Model and Algorithm for Co-scheduling of Stackers and Single RGV during retrieval Process in AS/RS

HU Pengpeng, CHEN Hua*, WANG Xinping and SHI Menghua

School of management, Xi’an University of Science and Technology, Xi’an, 710054, P.R. China

*Corresponding author’s e-mail: chenchh@foxmail.com

Abstract: This paper studies the co-scheduling problem of the stackers and single rail guided vehicle system in automated storage and retrieval systems. We formulate the problem into a mixed integer linear programming model with the aim of minimizing the total retrieval time. In order to solve the problem, an improved genetic algorithm is proposed, and the properties of the optimal solution are applied to improve the offspring chromosomes. A lower bound of the total retrieval time is developed. The numerical experiment indicates that the proposed algorithm can solve the problem effectively. Compared with the solutions obtained by using first come first served, the total retrieval time obtained by the proposed algorithm is reduced by an average of 8.36%. The average deviation from the lower bound is 2.00%.

1. Introduction

With the rapid development of science and technology, automatic storage and retrieval system (AS/RS) gradually replaces the traditional warehousing. A typical AS/RS consists of high-rack storage system, material handling system (MHS), input/output stations and computer control system. Rail Guided Vehicle (RGV) system with rectilinear rail is widely used as MHS in AS/RS. Figure 1 shows the layout of an AS/RS with RGV system. When materials are handled in or out of the AS/RS, materials are always transported both by stackers and RGV system. Due to the reason that more stackers are required than RGV, the RGV is utilized all the time, while stackers are only utilized 15% of the time [1]. Therefore, the performance of the AS/RS mainly depends on the performance of RGV system. Properly co-scheduling RGV system and stackers helps reducing the extra RGV travelling distance, which in turn would improve the performance of the RGV system.

In recent years, with the rising application of RGV system, some studies of RGV scheduling problem has been reported. Dotoli & Fanti proposed a modular and modeling framework of RGV and the material handling system of aisle stacker, and utilized colored Petri net to apply control policies to the RGV scheduling problem [2]. Liu et al. proposed two operating strategies of two-RGV systems: path partitioning mode and path integration mode, and compared the advantages and disadvantages of the two strategies through simulation experiments [3]. Wu et al. proposed a dynamic model of RGV system based on two-layer Petri net, and adopted the shortest-path scheduling strategy [4]. Yang et al. applied queuing theory to analyze the optimal number and capability of multi-RGV system with circular rail [5]. Hu et al. studied dynamic routing problem of an RGV system for two-sided loading/unloading automated freight handling system [6]. Xiang et al. applied queuing theory to formulate the scheduling problem of RGV system with circular rail into mathematical model, analyzed and evaluated the operation parameters of an actual case [7]. To sum up, the aforementioned studies mainly focused on the simulation of RGV dispatching rules, route selection rules, scheduling rules and
intelligent optimization algorithm. Though feasible solutions can be easily obtained by these scheduling rules, they may not be the optimal ones. Furthermore, the literatures which studied the RGV scheduling problem in AS/RS didn’t consider the co-scheduling of RGV and stacker.

Figure 1. An AS/RS layout with RGV system with rectilinear rail

This paper studies the co-scheduling problem of stackers and an RGV system with rectilinear tail in AS/RS, and formulates the problem into a mixed integer programming model. An improved genetic algorithm is proposed to solve the problem. Computational experiments are presented and discussed.

2. Problem formulation

2.1 Problem Description and Variables

The co-scheduling problem of stackers and single RGV system can be described as follows: An AS/RS has \( m \) aisles. Each aisle has two sets of racks, which are served by an aisle stacker. There are \( N \) output stations. Figure 1 shows the layout of the AS/RS. Each item (a unit of materials) is stored in a bay on the rack. The items to be retrieved from the same aisle is uniformly numbered in this paper. Let \( J_{aj} \) represents the \( j \)th item in aisle \( a \) (\( j \) does not represent the retrieval order of stacker \( m \)). Let \( n_a \) denotes the number of items to be retrieved in aisle \( a \). During the retrieval operation, item \( J_{aj} \) is firstly transported from storage bay to the input/output (I/O) conveyor in aisle \( a \) by the stacker, and then transported to one of the output stations by idle RGV. Let move \( aj \) represent the operation of RGV transporting item \( J_{aj} \). Let \( V \) denotes the average speed of RGV, and \( u \) denotes the average loading or unloading time of RGV. \( t_{ae} \) denotes the time required for the RGV to travel from I/O conveyor \( a \) to output station \( e \), or from output station \( e \) to I/O conveyor \( a \). \( r_{aj} \) represents the retrieval time required for the stacker. \( P_a \) denotes the location of I/O conveyor \( a \). \( q_e \) denotes the location of the output station \( e \). \( M \) is a large positive number. There are some decision variables used in this paper. \( R_{aj} \) represents the start time of RGV move \( aj \). \( C_{aj} \) represents the complete time of move \( aj \). \( P_{aj}^s \) represents the starting location of the move \( aj \). \( P_{aj}^c \) represents the completion location of move \( aj \). \( C_{\text{max}} \) represents the latest completion time of all items, which is also equal to the total retrieval time of all items. In addition,

