A tour through a maze of project and machine scheduling problems

Tamás Kis
Senior research fellow at the Institute of Computer Science and Control, Hungary

October 18, 2016
Who I am?

- PhD in Mathematics at Ecole Polytechnique Fédérale de Lausanne, Suisse (2001)
- Oui, je parle un peu français
- Main areas of research
  - Machine scheduling
  - Project scheduling
  - Integer programming
A non-exhaustive list of problems...

- The car-sequencing problem
- *Project scheduling with variable intensity activities
- Project scheduling with earliness/tardiness costs
- *Resource leveling in a machine environment
- Large scale multi-mode job-shop type problems
- *Bilevel scheduling problems
- Integrated planning and scheduling problems
- Scheduling with energy constraints (CECSP) (collaboration with LAAS)
- *Machine scheduling with non-renewable resources
- *Resource-constrained shortest path problem
- Vehicle and crew scheduling problems
- Disjunctive cuts and integer programming
Scheduling with variable intensity activities and feeding precedence constraints
Outline

- Problem data and MILP formulation
- Computational complexity
- Polyhedral results
- Branch-and-Cut
- Computational Results
- Applications
Problem Data (1)

- Discrete time horizon: $t = 1, \ldots, T$
- Continuously divisible renewable resources: $k \in R$
  - normal capacity: $b^k_t \geq 0$
  - extra capacity: $b^k_t \geq 0$
  - cost of using extra capacity: $c^k_t \geq 0$
- Variable intensity activities: $i \in N$
  - maximum intensity: $0 < a^i \leq 1$
  - time window: $\{r^i, \ldots, d^i\}$
  - resource requirements: $q^i_k \geq 0$, $k \in R$
Problem Data (2)

- Feeding Precedence Constraints: \((i, j, f) \in A\)
  - At least \(f\) fraction of activity \(i\) must be completed before \(j\) may start
  - The fraction of activity \(j\) completed up to any period \(\tau\) never exceeds the fraction of activity \(i\) completed up to period \(\tau\), i.e., \(\sum_{t=r_i}^{\tau} x_t^i \geq \sum_{t=r_j}^{\tau} x_t^j, \forall \tau\)

![Diagram showing precedence constraints](image-url)
The MILP formulation (1)

Variables:

\[ x_t^i = \text{intensity of activity } i \text{ in period } t \]
\[ z_t^{i,f} = \text{f-fraction mask of activity } i \text{ in period } t, \quad z_t^{i,f} \in \{0,1\} \]
\[ y_t^k = \text{external capacity of resource } k \text{ used in period } t \]

Objective:

\[ \min \sum_t \sum_{k \in R} c_t^k y_t^k \]
The MILP formulation (2)

Constraints:

$$\sum_{t=r_i}^{d_i} x_t^i = 1, \quad i \in N$$  \hspace{1cm} (1)

$$0 \leq x_t^i \leq a_i, \quad i \in N, \quad t \in \{r_i, \ldots, d_i\}$$ \hspace{1cm} (2)

$$\sum_{i \in N, \ t \in \{r_i, \ldots, d_i\}} q_k^i x_t^i \leq b_t^k + y_t^k, \quad k \in R, \ t \in \{1, \ldots, T\}$$ \hspace{1cm} (3)

$$0 \leq y_t^k \leq b_t^k, \quad k \in R, \ t \in \{1, \ldots, T\}$$ \hspace{1cm} (4)
The MILP formulation (3)

\[
\sum_{t=r^j}^{\tau} x_t^j - \sum_{t=r^i}^{\tau} x_t^i \leq 0, \quad (i, j, f) \in A, \quad \tau \in \{r^i, \ldots, d^i\} 
\] (5)

\[
\sum_{t=r^i}^{\tau-1} x_t^i + f \cdot z_t^{i,f} \geq f, \quad (i, j, f) \in A, \quad \tau \in \{r^i + p^{i,f}, \ldots, d^i\} 
\] (6)

\[
\sum_{t=\tau+1}^{d^j} x_t^j - z_t^{i,f} \geq 0, \quad (i, j, f) \in A, \quad \forall \tau 
\] (7)

\[
z_t^{i,f} - z_{t+1}^{i,f} \geq 0, \quad \forall i, f, t 
\] (8)
Other objective functions

Below we assume that there are no extra capacities: $\overline{b}_t^k = 0$

- Min Makespan: by dichotomic search find the smallest $T$ such that a feasible solution exists with $d^i = T$.

- Min Maximum Tardiness: $d^i = \tilde{d}^i + T_{\text{max}}$ and dichotomic search on $T_{\text{max}}$, where $\tilde{d}^i$ is the due-date of $i \in N$.

- Min Weighted Total Tardiness: define weights $w_t^i = 0$ if $t \in \{r^i, \ldots, \tilde{d}^i\}$ and $w_t^i = w^i$ if $t \in \{\tilde{d}^i + 1, \ldots, T\}$. The objective is $\min \sum_{i \in N} \sum_{t \in \{r^i, \ldots, T\}} w_t^i z_t^i$. 
Computational complexity

- If extra capacity is unlimited, a feasible solution can be found in polynomial time.
  - Heuristics often assume and exploit that a feasible solution can be found quickly
- If extra capacity is bounded, then feasible solution existence is NP-complete in the strong sense.
Cutting planes from precedence constraints

- Relaxation: drop all resource constraints (RIP)
- Feasible solutions of RIP = all intensity assignments to activities respecting the precedence constraints
- A valid inequality for RIP is valid for the original IP
- LP-RIP, the LP relaxation of RIP, has vertices with fractional coordinates
- LP-RIP can be tightened by automatically generating valid inequalities
The polytope $K_{ij}^f$ (1)

Consider $(i, j, f) \in A$ and let the polytope $K_{ij}^f$ be the convex hull of those $(x^i, x^j, z^i, f)$ satisfying the following:

\[
\sum_{t=r^i}^{d^i} x_t^i = 1, \quad (9)
\]
\[
0 \leq x_t^i \leq a_t^i, \quad t \in \{r^i, \ldots, d^i\} \quad (10)
\]
\[
\sum_{t=r^i}^{\tau-1} x_t^i + f \cdot z_t^{i,f} \geq f, \quad \tau \in \{r^i + p_t^{i,f}, \ldots, d^i\} \quad (11)
\]
\[
z_t^{i,f} - z_{t+1}^{i,f} \geq 0, \quad t \in \{r^i + p_t^{i,f}, \ldots, d^i - 1\} \quad (12)
\]
\[
z_t^{i,f} \in \{0, 1\} \quad (13)
\]
The polytope $K_{ijf}^2$ (2)

\[
\sum_{t=r}^{d} x_{t}^j = 1, \tag{14}
\]

\[
0 \leq x_{t}^j \leq a^j, \quad t \in \{r^j, \ldots, d^j\} \tag{15}
\]

\[
\sum_{t=\tau+1}^{d} x_{t}^j - z_{\tau}^{i,f} \geq 0, \quad \tau \in \{r^i + p^{i,f}, \ldots, d^i\} \tag{16}
\]

(To simplify notation, assume $r^i + p^{i,f} = r^j$ and $d^i + p^j \leq d^j$.)
Results for $K^{ijf}$

Decompose $K^{ijf}$ into two smaller dimensional polytopes:

$$K_i^{ijf} = \{ (x^i, z_i^f) \mid (x^i, z_i^f) \text{ satisfies } (9) - (13) \}$$

$$K_j^{ijf} = \{ (x^j, z_j^f) \mid (x^j, z_j^f) \text{ satisfies } (12) - (16) \}$$

