_{1r} is distance between the location of first cargo "a" and "P&D station" row-wise; a_{1c} is distance between the location of first cargo "a" and the "P&D station" column-wise; w is the maximum load-capacity (i.e., maximum number of cargos) of the forklift truck each time; a_{wr} is distance between the location of last cargo "w" and the "P&D station" row-wise; a_{wc} is distance between the location of last cargo "w" and the "P&D station" column-wise.

When the outer-layer cargo stops the inner-layer cargo, we must first move the outer-layer cargo to the nearest empty place and then move back to the original place to retrieve the cargo. The formula for calculating M can be represented as follows:

$$M = \begin{cases} 2 \cdot \left(X_d \cdot (p_d + q_d) + \sqrt{\left(X_r \cdot (p_r - q_r)\right)^2 + \left(X_c \cdot (p_c - q_c)\right)^2} \right) \\ \qquad \text{if depth} = 2 \text{ when one different type} \\ \qquad\qquad \text{of cargo at depth} = 1, \\ 0, \qquad \text{otherwise}, \end{cases} \tag{3}$$

where X_r, X_c, and X_d are center distance between two adjacent cells in a row, column, and depth, respectively, p is the place of the cargo that we want to retrieve; q is the nearest empty place that we can put the cargo; p_r and q_r are distances between the locations "p" and "q" row-wise; p_c and q_c are distances between the locations "p" and "q" column-wise; p_d and q_d are distance between the locations "p" and "q" depth-wise.

The objective is to develop an optimal control program for the AS/RS in order to minimize the total distance travelled by the S/R machine. The objective function is formulated as follows:

$$\begin{aligned} &\text{total distance } Z \\ &= P + E + M_1 + 2 \cdot a_{1d} \cdot X_d \\ &\quad + \left\{ \left[\sqrt{\left((a_{1r} - a_{2r}) \cdot X_r\right)^2 + \left((a_{1c} - a_{2c}) \cdot X_c\right)^2} \right. \right. \\ &\qquad \left. + M_2 + 2 \cdot a_{2d} \cdot X_d \right] \\ &\qquad + \left[\sqrt{\left((a_{2r} - a_{3r}) \cdot X_r\right)^2 + \left((a_{2c} - a_{3c}) \cdot X_c\right)^2} \right. \\ &\qquad\quad \left. + M_3 + 2 \cdot a_{3d} \cdot X_d \right] + \cdots \\ &\qquad + \left[\sqrt{\left((a_{(w-1)r} - a_{wr}) \cdot X_r\right)^2 + \left((a_{(w-1)c} - a_{wc}) \cdot X_c\right)^2} \right. \\ &\qquad\quad \left. \left. + M_w + 2 \cdot a_{wd} \cdot X_d \right] \right\} \\ &=: P + E + M_1 + 2 \cdot a_{1d} \cdot X_d + \sum_{i=1}^{w-1} D_i, \end{aligned} \tag{4}$$

where Z is total distance travelled by the S/R machine; w is the maximum load capacity (i.e., maximum number of cargos) of the forklift truck each time.

3. Implementation of Evolutionary Algorithms

3.1. Genetic Algorithm (GA). The following GA procedure is adopted to perform the double-deep AS/RS retrieval scheduling optimization.

Step 1 (chromosome representation). Produce various combinations of different retrieval orders and give the number to represent them.

Table 1: The addresses of the stored cargo for simulation in Case 1.

Type and number of cargo	Address			Order		
	R	*C*	*D*	GA	IGA	PSO
A1	9	8	1	2	10	2
A2	8	20	1	8	6	8
A3	20	13	1	5	5	7
A4	19	10	2	7	2	4
A5	2	12	2	10	7	10
A6	19	12	2	4	4	6
A7	6	4	1	1	1	1
A8	21	9	1	6	3	5
A9	9	9	1	3	9	3
A10	7	11	2	9	8	9
B1	29	4	2	16	18	15
B2	17	14	1	14	15	14
B3	23	7	2	17	17	16
B4	21	3	2	15	20	18
B5	6	7	2	12	12	20
B6	13	9	2	19	14	12
B7	11	7	1	20	13	19
B8	13	17	2	13	16	13
B9	23	5	2	18	19	17
B10	4	8	1	11	11	11
C1	17	7	2	23	28	23
C2	1	11	1	29	22	30
C3	27	4	1	22	29	22
C4	16	16	2	24	27	24
C5	1	18	1	27	24	28
C6	20	2	1	21	30	21
C7	7	14	1	28	23	27
C8	1	10	1	30	21	29
C9	11	20	1	25	26	25
C10	6	20	1	26	25	26
D1	29	14	1	35	36	36
D2	4	3	1	40	31	31
D3	15	8	2	33	40	40
D4	3	20	2	38	32	32
D5	17	9	2	31	39	39
D6	28	18	2	34	35	37
D7	14	20	2	37	34	34
D8	17	11	1	32	38	38
D9	23	15	1	36	37	35
D10	8	19	1	39	33	33

Step 2 (parameter settings). Consider the following:

population size: 100;

select a maximum generation number $t_{\max} = 100$, and set $t = 0$ at the beginning;

crossover rate: 0.6;

mutation rate: 0.05.

Table 2: The addresses of the stored cargo for simulation in Case 2.

Type and number of cargo	Address			Order		
	R	*C*	*D*	GA	IGA	PSO
A1	9	8	2	9	10	2
A2	8	20	1	2	4	8
A3	20	13	1	6	5	7
A4	19	10	2	8	8	5
A5	2	12	2	1	3	9
A6	19	12	2	5	6	6
A7	6	4	1	10	1	1
A8	21	9	1	7	7	4
A9	9	9	1	4	9	3
A10	7	11	2	3	2	10
B1	29	4	2	15	18	16
B2	9	8	1	13	13	19
B3	23	7	2	17	17	14
B4	21	3	2	14	20	17
B5	6	7	2	20	12	20
B6	13	9	2	19	15	13
B7	11	7	1	12	14	18
B8	13	17	2	18	16	12
B9	23	5	2	16	19	15
B10	4	8	2	11	11	11
C1	17	7	2	30	22	30
C2	1	11	1	23	29	23
C3	27	4	1	28	24	29
C4	16	16	2	26	26	26
C5	1	18	1	24	28	24
C6	20	2	1	29	23	28
C7	19	12	1	27	25	27
C8	1	10	1	22	30	22
C9	11	20	1	25	27	25
C10	4	8	1	21	21	21
D1	29	14	1	36	35	37
D2	4	3	1	40	31	40
D3	15	8	2	39	34	31
D4	17	7	1	37	32	32
D5	17	9	2	38	33	33
D6	28	18	2	35	36	36
D7	14	20	2	32	39	38
D8	16	16	1	33	38	34
D9	23	15	1	34	37	35
D10	8	19	1	31	40	39

Step 3. According to population size, randomly select the retrieval order number and put it into the genetic pool.

Step 4. Calculate the cost value through the fitness function formula (4) for each gene.

Table 3: The addresses of the stored cargo for simulation in Case 3.

Type and number of cargo	Address			Order		
	R	*C*	*D*	GA	IGA	PSO
A1	9	8	2	1	2	3
A2	8	20	1	7	8	9
A3	20	13	1	5	7	7
A4	19	10	2	3	6	5
A5	2	12	2	8	10	10
A6	19	12	2	6	4	8
A7	6	4	1	10	1	1
A8	21	9	1	4	5	6
A9	9	9	1	2	3	4
A10	7	11	2	9	9	2
B1	29	4	2	15	17	17
B2	9	8	1	18	13	12
B3	23	7	2	13	16	18
B4	21	3	2	17	20	15
B5	6	7	2	19	11	11
B6	13	9	2	12	14	13
B7	21	3	1	16	19	14
B8	13	17	2	11	15	19
B9	23	5	2	14	18	16
B10	4	8	2	20	12	20
C1	17	7	2	28	30	22
C2	13	17	1	25	25	28
C3	27	4	1	29	28	24
C4	16	16	2	26	26	26
C5	1	18	1	23	23	29
C6	20	2	1	30	29	23
C7	19	12	1	27	27	25
C8	1	10	1	22	22	30
C9	11	20	1	24	24	27
C10	4	8	1	21	21	21
D1	29	14	1	34	35	35
D2	19	10	1	33	36	36
D3	15	8	2	39	40	40
D4	17	7	1	38	38	38
D5	17	9	2	32	37	37
D6	28	18	2	35	34	34
D7	14	20	2	36	33	33
D8	16	16	1	37	32	32
D9	7	11	1	40	31	31
D10	15	8	1	31	39	39

Table 4: The addresses of the stored cargo for simulation in Case 4.

