Matematika, 2004, Jilid 20, bil. 1, hlm. 1–17

A Branch and Bound and Simulated Annealing Approach for Job Shop Scheduling

Tan Hui Woon & Sutinah Salim
Department of Mathematics, Universiti Teknologi Malaysia, 81310 UTM, Skudai, Johor, Malaysia

Abstract This paper presents two approaches to the solution of the job shop scheduling problem, namely the branch and bound, and simulated annealing approach. The objective is to schedule the jobs on the machines so that the total completion time is minimized. In the branch and bound approach, the job shop scheduling problem is represented by a disjunctive graph, then the optimal schedule is obtained using the branch and bound algorithm while simulated annealing is a local search based algorithm which will slightly perturb the initial feasible solution to decrease the makespan.

Keywords Job shop scheduling, branch and bound, disjunctive graph formulation, simulated annealing.

Abstrak *Kertas ini membincangkan dua penyelesaian untuk menyelesaikan masalah penjadualan bengkel kerja iaitu kaedah cabang dan batas serta kaedah simulasi penyepuhlindapan. Objektifnya adalah untuk menjadualkan kerja-kerja kepada mesin-mesin supaya jumlah masa penyelesaian adalah minimum. Dalam kaedah cabang dan batas, masalah penjadualan bengkel kerja telah diwakili oleh graf disjungtif, selepas itu, jadual optima boleh didapati dengan menggunakan algoritma cabang dan batas manakala simulasi penyepuhlindapan merupakan algoritma pencarian tempatan yang akan membuat usikan kecil kepada penyelesaian awal tersaur untuk mengurangkan "makespan."*

Katakunci *Penjadualan bengkel kerja, cabang dan batas, formulasi graf disjungtif, simulasi penyepuhlindapan.*

1 Introduction

The history of the job shop scheduling problem starting more than 30 years ago, is also the history of a well known benchmark problem consisting of ten jobs and ten machines and introduced by Fisher and Thompson in 1963 (Blazewicz et al, [2]). This 10-job, 10-machine problem leading to a competition among researchers to find the most powerful solution procedure. Since then, branch and bound procedure have received substantial attention from numerous researchers.

Recently, another approach that becomes very popular in job shop scheduling is local search approach. The algorithms for the local search approach are all based on a certain neighborhood structure and some rules on how to obtain a new solution from the existing one. These approaches include simulated annealing, parallel tabu search, and genetic algorithm.

The general job shop structure fits many scheduling problem arising in business, computing, government, social services, industries and et cetera. Suppose we have n jobs $\{J_1, J_2, \ldots, J_n\}$ to be processed through m machines $\{M_1, M_2, \ldots, M_m\}$. In computer scheduling, the machines are referred as processors. Here, we suppose each job must pass through each machine once. The technological constraints demand that each job should be processed through the machine in a particular order. For general job shop problem, each job has its own processing order.

An operation is the processing of a job on a machine. A processing time is the length of time that each operation takes. For ith machine and jth job, we denote the operation as (i, j) and the processing time to be P_{ij}. We assume that P_{ij} are fixed and known in advance. Now, we shall make an important restriction throughout this paper that we assume; every numeric quantity is deterministic and known by the scheduler. Besides, we also assume that the machines are always available but not necessary for the jobs because some jobs may not become available until after the scheduling period has started.

In this paper, we will only discuss one objective function, i.e. to minimize the overall length of the scheduling period. By definition of Barker [1], the makespan is the length of time required to complete all jobs and denote as $C_{\max}$. Therefore, our objective now is to minimize the makespan.

A simple notation is needed so that we can represent the types of the scheduling problem easily. French [4] classify the problems according to the form of $n/m/A/B$ where n is the number of jobs and m is the number of machines. A described the flow pattern or the disciplines within the machine shop. A is left blank when $m = 1$. For general job shop case where there is no restriction on the form of the technological constraints, we denote it as G. Lastly, B describes the performance index which is to minimize the makespan in this case.

2 Disjunctive graph formulation

In general, job shop scheduling problem can be represented by a disjunctive graph. In a directed graph $G = (N, X, Y)$, the node N corresponds to all of the processing operations performed on the n jobs.

$$N = \{(i, j) | (i, j) \text{ is operation of job } j \text{ on machine } i \}.$$

The conjunctive (solid) arc, X, represents the processing order of the task belonging to the same job $(i, j) \longrightarrow (k, j) \in X$, which means that operation (i, j) precedes (k, j) in completing job j. In this case, (i, j) is called the predecessor and (k, j) is called the successor of job (i, j).

The disjunctive (dotted) arc, Y, represents the conflicts on machines. It can be say as two operations, belonging to two different jobs that are to be processed on the same machine are connected to one another by two disjunctive arcs going in opposite direction, $(i, j) \dashrightarrow\!\!\!\!\leftarrow\!\!\!\!\dashrightarrow (i, l) \in Y$.

So, in the m machine job shop problem, the disjunctive arcs will form m cliques of double arcs in the disjunctive graph, one clique for each machine. In this case a clique is referred to a graph in which all pairs of nodes are connected to one another. All nodes (operations) that are connected to each other in a clique have to be processed on the same machine. All the arcs (conjunctive and disjunctive) that emanating from a node have their own length of processing time. In addition, a disjunctive graph starts and ends with two dummy nodes,

U and V which representing the source and the sink respectively. The source emanates conjunctive arcs to all of the first operations on each job and all the last operations of the jobs, will comes to the sink at the end. The arcs emanating from source have zero length.

