BIOINFO Computer Engineering
ISSN: 2249-3980 & E-ISSN: 2249-3999, Volume 2, Issue 1, 2012, pp.-22-24.
Available online at http://www.bioinfo.in/contents.php?id=322

PRIORITY BASED FLEXIBLE JOB-SHOP SCHEDULING PROBLEM

RAICH A.R.* AND NIKUMBH P.J.

Ramrao Adik Institute of Technology, Nerul, Navi Mumbai, India.
*Corresponding Author: Email- anagha.koranne@gmail.com

Received: February 28, 2012; Accepted: March 06, 2012

Abstract- Priority base flexible Job-shop Scheduling Problem (P-FJSP) is one of extremely hard problems because it requires very large combinatorial search space, as it is classified as NP-Hard problem. The main objective of the FJSP is to find a schedule of operations, to minimize the maximum completion time. To obtain an optimal or a near the optimal solution, we have proposed a method for solving job-shop scheduling problem using Simulated Annealing (SA).
Keywords- Flexible Job-Shop Scheduling, Simulated Annealing, Minimize makespan, Minimize Workload.

Citation: Raich A.R. and Nikumbh P.J. (2012) Priority Based Flexible Job-Shop Scheduling Problem. BIOINFO Computer Engineering, ISSN: 2249-3980 & E-ISSN: 2249-3999, Volume 2, Issue 1, pp.-22-24.

Introduction

The Job-Shop Scheduling Problem (JSSP) is one of the most difficult problems, as it is classified as NP-Hard problem. The main objective of the FJSP is to find a schedule of operations, to minimize the maximum completion time. This is termed as makespan that is the time required to schedule all the operations for 'n' jobs on 'm' machines. In many cases, the combination of goals and resources exponentially increases the search space, and thus the generation of consistently good scheduling is particularly difficult, because we have a very large combinatorial search space and precedence constraints between operations. In order to overcome this difficulty, it is more sensible to obtain an optimal or a near the optimal solution. The P-FJSP is a generalization of the classical FJSP. And it is more difficult than the classical JSP, because each job consists of a set of operations which must be processed in a right order that depends on the job. So, it introduces a further decision level besides the sequencing one, i.e., the job routes. Stochastic search techniques such as evolutionary algorithms such as genetic algorithm [2] [3], Simulated Annealing (SA) [2] and tabu[10] search can be used to find an optimal solution.

In this paper we have proposed a method for solving job-shop scheduling problem using SA. Here we assign priority to each sequence by calculating ration of job[11]. In Section 2, the flexible job shop scheduling problem is presented & priority assign method is describe . In Section 3, a Simulated Annealing is discussed. An experiment is given and the results are reported is Section 4. In Section 5, there is a conclusion. In Section 6, there are references mentioned in this paper.

The FJSP

FJSP can be described as follows:

1. A set of jobs, J = {J_1, J_2... J_n}. The number of jobs in this set is 'n'.
2. A set of machines, M = {M_1, M_2... M_m}.The number of machines in this set is 'm'.
3. Every job consists of a sequence of operations. Ji = {O_{i1}, O_{i2},, O_{ini} }. O_{ij} means that this operation is the j^{th} operation in J_i. The number of operations in J_i is ni.
4. Every operation O_{ij} can be processed in a machine subset of M. We called M_{ij} is the machine set of operation O_{ij}, which means that all the machines in M_{ij} can handle O_{ij}.
5. If O_{ij} can be processed on machine M_k, its processing time is P_{ijk} (P_{ijk} >0). If O_{ij} cannot be processed on machine M_k, then P_{ijk}=0[1][3].

The object of scheduling algorithm is to assign each operation to a right machine. In other words, its object is to get a sequence of the operations on machines to minimize the makespan, i.e., the total processing time of all the jobs. In solving this scheduling problem, some constraints should be kept. As follows: Job preemption is not allowed. Every machine can process only one operation at a time. All the jobs can be available at time 0. The transportation time is ignored, i.e. after a job is processed on a machine, it is immediately transported to the next machine. Setup time for all operations on machines is ignored too.

To demonstrate FJSP model clearly, we first prepare a simple example. Table 1 gives the data set of an FJSP including 3 jobs operated on 4 machines. It is obviously a problem with *total flexibility* because all the machines are available for each operation ($U_{ik}=U$). There are several traditional heuristic methods that can be used to make a feasible schedule. In this case, we use the SPT [1] (select the operation with the shortest processing time) as selective strategy to find an optimal solution.

