SCHEDULING STOCHASTIC JOBS ON PARALLEL MACHINES TO MINIMIZE MAKESPAN OR FLOWTIME

Richard R. Weber

Abstract

A number of identical machines operating in parallel are to be used to complete the processing of a collection of jobs so as to minimize the jobs' makespan or flowtime. The amount of processing required to complete the jobs have known probability distributions. It has been established by several researchers that when the required amounts of processing are all distributed as exponential random variables, then the strategy (LEPT) of always processing jobs with the longest expected processing times minimizes the expected value of the makespan, and the strategy (SEPT) of always processing jobs with the shortest expected processing times minimizes the expected value of the flowtime. We prove these results and describe a more general instance in which they are also true: when the jobs have received differing amounts of processing prior to the start, their total processing requirements are identically distributed, and the common distribution of total processing requirements has a monotone hazard rate. Under the stronger assumption that the distribution of the total processing requirements has a density whose logarithm is concave or convex, LEPT and SEPT minimize the makespan and flowtime in distribution.

1. Scheduling to Minimize Makespan or Flowtime

A number of identical machines operating in parallel are to be used to complete the processing of a collection of jobs so as to minimize the jobs' makespan or flowtime. Makespan and flowtime are two criteria commonly used to evaluate scheduling strategies. Suppose that there are n jobs and they they are completed at times $C_1,\ldots,C_n$. The <u>makespan</u> $\max\{C_i\}$ is the time at which the last job is completed. The <u>flowtime</u> ΣC_i is the sum of all the times at which jobs are completed. We suppose that preemptive scheduling is permitted, so that any job may be instantaneously removed from a machine and another job processed instead. A job's processing requirement is the length of time for which it must to be processed by a single machine in order to be completed. We show that when the jobs have processing requirements that are distributed as exponential random variables, then the optimal strategies have simple forms. The expected value of the makespan is minimize by a strategy (LEPT) of always processing jobs with the <u>longest expected processing times</u>. The expected value of the flowtime is minimized by a strategy (SEPT) of always processing jobs with the <u>shortest expected processing times</u>. We prove these results and describe a more general instance in which they are also true: when the jobs are identical but have received differing amounts of processing prior to the start and the distribution of the jobs' total processing requirements has a monotone hazard rate.

It is well known that the LEPT and SEPT strategies are optimal when jobs have known processing requirements. McNaughton (1959) has shown that amongst preemptive scheduling strategies the makespan is minimized by always processing those jobs with the greatest amounts of remaining processing, (so that either all jobs finish together, or the makespan is equal to the processing requirement of the longest job). The

flowtime is minimized by always processing those jobs with the least amounts of remaining processing (see Conway, Maxwell and Miller (1967) and Schrage (1968)).

In certain circumstances these results are true when the processing requirements are unknown. Several authors have investigated the case in which processing requirements are distributed as exponential random variables with differing means. Glazebrook (1976 and 1979) has shown that SEPT minimizes the expected value of the flowtime. Bruno (1976) has proved this for just two machines, and Weiss and Pinedo (1979) for any number. Bruno and Downey (1977) have shown that LEPT minimizes the expected value of the makespan for two machines. Bruno, Downey and Frederickson (1981) have shown that LEPT and SEPT are optimal for any number of machines, as has Van der Heyden (1981) for LEPT.

These authors have proved their results by using dynamic programming equations to examine the difference in the expected values of the makespan and flowtime resulting from two strategies which differ only in the jobs scheduled at the start. Complicated notations make the proofs difficult to follow and none can be generalized to processing requirements which are not exponentially distributed. In the following section we present a new proof of the optimalities of LEPT and SEPT in the case of exponentially distributed processing requirements in order to illustrate the method of proof which is used in Weber (1982) to generalize these results to other distributions of processing requirements. Restricting attention to exponentially distributed processing requirements is the best way to put across the flavour of the method. It should become clear that the results may be generalized in several directions. These are discussed in the final section.

2. Proofs When Processing Requirements have Exponential Distributions

Suppose that n jobs are to be completed on m identical machines operating in parallel. All jobs are available for processing from the start. They have processing requirements which are distributed as exponential random variables with means $1/\lambda_1, 1/\lambda_2, \ldots, 1/\lambda_n$.