To clarify the disjunctive graph model, let us look at Example 1, a $3/3/G/C_{\max}$ problem. The machine sequence and the processing time for each job is shown in Table 1 below.

Example 1:

Table 1: $3/3/G/C_{\max}$ problem

Job	Machine sequence	Processing time
1	$1, 3, 2$	$P_{11} = 3, P_{31} = 2, P_{21} = 3$
2	$2, 3$	$P_{22} = 2, P_{32} = 3$
3	$2, 1, 3$	$P_{23} = 2, P_{13} = 3, P_{33} = 4$

The disjunctive graph model for Example 1 can be illustrated as follows:

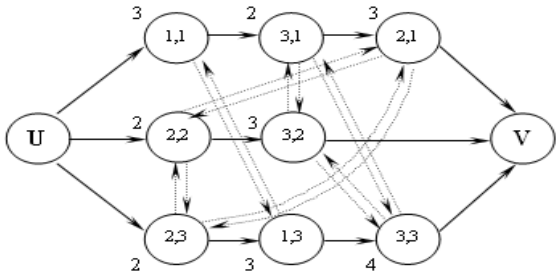


Figure 1: Disjunctive graph model for Example 1

To solve the disjunctive graph model, we need to choose a feasible selection. A subset $D \subset Y$ is called a selection if it contains exactly one directed disjunctive arc from each pair of it. A selection D is called a feasible selection if the resulting directed graph with all the conjunctive arcs and selected disjunctive arcs is acyclic. Such a feasible selection determines the sequence of all the operations that to be produced on the same machine. Therefore, each feasible selection leads to a feasible schedule.

To determine the makespan in the feasible schedule represented by the disjunctive graph, we need to calculate the longest path from U to all other nodes in $G(D)$. The length of a longest path from U to V is equal to the makespan of the schedule. Such a longest path is also called the critical path.

3 A Branch and Bound Approach

In solving job shop scheduling problem, one of the branching procedures is to generate all the active schedules and choose the optimal schedule i.e. the schedule with minimum makespan. According to Pinedo [8], a feasible schedule is called active if no operation can be completed earlier by altering processing sequence on machines and not delaying any other operation. However, it would spend quite a long time to produce all active schedules. Even

so, some improvements can be made by using the generation scheme in a branch and bound setting. Hence, we need a generation scheme to produce all the active schedules for a job shop problem.

All active schedules can be generated in the following algorithm. Before that, there are some notations to be introduced. First, the set of all operations of whose predecessors have already been scheduled is denoted by Ω. Next, r_{ij} denotes the earliest possible starting time of operation $(i,j) \in \Omega$ and Ω' is the subset of Ω. In this case, r_{ij} can be calculated via the longest path calculations.

Active schedule algorithm Pinedo, [8])

Step 1: (Initial condition)

$\Omega := \{$ First operation of each job $\}$

$r_{ij} := 0$ for all $(i,j) \in \Omega$

Step 2: (Machine selection)

Compute $t(\Omega)$ for current partial schedule.

$t(\Omega) := \min_{(i,j)\in\Omega} \{r_{ij} + P_{ij}\}$

$i^* :=$ machine on which the minimum is achieved.

Step 3: (Branching)

$\Omega' := \{ (i^*, j) | r_{i^*j} < t(\Omega) \}$

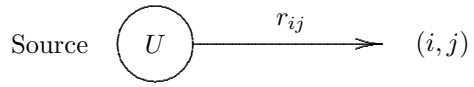
- For all $(i^*, j) \in \Omega'$, extend partial schedule by scheduling (i^*, j) next on machine i^*.
- For each such choice, delete (i^*, j) from Ω.
- Add job successor of (i^*, j) to Ω.
- Return to Step 2.

The algorithm given is based on the branching scheme. The nodes of the branching tree are corresponding to the partial schedules. Step 3 branches from the node corresponding to the current partial schedule. The number of branched is depend on the number of operations in Ω'. A branch corresponds to the choice of an operation (i^*, j) to be scheduled next on machine i^*. In other word, a branch fixes new disjunctions. To find a lower bound of the makespan, consider graph $G(D')$ i.e. the graph for branches of selection D. A simple lower bound can be calculated as the length of the critical path in $G(D')$. In fact, a better (higher) lower bound can be easily obtained. Consider machine i and assume that all other machines are allowed to process more than one job at any time. It is because not all disjunctive arcs are selected yet in $G(D')$. Hence, more than one operation may be processed on a machine at one time. To obtain a better lower bound, we shall consider a 1-machine problem for each machine and find the maximum lateness, $L_{\max}$. Now, the sequencing of all operations on machine i for $i = 1, 2, \ldots, m$ is equivalent to $n/1/L_{\max}$. There is simple algorithm to solve this problem although this problem is strongly *NP-hard.*

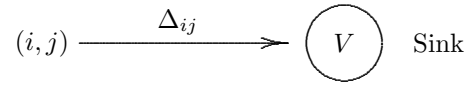
Algorithm of solving $n/1/L_{\max}$ problem

Step 1: Calculate the lower bound (LB), i.e. the longest path from the U to V in $G(D')$.

