A Decomposition and Coordination Scheduling Method for Flow-shop Problem Based on TOC[1)]

ZHANG Hong-Yuan XI Yu-Geng GU Han-Yu

(*Institute of Automation, Shanghai Jiaotong University, Shanghai* 200030)

(E-mail: irry_zhang@sjtu.edu.cn, ygxi@sjtu.edu.cn, guhy@sjtu.edu.cn)

Abstract There are many flow shop problems of throughput (denoted by FSPT) with constraints of due date in real production planning and scheduling. In this paper, a decomposition and coordination algorithm is proposed based on the analysis of FSPT and under the support of TOC (theory of constraint). A flow shop is at first decomposed into two subsystems named PULL and PUSH by means of bottleneck. Then the subsystem is decomposed into single machine scheduling problems, so the original NP-HARD problem can be transferred into a serial of single machine optimization problems finally. This method reduces the computational complexity, and has been used in a real project successfully.

Key words Flow-shop scheduling, TOC, bottleneck, decomposition, coordination

1 Introduction

The flow-shop is an important research model for flexibe manufacture shop, and there have been many research works done about it[1]. In general, the scheduling problem of flow-shop is NP-hard even if there are more than one machine in shop. So many researchers focus on the approximation algorithm. There are many research on MAKESPAN problems[2], but the throughput during certain period is also an important objective for manufacturer in practice for the flow-shop problem of throughput (denoted by FSPT). Many algorithms for such problem are not practical in real project because of its complexity. So it is necessary to study algorithms with less computational complexity and of acceptable precision to satisfy the project′s need.

Theory of constraints (denoted by TOC[3]) is a kind of scheduling methodology, which was brought forward firstly in an optimization scheduling software named OPT, by a research team leading by Ei Goldratt in 1979, and it was accepted as a theory by APICS (American Production and Inventory Control Society) after 1998[2)]. Some regard TOC as a kind of scheduling method, because its main character is to improve the production performance by means of adjusting some critical machines' production named bottleneck. But there is also another comprehension that TOC is a philosophical concept, which just provides strategy and direction other than an exact method, and detail algorithm should be constructed according to concrete problems[3].

According to TOC, one can simplify a general NP-hard scheduling problem by solving the scheduling of bottlenecks in the system, and achieved approximation result will be satisfactory to some extent. In this paper we try to simplify a FSPT problem with multiple machines, and coordinate the whole flow shop production according to the scheduling result of bottleneck. Some concepts such as PULL system, PUSH system and single machine model are introduced firstly. Then the detail algorithm is described. A simple FSPT example is given and the result is analyzed.

2 Basic concept

2.1 Decomposition principle

Bottleneck is the key issue in TOC. The scare resource in production system is called bottleneck in general[3]. The objective of production scheduling is to make good use of whole resource and reduce production cost. Since some critical machines determine the efficiency of whole production line, making good use of bottleneck to improve efficiency of whole line becomes the key issue of TOC. The objective

1) Supported by National Natural Science Foundation of P. R. China (60274013)

2) Robert E. Fox, The Theory of Constraints—Fad or Future, www.tocc.com

Received November 26, 2003; in revised form May 20, 2004

of the whole system may be kept to some extent if bottleneck′s production is optimized. The above is the main idea of the algorithm in this paper. In order to describe the decomposition and coordination algorithm clearly, we assume there is only one bottleneck in FSPT.

In [4,5], PULL system and PULL system are described as operational paradigms: "In a push system, a preceding machine produces parts without waiting for a request from the succeeding machine. On the other hand, in a pull system a preceding machine produces parts only after it receives a request from the succeeding machine". So in a PULL system, the succeeding machine controls the preceding operation, and in a PUSH system, the preceding machine pushes the jobs to the next machine till it was completely processed in the production line. Referring to the concept of bottleneck, we make definitions of PULL system and PUSH system in this paper:

PULL system. Bottleneck and upstream machines form a PULL system. Bottleneck controls the production rhythm of the whole line, and other machines′ production action is pulled by bottleneck. In a PULL system, the optimization performance of the whole system is mostly affected by optimization result of bottleneck, and what other machines need to do is just to satisfy the release need of downstream machines.

PUSH system. Bottleneck and downstream machines form a push system. Upstream machines control the jobs′ release time and other production parameters. So different to PULL system, the upstream machines mostly affect the whole system performance.