Theorem 1a. LEPT minimizes the expected value of the makespan.

Proof. Notation will be simplified, but the proof sufficiently illustrated, if we carry it through for the case of just two machines. The proof is essentially the same when there are more than two machines, but the notation is more complicated. The proof is by induction on the number of jobs to be processed. Suppose that the theorem is true whenever there are less than n jobs to process. We will show that it is true when the number of jobs to be processed is n. It is clear that the optimal strategy is non-preemptive, and that if it is optimal to process jobs i and j at time t then it is optimal to continue to process them until one of them is completed. Suppose $\lambda_1<\lambda_2<\cdots<\lambda_n$. Let U^I denote the expected value of the remaining time needed to ensure that all n jobs are completed, given that an optimal strategy is employed and that the jobs in the list $I \equiv i_1,\ldots,i_\ell$ have already been completed. Let V^I denote the same quantity when the strategy used to complete the jobs is LEPT. By the inductive hypothesis LEPT is optimal once one job has been completed and so $U^I = V^I$ when I is a list of at least one job. By conditioning on the event that occurs at the first job completion we obtain

$$U = \min_{i\neq j}\{(1 + \lambda_i V^i + \lambda_j V^j)/(\lambda_i + \lambda_j)\}. \quad (1)$$

It is simple to check that (1) is equivalent to

$$0 = \min_{i\neq j}\{1 + \lambda_i(V^i-V) + \lambda_j(V^j-V) + (\lambda_i+\lambda_j)(V-U)\}. \quad (2)$$

In (1) and (2) the minimums are attained by the same pair $\{i,j\}$. Since λ_1 and λ_2 are the two smallest values of λ_1 and $V \geq U$, the fourth term on the right hand side of (2) is minimized by $\{i,j\} = \{1,2\}$. Hence to show that LEPT is optimal it is sufficient to show that $\{i,j\} = \{1,2\}$ also minimizes the sum of the second and third terms. We define

$$V_i = \lambda_i(V^i-V) \quad \text{and} \quad D_{ij} = V_i - V_j.$$

It is interesting to note that $\delta V_i + o(\delta)$ may be interpreted as the amount by which the expected value of the makespan would change from V if when employing LEPT we were to give job i an extra amount δ of processing just before the start. It is the result of theorem 1b that if $\lambda_i < \lambda_j$ then D_{ij} is less than or equal to zero. Hence the sum of the second and third terms on the right hand side of (2), $V_i + V_j$, is minimized by $\{i,j\} = \{1,2\}$ and the induction is complete.

Throughout the rest of this section we shall consider V, V_i and D_{ij} as functions of the variables $\lambda_1,\ldots,\lambda_n$. V_i^I and D_{ij}^I are defined similarly to V_i and D_{ij} when $i,j \notin I$. For example, V_i^I is $\lambda_i(V^{Ii} - V^I)$, where Ii denotes the list I with job i appended.

Theorem 1b. Suppose $\lambda_i < \lambda_j$, $\lambda_1<\cdots<\lambda_n$. Then

$$D_{ij} \leq 0, \quad (3)$$

and

$$dD_{12}/d\lambda_1 \geq 0. \quad (4)$$

Proof. The proof is by induction on n, the number of jobs to be processed. When $n=2$, $D_{ij} = (\lambda_i/\lambda_j) - (\lambda_j/\lambda_i)$ and the theorem is true trivially. If i and j are the two smallest indices not in the list I then jobs i and j will be processed first. Conditioning on the event that occurs at the first job completion we have $(\lambda_i+\lambda_j)V^I = 1 + \lambda_i V^{Ii} + \lambda_j V^{Ij}$. By using this fact, and the definition of V_i^I, we can derive the following identities.

