An algorithm based on quasi critical path strategy for exchanging adjacent parallel processes of the same device

Zhang xiaohuan
Huizhou University
Wang zhen (wangzhen@hzu.edu.cn)
Huizhou University
Zhang dan
Huizhou University
Xu tao
Huizhou University

Research Article

Keywords: quasi critical path, vertical scheduling, horizontal scheduling, adjacent parallel processes, basic scheduling scheme, optimal product scheduling scheme

Posted Date: February 28th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1346289/v1

Abstract

Aiming at the defect that quasi critical path algorithm can't take into account both vertical scheduling and horizontal scheduling when scheduling products, this paper proposes an algorithm based on quasi critical path strategy for exchanging adjacent parallel processes of the same device. The algorithm takes the scheduling scheme formed by the quasi critical path strategy as the basic scheduling scheme, proposes and applies the adjacent parallel processes interchange strategy and the adjacent parallel processes adjustment strategy, generates several new product scheduling schemes, and adds them to the product scheduling scheme set. Finally, according to the optimal product scheduling scheme selection strategy, the product scheduling scheme with the minimum total processing time is selected as the final scheduling scheme. The proposed algorithm ensures the compactness of the serial processes, improves the parallel processing of the parallel processes, and optimizes the scheduling results.

1. Introduction

Scheduling, as a key factor affecting the production efficiency of enterprises, has always been a hot issue studied by scholars. Efficient scheduling optimization algorithm can maximize production efficiency and help enterprises to achieve higher benefits. At present, there are two main research directions in this field, namely, single processing (assembly) scheduling and integrated scheduling. Typical representatives of the former scheduling are job-shop and flow-shop scheduling. This kind of production mode of processing first and assembling later is quite common in mass production. The reason is that: due to the large quantity of products ordered, the centralized production of disassembled parts will produce a large amount of inventory, and then the inventory will be assembled. In this way, synchronous processing and assembly can shorten the production cycle. At present, there are many research achievements in this direction, and more advanced methods include genetic algorithm [1], tabu search [2], neural network method [3][4], heuristic algorithm [5] and various hybrid algorithms [6–8]. However, at present, enterprises are facing more and more single and small batch orders. The production of these orders often does not produce inventory, or produces a small amount of inventory. If the job shop and flow shop scheduling methods are still used, it will not shorten the production cycle, but will prolong the production cycle because of the inevitable connection between production and assembly. Therefore, Xie et al. [9] proposed an integrated scheduling method considering production and processing activities and assembly activities at the same time. Among them, the quasi critical path method is an effective method to solve the integrated scheduling problem. The algorithm points out the influence of the scheduling of long path nodes in the process tree on the final scheduling results, and proposes to determine the scheduling order of nodes by calculating the path length of each process node in the process tree, so as to improve the scheduling priority of nodes on the long path and optimize the scheduling results. Many scholars then improved the algorithm, including layer first, dynamic critical path and so on [10–13], but these methods have high complexity. Based on the quasi critical path method, this paper proposes the adjacent parallel process exchange strategy of the same device to optimize the scheduling results without increasing the complexity of the algorithm [14].

The rest of this paper is organized as follows: Section 2 contains problem description and analysis. Section 3 discusses the strategy design. Section 4 shows the example analysis. Section 5 shows experimental comparison and analysis. Section 6 concludes the paper.

2. Problem Description And Analysis

The product processing technology of the integrated scheduling problem is shown as a tree structure, there is a product process tree, as shown in Fig. 1. Where, the node in the tree represents the working procedure of the product, Among them, the node with a penetration of 1 is the processing process node, and the node with a penetration greater than 1 is the assembly process node. In this paper, these two types of operations are treated together, collectively referred to as processes. The text in the node represents the process number, processing device number and processing time respectively. For example, A1/M1/15 represents that process A1 will be processed on device M1, and the processing time is 15 time units. The directed edge represents the partial order relationship of the processing order of the operation. The root node represents the last operation in the product. The processing of the root node indicates that the product has been processed. In addition, the integrated scheduling problem must meet the following constraints.

(1)Each process must strictly abide by the agreed partial order relationship in the processing tree;

(2)Each device can only process one process at any time, and the processing process cannot be interrupted;

(3)There is no device with the same function in the device set;

(4)If and only if all the foreprocesses of a process are in the state of finished processing (or no foreprocesses), the process can be processed;

(5)The same device does not exist in the workshop;

(6)Waiting between processing activities of different processes is allowed, and the device is allowed to idle before the process arrives.

The difference between the processing end time of the latest end process and the processing start time of the earliest start process is the total processing time of the product. The scheduling objective of general integrated scheduling problem is to minimize the total processing time.

The general description of integrated scheduling problem is shown in formula (1) - formula (4).

$T = \min \{\max\{E_i\}\}$ (1)

s.t. $\min(t_{ij})$ (2)

$t_{ij+1} \leq t_{ij} + g_{ij}$, $i = 1,2,..., M$; $j = 1,2,...,N$ (3)

$t_{xy} \geq \max(t_{ij} + g_{ij})$ (4)

Where, in formula (1), T is the scheduling optimization objective: the product is completed as soon as possible, and E_i is the completion time of the last process on device i (i = 1, 2,..., m);In Eq. (2), t_{ij} is the start processing time of the jth process of equipment I, and $min(t_{ij})$ means that each process starts as soon as possible under the constraints of equations (3) and (4);Equations (3) and (4) respectively represent the constraints of process processing, $t_{ij+1} <= t_{ij}+g_{ij}$ means that the start time of the subsequent process on the same equipment must be after the end of the previous process, g_{ij} is the continuous processing time of the j-th process of device i, $t_{xy} >= \max(t_{ij}+g_{ij})$ indicates that the start time of the subsequent process in the process diagram must be after the end of the previous process. Process t_{xy} is the subsequent process of process t_{ij}.

3. Strategy Design

The algorithm design idea proposed in this paper is to optimize the scheduling results without increasing the time complexity of the comprehensive scheduling algorithm. Through analysis, it is found that the quasi critical path scheduling method pays too much attention to the vertical scheduling of processes in the process tree and ignores the parallel scheduling of horizontal processes [7], resulting in unsatisfactory scheduling results. Therefore, based on the scheduling results of quasi critical path algorithm, this paper proposes to adjust the scheduling sequence of adjacent processes and parallel processes on each process, so as to adjust the scheduling of subsequent processes, so as to find a better solution. The algorithm framework is as follows:

1. Set the number of device required during product processing as N and the total number of processing processes on each device as nd;

2. Calculate the processing path length of each process node in the product processing process tree;
3. Use the quasi critical path strategy to generate the product scheduling scheme and add it to the set of alternative product scheduling schemes;
4. For i = 1 to N;
5. For j = 1 to Nd-1;
6. Based on the product scheduling scheme generated by the proposed critical path strategy, use the adjacent process exchange strategy of the same device to select the exchange process and exchange the scheduling sequence;
7. Use the same device process exchange adjustment strategy to adjust the processes affected by the exchange process in step 6 to generate a new product scheduling scheme;
8. Add the scheduling scheme obtained in step 7 to the set of alternative product scheduling schemes;
9. Next i;
10. Next j;
11.In the set of alternative product scheduling schemes, the optimal product scheduling scheme selection strategy is used to select the optimal scheduling scheme.
12.Generate product scheduling Gantt chart.

Next, the adjacent process exchange strategy of the same device, the adjacent process exchange adjustment strategy of the same device and the optimal product scheduling scheme selection strategy are introduced in detail.

3.1Analysis and design of the adjacent process exchange strategy of the same device

Suppose there are 20 processes in the process tree, as shown in Fig. 1,namely *A*, *B*, *C*, *D*, *E*, *F*, *G*, *H*, *I*, *J*, *K*, *L*, *M*, *N*, *O*, *P*, *Q*, *R*, *S* and *T*. These processes are scheduled according to the quasi-critical path algorithm, and the result is, the sequence of processes on *M*1 is: *A*, *B*, *C*, *D*, *E*, *F* and *G*; The process sequence on *M*2 is: *H*, *I*, *J*, *K* and *L*; The process sequence of *M*3 is: *M*, *N*, *O*, *P*, *Q*, *R*, *S* and *T*. As shown in Fig. 1, the processes on each processing device are arranged in a queue in order, and the final order is *A*, *B*, *C*, *D*, *E*, *F*, *G*, *H*, *I*, *J*, *K*, *L*, *M*, *N*, *P*, *Q*, *R*, *S* and *T*. Start from the first step *A*, look for adjacent procedures on the same device one by one, and judge whether they can be exchanged. Obviously, process *G* and process *H*, process *L* and process *M*, these two adjacent processes cannot be exchanged because they do not belong to the same processing device. The remaining adjacent processes need to be further determined whether the processing sequence can be interchangeable. The method is: to judge whether there is A serial relationship between the current process and the tight post-process in the process queue. If there is, it cannot be exchanged; otherwise, the processing order of the tight post-process in the process A and the same device will be exchanged.

As shown in Fig. 3, processes *A* and *B* on the same processing device *M*3 processes, in its processing device process *C* has already on the *M*3 is processed, in addition, the process is *A* process before processing technology in the tree close to process *A* ', is processed on device *M*2, tight in the working procedure in process *B* tree before working procedure for process *B* ', the device is processed on the *M*1, the two can exchange, exchange process is divided into three kinds of circumstances:

(1) When the end time of process *A* 'and *B*' are less than the end time of process *C*, the beginning time of process *A* is the end time of process *C*, and the beginning time of process *B* is the end time of process *A*. When process *A* and process *B* are exchanged, only the start time of process *A* is set as the start time of process *B*, and the end time of process *B* is set as the start time of process *A*.

(2) When the end time of process *B* 'is greater than that of process *C*, the beginning time of process *A* is the end time of process *C*, and the beginning time of process *B* is the end time of process *A*. When process *A* and process *B* are exchanged, only the start time of process *B*' is set as the start time of process *B*, and the end time of process *B* is set as the start time of process *A*.
(3) When the end time of process *A* 'is greater than the end time of process *C*, the beginning time of process *A*' is the end time of process *A* ', and the beginning time of process *B* is the end time of process *A*. When process *A* and Process *B* are exchanged, only the start time of process *B* 'is set as the start time of process *B*, and the end time of process *B* is set as the start time of process *A*.
To sum up, in order to realize the exchange between process *A* and process *B*, the starting time of process *B* can be determined first, and then the starting time of process *A* can be determined.

The specific steps of adjacent process interchange strategy of the same device are as follows:

Step 1: Determine whether process *A* and its tight rear process *B* in queue *Q* belong to *A* processing device; if yes, go to 2; otherwise, go to 5.

Step 2: Determine whether there is *A* serial relationship between process *A* and process *B*; if yes, go to 5; otherwise, go to 3.

Step 3: Adjust process *B* to be before process *A*, and the processing start time is the maximum value of the end time of the pre-tightening process of original process *A* and the end time of the pre-tightening process of process *B* in the process tree.

Step 4: Adjust the process *A* to be before the process *B*, and take the maximum value of the end time of the process *B* and the end time of the process *A* in the process tree.

Step 5: end.

3.2 Analysis and design of the adjacent process exchange adjustment strategy of the same device

For a process, its current optimal processing time should be the maximum value of the end time of the process before tightening the device and the end time of the process before tightening the process in its process tree.

The algorithm steps are as follows:

Step 1: Judge whether there is a pre-tightening process in the process tree. If there is no turning to 2, if there is turning to 4;

Step 2: Set the finishing time of the pre-tight work in the process tree of this process to 0 and turn it to 4;

Step 3: Judge whether there is a pre-tightening process of the device in this process. If it does not turn to 4, if it does turn to 5;

Step 4: Set the finishing time of the process before tightening the device in this process to 0 and turn it to 5;

Step 5: Take the maximum value of the end time of the process before tightening the device of the process and the end time of the process before tightening in the process tree of the process as the processing time of the current process;

Step 6:end.

3.3 Optimal product scheduling scheme selection strategy analysis and design

The algorithm steps are as follows:

Step 1: Judge whether the current scheduling set is empty. If it is, go to 4; if not, go to 2.

Step 2: Select the scheme with the minimum total product processing time in the set as the selection scheme;

Step 3: Return the selected object scheme;

Step 4:end.

3.4 Implementation steps of the proposed algorithm

The algorithm steps are as follows:

Step 1: The quasi-critical path strategy is adopted to determine the scheduling sequence of each process in the product processing tree.

Step 2: Use the first adaptation strategy to determine the scheduling time of each process in the product processing process tree, and form the original scheduling plan *D*.

Step 3: Set up a total of *M* processing device.

Step 4: *i* = 1.

Step 5: Judge that *I*< = *M*, if true, go to 6, if not, go to 8.

Step 6: Put all the processes on the ith device in the original scheduling scheme into queue *Q* according to the processing order.

Step 7: *i*++, go to 5.

Step 8: Determine whether there is only one process in queue *Q*, instead of going to 9, go to 17.

Step 9: Queue *A* from queue *Q*.