\[
x_{a}^{ji} = \begin{cases} 1 & \text{if } J_{aj} \text{ is the } i\text{th item retrieved by stacker } a \\ 0 & \text{otherwise} \end{cases},
\]

\[
w_{a}^{je} = \begin{cases} 1 & \text{if } J_{aj} \text{ is transported to output station } e \\ 0 & \text{otherwise} \end{cases}, \text{ as well as}
\]

\[
y_{a}^{ss} = \begin{cases} 1 & \text{if } R_{aj} \leq R_{bk} \\ 0 & \text{otherwise} \end{cases}.
\]

Assuming that the capacity of the conveyor is one unit, the stacker only starts to pick up another item after the item on the I/O conveyor is taken away by RGV. If one specific stacker didn’t finish one
retrieval operation, the RGV needs to wait or travel to other I/O conveyor to transport other items. Which in turn would cause extra RGV traveling distance. The aim of this study is to minimize the total retrieval time of all items by co-scheduling of RGV and stackers.

2.2 Mathematical Model

Minimize \( C_{\text{max}} \)

Subject to:
\[
R_{bk} - R_{aj} \leq M \gamma_{aj,bk}^s \\
\gamma_{aj,bk}^s + \gamma_{bk,aj}^s = 1 \\
\sum_{i=1}^{a} x_{aji} = 1 \\
\sum_{j=1}^{n} x_{aji} = 1 \\
R_{aj} \geq r_{aj} + M(x_{aji} - 1) \\
R_{aj} \geq r_{aj} + R_{ah} + M(x_{aji} + x_{ah(i-1)} - 2) \\
\sum_{e=1}^{N} w_{aje} = 1 \\
C_{aj} = r_{aj} + \sum_{e=1}^{N} t_{ae}w_{aje} + 2u \\
R_{bk} \geq R_{aj} + \sum_{e=1}^{N} t_{ae}w_{aje} + 2u + \sum_{e=1}^{N} t_{be}w_{aje} + M(\gamma_{aj,bk}^s - 1) \\
C_{aj} \leq C_{\text{max}} \\
p_{aj}^c = \sum_{e=1}^{N} q_{e}w_{aje} \\
p_{aj}^c = \begin{cases} 
1 & a = 1, ..., m; j = 1, ..., n_a 
\end{cases}
\]

Equation (1) gives the objective function. Constraints (2) and (3) guarantee that the variables are correctly defined. Constraints (4) and (5) ensure that each stacker only retrieves one item at a time, and each item can only be retrieved once. Constraint (6) ensures that when item \( aj \) is the first item to be retrieved in aisle \( a \), the corresponding RGV move start time should be at least greater than or equal to \( r_{aj} \). When item \( j_{aj} \) and item \( j_{ah} \) are two items consecutively retrieved in aisle \( a \), constraint (7) ensures that the stacker in aisle \( a \) has enough time to pick up item \( j_{aj} \) after finishing retrieval item \( j_{ah} \). Constraint (8) guarantees that one item can only be transported to an output station. Given the start time of the RGV move, constraint (9) gives the calculation equation of the completion time of RGV moves. For any two consecutive RGV moves, constraint (10) ensures that RGV has enough time to travel to the next I/O conveyor to load another item after completes the current move. Constraint (11) forces \( C_{\text{max}} \) to be the largest \( C_{aj} \). Constraints (12) and (13) guarantee that the starting location of one move is the corresponding I/O conveyor, and the complete location is the location of the output station which the items are unloaded onto.

