Applied Mathematics & Information Sciences
An International Journal

Research on Scheduling of Jobs in Plane Manufacturing

*Yanqing Qiu, Peng Ge**, *Peiyu Ren, Maozhu Jin, Yuyan Luo, Huafeng Gao*

Business School, Sichuan University, Chengdu 610065, China

Received: Nov. 10, 2011; Revised Jan. 04, 2012; Accepted Feb. 13, 2012

Abstract: This paper proposed a heuristic algorithm based on due-dates for a stretch machine scheduling problem in the plane manufacturing industry. An integer-Programming model incorporated the due-dates of jobs,capacity of oven, availability of stretch machines, and setting up times is built. The objective of this problem was to minimize the make-span of all jobs on stretch machines by scheduling. To guarantee the due-dates of different orders, maximizing the efficiency of bottleneck resource (i.e., stretch machine in this paper) must be considered. Because of the computational complexity of this problem, this paper developed a heuristic approach. Computational experiences indicated that the heuristic gave high quality solutions with significant savings in time over standard IP algorithms.

Keywords: Parallel Stretch Machines, Load Balancing, Integer Programming, Heuristics.

1. Introduction

Several of plates with different materials, different shapes, and different thicknesses have been used in the plane manufacturing industry. Among theoretical approaches to this problem, mathematical programming branch-and- bound and heuristic approaches appear to be most common. But the computation time always grows exponentially with increasing the problem size. Heuristics developed are limited largely to uniform parallel machine problems. Various literatures have shown that scheduling is a fundamental issue in achieving high performance on computers and computational grids [1–3]. Some researchers studied the approach to solve these complex problems. V. Srikrai et al. (2006) proposed a MOD-DBR model for a non-identical parallel machine flow shop [4]. Non-identical parallel machine flow shop is only one of the key problems of this system. Besides, the arrival times of jobs are stochastic, L. Baccouche et. al presented a RT-DBP method to assign the priority for real-time tasks scheduling, and this makes our schedule more difficult. We assume a given set of jobs to be scheduled in our system.

Development of a computer system for the scheduling of thousands of parts for airplane manufacturing is a relatively complex and important problem for managers. They plan to integrate the logistics both inside and outside of their company. Throughout the first few months of this effort a number of departments were identified as being fraught with danger with regard to the ultimate success of the project. Because of these problem areas, it was decided that an evolutionary approach to system design and development would increase the probability of success. This approach was to rely heavily on interaction between the potential users of the system, at both user and management levels, and the developers of the system. Following the development of a framework for system design and implementation the actual work on the system itself was begun. Managers find out some key departments which are the bottleneck of the whole manufacturing process and hope that we could do some optimization work to increase the throughput of them.

What we really consider is a partly backtrack system based on multiple-cluster with time windows. Shijie Sun et al. (1996) [5] took a research on batching scheduling problem of flow shop with the objective of minimize C_{max}. Weizhe Zhang et. al (2008) [6] introduced a time-varying performance model for multiple cluster jobs, and a multiple cluster scheduling scheme and four novel time-varying scheduling algorithms. But in the real world, many such sub-systems are combined in the system we considered. Arman R. Yaghubian etc. compared the results of heuristic with standard IP algorithm for a dry kiln scheduling problem in a furniture manufacturing industry [7].

It has been realized that scheduling is a fundamental issue in achieving high performance on stretch machines and oven. In plane manufacturing environments, there are many departments of schedulers to meet different performance goals. Resource schedulers coordinate user requests

* Corresponding author: e-mail: qyq_101@163.com

for accessing a given resource to ensure fairness and to optimize utilization (Chapter 13 of [8]). Application schedulers promote the performance of individual applications by optimizing performance measures such as execution time and speedup (Chapter 12 of [8]). Job schedulers aim to optimize the overall performance of a system, e.g., minimizing the average job response time and maximizing the number of jobs executed in certain period of time. In this paper, we are interested in the problem of job scheduling on a three-stage process, in which the first and the third stage are processed on the same machine.Describing the criterion by rephrasing, the users made it obvious that they envisioned it as one that could be evaluated subjectively. At the same time, the authors recognized that eventually it had to be converted to one that could be objectively evaluated by the computer. The final resolution of this problem will be described in the third article of this series.

