Technische Universität Berlin
FACHBEREICH 3
MATHEMATIK

Scheduling under Uncertainty: Optimizing against a Randomizing Adversary

by

Rolf H. Möhring

No. 681/2000

Scheduling under Uncertainty: Optimizing against a Randomizing Adversary

Rolf H. Möhring *

Technische Universität Berlin, 10623 Berlin, Germany,
moehring@math.tu-berlin.de,
WWW home page: http://www.math.tu-berlin.de/~moehring/

Abstract Deterministic models for project scheduling and control suffer from the fact that they assume complete information and neglect random influences that occur during project execution. A typical consequence is the underestimation of the expected project duration and cost frequently observed in practice. To cope with these phenomena, we consider scheduling models in which processing times are random but precedence and resource constraints are fixed. Scheduling is done by policies which consist of an an online process of decisions that are based on the observed past and the a priori knowledge of the distribution of processing times. We give an informal survey on different classes of policies and show that suitable combinatorial properties of such policies give insights into optimality, computational methods, and their approximation behavior. In particular, we present recent constant-factor approximation algorithms for simple policies in machine scheduling that are based on a suitable polyhedral relaxation of the performance space of policies.

1 Uncertainty in scheduling

In real-life projects, it usually does not suffice to find good schedules for fixed deterministic processing times, since these times mostly are only rough estimates and subject to unpredictable changes due to unforeseen events such as weather conditions, obstruction of resource usage, delay of jobs and others.

In order to model such influences, the processing time of a job $j \in V$ is assumed to be a random variable $\mathbf{p}_j$. Then $\mathbf{p} = (\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n)$ denotes the (random) vector of processing times, which is distributed according to a joint probability distribution Q. This distribution Q is assumed to be known and may also contain stochastic dependencies. Furthermore, like in deterministic models, we have precedence constraints given by a directed acyclic graph $G = (V, E)$ and resource constraints. In the classification scheme of [1], these problems are denoted by $PS \mid prec, p_j = sto \mid \kappa$, where κ is the objective (e.g. the project makespan $C_{\max}$ or the sum of weighted completion times $\sum w_j C_j$).

* Supported by Deutsche Forschungsgemeinschaft under grant Mo 346/3-3 and by German Israeli Foundation under grant I-564-246.06/97

The necessity to deal with uncertainty in project planning becomes obvious if one compares the "deterministic makespan" $C_{\max}(E(\mathbf{p}_1),\ldots,E(\mathbf{p}_n))$ obtained from the expected processing times $E(\mathbf{p}_j)$ with the expected makespan $E(C_{\max}(\mathbf{p}))$. Even in the absence of resource constraints, there is a systematic underestimation $C_{\max}(E(\mathbf{p}_1),\ldots,E(\mathbf{p}_n)) \leq E(C_{\max}(\mathbf{p}_1,\ldots,\mathbf{p}_n))$ which may become arbitrarily large with increasing number of jobs or increasing variances of the processing times [7]. Equality holds if and only if there is one path that is the longest with probability 1. This systematic underestimation of the expected makespan has already been observed by Fulkerson [2]. The error becomes even worse if one compares the deterministic value $C_{\max}(E(\mathbf{p}_1),\ldots,E(\mathbf{p}_n))$ with quantiles t_q such that $Prob\{C_{\max}(\mathbf{p}) \leq t_q\} \geq q$ for large values of q (say $q = 0.9$ or 0.95).

A simple example is given in Figure 1 for a project with n parallel jobs that are independent and uniformly distributed on [0,2]. Then the deterministic makespan $C_{\max}(E(\mathbf{p}_1),\ldots,E(\mathbf{p}_n)) = 1$, while $Prob(C_{\max} \leq 1) \to 0$ for $n \to \infty$. Similarly, all quantiles $t_q \to 2$ for $n \to \infty$ (and $q > 0$).

This is the reason why good practical planning tools should incorporate stochastic methods.

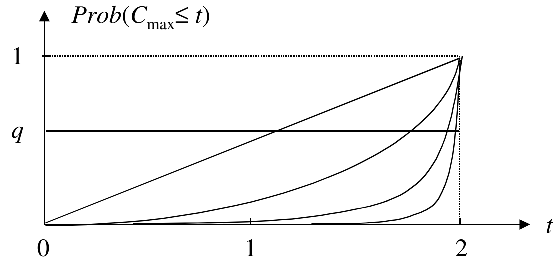


Figure 1. Distribution function of the makespan for $n = 1, 2, 4, 8$ parallel jobs that are independent and uniformly distributed on [0,2].

2 Planning with policies

If the problem involves only precedence constraints, every job can be scheduled at its earliest start, i.e., when its last predecessor completes. This is no longer possible when resource constraints are present. Planning is then done by *policies* or *strategies* that dynamically make scheduling decisions based on the observed past and the a priori knowledge about the processing time distributions. This can be seen as a special stochastic dynamic optimization problem or as an *online* algorithm against a "randomizing" adversary who draws job processing times according to a known distribution.