Step 2: Calculate the earliest possible starting time, r_{ij} for all operations (i, j) on machine i. This is equivalent to the longest path from the source to operation (i, j) in $G(D')$.



Step 3: Calculate minimum amount of time, Δ_{ij} between starting of (i, j) and end of schedule. This can be represented by the longest path from (i, j) to sink in $G(D')$.



Then, calculate the due date.

Due date, $d_{ij} = LB - \Delta_{ij} + P_{ij}$

Step 4: Solve the single machine problem with respects to release dates, no preemption and to minimize the lateness, L_i.

We need to repeat this algorithm for all the machines. Then we will get the maximum lateness, $L_{\max}$, from $L_1, L_2, \ldots, L_m$. A better lower bound can be obtained by using the following formula.

$$LB^{new} = LB + \max_{i=1}^{m} L_i$$

The largest makespan obtained this way can be used as the lower bound. So, continuing the branch and bound procedure to obtain the makespan, which is corresponding to the minimum lower bound.

Solution for Example 1:

The initial graph that contains only the conjunctive arcs for the problem above can be illustrated in the following figure. For convenience, we let M1 = machine 1, M2 = machine 2 and M3 = machine 3.

Level 0:

$$\Omega = \{(1,1), (2,2), (2,3)\}$$
$$t(\Omega) = \min\{0+3,\ 0+2,\ 0+2\} = 2$$
$$i^* = 2$$
$$\Omega' = \{(2,2), (2,3)\}$$

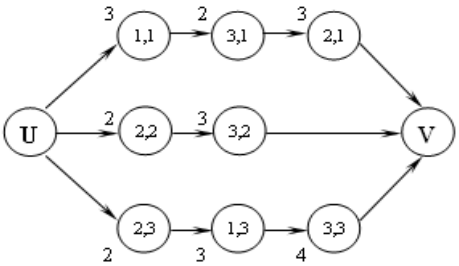


Figure 2: Initial graph for *3/3/G/*C_{max} problem

So, the initial problem will be branched into two parts. One of the branches is the operation (2,2) scheduled first and another branch is operation (2,3) scheduled first. Hence, we denote the operation (2,2) scheduled first as level 1(a) and operation (2,3) scheduled first as level 1(b). Then the branching procedure at level 0 can be shown in Figure 3 below.

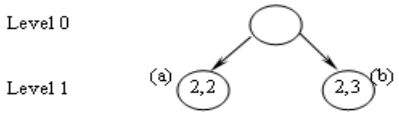
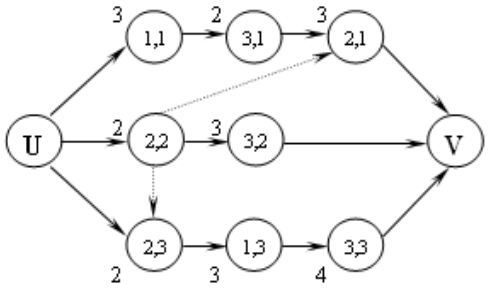


Figure 3: The branching procedure at level 0

Now, we will concentrate to the branching procedure of part (a) first, i.e. start from level 1(a).

Level 1(a):

Operation (2,2) scheduled first on M2. Therefore, it fixed two disjunctive arcs, $(2,2) \dashrightarrow (2,1)$ and $(2,2) \dashrightarrow (2,3)$ at level 1. The corresponding disjunctive graph is illustrated as follows:



$$LB = l(U,(2,2),(2,3),(1,3),(3,3),V)$$
$$= 11$$

Figure 4: Disjunctive graph at level 1

Now, solve the $n/1/C_{max}$ problem for M1, M2, and M3 to find the better lower bound.

Data for jobs on M1:

Table 2: Data of M1 at level 1(a)

Job 1	Job 3
$r_{11} = 0$	$r_{13} = 4$
$\Delta_{11} = 8$	$\Delta_{13} = 7$
$d_{11} = 6$	$d_{13} = 7$

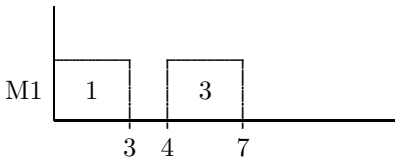


Figure 5: Job sequence of M1 at level 1(a)

From the table above, we can draw the job sequence for M1 as illustrated in Figure 5. From there, we know that the lateness, $L_1 = 0$. (As all the jobs on M1 is completed before the due date.)

Data for jobs on M2:

Table 3: Data of M2 at level 1(a)

Job 1	Job 2	Job 3
$r_{21} = 5$	$r_{22} = 0$	$r_{23} = 2$
$\Delta_{21} = 3$	$\Delta_{22} = 11$	$\Delta_{23} = 9$
$d_{21} = 11$	$d_{22} = 2$	$d_{23} = 4$

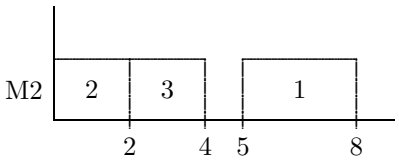


Figure 6: Job sequence of M2 at level 1(a)

The completion time for jobs on M2 is earlier than the due date, so $L_2 = 0$.