Type and number of cargo	Address			Order		
	R	*C*	*D*	GA	IGA	PSO
A1	9	8	2	1	4	3
A2	8	20	1	6	8	7
A3	20	13	1	7	7	5
A4	19	10	2	9	5	4
A5	2	12	2	4	10	9
A6	19	12	2	8	6	6
A7	6	4	1	10	1	10
A8	2	12	1	3	9	8
A9	9	9	1	2	3	2
A10	7	11	2	5	2	1
B1	29	4	2	17	17	17
B2	9	8	1	20	12	20
B3	23	7	2	18	18	19
B4	21	3	2	15	15	16
B5	6	7	2	12	20	12
B6	13	9	2	19	13	14
B7	21	3	1	14	14	15
B8	13	17	2	13	19	13
B9	23	5	2	16	16	18
B10	4	8	2	11	11	11
C1	17	7	2	28	29	23
C2	13	17	1	22	23	29
C3	27	4	1	26	27	25
C4	16	16	2	23	24	28
C5	23	7	1	25	25	26
C6	29	4	1	27	26	24
C7	19	12	1	24	28	27
C8	1	10	1	21	22	30
C9	6	7	1	29	30	22
C10	4	8	1	30	21	21
D1	29	14	1	34	34	34
D2	19	10	1	36	36	36
D3	15	8	2	40	40	40
D4	17	7	1	37	38	39
D5	17	9	2	38	37	37
D6	23	5	1	35	35	35
D7	14	20	2	33	32	32
D8	16	16	1	32	33	33
D9	7	11	1	31	31	31
D10	15	8	1	39	39	38

Step 5. Calculate the probability to be selected by the following formulas:

$$X_k = \frac{\sum_{i=1}^{n} f_i}{f_k}, \tag{5}$$

where f_k is the fitness value for the kth gene in population;

$$P_k = \frac{X_k}{\sum_{i=1}^{n} X_i}, \tag{6}$$

where P_k is the probability of the kth gene to be selected for reproduction.

According to the P_k, we proceed to select genes with the roulette wheel method.

Step 6. Crossover.

Step 7. Mutation.

Table 5: The addresses of the stored cargo for simulation in Case 5.

Type and number of cargo	Address			Order		
	R	*C*	*D*	GA	IGA	PSO
A1	9	8	2	9	9	1
A2	8	20	2	5	4	5
A3	20	13	2	8	5	4
A4	19	10	2	7	7	2
A5	2	12	2	4	1	7
A6	19	12	2	6	6	3
A7	6	4	2	10	10	10
A8	2	12	1	3	2	6
A9	9	9	2	1	8	9
A10	7	11	2	2	3	8
B1	29	4	2	16	16	15
B2	9	8	1	11	11	20
B3	23	7	2	17	17	17
B4	21	3	2	14	15	14
B5	6	7	2	19	19	12
B6	13	9	2	12	12	19
B7	21	3	1	13	13	13
B8	13	17	2	18	18	18
B9	23	5	2	15	14	16
B10	4	8	2	20	20	11
C1	17	7	2	22	29	29
C2	13	17	1	27	24	24
C3	6	4	1	21	30	30
C4	16	16	2	26	25	25
C5	23	7	1	23	28	27
C6	29	4	1	24	27	28
C7	19	12	1	25	26	26
C8	8	20	1	28	23	23
C9	6	7	1	30	22	22
C10	4	8	1	29	21	21
D1	13	9	1	38	40	40
D2	19	10	1	34	35	36
D3	15	8	2	37	39	39
D4	17	7	1	31	37	37
D5	9	9	1	40	31	32
D6	23	5	1	32	36	35
D7	20	13	1	33	34	34
D8	16	16	1	35	33	33
D9	7	11	1	39	32	31
D10	15	8	1	36	38	38

Step 8. Set $t = t + 1$ and go to Step 4, recursively. Stop if $t < t_{\max}$.

The flow chart of GA is shown in Figure 4.

3.2. Immune Genetic Algorithm (IGA). The IGA procedure is improved from GA, and the following IGA procedure is adopted to perform the double-deep AS/RS retrieval scheduling optimization.

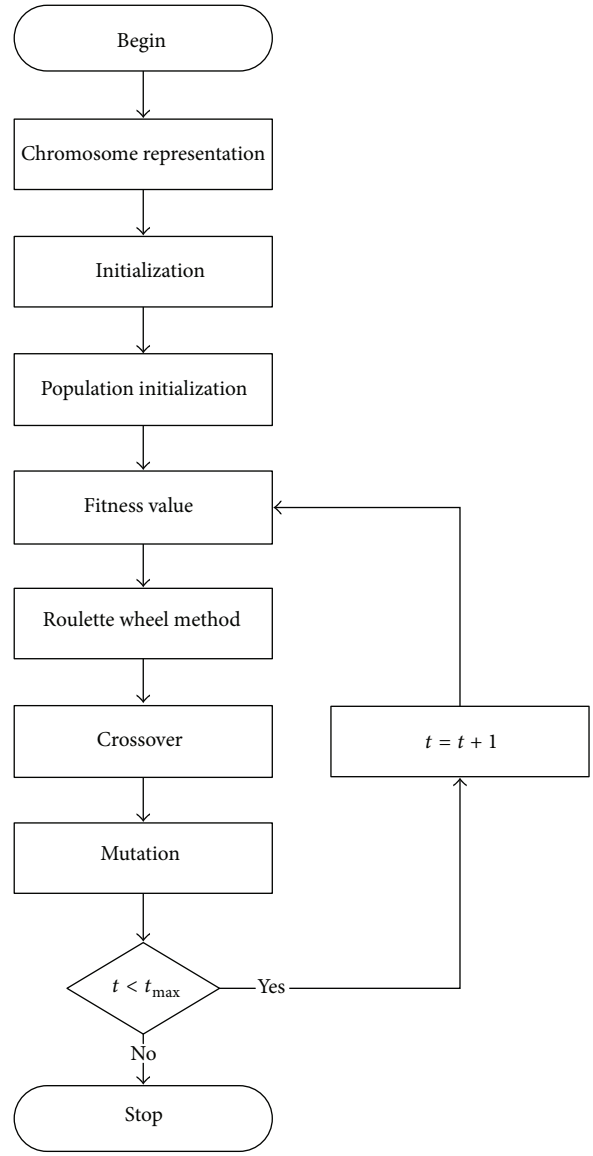


Figure 4: Flow chart of the Genetic Algorithm (GA).

Step 1 (chromosome representation). Produce various combinations of different retrieval orders and give the number to represent them.

Step 2 (parameter settings). Consider the following:

population size: 100;

select a maximum generation number $t_{\max} = 100$, and set $t = 0$ at the beginning;

crossover rate: 0.6;

mutation rate: 0.05.

Step 3. According to population size, randomly select the retrieval order number and put it into the genetic pool.

