Planning and Scheduling by Logic-Based Benders Decomposition

John Hooker
Carnegie Mellon University

CORS/INFORMS, Banff, May 2004

Outline

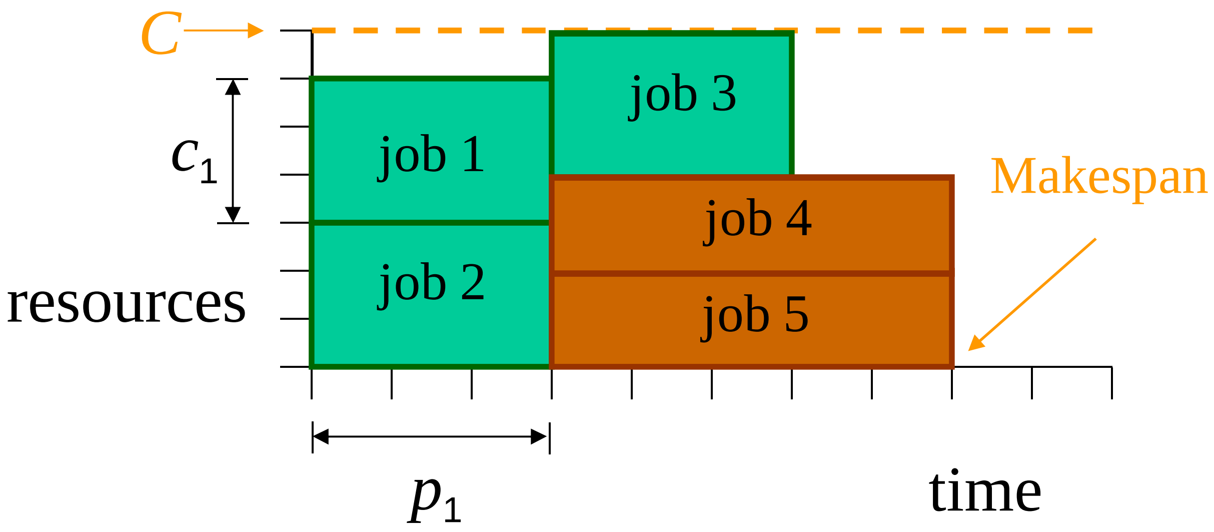
- The problem
- MILP models
- Constraint programming model
- Logic-based Benders approach
    - Basic idea
    - Previous work
    - Min cost
    - Min makespan
    - Min tardiness
- Computational results

The Problem

- Allocate jobs (tasks) to machines (facilities).
- Schedule jobs on each machine.
    - Subject to release times & deadlines.
    - Machines may run at different speeds and incur different costs.
- Cumulative scheduling
    - Several jobs may run simultaneously on a machine.
    - But total resource consumption must never exceed limit.