2.2 Single machine model

There has been much research on single machine model in literature[2]. In this paper, three types of single machine models are used in sub-problems.

1) $1//\Sigma C_j, C_j$ represents the completion time of job j, so the objective of this problem is to minimize the sum of completion time of all jobs. It is equivalent to the problem $CMax$ on makespan, but the jobs with shorter processing time should be processed firstly in $1//\Sigma C_j$. The optimization result of $1//\Sigma C_j$ could be obtained by the dispatch rule SPT (Shortest Processing Time)[2].

2) $1//L_{Max}, L_{Max} = \mathrm{Max}(L_j, j = 1, \cdots, n), L_j = D_j - C_j, D_j, C_j$ represent the due date and completion time of job j respectively. This problem is called lateness problem, and aims at minimizing the maximal lateness of jobs. The dispatch rule EDD (Earliest Due-data) can be used to get the optimization result of it[2].

3) $1/r_j/\Sigma C_j, r_j, C_j$ represent release time and completion time of job j, respectively. The optimization objective is to minimize the sum of jobs′ completion time. The problem is NP-hard, and some associated algorithms of it can be found in [2]. In this paper, we solve it by a simple dispatch rule, named first in first out (FIFO).

3 The Decomposition and coordination algorithm for FSPT

3.1 The model of FSPT

In this paper, we will pay more attention on decomposition and coordination method for FSPT, so we use a simple flow shop model having J discrete processing machines. There are R kinds of parts to be processed in this flow shop. We assume that WIP (work in process) is zero before and after production, and all jobs do not need setup time, but there is a least delivery amount for any part. The objective of FSPT is to maximize the throughput of the part having the least product in this flow shop during a certain period. In this paper, a part means a kind of workpiece, and a job means a processing step. The following symbols are used: $i = 1, \cdots, I$, index of jobs on machine; $r = 1, \cdots, R$, index of part types; $j = 1, \cdots, J$, index of machines; $X_{i,r,j}$ binary variable, if job i in machine j belongs to part r then $X_{i,r,j} = 1$, else $X_{i,r,j} = 0$; $X_{r,j}, X_r$ the product of part r in machine j and in flow shop respectively; since every machine processes the same jobs in flow shop, so here $X_{r,j} = X_r\ j = 1, \cdots, J$; $P_{r,j}$ data, the processing time of part r on machine j; T_{end} requested delivery time; D_r the least amount of part r must be processed before T_{end}; $CMax_j$ the maximal completion time on machine j; $ST_{i,j}, ET_{i,j}$ the start time and completion time of job i on machine j.

The mathematical model is given as follows.

$$\max\ (\min(X_r, r = 1, \cdots, R))$$

$$\text{s.t.} \sum_r X_{i,r,j} = 1, \text{ machine } j \text{ can process only one kind of part in job } i, X_r = \sum_i X_{i,r,j};$$

$$ET_{I,J} \leqslant T_{end}, \text{ all jobs must be completed before delivery time } T_{end};$$

$$\sum_i X_{i,r,J} \geqslant D_r, r = 1, \cdots, R, \text{ the number of part } r \text{ processed must be more than } D_r;$$

$$ST_{i,j} \geqslant ET_{i,j-1}, ST_{i,j} \geqslant ET_{i-1,j}, ET_{i,j} = ST_{i,j} + \sum_r X_{i,r,j} P_{r,j}, ST_{1,1} = 0 \tag{1}$$

This is an mixed integer programming problem, with known $P_{r,j}, D_r$ and delivery time T_{end}, the optimization variables are $X_{i,r,j}, ST_{i,j}$ and number of jobs in the shop I. The optimization objective is maximizing the throughput of shop before T_{end}. If $J > 2$, it is a NP-hard problem, and is impossible to get optimal solution in general, so it is necessary to study approximation method for FSPT. In order to reduce the computational complexity and satisfy a precision to some extent, we propose a decomposition and coordination algorithm.