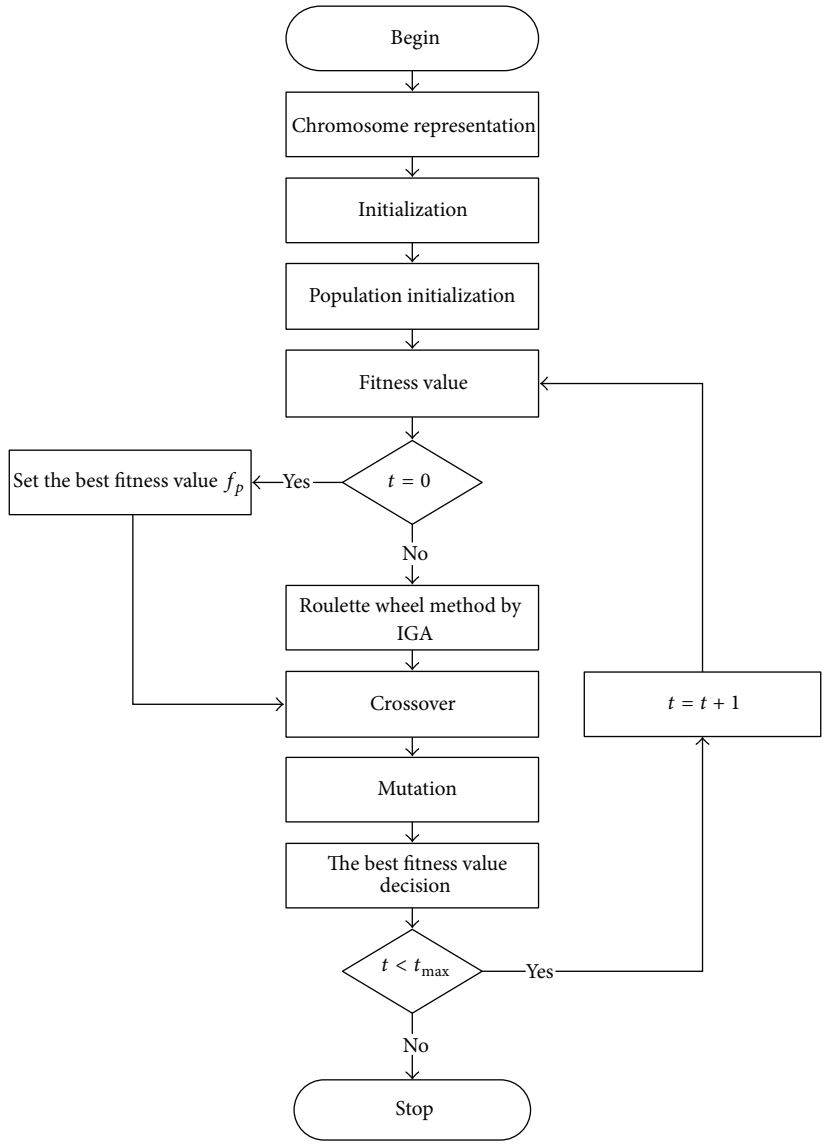


Figure 5: Flow chart of the Immune Genetic Algorithm (IGA).

Step 4. Calculate the cost value through the fitness function formula (4) for each gene.

Step 5. Set the best fitness value f_p.

Step 6. Calculate the probability of being selected with the following formula:

$$X_k = \frac{\sum_{i=1}^{n} f_i}{f_k}, \tag{7}$$

where f_k is the fitness value for the kth gene in population.

Evaluate the affinity between any pairs of strings with the following formula:

$$1 - \varepsilon \le Q_s(u, v) = \frac{X_u}{X_v} \le 1 + \varepsilon, \tag{8}$$

where $Q_s(u, v)$ is similarity, $u = \{u_1, u_2, \ldots, u_n\}$ and $v = \{v_1, v_2, \ldots, v_n\}$ are any two strings in population, and the fitness values for u_i and v_i are X_u and X_v, respectively; ε is the threshold value for gene similarity, and $\varepsilon > 0$ where we set $\varepsilon = 0.02$.

Calculate the expected rate of reproduction as follows:

$$e_k = \frac{X_k}{(C_k)^{\beta}}, \tag{9}$$

where e_k is the expected rate of reproduction; X_k is the estimate value for the kth string; C_k is the total number similar to the kth string in all the strings in the generation; β is the parameter to tune the X_k and C_k.

According to (10), we calculate the probability to be selected for each gene as follows:

$$P_{ik} = \frac{e_k}{\sum_{i=1}^{n} e_i}, \tag{10}$$

where P_{ik} is the probability of the kth gene being selected for reproduction; e_k is the expected rate of reproduction.

According to the P_{ik}, we proceed to select genes by using the roulette wheel method.

Step 7. Crossover.

Step 8. Mutation.

Step 9. If the string is with the best fitness value, and the fitness value is not equal to f_p, then replace the string with the worst fitness value by using f_p.

If the current fitness value is better than the best fitness value in history (f_p), we use the current value as the new best fitness value f_p.

Step 10. Set $t = t + 1$ and go to Step 4, recursively. Stop if $t < t_{\max}$.

The flow chart of IGA is shown in Figure 5.

3.3. Particle Swarm Optimization (PSO). The following PSO [33] procedure is adopted to perform the double-deep AS/RS retrieval scheduling optimization.

Step 1 (chromosome representation). Produce various combinations of different retrieval orders and give the number to represent them.

Step 2 (parameter settings). Consider the following:

learning factor $c_1 = 2$;

learning factor $c_2 = 2$;

population size: 100;

select a maximum generation number $t_{\max} = 100$, and set $t = 0$ at the beginning;

maximum velocity: $V_{\max}$.

Step 3. According to population size, randomly select the retrieval order number and regard it as the initial value of particle.

Step 4. Calculate the cost value with the fitness function formula (4) for each particle.

Step 5. Calculate the fitness value for each particle. If the fitness value is better than the best fitness value in history, we use the current value as the new best fitness value f_p.

Step 6. Select the particle with the best fitness value of all the particles as the global best fitness value f_g.

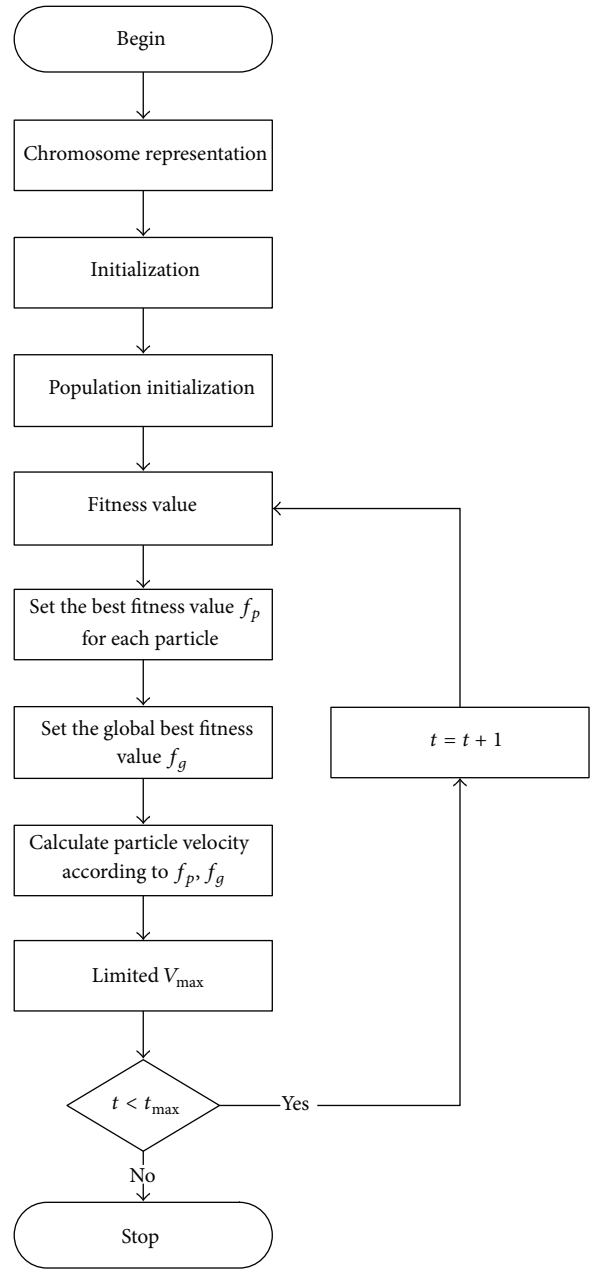


Figure 6: Flow chart of the Particle Swarm Optimization (PSO).

Step 7. For each particle, calculate particle velocity according to the following equation:

$$v = c_1 * r(\cdot) * \left(f_p - p_c\right) + c_2 * r(\cdot) * \left(f_g - p_c\right), \tag{11}$$

where v, simplified from $v[t]$, is the particle velocity; p_c, simplified from $p_c[t]$, is the current particle position; $p_c[t+1] = p_c[t] + v[t]$; $r(\cdot)$ is a random function 9 in the range $[0, 1]$; c_1, c_2 are learning factors = 2.

Step 8. Particle velocities on each dimension are bounded by a maximum velocity $V_{\max}$. If the sum of acceleration causes the velocity on that dimension to exceed $V_{\max}$ (specified by the user), the velocity on the dimension is limited to $V_{\max}$.