$$(\lambda_1+\lambda_2+\lambda_3)V_1 = \lambda_1(\lambda_1+\lambda_2+\lambda_3)V^1 - \lambda_1(\lambda_1+\lambda_2+\lambda_3)V$$

$$= \lambda_1(1+\lambda_1 V^1+\lambda_2 V^{12}+\lambda_3 V^{13}) - \lambda_1(1+\lambda_1 V^1+\lambda_2 V^2+\lambda_3 V)$$

$$= \lambda_1(\lambda_3 V^{13}-\lambda_3 V^1) + \lambda_2(\lambda_1 V^{12}-\lambda_1 V^2) + \lambda_3 V_1$$

$$(\lambda_1+\lambda_2)V_1 = \lambda_1 V_3^1 + \lambda_2 V_1^2.$$

We can establish the following similarly.

$$(\lambda_1+\lambda_2)V_2 = \lambda_1 V_2^1 + \lambda_2 V_3^2.$$

$$(\lambda_1+\lambda_2)V_i = \lambda_1 V_i^1 + \lambda_2 V_i^2, \quad i = 3,\ldots,n.$$

Combining these we have,

$$D_{12} = \frac{\lambda_1}{\lambda_1+\lambda_2} D_{32}^1 + \frac{\lambda_2}{\lambda_1+\lambda_2} D_{13}^2, \tag{5}$$

and

$$D_{2i} = \frac{\lambda_1}{\lambda_1+\lambda_2} D_{2i}^1 + \frac{\lambda_2}{\lambda_1+\lambda_2} D_{3i}^2, \quad i = 3,\ldots,n. \tag{6}$$

The inductive hypothesis states that (3) and (4) are true when there are less than n jobs to complete, and this hypothesis for (3) implies that both $D_{13}^2 \le 0$ and $D_{23}^1 \le 0$ are true when there are n jobs to complete. The hypothesis for (4) similarly implies that $dD_{13}^2/d\lambda_1 \ge 0$. By integrating this with respect to λ_1 we have $D_{13}^2 \le D_{23}^1 = -D_{32}^1 \le 0$, and thus remembering that λ_1 is less than λ_2 we can check that (5) is nonpositive. The inductive hypothesis also implies that D_{2i}^1 and D_{3i}^2 are nonpositive, and thus (6) is nonpositive. Since (5) and (6) have been shown to be nonpositive this establishes the inductive step for (3). The inductive step for (4) is established by differentiating the right hand side of (5) with respect to λ_1 and then using the inductive hypothesis to check that every term is nonnegative.

<u>Theorem 2a</u>. SEPT minimizes the expected value of the flowtime.

Proof. The proof is similar to that of theorem 1a. We suppose that $\lambda_i > \cdots > \lambda_n$, and redefine U and V in terms of the expected value of the flowtime. Equation (2) becomes

$$0 = \min_{i \ne j}\{n + \lambda_i(V^i-V) + \lambda_j(V^j-V) + (\lambda_i+\lambda_j)(V-U)\}.$$

The proof is completed using theorem 2b along the same lines as theorem 1a.

<u>Theorem 2b</u>. Suppose $\lambda_j > \lambda_i$, $\lambda_1 > \cdots > \lambda_n$. Then

$$-1 \le D_{ij} \le 0, \tag{7}$$

and

$$dD_{12}/d\lambda_1 \le 0. \tag{8}$$

Proof. The proof is by induction and similar to that of theorem 1b. When $n = 2$, $D_{12} = 0$ and the theorem is true trivially. Instead of (4) and (5) we get

$$D_{12} = \frac{\lambda_1}{\lambda_1+\lambda_2}(D_{32}^1-1) + \frac{\lambda_2}{\lambda_1+\lambda_2}(D_{13}^2+1), \tag{9}$$

and

$$D_{2i} = \frac{\lambda_1}{\lambda_1+\lambda_2} D_{2i}^1 + \frac{\lambda_2}{\lambda_1+\lambda_2}(D_{3i}^2-1), \quad i = 3,\ldots,n. \tag{10}$$

Using (9) and (10) and the inductive hypotheses it is now easy to check the inductive steps for (7) and (8).

3. Generalizations

There are a number of directions in which the results of the previous section may be generalized by using proofs similar to those of theorem 1a and 1b. We shall not give the details of the proofs here, but refer the reader to Weber (1980 and 1982), where they are proved within a framework of optimal control theory and continuous dynamic programming.