Data for jobs on M3:

Table 4: Data of M2 at level 1(a)

Job 1	Job 2	Job 3
$r_{31} = 3$	$r_{32} = 2$	$r_{33} = 7$
$\Delta_{31} = 5$	$\Delta_{32} = 3$	$\Delta_{33} = 4$
$d_{31} = 8$	$d_{32} = 11$	$d_{33} = 11$

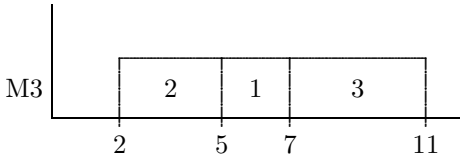


Figure 7: Job sequence of M2 at level 1(a)

The completion time for jobs on M3 is same as the due date, so $L_3 = 0$. Hence, the better lower bound can be obtained as follows:

$$\begin{aligned} LB^{new} &= LB + \max_{i=1}^{m} L_i \\ &= 11 + \max\{0, 0, 0\} \quad = 11 + 0 \quad = 11 \end{aligned}$$

To find the next operation to be schedule, we delete the operation (2,2) from Ω and then add the operation (3,2) i.e. the immediate operation after (2,2) to Ω.

$$\begin{aligned} &\Omega = \{(1,1), (3,2), (2,3)\} \\ &t(\Omega) = \min\{0+3,\ 2+3,\ 2+2\} \quad = 3 \\ &i^* = 1 \\ &\Omega' = \{(1,1)\} \end{aligned}$$

So the next branching will be scheduled operation (1,1) on M1 after operation (2,2). The branching procedure at level 1(a) can be shown in Figure 8 below.

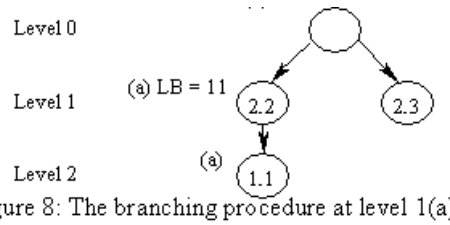


Figure 8: The branching procedure at level 1(a)

Level 2(a):

Operation (1,1) scheduled on M1 after operation (2,2). The corresponding disjunctive graph is depicted as follows:

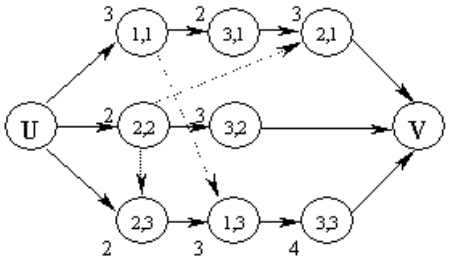


Figure 9: Disjunctive graph at level 2(a)

$$LB = l(U,(2,2),(2,3),(1,3),(3,3),V) = 11$$

Data for jobs on M1:

Table 5: Data of M1 at level 2(a)

Job 1	Job 3
$r_{11} = 0$	$r_{13} = 4$
$\Delta_{11} = 10$	$\Delta_{13} = 7$
$d_{11} = 4$	$d_{13} = 7$

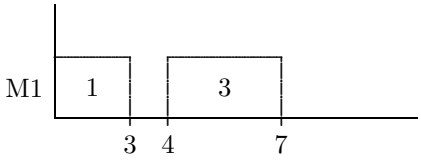


Figure 10: Job sequence of M1at level 2(a)

Since the completion time for jobs on M1 is same as the due date, $L_1 = 0$.

Data for jobs on M2:

Table 6: Data of M2 at level 2(a)

Job 1	Job 2	Job 3
$r_{21} = 5$	$r_{22} = 0$	$r_{23} = 2$
$\Delta_{21} = 3$	$\Delta_{22} = 11$	$\Delta_{23} = 9$
$d_{21} = 11$	$d_{22} = 2$	$d_{23} = 4$

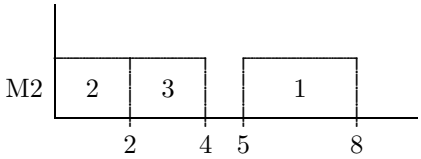


Figure 11: Job sequence of M2 at level 2(a)

The completion time for jobs on M2 is earlier than the due date, so $L_2 = 0$.

Data for jobs on M3:

Table 7: Data of M3 at level 2(a)

Job 1	Job 2	Job 3
$r_{31} = 3$	$r_{32} = 2$	$r_{33} = 7$
$\Delta_{31} = 5$	$\Delta_{32} = 3$	$\Delta_{33} = 4$
$d_{31} = 8$	$d_{32} = 11$	$d_{33} = 11$

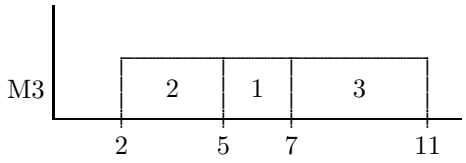


Figure 12: Job sequence of M3 at level 2(a)

From Figure 12, we know that $L_3 = 0$. Hence,

$$\begin{aligned} LB^{new} &= LB + \max_{i=1}^{m} L_i \\ &= 11 + \max\{0, 0, 0\} \quad = 11 + 0 \quad = 11 \end{aligned}$$

To find the next operation to be schedule, we delete the operation (1,1) from Ω and then add the operation (3,2) to Ω.

$$\begin{aligned} &\Omega = \{(3,1), (3,2), (2,3)\} \\ &t(\Omega) = \min\{3+2,\ 2+3,\ 2+2\} \quad = 4 \\ &i^* = 2 \\ &\Omega' = \{(2,3)\} \end{aligned}$$

So the next branching will be schedule operation (2,3) on M2 at level 3(a). The branching procedure at level 2(a) can be shown in Figure 13 below.

