模糊處理時間工作之排程
Scheduling Tasks of Fuzzy Processing Time

洪宗貝
Tzung-Pei Hong

義守大學資訊管理系
Department of Information Management
I-Shou University, Kaohsiung
e-mail: tphong@csa500.isu.edu.tw

莊宗南
Tzung-Nan Chuang

義守大學資訊管理系
Institute of Management Science
I-Shou University, Kaohsiung
e-mail: m831017@csa500.isu.edu.tw

摘要

在這篇文章我們應用模糊觀念至一個啟示性的工作流程排程演算法，Gupta演算法，因為在不確定的環境中求最佳解似乎是不需要的。我們假設每一個工作有一三角模糊隸屬函數來表示它可能的執行時間，並且提出一個新的模糊Gupta排程演算法以求出一個具有模糊完成時間的排程。

關鍵字: 完成時間，模糊排程，Gupta 演算法

Abstract

In this paper, we apply the fuzzy concept to a heuristic flow-shop scheduling algorithm (the Gupta algorithm) since optimal solutions seem not necessary for uncertain environments. We assume each job has a triangular membership function for its possible processing time and propose a new fuzzy Gupta scheduling algorithm to yield a scheduling result with a fuzzy membership function of the final completion time.

Keywords: Completion time, fuzzy scheduling, Gupta algorithm

1. Introduction

One kind of scheduling problems that frequently occur in real-word applications are the flow-shop problems. The processing time for each job has usually been assigned or estimated as a fixed value. In many real-world applications, however, job processing time may vary dynamically with the situation. If the time required to process each job is uncertain, then the finish time of a scheduling schema is apparently also uncertain.

Several theories such as *fuzzy set theory* [15][16], *probability theory*, *D-S theory*[9], and approaches based on *certainty factors*[1], have been developed to account for uncertainty. Among them, fuzzy set theory is more and more frequently used in intelligent control, because of its simplicity and similarity to human reasoning. The theory has been applied to many fields such as manufacturing, engineering, diagnosis, economics, and others[3][8] [12][13].

Although fuzzy set concepts are used mainly in linguistic domains, they can also be used in numerical domains by assigning each number a membership value. Examples are Gazdik's fuzzy network planning [14], Klein's fuzzy shortest path [8], Nasution's fuzzy critical path [12], and Hong et al's fuzzy LPT scheduling [5].

In this paper, the fuzzy concept is applied to a heuristic flow-shop scheduling algorithm (the Gupta algorithm) since optimal solutions seem not necessary for uncertain environments. Each job ia assumed to have a triangular membership function for its possible processing time. A new fuzzy Gupta scheduling algorithm, which adopts the fuzzy Johnson procedure [7], is then proposed to yield a scheduling result with a fuzzy membership function of the final completion time.

2. Review of Gupta algorithm for *m*-machine flow shop

In this section, we state an *m*-machine (m>2) flow shop problem. Given a set of *n* independent jobs, each having *m* tasks (T_{11}, T_{21}, ..., T_{m1}, T_{12}, T_{22}, ..., $T_{(m-1)n}$, T_{mn}) that must be executed in the same sequence on *m* machines (P_1, P_2, ..., P_m), scheduling seeks the minimum completion time of the last job. Since this problem is a NP-hard problem, Gupta proposed the following heuristic algorithm to solve this problem in polynomial time [11]:

The Gupta algorithm for scheduling jobs in a m-machine (m>2) flow shop:

Input: A set of n jobs, each having m (m > 2) tasks executed respectively on each of m machines.

Output: A schedule with a nearly minimum completion time of the last job.

Step 1: Form the group of jobs U that take less time on the first machine than on the last, such that $U = \{ i \mid t_{1i} < t_{mi} \}$.

Step 2: Form the group of jobs V that take less time on the last machine than on the first, such that $V = \{ j \mid t_{mj} \leqq t_{1j} \}$.

Step 3: For each job J_i in U, find the minimum of ($t_{ki} + t_{(k+1)i}$) for k = 1 to m-1; restated, set:

$$\pi_i = \min_{k=1}^{(m-1)} (t_{ki} + t_{(k+1)i})$$

Step 4: For each job J_j in V, find the minimum of ($t_{kj} + t_{(k+1)j}$) for k = 1 to m-1; restated set:

$$\pi_j = \min_{k=1}^{(m-1)} (t_{kj} + t_{(k+1)j})$$

Step 5: Sort the jobs in U in ascending order of π_i's; if two or more jobs have the same value of π_i, sort them in an arbitrary order.