The rest of the paper is organized as follows. In Section 2 we represent our system. In Section 3 we present the mathematical model and our heuristic algorithm. We compared the results of the mathematical model and our heuristic algorithm in Section 4. We conclude our paper and suggest some possible future directions in Section 5.

2. System

There are two stretch machines and two ovens in the system of this paper. According to the Theory of Constraint, the bottleneck-stretch machine-restricts the throughput of whole system. Then, the first step of our work is to solve the problem of parallel machines scheduling. One way to solve this problem is to use artificial intelligent based on computer (for example, genetic algorithm (GA), greedy algorithm, tabu search, simulated annealing, neural network, etc.) [9], develop a reasonable production schedule with associated setting and processing time, and then, use the original schedule as a starting feasible point. This process can be repeated until the user has obtained the best job queue. A drawback of this procedure, especially for large problems, is the difficulty in deciding which jobs to be scheduled conterminous. This approach turns out to be just a simple trial-and-error procedure. Yan Zuo et. al. (2007) reduced the problem with an efficient heuristic, with an idea that non- constraint machines wont impact the throughput [10]. Another approach would be to automate the time analysis by developing an integer programming model of the sort problem based on TOC with time windows. In this study, the objective is to maximize the earliest process time of jobs on stretch machine while meeting all due-dates.

First of all, we need to do some pre-stretch to the plate, and after it obtains a certain size, a heat treatment in an oven is needed. Then, some of the jobs must be re-stretched to ensure the changelessness of its shape. The flow process can be figured as Figure 1.

Due to the constraint of rules of the ovens packing, we must sort parts by the principle that parts with the same

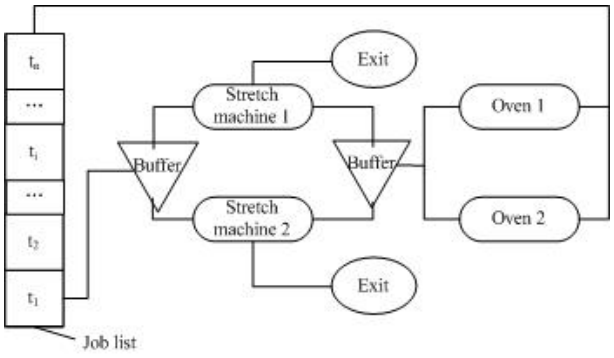

Figure 1 Flow chart of production

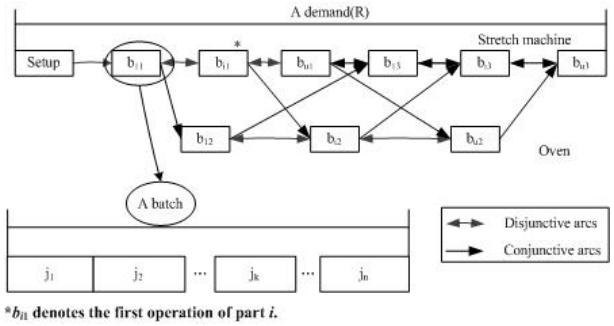

Figure 2 Relationship of demand, batch and part

material, approximate thickness, and the same heat treatment time be sorted close, and we call these homogeneous parts a batch. The relationship of demand, batch and part is described in Figure 2: There are many demands in the system, and each demand can be divided into several batches according to the capacity of oven. Now, the whole flow of process can be described as two sections as follows:

Stretch: operations pre-stretch and re-stretch use the same parallel stretch machines. According to the definition of bottleneck [11], its a machine whose available capacity is lower or equal to its demand, and it apparently constraints the throughput of the overall system. After analyzing the productive process of the real shop for each job, the process time of heat treatment is much longer than stretch, but oven can hold several plates in it once, while stretch machine only one at a time. Therefore, if the division is appropriate, the machine which restricts the system throughput is stretch machine, other than oven. In view of the standardization of aircraft manufacturing, jobs with the same shape must use the same die while stretching. When processing jobs with different shape, the die must be changed, this will be time consumption (often doubled the processing time). Since the setup time of replacing dies for stretch machine is generally much longer than the process time, jobs with the same shape should be sorted coterminous based on their due-dates.