This model is somewhat related to certain online scenarios, which recently have received quite some attention. These scenarios are also based on the assumption that the scheduler does not have access to the whole instance at once, but rather learns the input piece by piece over time and has to make decisions based on partial knowledge only. When carried to an extreme, there is both a lack of knowledge on jobs arriving in the future and the running time of every job is unknown until it completes. In these models, online algorithms are usually analyzed with respect to optimum off-line solutions, whereas here we compare ourselves with the best possible policy which is subject to the same uncertainty. Note that our model is also more moderate than online scheduling in the sense that the number of jobs to be scheduled as well as their joint processing time distribution are known in advance. We refer to [16] for an overview on the other online scheduling models.

Our model with random processing times has been studied in machine scheduling, but much less in project scheduling. The survey [9] has stayed representative for most of the work until the mid 90ties.

A policy Π takes actions at *decision points*, which are $t = 0$ (project start), job completions, and tentative decision times where information becomes available. An action at time t consists in choosing a *feasible set* of jobs to be started at t, where feasible means that precedence and resource constraints are respected, and in choosing the next tentative decision time t^{planned}. The actual next decision time is the minimum of t^{planned} and the first job completion after t.

The decision which action to take may of course only exploit information of the past up to time t and the given distribution Q (*non-anticipative* character of policies). After every job has been scheduled, we have a realization $p = (p_1, \ldots, p_n)$ of processing times and Π has constructed a schedule $\Pi[p] = (S_1, S_2, \ldots, S_n)$ of starting times S_j for the jobs j. If $\kappa^{\Pi}(p)$ denotes the "cost" of that schedule, and $E(\kappa^{\Pi}(\mathbf{p}))$ the expected cost under policy Π, the aim then is to find a policy that minimizes the expected cost (e.g., the expected makespan).

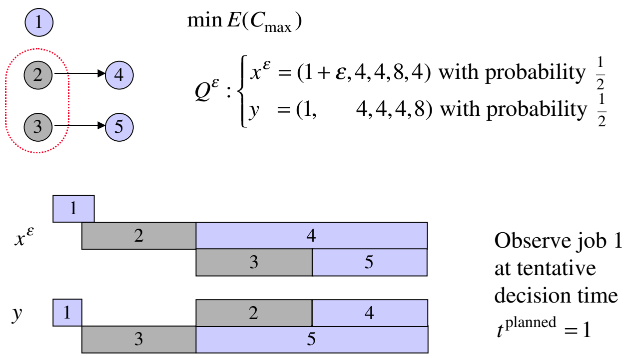


Figure 2. Optimal policies may involve tentative decision times.

As an example, consider the problem in Figure 2. The precedence constraints are given by the digraph in the upper left corner. The two encircled jobs 2 and 3 compete for the same scarce resource and may not be scheduled simultaneously.

There are two possible realizations x^ε and y that occur with probability $\frac{1}{2}$ each. The aim is to minimize the expected makespan. If one starts jobs 1 and 2, say, at time 0, one achieves only an expected makespan of 14. Observing job 1 at the tentative decision time $t^{\text{planned}} = 1$ yields 13 instead. The example shows in particular that jobs may start at times where no other job ends.

In general, there need not exist an optimal policy. This can only be guaranteed under assumptions on the cost function κ (e.g. continuity) or the distribution Q (e.g. finite discrete or with a Lebesgue density), see [9] for more details.

Stability issues constitute an important reason for considering only restricted classes of policies. Data deficiencies and the use of approximate methods (e.g. simulation) require that the optimum expected cost $OPT(\bar{\kappa}, \bar{Q})$ for an "approximate" cost function $\bar{\kappa}$ and distribution $\bar{Q}$ is "close" to the optimum expected cost $OPT(\kappa, Q)$ when $\bar{\kappa}$ and $\bar{Q}$ are "close" to κ and Q, respectively. (This can be made precise by considering uniform convergence $\bar{\kappa} \to \kappa$ of cost functions and weak convergence $\bar{Q} \to Q$ of probability distributions.)

Unfortunately, the class of all policies is unstable. The above example illustrates why. Consider Q^ε as an approximation to $Q = \lim_{\varepsilon \to 0} Q^\varepsilon$. For Q, one is no longer able to obtain information by observing job 1, and thus only achieves an average makespan of 14. Figure 3 illustrates this.

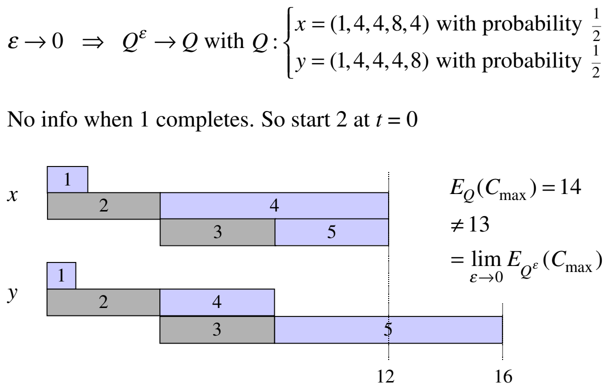


Figure 3. The class of all policies is unstable.

