Scheduling with uncertain durations: generating β-robust schedules using constraint programming

Christine Wei Wu and **Kenneth N. Brown**
Cork Constraint Computation Center,
Department of Computer Science,
University College Cork, Ireland
{cww1, k.brown}@cs.ucc.ie

J. Christopher Beck
Toronto Intelligent Decision Engineering
Laboratory, Dept. of Mechanical and
Industrial Engineering,University of Toronto,
Canada. jcb@mie.utoronto.ca

Abstract

Many real-world scheduling problems are subject to change, and scheduling solutions should be robust to those changes. We consider a single-machine scheduling problem where the processing time of each activity is characterized by a normally-distributed random variable, and we attempt to minimize flowtime. We develop an initial constraint model for generating the β-robust schedule - the schedule that has highest probability of producing a flowtime less than a stated bound. Experiments with this initial model show that a constraint-based approach is feasible, but that better propagation methods will be required.

Introduction

Many real-world scheduling problems are subject to change: new jobs arrive, resources fail, or tasks take longer than expected. If these changes are significant, it may be better to generate solutions that are robust to them. A number of approaches have been proposed to handle uncertain scheduling problems. Redundancy-based Scheduling generates schedules with temporal slack so that unexpected events during execution can be handled by using that reserved slack (Davenport, Gefflot, & Beck 2001; Gao 1995). Contingent scheduling anticipates likely disruptive events and generates multiple schedules which optimally respond to the anticipated events (Drummond, Bresina, & Swanson 1994; Fowler & Brown 2003). Probabilistic scheduling uses probabilities over possible events, and searches for schedules which optimize the expected value of some performance measure (Daniels & Carrillo 1997; Walsh 2002; Beck & Wilson 2004; 2005). A number of approaches use sampling and scenarios, in order to produce robust decisions (Bent & Hentenryck 2004; Beck & Wilson 2004).

In particular, Daniels and Carrillo (Daniels & Carrillo 1997) introduced the concept of the β-robust schedule for a single machine scheduling problem with processing time uncertainty, which aims to maximize the probability of achieving a given performance level. They considered flowtime (the amount of time the tasks remain in the system) as the performance metric. They solved the problem by a branch-and-bound method with dominance rules, and heuristics for branch selection.

Constraint-based methods have proven to be very effective in a wide range of industrial scheduling problems. The advantage comes from the flexibility of the modeling language, and the ability of the solvers to deliver effective performance despite the presence of a wide range of different constraints and objectives. For these reasons, we develop an initial constraint model for solving the β-robust scheduling problem. For any given schedule, we will measure the probability of the total flowtime being less than a target level. We will then generate a schedule which maximizes the probability. Also, we will show the benefits of using standard constraint programming techniques. We introduce another objective, which is to find a schedule which optimizes the target level that can be achieved with a given probability. In this paper, we present a primal CP model to solver those problems.

The flowtime of a schedule

Before we show the primal model of the β-robust scheduling problem, we need to go through series of formal definitions and mathematical calculations. Those mathematical formulas will then give us a clear indication of any necessary variables and values for modeling the problem as a constraint satisfaction problem (CSP). There are several criteria of measuring schedules. In this report, we use the total flowtime to judge the solutions.

In a single machine scheduling problem, in which each job consists of a single task, a machine can only process one job at a time, and a job cannot be interrupted once started, a solution is a sequence of the jobs, and we assume the jobs are executed in sequence with no delay between them. Suppose we have a sequence $J_1, J_2, \ldots J_n$. Each job J_i has an arrival time A_i (its earliest possible start time), a start time ST_i, a duration d_i, and an end time E_i. We assume that each job is available for processing at time 0 (i.e. $A_i = 0$). We note the following simple relations: $E_i = ST_i + d_i$, $ST_1 = 0$, $ST_i = E_{i-1}$, and hence $E_i = \sum_{j=1}^{i} d_j$.

The *flowtime* is the total time the jobs are in the system: $TFT = \sum_{i=1}^{n}(E_i - A_i)$. Because we assume $Ai = 0$, we can rewrite the equation for total flowtime as follows:

$$TFT = \sum_{i=1}^{n} E_i = \sum_{i=1}^{n}\sum_{j=1}^{i} d_j = \sum_{i=1}^{n}(n+1-i) * d_i \quad (1)$$

We now assume that each job J_i's duration is an independent normally distributed random variable $d_i \sim N(\mu_i, \sigma_i^2)$.