Heat treatment: because the process time for plates with different materials, thickness and characteristics in oven is quite different, it should be sorted properly to improve the utilization of oven, so that it wont become bottleneck instead of stretch machine. We call jobs with the same material, thickness and characteristic (i.e. can be heated in oven at the same time) homogeneous jobs. We strive to make these homogeneous jobs arrive the operation of oven simultaneously or in succession. Then, our task is to arrange these jobs in an oven with constraint of its capacity. Packing jobs in an oven is a problem of three-dimension (3D) knapsack. This is strongly NP-hard, we will simplify it later.

Generally, each of those three operations has more than one selection of machines. This scheduling problem can be treated as a special case of scheduling n jobs on m parallel machines in a three operations system, which has been widely investigated, and shown to be NP-hard [12]. Assume that parts with the same due-date are finite, and the parts cant be overlapped in the oven. This is a rectangular packing problem by calculating the surrounding boxes of irregular parts. But if the deformation range of the job is properly large, this method is also hard to avoid taking up much space, and it loses its expected effect. We established a group strategy in the same oven.

3. Mathematical and heuristic models

3.1. Mathematical model

To guarantee the due-dates of different orders, maximizing the efficiency of bottleneck resource (i.e., stretch machine in this paper) must be considered. Then, the problem is translated into an analysis on minimizing the makespan of all jobs on stretch machine by scheduling. Assume that: (i) a stretch machine can process only one plate at a time; (ii) raw plate is available in sufficient quantity during scheduling; (iii) no preemption of jobs is allowed; (iv) machines are available after the given availability time; (v) jobs in the same demand use the same machine when they face with a choice; (vi) jobs in a batch be processed continuously and share the same process time; (vii) there is no-preemption once a batch begin to be processed. Assumption (i) is not limiting because of the limitation of the stretch machine itself. Assumption (ii) is not limiting because it is not the factor which will affect our solution. Assumption (iii) is not limiting since preempting a job would increase the setup time of each stretch machine, and may possibly cause the waste of oven space. Since in real practices, demands are divided by the material/thickness of jobs, assumption (v) does not limit the usefulness of the model. Assumption (vi) and (vii) is a constraint prescribed by the real implement.

Parameters:

n=total number of plates which are needed to be processed;

m_k=total number of machines in process k;

S_{rik}=setup time of demand r on machine i in process k;

$R = \{R_1, R_2, , R_v\}$total demands;

$R_v = \{b_1, b_2, , b_u\}$demand v can be divided into u batches;

l_b=total number of jobs in batch b;

p_{ijk}=processing time of job j on machine i in process k;

q=capacity of oven;

bt_{bik}=begin time of batch b on machine i in process k;

J_k=set of machines in process k.

Decision variables:

$$x_{bik} = \begin{cases} 1, batch\ \text{b}\ is\ processed\ on\ machine\ \text{i}\ in \\ \quad process\ \text{k} \\ 0, otherwise(b = 1, ..., \lceil n/l_b \rceil; i = 1, ..., m_k; \\ \quad k = 1, 2, 3) \end{cases}$$

Jobs are numbered in increasing order of due-dates. Our object is to minimize the maximum completion time of all jobs. The integer linear program is stated as follows:

Min C_{max}

subject to:

$$l_b \le q(b = 1, ..., \lceil n/l_b \rceil) \tag{1}$$

$$\sum_b l_b = n(b = 1, ..., \lceil n/l_b \rceil) \tag{2}$$

$$x_{bik} bt_{bik} + x_{bik} \sum_{j=1}^{l_b} p_i jk \le x_{bi(k+1)} bt_{bi(k+1)} \quad (b = 1, ..., \lceil n/l_b \rceil; i = 1, ..., m_k; k = 1, 2, 3) \tag{3}$$

$$x_{bik} bt_{bik} + x_{bik} \sum_{j=1}^{l_b} p_i jk \le x_{(b+1)ik} bt_{(b+1)ik} \quad (b = 1, ..., \lceil n/l_b \rceil; i = 1, ..., m_k; k = 1, 2, 3) \tag{4}$$

$$x_{b_u ik} bt_{b_u ik} + x_{b_u ik} \sum_{j=1}^{l_b} p_i jk \le x_{b_1 ik} bt_{b_1 ik} \quad (b_u \in R_{v1}; b_1 \in R_{v2}; v1 \ne v2; i = 1, ..., m_k; k = 1, 2, 3) \tag{5}$$

$$\sum_{b=1}^{\lceil n/l_b \rceil} x_{bik} (S_{rik} + \sum_{j=1}^{l_b} p_i jk \le DDT \quad (r \in R; i = 1, ..., m_k; k = 1, 2, 3) \tag{6}$$

$$J_1 = J_3 \tag{7}$$

Where $\lceil x \rceil$ takes upper integer of x.

The objective function represents minimizing the maximum completion time of the total jobs. Constraint set Eq. 1 defines the number of jobs in each batch is set to be less than or equal to the capacity of oven. Constraint set Eq. 2 specifies that the number of jobs should be equal to the total job numbers that are needed to be processed. Constraint set Eq. 3 denotes that the completion time of each operation on a batch is less than or equal to the begin time of

its next operation. Constraint Eq. 4 indicates that the completion time of a batch in an operation is less than or equal to the begin time of the next batch in the same operation. Constraint Eq. 5 indicates that demands allocated on the same machine cant share the same time space. Constraint Eq. 6 indicates the total time of batches in the system must satisfy their due-dates. Constraint Eq. 7 points out that the operations pre-stretch and re-stretch share the same machines.

Given that $Q_2||C_{max}$ has been shown to be NP hard [13], while the operations pre-stretch and re-stretch share the same stretch machine, this adds the complexity of solving the mathematical model. The main disadvantage of this model is that we cant consider the constraint of the batch according to the real situation.

3.2. Heuristic model

It is difficult to obtain the optimal solutions, and there is no need to find the optimal solutions in the real practice. Actually, we only need to gain the heuristic or approximate solutions in real practice, and we considered a heuristic model in Figure 3. Many researchers have proposed some heuristic algorithms such as LPT (Longest Processing Times) and Multifit algorithm [14]. Graham have testified the solution of LPT order do not exceed $4/3-1/(3m)$ times of the optimization literature, and this bound cannot be improved. Coffman et al. have testified the solution of Multi-fit algorithm is less than or equal to 1.22.

The jobs can be divided into two types: (1) jobs that should be processed on a certain machine, we call them 'A_1'; (2) jobs that can share these machines, we call them 'B_1'. Jobs with different types have different weights:

$$w_{ji} = \begin{cases} 0, \text{ job j can not be processed on machine i;} \\ 1, \text{ job j can be processed on machine i;} \\ 2, \text{ process job j on machine i with priority.} \end{cases}$$

The algorithm first allocates A_1, and then, B_1 with priority of their weights, and finally, using LPT rule to balance their loads. Thus, we find a quick and effective (but not always excellent) original solution.

In the second stage, we use SPT rule to sort jobs on a certain machine. Then, check if the completion time of each job can meet its due-date; if so, go to the next stage, otherwise, go forward to stage 4 for reorder.

In the third stage, judge whether the conjoint jobs can be processed in the same oven, if not, go ahead to stage 4 for reorder; otherwise, lot sizing jobs into batches with the group strategy (GS), where:

$G = (g_1, , g_v, , g_V)$ denotes the set of group g;

$g =< s, R, p >$ denotes the group strategy in an oven;

s denotes the sequence number of ovens;

$$R_g = \frac{\sum_{i=1}^{N} \sum_{j=i+1}^{N} r_{ij}}{C_N^2}$$

($r_{ij} = \begin{cases} 1, \text{ i and j are homogeneous;} \\ 0, \text{ otherwise.} \end{cases}$ N=number of jobs in g), denotes correlation degree of jobs in g;

p=process time of group p on oven s.