Step 6: Sort the jobs in V in descending order of π_j's; if two or more jobs have the same value of π_j, sort them in an arbitrary order.

Step 7: Schedule the jobs on the machines in the sorted order of U, then in the sorted order of V.

3. Review of related fuzzy set operations

In this section, fuzzy set concepts used in this paper are reviewed. There are a variety of fuzzy set operations. Among them, three basic and commonly used operations are *complementation*, *union* and *intersection*. Let X be the universal set. Zadeh proposed the following definitions of complement, union, and intersection[9][13]:

(1) The *complement* of a fuzzy set A is denoted by $\neg A$ and the membership function of $\neg A$ is given by:

$$\mu_{\neg A}(x) = 1 - \mu_A(x) \quad \forall x \in X .$$

(2) The *intersection* of fuzzy sets A and B is denoted by $A \cap B$ and the membership function of $A \cap B$ is given by:

$$\mu_{A \cap B}(x) = Min\{ \mu_A(x), \mu_B(x) \} \quad \forall x \in X .$$

(3) The *union* of fuzzy sets A and B is denoted by $A \cup B$ and the membership function of $A \cup B$ is given by:

$$\mu_{A \cup B}(x) = Max\{ \mu_A(x), \mu_B(x) \} \quad \forall x \in X .$$

Triangular membership functions are used here to represent the fuzzy processing time of tasks. A triangular fuzzy membership function can be denoted by A=(a, b, c), where $a \le b \le c$ (Figure 1). The abscissa b represents the variable value with the maximal grade of membership value, i.e. $\mu_A(b)$=1; a and c are the lower and upper bounds of the available area. They are used to reflect the fuzziness of the data.

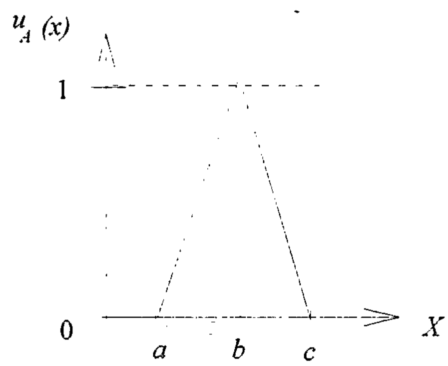

Figure 1: A triangular fuzzy membership function by (a, b, c).

A triangular fuzzy addition operation is usually defined for fuzzy numerical domains as follows. Let A and B be two triangular fuzzy membership functions. A and B can then be represented as follows:

$$A = (a_A, b_A, c_A),$$
$$B = (a_B, b_B, c_B).$$

The *addition* of fuzzy sets A and B is denoted as follows:

$$A + B = (a_A + a_B, b_A + b_B, c_A + c_B)$$

The above fuzzy operations will be used in this paper to schedule jobs with uncertain time .

4. Assumption and notations

The assumptions and notations used in this paper are stated as follows.

Assumptions:

- Jobs are not preemptive.
- Each job has m tasks to be executed in sequence on m machines ($m > 2$).
- The execution-time membership function of each task is triangular and known.

Notations:

- *m: the number of machines;*
- *n: the number of jobs;*
- *T_{ij}: the i-th task for the j-th job, $i=1, 2, ..., m$ and $j = 1, 2, ..., n$;*
- *t_{ij}: the fuzzy execution time $(a_{t_{ij}}, b_{t_{ij}}, c_{t_{ij}})$ (a triangular fuzzy set) of T_{ij}, $i =1, 2, ..., m$ and $j = 1, 2, ..., n$;*
- *P_k: the k-th machine, $k=1, 2, ..., m$;*
- *p_k: the fuzzy execution time (a_{pk}, b_{pk}, c_{pk}) (a triangular fuzzy set) of the k-th machine P_k, $k= 1, 2, ..., m$;*
- *f: the final completion time (a triangular fuzzy set) of the whole schedule.*

5. Triangular fuzzy Gupta algorithm for *m*-machine flow shop

The triangular fuzzy Gupta scheduling algorithm using the averaging ranking method is shown below.

The triangular fuzzy Gupta scheduling algorithm:

Input: A set of n jobs, each with m tasks to be executed respectively on each of m machines; each task has a triangular processing time membership function.

Output: A fuzzy schedule with a completion time membership function f.

Step 1: For each job T_j, find the average height of T_{1j} and T_{mj}, using the following formula:

$$h(t_{ij})=1/3(a_{t_{ij}} + b_{t_{ij}} + c_{t_{ij}}).$$

Step 2: Form the group of jobs U that take fuzzily less time on the first machine than on the last, such that $U = \{ i \mid h(t_{1i}) < h(t_{mi})\}$.