3.2 Decomposition and coordination algorithm

In problem (1), the optimization objective is to maximize the throughput of whole production line. According to TOC, the efficiency of bottleneck will mostly affect the performance of the whole system. So we try to pay more attention to bottleneck′s efficiency. If the throughput of bottleneck is maximized, and other machines process jobs according to the request of bottleneck, the performance of the whole line will be improved to large extent. Such a schedule will be an approximation to original problem. After the bottleneck is chosen, we assume that the throughput of whole system is determined by bottleneck. So the problem (1) can be transferred to a single machine scheduling problem. For scheduling of other machines′ production, we introduce the PULL system and PUSH system defined above. According to the definition of PULL, the downstream machine controls the production of upstream machine. We can build single machine problem $1//L_{Max}$ from bottleneck to upstream machines recursively. According to the definition of PUSH, the upstream machines control the downstream machines′ production. We can build single machine problem $1/r_j/\Sigma C_j$ from bottleneck to downstream machines recursively. When solving scheduling problem of bottleneck, we do not know the exact start processing time and due date of bottleneck. They will be estimated in model at first. The real completion time can only be obtained after production schedule is made. We will check the constraints on the original problem, and if completion time of flow shop is later than the requested delivery time, the throughput of the bottleneck should be reduced. The above procedure will be done iteratively.

As show in Fig. 1, the flow shop is decomposed into three parts: upstream machines, bottleneck and downstream machines. A PULL system and a PUSH system are then established according to the above definitions. The PULL system is further decomposed into single machine scheduling problem $1//L_{Max}$ and is solved from bottleneck K to head machine 1. The PUSH system is further decomposed into single machine scheduling problem $1/r_j/\Sigma C_j$ and is solved from bottleneck K to last machine J.

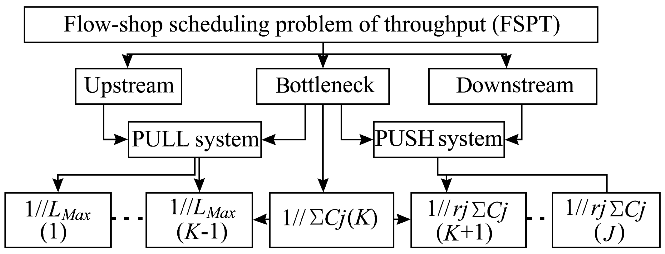

Fig. 1 Structure of decomposition for FSPT

The main steps of decomposition and coordination algorithm for FSPT are given as follows.

Step 1. Choose bottleneck and estimate the throughput X_r of part $r, r = 1, \cdots, R$;

Step 2. Optimize the scheduling problem $1//\Sigma C_j$ of bottleneck K, and modify X_r;

Step 3. Build and solve problems $1//L_{Max}$ from $j = K-1$ to 1 according to the optimized result of succeeding machine;

Step 4. Build and solve problems $1/r_j/\Sigma C_j$ from $j = K+1$ to J according to the optimized result of proceeding machine;

Step 5. Let the start processing time of the first machine $ST_{1,1} = 0$, and calculate the maximal completion time of flow shop $CMax_J$;

Step 6. If $CMax_J > T_{end}$, reduce X_r and goto Step 3, otherwise complete algorithm.

Next we will describe the main steps of the above algorithm in detail.

3.3 Choose and schedule bottleneck

In TOC, choice of bottleneck is critical, and there are many methods in literature for different scheduling models[3,4,6].

For problem (1), the optimization objective is to maximize throughput, and the optimization variables are sequence of parts and number of jobs. Since in flow shop all machines have the same throughput, we take the same optimization objective as in (1) for bottleneck. In original problem (1), one critical constraint is delivery time of flow shop. But we can not give its delivery time for bottleneck, so we assume the bottleneck has the same delivery time as the whole line at the beginning of the algorithm. It may result in an unfeasible schedule, so some coordination and modification are required. The following is the single machine model.

Let machine K be the bottleneck, refer to problem (1), the mathematical model of the bottleneck scheduling can be described as follows.

$$\begin{aligned}&\max\ (\min(X_r, r = 1,\cdots,R))\\ &\text{s.t. } ET_{I,K} \leqslant T_{end},\ ST_{1,K} = 0,\ X_r = \sum_r X_{i,r,K} \geqslant D_r,\ r = 1,\cdots,R\end{aligned} \tag{2}$$

The optimization variables of problem (2) are number of jobs and process sequence of parts, and the constraint is the delivery time. In the dual problem of (2), we assume that the number of jobs is known, so the optimization objective becomes completion time. Since in problem $1//\Sigma C_j$, the job with shorter processing time will be processed firstly, which will improve the efficiency of downstream machines in flow shop, so we use $1//\Sigma C_j$ as the dual problem of problem (2). In order to use scheduling model $1//\Sigma C_j$, the number of jobs of different parts is estimated as first. Without losing generality, we assume bottleneck can process more TH_K jobs, which more than delivery needed. TH_j denotes the productivity of machine j and $\triangle_r$ denotes as the increased number to part r. The product of different parts can be estimated as follows:

Let $\triangle_0 = (TH_K - \sum_{r=1}^{R} D_r), \triangle_r = \triangle_0, r = 1,\cdots,R$; then the initial product of part r

$$X_r = D_r + \triangle_r \tag{3}$$

After all products of R kinds of parts are fixed on, problem (2) can be transferred into $1//\Sigma C_j$ problem and SPT can be used to get optimal solution[2]. If $CMax_K > T_{end}$, assuming $X_i = Max(X_r, r = 1,\cdots,R)$ and $\triangle_i > 0$ then jobs of part i with maximal processing number could be reduced, and $X_{i^*} = Max(X_r, r = 1,\cdots,R)$ and $\triangle_{i^*} > 0, i^* \neq I$, then jobs of part i^* with maximal processing number except i can be reduced. If i^* is not exist, let $i^* = i$.

Let

$$\triangle_i = \begin{cases} \triangle_i - X_i + X_{i^*}, & \triangle \geqslant X_i - X_{i^*} > 0 \\ 0, & X_i - X_{i^*} > \triangle_i \\ \triangle - 1, & X_i - X_{i^*} = 0 \end{cases} \tag{4}$$

(4) is used to estimate $\triangle_i$ and the smallest value of it is zero. (3) and (4) are used to compute the jobs′ number of part i, then the problem $1//\Sigma C_j$ can be solved. It is apparent that above bottleneck scheduling algorithm is convergent, because the jobs processed in bottleneck machine will be reduced with appropriate step till due date satisfies the delivery time.

3.4 Decomposition and coordination of PULL system and PUSH system

Let $Job_{r,m}$ be the job m of part $r, d_{r,m,j}$ the due date of $Job_{r,m}$ on machine $j, Rl_{r,m,j}$ the release time of $Job_{r,m}$ on machine j, and $C_{r,m,j}$ the completion time of $Job_{r,m}$ on machine j. After the production of bottleneck K is scheduled, we could get the release time $Rl_{r,m,K}$ and the completion time $C_{r,m,j}$ of $Job_{r,m}$. These data could be used to build up the constraints of PULL system and PUSH system.

In PULL system, the succeeding machine control production of whole shop, so we can construct scheduling problem $1//L_{Max}$ as the following.

$$\begin{aligned}
&\min\ (L_{Max})\\
&\text{s.t. } d_{r,m,j} = Rl_{r,m,j+1},\ r = 1,\cdots,R,\ m = 1,\cdots,X_r\\
&\quad L_{Max} = Max((d_{r,m,j} - C_{r,m,j}),\ r = 1,\cdots,R,\ m = 1,\cdots,X_r)
\end{aligned} \tag{5}$$

Let $J = K-1,\cdots,1$. We can construct $1//L_{Max}$ successively, and solve them by EDD rule. So all production schedules in PULL system could be obtained.

In PUSH system, we build the single machine scheduling problem $1/r_j/\Sigma C_j$ of machine j as follows

$$\begin{aligned}
&\min\ \Big(\sum_{r=1}^{R}\sum_{m=1}^{X_i} C_{r,m,j}\Big)\\
&\text{s.t. } RL_{r,m,j} = C_{r,m,j-1}
\end{aligned} \tag{6}$$

Let $j = K+1$ to J. We can construct the single machine scheduling problems of PUSH system. Since this problem is NP-hard, we use simple dispatch rule FIFO to schedule the production. Other advanced algorithms can also be found in [2].

3.5 Coordination

In sections 3.3 and 3.4, we schedule the production of bottleneck at first, then build single machine scheduling problems and solve them in PULL system and PUSH system according to the schedule in bottleneck. Since the number of each part is estimated, and only makespan in bottleneck satisfies the time constraint, $ST_{1,k} = 0$ and $CMax_K \leqslant T_{end}$, so it is necessary to coordinate the schedule of flow shop to avoid its infeasibility.