2.3 Property of the optimal solution

Property of the optimal solution: Assuming that move \( aj \) and the move \( bk \) are any two consecutive moves in the optimal schedule, if move \( aj \) is started earlier than move \( bk \), then item \( j_{aj} \) is always transported to the output station which minimizes \( t_{ae} + t_{be} \). That is, the completion location of move \( aj \) must satisfy:
\[
p_{aj}^c = \sum_{e=1}^{N} q_{e}w_{aje} \\
p_{aj}^c = \begin{cases} 
1 & a = 1, ..., m; j = 1, ..., n_a 
\end{cases}
\]

Proof: (Reduction to absurdity) Suppose the proposition does not hold. That is, there are at least two consecutive RGV moves in the optimal schedule, move \( aj \) and move \( b'k \), which satisfy \( p_{bj}^c \neq q_{e} \), \( \bar{e} = \arg\min_{e=1,...,N} (t_{ae} + t_{be}) \). Assume move \( aj \) is transported to the output station \( e' \), and the corresponding optimal objective value is \( \bar{C} \). From the definition of \( \bar{e} \), we can get that \( t_{a're} + t_{b're} \geq t_{a'e} + t_{b'e} \). If other moves in the optimal schedule remain unchanged, then a new feasible solution can be obtained by switching the completion location of move \( aj \) from output station \( e' \) to output station \( \bar{e} \), and starts all moves, which are started later than these two moves, a little \( (t_{a're} + t_{b're}) - (t_{a'e} + t_{b'e}) \) earlier. Assume the new objective function value is \( \bar{C} \), then there must have \( \bar{C} \leq C \),
which contradicts the value of $C$ being the optimal objective function of the problem. Therefore, the original proposition is valid.

3. Improved Genetic Algorithm and Lower Bound

We propose a genetic algorithm to solve the problem. The property proved in 2.3 is applied to improve each offspring chromosomes. A lower bound is proposed to evaluate the performance of the improved genetic algorithm (IGA).

3.1 Improved Genetic Algorithm

1) Chromosomes representation

This paper proposes a sequence dependent chromosome representation schema to generate the initial sequence of the RGV move. Each chromosome represents a fixed retrieval sequence, and the length of the chromosome equals the total number of items. By allocating the output station which is nearest to the corresponding I/O conveyor to be the output station where the item is transported to, each chromosome represents a feasible solution to the original problem.

2) Calculation function of adaptive value

The objective function value of any chromosome is calculated and denoted by $C$, and the maximum value among all chromosomes is denoted by MC, the adaptive value of the individual is $MC - C$.

3) Select operation, Crossover and mutation

The best individual in the current population is passed on directly to next generation using an elite retention strategy. Two-point switch method is used for crossover. Inversion mutation operator is applied.

4) Algorithm implementation steps

Step 1: Initialization. Set the parameters of IGA. Generate an initial population. This paper adopts the method of randomly generating the initial population to ensure the diversity of the solution, and constructs each chromosome to be feasible solution by allocating output station to each gene. Set the initial population as the current population.

Step 2: Calculate the fitness value and use the elite retention strategy.

Step 3: Select operation. Keep a certain number of elite individuals directly to the next generation. Select a pair of parent chromosomes randomly.

Step 4: Crossover and mutation. Generate new populations through crossover and mutation operations, and check if the number of new populations reaches the preset population size. If not, go to Step 3. Otherwise, go to Step 5.

Step 5: Improve operation. To improve each chromosome in the offspring population, and set the new population as the current population, repeat Step 2–Step 5 until satisfy the stop criterion.

3.2 The lower bound of the problem

To evaluate the performance of the algorithm, a lower bound is developed and given in the following formula:

$$LB = 2 \sum_{a=1}^{m} (n_a \times (u + \tau_a)) - \bar{\tau} \quad (15)$$

In the formula: $\tau_a = \min \{ \tau_{ae}, e = 1, ..., N \}$, $\bar{\tau} = \max \{ \tau_a, a = 1, ..., m \}$, $\tau_{aj}, a = 1, ..., m, j = 1, ..., n_a$