Step 3: Form the group of jobs V that take fuzzily less time on the last machine than on the first, such that $V = \{ j \mid h(t_{mj}) \le h(t_{1j})\}$.

Step 4: For each job r, set $t'_{kr} = t_{kr} + t_{(k+1)r}$, $k = 1$ to $(m-1)$, using the triangular fuzzy addition operation.

Step 5: For each t'_{kr}, find the average processing time $h(t'_{kr})$ using the formula in Step 1.

Step 6: For each job J_i in U, find the minimum of $h(t'_{ki})$ for $k = 1$ to $m-1$; restated, set:

$$\pi_i = \min_{k=1}^{(m-1)} \{h(t'_{ki})\}$$

Step 7: For each job J_j in V, find the minimum of $h(t'_{kj})$ for $k = 1$ to $m-1$; restated, set:

$$\pi_j = \min_{k=1}^{(m-1)} \{h(t'_{kj})\}$$

Step 8: Sort the jobs in U in ascending order of π_i's; if two or more jobs have the same value of π_i, sort them in an arbitrary order.

Step 9: Sort the jobs in V in descending order of π_j's; if two or more jobs have the same value of π_j, sort them in an arbitrary order.

Step 10: Set the initial completion time $p_1, p_2, ..., p_m$ for machines $P_1, P_2, ..., P_m$ to zero with a fuzzy value of 1.

Step 11: Schedule the first job J_j in U to the machines such that T_{1j} is assigned to P_1, T_{2j} is assigned to P_2, ..., and T_{mj} is assigned to P_m.

Step 12: Set $p_1 = p_1 + t_{1j}$ using the triangular fuzzy addition operation.

Step 13: Set $p_{(i+1)}$ = *find-triangular-longer-time* $(p_i, p_{(i+1)}) + t_{(i+1)j}$, for $i = 1, 2, ..., (m-1)$.

Step 14: Remove task J_j from U.

Step 15: Repeat *Steps 11 to 14* until U is empty.

Step 16: Schedule jobs in V in a similar way (*Steps 11 to 14*).

Step 17: Set the final completion time $f = p_m$.

After *Step 17*, scheduling is finished and a completion time with a triangular membership function (f) has been found. In the above algorithm, the averaging height method is used as an example to show the triangular fuzzy Gupta algorithm. Note that other ranking methods can also be used in our fuzzy scheduling algorithm. The find-triangular-longer-time procedure is described below.

6. Find-triangular-longer-time procedure

The find-triangular-longer-time procedure in the fuzzy Johnson scheduling algorithm [7] can be used here. For a two-machine flow shop problem, tasks on machine 2 are executed only after the corresponding tasks on machine 1 are finished. In crisp scheduling, start time on machine 2 is the longer time between the completion time (p_2) of the previous job on machine 2 and the completion time (p_1) of the current job on machine 1. That is, p_2 is the start time for the next job on machine 2 if and only if p_1 is smaller than p_2; similarly, p_1 is the start time for the next job on machine 2 if and only if p_2 is smaller than p_1. This idea has been generalized to triangular fuzzy set.

Assume $\mu p_1(x)/x$ is in the set of completion time p_1 (a triangular fuzzy set) of the current job for machine P_1. As noted for crisp sets, x in p_1 is the start time for machine 2 if no completion time in p_2 is later than x. Therefore, the membership value $\mu_{sp_1}(x)$ of x in p_1 as the start time for machine 2 is:

$$\mu_{sp_1}(x) = \mu_{in\ p_1}(x) \wedge \mu_{not > in\ p_2}(x) = Min\,[\,\mu_{in\ p_1}(x),\ \mu_{not > in\ p_2}(x)\,], \quad (1)$$

where $\mu_{not} > in\ p_2(x)$ denotes the membership value for all the elements in p_2 not bigger than x. The following two cases exist:

Case (a) : x is at the right of b_{p_2}(Figures 2). For this case, $\mu_{not} > in\ P2(x) = 1 - \mu_{P2}(x)$.

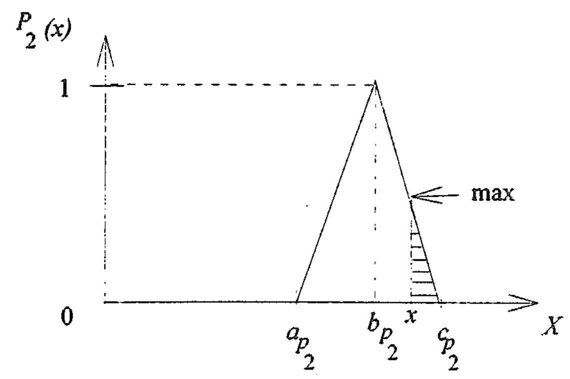

Figure 2. x is at the right of b_{P_2}

Case (b): x is at the left of b_{P_2}(Figures 3). For this case, $\mu_{not} > in\ P2(x) = 1 - 1 = 0$.