Cumulative Scheduling

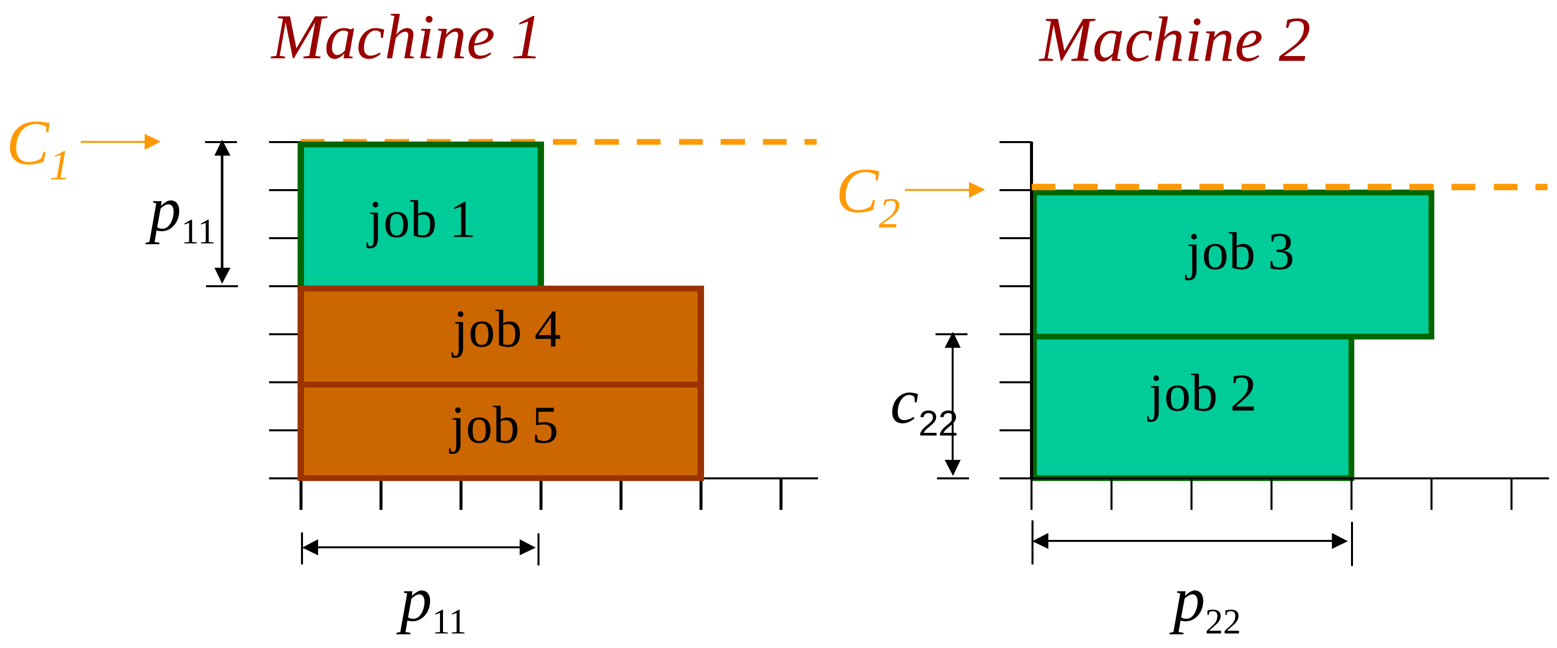
p_j = processing time of job j
c_j = rate of resource consumption of job j
C = resources available
r_j, d_j = release time & deadline for job j



Total resource consumption $\leq C$ at all times.

Multiple-machine cumulative scheduling

p_{ij} = processing time of job j on machine i
c_{ij} = resource consumption of job j on machine i
C_i = resources available on machine i



Total resource consumption ≤ C_i at all times.

Some Possible Objectives

Minimize cost = $\sum_{ij} g_{y_j j}$

machine assigned to job *j*

Fixed cost of assigning job *j* to machine y_j

Minimize makespan = $\max_{ij}\{t_j + p_{y_j j}\}$

Start time of job *j*

Minimize tardiness = $\sum_{ij}\left(t_j + p_{y_j j} - d_j\right)^+$

$\alpha^+ = \max\{0, \alpha\}$

Due date for job *j*

Discrete Time MILP Model

(Minimize Cost)

$$
\begin{aligned}
\text{min} \quad & \sum_{ijt} g_{ij} x_{ijt} \\
\text{subject to} \quad & \sum_{it} x_{ijt} = 1, \quad \text{all } j \\
& \sum_{j} \sum_{\substack{t' \\ t-p_{ij}<t'\le t}} c_{ij} x_{ijt'} \le C_i, \quad \text{all } i,t \\
& x_{ijt} = 0, \quad \text{all } j,t \text{ with } d_j - p_{ij} < t \\
& x_{ijt} = 0, \quad \text{all } j,t \text{ with } t > N - p_{ij} + 1 \\
& x_{ijt} \in \{0,1\}
\end{aligned}
$$

= 1 if job j starts at time point t on machine i $\quad (t = 1,\ldots,N)$

Job j starts at one time on one machine

Jobs underway at time t consume $\le C_i$ in resources

Jobs observe time windows

Discrete Event MILP Model

Idea: Türkay & Grossmann

= 1 if event k is start of job j on machine i $(k = 1,\ldots,2N)$

= 1 if event k is end of job

$$
\begin{aligned}
\text{min} \quad & \sum_{ijk} g_{ij} x_{ijk} \\
\text{subject to} \quad & \sum_{ik} x_{ijk} = 1, \quad \sum_{ik} y_{ijk} = 1, \quad \text{all } j \\
& \sum_{ij} x_{ijk} + y_{ijk} = 1, \quad \text{all } k \\
& \sum_{k} x_{ijk} = \sum_{k} y_{ijk}, \quad \text{all } i, j \\
& t_{i,k-1} \le t_{ik} \\
& \text{continued}\ldots
\end{aligned}
$$