The main reason for this instability is the fact that policies may use small, almost "not observable" pieces of information for their decision. This can be overcome by restricting to *robust* policies. These are policies that start jobs only at completion of other jobs (no tentative decision times) and use only "robust" information from the past, viz. only the fact whether a job is completed, busy, or not yet started.

3 Robust classes of policies

Policies may be classified by their way of resolving the "essential" resource conflicts. These conflicts can be modeled by looking at the set $\mathcal{F}$ of *forbidden sets* of jobs. Every proper subset $F' \subset F$ of such a set $F \in \mathcal{F}$ can in principle be scheduled simultaneously, but the set F itself cannot because of the limited resources. In the above example, $\mathcal{F} = \{\{2, 3\}\}$. For a scheduling problem with m identical machines, $\mathcal{F}$ consists of all $(m + 1)$-element independent sets of the digraph G of precedence constraints.

3.1 Priority policies

A well-known class of robust policies is the class of *priority policies*. They settle the resource conflicts by a priority list L, i.e., at every decision time, they start as many jobs as possible in the order of L. Though simple and easy to implement, they exhibit a rather unsatisfactory stability behavior (Graham anomalies). Let us view a policy Π as a function $\Pi : \mathbb{R}^n \to \mathbb{R}^n$ that maps every vector $p = (p_1, \ldots, p_n)$ of processing times to a schedule $\Pi[p] = (S_1, S_2, \ldots, S_n)$ of starting times S_j for the jobs j. In this interpretation as a function, priority policies are in general neither continuous nor monotone. This is illustrated in Figure 4 on one of Graham's examples [5]. When p changes continuously and monotonously from y into x, $\Pi[p]_7 = S_7$ jumps discontinuously and $\Pi[p]_5 = S_5$ decreases while p grows.

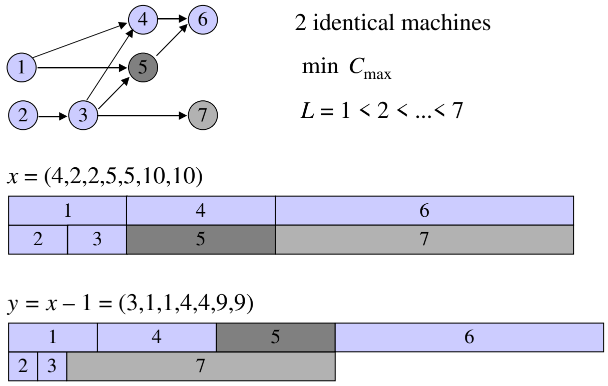


Figure 4. Priority policies are neither continuous nor monotone.

3.2 Preselective policies

Policies with a much better stability behavior are the *preselective policies* introduced in [8]. They solve every resource conflict given by a forbidden set $F \in \mathcal{F}$

by choosing a priori a *waiting job* $j_F \in F$ that can only start after at least one job $j \in F \setminus \{j_F\}$ has completed. This defines a *disjunctive waiting condition* $(F \setminus j_F, j_F)$ for every forbidden set $F \in \mathcal{F}$. A preselective policy then does early start scheduling w.r.t. the precedence constraints and the system $\mathcal{W}$ of waiting conditions obtained from choosing waiting jobs for the forbidden sets. The same idea is known in deterministic scheduling as *delaying alternatives*, see e. g. [1, Section 3.1].

A very useful combinatorial model for preselective policies has been introduced in [13]. Waiting conditions and ordinary precedence constraints are modeled by an AND/OR *graph* that contains AND-nodes for the ordinary precedence constraints and OR-nodes for the disjunctive precedence constraints. Figure 5 shows how.

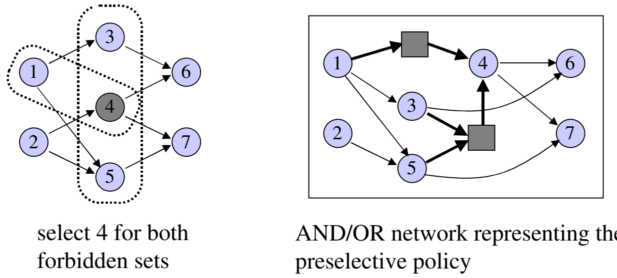


Figure 5. AND/OR graph induced by a preselective policy.

Since preselective policies do early start scheduling, it follows that the start time of a job j is the minimum length of a longest path to node j in the AND/OR graph, where the minimum is taken over the different alternatives in the OR-nodes. As a consequence, preselective policies are continuous and monotone (in the function interpretation) and thus avoid the Graham anomalies (see Figure 6). Surprisingly, also the reverse is true, i.e. every continuous robust policy is preselective and every monotone policy Π is dominated by a preselective policy, i.e., there is a preselective policy Π' with $\Pi' \leq \Pi$ [15]. This implies in particular that Graham anomalies come in pairs (discontinuous *and* not monotone).