Step 10: Start the adjacent processes interchange strategy process *A*, If the interchange occurs, go to 11; otherwise go to 12.

Step 11: Start the adjacent process interchange strategy and go to 12.

Step 12: Add the resulting new product scheduling solution to the product scheduling solution set, and go to 8.

Step 13: Calculate the total processing time of each product scheduling scheme in the product scheduling scheme set respectively.

Step 14: Select the final scheduling scheme of the product according to the optimal scheduling scheme selection policy.

Step 15: Output the gantt chart of scheduling results.

3.5 Complexity analysis of the proposed algorithm

Let all the cross points in the product processing tree whose input degree is greater than 1 be K, that is, the number of subtrees be K, the total number of work order be N, and the number of device be M. Based on literature [7], that the critical path algorithm complexity to $O\{\text{Max}[(n(n\text{-}m)/(2m)),(nn)/(16m)]\}$.

In this paper, the adjacent process interchange strategy of the same device is designed. The average number of processes involved in this algorithm is $N/2$, and the operational complexity of the algorithm for the interchange of processes is $O(1)$, and the time complexity of this strategy algorithm is $O(n/2)$. The adjacent process exchange adjustment strategy of the same device is designed, which involves at most N-1 processes. The operation algorithm complexity of the adjustment of process processing time is $O(1)$, and the time complexity of this strategy algorithm is $O(n\text{-}1)$.

Due to the quasi-critical path strategy, the adjacent process interchange strategy and the adjustment strategy for the interchange between adjacent processes are serial operations, so the time complexity of the algorithm in this paper is the same as that of the algorithm in literature [7], which is $O\{\text{Max}[(n(n\text{-}m)/(2m)), (nn)/(16m)]\}$.

4. The Example Analysis

In order to facilitate readers to understand the algorithm, the following is an example analysis. Product a is set as shown in Fig. 1. The total processing time obtained by scheduling product a according to the algorithm proposed in [7] is 265, and the Gantt chart for scheduling final product a is shown in Fig. 4. At the same time, scheduling the product processing process tree shown in Fig. 1 according to the algorithm proposed in this paper, the total processing time of product a is 255, and the Gantt chart for scheduling final product a is shown in Fig. 29.

According to the proposed algorithm, the scheduling results shown in Fig. 4 were respectively taken as the initial scheduling schemes, and the adjacent processes on the same device were swapped to try to find a better scheduling scheme.

Specific scheduling steps and results are shown in Fig. 5 to Fig. 28:

By comparing the above scheme with the original scheduling scheme, it is found that the total processing time of the product scheduling scheme obtained after the adjustment of process A21 and Process A14 on device M3 is 255. This scheme is the scheduling scheme with the minimum processing time, and is selected as the final product scheduling scheme, as shown in Fig. 29.

5. Experimental Comparison And Analysis

To verify the effectiveness of the proposed algorithm,four sets of data were randomly generated, with 50 products in each set. The parameters of products were randomly generated.The software used on the experimental platform is Windows 10, 64 bit, GCC5.5, and the hardware of the experimental platform is an Intel Core I7-860 processor with 32 GB of memory. Five groups of experiments were designed as follows: The randomly generated data in Experiment 1 were that the number of processes was less than 20 and the total processing time of the workpiece was less than 50, the experimental results are shown in Fig. 30;The randomly generated data in Experiment 2 were that the number of processes was less than 30 and the total processing time of the workpiece was less than 100, the experimental results are shown in Fig. 31;The randomly generated data in Experiment 3 were that the number of processes was less than 50 and the total processing time of the workpiece was less than 200, the experimental results are shown in Fig. 32;The randomly generated data in Experiment 4 were that the number of processes was less than 60 and the total processing time of the workpiece was less than 300, the experimental results are shown in Fig. 33;Experiment 5 is the comparison of the average values of the above four experimental results, the experimental results are shown in Fig. 34.

Among the above five groups of experimental data, the black line represents the scheduling results obtained by the proposed algorithm, and the red line represents the scheduling results obtained by the quasi critical path method. It can be clearly seen from the figure that in the above experiments, the black line in the figure of each item is lower than or equal to the red line, indicating that the value of the scheduling result of the proposed algorithm is lower than that of the scheduling result of the quasi critical path method. Therefore, the scheduling results of the two algorithms are better than the quasi critical path scheduling method, which can prove the effectiveness of the proposed algorithm.

6. Conclusion

The key contributions of this work are:

(1) An algorithm based on quasi critical path strategy for exchanging adjacent parallel processes of the same device is proposed.

(2)The performance of the proposed algorithm is analyzed and compared with current algorithm.

(3)The proposed algorithm lays a theoretical foundation for the subsequent research of the algorithm.

It may be the next step to apply the idea of adjacent parallel processes interchange strategy to the field of integrated scheduling to solve other problems, such as: integrated scheduling problems with multi-device processes, batch integrated scheduling problems, distributed integrated scheduling problems, etc.

Declarations

Compliance with Ethical standards:

Humans and animals are not involved in this research work.

We used our own data.

The authors declare that they have no conflicts of interest

Acknowledgements: This work was supported by the Project of Educational Commission of Guang dong (No.2019KTSCX177), The Professorial and Doctoral Scientific Research Foundation of Huizhou University(No.2019JB014 No.2018JB007), Science and Technology Planning Project of Guangdong (No. 2020A1414010235), Science and Technology Planning Project of Huizhou (No.2020SC0306023).

Statements and Declarations

Funding

This work was supported by the Project of Educational Commission of Guang dong (No.2019KTSCX177), The Professorial and Doctoral Scientific Research Foundation of Huizhou University(No.2019JB014 No.2018JB007), Science and Technology Planning Project of Guangdong (No. 2020A1414010235), Science and Technology Planning Project of Huizhou (No.2020SC0306023).

Competing Interests

Conflicts of interest The authors declare that they have no conflicts of interest

References

1. Chen R, Yang B, Li S (2020) “A Self-Learning Genetic Algorithm based on Reinforcement Learning for Flexible Job-shop Scheduling Problem,”Computers & Industrial Engineering, vol. :106778
2. Mathlouthi I, Gendreau M, Potvin J “A Metaheuristic based on Tabu Search for Solving a Technician Routing and Scheduling Problem,”Computers & Operations Research, vol. 2020:105079
3. Shukla R, Kansakar P, Munir A (2017) “A Neural Network-Based Appliance Scheduling Methodology for Smart Homes and Buildings with Multiple Power Sources,” IEEE International Symposium on Nanoelectronic & Information Systems, vol.IEEE,
4. Al-akashi F (Dec. 2021) "Improving Learning Performance in Neural Networks". Int J Hybrid Innov Technol 1(2):27–42. doi:10.21742/IJHIT.2021.1.2.02

5. Naik A, Satapathy S, Abraham A (2020) Modified Social Group Optimization-a meta-heuristic algorithm to solve short-term hydrothermal scheduling. Appl Soft Comput vol 95:106524
6. Silva Amila L, Ling, Karunasekera Shanika (2021). "Research on the Application of an Improved LEACH Algorithm in Smart Home". International Journal of Smartcare Home, vol.1, no.1, pp.1–8
7. Laxmi B, Chandra P, Sagar S (2017) Evolutionary Hybrid Particle Swarm Optimization Algorithm for Solving NP-Hard No-Wait Flow Shop Scheduling Problems. Algorithms 10:121
8. Kang Q, Feng S, Zhou M (2017) “Optimal Load Scheduling of. ” IEEE Transactions on Intelligent Transportation Systems 18:2557–2568Plug-In Hybrid Electric Vehicles via Weight-Aggregation Multi-Objective Evolutionary Algorithms
9. Xie Z, Liu S, Qiao P (2003) Dynamic Job-Shop Scheduling Algorithm Based on ACPM and BFSM. J Comput Res Dev 40:977–983
10. Xie Z, Yang J, Yang G (2008) Dynamic device job-shop scheduling algorithm with dynamic device set of operation having priority. China J Comput 31:502–508
11. Xie Z, Xin Y, Yang J (2011) Integrated scheduling algorithm based on event driven by devices’idle. JMech Eng 47:139–147
12. Xie Z, Qi Y, Yang J (2014) “Integrated Scheduling Algorithm with Multiple-Devices- Operation,” Journal of Mechanical Engineering, vol. 49, pp. 89 – 9,
13. Barthelemy Andrew M (Sep. 2021) "Highly Intelligent Recommendation Algorithm based on Matrix Filling". Int J Hybrid Inform Technol 1(1):83–96. doi:10.21742/IJHIT.2021.1.1.07
14. Park M, Sohn J-H, ISSN (2018) "Characteristics of Quasi Microwave-Optical Single-Sideband Signal Generation Using a Nonlinear Semiconductor Optical Amplifier", International Journal of Advanced Science and Technology, NADIA, : 2005–4238 (Print); 2207–6360 (Online), vol. 111, February pp. 1–10, http://dx.doi.org/10.14257/ijast.2018.111.01

Figures

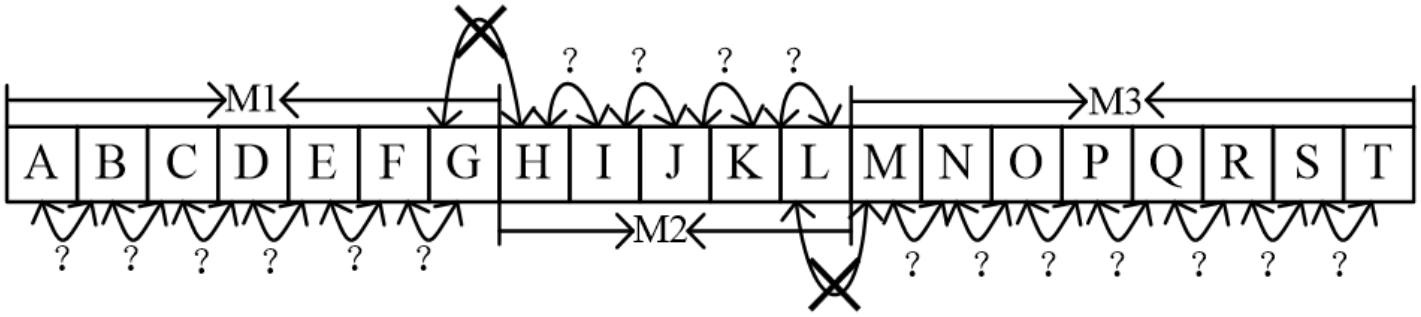

Figure 1

FIG.1 Schematic diagram of Adjacent process interchange strategy

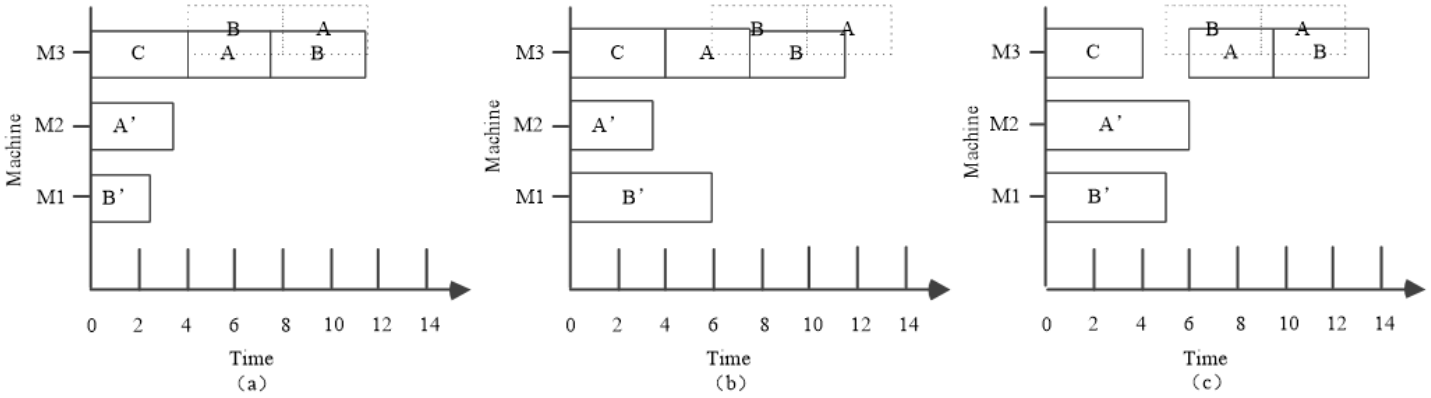

Figure 2

FiG.1 Schematic diagram of three situations of interchange between process *A* and process *B*