To illustrate that $LB$ is the lower bound of the optimal solution of the problem, this paper first considers the RGV moves during the retrieval process. In order to complete a move, the RGV always needs to start from a certain output station, drive to the I/O conveyor where the item is, loads the item, transports it to a certain output station and unloads it. By assuming that the RGV always starts from the nearest output station to the I/O conveyor, and RGV always transports the item to this output station, the operating time required to retrieve all items is at least $2 \sum_{a=1}^{m} (n_a \times (u + \tau_a))$, of which $\tau_a$ is the time that RGV travels from the I/O conveyor $a$ to the nearest output station. Considering that this paper assumes that RGV can be at any position of the tail at time 0, it can be considered that
RGV is waiting at the position of the conveyor corresponding to the first retrieval item at time 0. We need to subtract one \( \tau_a (a = 1, ..., m) \) from the total RGV traveling time, and by subtracting the largest \( \tau_a \), denoted by \( \tau \), then RGV operation time is at least \( 2 \sum_{a=1}^{m} (n_a \times (u + \tau_a)) - \tau \). Second, consider the stacker action during the output process. Since each RGV move can only be started when the stacker retrieves the corresponding item and unloads it on the I/O conveyor, the start time of the first of RGV move must not be less than the minimum time \( \bar{r} \). In addition, if the stacker does not complete retrieving operation of the item, the RGV needs to wait. By ignoring this possible waiting time, the total retrieval time of all items necessarily does not exceed \( 2 \sum_{a=1}^{m} (n_a \times (u + \tau_a)) - \tau + \bar{r} \).

4. Computational experiments

To evaluate the performance of the proposed IGA algorithm, 10 sets of randomly generated test problems are calculated. The parameters of the AS/RS and RGV system are collected from an AS/RS which preserves relief supplies. The \( x \) coordinates of the location of I/O conveyors are 0,3,6,9,15,18,21 (meters) respectively; the \( x \) coordinates of the locations of the output station are 3,6,15,15 (meters) respectively. The average speed of RGV, speed of stackers and the load/unload time are randomly generated respectively obeying \( U[1, 3] \), \( U[15, 22] \), and \( U[6, 10] \). Table 1 shows the parameter settings of the scale of 10 problem sets. The results of IGA are compared with the results of CPLEX12.5. When the scale of the problem is big, CPLEX couldn’t solve the problem in reasonable time, thus we set a calculation time for some of the instances. This paper also applies the general FCFS (First Come First Served) scheduling strategy to calculate each example, and compares the results with the results of IGA. The proposed IGA algorithm is coded in C++, and runs on a computer with 3.10 GHz CPU and 4GB RAM. Parameter settings of IGA algorithm is shown in Table 2. For each set of problem, the IGA and FCFS both runs 10 times to get solutions. The LB of each problem is also calculated. The results are given in Table 3.

### Table 1 Parameter setting of problem scale

| Numerical example | \( m \times N \) | \( n_a \) | Total amount of material out of warehouse |
|-------------------|-----------------|--------|----------------------------------------|
| 1                 | 4*2             | 1,2,2,1 | 6                                      |
| 2                 | 4*2             | 2,2,2,2 | 8                                      |
| 3                 | 4*2             | 3,2,1,4 | 10                                     |
| 4                 | 4*2             | 3,3,3,3 | 12                                     |
| 5                 | 8*2             | 2,1,3,1,2,1,2,2 | 14                                    |
| 6                 | 8*3             | 4,4,4,4,4,4,4,4 | 32                                    |
| 7                 | 8*3             | 7,7,7,7,7,7,7,7 | 56                                    |
| 8                 | 8*4             | 9,9,9,9,9,9,9,9 | 72                                    |
| 9                 | 8*4             | 10,10,10,10,10,10,10,10 | 80                                    |
| 10                | 8*4             | 12,12,12,12,12,12,12,12 | 96                                    |

### Table 2 Sets parameters of the improved genetic algorithm

| Parameters                  | The values |
|-----------------------------|------------|
| Population size             | 20, 20, 20, 30, 30, 60, 80, 120, 120, 120 |
| Number of iterations of genetic algorithm | 10, 10, 10, 20, 20, 40, 60, 100, 100, 100 |
| Crossover probability       | 0.7        |
| Mutation probability        | 0.1        |
| Elite retention             | 2, 2, 2, 3, 3, 6, 8, 8, 8, 8 |

Note: \( U [a, b] \) means that the parameters are uniformly distributed in the range \([a, b] \).

1) Comparison of IGA calculation results with CPLEX and lower bound

In Table 3, dev and gap respectively represents average percentage deviation of C_IGA from C_CPLEX, and average percentage deviation of C_IGA from LB. From Table 3, we can see that the result of IGA can obtain optimal solutions or can obtain better solutions when CPLEX couldn’t solve the problem in preset calculation time. By comparing with the proposed LB, we can see that the total average deviation is 2.00%.
2) Comparison between IGA solution results and FCFS solution results

From Table 3, we can see the total average deviation between the results of IGA and FCFS is -8.36%, it is obvious that IGA is far better than the scheduling results obtained by FCFS.