We divide jobs into two types: A_2 and B_2 as well as in stage 1. Take notice, jobs in the same batch must keep SPT order.

In the fourth stage, the algorithm SHAKES the solution. First, it finds out the bottleneck machine through utilization (bottleneck machine holds the biggest utilization), then, checks for idle times between conjoint demands on this machine. These idle times are usually caused by operation of heat treatment. Thus, we can remove the demand which needs to wait on bottleneck machine, and search for continuous time gaps on the oven processing this demand backward, and the time gap must be greater than or equal to the process time of this demand. If such time gap exists, select a demand with random and insert it into here, then return to stage 2; otherwise, switch this demand with another demand processed on the other oven that can serve for this demand also, and return to stage 1; if both of these two exchanges cannot active or no time gap exists on the bottleneck machine, move on to the next step. It tries to improve the solution (i.e., the algorithm performs a More Local search) by: (1-1) changing the order of the jobs processed on a certain machine randomly; (1-2) re-allocate jobs among the uniform machines with its weights being altered between 1 and 2. In the second routine, the jobs which must be processed on a certain machine are not included (i.e., random switching routines work only within the $B_k(k = 1, 2)$ jobs). We explain these two rules of exchange in Figure 4. When a switching rule is applied, it is tested by the stages above until a successful switch can be made.

4. Result

The algorithms performance was compared with the LINGO Integer-Programming (IP) package. Generally speaking, there are three methods to evaluate the solutions we achieved: (a). compare the quantity of solutions, we can evaluate both how close our solution is to the optimal solution and how large the variation of our solution is. (b). compare the solution time including both the time consumed and stability during calculation. (c). use both (a) and (b). Due to the small scale of the examples above, both methods can generate final object value quickly. We consider solutions under different job numbers, and compared the effectiveness of the two solutions.

Considering the complexity of calculation of the IP model, we simplify the problem in the real world in this experiment: 1.assume that every job must go through three operations as following: pre-stretch, heat-treatment, and re-stretch, and this help us to transform the problem from job shop to permutation flow shop. Distinguishingly, when the effective process of a certain job dont have one or more