Each job is assigned to one machine and starts once and ends once

Start time of event k (disaggregated by machine)

Events in chronological order

Release date
and deadline

Finish time of job *j*
(disaggregated by machine)

Definition of finish time

$$0 \le t_{ik}, \quad f_{ij} \le \boxed{d_j}, \quad \text{all } i, j, k$$

$$t_{ik} + p_{ij} x_{ik} - M(1 - x_{ijk}) \le f_{ij} \le t_{ik} + p_{ik} x_{ijk} + M(1 - x_{ijk}), \quad \text{all } i, j, k$$

$$t_{ik} - M(1 - y_{ijk}) \le f_{ij} \le t_{ik} + M(1 - y_{ijk}), \quad \text{all } i, j, k$$

$$R_{ik} \le C_i, \quad \text{all } i, k$$

Resource limit

$$R_{i1} = R_{i1}^s, \quad R_{ik}^s = \sum_j c_{ij} x_{ijk}, \quad R_{ik}^f = \sum_j c_{ij} y_{ijk}, \quad \text{all } i, k$$

$$R_{ik}^s + R_{i,k-1} - R_{ik}^f = R_{ik}, \quad \text{all } i, k$$

$$x_{ijk}, y_{ijk} \in \{0,1\}$$

Calculation of resource
consumption on
machine *i* at time of
each event

Constraint Programming Model

$$\text{cumulative}\begin{pmatrix} (t_1,\ldots,t_n) \\ (p_1,\ldots,p_n) \\ (c_1,\ldots,c_n) \\ C \end{pmatrix}$$

is equivalent to

$$\sum_{\substack{j \\ t_j \le t < t_j + p_{ij}}} c_j \le C \ , \quad \text{all}\, t$$

Schedules jobs at times t_1, …, t_n so as to observe resource constraint.

Edge-finding algorithms, etc., reduce domains of t_j.

Minimize Cost: CP Model

y_j = machine assigned to job j

start times of jobs
assigned to machine i

$$\min \quad \sum_j g_{y_j j}$$

$$\text{subject to} \quad \text{cumulative}\begin{pmatrix} (t_j \mid y_j = i) \\ (p_{ij} \mid y_j = i) \\ (c_{ij} \mid y_j = i) \\ C_i \end{pmatrix}, \quad \text{all } i$$

$$r_j \le t_j \le d_j - p_{y_j j}, \quad \text{all } j$$

Observe time windows

Observe resource limit
on each machine

This is how it looks in OPL Studio…

```
[Declarations]
DiscreteResource machine [i in Machines] (Limit [i]);
AlternativeResources mset(machine);
Activity sched [ j in Jobs];

minimize
   sum( j in Jobs) cost [ j ]
subject to {
   forall( j in Jobs) {
      sched [ j ] requires(jobm[i,j ].resource) mset;
      forall(i in Machines)
        activityHasSelectedResource(sched [ j ],mset,machine[ j ])
           <=> sched [ j ].duration = jobm[i,j ].duration &
                     cost [ j ] = jobm[i,j ].cost;
      sched [ j ].start >= job[ j ].release;
      sched [ j ].end <= job[ j ].deadline;
   };
};
search {
   assignAlternatives;
   setTimes;
};
```

enforces cumulative

assigns jobs to machines

defines resource requirements

determines cost and durations on the assigned machine

time windows

invokes specialized search procedure (needed for good performance)

Logic-Based Benders: Basic Idea

- **Decompose** problem into

assignment *assign jobs to machines*	+	resource-constrained scheduling *schedule jobs on each machine*

- Use logic-based Benders to link these.
- Solve: master problem with **MILP**
  -- good at resource allocation
  subproblem with **Constraint Programming**
  -- good at scheduling
- Generate Benders cuts from subproblem solutions, and add them to master problem.

Previous Work

1995 (JH & Yan) – Apply logic-based Benders to circuit verification.

- Better than BDDs when circuit contains error.

1995, 2000 (JH) – Formulate general logic-based Benders.

- Specialized Benders cuts must be designed for each problem class.
- Branch-and-check proposed.

2001 (Jain & Grossmann) – Apply logic-based Benders to multiple-machine scheduling using CP/MILP.

- Substantial speedup wrt CPLEX, ILOG Scheduler.
- But… easy problem for Benders approach

2001 (Thorsteinsson) – Apply branch-and-check to CP/MILP.