We assume that the jobs will still be executed in the given sequence, regardless of the actual values of the durations. We note that for any two independent random variables $X \sim N(\mu_x, \sigma_x^2)$ and $Y \sim N(\mu_y, \sigma_y^2)$, and two constants a and b, the sum $aX + bY$ is also a normally distributed random variable, such that $aX+bY \sim N(a\mu_x+b\mu_y, a^2\sigma_x^2+b^2\sigma_y^2)$.

Since the activity durations are independent normally distributed random variables, and flowtime is a linear combination of durations, then for any particular sequence of jobs, the flowtime is also a normal random variable. From (1): $TFT \sim N(\sum_{i=1}^{n}(n-i+1)\mu_i, \sum_{i=1}^{n}(n-i+1)^2\sigma_i^2)$.

β-robust schedules

The β-robust schedule is useful because rather than gambling on the expected performance (or the average actual performance over a number of runs), it gives a lower limit on the performance, and to state the confidence in being able to achieve that level.

For example, consider the simple problem consisting of three jobs $\{x, y, z\}$, with uncertain durations $\{d_x \sim N(9, 2), d_y \sim N(5, 1), d_z \sim N(8, 7)\}$. The sequence $s_e = \langle y, z, x\rangle$ has a flowtime which is distributed as $N(40, 39)$. 40 is, in fact, the smallest expected flowtime possible for this problem. An alternative sequence, $s_\beta = \langle y, x, z\rangle$, has flowtime $\sim N(41, 24)$, and thus has a higher expected flowtime. However, suppose we now introduce a desired maximum flowtime of (for example) 51: the scheduler will incur a penalty if the actual schedule has a flow time greater than 51. Sequence s_e has a probability of 0.04 of producing a flowtime greater than 51, while s_β has a probability of just 0.02 of delivering a flowtime greater than 51, and thus s_β is likely to be less expensive. s_β is the β-*robust* (Daniels & Carrillo 1997) schedule for the maximum flowtime of 51 - that is, it has the highest probability of delivering a flowtime no greater than 51. In addition, for the confidence level of 0.98, s_β also delivers the minimal flowtime limit (51).

Definition 1. *For the single machine scheduling problem with n jobs, with normally-distributed uncertain durations, and with a flowtime limit S, the β-robust scheduling problem is to find the sequence, s, which maximizes the probability of the flowtime being less than S. That is, find the s that maximizes $Prob(flowtime(s) \leq S)$ (Daniels & Carrillo 1997).*

First, we show how to compute $Prob(\ flowtime \leq S\ )$ for an arbitrary sequence of the n jobs. Since the random variables in the problem are normally distributed, we can use the formula below to compute the probability of $flowtime \leq S$, where μ is the mean flowtime of the sequence, and σ^2 is its variance: $\phi(x \leq X) = 1/\sigma\sqrt{2\pi}\ \int_{-\infty}^{X} e^{\frac{-(x-\mu)^2}{2\sigma^2}}\, dx$. An arbitrary normal distribution can be converted to a standard normal distribution by changing variables to $z = (x-\mu)/\sigma$, so the normal distribution function becomes:

$$\phi(x \leq X) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{\frac{-t^2}{2}}\, dt = \frac{1}{2} + \phi(z)$$

where $\phi(z) = 1/\sqrt{2\pi}\ \int_0^z e^{\frac{-t^2}{2}}\, dt$. Hence, the probability of $flowtime \leq S$ can be computed by

$$Prob(\ flowtime \leq S\ ) = \frac{1}{2} + \phi(z) \tag{2}$$

where

$$z = \frac{S - mean(flowtime)}{\sqrt{var(flowtime)}}. \tag{3}$$

For each possible schedule, we can compute the mean and variance of the flowtime by $mean(flowtime) = \sum_{i=1}^{n}(n-i+1)\mu_i$ and $var(flowtime) = \sum_{i=1}^{n}(n-i+1)^2\sigma_i^2$ as in equation (1). Then, $\phi(z)$ can be obtained by checking z in the standard normal distribution table (Z-table).