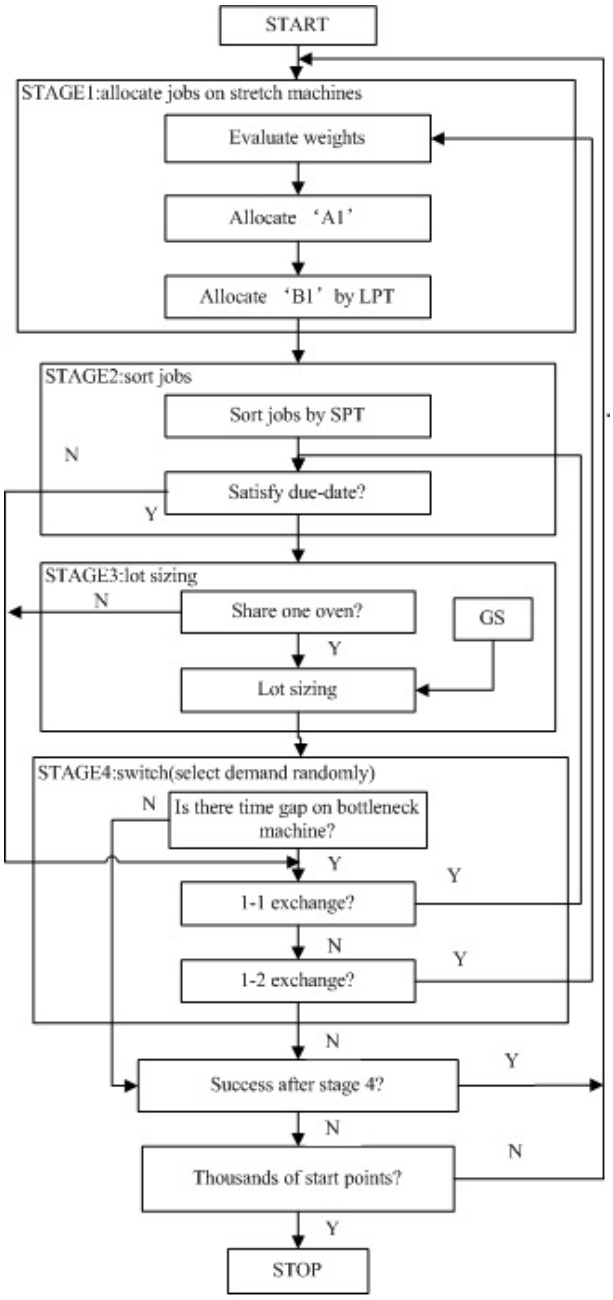

Figure 3 Flow chart of the algorithm

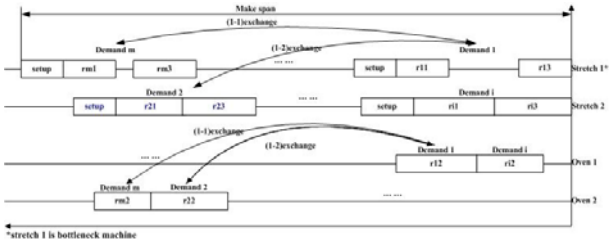

Figure 4 Switch of stage 4

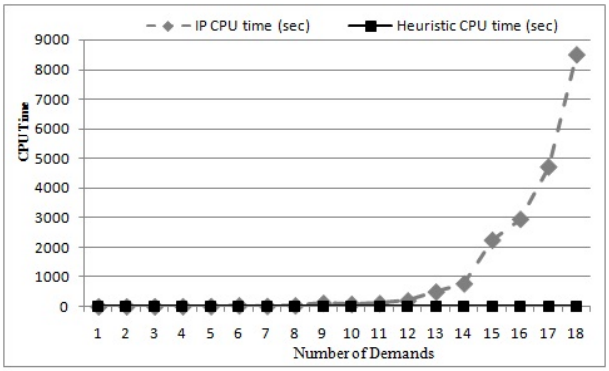

Figure 5 Time performance between the IP and the heuristic procedures

operations, we can let its process time be zero in this process; 2.there is no constraint of time windows (i.e. any machine is working for 24 hours every day); 3.each operation has two parallel machines to provide services. To make the experiment comparable, we bring these three assumptions into both the LINGO procedure and our heuristic algorithm.

Eighteen problems (up to 344 jobs) were randomly generated in the first stage of the experiments. After the analysis of GS, each demand contains two batches, and the size of each batch can vary between 0 and 15, while oven capacities (which determined the size of our batches) typically range from 4 to 20. Processing times (p_{ijk}) of ovens can vary from 20 minutes to 120 minutes. Results for both the heuristic and IP procedures are presented in Table 1. The first three columns of Table 1 contain the inputs of our problem. NJ means the Number of Jobs, ND means the Number of Demands, and NV/NC means the Number of Variables or the Number of Constraints. The last four columns show us the performance of both IP and Heuristic methods, in which IPT and HT means the CPU Time cost by method of IP and Heuristic respectively, IPC and HC means the completion time of solutions under IP and Heuristic algorithm. When NJ exceed 150, IPT experiencing a certain fluctuations and with the NJ continually increased, the time consumed by IP quickly increased and Soon reach a larger value. Take the last two cases for example, NJ increase 11% and IPT increase 80%, and reached 2.35 hour; HT increase 46%, and its computation time is 19 second. The solution of our heuristic algorithm can find the optimal solution mostly while NJ less than or equal to 200.