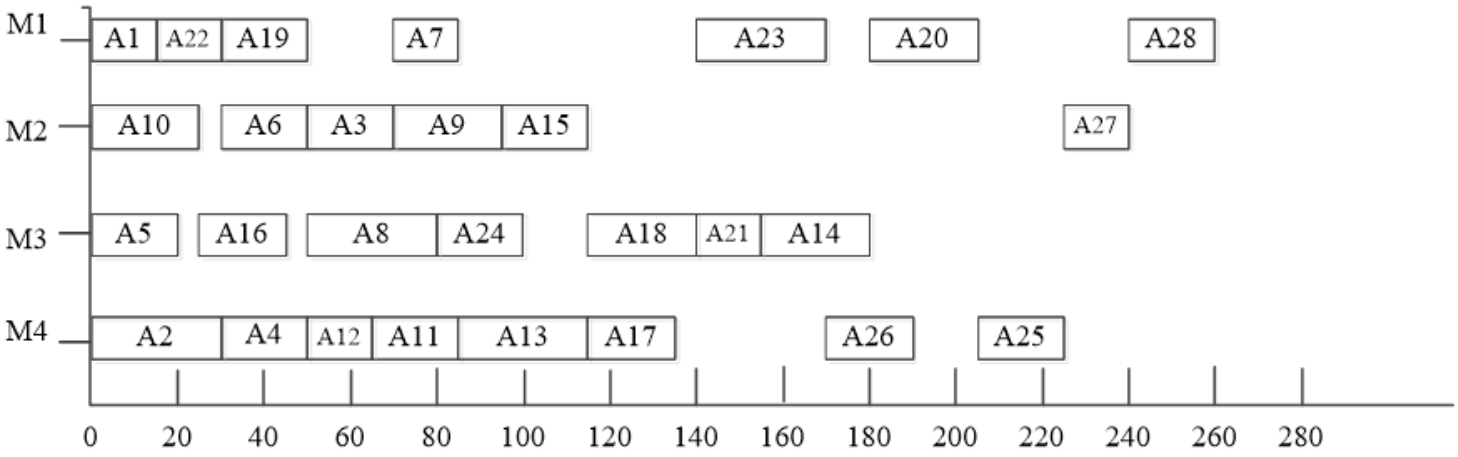

Figure 3

FiG.4 Results of quasi-critical path strategy scheduling

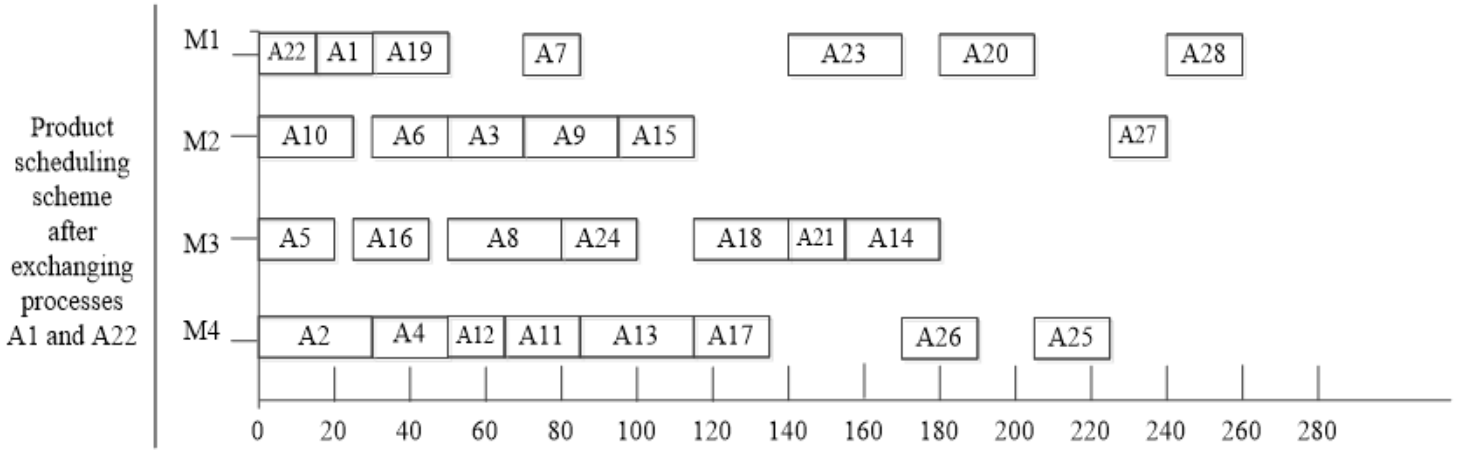

Figure 4

FIG.5 Scheduling scheme obtained from switching process A1 and process A22

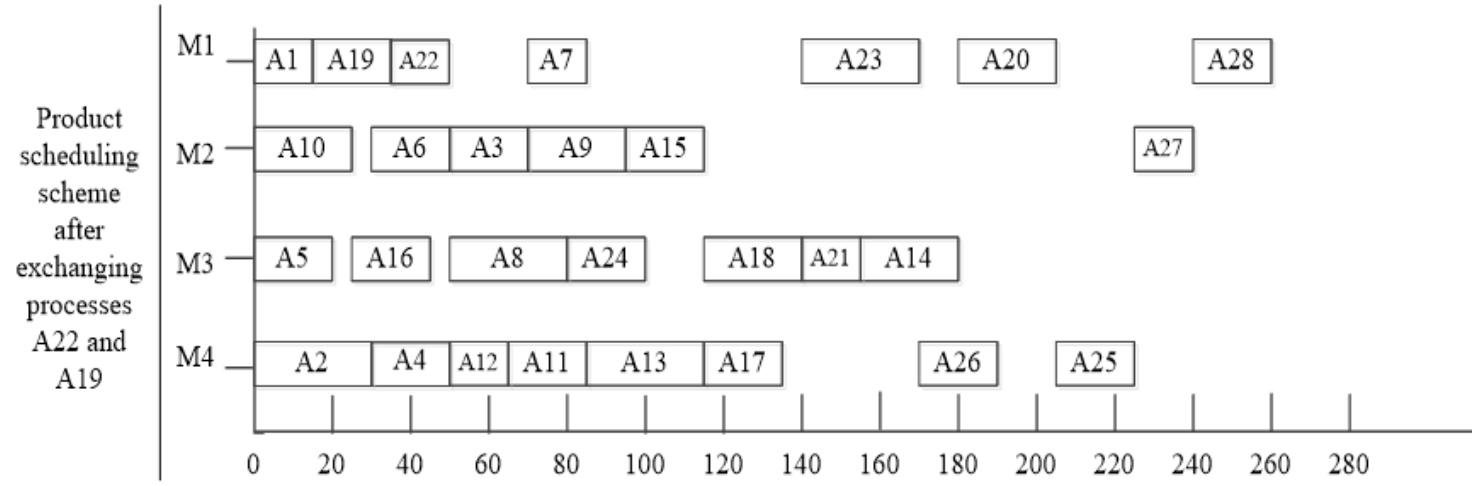

Figure 5

FIG.6 Scheduling scheme of reprogramming process A22 and Process A19

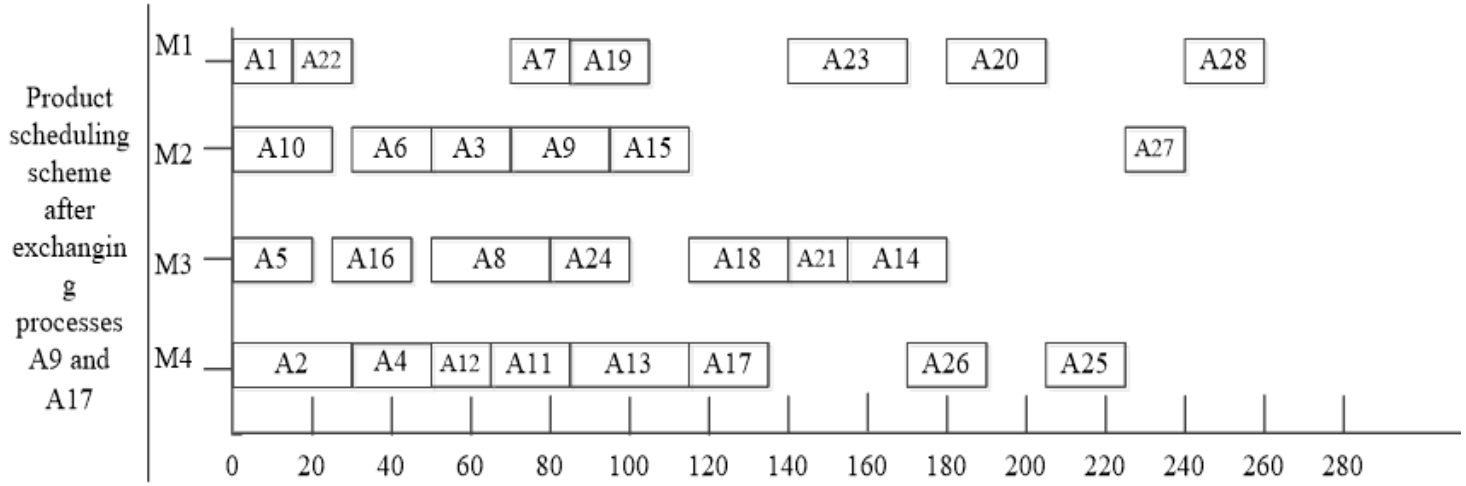

Figure 6

FIG.7 Scheduling scheme obtained from switching process A19 and process A7

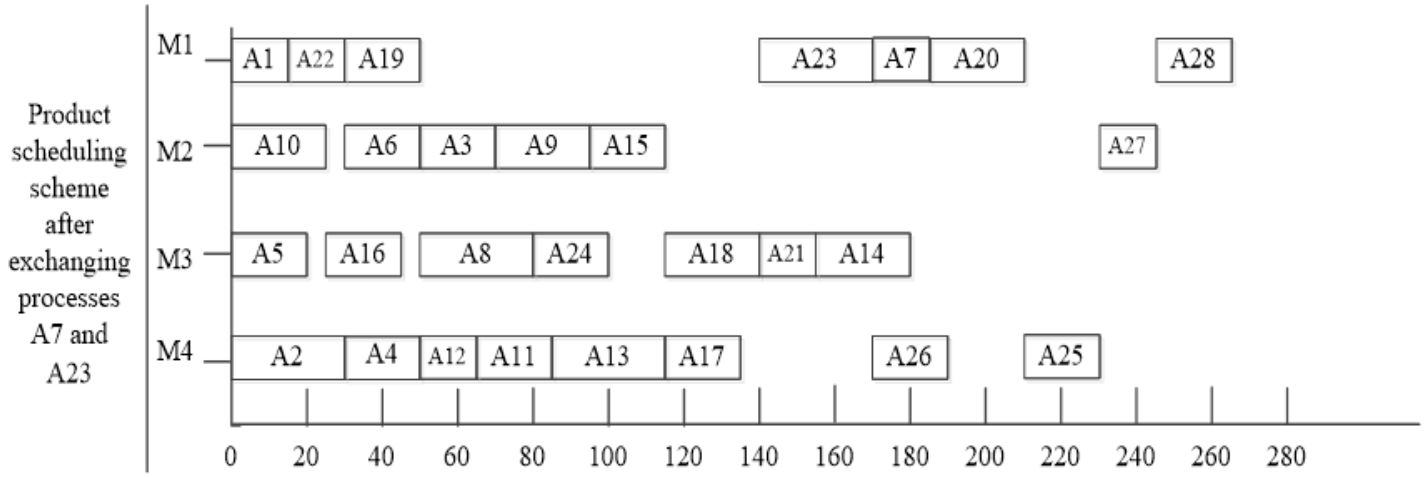

Figure 7

FIG.8 Scheduling scheme obtained from switching process A7 and process A23