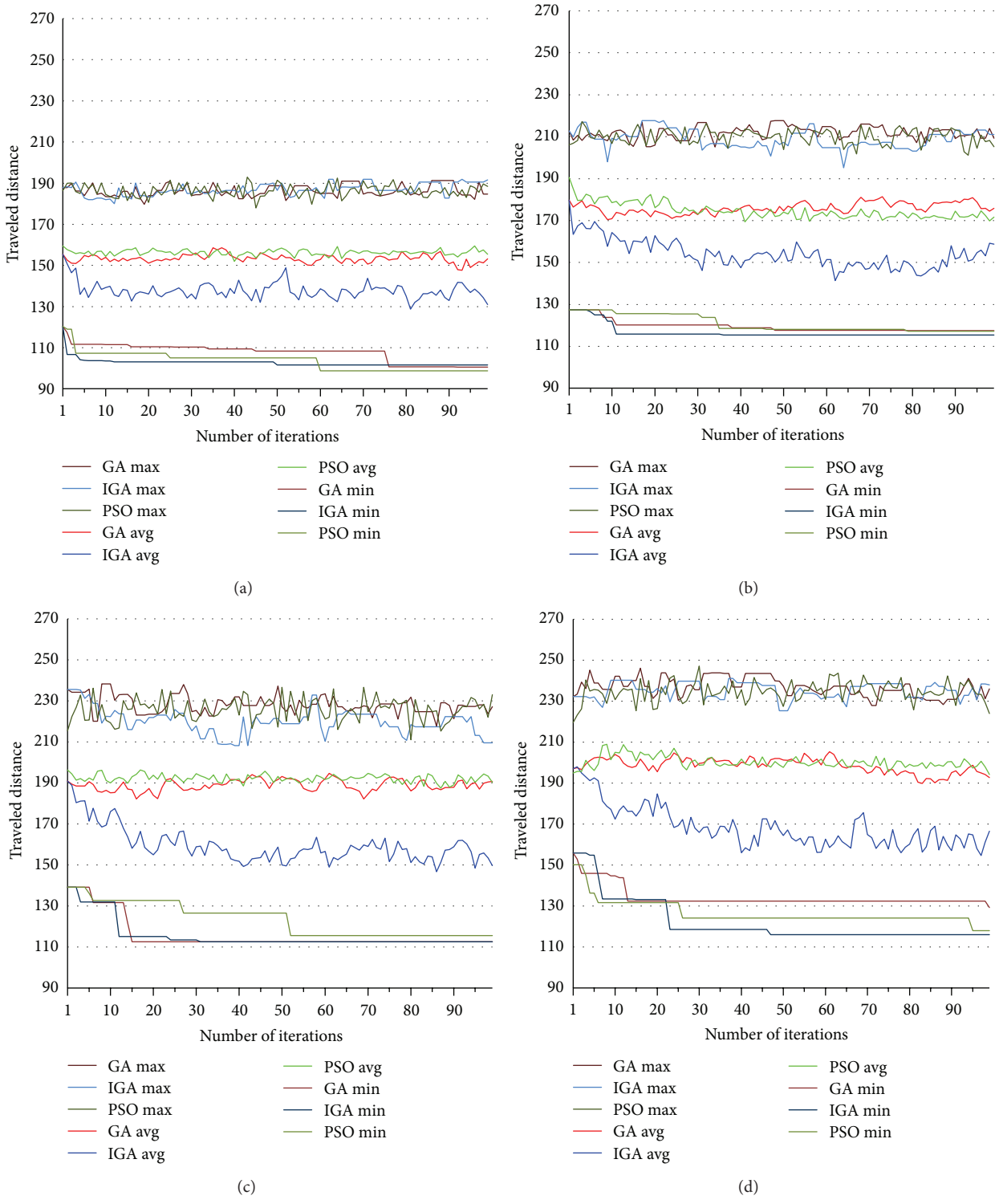


FIGURE 7: The shortest traveled distance for simulation results for Case 1. All the goods do not overlap by GA, IGA, and PSO. (a) Goods type A, (b) goods type B, (c) goods type C, and (d) goods type D.

Step 9. Set $t = t + 1$ and go to Step 4, recursively. Stop if $t < t_{\max}$.

The flow chart of PSO is shown in Figure 6.

4. Simulation and Discussion

In the simulation, MATLAB (7.10.0 (R2010a), 64 bits) is used to analyze the performance. The specifications of the PC

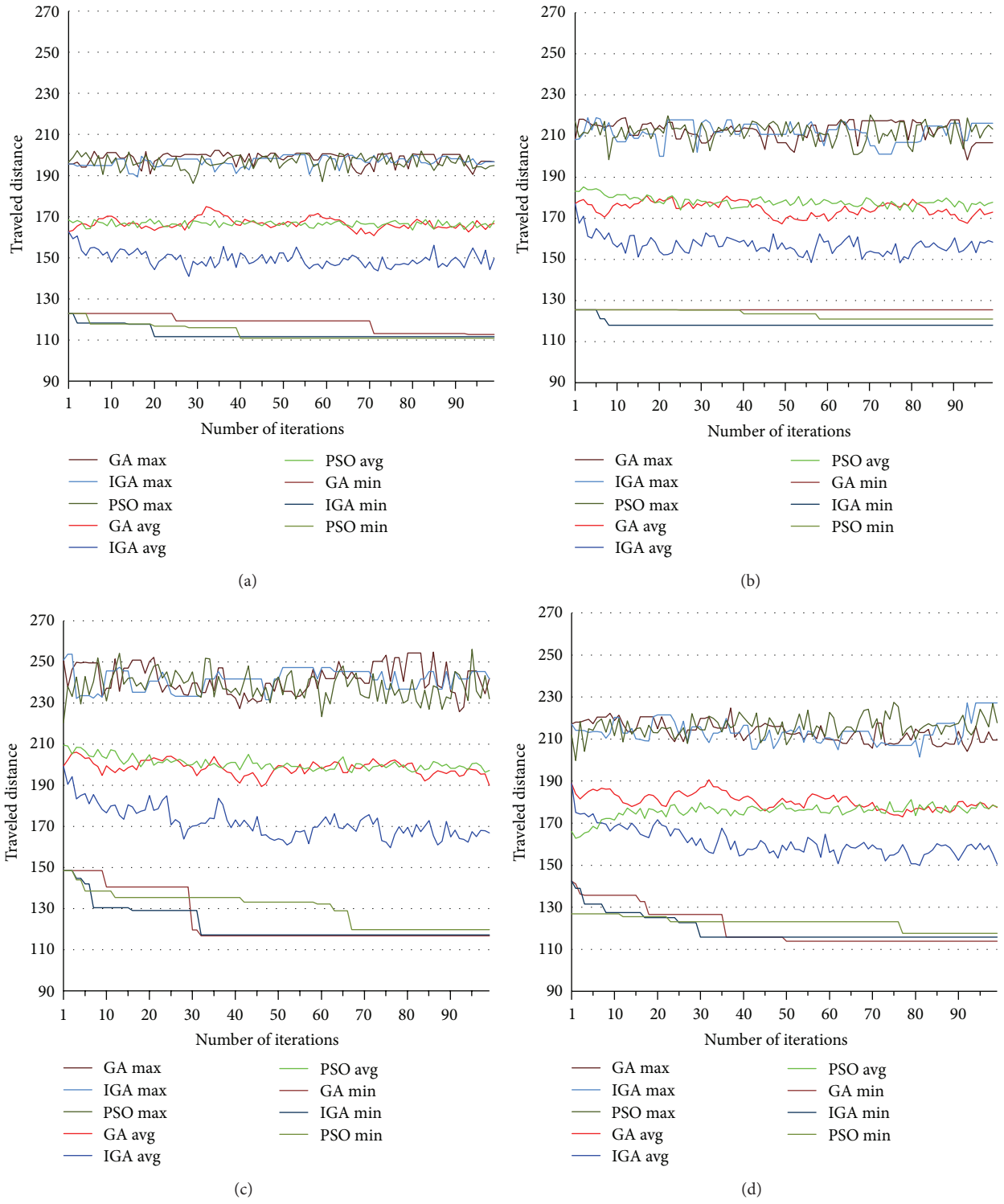


FIGURE 8: The shortest traveled distance for simulation results for Case 2. 5 goods overlap by GA, IGA, and PSO. (a) Goods type A, (b) goods type B, (c) goods type C, and (d) goods type D.