Figure 5 shows us the time performance between the IP and the heuristic procedures. As it turns out, the time consumed by IP model increasing exponential and show a great range of fluctuation as the number of demands in-

Table 1 A Comparison between the heuristic and IP solutions

NJ	ND	NV/NC	IPT(sec)	HT(sec)	IPC(min)	HC(min)
39	2	36/37	0	0	520	520
59	3	54/58	0	0	900	900
79	4	72/81	2	0	980	980
114	6	108/133	3	0	1360	1360
120	7	126/162	9	0	1560	1560
156	9	162/226	34	1	1960	1961
169	10	180/261	15	1	2160	2160
183	11	198/298	33	1	2300	2300
199	12	216/337	114	2	2500	2680
214	13	234/378	83	1	2740	2881
227	14	252/421	107	3	2980	3001
241	15	270/466	198	3	3140	3280
251	16	288/513	495	8	3300	3520
263	18	324/613	773	6	3620	3870
276	19	342/666	2235	10	3840	4200
291	20	360/721	2940	9	4000	4250
310	22	396/837	4700	13	4260	4538
344	25	450/1026	8470	19	4640	5221

creased (Figure 5). This will be time consuming in the actual implementation. The heuristic found good solutions for the test problems within one second. This proves that our algorithm is feasible, especially in the case of large size of demands.We compute the cases in which ND=40, ND=75, and ND=100. The IP takes 72 hours and find relatively good solutions. The best solutions found by heuristic algorithm are larger than IP, but the time consumed are less than or equal to 35 seconds. In real practice, we prefer an approximate solution in a relatively short time, rather than spending several days to achieve a relatively accurate result.

For problems larger than those detailed in Table 1, the computational results for three larger problems are shown in Table 2. The mean and standard deviation of the objective time extents is reported.

Table 2 Heuristic versus IP for large problems.

Technique	Problem instance	40 demands	75 demands	100 demands
IP	Object bound (min)	6445	14885	12470
	Best solution found (min)	7185	21870	17370
	Time (hour)	72	72	72
Heuristic	Best solution found (min)	7515	15006	18958
	Average solution	7677.6	15535.4	19285.8
	std. dev.	75.2	523.6	762.5
	Time (sec)	30	35	35

*** The configuration of computer is: Pentium IV CPU 2.66GHZ, RAM 1G.**

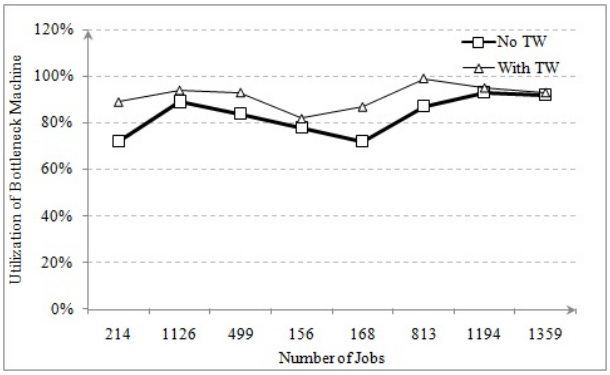

Figure 6 Utilization of bottleneck machine

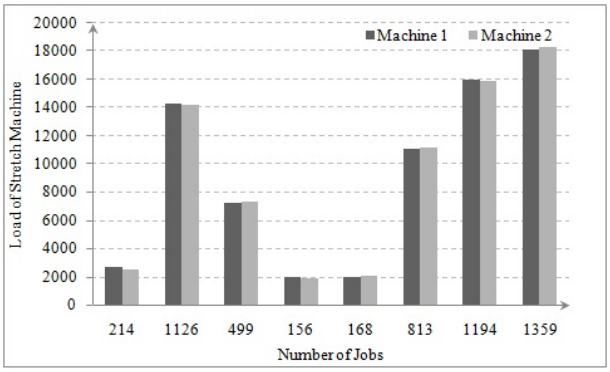

Figure 7 Comparison of loads of the two stretch machines

The next experiment helps to determine the stability of the heuristic over calculating the deviation of utilization of bottle-neck machine and load equilibrium of each stretch machine after time windows being absorbed into it based on a number of different cases. And show the analytical results in Figure 6.

As seen in Figure 7, we can calculate the standard deviation rate (CV) of the loads of two parallel machines from the values in the last column as follows: $CV = \sigma/\bar{x} = 30.59/127.5 = 0.24$. The heuristic shows that the utilization of the bottleneck machine holds a steady high level without time windows (with average value: 83%), and the loads of the parallel machines are also harmonious under a random sample of job numbers.