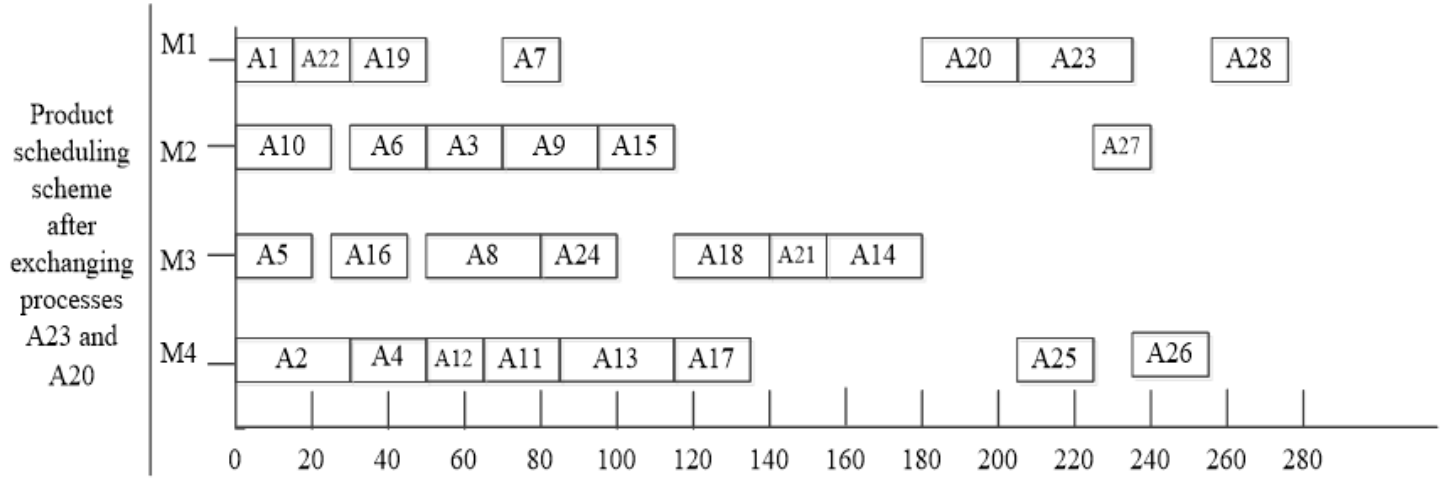

Figure 8

FIG.9 Scheduling scheme obtained from switching process A23 and process A20

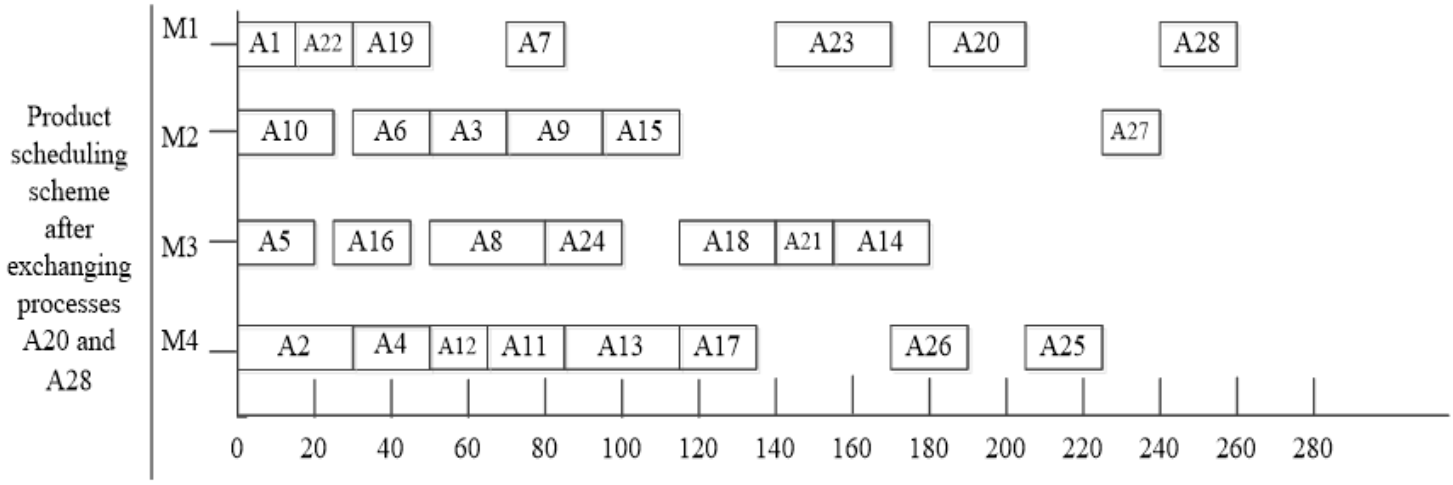

Figure 9

FIG.10 Scheduling scheme obtained from switching process A20 and process A28

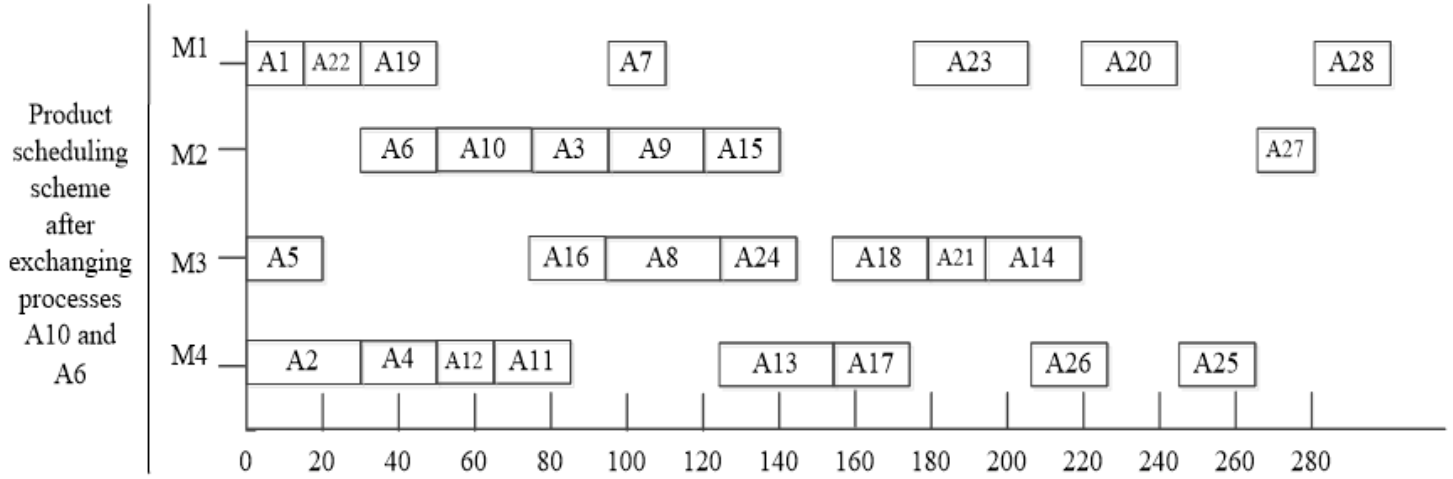

Figure 10

FIG.11 Scheduling scheme obtained from switching process A10 and process A6

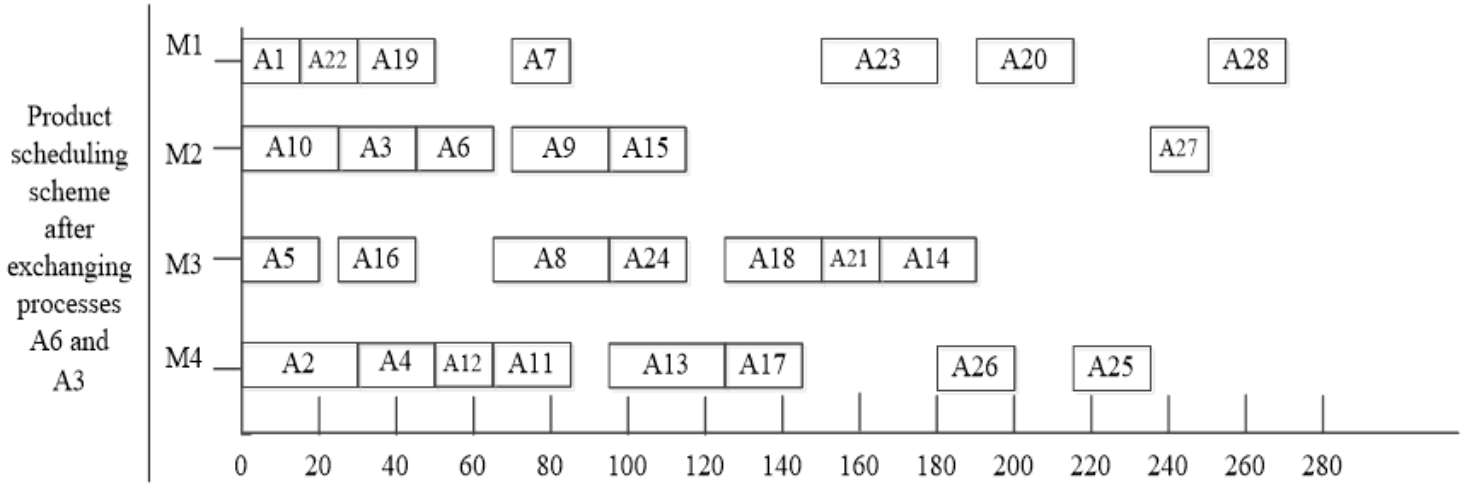

Figure 11

FIG.12 Scheduling scheme obtained from switching process A6 and process A3

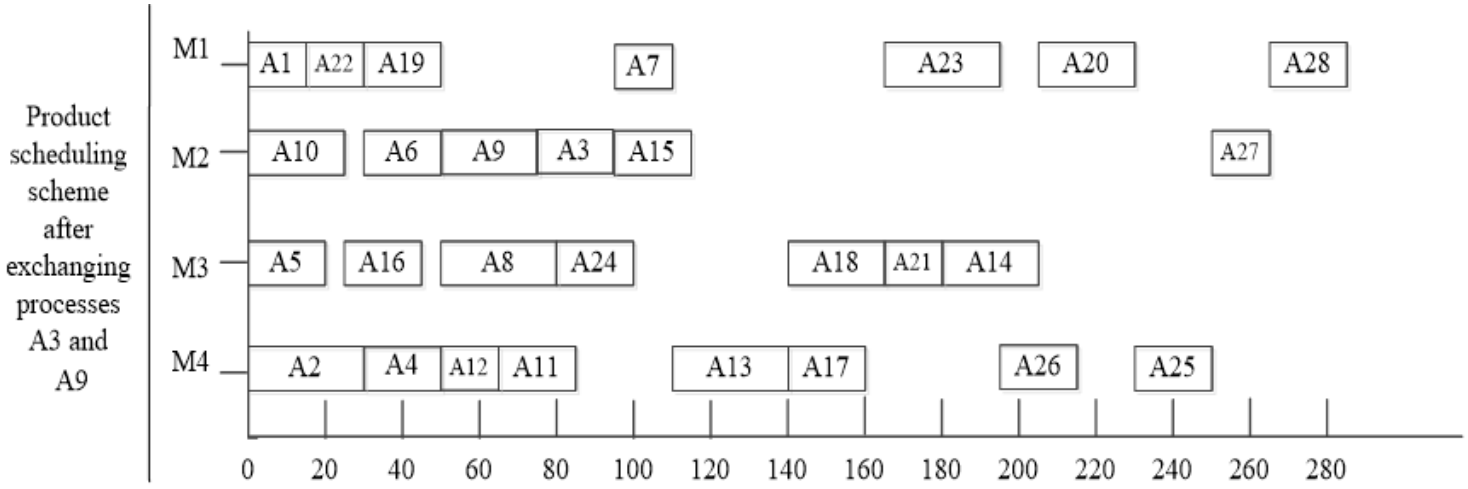

Figure 12

FIG.13 Scheduling scheme obtained from switching process A3 and process A9