are CPU: Intel I7-3770, quad-core, 3.4 GHz (Turbo 4.2 GHz); RAM: 16 GB, DDR3, 1600 MHz, CL 11; OS: Windows 7 Enterprise, 64 bits. The chosen AS/RS structure model is shown in Section 2 with m rows, n columns, and 2 depths, where m is set to be 20 and n is set to be 30. There are 4 types of goods and 10 for each type. The Genetic Algorithm (GA), Immune Genetic Algorithm (IGA), and Particle Swarm Optimization (PSO) are applied. In GA and IGA, we select a coding to

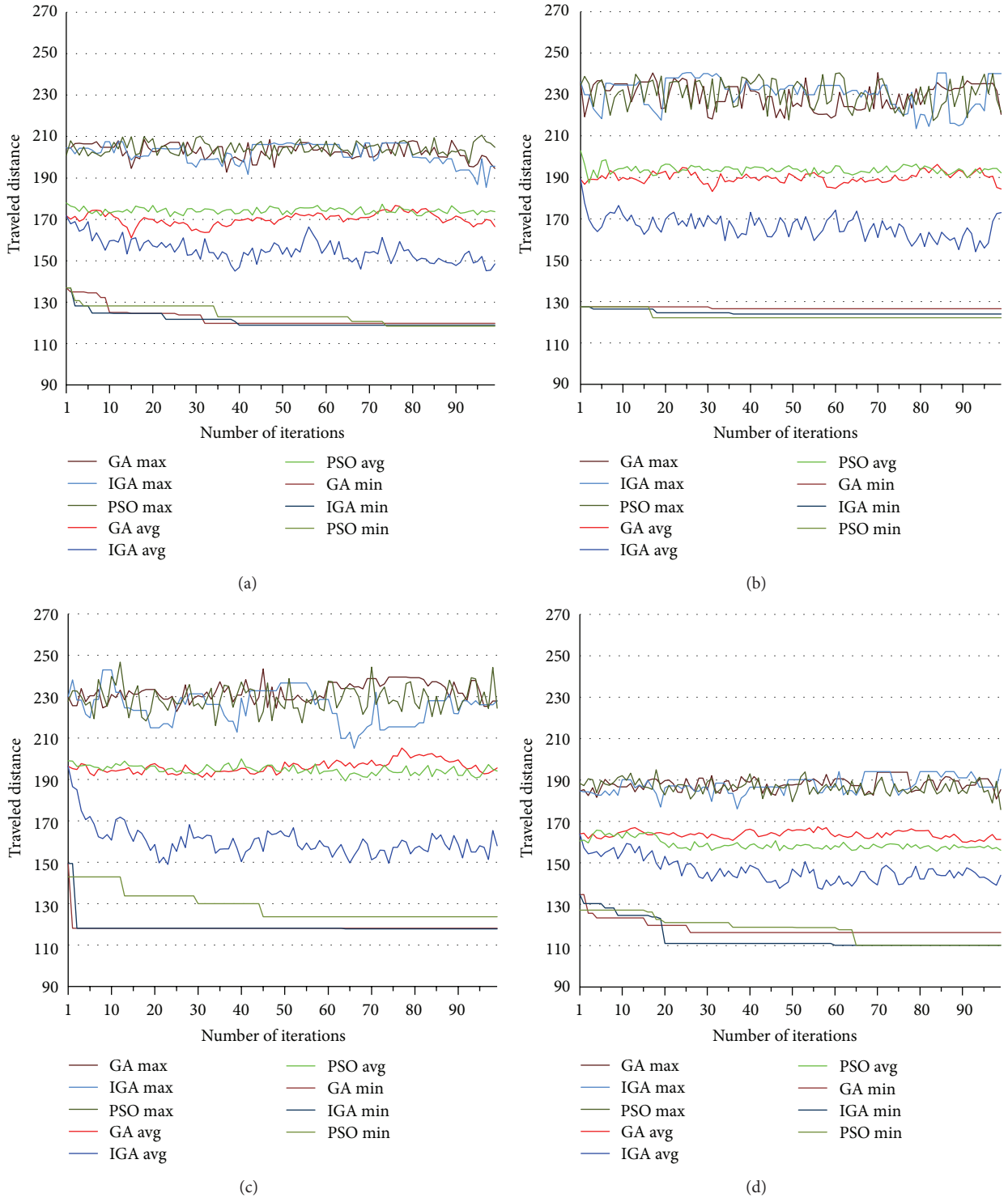


FIGURE 9: The shortest traveled distance for simulation results for Case 3. 10 goods overlap by GA, IGA, and PSO. (a) Goods type A, (b) goods type B, (c) goods type C, and (d) goods type D.

represent problem parameters and choose population size to be 100, crossover probability to be 0.6, and mutation probability to be 0.05. Additionally, we initialize a random population of strings and each string has a size of 22 bits. In PSO, we randomly initialize a population of 100 particles and the learning factors are set to be 2. The relevant parameters and details are presented in Section 3. Tables 1–5 illustrate random examples with the possible addresses of cargo stored in the AS/RS with the fixed overlapped number. The R, C, and D represent row, column, and depth, respectively.

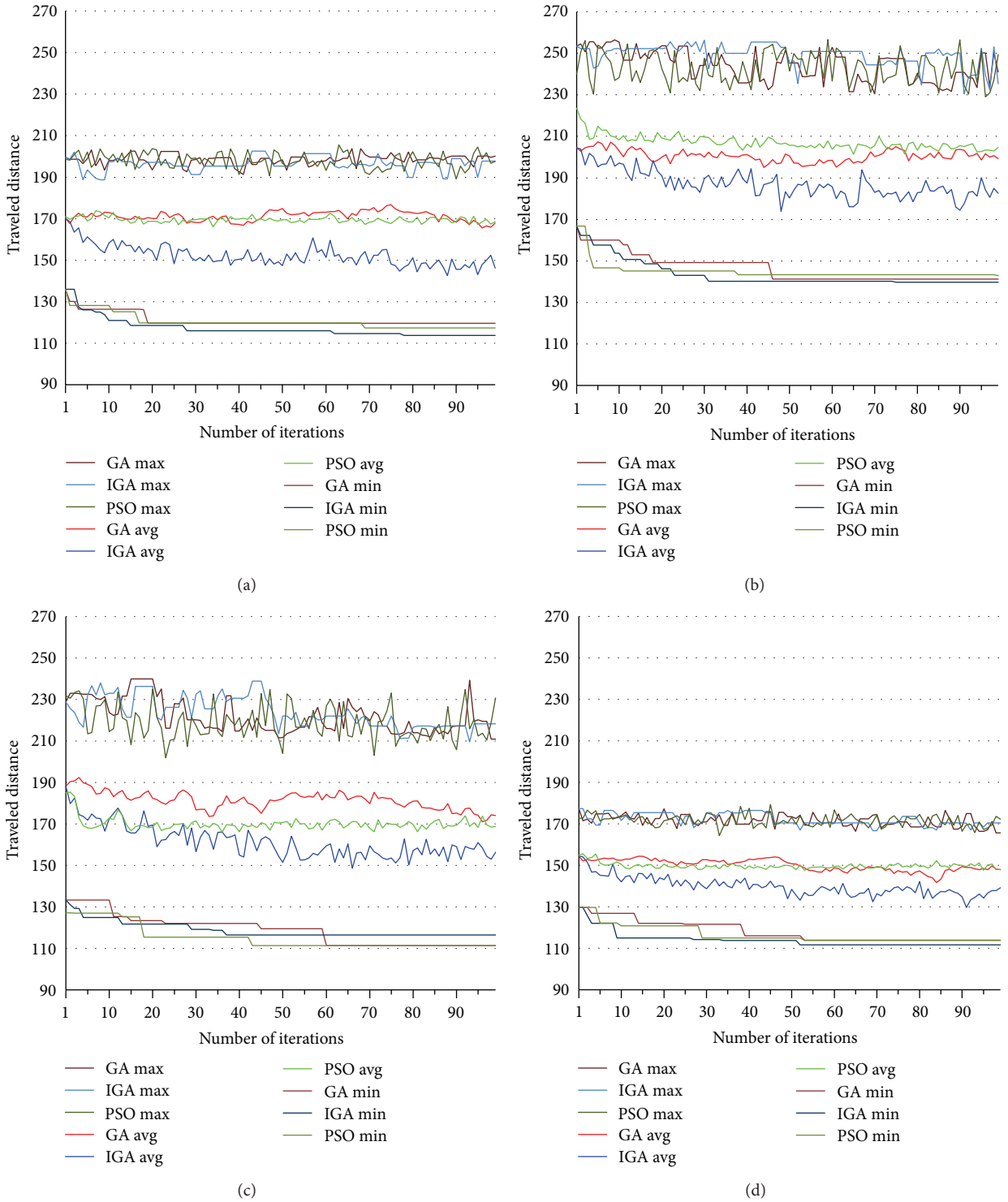


FIGURE 10: The shortest traveled distance for simulation results for Case 4. 15 goods overlap by GA, IGA, and PSO. (a) Goods type A, (b) goods type B, (c) goods type C, and (d) goods type D.