We fix the processing sequence as in the current schedule, and set the start processing time of the first machine $ST_{1,1} = 0$, then modify $ST_{i,j}$ and $ET_{i,j}$ of all jobs accordingly. If $CMax_J > T_{end}$, the products of all parts will be modified according to (3), (4), and the scheduling problems (2,), (5), (6) should be solved repeatedly. The above procedure will be done iteratively till all the constraints in problem (1) are satisfied.

4 A simple example and analysis

The above decomposition and coordination algorithm was tested in a real project of hybrid flow shop for a company. The achieved schedule has good performance with low computational burden and result is satisfactory. In order to show the algorithm′s performance, we give a simple FSPT problem with 10 machines and 9 kinds of parts here. In simulation, different schedules are obtained with choosing different machine as bottleneck. The processing time for different part $P_{r,j}$ is show in table 1, and the start processing time of the flow shop is zero, and the delivery time T_{end} is 1800.

Table 1 Processing time $P_{r,j}$ of part r on machines j

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
Part 1	10	6	2	15	6	5	5	19	13	16
Part 2	2	7	17	3	5	21	12	2	17	18
Part 3	3	4	18	10	16	15	16	4	6	2
Part 4	11	8	3	18	3	3	9	12	4	7
Part 5	3	10	11	11	16	21	8	2	3	17
Part 6	2	11	8	3	20	15	8	5	11	10
Part 7	14	13	12	10	4	15	16	19	19	3
Part 8	20	6	8	18	13	8	18	13	21	18
Part 9	11	8	17	14	19	12	19	3	14	5

In Fig. 2, the simulation result is shown when different machine is chosen as bottleneck. The y-coordinate shows the throughput of the flow shop while x-coordinates is the index of the bottleneck machine. Comparing the throughput for different bottlenecks, the throughput for M4 is 91, and M5 is 109. The maximal difference of throughput for choosing different bottleneck is 18%, which shows the importance of choosing bottleneck in this algorithm.

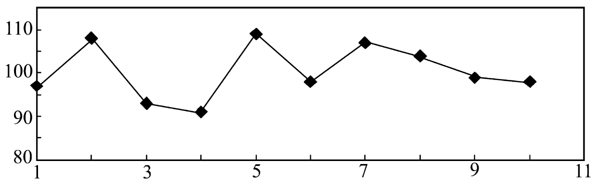

Fig. 2 Different throughput for different bottleneck choice

This algorithm decomposes NP-hard problem into a serial of single machine scheduling problems. Although it is an approximation method for solving the original problem, the computational complexity could be reduced mostly. If there are N kinds of parts and M machines in FSPT, the complexity of original problem is $(N!)^M$, while the complexity of this algorithm based on TOC is about $(N!)^*M$. The time expense for above example is only 210 ms.

5 Conclusion

In this paper, a decomposition and coordination algorithm for FSPT is proposed. The algorithm based on TOC is practical due to its less computational complexity and acceptable precision. For applying it to other FSPT there are still some important aspects to be investigated, such as reentrance, more bottlenecks and mixed discrete and batching machines.

References

1 Wang Li, Zhu Jing. The status and advances of flexible flow shop scheduling problem, *Journal of Anshan Normal University*, 2002, **4**(1): 9∼13

2 David Karger, Cliff Stein, Joel Wein. Scheduling Algorithms, Anshan: 3rd edition, Springer Verlag, 2001, 1∼9

3 Verma R. Management science, the theory of constraints/optimized production technology and local optimization. Omega. *International Journal of Management Science*, 1997, **25**(2): 189∼200

4 Lee L C. A comparative study of the PUSH and PULL productions systems, *International Journal of Operations and Production Management*, 1989, **9**(4): 5∼18

5 Nureddin Kyrkavak, Cemal Dincfler. The general behavior of pull production systems: The allocation problems, *European Journal of Operational Research*, 1999, **119**: 479∼494

6 Chen Hong, Madelbaum Avi. Discrete flow networks: Bottleneck analysis and fluid approximations, *Mathematics of Operation Research*, 1991, **16**(2): 408∼447

ZHANG Hong-Yuan Ph. D. candidate at Shanghai Jiaotong University. His research interests include complicated system schedluling and radio network base station planning.

XI Yu-Geng Professor of Shanghai Jiaotong University. Received his Ph. D. degree from Technical University of Munich, Germany in 1984. His research interests include predictive control, large-scale system, and intelligent robotics.

GU Han-Yu Associate professor of Shaihai Jiaotong University. His research interests include complex aystem modeling and optimization.