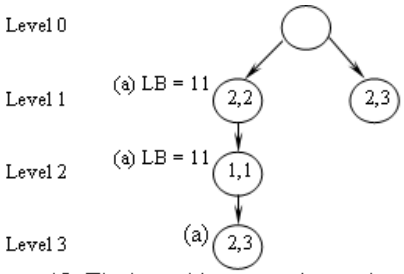
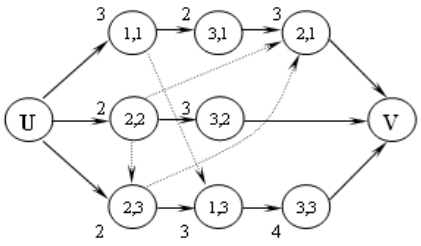


Figure 13: The branching procedure at level 2(a)

Level 3(a):

Operation (2,3) scheduled on M2 after operation (1,1). So it means that the job sequence for M2 is 2-3-1. Therefore, it fixes a disjunctive arc from (2,3) to (2,1). The corresponding disjunctive graph is shown as follows:



$$\begin{aligned} LB &= l(U, (2,2), (2,3), (1,3), (3,3), V) \\ &= 11 \end{aligned}$$

Figure 14: Disjunctive graph at level 3(a)

Even though it fixes a new disjunctive arc on the graph, the lower bound is still the same. If we gone through the $n/1/L_{max}$ problem for M1, M2, and M3, we will notice that the better lower bound is still 11.

To find the next operation to be schedule, we delete the operation (2,3) from Ω and then add the operation (1,3) to Ω.

$$\Omega = \{(3,1),(3,2),(1,3)\}$$
$$t(\Omega) = \min\{3+2,\ 2+3,\ 4+3\} \quad = 5$$
$$i^* = 3$$
$$\Omega' = \{(3,1),(3,2)\}$$

The branching procedure at level 3(a) can be shown in Figure 15 below.

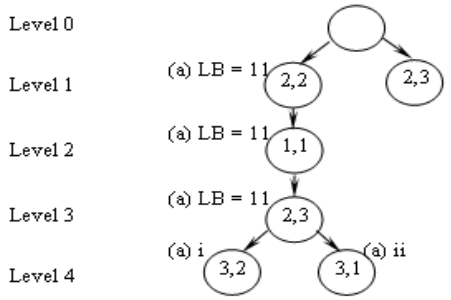


Figure 15: The branching procedure at level 3(a)

The branching procedure above is just for the part 3(a), the calculation for the lower bound is the same for the branching procedure for part 4(a)i, 4(a)ii, part (b) and vice versa. Since the calculation is too long and takes a lot of space, the other branching procedure is not shown in this paper. However, the final branching tree is shown in Figure 16 below.

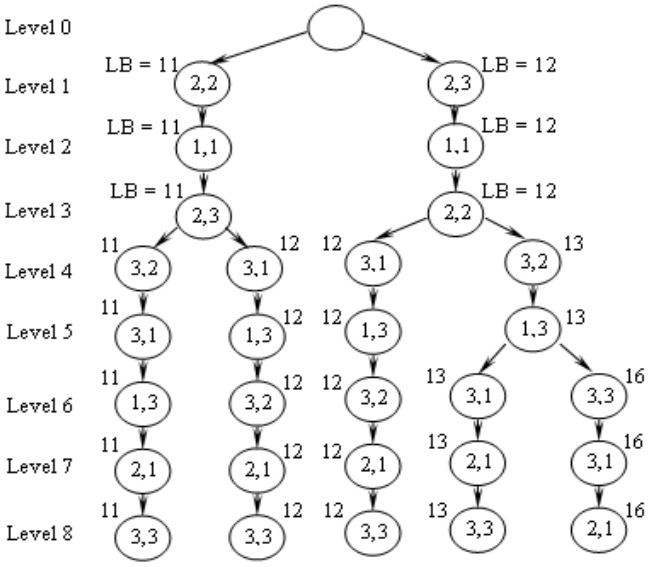


Figure 16: The final branching tree for Example 1

From the branching tree above, we can see that the minimum lower bound (makespan) is 11 hours. Hence, the optimal schedule for Example 1 is shown in Figure 17.

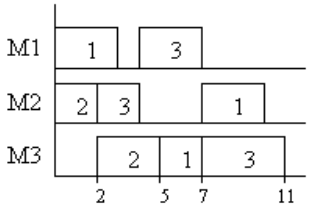


Figure 17: The optimal schedule for Example 1

4 Simulated Annealing Approach

Local search based scheduling of job shop plays an important role to solve our problem and become very popular in recent years. The applicable approximation procedures in the local search process include tabu search, simulated annealing and genetic algorithm. These methods can be viewed as tools for searching a space of legal alternatives in order to find a best solution within reasonable time limitation (Blazewicz et al, [2]).