Software framework is used to run 100 iterations to obtain optimal results. Five cases are studied—Case 1: all goods do not overlap at the front and rear positions, as shown in Table 1; Case 2: five goods sets overlap, as shown in Table 2; Case 3: ten goods sets overlap, as shown in Table 3; Case 4: fifteen goods sets overlap, as shown in Table 4; and Case 5: twenty goods sets overlap, as shown in Table 5. For example, in Table 2, five goods sets overlap; that is, Cargo B2 overlaps A1 at the place ([9, 8, 1], [9, 8, 2]), Casrgo C7 overlaps A6 at the place ([19, 12, 1], [19, 12, 2]), Casrgo D8 overlaps C4

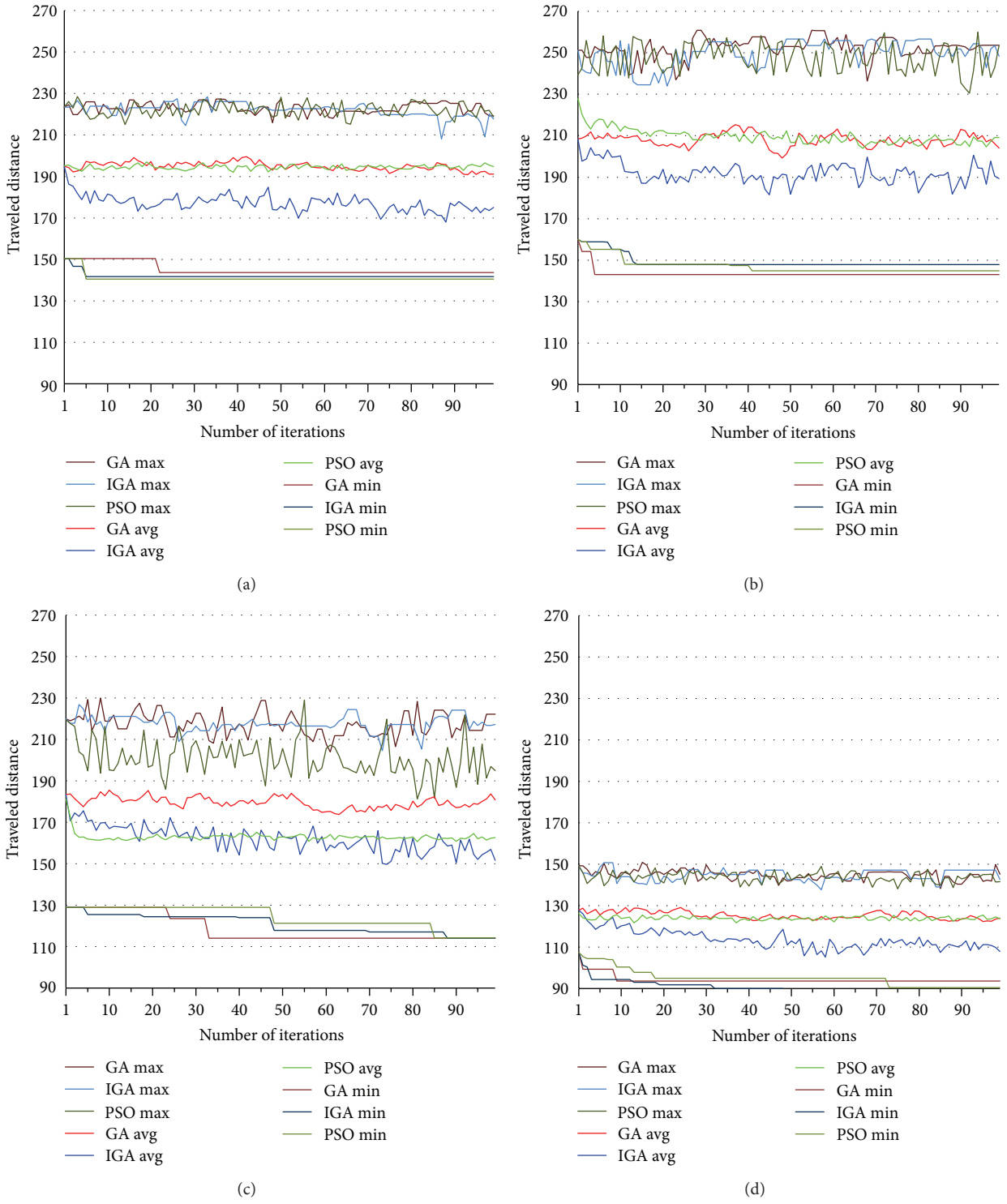


Figure 11: The shortest traveled distance for simulation results for Case 5. 20 goods overlap by GA, IGA, and PSO. (a) Goods type A, (b) goods type B, (c) goods type C, and (d) goods type D.

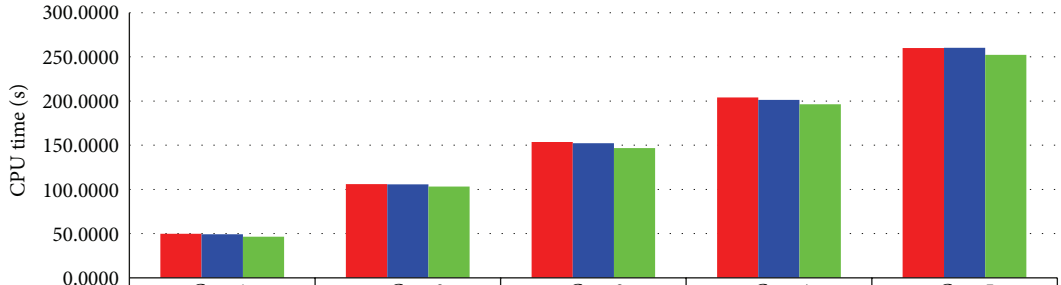
at the place ([16, 16, 1], [16, 16, 2]), Casrgo D4 overlaps C1 at the place ([17, 7, 1], [17, 7, 2]), and Casrgo C10 overlaps B10 at the place ([4, 8, 1], [4, 8, 2]). The optimal scheduling orders for five cases with three algorithms can be obtained and are shown in the "Order" column of Tables 1–5. For example, in Table 1 for Case 1 by GA method, we can get the optimal retrieval order as (1) Cargo A7, (2) Cargo A2, (3) Cargo A9, . . ., and finally (40) Cargo D2. The simulation results for these five different cases are illustrated in Figures 7, 8, 9, 10, and 11. The five cases, from Case 1 to Case 5, have

Table 6: The CPU time (sec) under different cases and by three different algorithms.

	Average	Maximum	Minimal	Max-min	Standard deviation
Case 1					
GA	49.80	61.29	29.98	31.31	5.23
IGA	49.22	59.65	29.95	29.70	5.18
PSO	46.44	57.00	28.39	28.61	4.86
Case 2					
GA	106.03	148.03	91.60	56.43	16.70
IGA	105.61	148.15	91.59	56.57	16.34
PSO	103.12	139.76	89.34	50.42	15.23
Case 3					
GA	153.52	212.33	133.57	78.76	21.95
IGA	152.30	223.71	133.51	90.20	22.32
PSO	146.84	206.33	130.14	76.19	20.58
Case 4					
GA	204.05	273.58	179.03	94.55	26.72
IGA	201.43	286.81	176.25	110.56	27.30
PSO	196.30	293.83	173.33	120.50	26.84
Case 5					
GA	260.06	381.33	233.78	147.55	33.58
IGA	260.26	373.89	233.46	140.43	33.13
PSO	252.24	352.52	228.18	124.33	30.20