Alternatively, there is a simple approximation of $\phi(z)$ which is good to two decimal places (Weisstein 2006), given by

$$\phi(z) \approx \varphi(z) \begin{cases} 0.1z(4.4-z) & (0 \leq z \leq 2.2) \\ 0.49 & (2.2 < z < 2.6) \\ 0.50 & (z \geq 2.6) \end{cases} \tag{4}$$

The β-robust schedule is one of those alternative sequences of the jobs, such that it has the maximum probability of $flowtime \leq S$. To find a β-robust schedule, we need to have an objective function to maximize the probability. We use the approximation of $\phi(z)$ to compute the probability, because $\varphi(z)$ increases on $[0, +\infty)$ that simplifies the calculation (the proof is straightforward and omitted here). By using that simplification, maximizing the probability of $flowtime \leq S$ is the same as maximizing z.

$$\begin{aligned} objective &= max(\ \frac{1}{2} + \phi(z)\ ) \\ &\approx \frac{1}{2} + max(\ \varphi(z)\ ) = \frac{1}{2} + \varphi(\ max(z)\ ). \end{aligned}$$

With above analysis and calculations, we are ready to introduce our constraint models for the β-robust scheduling problem.

Primal model

The primal model is shown in Figure 1. We assume a set $\{J_1, J_2, \ldots J_n\}$ of jobs, each with a normally-distributed random variable duration $D_i \sim N(\mu_i, \sigma_i^2)$. Differing from the previous sections, we now do not assume that the jobs are scheduled in the given sequence. With each job J_i, we associate a position variable, Pos_i, with domain $\{1, 2, \ldots, n\}$. The position variable Pos_i represents the position of J_i in the sequence: for instance, $Pos_2 = 3$ states that J_2 is scheduled to be the third job to start on the machine. Besides position variables, we also introduce additional variables for computing flowtime mean and variance and then the probability. The formula (1) indicates that flow time can be viewed as the sum of the contributions from all jobs. We define flowtime contribution of J_i as $FTContrib_i = (n - Pos_i + 1)D_i$. $meanFTContrib_i$ and $varFTContrib_i$ are the mean and variance of the flowtime contributions from J_i. The former has an integer value in $[\mu_i, n\mu_i]$, and the latter has a value in $[\sigma_i^2, n^2\sigma_i^2]$. The goal is to sequence those jobs, i.e. assign a distinct value to each Pos_i, such that the likelihood of the sequence (schedule) to

Figure 1: Primal Model

Variables:
Job positions: $Pos_1, ..., Pos_n$
Job mean flowtime contributions: $meanFTContrib_1, ..., meanFTContrib_n$
Job variance flowtime contributions: $varFTContrib_1, ..., varFTContrib_n$
Constraints:
`allDifferent`(Job positions)
$meanFTContrib_i = (n - Pos_i + 1)\mu_i$
$varFTContrib_i = (n - Pos_i + 1)^2\sigma_i^2$
$mean(flowtime) = \sum_{i=1}^{n} meanFTContrib_i$
$var(flowtime) = \sum_{i=1}^{n} varFTContrib_i$
Dominance constraints:
for $0 < i < j \leq n$,
$\mu_i \leq \mu_j$ and $\sigma_i^2 \leq \sigma_j^2 \Rightarrow Pos_i < Pos_j$
$\mu_i \geq \mu_j$ and $\sigma_i^2 > \sigma_j^2 \Rightarrow Pos_i > Pos_j$
$\mu_i > \mu_j$ and $\sigma_i^2 = \sigma_j^2 \Rightarrow Pos_i > Pos_j$

$$objective = max(z) = max(\ \frac{S - mean(flowtime)}{\sqrt{var(flowtime)}}\ )$$

achieve a fixed system performance level S is optimized, i.e. $max(\ probability(\ flowtime \leq S\ )\ )$.

Firstly, we have a permutation constraint that ensures each job takes a different position in the sequence. This can be implemented as a global all-different constraint on all the Pos_i. Also if we consider the flowtime as a sum of contributions from each job, from formula (1), we have

$$\begin{aligned} mean(flowtime) &= \sum_{i=1}^{n} meanFTContrib_i \\ &= \sum_{i=1}^{n} (n - Pos_i + 1)\mu_i, \\ var(flowtime) &= \sum_{i=1}^{n} varFTContrib_i \\ &= \sum_{i=1}^{n} (n - Pos_i + 1)^2\sigma_i^2. \end{aligned}$$

With those additional variables, we can use formula (2), (3) and (4) to compute the probability of a schedule's actual flowtime being less than S.

We are also able to impose some dominance constraints as in figure 1, using the properties of the β-robust schedule.