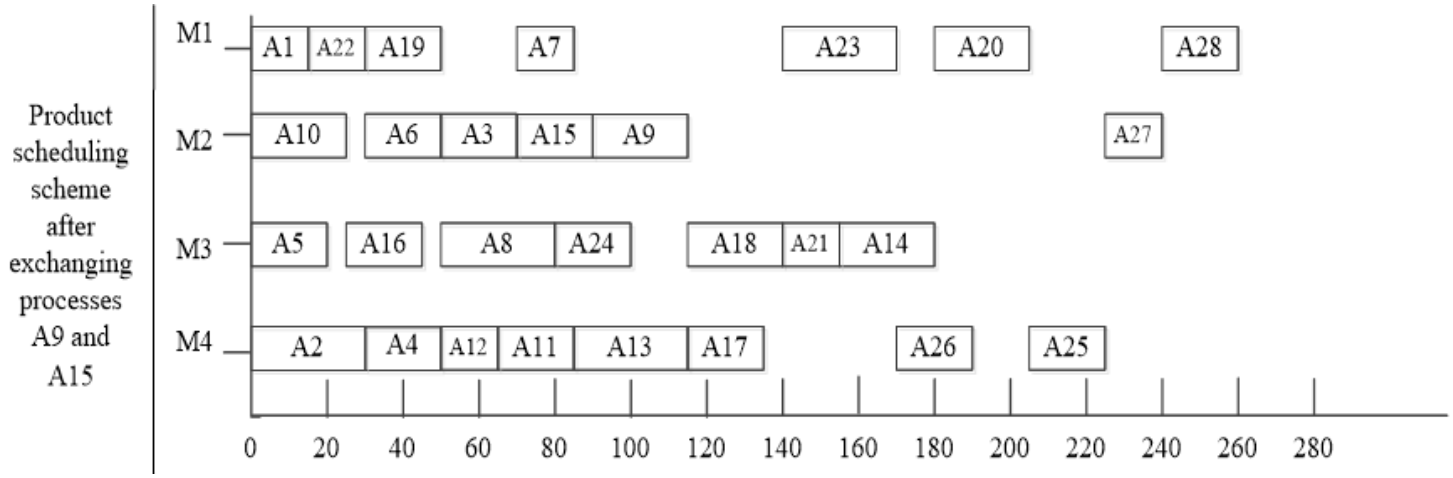

Figure 13

FIG.14 Scheduling scheme obtained from switching process A9 and process A15

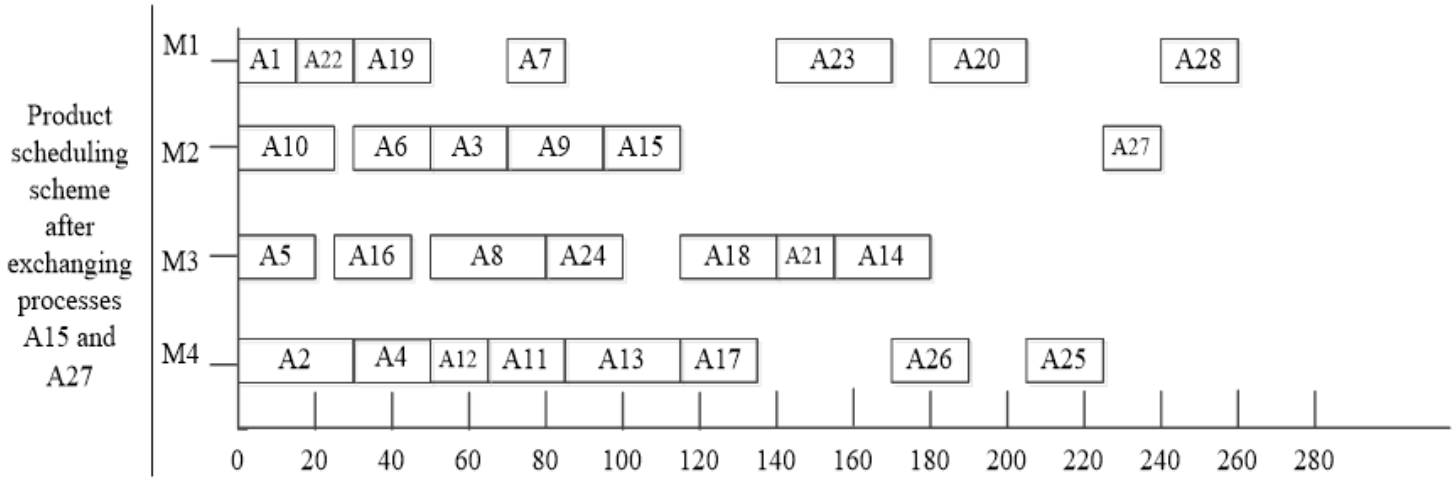

Figure 14

FIG.15 Scheduling scheme of reprogramming process A15 and process A27

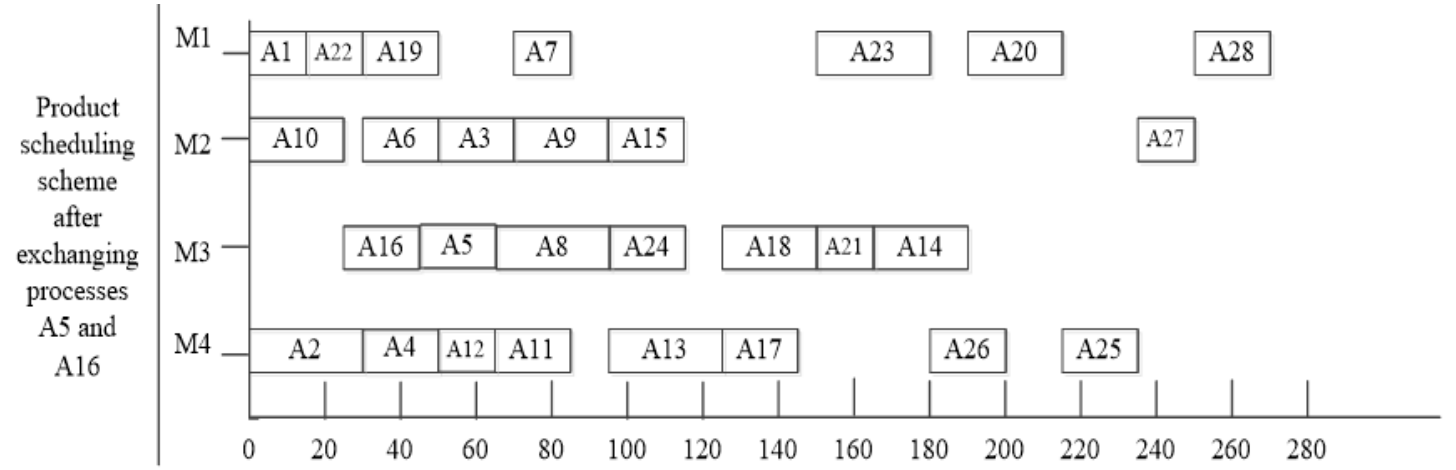

Figure 15

FIG.16 Scheduling scheme obtained from switching process A5 and process A16

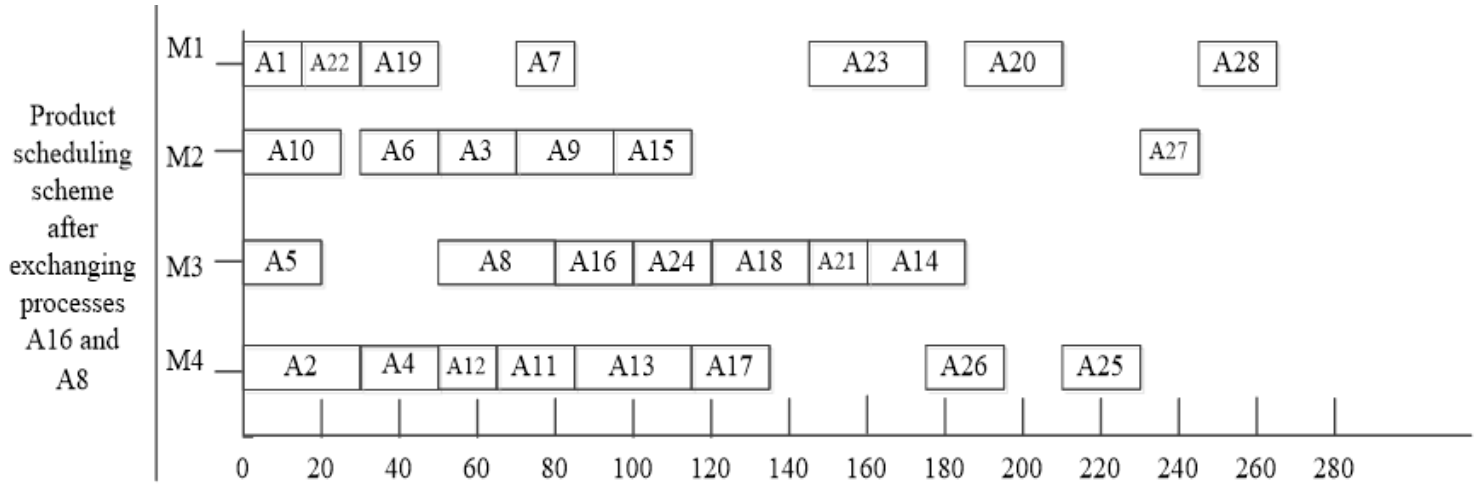

Figure 16

FIG.17 Scheduling scheme of reprogramming process A16 and process A8

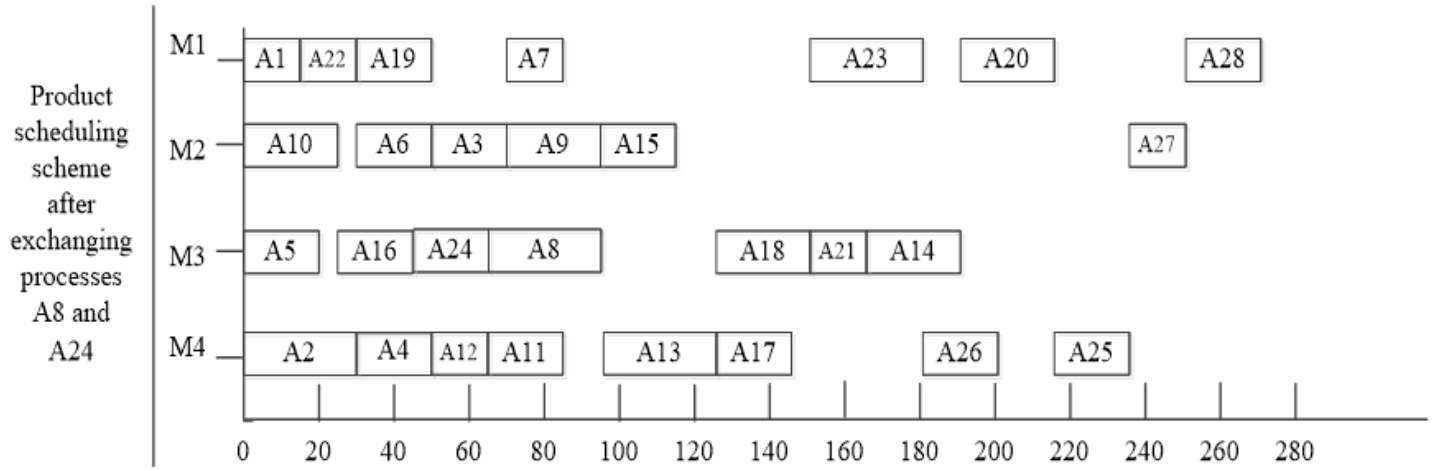

Figure 17

FIG.18 Scheduling scheme obtained from switching process A8 and process A24