been calculated using 100 iterations for each time, as well as a total of 100 times, to obtain an average CPU time for the analysis, as illustrated in Table 6 and Figure 12. In Table 6, the average CPU time, maximum CPU time, minimum CPU time, the difference in the maximum and minimum values, and the standard deviation are all illustrated. The smaller the standard deviation is, the more concentrative the average CPU time will be for all 100 data by running the process 100 times. Therefore, we can see that PSO will cost less average CPU time in each case, and that PSO will almost get the smallest standard deviation. Although the CPU time for the three algorithms is very similar in the five illustrated cases, regarding CPU time, PSO performs better than the other two methods. With increasing complexity, computing time will also increase. However, PSO still performs the best. This can be easily found in Table 6 and Figure 12. In Figures 7–11, we run 100 iterations for each time and show the best curve (i.e., with the shortest distance at the 100th iteration) out of the 100 times as the minimum curve. Although the three types of algorithms result in little difference in convergent speeds under different cases, all the travelled distances tend to converge. In order to create criteria for the selection of algorithm in the figures, we use the similar way in [38] to analyze the curve tendency. The maximum and average travel distance values out of the genes by the three algorithms are illustrated in Figures 7–11. We can find that the maximum values for the three algorithms are almost the same. However, the IGA average curve shows a lower traveled distance than the other two algorithms. We can therefore infer that, by IGA, we will have a higher probability to select the gene with the shortest travelled distance and finally obtain the shortest travelled distance. The different travelled distances are shown in the Gantt charts as illustrated in Figures 13–17. The charts clearly show that retrieval scheduling with three evolutionary algorithms performs much better than that with random retrieval. In general, the shortest travelled distance represents the shortest time to complete the job. IGA performs best based on the criteria stated earlier, as shown in Figures 7–11 and 13–17. Therefore, the user can select the most suitable method for optimal retrieval scheduling in a double-deep AS/RS either based on the viewpoint of CPU time or the travelled distance. It can also be observed in Figures 7–11 that there is chattering shown in maximum and average values since there are crossover and mutation in GA and IGA and there is calculation of particle velocity in PSO. A decreasing tendency in minimum values results from the fact that the best gene will be kept in each iteration for GA, IGA, and PSO.

In our simulation results, approximate convergence can be achieved randomly at, for example, the 50th, the 80th, and the 90th generations in different curves. Therefore, we perform the iterations 100 times. If one were to set the iteration at only the 80th generation, it would perhaps get suboptimal solutions, but it would save CPU time. However, if one were to set the iteration at the 3000th generation, it might obtain optimal solutions, but it would perhaps waste significant CPU time. We could also establish a rule to determine the number of iterations. For example, if there is no change in travel distance or limited travel distance after several iterations have passed from some iteration instant, we thus stop and use this as the result. Under this situation, the iteration number is set and modified by ourselves or is dependent on the experience. In this paper, the proposed ideas and methods focus mainly on double-deep AS/RS retrieval scheduling. However, it can



	Case 1	Case 2	Case 3	Case 4	Case 5
GA	49.7983	106.0339	153.5212	204.0499	260.0596
IGA	49.2238	105.6096	152.2997	201.4332	260.2566
PSO	46.4395	103.1196	146.8405	196.2993	252.2393

Figure 12: The CPU time (sec) under different cases and by three different algorithms.

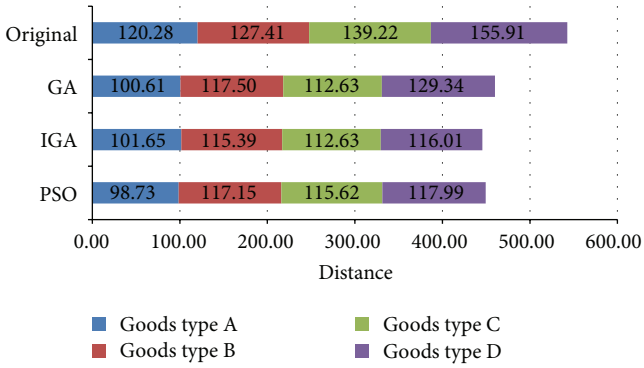


Figure 13: The Gantt chart for different travelled distances under Case 1.

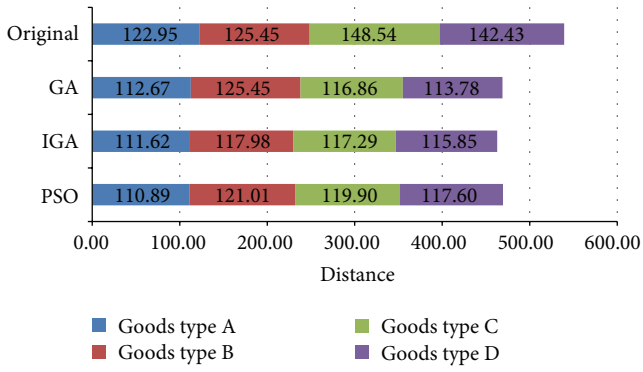


Figure 14: The Gantt chart for different travelled distances under Case 2.

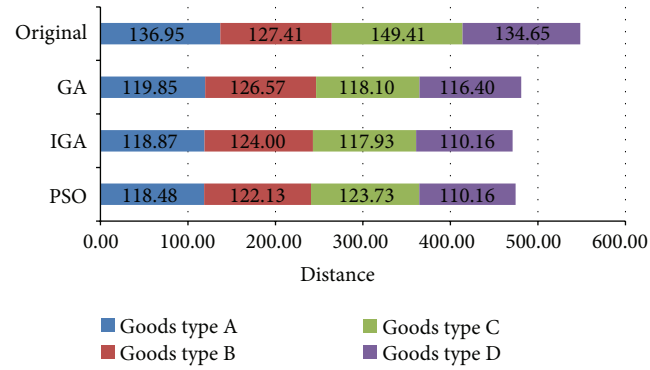


Figure 15: The Gantt chart for different travelled distances under Case 3.

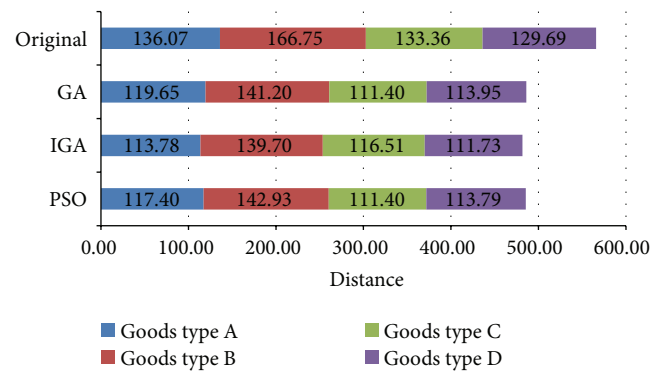


Figure 16: The Gantt chart for different travelled distances under Case 4.

easily be similarly extended to complicated systems with multiple objectives in real applications.

5. Conclusions

A survey of the literature indicates that optimal retrieval scheduling for double-deep type Automated Storage and Retrieval Systems (AS/RS) has not previously been studied. In this paper, a systematic model for the travelled distance, that is, the shortest time, is presented. Three types of evolutionary algorithms, including the Genetic Algorithm (GA), the Immune Genetic Algorithm (IGA), and the Particle Swarm Optimization (PSO), are used to optimize the retrieval scheduling problem for the studied case. With the minimum objective function value, that is, the shortest retrieval time travelled by the Storage and Retrieval (S/R) machine, the corresponding optimal retrieval schedule can be selected. In this simulation with five different cases, PSO costs less CPU

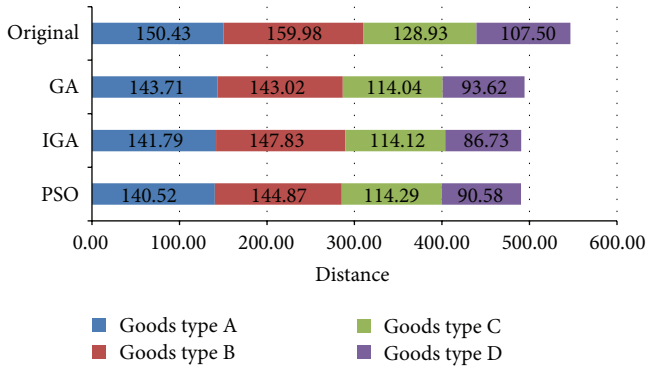


Figure 17: The Gantt chart for different travelled distances under Case 5.

time than GA or IGA. Although the three types of algorithms result in little difference in convergent speeds under different cases, all the travelled distances tend to converge. In terms of CPU time, PSO performs better than the other two methods. In terms of the shortest travelled distance, IGA performs better than the other two methods. Therefore, the user can select a suitable method for optimal retrieval scheduling in a double-deep AS/RS. Based on this study, extension to more complicated systems with multiple objectives can thus be further investigated for real applications in the future.