(a) By redefining U and V it can be shown that LEPT and SEPT minimize the makespan and flowtime in distribution. This means that, for any γ, the probabilities that the makespan and flowtime are greater than γ are minimized by LEPT and SEPT respectively. It is no longer obvious that the optimal strategy is non-preemptive, but this can be established in the proof.

(b) As Weiss and Pinedo (1979), we may consider non-identical machines which process jobs at rates $s_1 \geq \cdots \geq s_m$. If job i is processed on machine j its instantaneous hazard rate is $\lambda_i s_j$. The expected value of the makespan is now minimized by a version of LEPT which always processes the job of least λ on machine 1, the job of second least λ on machine 2, and so on. The expected valued of the flowtime is minimized by a version of SEPT which always processes the job of greatest λ on machine 1, the job of second greatest λ on machine 2, and so on. The proof is along the lines of the previous section.

(c) For processing requirements that are not distributed as exponential random variables, LEPT and SEPT are still optimal in the following circumstance. Assume the jobs have identical total processing requirements, but have received different amounts of processing prior to the start. A job that has received an amount of processing x has an instantaneous hazard rate of $\rho(x)$. Suppose that $\rho(x)$ is a <u>monotone hazard rate</u> that is increasing or decreasing in x. In these circumstances LEPT and SEPT are still optimal. The proof is along the lines of the previous section. For example, (4) is reformulated in terms of dD_{12}/dx_1, where x_1 is the amount of processing job 1 has so far received. [Note that if several jobs have received equal amounts of

processing, LEPT or SEPT may require a 'sharing' of machine effort. For example, if the hazard rate is increasing and exactly three jobs, which have had equal amounts of processing, are to be completed on 2 machines, then LEPT is realized by processing each job at 2/3 the full rate of one machine until one job is completed. In practice, sharing is approximated by frequently changing the set of jobs that are being processed, so that the amounts of processing the jobs have received remain nearly equal.]

This result includes the result of Pinedo and Weiss (1979) who proved that LEPT and SEPT are expected value optimal when the processing requirements are distributed as different mixtures of two exponential distributions. Their model corresponds to a decreasing hazard rate model of the above form. The result also extends the work of Nash (1973) who proved the optimality of SEPT for the case in which all jobs have received identical amounts of processing at the start and the distribution of total processing requirement has an increasing hazard rate.

(d) If in addition to the conditions of (c) the distribution of total processing requirement has a density whose logarithm is concave or convex, then LEPT and SEPT minimize the makespan and flowtime in distribution. A density whose logarithm is concave or convex is said to be <u>sign-consistent of order two</u> (SC_2). Karlin (1968) has made a detailed study of sign-consistent densities. He and other authours have described their importance in areas of statistical theory, reliability, game theory and mathematical economics. The uniform, gamma, hyperexponential, folded-normal and Weibull distributions all have SC_2 densities. The proof is along the same lines as above, with, for example, (4) reformulated in terms of $d\{D_{12}/\rho(x_1)\}/dx_1$.

(e) When the processing requirements are distributed according to any of the models of this paper, the results are still true even if the number of available machines is a non-decreasing function of time. This follows from the fact that in order to carry out the induction in theorem 1b we only needed to be sure that in (5) job 3 could take the role of jobs 1 or 2 in an application of the inductive hypothesis for (4). This will be the case if the number of machines is non-decreasing in time. A stronger result can be proved if the distribution of total processing requirement has a SC_2 density. In this case LEPT minimizes the makespan in distribution even if the number of available machines is an arbitrary function of time and some jobs are not present at the start, but only arrive later according to a stochastic process.

(f) For any of the models of this paper, LEPT also maximizes in distribution the time at which the number of machines first exceeds the number of uncompleted jobs. Thus, if a system requires m components to operate, the length of time for which it can be kept running by using a stock of $n > m$ components is maximized in distribution by LEPT. When $m = 2$ this is the 'lady's nylon stocking problem' of Cox (1959), for which he hypothesised LEPT optimality in the case of monotone hazard rates. Weber and Nash (1979) give further details.