Simulated annealing is based on the analogy to the physical process of cooling and recrystalization of metals. According to Jones and Rabelo [5], the current state of the thermodynamic system is equivalent to the current scheduling solution, the energy function for the thermodynamic system is analogous to the objective function and the ground state is similar to the global optimum. In the algorithm of simulated annealing, the initial solution is chosen at random. Then, a neighbor of this solution is generated based on a certain neighborhood structure and the change in the energy function is calculated. If a reduction in the energy function is obtained, the current solution is replaced by the generated neighbor (Krishna et al, [7]). On the other hand, if the energy function is more than the initial one, the generated neighbor will replace the current solution with a Boltzmann probability function given by:

$$\Pr(\text{accepted}) = Exp\left(-\left\{E[j] - E[i]\right\}/T\right)$$

where $E[j]$ is the energy function of the generated state while $E[i]$ is the energy function at the initial state. T is a control parameter, which corresponds to the temperature in the physical annealing process. If we denote $E[j] - E[i]$ as ΔE, the Boltzmann probability function will become $\Pr(\text{accepted}) = EXP(-\Delta E/T)$ (Kirkpatrick et al, [6]).

A small value ε is used to measure the acceptance or rejection for the probability. The above acceptance function implies that small increases in E (energy function) are more likely to be accepted than the large increases. Besides, when T is high, most of the generated neighbors are accepted. However, as T approaches zero, most of the energy increasing transitions are rejected. In addition, the initial temperature in the simulated annealing algorithm is kept high so that the algorithm does not get trapped in a local optimum (Krishna et al, [7]). The algorithm proceeds by generating a certain number of neighbors at each temperature, while the temperature parameter drops gradually. By Krishna et al, [7], this algorithm leads to a near optimal solution.

To apply the simulated annealing approach, the job shop scheduling problem has to be presented in a disjunctive graph. As the objective function in our job shop scheduling problem is to minimize the makespan which is analogous to the length of the critical path

in the disjunctive graph, the energy function in the simulated annealing will be the length of the critical path. According to Salleh and Zomaya, [9], a simulated annealing algorithm for single-row routing problem is given as follows:

Simulated Annealing Algorithm for Single-row Routing Problem (Salleh and Zomaya, [9])

(i) Select a large starting value for the temperature $T = T_0$.

(ii) Select a suitable value for the reducing parameter α, $0 < \alpha < 1$.

(iii) Set the initial net ordering at random.

(iv) Evaluate the initial energy (makespan), $E(i)$.

(v) While T is in the cooling range,

(a) Select a net at random and change its position.

(b) Evaluate the new energy, $E(j)$ from this configuration and the resulting change in the energy ΔE.

(vi) If $\Delta E < 0$, accept the new state.

If $\Delta E > 0$, accept the new state if $\Pr(\text{accepted}) = EXP(-\Delta E/T) > \varepsilon$.

(vii) Update T according to $T = \alpha T$.

(viii) Repeat steps from (v) to (vii) until ΔE is not significant for small changes in T.

The algorithm above in Salleh and Zomaya, [9] is suitable for the single-row routing problem. So, the algorithm needs to be modified to fulfill the job shop scheduling problem. So in step (iii), it sets an initial net ordering at random while in the job shop scheduling problem, we need to set the initial feasible schedule in the disjunctive graph. Besides, step v(a) above obtains the new solution by selecting a net at random and changes its position. However, in the job shop problem, we obtain the new solution by perturb the initial disjunctive graph with some neighborhood structure. The modifications that have been made is shown as follows:

Simulated Annealing Algorithm for Job Shop Scheduling Problem

(i) Select a large starting value for the temperature $T = T_0$.

(ii) Select a suitable value for the reducing parameter α, $0 < \alpha < 1$.

(iii) Set initial feasible schedule in disjunctive graph.

(iv) Evaluate the initial energy (makespan), $E(i)$.

(v) While T is in the cooling range,

(a) Perturb the disjunctive graph with some neighborhood structure.

(b) Evaluate the new energy, $E(j)$ from this configuration and the resulting change in the energy ΔE.

(vi) If $\Delta E < 0$, accept the new state.
If $\Delta E > 0$, accept the new state if $\Pr(\text{accepted}) = EXP(-\Delta E/T) > \varepsilon$.

(vii) Update T according to $T = \alpha T$.

(viii) Repeat steps from (v) to (vii) until ΔE is not significant for small changes in T.

According to Blazewicz et al [3], a very simple neighborhood structure can be shown as follows:
A simple neighborhood structure

- A transition from a current solution to a new one is generated by replacing in the disjunctive graph representation of the current solution, a disjunctive arc (i, j) on a critical path by its opposite arc (j, i).

Solution for Example 1 by simulated annealing:

From the disjunctive graph model in Figure 1, we can choose the initial feasible schedule randomly and one of them is depicted in Figure 18 below. Hence, by the $T = 100, \alpha = 0.95$ and $\varepsilon = 0.98$, let us solve Example 1 by simulated annealing approach.