**Lemma** $(x^i, x^j, z_i^f) \in K^{ijf}$ if and only if $(x^i, z_i^f) \in K_i^{ijf}$ and $(x^j, z_j^f) \in K_j^{ijf}$

**Observation** Both $K_i^{ijf}$ and $K_j^{ijf}$ are instances of the polytope $K^{f'}$:

$$K^{f'} = \text{conv}\{(x, z) \in \mathbb{R}^n \times \mathbb{B}^m \mid \sum_{t=1}^{n} x_t = 1, \forall t : 0 \leq x_t \leq a$$

$$\forall \tau : \sum_{t=\tau+1}^{m} x_t - f' \cdot z_{\tau} \geq 0, \forall t : z_t \geq z_{t+1} \}$$
Results for $K^f(1)$

► If $f = 1$, a minimal linear representation of $K^f$ is known. Cuts derived from a network flow

\[ \lambda_1 = 1 - z_1, \lambda_{\ell} = z_{\ell-1} - z_{\ell} \text{ and } \lambda_{m+1} = z_m \]

► The non-trivial inequalities are of the form:

\[
a_{\text{rem}} z_{t_1} + \sum_{t \in S_1 \setminus \{t_1\}} a_z t \leq \sum_{t \in \{t_1, \ldots, n\} \setminus (S_1 \cup S_2)} x_t,
\]

where $(p - 1)a < 1 \leq pa$, $a_{\text{rem}} = 1 - (p - 1)a$, $t_1 = \min S_1$, $S_1 \subseteq \{1, \ldots, m\}$, $S_2 \subseteq \{m + 1, \ldots, n\}$ and $|S_1| + |S_2| = p$

► There is an $O(n \log n)$ algorithm for separating $(S_1, S_2)$ inequalities
Results for $K^f (2)$

- When $0 < f \leq 1$, then the non-trivial facets of $K^f$ are

$$h_r z_{t_1} + h \sum_{t \in S_1 \setminus \{t_1\}} z_t + \sum_{t \in \{1, \ldots, t_1\} \setminus (S_1 \cup S_2)} x_t \geq f - h|S_2|,$$

for every $\emptyset \neq S_1 \subseteq \{p + 1, \ldots, m\}$, $S_2 \subseteq \{1, \ldots, p\}$, $|S_1| + |S_2| = p$, and $t_1$ being the greatest element of $S_1$;

$$(f - 1 + h|U^2_\ell|) z_\ell + h \sum_{t \in U^1_\ell} z_t + \sum_{t \in \{1, \ldots, n\} \setminus (U^1_\ell \cup U^2_\ell)} x_t \geq f,$$

for $\ell \in \{p + 1, \ldots, m\}$, $U^1_\ell \subseteq \{p + 1, \ldots, \ell\}$, $U^2_\ell \subseteq \{\ell + 1, \ldots, n\}$, $h|U^1_\ell \cup U^2_\ell| \leq 1$, and $h|U^2_\ell| \leq 1 - f$;

$$h \sum_{t \in U} z_t + \sum_{t \in \{1, \ldots, m\} \setminus U} x_t \geq f,$$

for $U \subseteq \{p + 1, \ldots, m\}$ with $h|U| \leq 1$. 
Test Instances

- De Boer instances:
  - $f = 1$
  - randomly generated precedence constraints
  - $1/a^i$ between 1 and 5
  - $n = 10, 20$ or $30$ activities
  - $r = 3, 5$ or $20$ resources
  - average $d^i - r^i - p^i + 1$ value: $s = 2, 5, 10, 15$ or $20$ (over the activities of the instance)
  - extra capacity is unlimited ($\bar{b}_t^k = \infty$)
  - 10 instances with the same $n$, $r$ and $s$ parameters

- My computational environment: PC, Pentium 4, 1.6GHz

- Algorithm B: Branch-and-Cut truncated after 675 seconds
Comparison to Branch-and-Price

Algorithm \( H \):


<table>
<thead>
<tr>
<th>( s )</th>
<th>( n = 10 )</th>
<th>( n = 20 )</th>
<th>( n = 50 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( r = 3 )</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( r = 20 )</td>
<td>0.86</td>
<td>0.97</td>
<td>0.98</td>
</tr>
<tr>
<td>( r = 20 )</td>
<td>0.71</td>
<td>0.91</td>
<td>0.97</td>
</tr>
<tr>
<td>( r = 20 )</td>
<td>0.56</td>
<td>0.92</td>
<td>0.96</td>
</tr>
</tbody>
</table>

**Table** The average of \( ub(B)/ub(H) \)
Results on instances with $0 < f < 1$

For $f = 0.5$, average $ub/lb$ values for algorithm $B^+$ (first row), and $B^-$ (second row), respectively, in each class $(n, r, s)$.

<table>
<thead>
<tr>
<th>s</th>
<th>$r = 3$</th>
<th>10</th>
<th>20</th>
<th>3</th>
<th>10</th>
<th>20</th>
<th>3</th>
<th>10</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.02</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.02</td>
<td>1.06</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.01</td>
<td>1.01</td>
<td>1.37</td>
<td>1.17</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.01</td>
<td>1.01</td>
<td>1.01</td>
<td>1.5</td>
<td>1.16</td>
</tr>
</tbody>
</table>
Application to Production Planning

- The first optimal solution with cost $C^* = \sum_{t,k} c_t^k y_t^k$ is usually unsatisfactory.

- Push the activities to the left (or to the right) by minimising $\sum_{i,t} w_t^i z_t^i$ with appropriate weights $w_t^i$ under the additional constraint $\sum_{t,k} c_t^k y_t^k \leq C^*$.

- Transform the solution to this problem by e.g., solving a linear program with quadratic objective function after fixing the $z_t^{i,f}$ to the values found previously.
Publications

Resource leveling in a machine environment

joint work with Márton Drótos
Outline

Definition of the problem

Complexity

Exact solution

Test results
Resource usage in a machine scheduling problem
Minimization of resource overuse

- $m$ parallel dedicated machines
- One renewable resource with capacity $C$
- Set of jobs to be executed on machine $i$: $N_i$
- Parameters of job $T_j$:
  - processing time ($p_j$)
  - time window ($e_j$, $d_j$)
  - resource demand ($b_j$)
Objective function

- Minimize total resource usage above $C$
- Minimize total squared resource usage above $C$
- Minimize quadratic deviation from mean resource demand (resource levelling)
Complexity

- NP-hard in the strong sense
- One machine, variable resource availability: NP-hard in the strong sense
- One machine, variable resource availability, fixed sequence: polynomial
Formulation as an integer program

\[
\min \sum_{t=0}^{D} y_t
\]

subject to

\[
\sum_{t=0}^{D} x_{jt} = 1, \quad \forall T_j \in N,
\]

\[
\sum_{T_j \in N} \sum_{\tau=t-p_j+1}^{t} x_{j\tau} \leq 1, \quad t \in \{0, \ldots, D\}, \quad \forall i \in M,
\]

\[
\sum_{T_j \in N} \sum_{\tau=t-p_j+1}^{t} b_j x_{j\tau} \leq C + y_t, \quad t \in \{0, \ldots, D\}
\]

\[
x_{jt} \in \{0, 1\} \quad \forall T_j, t
\]

\[
y_t \geq 0 \quad \forall t
\]
Lagrangean relaxation

Original problem

\[ \min cx \]
subject to
\[ Ax \leq b \]
\[ Dx \leq e \]
\[ x \geq 0, \ x \in \mathbb{Z}^r \]