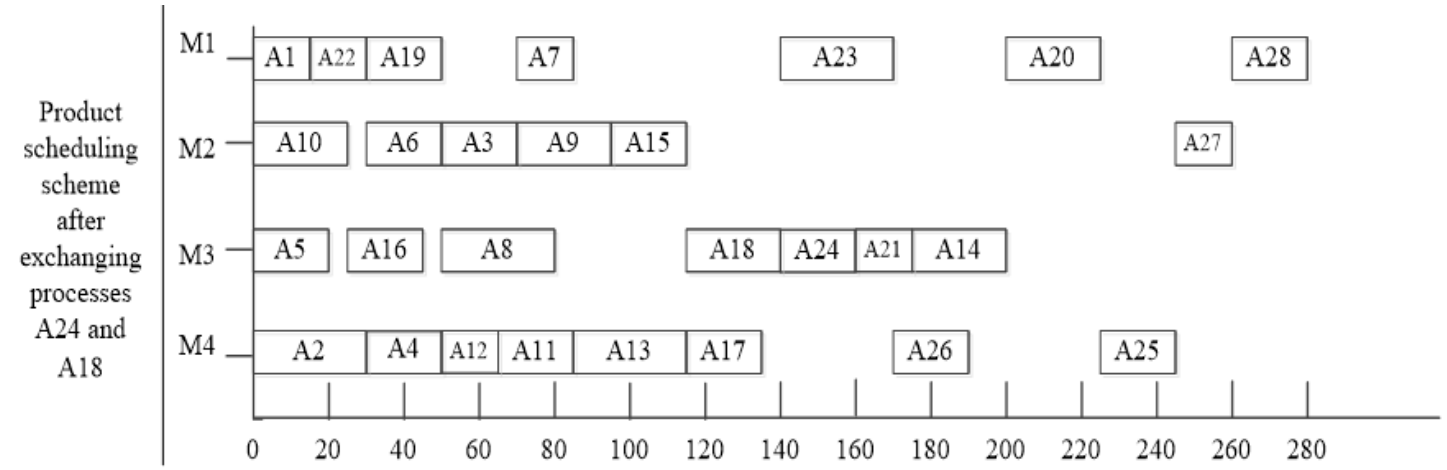

Figure 18

FIG.19 Scheduling scheme of reprogramming process A24 and process A18

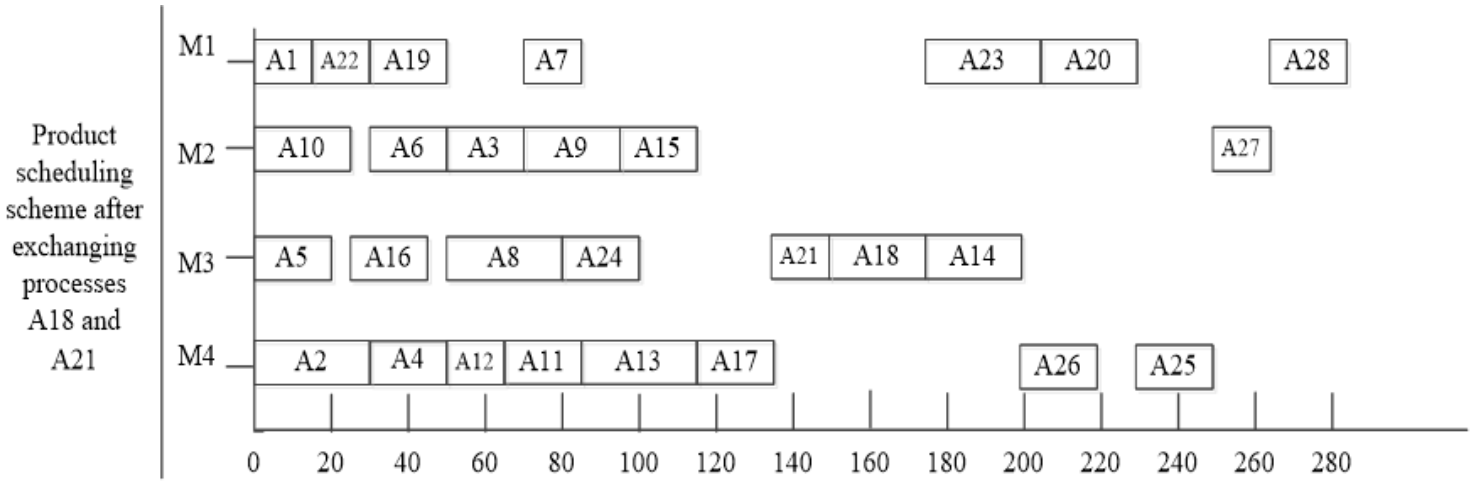

Figure 19

FIG.20 Scheduling scheme obtained from switching process A18 and process A21

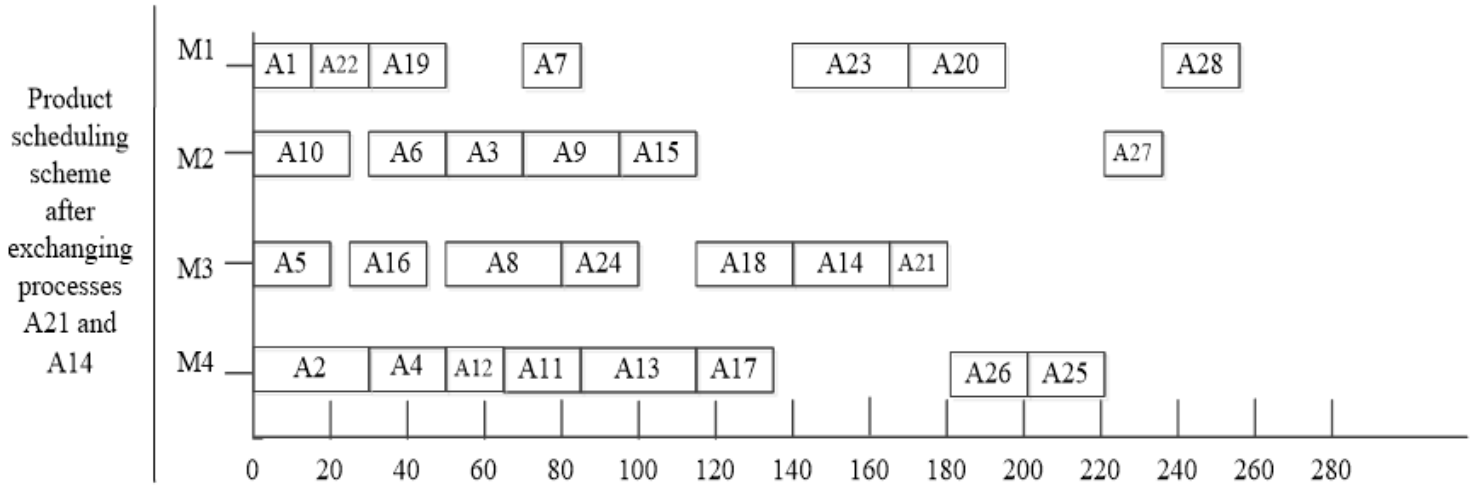

Figure 20

FIG.21 Scheduling scheme obtained from switching process A21 and process A14

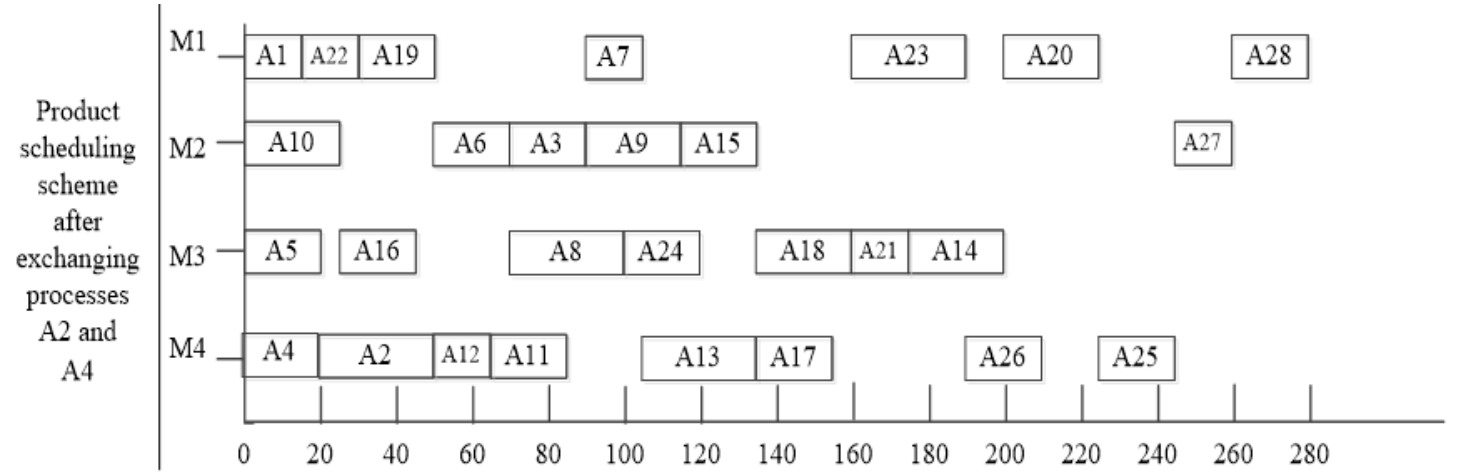

Figure 21

FIG.22 Scheduling scheme of switching process A2 and process A4

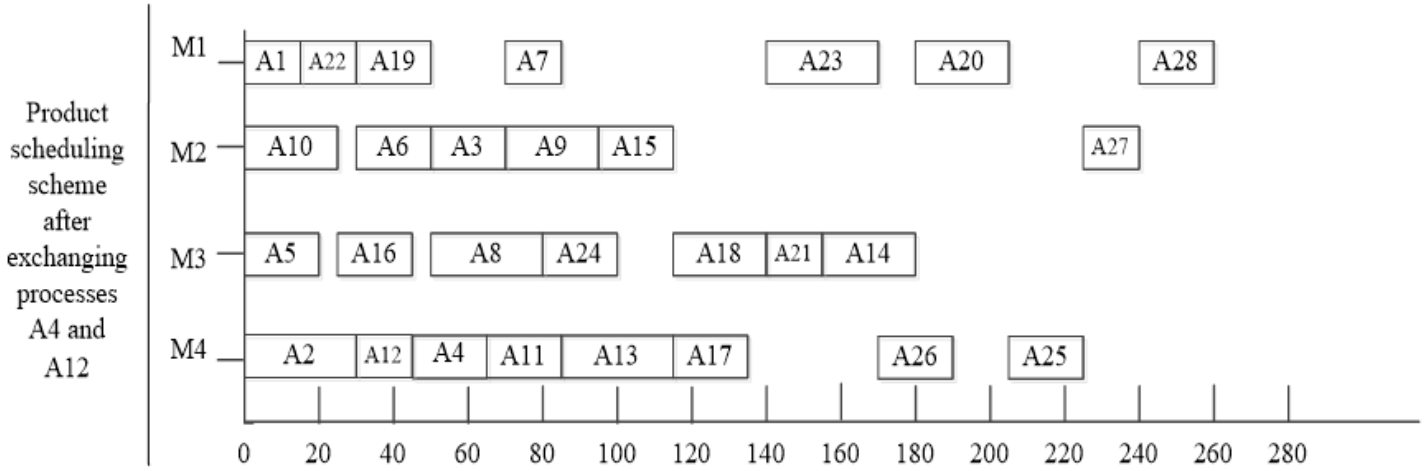

Figure 22

FIG.23 Scheduling scheme of reprogramming process A4 and process A12