- 1-2 orders of magnitude speedup on multiple machine scheduling.

2002 (JH, Ottosson) – Apply logic-based Benders to SAT, IP.

Today – Apply logic-based Benders to resource-constrained planning/scheduling problems.

- Multiple machines, parallel processing on each machine with resource constraint (cumulative scheduling)
- Min cost, makespan, and tardiness.

Also:

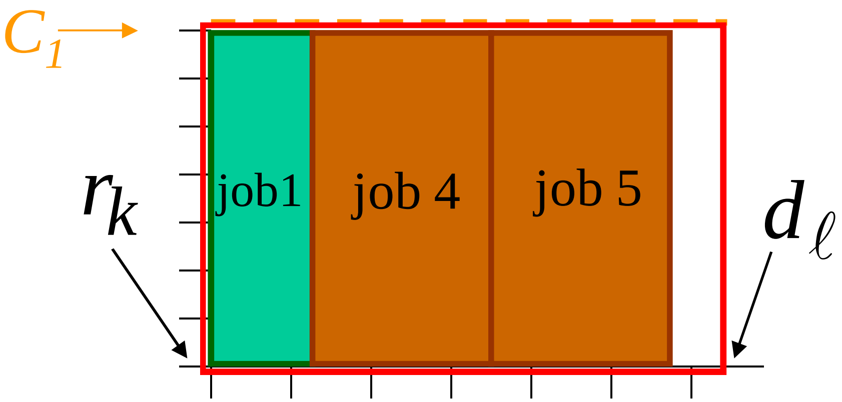
2001 (Eremin & Wallace) - Classical Benders + CP

Yesterday (Cambazard & Hladik) – Real-time task allocation & scheduling

Minimize Cost: Logic-Based Benders

Master Problem: Assign jobs to machines

$$
\begin{array}{ll}
\text{min} & \sum\limits_{ij} g_{ij} x_{ij} \\
\text{subject to} & \sum\limits_{i} x_{ij} = 1, \quad \text{all } j \\
& \sum\limits_{\substack{j \\ r_j \geq r_k \\ d_j \leq d_\ell}} p_{ij} c_{ij} x_{ij} \leq C_i (d_\ell - r_k), \quad \text{all } i, \text{all distinct } r_k, d_\ell \\
& \text{Benders cuts}
\end{array}
$$



Relaxation of subproblem:
“Area” of jobs in time window $[r_k, d_\ell]$ must fit.

Subproblem: Schedule jobs assigned to each machine
Solve by constraint programming

solution of master problem

$$\left\{ \begin{array}{l} \text{cumulative}\begin{pmatrix} (t_j \mid \boxed{\bar{x}_{ij}} = 1) \\ (p_{ij} \mid \bar{x}_{ij} = 1) \\ (c_{ij} \mid \bar{x}_{ij} = 1) \\ C_i \end{pmatrix} \\ r_j \le t_j \le d_j \end{array} \right\}, \quad \text{all } i$$

Let J_{ih} = set of jobs assigned to machine i in iteration h.

If subproblem i is infeasible, solution of subproblem dual is a proof that not all jobs in J_{ih} can be assigned to machine i.

This provides the basis for a (trivial) Benders cut.

Master Problem with Benders Cuts
Solve by MILP

$$
\begin{array}{lll}
\min & \sum_{ij} c_{ij} x_{ij} & \\
\text{subject to} & \sum_{i} x_{ij} = 1, & \text{all } j \\
& \sum_{\substack{j \\ r_j \le r_k \\ d_j \le d_\ell}} p_{ij} r_{ij} x_{ij} \le C_i (d_\ell - r_k), & \text{all } i, \text{all distinct } r_k, d_\ell \\
& \sum_{j \in J_{ih}} (1 - x_{ij}) \ge 1, & \text{all } i, h \\
& x_{ij} \in \{0,1\} &
\end{array}
$$

Benders cuts

Important observation: Putting a **relaxation of subproblem** in the master problem is essential for success.