Stage 0:

Let the arrow, $\longrightarrow$ be the critical path in the disjunctive graph. Then, E is calculated by the summation of the length of these arrows.

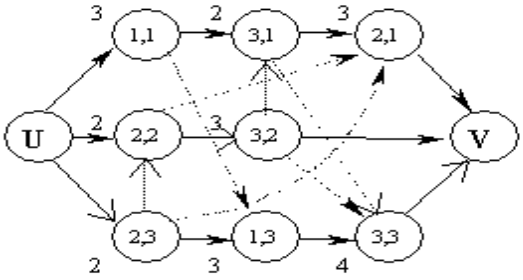


Figure 18(a)

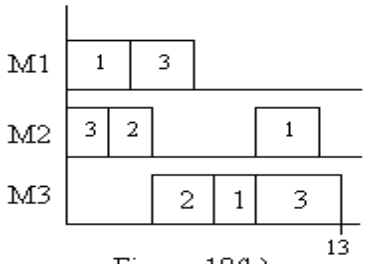


Figure 18(b)

Figure 18: (a) Disjunctive graph at stage 0
(b) Gantt chart at stage 0

$$T = 100,\ E(0) = 13$$

Stage 1:

Next, we generate a new solution by the neighborhood structure discussed earlier, i.e. we arbitrary change a disjunctive arc on the critical path to its opposite direction. Let us change the disjunctive arc from $(2,3) \dashrightarrow (2,2)$ to $(2,2) \dashrightarrow (2,3)$ in the critical path.

Then, the new disjunctive graph is generated as in Figure 19(a) below. After that, the temperature, new energy $E(1)$, and ΔE is calculated.

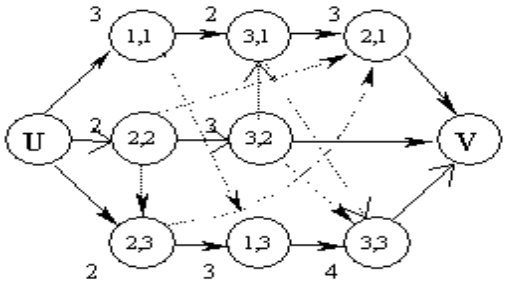


Figure 19(a)

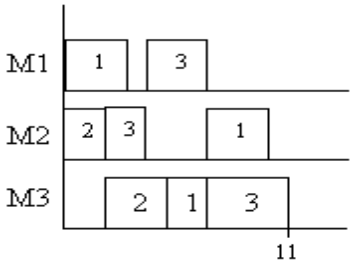


Figure 19(b)

Figure 19: (a) Disjunctive graph at stage 1
(b) Gantt chart at stage 1

$T = \alpha T$
$= 0.95(100)$
$= 95$

$E(1) = 11$
$\Delta E = 11 - 13$
$= -2 < 0$
$\therefore$ We accept this as a new schedule.

Stage 2:

Next, we generate a new solution from the schedule at stage 1. We change the disjunctive arc from $(3,2) \dashrightarrow (3,1)$ to $(3,1) \dashrightarrow (3,2)$ and calculate the new energy function as follows:

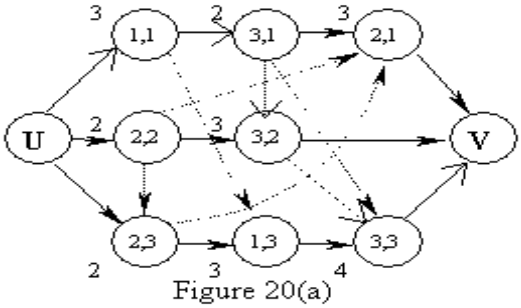


Figure 20(a)

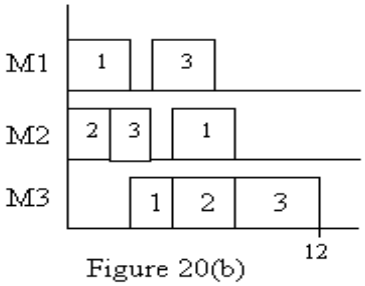


Figure 20(b)

Figure 20: (a) Disjunctive graph at stage 2
(b) Gantt chart at stage 2

$T = \alpha T$
$= 0.95(95)$
$= 90.25$

$E(2) = 12$
$\Delta E = 12 - 11$
$= 1$

$\Pr(\text{accepted}) = \exp(-\Delta E/T)$
$= \exp(-1/90.25)$
$= 0.9890 > \varepsilon$

$\therefore$ Since the $\Pr(\text{accepted}) > \varepsilon$, this schedule is accepted.