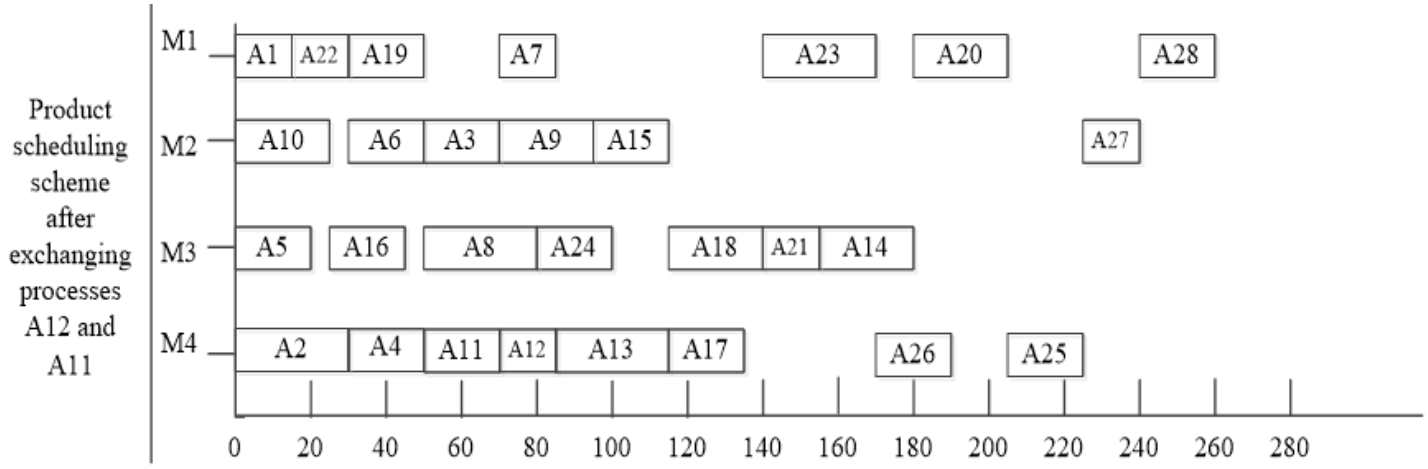

Figure 23

FIG.24 Scheduling scheme of reprogramming process A12 and process A11

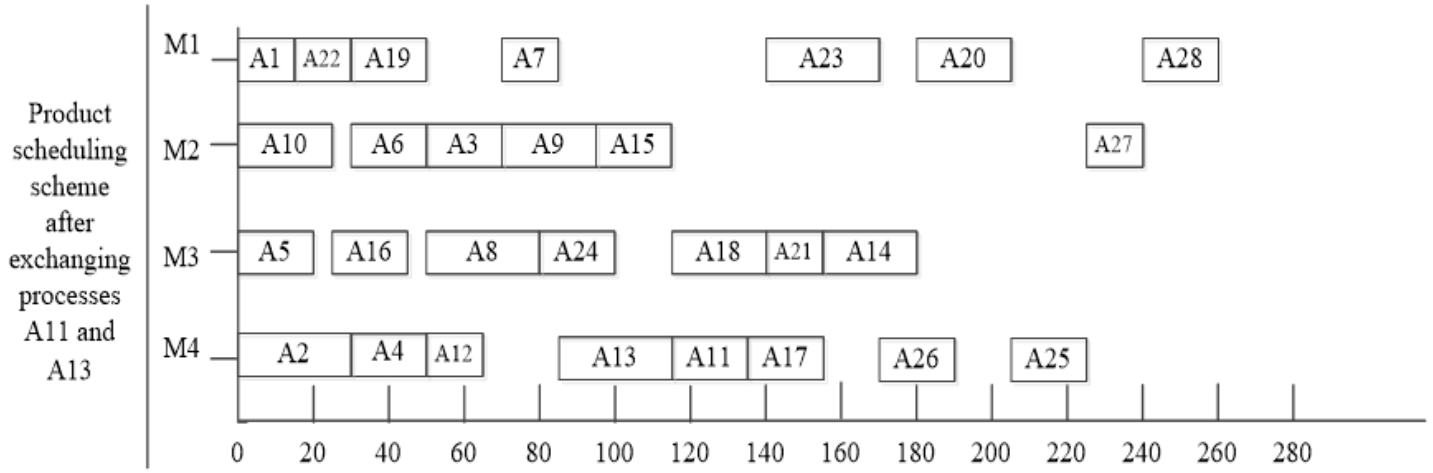

Figure 24

FIG.25 Scheduling scheme of reprogramming process A11 and process A13

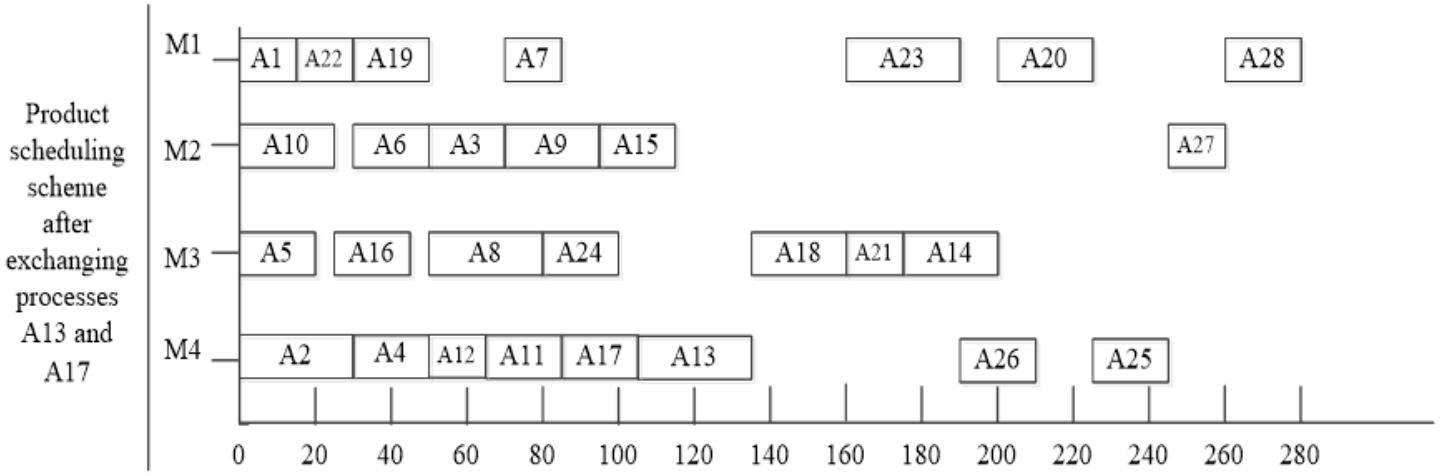

Figure 25

FIG.26 Scheduling scheme of reprogramming process A13 and process A17

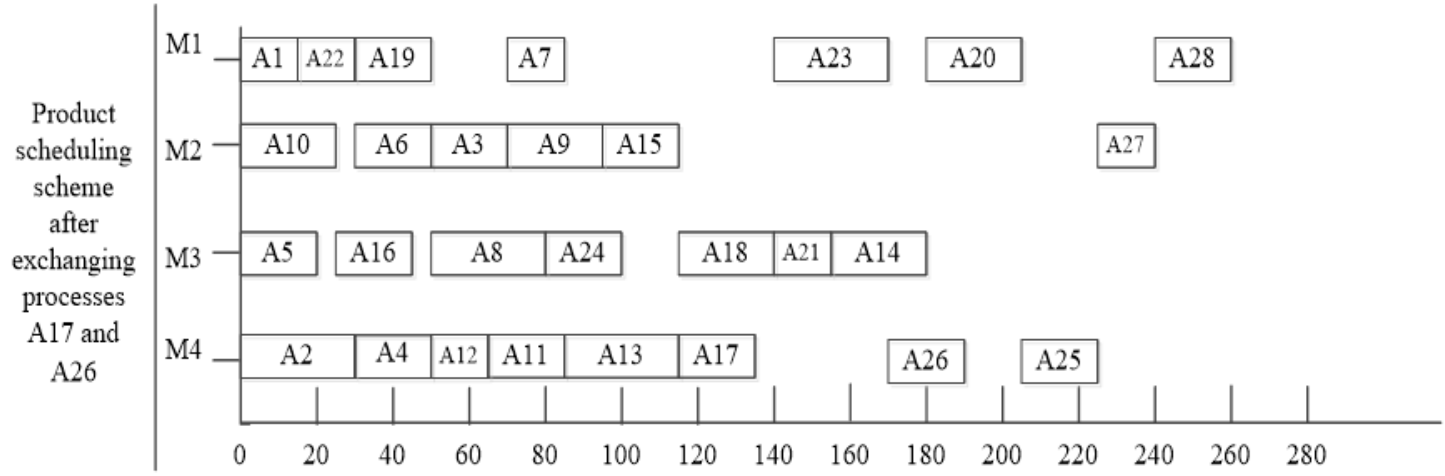

Figure 26

FIG.27 Scheduling scheme obtained from switching process A17 and process A26

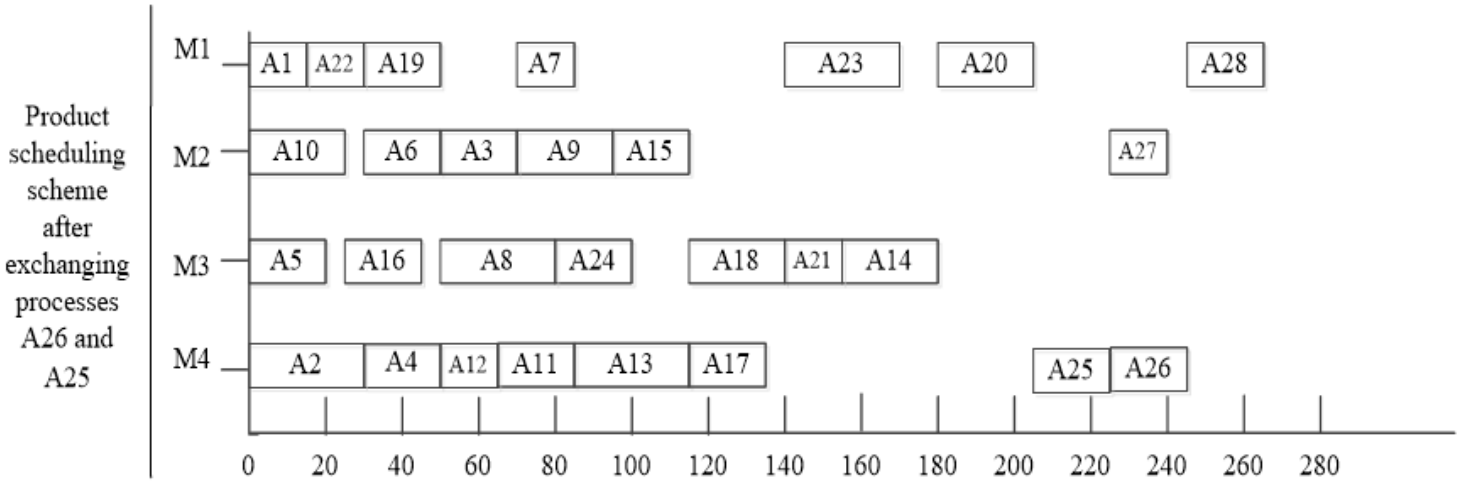

Figure 27

FIG.28 Scheduling scheme of reprogramming process A26 and process A25