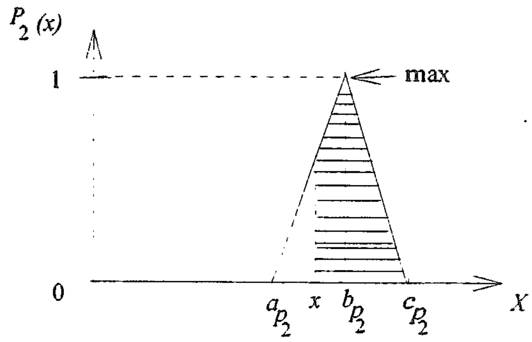

Figure 3. x is at the left of b_{P_2}

Therefore, for a triangular membership function of P_2 (a_{p_2}, b_{p_2}, c_{p_2}), its $\mu_{not > in}\ P2(x)$, denoted by $\mu_{P2'}(x)$ and called *half-inverse fuzzy set* is presented as follows (Figure 4).

$$\mu_{p'_2}(x) = \begin{cases} 0 & , \quad x < b_{P_2} \\ 1-\mu_{P_2}(x) & , \quad b_{P_2} \le x \le c_{P_2} \\ 1 & , \quad x > c_{P_2} \end{cases} \quad (2)$$

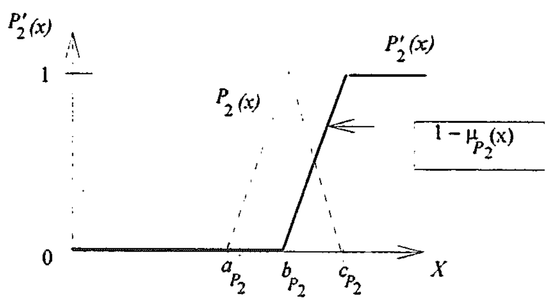

Figure 4: Fuzzy Half-inverse fuzzy set $\mu_{P'_2}(x)$

Taking Formula (2) into (1), we then have:

$$\mu_{sp_1}(x) = Min\,[\mu_{in\ p_1}(x),\ \mu_{not > in\ p_2}(x)\,] = Min\,[\mu_{p_1}(x),\ \mu_{p'_2}(x)]. \quad (3)$$

Similarly, we can get $\mu_{sp_2}(x)$ in p_2 as the start time for machine 2. The membership value $\mu_S(x)$ of x as the start time for machine 2 is then:

$$\mu_S(x) = \mu_{sp_1}(x) \vee \mu_{sp_2}(x) = Max\,[\mu_{sp_1}(x),\ \mu_{sp_2}(x)]. \quad (4)$$

According to the above derivation, the Find-Triangular-Longer-Time procedure can be designed as follows.

The find-triangular-longer-time procedure:

Input: Two triangular fuzzy sets with completion time p_1 and p_2 for each machine.

Output: The fuzzy start time s (a fuzzy set) for the next job to be executed on machine 2.

Step 1: Find the *half-inverse fuzzy set* p'_1, p'_2.

Step 2: Set

$$\mu_{sp_1}(x) = Min\ [\mu_{p_1}(x), \mu_{p'_2}(x)], \text{ and}$$

$$\mu_{sp_2}(x) = Min\ [\mu_{p'_1}(x), \mu_{p_2}(x)].$$

Step 3: Set $\mu_S(x) = Max\ [\mu_{sp_1}(x), \mu_{sp_2}(x)]$.

Step 4: Normalize $S(x)$. The function $S(x)$ is then output as the fuzzy start time for the next job to be executed on machine 2.

According to the above procedure, we can use linear equations and fuzzy operations to find the fuzzy start time for the next job to be executed on machine 2. Although this membership function can actually reflect the fuzzy start time for the next job to be executed on machine 2, its calculation is a little complicated. We hope to reduce the time-complexity and the membership function complexity. Below, we use a heuristic procedure to find the fuzzy start time for the next job to be executed on machine 2. We want to get three points to describe the new $S(x)$ with the b point representing the highest membership value.

There are totally 24 cases about p_1 and p_2 to find the fuzzy start time for the next job to be executed on machine 2 [2]. From an analysis of the 24 cases, we can design the following procedure.