Original problem

\[ \max \min cx + \lambda(Ax - b) \quad \lambda \geq 0 \]
subject to
\[ Dx \leq e \]
\[ x \geq 0, \ x \in \mathbb{Z}^r \]
Lagrangean relaxation of the problem

\[
\min \sum_{t=0}^{D} y_t
\]

subject to

\[
\sum_{t=0}^{D} x_{jt} = 1, \quad \forall T_j \in N,
\]

\[
\sum_{T_j \in N; \, \tau = t - p_j + 1}^{t} \sum_{t} x_{j\tau} \leq 1, \quad t \in \{0, \ldots, D\}, \quad \forall i \in M,
\]

\[
\sum_{T_j \in N; \, \tau = t - p_j + 1}^{t} \sum_{t} b_j x_{j\tau} \leq C + y_t, \quad t \in \{0, \ldots, D\}
\]

\[
x_{jt} \in \{0, 1\} \quad \forall j, t
\]

\[
y_t \geq 0 \quad \forall t
\]
Lagrangean relaxation of the problem

\[
\max_{\lambda \geq 0} LB(\lambda), \quad \lambda = (\lambda_1, \lambda_2, \ldots, \lambda_D)
\]

\[
LB(\lambda) = \min \sum_{t=0}^{D} ((1 - \lambda_t)y_t - \lambda_t C) + \sum_{i \in M} \sum_{T_j \in N_i} \sum_{t=0}^{D} \sum_{\tau = t - p_j + 1}^{t} \lambda_t b_j x_{j\tau}
\]

subject to

\[
\sum_{t=0}^{D} x_{jt} = 1, \quad \forall i \in M, \forall j \in N_i, \quad x_{jt} \geq 0 \quad \forall T_j, t
\]

\[
\sum_{T_j \in N_i} \sum_{\tau = t - p_j + 1}^{t} x_{j\tau} \leq 1, \quad t \in \{0, \ldots, D\}, \quad \forall i \in M
\]
Shaving

- $\lambda$ is fixed
- Lower bound if $T_j$ is started at time $t$: $\tilde{LB}_{jt}$
Shaving

- $\lambda$ is fixed
- Lower bound if $T_j$ is started at time $t$: $\bar{L}B_{jt}$
Branch&Bound

1. Constraint propagation
2. Calculation of lower bound
3. Shaving
4. Calculation of upper bound
5. Branching
<table>
<thead>
<tr>
<th>Method</th>
<th>data used</th>
<th>result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint propagation</td>
<td></td>
<td>narrowed time windows</td>
</tr>
<tr>
<td>Calculation of lower bound</td>
<td>$\lambda$ in the parent</td>
<td>lower bound $\lambda$</td>
</tr>
<tr>
<td>Shaving</td>
<td>$\lambda$</td>
<td>improved lower bound narrowed time windows</td>
</tr>
<tr>
<td>Calculation of upper bound</td>
<td>solutions of LP-s</td>
<td>upper bound schedule</td>
</tr>
<tr>
<td>Branching</td>
<td>shaving</td>
<td>lower bound of children</td>
</tr>
</tbody>
</table>
Quadratic objective function

\[
\min \sum_{t=0}^{D} y_t^2
\]

subject to

\[
\sum_{t=0}^{D} x_{jt} = 1, \quad \forall T_j \in N,
\]

\[
\sum_{T_j \in N; \tau = t - p_j + 1}^{t} \sum_{\tau} x_{j\tau} \leq 1, \quad t \in \{0, \ldots, D\}, \quad \forall i \in M,
\]

\[
\sum_{T_j \in N; \tau = t - p_j + 1}^{t} \sum_{\tau} b_j x_{j\tau} \leq C + y_t, \quad t \in \{0, \ldots, D\}
\]

\[
x_{jt} \in \{0,1\} \quad \forall T_j, t \quad y_t \geq 0 \quad \forall t
\]
Multiple renewable resources

\[
\min \sum_{\rho=1}^{R} \sum_{t=0}^{D} w_{\rho} y_{\rho t} \quad \text{vagy} \quad \min \sum_{\rho=1}^{R} \sum_{t=0}^{D} w_{\rho} y_{\rho t}^2
\]

subject to

\[
\sum_{t=0}^{D} x_{jt} = 1, \quad \forall T_j \in N,
\]

\[
\sum_{T_j \in N_i} \sum_{\tau=t-p_j+1}^{t} x_{j\tau} \leq 1, \quad t \in \{0, \ldots, D\}, \quad \forall i \in M,
\]

\[
\sum_{T_j \in N} \sum_{\tau=t-p_j+1}^{t} b_j x_{j\tau} \leq C_{\rho} + y_{\rho t}, \quad t \in \{0, \ldots, D\}, \quad \rho \in \{1, \ldots, R\}
\]

\[
x_{jt} \in \{0, 1\} \quad \forall T_j, t \quad y_t \geq 0 \quad \forall t
\]
Test results

- Generated test instances
- Resource limit:
  - Linear objective function: mean resource requirement
  - Quadratic objective function: 0 (resource levelling)
- 3 resources, with weights 4:3:2
- Initial solution can be found easily
- Runtime limit: 30 minutes
**Linear objective function**

<table>
<thead>
<tr>
<th></th>
<th>m5</th>
<th>m10</th>
<th>m20</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>t10</td>
<td>6.26%</td>
<td>0.67%</td>
<td>0.34%</td>
<td>2.42%</td>
</tr>
<tr>
<td>t15</td>
<td>13.28%</td>
<td>5.94%</td>
<td>0.32%</td>
<td>6.51%</td>
</tr>
<tr>
<td>t20</td>
<td>18.29%</td>
<td>5.53%</td>
<td>2.23%</td>
<td>8.69%</td>
</tr>
<tr>
<td>avg</td>
<td>12.61%</td>
<td>4.05%</td>
<td>0.97%</td>
<td>5.87%</td>
</tr>
</tbody>
</table>

**Branch&Bound**

<table>
<thead>
<tr>
<th></th>
<th>m5</th>
<th>m10</th>
<th>m20</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>t10</td>
<td>3.10%</td>
<td>0.24%</td>
<td>0.37%</td>
<td>1.24%</td>
</tr>
<tr>
<td>t15</td>
<td>11.28%</td>
<td>5.96%</td>
<td>1.62%</td>
<td>6.29%</td>
</tr>
<tr>
<td>t20</td>
<td>17.15%</td>
<td>7.48%</td>
<td>10.39%</td>
<td>11.67%</td>
</tr>
<tr>
<td>avg</td>
<td>10.51%</td>
<td>4.56%</td>
<td>4.13%</td>
<td>6.40%</td>
</tr>
</tbody>
</table>

**ILOG CPLEX**

$m_x$ indicates that the number of machines is $x$, $t_y$ means that the number of tasks one each machine is $y$, so there are $x \cdot y$ tasks in the corresponding instances.
### Quadratic objective function

<table>
<thead>
<tr>
<th></th>
<th>m5</th>
<th>m10</th>
<th>m20</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>t10</td>
<td>3.55%</td>
<td>1.10%</td>
<td>0.28%</td>
<td>1.65%</td>
</tr>
<tr>
<td>t15</td>
<td>4.22%</td>
<td>1.92%</td>
<td>1.14%</td>
<td>2.43%</td>
</tr>
<tr>
<td>t20</td>
<td>3.36%</td>
<td>1.29%</td>
<td>0.56%</td>
<td>1.73%</td>
</tr>
<tr>
<td>avg</td>
<td>3.71%</td>
<td>1.44%</td>
<td>0.66%</td>
<td>1.94%</td>
</tr>
</tbody>
</table>