5. Conclusions

This paper studies the co-scheduling problem of single RGV and stackers during retrieval operation in AS/RS. The problem is formulated into a mixed integer programming model with the goal of minimizing the total retrieval time. The properties of the optimal solution are proved and the lower bound of the problem is developed. An improved genetic algorithm is proposed to solve the problem. The numerical experiment shows that the proposed algorithm can solve the problem efficiently. Compared with FCFS, the solutions obtained by IGA are much better.

| Numerical example | IGA Best solution (s) | Average value (s) | CPU (s) | FCFS Best solution (s) | Average solution (s) | MILP Optimal or best solution(s) | LB (%) | gap (%) | dev (%) | FCFS deviation (%) |
|-------------------|-----------------------|-------------------|--------|------------------------|----------------------|-------------------------------|--------|--------|---------|-------------------|
| 1                 | 111.5                 | 111.5             | 0.047  | 111.5                  | 111.5                | 0.08                          | 107.5  | 0      | 3.72    | 0                 |
| 2                 | 145.5                 | 145.5             | 0.094  | 149.5                  | 152.9                | 0.73                          | 143.5  | 0      | 1.39    | -4.84             |
| 3                 | 192                   | 192               | 0.066  | 196.5                  | 202.53               | 191.6                         | 59.87  | 0.21   | 2.4     | -5.20             |
| 4                 | 216                   | 216.45            | 0.164  | 222                    | 227.54               | 216                           | 6043.4 | 0.21   | 0.79    | -4.87             |
| 5                 | 278.8                 | 279               | 0.128  | 290.8                  | 309.518              | 278.8                         | -       | 0.07   | 0.79    | -9.86             |
| 6                 | 591.2                 | 595.48            | 4.884  | 625.2                  | 688                  | 589.2                         | 583.2  | 1.07   | 2.11    | -13.45            |
| 7                 | 1116.5                | 1120.8            | 19.453 | 1166                   | 1209.25              | 1138                          | 1108.5 | -1.51  | 1.11    | -7.31             |
| 8                 | 1390.2                | 1399.89           | 115.109| 1575.2                 | 1718.95              | 1539                          | 1347.7 | -9.04  | 3.87    | -18.56            |
| 9                 | 1524.1                | 1528.39           | 132.343| 1666.55                | 1710.35              | 1591.7                        | 1494.75| -3.98  | 2.25    | -10.64            |
| 10                | 2055                  | 2062.2            | 156.093| 2207.3                 | 2262.16              | 2060.3                        | 2032   | 0.09   | 1.49    | -8.84             |
| Average           |                       |                   |        |                        |                      |                               | -1.29  | 2.00   | -8.36   |                   |

Note: * means the calculation time is 24h

Acknowledgement

This research is supported by the Education Department of Shaanxi Provincial Government foundation (Grant No. 18JK0494), a startup fund and a Key Project of Philosphy and Social Science Research fund provided by Xi’an University of Science and Technology (Grant No. 2017QDJ055, 2017SZ02). The authors also want to thank Prof. Zhou Zhili for his kind guidance.

Reference

[1] Lee, S. G., de Souza, R., and Ong, E. K. (1996) Simulation modelling of a narrow aisle automated storage and retrieval system (AS/RS) serviced by rail-guided vehicles. Computers in Industry. 30(3), 241–253.
[2] Dotoli M, Fanti M P. (2005) A colored Petri net model for automated storage and retrieval systems serviced by rail-guided vehicles: a control perspective. International Journal of Computer Integrated Manufacturing. 18(2-3): 122-136.
[3] Liu Y, Li S, Li J, et al. (2010) Operation Policy Research of Double Rail-Guided Vehicle Based on Simulation// E-Product E-Service and E-Entertainment¬(ICEEE). 2010 International Conference on IEEE,2010:1-4.
[4] Wu Changqing, Luo Jian, Chen Huoguo, et al. (2009) Research on loop deadlock in RGV system based on Petri net. Computer science. 36(4):250-253.
[5] Yang Shaohua, Zhang Jiayi, and Zhao Li. (2011) Analysis of the number and capacity of multiple rail vehicles based on queuing theory. Manufacturing automation. 33(16): 102-104.
[6] Hu, W., Mao, J., and Wei, K. (2017) Energy-efficient rail guided vehicle routing for two-sided loading/unloading automated freight handling system. European Journal of Operational Research. 258(3): 943–957.

[7] Xiang Wang, Wu Shuang, Zhang Keyi, et al. (2018) Circular RGV system's operation parameter analysis based on queuing theory. Manufacturing Automation. 40(06), 151-153.