The approximated find-triangular-longer-time procedure:

Input: Two triangular fuzzy sets with completion time $p_1\ (a_{p_1}, b_{p_1}, c_{p_1})$ and $p_2\ (a_{p_2}, b_{p_2}, c_{p_2})$ for each machine.

Output: The fuzzy start time $s\ (a_s, b_s, c_s)$ (a triangular fuzzy set) for the next job to be executed on machine 2.

Step 1: Set $a_s = max\ \{max(a_{p_1}, a_{p_2}), min(b_{p_1}, b_{p_2})\}$

Step 2: Set $c_s = max(c_{p_1}, c_{p_2})$

Step 3: Set $b_s =$

$$\begin{cases} \max\{b_{P_1}, b_{P_2}\}, \text{if } \max\{b_{P_1}, b_{P_2}\} \geq \min\{c_{P_1}, c_{P_2}\} \\ \dfrac{c_{P_1} \times c_{P_2} - b_{P_1} \times b_{P_2}}{(c_{P_1} + c_{P_2}) - (b_{P_1} + b_{P_2})}, \text{if } \max\{b_{P_1}, b_{P_2}\} < \min\{c_{P_1}, c_{P_2}\} \end{cases}$$

7. Conclusion

In this paper, triangular fuzzy set concepts were used with the Gupta algorithm to schedule jobs with uncertain time for a more-than-two-machine flow shop. Given a set of jobs, each having m (m>2) tasks that must be executed on each of m machines and their processing time membership functions, the triangular fuzzy Gupta algorithm yields a scheduling result with a membership function for the final completion time. The results can then help managers gain a broader overall view of scheduling. In the future, we will try to apply other characteristics of fuzzy sets to the scheduling field.

References

[1] B. G. Buchanan and E. H. Shortliffe, *Rule-Based Expert System: The MYCIN Experiments of the Standford Heuristic Programming Projects*, Addison-Wesley, MA., 1984.

[2] T. N. Chuang, *Application of Fuzzy Set theory to Industrial Scheduling*, Master Thesis, Kaohsiung Polytechnic Institute, Taiwan, R. O. C., 1996

[3] I. Gazdik, "Fuzzy-network planning," *IEEE Transactions on Reliability*, Vol. 32-3, 1983, 304-313.

[4] I. Graham and P. L. Jones, *Expert Systems - Knowledge, Uncertainty and Decision*, Chapman and Computing, Boston, 1988, 117-158.

[5] T. P. Hong, C. M Huang and K. M. Yu, "LPT scheduling for fuzzy tasks," accepted and to appear in *Fuzzy Sets and Systems*, 1997.

[6] T. P. Hong and T. N. Chuang, "Fuzzy scheduling on two-machine flow shop," accepted and to appear in *Journal of Intelligent & Fuzzy Systems: Applications in Engineering and Technology*, 1997.

[7] T. P. Hong and T. N. Chuang, "A new fuzzy Johnson algorithm for continuous fuzzy domain," *Seventh International Fuzzy Systems Association World Congress*, 1997, Czech Republic.

[8] C. M. Klein, "Fuzzy shortest paths," *Fuzzy Sets and Systems*, 1989, 27-41.

[9] G. J. Klir and T. A. Folger, *Fuzzy Sets, Uncertainty, and Information*, Prentice Hall, New Jersey, 1992, 4-14.

[10] S. McCahon and E. S. Lee, "Job sequencing with fuzzy processing times", *Computer Math. Applic.*, Vol. 19, No. 7, pp. 31-41, 1990.

[11] T. E. Morton and D. W. Pentico, *Heuristic Scheduling Systems with Applications to Production Systems and Project Management,* John Wiley & Sons Inc., New York, 1993.

[12] S. H. Nasution, "Fuzzy critical path method," *IEEE Transactions on Systems, Man, and Cybernetics,* Vol. 24-1, 1994, 48-57.

[13] D. S. Palmer, "Sequencing jobs through a multi-stage process in the minimum total time-a quick method of obtaining a near optimum," *Operational Research*, Vol. 16, 1965, 101-107.

[14] L. A. Zadeh, "Fuzzy logic," *IEEE Computer,* 1988, 83-93.

[15] H. J. Zimmermann, *Fuzzy Sets, Decision Making, and Expert Systems,* Kluwer Academic Publishers, Boston, 1987.

[16] H. J. Zimmermann, *Fuzzy Set Theory and Its Applications,* Kluwer Academic Publisher, Boston, 1991.