1. Starting from a Table 1 presenting the processing times possibilities.
2. We assign O_{11} to M_1, and add the processing time P_{111}=1 to the elements of the first column of Table 1.
3. The objective function can be calculated as follows:

i. t_M - Gives the first objective makespan and also means to minimize the maximum finishing time considering all the operations.

ii. W_M - Gives the second objective which is to minimize the maximum of workloads for all machines.

iii. W_T - Gives the objective total workloads

Table 1.3- Jobs and 4 machine data set

NO. OF JOBS	NO. OF OPERATIONS	NO. OF MACHINES			
		M1	M2	M3	M4
J_1	O_{11}	1	3	4	1
	O_{12}	3	8	2	1
	O_{13}	3	5	4	7
J_2	O_{21}	4	1	1	4
	O_{22}	2	3	9	3
	O_{23}	9	1	2	2
J_3	O_{31}	8	6	3	5
	O_{32}	4	5	8	1

$t_M = \max\{\max\{t^F_{ik}\}\}$

$W_M = \max\{W_j\}$

$W_T = \sum W_j$

$_{ratio} = \sum M_{i\,/}\,J_i$

Table 2- Assign O_{12} to M_3, P_{123}=1

NO. OF JOBS	NO. OF OPERATIONS	RATIO OF JOB			
		M1	M2	M3	M4
J_1	O_{11}	0.2	0.5	0.3	0.37
	O_{12}	0.29	0.43	0.21	0.5
	O_{13}	0.26	0.28	0.42	0.58
J_2	O_{21}	0.44	0.15	0.36	0.37
	O_{22}	0.55	0.31	0.42	0.41
	O_{23}	0.61	0.3	0.39	0.33
J_3	O_{31}	0.35	0.34	0.33	0.25
	O_{32}	0.11	0.15	0.24	0.04

t_M = max{1, 3, 1, 2, 2, 3, 4, 4} = 0.85

W_M = max{0.46, 0.46, 0.6,0.29} = 0.46

W_T = 0.46 + 0.46 + 0.6 + 0.29 = 1.81

Table 3- Data set of 3 - Jobs and 4 - Machines with T_M, W_M, W_T

NO. OF JOBS	NO. OF OPS	SEQUENCE OF OPERATION	T_M	W_M	W_T
J_1	O_{11}	S = 2 4 1 2 2 3 4 4	0.39	0.96	2.4
	O_{12}	S' = 1 3 1 2 2 3 4 4	0.85	0.46	1.81
	O_{13}	S' = 3 3 1 2 2 4 4 1	0.79	0.58	1.92
J_2	O_{21}	S' = 1 4 2 2 4 3 3 1	0.98	0.91	2.37
	O_{22}	S' = 1 3 2 3 2 4 4 1	1	0.59	2.05
	O_{23}	S' = 4 3 1 2 2 4 3 1	0.84	0.7	2.07
J_3	O_{31}	S' = 1 1 2 2 4 4 3 3	0.89	0.74	2.23
	O_{32}	S` = 1 3 2 4 2 3 4 1	1.07	0.6	1.86

Simulated Annealing

An SA[8] algorithm is an artificial intelligence technique based on the behavior of cooling metal. It can be used to find solutions to difficult or impossible combinatorial optimization problems. SA procedure is introduced as a model-free optimization for solving NP-Hard problems. SA is one of the earliest methods for derivative-free optimization such as Tabu Search (TS) (Kirkpatrick et al. (1983)). SA is a random search technique and draws its analogy from physical annealing of solids [6]. Annealing is the process of submitting a solid to high temperature, with subsequent cooling, so as to obtain high-quality crystals (i.e., crystals whose structure form perfect lattices). If the cooling is carried out rapidly, it results in non uniform hardness and various pockets of high and low hardness occur. This basic concept of annealing of solids is simulated to arrive at the near global optimum solution of the combinatorial optimization problems. SA is an improvement type of algorithm and therefore starts with an initial configuration and a value for the control parameter (T) which is analogous to the temperature in physical annealing.

1. Get an initial solution S.
2. Get an initial temperature T >0.
3. While not yet frozen do the following

i. Perform the following loop L times.

ii. Pick a random number S' of S.

iii. Let $\Delta = W_T(S') - W_T(S)$.

iv. If ($\Delta < 0$) (downhill move), set S = S'.

v. If ($\Delta>0$) (uphill move), set S = S' with Probability = exp (-Δ/T).

vi. Set T = r * T (reduce temperature).

4. Return S.

Problem Description Using Simulated Annealing

In this section we explain how simulated annealing is used for flexible job shop scheduling; for this we initially generate a random sequence called S and calculate other components as shown below in example

Notations

S: Initially randomly generated sequence

S': Randomly generated operation sequence in next step

T: Initial temperature

ΔS: Difference between two operation sequences

R: Reject which collect the number of sequence rejected so far (R>=3, then stop)

n: Number of steps

U: Randomly generated number

P: Probability

G: Best Sequence Encounter

$\Delta G: SU_{s`} - SU_G$.

Numerical example

Now we present a numerical example, illustrating the application of the proposed SA algorithm. The description of the problem is shown in Table 4. We find all possible job sequences and then apply the simulated annealing algorithm to calculate other components that is find out $\Delta S = S` - S$. if $\Delta S > 0$ then calculate probability P , if P<U then increase R by 1and if P > U then calculate T and increase n by 1 , if n >15 then STOP process.

