A Scheduling Problem with Uncertain Activity Durations and Alternative Resources

December 4, 2003

1 General Description

The problem is a randomly generated $n \times m$ scheduling problem and consists in n process plans. Each process plan consists in m activities. For each activity a given number k of randomly picked alternative resources constitute the set of possible resources. There are $n \times m$ activities to be processed and the total number of resources is equal to q. A process plan is thus defined as a job whose each activity will be executed on one of k different alternative resources. A precision *precision* is applied to data.

We consider activities are pre-emptive: if a resource breaks down, then all the activities requiring this resource and being executed are suspended; when the corresponding resource is repaired all the activities requiring this resource and suspended resume execution.

2 Costs

Concerning costs, we have to maintain the schedule costs under cost thresholds by adding constraints. We distinguish two types of costs: tardiness and allocation costs and we consider P, a set of n process plans, A, a set of $n \times m$ activities, and R, a set of q resources.

2.1 Tardiness Cost

Each process plan $p_i \in P$ is associated with a due date due_i that is defined as follows: $due_i = tight_i \times dur_i$, where $tight_i$ is the tightness parameter and dur_i equals the mean process plan duration. In other words, dur_i is equal to the sum of the mean durations of the activities of p_i. $tight_i$ is randomly picked given a probability distribution called $tight$. If the last executed activity of p_i finishes later than due_i, then a cost $tardi_i$, called *tardiness cost*, is generated. $tardi_i$ depends on how late p_i finishes. Tardiness costs increase linearly with respect to how late process plans finish: the tardiness cost is a weight that is applied to each time unit after the due date that the process plan has not yet finished. If p_i finishes earlier than or at due_i, then $tardi_i$ is equal to zero. With each process plan we associate a different linear tardiness cost function: each slope ϕ_i is an integer randomly picked given a probability distribution noted ϕ. In a formal way, $tardi_i = \phi_i \times max(endp_i^{\text{eff}} - due_i, 0)$, where $endp_i^{\text{eff}}$ is the effective end time of p_i, observed during execution.

The sum of the tardiness costs must be less than or equal to $K_{\text{tardiness}}$.

$$\sum_{\forall p_i \in P} tardi_i \leq K_{\text{tardiness}}$$

2.2 Allocation Cost

Another cost, called *allocation cost*, is associated with each resource allocation: when a given activity $a_l \in A$ is executed by a given resource $r_j \in R$ it generates a cost $alloc_{jl}$ randomly picked from a probability distribution called *alloc*.

The sum of the allocation costs must be less than or equal to $K_{\text{allocation}}$.

$$\sum_{\forall r_j \in R, \forall a_l \in A} alloc_{jl} \leq K_{\text{allocation}}$$

3 How to Generate Instances of this Problem?

I propose the following: $n \times m = 100$, $k = 3$, $q = 15$, and $precision = 1,000$. Concerning costs, ϕ is a uniform probability distribution with $\phi^{\min} = 1$ and $\phi^{\max} = 10$; $tight$ is a uniform probability distribution with $tight^{\min} = 1$ and $tight^{\max} = 3$; $alloc$ is a uniform probability distribution such that: $alloc^{\min} = 1$ and $alloc^{\max} = 39$. There are three uncertainty levels: $\alpha_{\text{dur}} \in [0, \infty)$ that concerns activity durations, $\alpha_{\text{timeBreak}} \in [0, \infty)$ that concerns times between two consecutive breakdowns of resources, and $\alpha_{\text{breakDur}} \in [0, \infty)$ that concerns breakdown durations. The initial problem is generated as follows:

- we consider each activity a_l of the scheduling problem and randomly select three different resources out of 15 resources with equiprobability;
- each activity duration is described by a truncated normal probability distribution whose mean, $mDur_l$, is an integer randomly picked given a uniform probability distribution $mDur \in [1, 99]$ and standard deviation σDur_l is defined as follows: $\sigma Dur_l = \alpha_{\text{dur}} \times mDur_l$; each $mDur_l$ is truncated as follows: $mDur_l^{\min} = max(mDur_l - 5 \times \alpha_{\text{dur}} \times mDur_l, 0)$ and $mDur_l^{\max} = mDur_l + 5 \times \alpha_{\text{dur}} \times mDur_l$;
- each resource r_j is associated with a distribution $timeBreak_j$ describing the time between two consecutive breakdowns of r_j and a distribution $breakDur_j$ describing how long r_j breaks down. $timeBreak_j$ is a truncated normal probability distribution whose mean is $mTimeBreak_j$ and standard deviation is $\sigma TimeBreak_j$. $mTimeBreak_j = \theta TimeBreak_j \times \Delta$, where Δ is the average duration over all activities and $\theta TimeBreak_j$ is randomly picked in $[5, 15]$ with equiprobability. $\sigma TimeBreak_j = \alpha_{\text{timeBreak}} \times mTimeBreak_j$; each $timeBreak_j$ is truncated as follows: $timeBreak_j^{\min} = max(mTimeBreak_j - 5 \times \alpha_{\text{timeBreak}} \times mTimeBreak_j, 0)$ and $timeBreak_j^{\max} = mTimeBreak_j + 5 \times \alpha_{\text{timeBreak}} \times mTimeBreak_j$. $breakDur_j$ is a truncated normal probability distribution whose mean $mBreakDur_j = \theta BreakDur_j \times \Delta$ where $\theta BreakDur_j$ is randomly picked in $[1, 3]$ with equiprobablility and standard deviation $\sigma BreakDur_j = \alpha_{\text{breakDur}} \times mBreakDur_j$; each $breakDur_j$ is truncated as follows: $breakDur_j^{\min} = max(mBreakDur_j - 5 \times \alpha_{\text{breakDur}} \times mBreakDur_j, 0)$ and $breakDur_j^{\max} = mBreakDur_j + 5 \times \alpha_{\text{breakDur}} \times mBreakDur_j$; we start to simulate breakdowns before the date 0 because we assume that all resources have just broken down before 0 otherwise: I suggest to start simulation at the date $-55 \times \Delta / \alpha_{\text{break}}$ and account for breakdowns that occur at or after 0.

I propose to experiment with $\alpha_{\text{dur}} = \alpha_{\text{timeBreak}} = \alpha_{\text{breakDur}} = \alpha$ and $\alpha \in \{0.2, 0.5, 0.8\}$.

4 Comments

$\alpha = 0$ means the scheduling problem is deterministic with alternative resources: we know before execution starts activity durations, when and how long each resource breaks down. When $\alpha > 0$ that means the scheduling problem is non-deterministic with alternative resources, bigger α, more uncertain the scheduling problem.

5 File Formats

The instance and scenario generator is composed of two executable files *exec1* and *exec2*; *exec1* creates files describing problem instances and *exec2* creates files describing different execution scenarios.

exec1 is executed by giving it some arguments: the number n of process plans, the number m of activities per process plan, the number k of possible resources associated to each resource, the total number q of resources, the maximum allocation cost $K_{\text{allocation}}$, the maximum tardiness cost $K_{\text{tardiness}}$, and the uncertainty levels α_{dur}, $\alpha_{\text{timeBreak}}$, and α_{breakDur}.

Files created by *exec1* are each as follows:

n	m	k	q	$K_{\text{tardiness}}$	$K_{\text{allocation}}$				
r_1	$alloc_{11}$	r_4	$alloc_{41}$	$\dots$	$mDur_1$	σDur_1	$\dots$	due_1	ϕ_1
r_8	$alloc_{8m+1}$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	due_2	ϕ_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
r_6	$alloc_{61+m\times(n-1)}$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	due_n	ϕ_n

$mTimeBreak_1$	$\sigma TimeBreak_1$	$mBreakDur_1$	$\sigma BreakDur_1$
$mTimeBreak_2$	$\sigma TimeBreak_2$	$mBreakDur_2$	$\sigma BreakDur_2$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$mTimeBreak_q$	$\sigma TimeBreak_q$	$mBreakDur_q$	$\sigma BreakDur_q$