There are several interesting and natural questions related to AND/OR graphs (or preselective policies). One is *feasibility*, since AND/OR graph may contain cycles. Here feasibility means that all jobs $j \in V$ can be arranged is a linear list L such that all waiting conditions given by the AND/OR graph are satisfied (all AND predecessors of a job j occur before j in L, and at least one OR predecessor occurs before j in L). Another question is *transitivity*, i.e., is a new waiting condition "j waits for at least one job from $V' \subseteq V$ " implied by the given ones? Or *transitive reduction*, i.e., is there a unique "minimal" AND/OR graph that is equivalent to a given one (in the sense that they admit the same linear lists)? All these questions have been addressed in [13]. Feasibility can be detected in linear time. A variant of feasibility checking computes transitively forced waiting

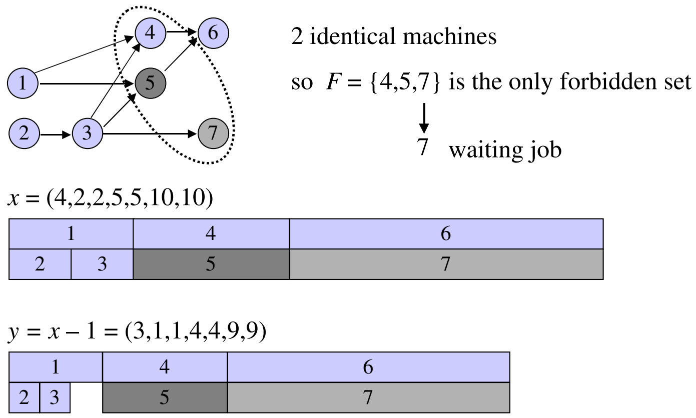


Figure 6. A preselective policy for Graham's example.

conditions, and there is a unique minimal representation that can be constructed in polynomial time.

The AND/OR graph is also useful for several computational tasks related to preselective policies.

First, it provides the right structure to compute the start times $\Pi[p]$ of a policy Π for a given realization p of processing times. This can be done by a Dijkstra-like algorithm since all processing times p_j are positive, see [12]. For general arc weights, the complexity status of computing earliest start times in AND/OR graphs is open. The corresponding decision problem "Is the earliest start of node v at most t ?" is in $NP \cap coNP$ and only pseudopolynomial algorithms are known to compute the earliest start times. This problem is closely related to mean-payoff games considered e.g. in [19]. This and other relationships as well as more applications of AND/OR graphs (such as disassembly in scheduling [4]) are discussed in [12]. [12] also derives a polynomial algorithm to compute the earliest start times when all arc weights are non-negative. This is already a non-trivial task that requires checking for certain 2-connected subgraphs with arc weights 0, which can be done by a variant of the feasibility checking algorithm.

Second, it can be used to detect implied waiting conditions, which is useful if "good" or optimal preselective policies are constructed in a branch and bound approach. There, branching is done on the possible choices of waiting jobs for a forbidden set F as demonstrated in Figure 7. The algorithm for detecting implied waiting conditions can then be used to check if the forbidden set F of the current tree node N has already an implicit waiting job that is implied by the earlier choices of waiting jobs in ancestors of N.

This also provides a criterion for dominance shown in [14]: A preselective policy is dominated iff no forbidden set F has a transitively implied waiting job that is different from the chosen waiting job.

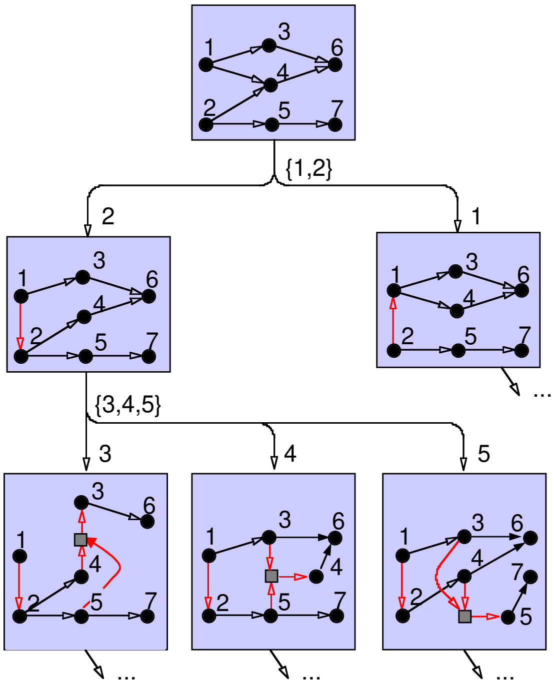


Figure 7. Computing preselective policies by branch and bound.

3.3 Special preselective policies

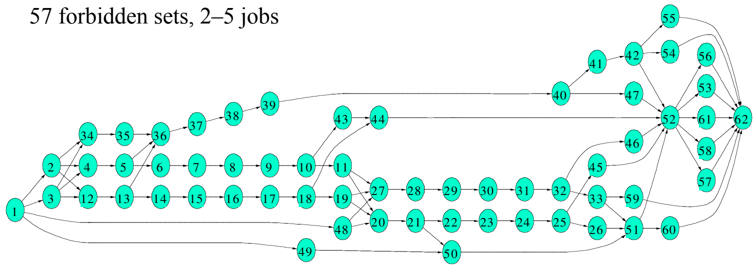
There are several interesting subclasses of preselective policies.