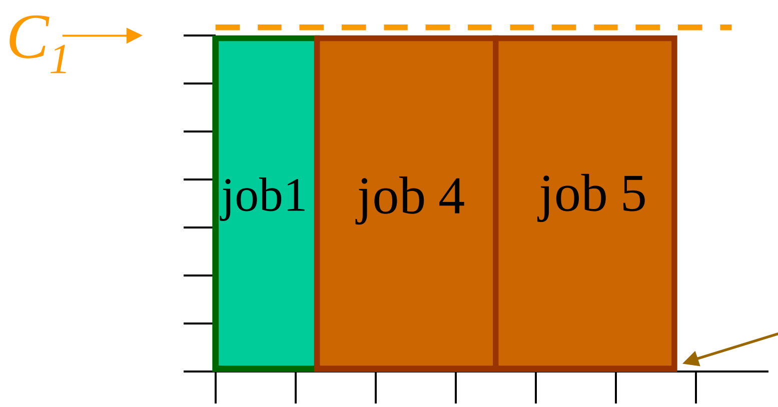
Min cost problem is particularly easy for logic-based decomposition:

	Min cost	*Min makespan, tardiness*
Objective function	Computed in master problem, which yields tighter bounds for MILP	Available only thru Benders cuts.
Subproblem	Feasibility problem, simple Benders cuts	Optimization problem (harder for CP), more interesting cuts
Relaxation	Trivial	More interesting, nice duality with cuts

Minimize Makespan: Logic-Based Benders

Master Problem: Assign jobs to machines

$$
\begin{array}{lll}
\text{min} & m & \leftarrow \text{makespan} \\
\text{subject to} & \sum_i x_{ij} = 1, & \text{all } j \\
 & m \geq \dfrac{1}{C_i} \sum_j p_{ij} c_{ij} x_{ij}, & \text{all } i \\
 & \text{Benders cuts} &
\end{array}
$$



Relaxation of subproblem:
“Area” of jobs provides
lower bound on makespan.

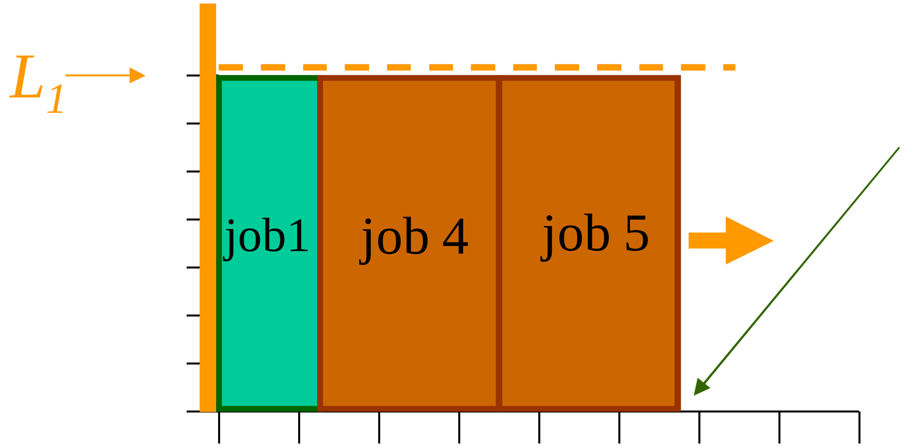
Subproblem: Schedule jobs assigned to each machine
Assume same time window for all jobs
Solve by constraint programming

$$
\begin{array}{ll}
\text{min} & m \\
\text{subject to} & \left\{ \begin{array}{l} m \ge t_j + d_{ij}, \quad \text{all } j \\ \text{cumulative}\begin{pmatrix} (t_j \mid \bar{x}_{ij} = 1) \\ (p_{ij} \mid \bar{x}_{ij} = 1) \\ (c_{ij} \mid \bar{x}_{ij} = 1) \\ C_i \end{pmatrix} \\ 0 \le t_j \le d_0, \quad \text{all } j \end{array} \right\}, \quad \text{all } i
\end{array}
$$