It does not appear possible to find simple strategies minimizing the expected values of the makespan or flowtime if the processing requirements are not distributed according to one of the above models. When the distribution of total processing requirement has a non-monotone hazard rate LEPT and SEPT are generally not optimal.

4. References

[1] Bruno, J. (1976), "Sequencing Tasks with Exponential Service Times on Parallel Machines," Technical report, Department of Computer Science, Pennsylvania State University.

[2] Bruno, J. and Downey, P. (1977), "Sequencing Tasks with Exponential Service Times on Parallel Machines," Technical report, Department of Computer Sciences, University of California at Santa Barbara.

[3] Bruno, J., Downey, P. and Frederickson, G. N. (1981), "Sequencing Tasks with Exponential Service Times to Minimize the Expected Flowtime or Makespan," J. Assoc. Comput. Mach., 28, 100-113.

[4] Conway, R. W., Maxwell, W. L. and Miller, L. W. (1967), The Theory of Scheduling, Addison-Wesley, Reading, Massachusetts.

[5] Cox, D. R. (1959), "A Renewal Problem with Bulk Ordering of Components," J. Roy. Statist. Soc. Ser. B, 21, 180-189.

[6] Glazebrook, K. D. (1976), "Stochastic Scheduling," Ph. D. Thesis, University of Cambridge.

[7] Glazebrook, K. D. (1979), "Scheduling Tasks with Exponential Service Times on Parallel Processors," J. Appl. Probab., 16, 685-689.

[8] Karlin, S. (1968), Total Positivity, Vol. I., Stanford University Press, Stanford.

[9] McNaughton, R. (1959), "Scheduling with Deadline and Loss Functions," Management Sci., 6, 1-12.

[10] Nash, P. (1973), "Optimal Allocation of Resources to Research Projects," Ph. D. Thesis, University of Cambridge.

[11] Pinedo, M. and Weiss, G. (1979), "Scheduling Stochastic Tasks on Two Parallel Processors," Naval Res. Logist. Quart., 27, 528-536.

[12] Schrage, L. E. (1968), "A Proof of the Shortest Remaining Process Time Discipline," Oper. Res., 16, 687-689.

[13] Van der Heyden, J. (1981), "Scheduling Jobs with Exponential Processing and Arrival Times on Identical Processors so as to Minimize the Expected Makespan," Math. Oper. Res., 6, 305-312.

[14] Weber, R. R. and Nash, P. (1979), "An Optimal Strategy in Multi-Server Stochastic Scheduling," J. Roy. Statist. Soc. Ser. B, 40, 323-328.

[15] Weber, R. R. [illegible], "Optimal Organization of Multi-Server Systems," Ph. D. Thesis, University of Cambridge.

[16] Weber, R. R. (1982), "Scheduling Jobs with Stochastic Processing Requirements on Parallel Machines to Minimize Makespan and Flowtime," J. Appl. Probab., 19, (to appear).

[17] Weiss, G. and Pinedo, M. (1979), "Scheduling Tasks with Exponential Service Times on Non-Identical Processors to Minimize Various Cost Functions, J. Appl. Probab., 17, 187-202.

Cambridge University, Engineering Department, Control and Management Systems Division, Mill Lane, Cambridge CB2 1RX, ENGLAND.

Discussant's Report on "Scheduling Stochastic Jobs on Parallel Machines," by Richard Weber

The problem of optimally sequencing a collection of independent jobs with known processing times on parallel machines has a long history in the deterministic scheduling literature [3]. The simplest of these results, regarding the "shortest job first" policy for flowtime [4] and the "longest remnant first" policy for preemptive makespan [10], are so well known that they have become pedagogical "standards", used to convince students that there is something to all this business of scheduling after all.

A few years ago, investigators began to consider whether similar optimal policies exist for jobs whose runtime is described by a random variable. Perhaps forgivably, they first attacked that noble distribution that seems to occur exponentially in the literature [6, 12, 14, 1], allowing for differing mean times for different jobs. The deterministic results carried over (with the adjective "expected" appropriately inserted), but was this merely a trick of our (noble but forgetful) distribution? Pinedo and Weiss [12] were able to push the result to the case of two machines where each job's time is chosen as a different hyperexponential "mix" of the same two fixed exponentials. Nash [11] showed that if jobs shared a common distribution with increasing failure rate function then "shortest expected processing time first" minimized the expected flowtime.