For instance, one may in addition to the waiting job j_F from a forbidden set F also choose the job $i_F \in F$ for which j must wait. That means that the conflict given by F is solved by introducing the additional precedence constraint $i_F < j_F$. Then the AND/OR graph degenerates into an AND graph, i.e., consists like G of ordinary precedence constraints. Feasibility of AND/OR graphs then reduces to being acyclic and early start scheduling can be done by longest path calculations. This class of policies is known as *earliest start policies* [8]. They are convex functions and every convex policy is already an earliest start policy [15].

Another class, the class of *linear preselective policies* has been introduced in [14]. It is motivated by the precedence tree concept used for $PS \mid prec \mid C_{\max}$, see e. g. [1, Section 3.1]), and combines the advantages of preselective policies and priority rules. Such a policy Π uses a priority list L (that is a topological sort of the graph G of precedence constraints) and chooses the waiting jobs $j_F \in F$ as the last job from F in L. It follows that a preselective policy is linear iff the corresponding AND/OR graph is acyclic. This class of policies possesses many favorable properties regarding domination, stability, computational effectiveness, and solution quality. Computational evidence is given in [17]. An example is presented in Figure 8. Here "Nodes" refers to the number of nodes considered in the branch and bound tree.

Linear preselective policies are related to *job-based priority rules* known from deterministic scheduling. These behave like priority policies, but obey the additional constraint that the start times S_j preserve the order given by the priority list L, i.e. $S_i \leq S_j$ if i precedes j in L. Every job based priority rule is linear preselective [14] but may be dominated by a better linear preselective policy. The advantage of job-based priority policies lies in the fact that they do not explicitly need to know the (possibly large) system $\mathcal{F}$ of forbidden sets. They settle the conflicts online by the condition on the start times.

Truncated Erlang distribution on [0.2*mean; 2.6*mean]
57 forbidden sets, 2–5 jobs



Optimum deterministic makespan	203	CPU: .17 sec
Optimum expected makespan	243.2	
Optimal preselective policy	Nodes: 115007	CPU: 3772.01 sec
Opt. linear preselective policy	Nodes: 4209	CPU: 49.85 sec

Figure 8. Computational results for linear preselective policies.

3.4 General robust policies

The class of all robust policies has been studied in [10] under the name set policies (as the decision at a decision time t is only based on the knowledge of the *set* of completed jobs and the *set* of busy jobs).

These policies behave locally like earliest start policies, i.e., for every robust policy Π, there is a partition of $\mathbb{R}^n_+$ into finitely many polyhedral cones such that, locally on each cone, Π is an earliest start policy and thus convex, continuous and monotone. This shows that Graham anomalies can only occur at the boundaries of these cones.

It turns out that for problems with independent exponential processing time distributions and "additive" cost functions, there is an optimal policy that is robust.

Here *additive* means that there is a set function $g : 2^V \to \mathbb{R}$ (the cost rate) such that $\kappa(C_1, \ldots, C_n) = \int g(U(t))dt$, where $U(t)$ denotes the set of jobs that are still uncompleted at time t. Special cases are $\kappa = C_{max}$, where $g(\emptyset) := 0$ and $g(U) = 1$ otherwise, and $\kappa = \sum w_j C_j$, where $g(U) = \sum_{j \in U} w_j$.

In more special cases (no precedence constraints, m identical parallel machines) there may even be optimal policies that are priority policies (again for independent exponential processing time distributions). If $\kappa = C_{max}$, then LEPT (longest expected processing time first) is known to be optimal, while for $\kappa = \sum C_j$, SEPT (longest expected processing time first) is optimal [18]. It is an open problem if there is also an optimal priority policy for $\kappa = \sum w_j C_j$.

4 How good are simple policies?

Compared to its deterministic counterpart, only little is known about the approximation behavior of (simple) policies for arbitrary processing time distributions. A first step into this direction is taken in [11] for the problem of minimizing the average weighted completion on identical parallel machines, i.e., $P \mid r_j,\ p_j = sto \mid \sum w_j C_j$.

This approach is based on a suitable polyhedral relaxation of the "performance space" $\mathcal{E} = \{(E(C_1^{\Pi}), \ldots, E(C_n^{\Pi})) \mid \Pi \text{ policy}\}$ of all vectors of expected completion times achieved by any policy. The optimal solution of an LP over this polyhedral relaxation is then used to construct priority and linear preselective policies, and these are shown to have constant factor performance guarantees, even in the presence of release dates. This generalizes several previous results from deterministic scheduling and also yields a worst case performance guarantee for the well known WSEPT heuristic.

We will illustrate this in the simplest case $P \mid p_j = sto \mid \sum w_j C_j$ and refer to [11] for more information and also related work on the optimal control of stochastic systems [3].