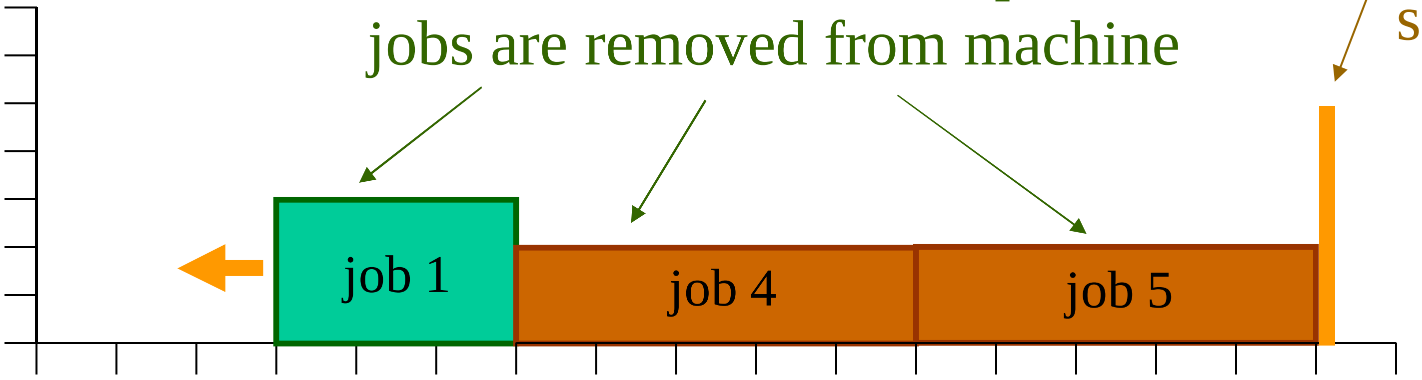
Let J_{ih} = set of jobs assigned to machine i in iteration h.

We get a Benders cut even when subproblem is feasible.

Duality of Linear Relaxation and Linear Benders Cuts



Relaxation:
Lower bound on makespan



Benders cut:
Lower bounds on makespan as jobs are removed from machine

Minimum makespan for machine *i* in subproblem

Lemma.

Let m^* = min makespan for an n-job problem on machine i

m' = min makespan when jobs 1,…,s are removed.

Then

$$m' \geq m^* - \sum_{j=1}^{s} p_{ij}$$

Idea: Consider solution of problem with jobs 1,…,s removed. Obtain a solution for the original problem by adding jobs 1,…,s sequentially at the end (starting at time m'). Lemma holds whether this solution is feasible (completes before d_0) or infeasible.

Lemma is false when deadlines differ.

Master Problem: Assign jobs to machines
Solve by MILP

$$\begin{array}{ll} \min & m \\ \text{subject to} & \sum_i x_{ij} = 1, \quad \text{all } j \\ & m \geq \frac{1}{C_i} \sum_j p_{ij} c_{ij} x_{ij}, \quad \text{all } i \\ & m \geq m^*_{hi} - \sum_{j \in J_{ik}} (1 - x_{ij}) p_{ij}, \quad \text{all } i, h \\ & x_{ij} \in \{0,1\} \end{array}$$

Relaxation

Benders cuts

Makespan on machine *i* in iteration *h*

Minimize Tardiness: Logic-Based Benders

Master Problem: Assign jobs to machines

tardiness

$$
\begin{array}{ll}
\min & \boxed{T} \\
\text{s.t.} & \sum_i x_{ij} = 1, \quad \text{all } j \\
& \text{relaxation of subproblem} \\
& \text{Benders cuts}
\end{array}
$$

Relaxation of subproblem

Lemma. Consider a min tardiness problem that schedules jobs 1, …, n on machine i, where $d_1 \le \ldots \le d_n$. The min tardiness T^* is bounded below by

$$L = \sum_{k=1}^{n} L_k$$

where

$$L_k = \left(\frac{1}{C_i} \sum_{j=1}^{k} p_{i\pi_i(j)} c_{i\pi_i(j)} - d_k \right)^+$$

and π is a permutation of 1, …, n such that

$$p_{\pi_i(1)} c_{\pi_i(1)} \le \cdots \le p_{\pi_i(n)} c_{\pi_i(n)}$$

From the lemma, we can write the relaxation

$$T \geq \sum_{i} \sum_{k=1}^{n} T'_{ik} x_{ik}$$

where $T'_{ik} \geq \frac{1}{C_i} \sum_{j=1}^{k} p_{i\pi_i(j)} c_{i\pi_i(j)} x_{i\pi_i(j)} - d_k$

To linearize this, we write $T \geq \sum_{i} \sum_{k=1}^{n} T_{ik}$

and $T_{ik} \geq \frac{1}{C_i} \sum_{j=1}^{k} p_{i\pi_i(j)} c_{i\pi_i(j)} x_{i\pi_i(j)} - d_k - (1 - x_{ik}) M_{ik}$

where $T_{ik} \geq 0, \quad M_{ik} = \frac{1}{C_i} \sum_{j=1}^{k} p_{i\pi_i(j)} c_{i\pi_i(j)} - d_k$

Benders cuts

Lemma.