### Branch & Bound

<table>
<thead>
<tr>
<th></th>
<th>m5</th>
<th>m10</th>
<th>m20</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>t10</td>
<td>1.51%</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>t15</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>t20</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>avg</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

### ILOG CPLEX

mx indicates that the number of machines is x, ty means that the number of tasks one each machine is y, so there are x \cdot y tasks in the corresponding instances.
Speed up using parallel computer\(^1\)

<table>
<thead>
<tr>
<th></th>
<th>m10</th>
<th>m20</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>thr5</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t10</td>
<td>2.65</td>
<td>2.58</td>
<td>2.62</td>
</tr>
<tr>
<td>t15</td>
<td>3.09</td>
<td>3.17</td>
<td>3.13</td>
</tr>
<tr>
<td>t20</td>
<td>3.29</td>
<td>3.33</td>
<td>3.31</td>
</tr>
<tr>
<td><strong>avg</strong></td>
<td>3.01</td>
<td>3.03</td>
<td>3.02</td>
</tr>
<tr>
<td><strong>thr10</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t10</td>
<td>3.18</td>
<td>3.24</td>
<td>3.21</td>
</tr>
<tr>
<td>t15</td>
<td>4.27</td>
<td>4.37</td>
<td>4.32</td>
</tr>
<tr>
<td>t20</td>
<td>4.81</td>
<td>4.92</td>
<td>4.87</td>
</tr>
<tr>
<td><strong>avg</strong></td>
<td>4.09</td>
<td>4.18</td>
<td>4.13</td>
</tr>
</tbody>
</table>

\(m_x\) indicates that the number of machines is \(x\),
\(t_y\) means that the number of tasks one each machine is \(y\), so there are \(x \cdot y\) tasks in the corresponding instances
\(thr_z\) indicates that the number of processor threads is \(z\)

\(^1\)Tested on the Sun Fire 15000 supercomputer of the National Information Infrastructure Development Institute of Hungary
Possible usage of other objective functions

- The lower bound can be calculated similarly for any objective function $f(y)$, for which

$$
\min_{y_l \leq y \leq y_u} f(y) - \lambda y
$$

can be solved “easily”

- The heuristic used for calculating the upper bound can be applied – and is at least pseudopolynomial – for any objective function $f(y)$ that can be formulated as

$$
f(y) = \sum_{t=0}^{D} f_t(y_t)
$$
Machine scheduling with non-renewable resources

joint work with Péter Györgyi and Márton Drótos
Outline

Producer jobs

Consumer jobs

Results
  Reductions
  Approximability

Publications
Producer jobs

Cumulative demand

Cumulative production

\[ S \]

\[ u_1 \quad u_2 \quad u_3 \quad u_4 \]

1. delivery 2. delivery 3. delivery 4. delivery

\[ J_{j^*} \]

delay
Problem description

Problem data:

- Single machine
- Set of product types $\mathcal{R}$.
- Set of jobs $\mathcal{J}$ with processing times $p_j$, and a quantity $a_{ij}$ produced from each product type $i \in \mathcal{R}$.
- External demand requests with deadlines $u_1 \leq \cdots \leq u_q$, and quantities requested $b_{i,\ell}$, $(i \in \mathcal{R})$.
- Assumptions: $\sum_j a_j = \sum_\ell b_\ell$ and $u_q = \sum p_j$.

Schedules:

- Starting time $S_j$ for each job $J_j$,
- Feasible if and only if jobs do not overlap in time
- Delivery time of the $\ell$th external demand request is the earliest time point when the total amount produced from product $i$ equals or exceeds $\sum_{k=1}^\ell b_{i,k}$, $i \in \mathcal{R}$
Parallel machines and consumer jobs

- We have \( m \) parallel machines \((\mathcal{M})\), a set of jobs \(\mathcal{J}\) and a set of resources \(\mathcal{R}\).
- Preemption is not allowed.
- Each job has a processing time and some resource requirements, i.e. we can schedule it only if there are enough resources to cover its requirements.
- These resources are consumed by the jobs at the time of starting.
- Each resource has an initial stock, which is replenished in known quantities at given dates.
- Objective functions: \( C_{\text{max}}, L_{\text{max}}, \sum w_j C_j \).

Parallel machines and consumer jobs (cntd.)

- The problem was introduced by Carlier (1984).
- The problem has many practical application (production scheduling, logistics).
- Main goal: approximability results for different sub-problems.
- We have to decide whether there is a PTAS or an FPTAS for different variants of the problem.
- Notations: $n$ jobs, $m$ machines, $rm$ number of resources, $q$ supplies ($0 = u_1 < u_2 < \ldots < u_q$).
- $b_{\ell,i}$: amount of the resource supplied at $u_\ell$ from resource $i$.
- $a_{i,j}$: requirement of job $J_j$ from resource $i$.
Consumer jobs

3 machines, 2 resources, 3 supplies and 8 jobs

<table>
<thead>
<tr>
<th>$p_j$</th>
<th>3</th>
<th>3</th>
<th>2</th>
<th>5</th>
<th>1</th>
<th>1</th>
<th>4</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_{1j}$</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$a_{2j}$</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>

- $M_1$
- $M_2$
- $M_3$

$u_1 = 0$, $u_2 = 3$, $u_3 = u_q = 9$

$b_{11} = 4$, $b_{21} = 1$, $b_{31} = 1$

$b_{12} = 3$, $b_{22} = 5$, $b_{32} = 4$
A feasible schedule

\[ u_1 = 0 \quad u_2 = 3 \quad u_3 = u_q = 9 \]

\[ b_{11} = 4 \quad b_{21} = 1 \quad b_{31} = 1 \]
\[ b_{12} = 3 \quad b_{22} = 5 \quad b_{32} = 4 \]
Supplies and consumption (resource 1)

\[
\begin{align*}
M_1 & = 0 \\
M_2 & = 3 \\
M_3 & = 9 \\
b_{11} & = 4 \\
b_{12} & = 3 \\
b_{21} & = 1 \\
b_{22} & = 5 \\
b_{31} & = 1 \\
b_{32} & = 4 \\
\end{align*}
\]

\[
\begin{array}{cccccccc}
p_j & 3 & 3 & 2 & 5 & 1 & 1 & 4 & 2 \\
a_{1j} & 0 & 1 & 2 & 1 & 0 & 1 & 1 & 0 \\
a_{2j} & 1 & 2 & 1 & 2 & 1 & 0 & 1 & 4 \\
\end{array}
\]

Cumulative supply

\[
S \\
\text{Cumulative supply}
\]
Supplies and consumption (resource 1)

\[ \begin{align*}
M_1 & \quad J_7 \\
M_2 & \quad J_3 \quad J_1 \\
M_3 & \quad J_4
\end{align*} \]

\[ u_1 = 0 \quad u_2 = 3 \quad u_3 = 9 \]
\[ b_{11} = 4 \quad b_{21} = 1 \quad b_{31} = 1 \]
\[ b_{12} = 3 \quad b_{22} = 5 \quad b_{32} = 4 \]

\[
\begin{bmatrix}
3 & 3 & 2 & 5 & 1 & 1 & 4 & 2 \\
0 & 1 & 2 & 1 & 0 & 1 & 1 & 0 \\
1 & 2 & 1 & 2 & 1 & 0 & 1 & 4 
\end{bmatrix}
\]

Cumulative supply

Cumulative consumption
Supplies and consumption (resource 1)

\[ M_1 \]

\[ M_2 \]

\[ M_3 \]

\[ S \]

[Diagram showing cumulative supply and consumption with relevant variables and values]
Some easy special cases of the delivery tardiness problem

Let $\gamma$ be a non-decreasing function of the delivery times.