Theorem 1. *In a β-robust schedule, if job i with $D_i \sim N(\mu_i, \sigma_i^2)$ precedes job j with $D_j \sim N(\mu_j, \sigma_j^2)$, then either the mean duration of job i, μ_i, is no greater than the mean duration of job j, μ_j, or the duration variance of job i, σ_i^2, is no greater than the duration variance of job j, σ_j^2, that is $\mu_i \leq \mu_j$ or $\sigma_i^2 \leq \sigma_j^2$. (see Appendix for the proof)*

With this property, we post further constraints: for job i and job j ($0 < i < j \leq n$), if $\mu_i \leq \mu_j$ and $\sigma_i^2 \leq \sigma_j^2$, then $Pos_i < Pos_j$; if $\mu_i \geq \mu_j$ and $\sigma_i^2 > \sigma_j^2$ then $Pos_i > Pos_j$; if $\mu_i > \mu_j$ and $\sigma_i^2 = \sigma_j^2$ then $Pos_i > Pos_j$. Note that for the jobs have the same duration mean and the same duration variance, we take the lexicographical order on their indexes, i.e. if $\mu_i = \mu_j$ and $\sigma_i^2 = \sigma_j^2$ and $i < j$, then $Pos_i < Pos_j$.

As stated before, modeling the β-robust scheduling problem as a standard CSP enable us to change the objectives easily for different purposes. In particular, we can generate a schedule which optimizes the target level that can be achieved with a given probability.

Instead of maximizing the probability with a given system performance S, we might want to minimize S such that there exists a schedule that can achieve S with a fixed probability. That is $Min(\ S\ )$ such that $Probability(\ X \leq S\ ) \geq C$, where C is the fixed probability. Using the same primal model, we can get z value from formula (2) and (4) $z = \varphi^{-1}(C - \frac{1}{2})$. Then, from formula (3), we have a new objective function $min(S) = min(\ z * \sqrt{var(flowtime)} + mean(flowtime)\ )$. Note that φ^{-1} is not a one-to-one correspondence function (i.e. there are more than one z values for each value of C) at $C = 0.99$ and $C = 1.0$. For not over estimating S, we select the smallest z from all possible values. That is we set $z = 2.21$ when $C = 0.99$ and $z = 2.6$ when $C = 1.0$.

Besides those constraints we discussed above, we also implement a variable ordering heuristic to guide search. From formula (3), we can see that the β-robust schedule has the optimized combination of $mean(flowtime)$ and $var(flowtime)$. In order to find the β-robust schedule more quickly, we prefer to first schedule a job i, which has shorter mean processing time μ_i and smaller variance σ_i^2. We use a family of variable ordering heuristics, ordering the jobs by increasing $\mu_i + q * \sigma_i^2$, selecting a value for q based on the problem characteristics (probability levels). For higher probabilities, we expect the variance to be more significant, and so we choose higher values of q which give increasing weight to the duration variance in the variable ordering. Note that this variable ordering heuristic does not improve the total solving speed (i.e. the time of finding the schedule and proving it is the optimal), but does shorten the time to find the optimal solution.

Experimental results

We implemented the β-robust scheduling problem as a constraint satisfaction problem using ILOG Scheduler and Solver 6.0. Our first aim is to verify our initial constraint model, and so we expect to see the same pattern of results as obtained by (Daniels & Carrillo 1997). Secondly, we want to determine whether or not a constraint model is feasible for such problems, and so we hope to see runtimes of a similar order of magnitude. If we succeed in both aims, we will then investigate more sophisticated constraint models.

We consider problems with either 10 or 15 jobs, using the same experimental setup as (Daniels & Carrillo 1997). Table 1 contains the results for our constraint methods and the corresponding figures taking directly from (Daniels & Carrillo 1997). The CPU is the computation time for finding and proving the β-robust schedule. It also shows the differences (in average and in maximum deviation) between the mean

Table 1: Computational performance of β-robust solution procedure.