1. SPT = {G} = {S} = {1 3 1 2 2 3 4 4}, $SU_G = SU_S = 1.81$
2. T=200, R=0, n=0
3. INSERT ({S},{S`}, SUs`), {s`} = {3 3 1 2 2 4 4 1} , SUs` = 1.92
4. DIFF ({S}, {S`}), ΔS = S`-S, = 1.92- 1.81 = 0.11
5. Since ΔS > 0, hence go to Step 9
6. Assign {S}<-{S`} $SU_S$<-$SU_{s`}$
7. Compute ΔG, ΔG= $SU_{S`}$- SU_G
8. P = exp (-(-0.11)/200) = 1.01, U = 0.71, Since P> U, Hence go to
9. Assign {S} <- {3 3 1 2 2 4 4 1}, SUs <- 13 go to Step 13
10. R<-R+1
11. if R>3 then STOP
12. Change value of T, n = n+1 = 0+1 = 1

If (n>15)
STOP
Else
$T = T_0/(1+\log n)$
Result for different job sequences generated shown in Table 4.

Table 4- Optimum Result

SN	SEQUENCE OF OPERATION	W_T	ΔS = S`-S	P	U	N	R	T (°C)
1	S = 2 4 1 2 2 3 4 4	2.4	-	-	-	-	-	200
2	S' = 1 3 1 2 2 3 4 4	1.81	-0.56	-	-	1	0	200
3	S' = 3 3 1 2 2 4 4 1	1.92	0.11	1.00	0.71	2	0	153.72
4	S' = 1 4 2 2 4 3 3 1	2.37	0.45	0.99	0.94	3	0	104.06
5	S` = 1 3 2 3 2 4 4 1	2.05	-0.32	-	-	4	0	64.95
6	S' = 4 3 1 2 2 4 3 1	2.07	0.02	0.94	0.97	5	1	38.22
7	S' = 1 1 2 2 4 4 3 3	2.23	0.16	0.99	0.65	6	1	21.49
8	S` = 1 3 2 4 2 3 4 1	1.86	-0.37	-	-	7	1	11.64

We formulated the problem of flexible job shop scheduling as a search problem and applied a search technique "Simulated Annealing" to find the minimum makespan. From above table we conclude that the optimal solution for our problem statement is to find the maximum number of different jobs which are the best job sequence is 1 3 1 2 2 3 4 4 & minimum total workload WT is -0.56.

Here, Table 4 shows the Result for different job sequences graphical result shown below in Figure 2.

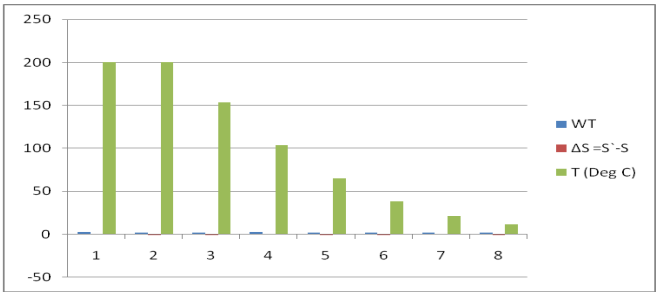

Fig. 2- Graph showing the result obtained using Simulated Annealing for FJSP

Conclusion and Future Research

In this paper, we develop Simulated Annealing algorithm for Priority Base Flexible Job-shop Scheduling Problem (FJSP). Assign priority give better result than random sequence, Similarly from the experiment, we find that our SA algorithm is better than the heuristic method. With our algorithm, we can get better result.

References

[1] Zhang Gen, *Waseda University*.
[2] Van Laarhoven P.J.M. Aarts E.H.L. and Lenstra J.K. (1992) *Computers and Operations Research*, 40(1). 113-25.
[3] Liang Sun, Xiaochun Cheng and Liang (2010) *International Journal of Intelligent Information Processing,* 1(2).
[4] Jose Fernando, Jorge Mendes and Mauricio (2002) *AT & T Labs Research* TD-5EAL6J.
[5] Li-Ning Xing, Ying-Wu Chen and Ke-Wei Yang (2009) *Applied Soft Computing*, 362-376.
[6] Hongze Wanli and Hailong Wang (2009) *Fifth International Conference on Natural Computation*.
[7] Mukhopadhyay S.K., Midha S. and Murlikrishna (1992) *International Journal of Production Research*, 27(6), 1019-1034.
[8] ZRIBI N., KACEM I., EL KAMEL A. and BORNE P. *Optimization by Phases for Flexible Job Shop Scheduling Problem*.
[9] Arash Motaghedi-larijani, Kamyar Sabri-laghaie and Mahdi Heydari (2010) *International Journal of Engineering, Science and Technology,* 2(1), 144-151.
[10] Fatos Xhafa and Javier Carretero Bernab´e Dorronsoro (2009) *Computing and Informatics,* 28, 1001-1014.
[11] Kamrul Hasan S.M. *GA with Priority Rules for Solving Job-Shop Scheduling Problems*.