1. $'1| p_j = p, |R| = 1 | \gamma'$ is solvable in $O(n \log n)$ time
   ▶ Schedule the jobs in non-increasing $a_{1,j}$ order.

2. $'1| a_j = a | \gamma'$ is solvable in $O(n \log n)$ time
   ▶ Schedule the jobs in non-decreasing processing time order (SPT rule)
Reductions between optimization problems

Reduction from problem class $\Pi_1$ to problems class $\Pi_2$ is a pair of functions $f$ and $g$ where $f$ transforms instances of $\Pi_1$ to that of $\Pi_2$, and $g$ transforms solutions of $\Pi_2$ to that of $\Pi_1$:

- **Problems:** $\Pi_1 \rightarrow \Pi_2$
  - **Instances:** $l_1 \rightarrow f \rightarrow f(l_1)$
  - **Solutions:** $g(l_1, y) \leftarrow y$

1. **L-reduction:** preserves approximation ratio with a constant factor
2. **Strict reduction:** preserves approximation ratio
3. **PTAS reduction:** preserves PTAS
4. **FPTAS reduction:** preserves FPTAS
Summary of approximation preserving reductions between scheduling and knapsack problems

$q$ and $r$ are arbitrary

$q = 2$, $r$ arbitrary
Equivalence of the delivery tardiness-, and the material consumption problems

Lemma
Let \( I = \{ n, q, (p_j, a_j)_{j=1}^n, (u_\ell, b_\ell)_{\ell=1}^q \} \) be an instance of the Delivery tardiness problem. Define an instance \( I' = \{ n, q, (p_j, a_j)_{j=1}^n, (u'_\ell, b'_\ell)_{\ell=1}^q \} \) of the Material consumption problem:

\[
\begin{align*}
  u'_\ell &= u_q - u_{q+1-\ell} & \ell = 1, \ldots, q. \\
  b'_\ell &= b_{q+1-\ell}
\end{align*}
\]

Then, if \( \sigma \) is a sequence of jobs with maximum delivery tardiness of \( T^\sigma_{\text{max}} \) for \( I \), then scheduling the jobs in reverse \( \sigma \) order gives a schedule of makespan \( u_q + T^\sigma_{\text{max}} \) for instance \( I' \) of the Material consumption problem.
Equivalence (part 2)

Lemma

Given an instance \( I = \{n, q, (p_j, a_j)_{j=1}^n, (u_\ell, b_\ell)_{\ell=1}^q\} \) of the Material consumption problem. Define an instance \( I' = \{n, q, (p_j, a_j)_{j=1}^n, (u'_\ell, b'_\ell)_{\ell=1}^q\} \) of the Delivery tardiness problem:

\[
\begin{align*}
    u'_\ell &= u_q - u_{q+1-\ell} \\
    b'_\ell &= b_{q+1-\ell}
\end{align*}
\]

\( \ell = 1, \ldots, q. \)

Then, if \( S \) is a schedule with a makespan of \( C^S_{\text{max}} \) for \( I \), then scheduling the jobs in reverse order (without any delays among them) gives a schedule of maximum delivery tardiness at most \( C^S_{\text{max}} - u_q \) for instance \( I' \) of the Delivery tardiness problem.
1. delivery  2. delivery  3. delivery  4. delivery

delay

Cumulative production

Cumulative demand

Cumulative supply

Cumulative consumption

1. supply  2. supply  3. supply  4. supply  \( C_{\text{max}} \)
Consequences of the equivalence

Corollary
Let \((I_M, I_D)\) be corresponding instances of the Material consumption- and the Delivery tardiness problems. Then the optimum value \(C_{\max}^*(I_M)\) of the Material consumption problem equals \(u_q(I_D) + T_{\max}^*(I_D)\).

In order to approximate the Delivery tardiness problem, use the shifted tardiness objective: \((u_q - u_1) + T_{\max}\)

Theorem
The Delivery tardiness problem admits a FPTAS / PTAS for some \(q\) and \(r\) if and only if the Material consumption problem admits an FPTAS / PTAS for the same \(q\) and \(r\).
Consequences of the reductions

**Theorem**

Any $1/(1 - \varepsilon)$-approx. algorithm for $r-$DKP yields an $(1 + \varepsilon)$-approx. algorithm for MCP$_2^r$ and DTP$_2^r$.

**Theorem**

There is no FPTAS for MCP$_2^r$ and DTP$_2^r$ for any fixed $r \geq 2$, unless $P = NP$. 
## Approximability results

<table>
<thead>
<tr>
<th>Obj.</th>
<th>#Machines</th>
<th>#Supplies</th>
<th>#Resources</th>
<th>Release dates</th>
<th>PTAS</th>
<th>FPTAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>(C_{\text{max}})</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>yes</td>
<td>yes</td>
<td>?</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>(\text{const.} \geq 2)</td>
<td>yes/no</td>
<td>yes</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>arbitrary</td>
<td>yes/no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>(\text{const.} \geq 3)</td>
<td>1</td>
<td>yes/no</td>
<td>yes</td>
<td>?</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>(\text{const.} \geq 3)</td>
<td>(\text{const.} \geq 2)</td>
<td>yes/no</td>
<td>yes</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>arbitrary</td>
<td>1*</td>
<td>yes/no</td>
<td>yes</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>(L'_{\text{max}})</td>
<td>(\text{const})</td>
<td>arbitrary</td>
<td>1*</td>
<td>no</td>
<td>yes</td>
<td>?</td>
</tr>
<tr>
<td>(\sum w_j C_j)</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

* under the condition \(a_j = \lambda p_j\)

** even if only a \(J' \subseteq J\) subset of jobs is dedicated
Publications


Solving resource constrained shortest path problems with LP-based methods

joint work with Markó Horváth
The Resource Constrained Shortest Path Problem (RCSPP)

**Given:**
- \( D = (V, A) \) directed graph, \( s \neq t \) selected nodes
- \( c : A \to \mathbb{Z} \) cost function
- \( w^r : A \to \mathbb{Z} \) (\( r = 1, \ldots, m \)) resource consumptions
- \( W^r \in \mathbb{Z} \) (\( r = 1, \ldots, m \)) resource limits

**Goal:** Find a minimal cost \( s-t \) path respecting the resource limits:

\[
\min_{P \in \mathcal{P}_{st}} \left\{ \sum_{a \in P} c_a : \sum_{a \in P} w^r_a \leq W^r \text{ for all } r = 1, \ldots, m \right\}
\]

**Assumption:** \( D \) contains no directed cycle of negative total cost, or of negative total resource consumption for any of the resource types
The RCSPP (and its variants) has several application fields in practice, and naturally arises as auxiliary problem in column generation schemes

- Crew scheduling, Integrated vehicle and crew scheduling
- Crew rostering
- Fleet management
- Air cargo planning and routing
- ...