Below is an exemple of such file given a 3×2 scheduling problem with 2 possible resources associated with each activity, a total of 8 resources, a maximum tardiness cost equal to 123, and a maximum allocation cost equal to 214. p_1 is composed of a_1 followed by a_2: a_1 is associated with r_2 and r_4; $alloc_{21} = 13$ and $alloc_{41} = 22$; $mDur_1 = 15$ and $\sigma Dur_1 = 4$; a_2 is associated with r_3 and r_6; $alloc_{32} = 29$ and $alloc_{62} = 52$; $mDur_2 = 27$ and $\sigma Dur_2 = 3$; $due_1 = 50$ and $\phi_1 = 2$. p_2 is composed of a_3 followed by a_4; a_3 is associated with r_1 and r_6; $alloc_{13} = 23$ and $alloc_{63} = 77$; $mDur_3 = 21$ and $\sigma Dur_3 = 2$; a_4 is associated with r_7 and r_5; $alloc_{74} = 18$ and $alloc_{54} = 34$; $mDur_4 = 23$ and $\sigma Dur_4 = 5$; $due_2 = 60$ and $\phi_2 = 3$. p_3 is composed of a_5 followed by a_6; a_5 is associated with r_3 and r_5; $alloc_{35} = 17$ and $alloc_{55} = 56$; $mDur_5 = 24$ and $\sigma Dur_5 = 3$; a_6 is associated with r_1 and r_4; $alloc_{16} = 26$ and $alloc_{46} = 37$; $mDur_6 = 32$ and $\sigma Dur_6 = 4$; $due_3 = 55$ and $\phi_3 = 4$.

```
3   2   2   8   123 214
2   13  4   22  15  4   3  29  6  52  27  3  50  2
1   23  6   77  21  2   7  18  5  34  23  5  60  3
3   17  5   56  24  3   1  26  4  37  32  4  55  4
61  4   12  2
55  6   20  3
47  5   18  1
70  7   19  3
67  6   24  4
48  5   22  3
57  7   16  2
39  4   18  3
```

exec2 is executed by giving it a file generated by *exec1* as argument.

n	m	k	q	$K_{\text{tardiness}}$	$K_{\text{allocation}}$	
$duration_1^{\text{eff}}$	$duration_2^{\text{eff}}$	$\dots$	$\dots$	$\dots$	$\dots$	$duration_m^{\text{eff}}$
$duration_{m+1}^{\text{eff}}$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$duration_{2\times m}^{\text{eff}}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$duration_{1+m\times(n-1)}^{\text{eff}}$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$duration_{n\times m}^{\text{eff}}$

$timeBreak_1^{\text{eff}}$	$breakDur_1^{\text{eff}}$	$timeBreak_1^{\text{eff}}$	$breakDur_1^{\text{eff}}$	$\dots$	$breakDur_1^{\text{eff}}$
$timeBreak_2^{\text{eff}}$	$breakDur_2^{\text{eff}}$	$\dots$	$breakDur_2^{\text{eff}}$		
$\vdots$	$\vdots$				
$timeBreak_q^{\text{eff}}$	$breakDur_q^{\text{eff}}$	$\dots$	$breakDur_q^{\text{eff}}$		

Files created by *exec2* are each as follows:

Remarks: the number of breakdowns per resource is variable. For each resource r_j the effective first breakdown date $timeBreak_j^{\text{eff}}$ must be greater than or equal to zero and the last breakdown date $timeBreak_j^{\text{eff}}$ must be less than an upper bound on the schedule makespan. I propose the following upper bound:

$$\left(1 + \sum_{\forall r_j \in R} \frac{breakDur_j^{\max}}{\theta_j^{\min}}\right) \times \sum_{\forall a_l \in A} duration_l^{\text{eff}}$$

$duration_l^{\text{eff}}$ is the effective duration of the activity a_l and $breakDur_j^{\text{eff}}$ is the effective breakdown duration of the resource r_j.

Below is an exemple of such file given the scheduling problem presented above:

3	2	2	8	123	214				
13	28								
22	24								
21	30								
32	10	110	11	176	13	235	12		
24	18	100	21	174	19	252	17		
11	16	74	19	139	20	206	18	268	17
52	18	149	23	229	19				
5	25	97	26	184	28				
41	21	110	23	150	20	210	25		
34	15	114	14	193	16				
23	17	83	19	145	18	206	16	263	15