A policy is called an α-*approximation* if its expected cost is always within a factor of α of the optimum value, and if it can be determined and executed in polynomial time with respect to the input size of the problem. To cope with the input size of a stochastic scheduling problem, which includes non-discrete data in general, we assume that the input is specified by the number of jobs, the number of machines, and the encoding lengths of weights w_j, release dates r_j, expected processing times $E[\mathbf{p}_j]$, and, as the sole stochastic information, an upper bound Δ on the coefficients of variation of all processing time distributions $\mathbf{p}_j$, $j = 1, \ldots, n$. The coefficient of variation of a given random variable X is the ratio $\sqrt{\mathrm{Var}[X]}/E[X]$. Thus, it is particularly sufficient if all second moments $E[\mathbf{p}_j^2]$ are given. This notion of input size is motivated by the fact that from a practitioner's point of view the expected processing times of jobs together with the assumption of some typical distribution "around them" is realistic and usually suffices to describe a stochastic scheduling problem. Note, however, that the performance guarantees obtained actually hold with respect to optimal policies that make use of the *complete* knowledge of the distributions of processing times.

The polyhedral relaxation $\mathcal{P}$ is derived by a pointwise argument from known valid inequalities in completion time variables for the deterministic case [6].

Besides $E(\mathbf{C}_j) \geq E(\mathbf{p}_j)$ the crucial valid inequalities are

$$\sum_{j\in A} E(\mathbf{p}_j)\,E(\mathbf{C}_j) \geq \frac{1}{2m}\bigg(\big(\sum_{j\in A} E(\mathbf{p}_j)\big)^2 + \sum_{j\in A}(E(\mathbf{p}_j))^2\bigg) - \frac{m-1}{2m}\sum_{j\in A}\mathrm{Var}(\mathbf{p}_j) \qquad \text{for all } A \subseteq V.$$

They differ from the deterministic counterpart in the term involving the variances of the processing times. With the upper bound Δ on the coefficients of variation they may be rewritten as

$$\sum_{j\in A} E(\mathbf{p}_j)\,E(\mathbf{C}_j) \geq \frac{1}{2m}\bigg(\sum_{j\in A} E(\mathbf{p}_j)\bigg)^2 + \frac{1}{2}\sum_{j\in A}(E(\mathbf{p}_j))^2 - \frac{(m-1)(\Delta-1)}{2m}\bigg(\sum_{j\in A} E(\mathbf{p}_j)^2\bigg) \qquad \text{for all } A \subseteq V.$$

The LP relaxation $\min\{\sum_{j\in V} w_j\, C_j \mid C \in \mathcal{P}\}$ can be solved in polynomial time by purely combinatorial methods in $\mathrm{O}(n^2)$ time [11]. An optimal solution $C^{LP} = (C_1^{LP}, \ldots, C_n^{LP})$ to this LP defines an ordering L of jobs according to nondecreasing values of C_j^{LP}. This list L is then used to define a priority policy or linear preselective policy for the original problem.

If Π denotes such a policy, clearly $\sum_{j\in V} w_j C_j^{LP} \leq OPT \leq \sum_{j\in V} w_j E(\mathbf{C}_j)$, and the goal is to prove $\sum_{j\in J} w_j E(\mathbf{C}_j) \leq \alpha \sum_{j\in V} w_j C_j^{LP}$, for some $\alpha \geq 1$. This leads to a performance guarantee of α for the policy Π and also to a (dual) guarantee for the quality of the LP lower bound: $\sum_{j\in V} w_j E(\mathbf{C}_j) \leq \alpha \cdot OPT$ and $\sum_{j\in V} w_j C_j^{LP} \geq \frac{1}{\alpha}\, OPT$.

The performance guarantee thus obtained is $\alpha = 2 - \frac{1}{m} + \max\{1, \frac{m-1}{m}\Delta\}$, which may be improved to $\alpha = (1 + \frac{(\Delta+1)(m-1)}{2m})$ by the use of a specific priority policy (weighted expected processing time first). For problems with release dates, this priority policy can be arbitrarily bad, and the best guarantee is given by a job-based priority policy defined via the LP. The guarantees become stronger if $\Delta \leq 1$, which is the case for distributions that are NBUE (new better than used in expectation), which seems to be a reasonable class for applications.

These are the first non-trivial approximation algorithms with constant performance guarantees for stochastic scheduling problems. It is an open problem how to derive such algorithms for stochastic problems with precedence constraints.

References

1. P. Brucker, A. Drexl, R. H. Möhring, K. Neumann, and E. Pesch. Resource-constrained project scheduling: Notation, classification, models, and methods. *European J. Oper. Res.*, 112(1):3–41, 1999.