Motivations
Aims and results

- LP-based branch-and-bound procedure
  - Variable fixing method
  - Primal heuristic
  - Valid inequalities (cutting planes)

- Thorough computational results
  - Evaluation of the components
  - Comparison with state-of-the-art methods
The RCSPP can be formulated as:

\[
\min \sum_{a \in A} c_a x_a \quad (1)
\]
\[
\sum_{a \in \delta^-(i)} x_a - \sum_{a \in \delta^+(i)} x_a = \begin{cases} 
1 & \text{if } i = s \\
-1 & \text{if } i = t \\
0 & \text{otherwise} 
\end{cases} \quad \forall \ i \in V \quad (2)
\]
\[
\sum_{a \in A} w^r_a x_a \leq W^r_a \quad \forall \ r = 1, \ldots, m \quad (3)
\]
\[
x_a \in \{0, 1\} \quad \forall \ a \in A \quad (4)
\]
Dumitrescu-Boland preprocessing scheme

Preprocessing scheme of Dumitrescu and Boland (2003):

Input:
- Original graph $D$
- Nodes $s$ and $t$, functions $c$ and $w^r$ ($r = 1, \ldots, m$)
- Current upper bound $U$

Process (repeatedly):
- Remove arcs (and nodes) if they cannot appear in an optimal solution
- Feasible path is found $\Rightarrow$ update $U$

Termination:
- Optimal solution is found $\Rightarrow$ STOP (problem is solved)
- Problem is proved to be infeasible $\Rightarrow$ STOP (problem is solved)
- No more arcs/nodes can be removed
### Computational results — Dumitrescu-Boland preprocessing scheme

#### Table: Properties of well-known instance sets

<table>
<thead>
<tr>
<th>Set</th>
<th>Ins</th>
<th>Res</th>
<th>Nodes Min</th>
<th>Nodes Max</th>
<th>Arcs Min</th>
<th>Arcs Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>I1</td>
<td>12</td>
<td>1</td>
<td>100</td>
<td>500</td>
<td>955</td>
<td>4978</td>
</tr>
<tr>
<td>I2</td>
<td>12</td>
<td>10</td>
<td>100</td>
<td>500</td>
<td>990</td>
<td>4868</td>
</tr>
<tr>
<td>D1</td>
<td>8</td>
<td>1</td>
<td>10002</td>
<td>135002</td>
<td>29900</td>
<td>404850</td>
</tr>
<tr>
<td>D2</td>
<td>56</td>
<td>1</td>
<td>625</td>
<td>40000</td>
<td>2400</td>
<td>159200</td>
</tr>
<tr>
<td>S</td>
<td>880</td>
<td>1</td>
<td>10000</td>
<td>40000</td>
<td>15000</td>
<td>800000</td>
</tr>
</tbody>
</table>

#### Table: Summary of experiments on well-known instance sets

<table>
<thead>
<tr>
<th>Set</th>
<th>Ins</th>
<th>Preprocessing Time</th>
<th>Preprocessing Solved</th>
<th>MIP BBN</th>
<th>MIP Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>I1</td>
<td>12</td>
<td>0.01</td>
<td>12</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>I2</td>
<td>12</td>
<td>0.03</td>
<td>10</td>
<td>1.0</td>
<td>0.45</td>
</tr>
<tr>
<td>D1</td>
<td>8</td>
<td>0.38</td>
<td>0</td>
<td>1.0</td>
<td>179.28</td>
</tr>
<tr>
<td>D2</td>
<td>56</td>
<td>0.12</td>
<td>1</td>
<td>3.0</td>
<td>22.96</td>
</tr>
<tr>
<td>S</td>
<td>880</td>
<td>0.12</td>
<td>878</td>
<td>2.0</td>
<td>160.71</td>
</tr>
</tbody>
</table>
### Table: Summary of experiments on instance set \( S \) (group 1)

<table>
<thead>
<tr>
<th>Set</th>
<th>RPM(^1)</th>
<th>PA(^2)</th>
<th>LPB(^3)</th>
<th>Set</th>
<th>RPM</th>
<th>PA</th>
<th>LPB</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>0.88</td>
<td>0.01</td>
<td>0.01</td>
<td>S7</td>
<td>3.00</td>
<td>0.02</td>
<td>0.03</td>
</tr>
<tr>
<td>S2</td>
<td>1.61</td>
<td>0.02</td>
<td>0.01</td>
<td>S8</td>
<td>5.73</td>
<td>0.03</td>
<td>0.04</td>
</tr>
<tr>
<td>S3</td>
<td>4.20</td>
<td>0.02</td>
<td>0.02</td>
<td>S9</td>
<td>13.45</td>
<td>0.06</td>
<td>0.04</td>
</tr>
<tr>
<td>S4</td>
<td>7.73</td>
<td>0.04</td>
<td>0.04</td>
<td>S10</td>
<td>28.84</td>
<td>0.11</td>
<td>0.09</td>
</tr>
<tr>
<td>S5</td>
<td>11.38</td>
<td>0.07</td>
<td>0.06</td>
<td>S11</td>
<td>43.66</td>
<td>0.17</td>
<td>0.14</td>
</tr>
<tr>
<td>S6</td>
<td>16.61</td>
<td>0.07</td>
<td>0.08</td>
<td>S12</td>
<td>64.85</td>
<td>0.17</td>
<td>0.22</td>
</tr>
<tr>
<td>S13</td>
<td>13.39</td>
<td>0.05</td>
<td>0.06</td>
<td>S14</td>
<td>22.05</td>
<td>0.08</td>
<td>0.08</td>
</tr>
<tr>
<td>S15</td>
<td>53.39</td>
<td>0.12</td>
<td>0.15</td>
<td>S16</td>
<td>112.92</td>
<td>0.21</td>
<td>0.22</td>
</tr>
<tr>
<td>S17</td>
<td>167.53</td>
<td>0.29</td>
<td>0.36</td>
<td>S18</td>
<td>248.18</td>
<td>0.39</td>
<td>0.49</td>
</tr>
</tbody>
</table>

### Figure: Execution times scaled to set S1

---

\(^1\) Puglieze & Guerriero (2013): tested with Intel Core i7-620M, 2.67 GHz CPU, under Windows 7
\(^2\) Lozano & Medaglia (2013): tested with Intel Core 2 Duo P8600, 2.4 GHz CPU, under Windows XP
\(^3\) [?]: tested with Intel Core i7-4710MQ, 2.5 GHz CPU, under Windows 7
Variable fixing in search-tree nodes

Figure: Fixed variables (arcs) in graph $D$
Variable fixing in search-tree nodes

Figure: Subgraph $\hat{D}$ according to fixed variables
Variable fixing method of [?]:

Input:
- Original graph $\mathcal{D}$
- Subgraph $\hat{\mathcal{D}}$
- Nodes $s$ and $t$, functions $c$ and $w^r$ ($r = 1, \ldots, m$)
- Current upper bound $U = \text{upper cutoff value}$

Process (repeatedly):
- Remove arcs (and nodes) Fix arcs to zero if they cannot appear in an (global) optimal solution
- Feasible path is found $\Rightarrow$ update $U$ and the upper cutoff value

Termination:
- Optimal solution is found $\Rightarrow$ STOP Prune the node
- Problem is proved to be infeasible $\Rightarrow$ STOP Prune the node
- No more arcs/nodes can be removed $\Rightarrow$ can be fixed
Computational results

Instance sets of Horváth and Kis (2016):

- Our randomly generated 240 instances can be grouped into 12 classes (G1-G12) according to their sizes, and the method used to generate their resource limits
- In all cases we used 10 resource functions

<table>
<thead>
<tr>
<th>parameters of instance groups</th>
<th>number of nodes*</th>
</tr>
</thead>
<tbody>
<tr>
<td>reduced (p = 20%)</td>
<td>G1   G5   G9</td>
</tr>
<tr>
<td>uniform (W = 20)</td>
<td>G2   G6   G10</td>
</tr>
<tr>
<td>uniform (W = 30)</td>
<td>G3   G7   G11</td>
</tr>
<tr>
<td>uniform (W = 40)</td>
<td>G4   G8   G12</td>
</tr>
</tbody>
</table>