5. Conclusions

A four-stage heuristic algorism based on exchanging jobs in and between the parallel machine(s) was compared with the integer linear programming model for the scheduling of jobs on parallel stretch machinees has been developed. Due-dates for each job, oven capacity, stretch machine availability, and setting times are incorporated into the model.

Owing to the complexity of the problem, the development of a heuristic approach was motivated. Optimal solutions to a wide range of problems were compared with solutions obtained by the heuristic. Comparison indicates that the heuristic offers not only a tremendous advantage in speed, but also provides high quality solutions. Additionally, we add time windows in our heuristic algorithm. These discontinuous time spaces will affect the practicable process time of the oven, and then, react on the bottle-neck machine ulteriorly. In our heuristic algorithm, they were treated as patches with settled beginning times and completion times which were not processed on any machine. As we have seen in Figure 6 and Figure 7, this increases the utilization of bottleneck machine (with average value: 92%), but it makes the entire make span much longer than the perfect state. We exchange the order of jobs in the forth stage of our algorithm to ease this problem, and approach to our objective as much as possible. We allow users to modify any information in this system when necessary also, and this will help to meet the diverse needs of the enterprise.

Acknowledgement

This work was supported by the Major International Joint Research Program of the National Natural Science Foundation of China (Grant No. 71020107027), the National High Technology Research and Development Major Program of China (863 Program)(Grant No. 2008AA04A107), the National Natural Science Foundation of China (Grant No. 71001075), 985 and 211 project of Sichuan University.

References

[1] Goldratt, E.M. and Cox, J.,The Goal: A Process of Ongoing Improvement, North River Press: Croton-on-Hudson, NY (1984).

[2] I. Foster and C. Kesselman, The Grid: Blueprint for a New Computing Infrastructure (Morgan Kaufmann, San Francisco, CA)11, 115-128 (1997).

[3] J. Gehring and T. Prei?. Proceedings of the 5th Workshop on Job Scheduling Strategies for Parallel Processing, LNCS, 5, 1-16 (1999).

[4] V. Di Martino and M. Mililotti. Proceedings of the 3rd International Workshop on Parallel and Distributed Scieti?c and Engineering Computing with Applications, Fort Lauderdale, Florida, 2, 1812-1822 (2002).

[5] V. Sirikrai, P.Yenradee, INT J PROD RES, 44, 3509-3531 (2006).

[6] Shijie Sun, Qi Xie. Journal of Shanghai University, (Natural Science Edition, 5, 473-478 (1996).

[7] Weizhe Zhang, Hongli Zhang, Xinran Liu, Xuemai Gu. The 9th International Conf. for Young Computer Scientists, 5, 18-21 (2008).

[8] I. Foster and C. Kesselman, The Grid: Blueprint for a New Computing Infrastructure, Morgan Kaufmann, San Francisco, CA, (1999).

[9] Gupta,J. N. D. Heuristic Algorithms for Multistage Flow shop Scheduling Problem , AIIE Transactions, 4, 11-18 (1972).

[10] Yan Zuo, Hanyu Gu, Yugeng Xi. Study on Constraint Scheduling Algorithm for Job Shop Problems with Multiple Constraint Machines, International Journal of Production Research, 6, 1-17 (2007).

[11] Arman R. Yaghubian, Thom J. Hodgson and Jeffrey A. Joines, IIE Trans, 33, 131-136 (2001).

[12] Yu,L., Shih, H.M., Pfund, M., Carlyle, W.M. and Powler, J.W., IIE Trans, 34, 921-931 (2002).

[13] Lawler, E.L., J.K. Lemstra. A.H.G. Rinnooy Kan and D.B. Shmoys. Sequencing and scheduling: Algorithms and complexity. in: Graves, S.C. etal. Handbooks in OR. & MS. Vol 4 (Elsevier Science Publishers B.V., Amsterdam), 5, 445-522 (1993).

[14] Graham R L. SIAM J, Appl. Math, 17, 263-269 (1969).