Enter Richard Weber. He has shown that these classic scheduling policies are optimal for fundamental reasons having little to do with the detailed structure of the service time distribution of jobs. Suppose that the service time of each job is chosen independently from the same underlying distribution, but different jobs have "aged" different amounts at the start of scheduling (so that they begin at different points in the failure rate curve for the underlying distribution). All the above problems can be reinterpreted from this new point of view. Weber's discovery is that, assuming scheduling is preemptive, that processor-sharing is allowed, and that the failure rate curve is monotone increasing or decreasing throughout, then "shortest expected processing time first" (SEPT) minimizes expected flowtime and "longest expected processing time first" (LEPT) minimizes expected makespan [13]. From these results, the SEPT policy is reinterpreted as that policy which schedules the job with largest instantaneous failure rate; LEPT schedules the job with smallest instantaneous failure rate; ties must share processors equally.

More is true. LEPT and SEPT are policies with are optimal in distribution (not just in expectation) if the more stringent (but still mild) assumption is made that the processing time density is "sign-regular of order 2" [8]. Such an assumption implies that the failure rate curve is monotone [9].

Thus we have in Weber's work the pleasure of seeing several distinct results suddenly pieced together in a novel way, be shown to be "cases" of a fundamental result, and to obtain a vast generalization of the class of distributions to which the result is applicable. Frankly, results this satisfying are few and far between, and worth the wait when they arrive.

The paper in the current proceedings provides a clear exposition of Weber's proof technique for a particular distribution (the exponential). Shorn of the details necessary to obtain the results of greatest generality, this paper provides a useful introduction to the more elaborate proofs in [13].

Of course, the settling of a question merely causes the bystanders to suggest new questions, and so I will do what is required. There are

some possible directions for investigation suggested by both the literature on this problem and by "discrete analogues" from deterministic scheduling theory.

One direction was pursued by Weiss and Pinedo [14] for the exponential distribution. They define a large class of cost functions for schedules (of which makespan and flowtime are but special cases) and show conditions under which LEPT and SEPT minimize expected cost. Does their theorem go through in the more general setting of aged jobs from a monotone failure rate distribution?

So far we have spoken only of scheduling with preemptions. When the failure rate curve is decreasing, then LEPT is nonpreemptive; similarly for SEPT when the failure rate curve is increasing. In these two cases the results yield simple optimal policies for nonpreemptive schedules. From deterministic scheduling results, we know that sequencing a set of jobs with known processing times on two machines to minimize makespan is an NP-complete problem (for three machines it is "strong" NP-complete [5]). It is likely then that no optimal policy exists which is simple enough to be useful.

In the terms of Weber's paper [13], a set of deterministic jobs with processing times $t_1 < t_2 < \cdots < t_n$ can be thought of as "aged" [illegible] with a unit step at t_n. Thus we are trying to minimize makespan when failure rates are increasing while disallowing preemption and processor sharing. This shows that the current results cannot be extended to the nonpreemptive case for all monotone failure rate distributions. But is there a "reasonable" class of distributions for which a "simple" nonpreemptive scheduling rule is optimal?

Another direction of generalization is to introduce precedence constraints among the (still stochastically independent) jobs. On the deterministic side, Hu [7] showed that unit processing time jobs constrained to be sequenced according to an "in-tree" form of partial order (Figure 1) can be finished in the shortest time on m processors if scheduled "highest level first" (HLF) among all those remaining (the levels of jobs are shown in Figure 1). Chandy and Reynolds [2] have shown that if job processing times are random variables with a common exponential or Erlang distribution, and jobs are constrainted according to an "in-tree", then HLF is still optimal to minimize expected makespan on two processors. The result does not extend to three processors: in Figure 1, HLF schedules jobs 1, 2 and 3 first while the optimal policy must pick 1, 2 and 4 first, assuming processing times are exponential [2].

The question in all this is: does the Chandy and Reynolds result carry over for a more general class of distribution? Why does the HLF policy break down for three processors? What, if any, is an optimal policy?