Acknowledgment

This research was supported by the National Science Council (NSC), Taiwan, under the Grant 101-2221-E-011-056.

References

[1] O. V. K. Chetty and M. S. Reddy, "Genetic algorithms for studies on AS/RS integrated with machines," *International Journal of Advanced Manufacturing Technology*, vol. 22, no. 11-12, pp. 932–940, 2003.

[2] W. Wang, W. Fu, M. Mingyun, D. Li, and S. Yang, "Selection of AS/RS scheduling rules based on genetic algorithm," in *Proceedings of the IEEE International Conference on Automation and Logistics (ICAL '07)*, pp. 536–540, Jinan, China, August 2007.

[3] P. Asokan, J. Jerald, S. Arunachalam, and T. Page, "Application of adaptive genetic algorithm and particle swarm optimisation in scheduling of jobs and AS/RS in FMS," *International Journal of Manufacturing Research*, vol. 3, no. 4, pp. 393–405, 2008.

[4] Q. Tang and F. Xie, "An approach for picking optimization in automated warehouse," in *Proceedings of the 5th International Conference on Natural Computation (ICNC '09)*, pp. 362–366, August 2009.

[5] W. Liu, Y. Huang, and P. Ding, "Research of compound operation optimization based on partheno genetic algorithm," in *Proceedings of the 3rd International Conference on BioMedical Engineering and Informatics (BMEI '10)*, pp. 2945–2948, October 2010.

[6] Y. Huang, W. Liu, P. Ding, and H. Liu, "Research on the compound operation optimization problem of AS/RS," in *Proceedings of the IEEE 5th International Conference on Bio-Inspired Computing: Theories and Applications*, pp. 1130–1133, September 2010.

[7] W. Yan, "The study on the simulation of a distribution center operating system and optimization of equipment configuration," in *Proceedings of the Chinese Control and Decision Conference*, pp. 3452–3456, May 2010.

[8] G. Zhou and L. Mao, "Design and simulation of storage location optimization module in AS/RS based on FLEXSIM," *International Journal of Intelligent Systems and Applications*, vol. 2, pp. 33–40, 2010.

[9] M. A. Tawhid, "Nonsmooth generalized complementarity as unconstrained optimization," *Journal of Industrial and Management Optimization*, vol. 6, no. 2, pp. 411–423, 2010.

[10] J. Zhou, N. Xiu, and J.-S. Chen, "Solution properties and error bounds for semi-infinite complementarity problems," *Journal of Industrial and Management Optimization*, vol. 9, no. 1, pp. 99–115, 2013.

[11] J. Zhou, J.-S. Chen, and G. M. Lee, "On set-valued complementarity problems," *Abstract and Applied Analysis*, vol. 2013, Article ID 105930, 11 pages, 2013.

[12] S. Pan, S. Bi, and J. S. Chen, "Nonsingularity conditions for FB system of reformulating nonlinear second-order cone programming," *Abstract and Applied Analysis*, vol. 2013, Article ID 602735, 21 pages, 2013.

[13] B. Xing, W.-J. Gao, F. V. Nelwamondo, K. Battle, and T. Marwala, "Storage and retrieval machine travel route planning strategy in an automated material handling environment," in *Proceedings of the World Automation Congress*, September 2010.

[14] W. Yang, X. Qiu, and H. Wang, "The research of hybrid scheduling model in AS/RS based on multi-agent system and Petri Net," in *Proceedings of the International Conference of Information Science and Management Engineering*, pp. 120–124, August 2010.

[15] A. Zheng, W. Liu, C. Lu, H. Wang, and C. Song, "Modeling and optimizing research for the steel tube AS/RS system," in *Proceedings of the Chinese Control and Decision Conference*, pp. 3979–3983, Mianyang, China, May 2011.

[16] C. Huang and S. Fu, "Research on movement control technology of Stacking Crane," in *Proceedings of the International Conference on Electric Information and Control Engineering*, pp. 5254–5256, April 2011.

[17] M. Rajkumar, P. Asokan, N. Anilkumar, and T. Page, "A GRASP algorithm for flexible job-shop scheduling problem with limited resource constraints," *International Journal of Production Research*, vol. 49, no. 8, pp. 2409–2423, 2011.

[18] O. Casdin, P. Castagna, Z. Sari, and N. Meghelli, "Performance evaluation of in-deep class storage for flow-rack AS/RS," *International Journal of Production Research*, vol. 50, pp. 22–24, 2012.

[19] http://www.econorack.com/double_deep.

[20] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.

[21] M. V. Raj, S. S. Sankar, and S. G. Ponnambalam, "Genetic algorithm to optimize manufacturing system efficiency in batch selective assembly," *International Journal of Advanced Manufacturing Technology*, vol. 57, no. 5–8, pp. 795–810, 2011.

[22] M. Abbasi and M. Houshmand, "Production planning and performance optimization of reconfigurable manufacturing systems using genetic algorithm," *International Journal of Advanced Manufacturing Technology*, vol. 54, no. 1–4, pp. 373–392, 2011.

[23] W. Wang and Y. Koren, "Scalability planning for reconfigurable manufacturing systems," *Journal of Manufacturing Systems*, vol. 31, no. 2, pp. 83–91, 2012.

[24] K. Xing, L. Han, M. Zhou, and F. Wang, "Deadlock-free genetic scheduling algorithm for automated manufacturing systems based on deadlock control policy," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 42, no. 3, pp. 603–615, 2012.

[25] O. A. Arqub, Z. Abo-Hammour, S. Momani, and N. Shawagfeh, "Solving singular two-point boundary value problems using continuous genetic algorithm," *Abstract and Applied Analysis*, vol. 2012, Article ID 205391, 25 pages, 2012.

[26] N. K. Jerne, "The immune system," *Scientific American*, vol. 229, no. 1, pp. 52–60, 1973.

[27] J. D. Farmer, N. H. Packard, and A. S. Perelson, "The immune system, adaptation, and machine learning," *Physica D*, vol. 22, no. 1–3, pp. 187–204, 1986.

[28] L. Jiao and L. Wang, "A novel genetic algorithm based on immunity," *IEEE Transactions on Systems, Man, and Cybernetics A*, vol. 30, no. 5, pp. 552–561, 2000.

[29] J. Timmis and L. N. D. Castro, *Artificial Immune Systems: A New Computational Intelligence Approach*, Springer, 2002.

[30] S. Yang, "A comparative study of immune system based genetic algorithms in dynamic environments," in *Proceedings of the 8th Annual Genetic and Evolutionary Computation Conference*, pp. 1377–1384, July 2006.

[31] L. Yan and K. Yang, "Immunity genetic algorithm based on elitist strategy and its application to the TSP problem," in *Proceedings of the International Workshop on Education Technology and Training and 2008 International Workshop on Geoscience and Remote Sensing*, vol. 2, pp. 729–732, December 2008.

[32] T.-C. Chen and Y.-C. Hsieh, "Using immune-based genetic algorithms for single trader's periodic marketing problem," *Mathematical and Computer Modelling*, vol. 48, no. 3-4, pp. 420–428, 2008.

[33] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, December 1995.

[34] F. Tao, D. Zhao, Y. Hu, and Z. Zhou, "Resource service composition and its optimal-selection based on particle swarm optimization in manufacturing grid system," *IEEE Transactions on Industrial Informatics*, vol. 4, no. 4, pp. 315–327, 2008.

[35] M. V. Raj, S. S. Sankar, and S. G. Ponnambalam, "Particle swarm optimization algorithm to maximize assembly efficiency," *International Journal of Advanced Manufacturing Technology*, vol. 59, no. 5–8, pp. 719–736, 2012.

[36] X. Li, L. Gao, and X. Wen, "Application of an efficient modified particle swarm optimization algorithm for process planning," *International Journal of Advanced Manufacturing Technology*, 2012.

[37] T. Zhang, T. Hu, J.-W. Chen, Z. Wan, and X. Guo, "Solving bilevel multiobjective programming problem by elite quantum behaved particle swarm optimization," *Abstract and Applied Analysis*, vol. 2012, Article ID 102482, 20 pages, 2012.

[38] V. Roberge, M. Tarbouchi, and G. Labonté, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 132–141, 2013.