* a graph with n nodes contains approximately 10n arcs
<table>
<thead>
<tr>
<th>Set</th>
<th>BBN</th>
<th>Time</th>
<th>Set</th>
<th>BBN</th>
<th>Time</th>
<th>Set</th>
<th>BBN</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLAIN</td>
<td>G1</td>
<td>1360.1</td>
<td>57.7</td>
<td>G5</td>
<td>1499.2</td>
<td>235.3</td>
<td>G9</td>
<td>2165.9</td>
</tr>
<tr>
<td>VARFIX</td>
<td>G1</td>
<td>881.7</td>
<td>19.0</td>
<td>G5</td>
<td>1000.3</td>
<td>54.0</td>
<td>G9</td>
<td>1101.4</td>
</tr>
<tr>
<td>PLAIN</td>
<td>G2</td>
<td>1883.6</td>
<td>88.9</td>
<td>G6</td>
<td>2621.3</td>
<td>392.9</td>
<td>G10</td>
<td>2308.4</td>
</tr>
<tr>
<td>VARFIX</td>
<td>G2</td>
<td>1351.3</td>
<td>33.5</td>
<td>G6</td>
<td>1149.5</td>
<td>63.5</td>
<td>G10</td>
<td>782.5</td>
</tr>
<tr>
<td>PLAIN</td>
<td>G3</td>
<td>4275.3</td>
<td>187.1</td>
<td>G7</td>
<td>4877.5</td>
<td>709.0</td>
<td>G11</td>
<td>5533.7</td>
</tr>
<tr>
<td>VARFIX</td>
<td>G3</td>
<td>3775.9</td>
<td>84.5</td>
<td>G7</td>
<td>4281.7</td>
<td>235.4</td>
<td>G11</td>
<td>4864.3</td>
</tr>
<tr>
<td>PLAIN</td>
<td>G4</td>
<td>3699.0</td>
<td>143.4</td>
<td>G8</td>
<td>4222.5</td>
<td>560.8</td>
<td>G12</td>
<td>6044.6</td>
</tr>
<tr>
<td>VARFIX</td>
<td>G4</td>
<td>3294.8</td>
<td>63.9</td>
<td>G8</td>
<td>4620.6</td>
<td>223.2</td>
<td>G12</td>
<td>5636.2</td>
</tr>
</tbody>
</table>

**PLAIN:** preprocessing + branch-and-bound
**VARFIX:** preprocessing + branch-and-bound with variable fixing
Primal heuristic

Once a fractional solution $\bar{x}$ to the LP-relaxation is found we perform a depth-first-search from node $s$ on the support graph $\bar{D}$ of the solution

$$\bar{D} = (V, \bar{A} = \{a \in A : \bar{x}_a > 0\})$$

- node labels $\lambda_s(u)$ and $\lambda_t(u)$: distance (for a fixed resource) from the source node and to the sink node, respectively.
- When we visit a node $u$ we can store/update a label $\lambda_s(u)$ about the current $s$–$u$ path
- When we backtrack to a node $u$ we can store/update a label $\lambda_t(u)$ about an $u$–$t$ path
- When for a node $u$ both of the labels $\lambda_s(u)$ and $\lambda_t(u)$ exist we can check the feasibility of the appropriate $s$–$t$ path
## Computational results

### Table: Summary of variable fixing and primal heuristic experiments

<table>
<thead>
<tr>
<th>Set</th>
<th>BBN</th>
<th>Time</th>
<th>Set</th>
<th>BBN</th>
<th>Time</th>
<th>Set</th>
<th>BBN</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLAIN</td>
<td></td>
<td></td>
<td>PLAIN</td>
<td></td>
<td></td>
<td>PLAIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>G1</td>
<td>1360.1</td>
<td>57.7</td>
<td>G5</td>
<td>1499.2</td>
<td>235.3</td>
<td>G9</td>
<td>2165.9</td>
<td>644.7</td>
</tr>
<tr>
<td>HEURSOL</td>
<td>1191.3</td>
<td>49.6</td>
<td>HEURSOL</td>
<td>1223.9</td>
<td>175.2</td>
<td>HEURSOL</td>
<td>1841.0</td>
<td>501.1</td>
</tr>
<tr>
<td>VARFIX</td>
<td>881.7</td>
<td>19.0</td>
<td>VARFIX</td>
<td>1000.3</td>
<td>54.0</td>
<td>VARFIX</td>
<td>1101.4</td>
<td>93.5</td>
</tr>
<tr>
<td>ALL</td>
<td>775.3</td>
<td>19.4</td>
<td>ALL</td>
<td>877.8</td>
<td>53.6</td>
<td>ALL</td>
<td>1090.3</td>
<td>103.5</td>
</tr>
<tr>
<td>G2</td>
<td>1883.6</td>
<td>88.9</td>
<td>G6</td>
<td>2621.3</td>
<td>392.9</td>
<td>G10</td>
<td>2308.4</td>
<td>679.9</td>
</tr>
<tr>
<td>HEURSOL</td>
<td>1769.3</td>
<td>81.5</td>
<td>HEURSOL</td>
<td>2303.7</td>
<td>339.2</td>
<td>HEURSOL</td>
<td>2092.1</td>
<td>580.8</td>
</tr>
<tr>
<td>VARFIX</td>
<td>1351.3</td>
<td>33.5</td>
<td>VARFIX</td>
<td>1149.5</td>
<td>63.5</td>
<td>VARFIX</td>
<td>782.5</td>
<td>71.0</td>
</tr>
<tr>
<td>ALL</td>
<td>1265.9</td>
<td>31.8</td>
<td>ALL</td>
<td>1086.1</td>
<td>66.8</td>
<td>ALL</td>
<td>660.6</td>
<td>83.2</td>
</tr>
<tr>
<td>G3</td>
<td>4275.3</td>
<td>187.1</td>
<td>G7</td>
<td>4877.5</td>
<td>709.0</td>
<td>G11</td>
<td>5533.7</td>
<td>1400.7</td>
</tr>
<tr>
<td>HEURSOL</td>
<td>3261.0</td>
<td>134.2</td>
<td>HEURSOL</td>
<td>4410.4</td>
<td>615.0</td>
<td>HEURSOL</td>
<td>4637.2</td>
<td>1087.5</td>
</tr>
<tr>
<td>VARFIX</td>
<td>3775.9</td>
<td>84.5</td>
<td>VARFIX</td>
<td>4281.7</td>
<td>235.4</td>
<td>VARFIX</td>
<td>4864.3</td>
<td>432.6</td>
</tr>
<tr>
<td>ALL</td>
<td>3412.2</td>
<td>76.6</td>
<td>ALL</td>
<td>3722.3</td>
<td>213.0</td>
<td>ALL</td>
<td>3990.3</td>
<td>370.9</td>
</tr>
<tr>
<td>G4</td>
<td>3699.0</td>
<td>143.4</td>
<td>G8</td>
<td>4222.5</td>
<td>560.8</td>
<td>G12</td>
<td>6044.6</td>
<td>1434.0</td>
</tr>
<tr>
<td>HEURSOL</td>
<td>2742.5</td>
<td>101.0</td>
<td>HEURSOL</td>
<td>3644.5</td>
<td>444.9</td>
<td>HEURSOL</td>
<td>5150.7</td>
<td>1073.3</td>
</tr>
<tr>
<td>VARFIX</td>
<td>3294.8</td>
<td>63.9</td>
<td>VARFIX</td>
<td>4620.6</td>
<td>223.2</td>
<td>VARFIX</td>
<td>5636.2</td>
<td>469.2</td>
</tr>
<tr>
<td>ALL</td>
<td>3102.2</td>
<td>62.4</td>
<td>ALL</td>
<td>3982.5</td>
<td>197.9</td>
<td>ALL</td>
<td>5067.4</td>
<td>432.3</td>
</tr>
</tbody>
</table>