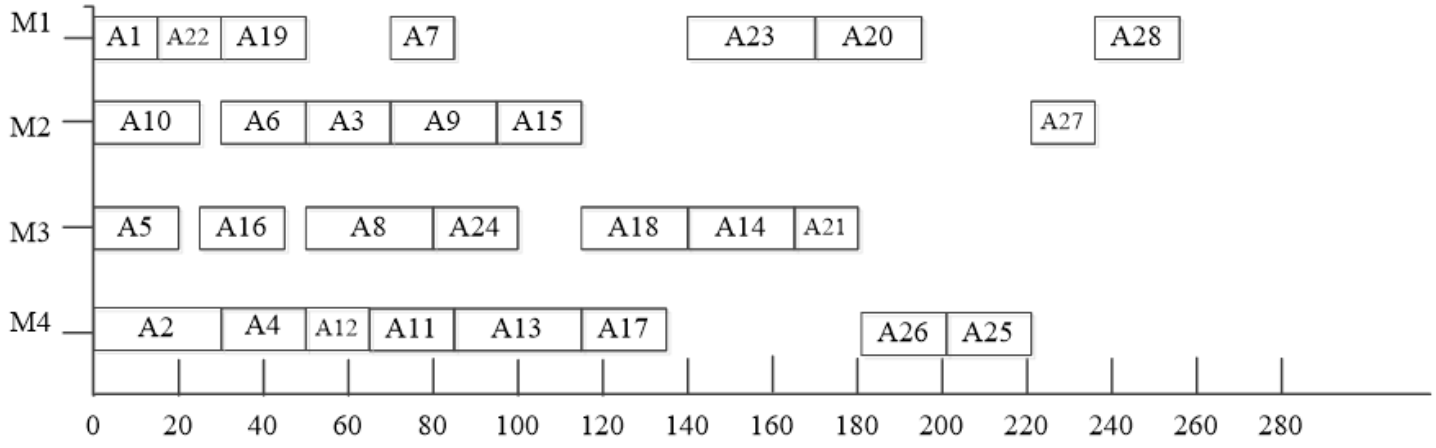

Figure 28

FIG.29 Final product scheduling scheme

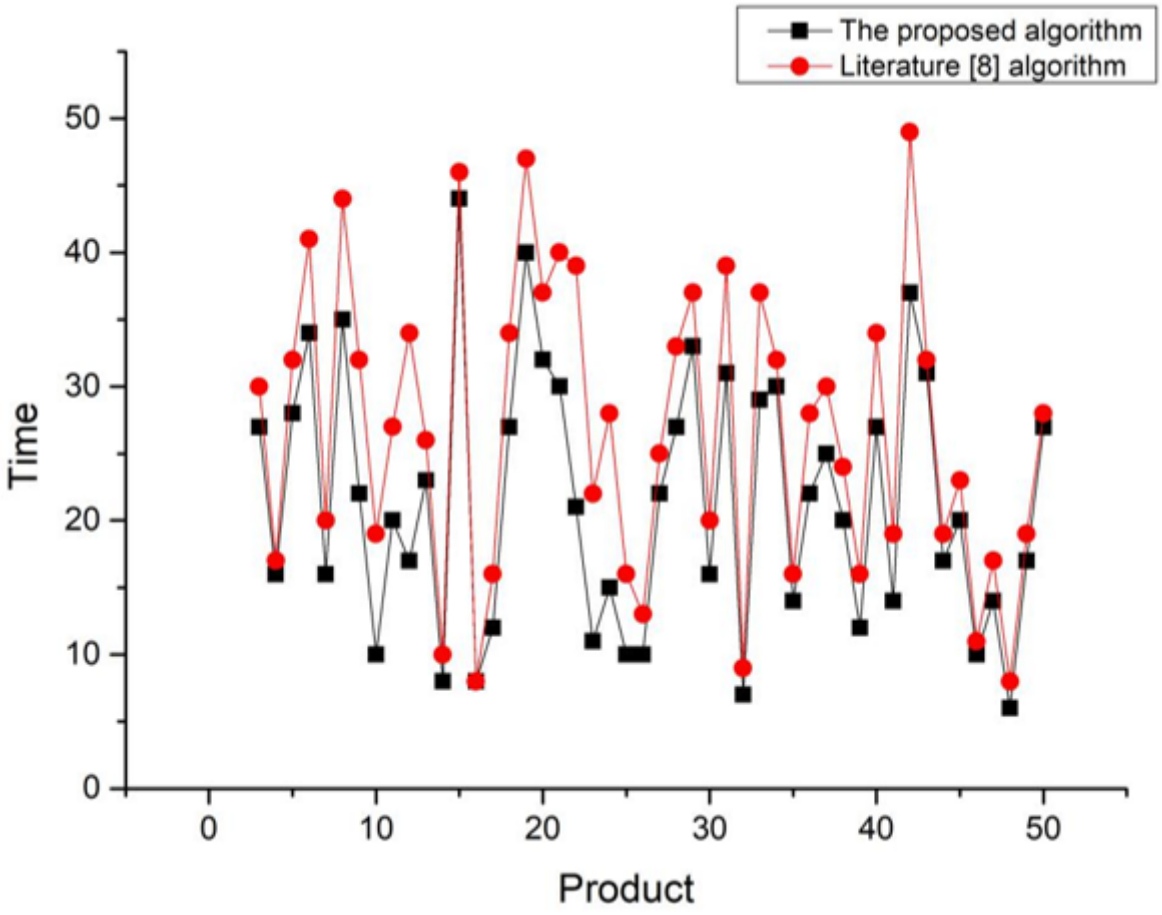

Figure 29

FIG.30 Comparison chart of experiment 1 results

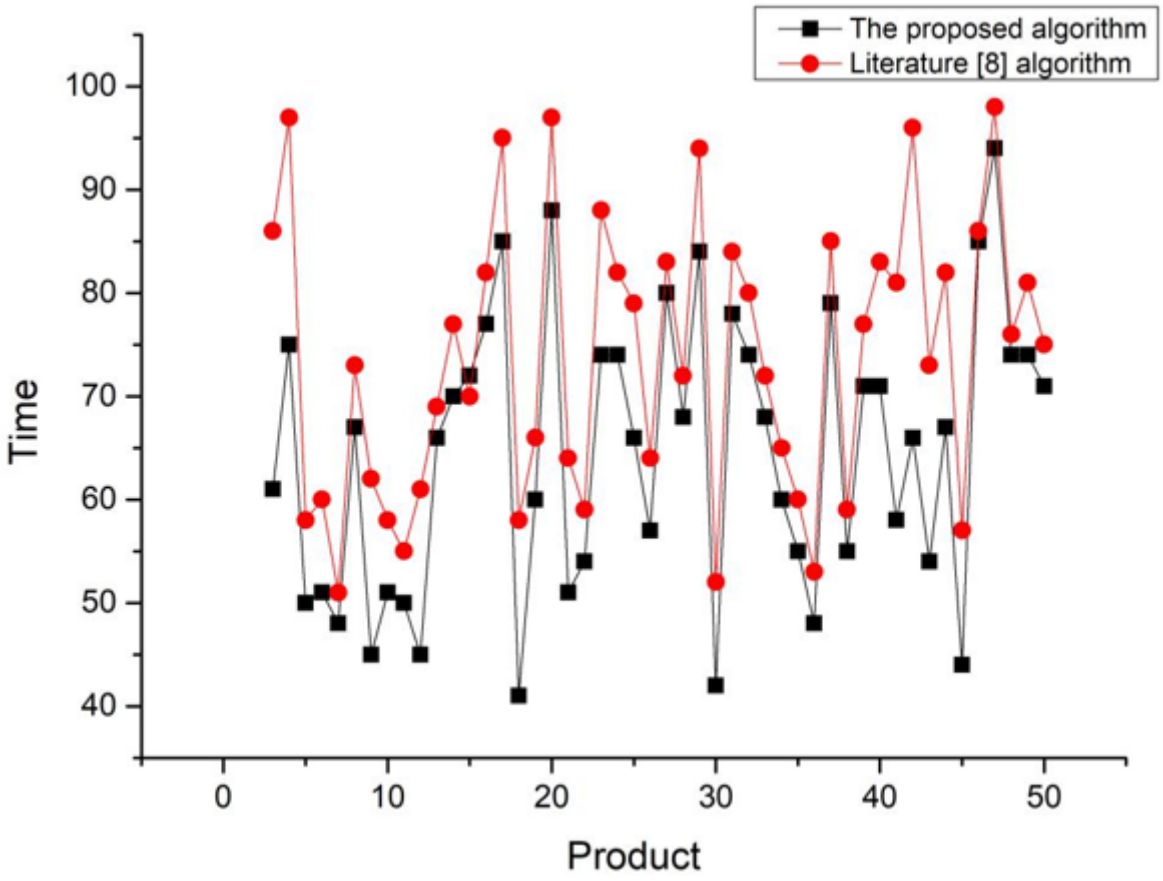

Figure 30

FIG.31 Comparison chart of experiment 2 results

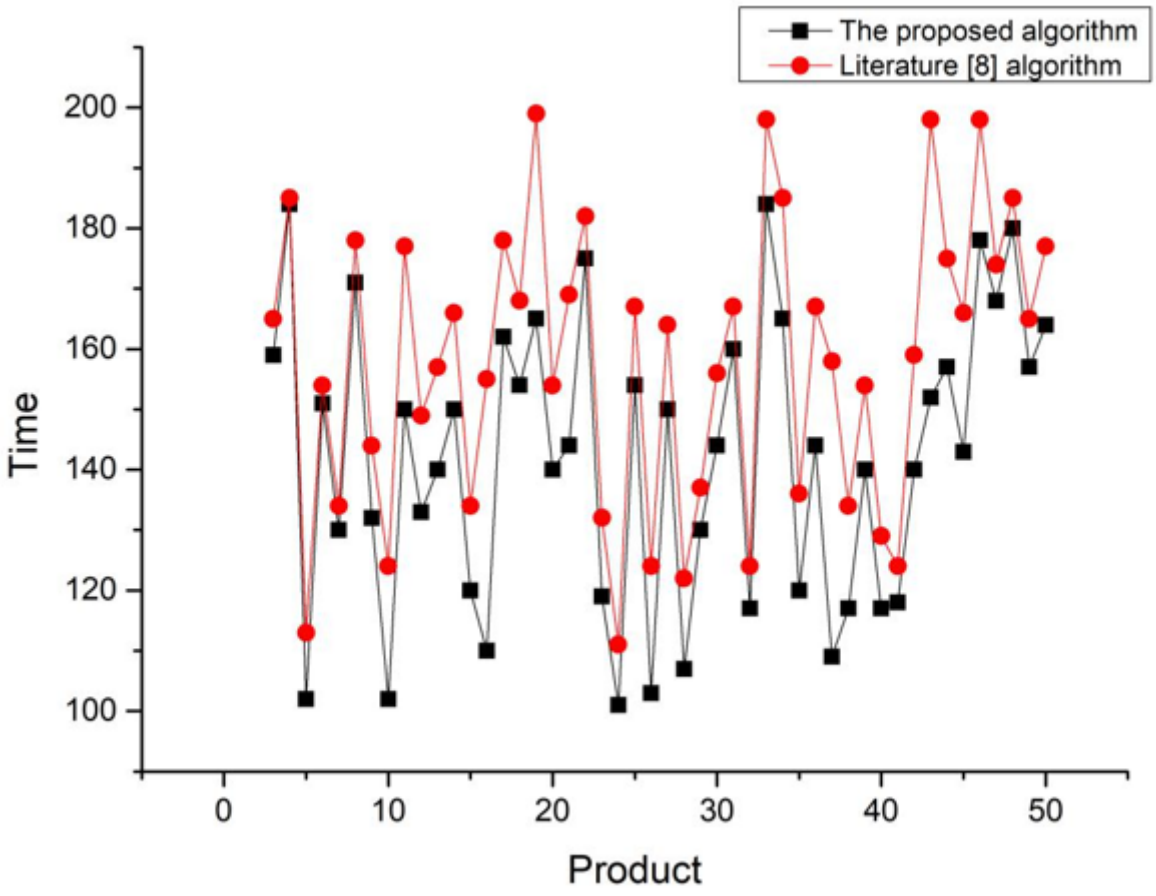

Figure 31

FIG.32 Comparison chart of experiment 3 results

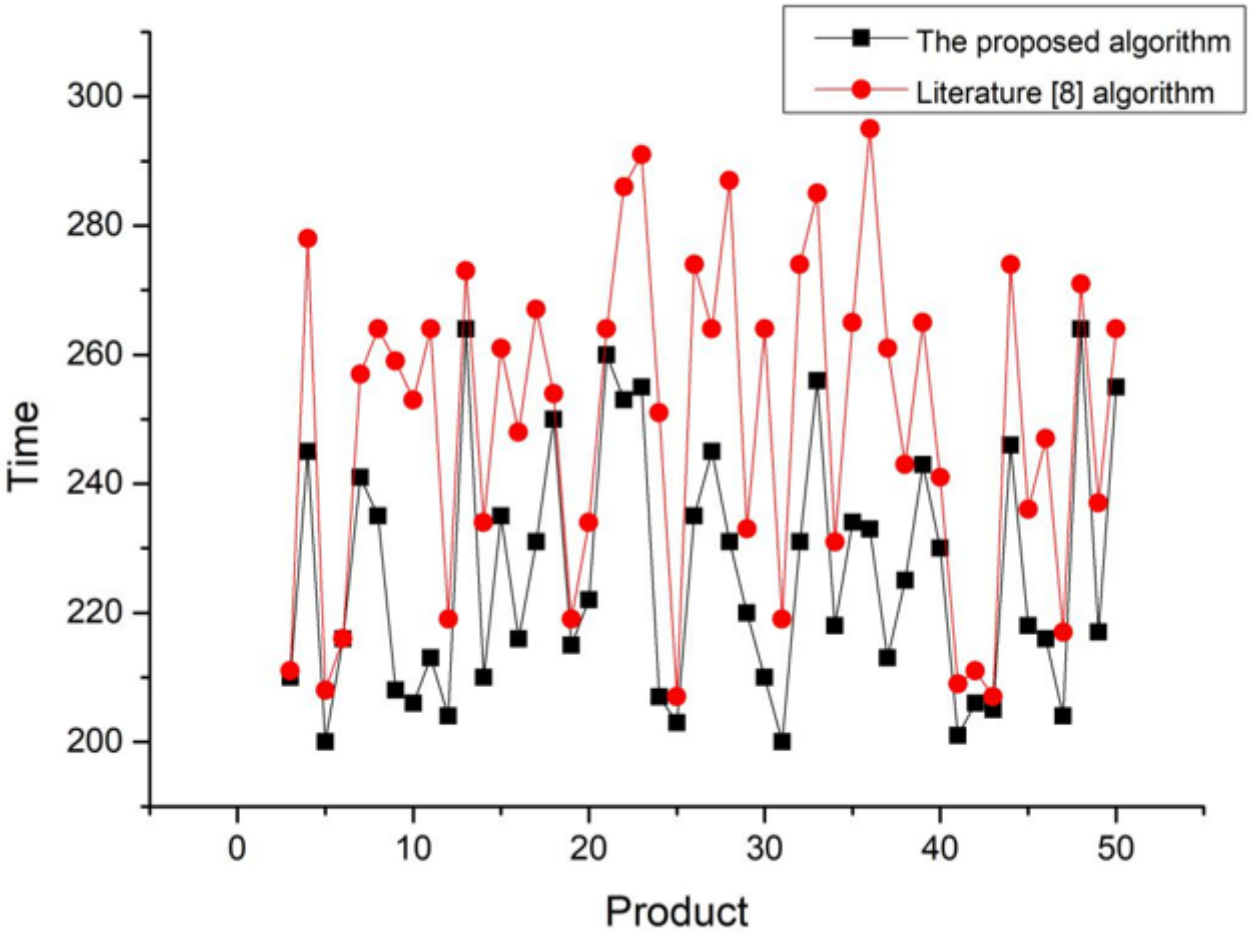

Figure 32

FIG.33 Comparison chart of experiment 4 results

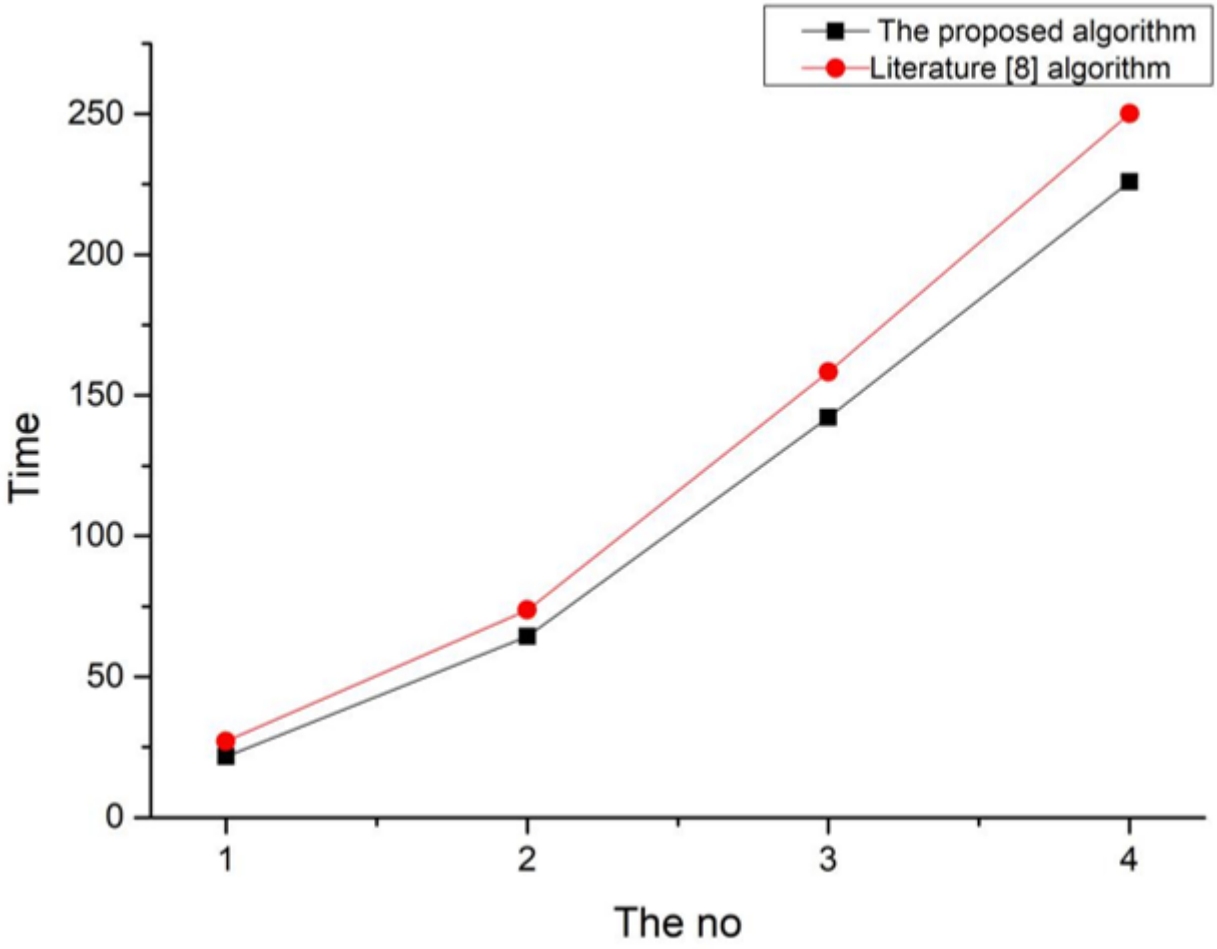

Figure 33

FIG.34 The mean comparison chart of the above four experimental results