2. D. R. Fulkerson. Expected critical path lengths in PERT networks. *Oper. Res.*, 10:808–817, 1962.
3. K. D. Glazebrook and J. Niño-Mora. Scheduling multiclass queueing networks on parallel servers: Approximate and heavy-traffic optimality of Klimov's rule. In R. Burkard and G. Woeginger, editors, *Algorithms — ESA'97, 5th Annual European Symposium*, pages 232–245. Springer-Verlag, Lecture Notes in Computer Science, vol. 1284, 1997.
4. M. H. Goldwasser and R. Motwani. Complexity measures for assembly sequences. *International Journal of Computational Geometry and Applications.* To apear.
5. R. L. Graham. Bounds on multiprocessing timing anomalies. *Bell Systems Tech. Journal*, 45:1563–1581, 1968.
6. L. A. Hall, A. S. Schulz, D. B. Shmoys, and J. Wein. Scheduling to minimize average completion time: off-line and on-line approximation algorithms. *Mathematics Oper. Res.*, 22(3):513–544, 1997.
7. U. Heller. On the shortest overall duration in stochastic project networks. *Methods Oper. Res.*, 42:85–104, 1981.
8. G. Igelmund and F. J. Radermacher. Preselective strategies for the optimization of stochastic project networks under resource constraints. *Networks*, 13:1–28, 1983.
9. R. H. Möhring and F. J. Radermacher. Introduction to stochastic scheduling problems. In K. Neumann and D. Pallaschke, editors, *Contributions to Operations Research, Proceedings of the Oberwolfach Conference on Operations Research, 1984*, pages 72–130. Springer-Verlag, Lecture Notes in Economics and Mathematical Systems, vol. 240, 1985.
10. R. H. Möhring, F. J. Radermacher, and G. Weiss. Stochastic scheduling problems II – Set strategies. *Z. Oper. Res. Ser. A*, 29:65–104, 1985.
11. R. H. Möhring, A. S. Schulz, and M. Uetz. Approximation in stochastic scheduling: the power of LP-based priority policies. *J. Assoc. Comp. Mach.*, 46(6):924–942, 1999.
12. R. H. Möhring, M. Skutella, and F. Stork. Scheduling with AND/OR precedence constraints. Technical Report 646, Technische Universität Berlin, Fachbereich Mathematik, Berlin, Germany, 1999. Revised July 2000.
13. R. H. Möhring, M. Skutella, and F. Stork. Forcing relations for AND/OR precedence constraints. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms, San Francisco, CA*, pages 235–236, 2000.
14. R. H. Möhring and F. Stork. Linear preselective strategies for stochastic project scheduling. Technical Report 612, Technische Universität Berlin, Fachbereich Mathematik, Berlin, Germany, 1998. To appear in and Mathematical Methods of Operations Research.
15. F. J. Radermacher. Analytical vs. combinatorial characterizations of well-behaved strategies in stochastic scheduling. *Methods Oper. Res.*, 53:467–475, 1986.
16. J. Sgall. On-line scheduling. In A. Fiat and G. J. Woeginger, editors, *Online Algorithms: The State of the Art*, pages 196–231. Springer-Verlag, Lecture Notes in Computer Science, vol. 1442, 1998.
17. F. Stork. A branch-and-bound algorithm for minimizing expected makespan in stochastic project networks with resource constraints. Technical Report 613, Technische Universität Berlin, Fachbereich Mathematik, Berlin, Germany, 1998.
18. G. Weiss and M. Pinedo. Scheduling tasks with exponential service times on non-identical processors to minimize various cost functions. *J. Appl. Prob.*, 17:187–202, 1980.
19. U. Zwick and M. Paterson. The complexity of Mean Payoff Games on graphs. *Theor. Comp. Sc.*, 158:343–359, 1996.

Reports from the group

"Combinatorial Optimization and Graph Algorithms"

of the Department of Mathematics, TU Berlin

689/2000 *Rolf H. Möhring, Martin Skutella, and Frederik Stork:* Scheduling with AND/OR precedence constraints

682/2000 *Rolf H. Möhring and Marc Uetz:* Scheduling Scarce Resources in Chemical Engineering

681/2000 *Rolf H. Möhring:* Scheduling under Uncertainty: Optimizing Against a Randomizing Adversary

680/2000 *Rolf H. Möhring, Andreas S. Schulz, Frederik Stork, and Marc Uetz:* Solving Project Scheduling Problems by Minimum Cut Computations (Journal version merging the two Reports 620 and 661)

674/2000 *Esther M. Arkin, Michael A. Bender, Erik D. Demaine, Sándor P. Fekete, Joseph S. B. Mitchell, and Saurabh Sethia:* Optimal Covering Tours with Turn Costs

669/2000 *Michael Naatz:* A Note On a Question of C. D. Savage

667/2000 *Sándor P. Fekete and Henk Meijer:* On Geometric Maximum Weight Cliques

666/2000 *Sándor P. Fekete, Joseph S. B. Mitchell, and Karin Weinbrecht:* On the Continuous Weber and k-Median Problems

664/2000 *Rolf H. Möhring, Andreas S. Schulz, Frederik Stork, and Marc Uetz:* A Note on Scheduling Problems with Irregular Starting Time Costs

661/2000 *Frederik Stork and Marc Uetz:* Resource-Constrained Project Scheduling: From a Lagrangian Relaxation to Competitive Solutions

658/1999 *Olaf Jahn, Rolf H. Möhring, and Andreas S. Schulz:* Optimal Routing of Traffic Flows with Length Restrictions in Networks with Congestion