		Constraint model			Branch-and-bound		
total jobs	prob. level	CPU (sec.)	Avg. abv. SEPT (%)	Max. abv. SEPT (%)	CPU (sec.)	Avg. abv. SEPT (%)	Max. abv. SEPT (%)
10	0.85	0.1	0.1	0.4	0.2	0.1	0.8
	0.95	0.1	0.3	1.7	0.2	0.3	1.9
	0.99	0.1	0.5	1.9	0.3	0.6	2.5
15	0.85	2.3	0.1	0.3	1.0	0.1	0.5
	0.95	2.4	0.2	0.7	1.7	0.2	1.0
	0.99	3.0	0.3	1.5	2.1	0.4	1.9

processing time of the β-robust schedule and the shortest expected processing time (SEPT). Table 1 indicates that we do have a similar pattern in term of the mean flowtime of the β-robust schedule compared to the SEPT schedule. In addition, our CPU time is comparable for the smaller problems, but is poorer for the larger problems. This indicates that a constraint-based approach may be feasible, but that a more sophisticated model with better propagation will be required. Moreover, we set up a further experiment to determine the effort require to prove that the solution is optimal. The results shows that the program takes little time (e.g. 7% of total) to find the best solution but usually a long time (e.g. 93% of total) to prove if it is the β-robust schedule. We believe that a problem is hard for our model if it has many jobs with similar duration mean and variance. The program is able to do little propagation, and thus spends a lot of time trying different permutations of the jobs for no benefit.

With the general model, we can also give the minimum system performance S for a problem, so that the jobs in the problem can be scheduled to achieve the minimized S with a desired probability level. The details of experimental results have been abridged.

Future work

In constraint programming, it is sometimes very useful to change view points to study the same problem. Particularly, for a permutation problem (a constraint satisfaction problem in which each decision variable takes an unique value), we can transpose the roles of the variables and the values in presenting the underlying problem to give a new dual model which is also a permutation problem (Hnich, Smith, & Walsh 2004).

We are currently working on the dual model of the primal model, and a third model which channels between the primal model and its dual. We believe using the combined model will help us to improve the solving speed. We also plan to investigate better bounds for pruning branches at the top of the search tree, better heuristics to guide the search, and the construction of a global constraint for achieving β-robustness. We are also conducting an investigation into the characteristics of the problems which make some of them much harder to solve than others. Finally, we plan to extend this work to consider problems with multiple machines and with non-zero arrival times, for which the probability calculations reported here will not apply.

Conclusion

In this paper, we presented a general constraint model for the β-robust scheduling problem, which allows us to produce schedules which are robust to uncertainty in the durations of tasks. With flowtime as the performance measure, we can optimize the probability and find a most promising schedule to satisfy the system performance requirement; or we can optimize the performance level for a fixed probability. Our initial model demonstrates that a constraint-based approach is feasible for this problem, but that more more sophisticated models are required for good performance.

References

Beck, J. C., and Wilson, N. 2004. Job shop scheduling with probabilistic durations. *Proceedings of the Sixteenth European Conference on Artificial Intelligence* 652–656.

Beck, J. C., and Wilson, N. 2005. Proactive algorithms for scheduling with probabilistic durations. *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence* 1201–1206.

Bent, R., and Hentenryck, P. V. 2004. Online stochastic and robust optimization. *Ninth Asian Computing Science Conference* 286–300.

Daniels, R. L., and Carrillo, J. E. 1997. Beta-robust scheduling for single-machine systems with uncertain processing times. *IIE Transactions* 29:977–985.

Davenport, A. J.; Gefflot, C.; and Beck, J. C. 2001. Slack-based techniques for robust schedules. *Proceedings of the Sixth European Conference on Planning* 7–18.

Drummond, M.; Bresina, J. L.; and Swanson, K. 1994. Just-in-case scheduling. *Proceedings of the 12thNational Conference on Artificial Intelligence (AAAI)* 1098–1104.

Fowler, D. W., and Brown, K. N. 2003. Branching constraint satisfaction problems and markov decision problems compared. *Annals of Operations Research* 118:85–100.

Gao, H. 1995. Building robust schedules using temporal protectionan empirical study of constraint-based scheduling under machine failure uncertainty. *Masters thesis, Department of Industrial Engineering, University of Toronto, Toronto, Canada*.

Hnich, B.; Smith, B.; and Walsh, T. 2004. Dual modelling of permutation and injection problems. *Journal of Artificial Intelligence Research* 21:357–391.

Walsh, T. 2002. Stochastic constraint programming. *Proceedings of 15th European Conference on Artificial Intelligence* 111–115.

Weisstein, E. W. 2006. Normal distribution function. *Mathworld, Wolfram Research, Inc.* http:// mathworld. wolfram. com/ Normal Distribution Function. html.