Let T^* = min tardiness for an n-job problem on machine i
T' = min tardiness when jobs 1,…,s are removed.

Then

$$T' \geq T^* - n \sum_{j=1}^{s} p_{ij}$$

Idea: Consider solution of problem with jobs 1,…,s removed. Obtain a feasible solution for the original problem by adding jobs 1,…,s sequentially at the beginning and pushing the other jobs forward.

From the lemma, we have for each iteration *h* the Benders cut

Min tardiness on machine *i* in subproblem

$$
\begin{aligned}
&T \geq \sum_i T_{hi} \\
&T_{hi} \geq T^*_{hi} - |J_{hi}| \sum_{j \in J_{hi}} (1 - x_{ij}) p_{ij}, \quad \text{all } i \\
&T_{hi} \geq 0
\end{aligned}
$$

Computational Results

- Random problems on 2, 3, 4 machines.
- Machines run at different speeds.
- All jobs have same time windows.
- Tardiness problems: still in progress.
- Implement with OPL Studio
    - CPLEX for MILP
    - ILOG Scheduler for CP

Min cost, 2 machines

Computation time in seconds
Average of 5 instances shown

Jobs	MILP*	CP	Benders
10	1.9	0.14	0.09
12	199	2.2	0.06
14	1441	79	0.04
16	3604+	1511	1.1
18		7200+	7.0
20			85

*Discrete time model only. Discrete event model very hard to solve.

+ At least one problem in the 5 exceeded 7200 sec (2 hours)

Min cost, 3 machines

Computation time in seconds
Average of 5 instances shown

Jobs	MILP*	CP	Benders
10	0.9	0.13	0.37
12	797	2.6	0.55
14	114	35	0.34
16	678*	1929	4.5
18		7200+	14.6
20			2.9
22			23
24			53

*CPLEX ran out of memory on 1 or more problems.

+ At least one problem in the 5 exceeded 7200 sec (2 hours)

Min cost, 4 machines

Computation time in seconds
Average of 5 instances shown

Jobs	MILP*	CP	Benders
10	2.0	0.10	0.6
12	7.2	1.4	4.0
14	158	72	2.8
16	906*	344	0.8
18		6343+	5.2
20			2.6
22			22
24			114

*CPLEX ran out of memory on 1 or more problems.

+ At least one problem in the 5 exceeded 7200 sec (2 hours)

Min makespan, 2 machines

Average (sec) of 5 instances shown

Jobs	CP	Benders
10	0.8	0.08
12	4.0	0.4
14	299	7.8
16	3737	30
18	7200+	461

+ At least one problem in the 5 exceeded 7200 sec (2 hours)

Min makespan, 3 machines

Average (sec) of 5 instances shown

Jobs	CP	Benders
10	0.9	0.06
12	7.5	0.3
14	981	0.7
16	4414	6.5
18	7200+	13.3
20		34
22		1509

+ At least one problem in the 5 exceeded 7200 sec (2 hours)

Min makespan, 4 machines

Average (sec) of 5 instances shown

Jobs	CP	Benders
10	0.07	0.09
12	1.9	0.09
14	524	0.8
16	3898	0.9
18	7200+	13.9
20		25
22		472

+ At least one problem in the 5 exceeded 7200 sec (2 hours)

Remarks

- Scheduling subproblem dominates as number of jobs per machine increases.
- Scheduling tends to be easier with precedence and other side constraints

Min cost & makespan, 2 machines

With precedence constraints

Jobs	Min cost sec	Min makespan sec	Min makespan value
12	0.02	0.2	17
14	0.05	0.3	13
16	0.5	1.6	27
18	0.02	25	27
20	0.9	0.7	37
22	0.7	600*	26-27
24	7.7	13	32
26	2.1	442	37
28	21	600*	35-37
30	73	600*	50-53

*Terminated at 600 sec

Min cost & makespan

Average (sec) of 5 instances shown

Jobs	Machines	Min cost	Min makespan
10	2	0.1	0.2
15	3	0.7	1.6
20	4	50	13
25	5	2.9	213
30	6	4.8	2075
35	7	128	
40	8	976	

Future Research

- Implement branch-and-check for Benders problem.
- Exploit dual information from the subproblem solution process.
- Explore other problem classes.
    - Min makespan with different time windows
    - Vehicle routing
    - Sequence-dependence setup times
    - Integrated long-term and short-term planning/scheduling