Stage 3:

Next, we generate a new solution from the schedule at stage 2 as the schedule is accepted. We change the disjunctive arc from $(3,2) \dashrightarrow (3,3)$ to $(3,3) \dashrightarrow (3,2)$ and calculate

the new energy function as follows:

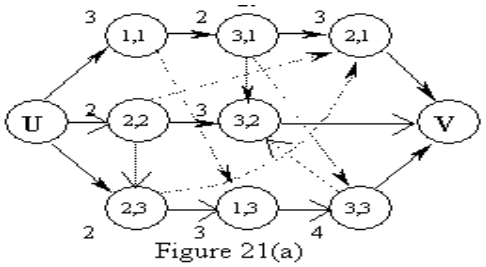


Figure 21(a)

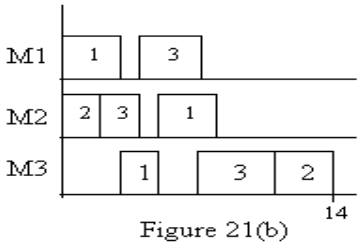


Figure 21(b)

Figure 21: (a) Disjunctive graph at stage 3
(b) Gantt chart at stage 3

$T = \alpha T$ $E(3) = 14$ $\Pr(\text{accepted}) = \exp(-\Delta E/T)$
$= 0.95(90.25)$ $\Delta E = 14 - 12$ $= \exp(-2/85.7375)$
$= 85.7375$ $= 2$ $= 0.9769 < \varepsilon$
$\therefore$ Since the Pr(accepted) $< \varepsilon$, this schedule is rejected.

Since the schedule at stage 3 is rejected, we start the neighborhood structure from stage 2 again. However, there is no more changes can be made in Figure 21(a) because the transition of both disjunctive arcs has been done in the previous stage. Hence, this is the annealing process for the Example 1. If we choose different neighborhood structure from the beginning of stage 0, we may need more or less stages to get the optimum solution. The change of the energy function at each stage of Example 1 can be shown in the following table.

Table 8: Simulated annealing accepted change for Example 1

Stage	E	ΔE	T	Boltzmann Probability
0	13		100.00	
1	11	-2	95.00	
2	12	1	90.25	
3	14	2	85.7375	0.9769

From Table 8 above, the best solution is performed at stage 1 with the minimum energy (makespan) of 11. Therefore, the optimal schedule for Example 1 which solve by simulated annealing is shown as follows:

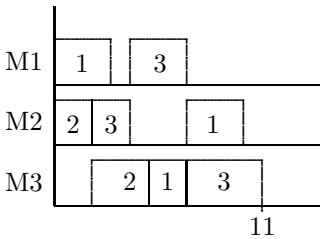


Figure 22: Optimal schedule for Example 1 solved by simulated annealing

5 Comparison between branch and bound and simulated annealing approaches in solving $3/3/G/C_{max}$ problem

In previous sections, we have shown the solution approach for the $3/3/G/C_{max}$ problem by using branch and bound and simulated annealing approach. We notice that the optimal solution obtained by these two approaches is the same i.e. we get a makespan of 11 hours. In the branch and bound approach, we have to find every possible branching procedure and calculate the lower bound. It takes a long time, as the problem grows larger. For instance, in Example 1, it needs 8 stages to get the optimal schedule by using branch and bound technique. On the other hand, the same example is solved within 4 stages by simulated annealing approach. Hence, we can conclude that simulated annealing is a better solution approach for solving job shop scheduling problem. However, the only disadvantage of simulated annealing is sometimes it may not get an optimal solution but just a near optimal solution. It is because this approach does not consider all the possible sequence of each job on each machine when the problem getting larger.

6 Conclusion

The problem of job shop scheduling has been examined and two solution approaches are shown in this paper. From the result of both approaches, we can conclude that simulated annealing is a very useful approach to solve larger problem as we can get the optimal solution or just a near optimal solution in a short time. To do that we need to use some programming tools, so that we can solve the larger problem easily and not by using manual calculation.

Acknowledgement

The authors would like to thank the Department of Mathematics, Universiti Teknologi Malaysia (UTM) for the support in this work.

References

[1] K. R. Barker, *Introduction to Sequencing and Scheduling,* John Wiley & Sons, New York, 1974.

[2] J. Blazewicz, W. Domschke, E. Pesch, *The Job Shop Scheduling Problem: Conventional And New Solution Techniques,* European Journal of Operational Research 93 (1996), 1–33.

[3] J. Blazewicz, K. H. Ecker, E. Pesch, G. Schmidt, J. Weglarz, *Scheduling Computer and Manufacturing Processes,* Springer-Verlag, Berlin Heidelberg New York, 2001.

[4] S. French, *Sequencing and Scheduling, An Introduction to the Mathematics of Job-Shop,* Ellis Horwood, New York, 1982.

[5] A. Jones & L. C. Rabelo, *Survey of Job Shop Scheduling Techniques,* http://www.mel.nist.gov/msidlibrary/doc/luis.pdf, 2003, 3.45 p.m..

[6] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi, *Optimization by Simulated Annealing,* Science, Vol. 220. No. 4598.(1983), 671–680.

[7] K. Krishna, K. Ganeshan, and D. Janaki Ram, *Distributed Simulated Annealing Algorithms for Job Shop Scheduling,* IEEE Transactions on Systems, Man. And Cybernetics. Vol. 25. No. 7.(1995), 1102–1109.

[8] M. Pinedo, *Scheduling Theory, Algorithms and Systems,* Prentice-Hall, Englewood Cliffs, New Jercey ll, 1995.

[9] Shaharuddin Salleh and A. Y. Zomaya, *Scheduling in Parallel Computing Systems, Fuzzy and Annealing Techniques,* Kluwer Academic Publishers, United States, 1999.