**PLAIN:** preprocessing + branch-and-bound

**HEURSOL:** preprocessing + branch-and-bound with primal heuristic

**VARFIX:** preprocessing + branch-and-bound with variable fixing

**ALL:** preprocessing + branch-and-bound with primal heuristic and variable fixing
Valid inequalities

We have generalized two classes of valid inequalities of Garcia (2009):

- $s-t$ cut based inequalities
  - polynomial separation procedure

- Infeasible subpath based inequalities
  - NP-hard to separate
  - heuristic separation procedure

- Cuts were tested, they are significantly better on average than solver’s built in cuts, but we can get even better results without generating any cuts, at least on the pure RCSPP instances generated by us.
Bilevel machine scheduling problems

joint work with András Kovács
Outline

- Bilevel optimization
- The bilevel total weighted completion time problem
- The bilevel order acceptance problem
- Open problems
Bilevel optimization

- Two players: leader and follower
- Leader’s role
  - controls the variables $x$
  - decides first in view of the possible actions of the follower
  - aims to optimise an objective function which depends on $x$ and on the Follower’s response $\bar{y}$
- Follower’s role
  - controls the variables $y$
  - decides second in view of the decision $\bar{x}$ of the Leader
  - aims to optimise an objective function which depends on $y$ and possibly $\bar{x}$
- Leader’s objective:
  - Optimistic: $\min_x f(x, y)$ subject to $(x, y) \in L$ and $y \in \text{opt}_F(x)$
  - Pessimistic: $\min_x \max_y f(x, y)$ subject to $(x, y) \in L$ and $y \in \text{opt}_F(x)$
The bilevel total weighted completion time problem

- $n$ jobs and $m$ parallel, identical machines, no preemption
- Leader: assigns jobs to machines ($J = J_1 \cup J_2 \cup \cdots \cup J_m$)
  - Optimistic objective: $\min \sum_{j \in J} w_j^1 C_j$
  - Pessimistic objective: $\min \max \sum_{j \in J} w_j^1 C_j$
- Follower: sequences the assigned jobs on each machine

$$\min \sum_{i=1}^{m} \sum_{j \in J_i} w_j^2 C_j$$

- For a given machine assignment $J_1, \ldots, J_m$, Follower solves $m$ single machine problems $1|| \sum_j w_j^2 C_j$ by Smith’s rule (WSPT order)
- Leader has to find the best assignment knowing the strategy of the Follower
## Results on bilevel total weighted completion time problem

<table>
<thead>
<tr>
<th>Restriction</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>no restriction</td>
<td>decision version is NP-complete</td>
</tr>
<tr>
<td>$w^1 \equiv 1, w^2$ induces</td>
<td>equivalent to $P</td>
</tr>
<tr>
<td>an increasing proc. time order</td>
<td></td>
</tr>
<tr>
<td>$w^1 \equiv 1, w^2$ induces</td>
<td>reduces to a special</td>
</tr>
<tr>
<td>A decreasing proc. time order</td>
<td>MAX $m$-CUT problem</td>
</tr>
<tr>
<td>$w^1 \equiv 1, w^2$ arbitrary</td>
<td>FPTAS of Sahni for $Pm</td>
</tr>
<tr>
<td>$m$ constant</td>
<td>can be generalized</td>
</tr>
</tbody>
</table>
The structure of optimal solutions

Lemma

There is a global ordering of jobs such that in an optimal solution on each machine the job sequence respects the global order. In the optimistic case the global order is WSPT with respect to $w^2$ and in case of ties WSPT w.r.t. $w^1$. In the pessimistic case the global order is WSPT with respect to $w^2$ and in case of ties reverse WSPT w.r.t. $w^1$. 
Reduction to the MAX $m$-CUT problem

MAX $m$-CUT (optimization version)

**input**: the number of vertices (of a complete graph) $n$, edge weights $c_e$ for all the $n(n-1)/2$ edges, a number $m$ with $m \leq n$ (all data in $\mathbb{Z}_+$)

**output**: a partitioning of the vertices into $m$ disjoint classes $V_1, \ldots, V_m$ such that the total weight of edges between the classes is maximized, i.e.,

$$\max_{(V_1,\ldots,V_m)} \sum_{k=1}^{m-1} \sum_{\ell=k+1}^{m} \sum_{i \in V_k, j \in V_\ell} c_{ij}$$

where the maximum is over all $m$-partitions of the $n$ nodes

**Reduction**: the nodes are identified with the $n$ tasks, and

$$c_{jk} = p_j w_k^1 \quad \text{if} \quad \frac{w_j^2}{p_j} > \frac{w_k^2}{p_k} ; \quad \text{or} \quad \frac{w_j^2}{p_j} = \frac{w_k^2}{p_k} \quad \text{and} \quad \frac{w_j^1}{p_j} \geq \frac{w_k^1}{p_k}$$
A special MAX $m$-CUT problem

Special weights
If $w^1 \equiv 1$ and $\frac{w^2_j}{p_j} > \frac{w^2_k}{p_k}$ iff $p_j > p_k$, then $c_{jk} = \max\{p_j, p_k\}$

Theorem
There exists an optimal solution to MAX $m$-CUT such that $V_1 = \{1, \ldots, k_1\}$, $V_2 = \{k_1 + 1, \ldots, k_2\}$, \ldots, $V_m = \{k_{m-1} + 1, \ldots, m\}$, where $p_j \geq p_k$ for $j < k$.

Corollary
The MAX $m$-CUT problem with the above weights can be solved by dynamic programming in polynomial time
The bilevel order acceptance problem

- There are $n$ jobs with processing times $p_j$, due-dates $d_j$, and job-weights $w^1_j, w^2_j$; and a single machine
- **Leader**: selects a subset of jobs $A$ (accepted jobs) to maximize $\sum_{j \in A} w^1_j$
- **Follower**: sequences the jobs non-preemptively to minimize $\sum_{j \in A} w^2_j C_j$
- The solution is feasible iff the optimal solution chosen by the Follower meets the due-dates of all jobs in $A$
- If the Leader is **optimistic**, it selects $A$ such that at least one optimal solution of the Follower meets all the due-dates
- If the Leader is **pessimistic**, it selects $A$ such that all the optimal solutions of the Follower with respect to $A$ meets all the due-dates
Results on the bilevel order acceptance problem

<table>
<thead>
<tr>
<th>Restriction</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>no restriction</td>
<td>decision version is NP-complete</td>
</tr>
<tr>
<td></td>
<td>solvable in pseudo-poly time</td>
</tr>
<tr>
<td>$w^1 \equiv 1$</td>
<td>Polynomial (generalized Moore-Hodgson alg.)</td>
</tr>
</tbody>
</table>
A polynomial algorithm for the \( w^1 \equiv 1 \) case

The Moore-Hodgson algorithm for \( 1\| \sum U_j \)

1. Order the jobs in EDD order: \( d_1 \leq \ldots \leq d_n \)

2. Starting with the first job, process the jobs one-by-one. If all jobs can be completed on time, stop. Otherwise, let \( k_1 \) be the first job such that \( \sum_{j=1}^{k_1} p_j > d_{k_1} \). Remove from the first \( k_1 \) jobs the one with largest \( p_j \) value, and proceed with the next job.

Modification for the bilevel order acceptance problem:

1. Order the jobs in the Follower’s WSPT order: \( j < k \) iff \( \frac{w_j^2}{p_j} > \frac{w_k^2}{p_k} \) and in case of ties if \( d_j < d_k \) (\( d_j > d_k \))
Publications