655/1999 *Michel X. Goemans and Martin Skutella:* Cooperative facility location games

654/1999 *Michel X. Goemans, Maurice Queyranne, Andreas S. Schulz, Martin Skutella, and Yaoguang Wang:* Single Machine Scheduling with Release Dates

653/1999 *Andreas S. Schulz and Martin Skutella:* Scheduling unrelated machines by randomized rounding

646/1999 *Rolf H. Möhring, Martin Skutella, and Frederik Stork:* Forcing Relations for AND/OR Precedence Constraints

640/1999 *Foto Afrati, Evripidis Bampis, Chandra Chekuri, David Karger, Claire Kenyon, Sanjeev Khanna, Ioannis Milis, Maurice Queyranne, Martin Skutella, Cliff Stein, and Maxim Sviridenko:* Approximation Schemes for Minimizing Average Weighted Completion Time with Release Dates

639/1999 *Andreas S. Schulz and Martin Skutella:* The Power of α-Points in Preemptive Single Machine Scheduling

634/1999 *Karsten Weihe, Ulrik Brandes, Annegret Liebers, Matthias Müller–Hannemann, Dorothea Wagner and Thomas Willhalm:* Empirical Design of Geometric Algorithms

633/1999 *Matthias Müller–Hannemann and Karsten Weihe:* On the Discrete Core of Quadrilateral Mesh Refinement

632/1999 *Matthias Müller–Hannemann:* Shelling Hexahedral Complexes for Mesh Generation in CAD

631/1999 *Matthias Müller–Hannemann and Alexander Schwartz:* Implementing Weighted b-Matching Algorithms: Insights from a Computational Study

629/1999 *Martin Skutella:* Convex Quadratic Programming Relaxations for Network Scheduling Problems

628/1999 *Martin Skutella and Gerhard J. Woeginger:* A PTAS for minimizing the total weighted completion time on identical parallel machines

627/1998 *Jens Gustedt:* Specifying Characteristics of Digital Filters with FilterPro

620/1998 *Rolf H. Möhring, Andreas S. Schulz, Frederik Stork, and Marc Uetz:* Resource Constrained Project Scheduling: Computing Lower Bounds by Solving Minimum Cut Problems

619/1998 *Rolf H. Möhring, Martin Oellrich, and Andreas S. Schulz:* Efficient Algorithms for the Minimum-Cost Embedding of Reliable Virtual Private Networks into Telecommunication Networks

618/1998 *Friedrich Eisenbrand and Andreas S. Schulz:* Bounds on the Chvátal Rank of Polytopes in the 0/1-Cube

617/1998 *Andreas S. Schulz and Robert Weismantel:* An Oracle-Polynomial Time Augmentation Algorithm for Integer Proramming

616/1998 *Alexander Bockmayr, Friedrich Eisenbrand, Mark Hartmann, and Andreas S. Schulz:* On the Chvátal Rank of Polytopes in the 0/1 Cube

615/1998 *Ekkehard Köhler and Matthias Kriesell:* Edge-Dominating Trails in AT-free Graphs

613/1998 *Frederik Stork:* A branch and bound algorithm for minimizing expected makespan in stochastic project networks with resource constraints

612/1998 *Rolf H. Möhring and Frederik Stork:* Linear preselective policies for stochastic project scheduling

609/1998 *Arfst Ludwig, Rolf H. Möhring, and Frederik Stork:* A computational study on bounding the makespan distribution in stochastic project networks

605/1998 *Friedrich Eisenbrand:* A Note on the Membership Problem for the Elementary Closure of a Polyhedron

596/1998 *Andreas Fest, Rolf H. Möhring, Frederik Stork, and Marc Uetz:* Resource Constrained Project Scheduling with Time Windows: A Branching Scheme Based on Dynamic Release Dates

595/1998 *Rolf H. Möhring Andreas S. Schulz, and Marc Uetz:* Approximation in Stochastic Scheduling: The Power of LP-based Priority Policies

591/1998 *Matthias Müller–Hannemann and Alexander Schwartz:* Implementing Weighted b-Matching Algorithms: Towards a Flexible Software Design

590/1998 *Stefan Felsner and Jens Gustedt and Michel Morvan:* Interval Reductions and Extensions of Orders: Bijections to Chains in Lattices

584/1998 *Alix Munier, Maurice Queyranne, and Andreas S. Schulz:* Approximation Bounds for a General Class of Precedence Constrained Parallel Machine Scheduling Problems

577/1998 *Martin Skutella:* Semidefinite Relaxations for Parallel Machine Scheduling

Reports may be requested from:

Hannelore Vogt-Möller
Fachbereich Mathematik, MA 6–1
TU Berlin
Straße des 17. Juni 136
D-10623 Berlin – Germany

e-mail: moeller@math.TU-Berlin.DE

Reports are also available in various formats from

`http://www.math.tu-berlin.de/coga/publications/techreports/`

and via anonymous ftp as

`ftp://ftp.math.tu-berlin.de/pub/Preprints/combi/Report-`*number*`-`*year*`.ps`