In many scheduling applications (e.g., computer job scheduling), one has mixtures of some short jobs and some long ones with few of middling length. Such a distribution causes the failure rate curve to be "basin" shaped--at first decreasing and then increasing. Based on the insights gained from this paper, what can be said about the relative performance of SEPT against the optimal policy minimizing flowtime? In this context SEPT is a suboptimal "heuristic", but commonly employed with good results.

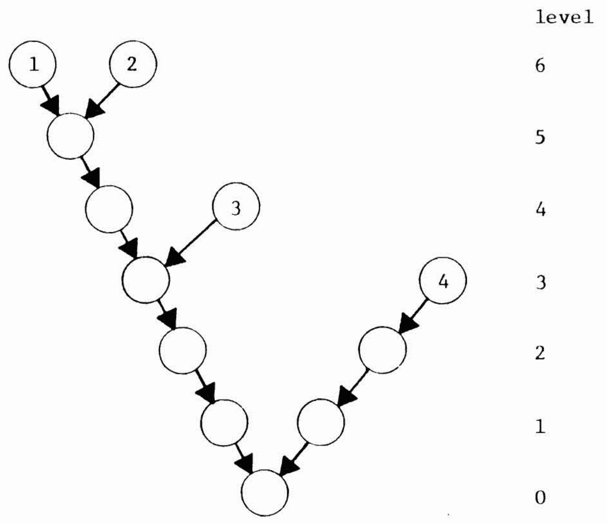

Figure 1. "In-tree" procedence constraint showing levels.

References

[1] Bruno, J., P. Downey and G. N. Frederickson (1981), "Sequencing Tasks with Exponential Service Times to Minimize the Expected Flow Time or Makespan," J. Assoc. Comput. Mach., 28, pp. 100-113.

[2] Chandy, K. M. and P. F. Reynolds (1975), "Scheduling Partially Ordered Tasks with Probabilistic Execution Times," Proceedings of the 5th Symposium on Operating Systems Principles, ACM, 169-177.

[3] Coffman, E. G., Jr. (ed.) (1976), Computer and Job-Shop Scheduling Theory, John Wiley and Sons, New York.

[4] Conway, R. W., W. L. Maxwell and L. W. Miller (1967), Theory of Scheduling, Addison-Wesley Publishing Co., Reading, MA.

[5] Garey, M. R. and D. S. Johnson (1979), Computers and Intractability: a Guide to the Theory of NP-Completeness, W. H. Freeman and Co., San Francisco, CA.

[6] Glazebrook, K. D. (1976), "Stochastic Scheduling," Ph.D. Thesis, Cambridge University.

[7] Hu, T. C. (1961), "Parallel Sequencing and Assembly Line Problems," Oper. Res., 9, pp. 841-848.

[8] Karlin, S. (1968), "Total Positivity, Vol. 1," Stanford University Press, Stanford, CA.

[9] Kaufmann, A., D. Grouchko and R. Crunon (1977), Mathematical Models For the Study of the Reliability of Systems, Academic Press, New York.

[10] McNaughton, R. (1959), "Scheduling with Deadlines and Loss Functions, Management Sci., 16, pp. 1-12.

[11] Nash, P. (1973), "Optimal Allocation of Resources to Research Projects," Ph.D. Thesis, Cambridge University.

[12] Pinedo, M. and G. Weiss (1979), "Scheduling Stochastic Tasks on Two Parallel Processors," Naval Res. Logist. Quart., 26, pp. 527-535.

[13] Weber, R. (1981), "Scheduling Jobs with Stochastic Processing Requirements on Parallel Machines to Minimize Makespan or Flowtime," J. Appl. Probab., 18, to appear.

[14] Weiss, G. and M. Pinedo (1980), "Scheduling Tasks with Exponential Service Times on Non-Identical Processors to Minimize Various Cost Functions, J. Appl. Probab., 17, pp. 187-202.

Discussant: Dr. Peter J. Downey, Department of Computer Science, The University of Arizona, Tucson, Arizona 85721.

At

ho
hi
on
so
pr
th
cl
di
re
tin
hea
hav
thr
alw

1.