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

\[ p_j = \text{processing time of job } j \]
\[ c_j = \text{rate of resource consumption of job } j \]
\[ C = \text{resources available} \]
\[ r_j, d_j = \text{release time \\& deadline for job } j \]

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 \( \leq C_i \) at all times.
Some Possible Objectives

Minimize cost = \( \sum_{ij} g_{y_j} \)

machine assigned to job j

Fixed cost of assigning job j to machine \( y_j \)

Minimize makespan = \( \max_{ij} \{ t_j + p_{y_j} \} \)

Start time of job j

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

\( \alpha^+ = \max\{0, \alpha\} \)

Due date for job j
Discrete Time MILP Model
(Minimize Cost)

\[\begin{align*}
\text{min} & \quad \sum_{ijt} g_{ij} x_{ijt} \\
\text{subject to} & \quad \sum_{it} x_{ijt} = 1, \quad \text{all } j \\
& \quad \sum_j \sum_{t'} c_{ij} x_{ijt'} \leq C_i, \quad \text{all } i, t \\
& \quad t - p_{ij} < t' \leq t \\
& \quad x_{ijt} = 0, \quad \text{all } j, t \text{ with } d_j - p_{ij} < t \\
& \quad x_{ijt} = 0, \quad \text{all } j, t \text{ with } t > N - p_{ij} + 1 \\
x_{ijt} & \in \{0,1\}
\end{align*}\]

Job \( j \) starts at one time on one machine

Jobs underway at time \( t \) consume \( \leq C_i \) in resources

Jobs observe time windows
Discrete Event MILP Model

Idea: Türkay & Grossmann

\[
\begin{align*}
\text{min} & \quad \sum_{ijk} g_{ijk} x_{ijk} \\
\text{subject to} & \quad \sum_{ik} x_{ijk} = 1, \quad \sum_{ik} y_{ijk} = 1, \quad \text{all } j \\
& \quad \sum_{ij} x_{ijk} + y_{ijk} = 1, \quad \text{all } k \\
& \quad \sum_{k} x_{ijk} = \sum_{k} y_{ijk}, \quad \text{all } i, j \\
& \quad t_{i,k-1} \leq t_{ik} \\
& \quad \text{Events in chronological order continued...}
\end{align*}
\]

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

= 1 if event \( k \) is end of job

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

Start time of event \( k \) (disaggregated by machine)
Release date and deadline

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

Definition of finish time

Resource limit

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

\begin{align*}
0 & \leq t_{ik}, \quad f_{ij} \leq d_j, \quad \text{all } i, j, k \\
t_{ik} + p_{ij} x_{ik} - M (1 - x_{ijk}) & \leq f_{ij} \leq t_{ik} + p_{ik} x_{ijk} + M (1 - x_{ijk}), \quad \text{all } i, j, k \\
t_{ik} - M (1 - y_{ijk}) & \leq f_{ij} \leq t_{ik} + M (1 - y_{ijk}), \quad \text{all } i, j, k \\
R_{ik} & \leq C_i, \quad \text{all } i, k \\
R_{i1}^{s} & = 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}^{f} - R_{ik}^{f} & = R_{ik}, \quad \text{all } i, k \\
x_{ijk}, y_{ijk} & \in \{0,1\}
\end{align*}
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

\[
\sum_j c_j \leq C, \quad \forall t
\]

\[
t_j \leq t < t_j + p_{ij}
\]

Schedules jobs at times \(t_1, \ldots, t_n\) so as to observe resource constraint.

Edge-finding algorithms, etc., reduce domains of \(t_j\).
Minimize Cost: CP Model

\[
\begin{align*}
\text{min} & \quad \sum_j g_{y_j,j} \\
\text{subject to} \quad \text{cumulative} & \quad \left\{ \begin{array}{l}
(t_j \mid y_j = i) \\
(p_{ij} \mid y_j = i) \\
(c_{ij} \mid y_j = i) \\
C_i
\end{array} \right\}, \quad \text{all } i \\
r_j \leq t_j \leq d_j - p_{y_j,j}, \quad \text{all } j
\end{align*}
\]

\(y_j = \text{machine assigned to job } j\)

\(\text{start times of jobs assigned to machine } i\)

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* + *resource-constrained scheduling*
  
  *assign jobs to machines* + *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.

  • 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

\[ \min \sum_{i,j} g_{ij} x_{ij} \]
subject to
\[ \sum_{i} x_{ij} = 1, \text{ all } j \]
\[ \sum_{j} p_{ij} c_{ij} x_{ij} \leq C_i (d_\ell - r_k), \text{ all } i, \text{ all distinct } r_k, d_\ell \]

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

\[
\begin{cases}
(t_j \mid \overline{x}_{ij} = 1) \\
(p_{ij} \mid \overline{x}_{ij} = 1) \\
(c_{ij} \mid \overline{x}_{ij} = 1) \\
C_i
\end{cases}
\]  

\[
r_j \leq t_j \leq d_j
\]

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

\[
\begin{align*}
\text{min} & \quad \sum_{ij} c_{ij}x_{ij} \\
\text{subject to} & \quad \sum_i x_{ij} = 1, \quad \text{all } j \\
& \quad \sum_j p_{ij}r_{ij}x_{ij} \leq C_i (d_\ell - r_k), \quad \text{all } i, \text{ all distinct } r_k, d_\ell \\
& \quad r_j \leq r_k \\
& \quad d_j \leq d_\ell \\
& \quad \sum_{j \in J_{ih}} (1 - x_{ij}) \geq 1, \quad \text{all } i, h \\
x_{ij} & \in \{0,1\}
\end{align*}
\]
**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:

<table>
<thead>
<tr>
<th></th>
<th>Min cost</th>
<th>Min makespan, tardiness</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Objective function</strong></td>
<td>Computed in master problem, which yields tighter bounds for MILP</td>
<td>Available only thru Benders cuts.</td>
</tr>
<tr>
<td><strong>Subproblem</strong></td>
<td>Feasibility problem, simple Benders cuts</td>
<td>Optimization problem (harder for CP), more interesting cuts</td>
</tr>
<tr>
<td><strong>Relaxation</strong></td>
<td>Trivial</td>
<td>More interesting, nice duality with cuts</td>
</tr>
</tbody>
</table>
Minimize Makespan: Logic-Based Benders

Master Problem: Assign jobs to machines

\[
\begin{align*}
\text{min} & \quad m \\
\text{subject to} & \quad \sum_{i} x_{ij} = 1, \quad \text{all } j \\
& \quad m \geq \frac{1}{C_i} \sum_{j} p_{ij} c_{ij} x_{ij}, \quad \text{all } i \\
\end{align*}
\]

Benders cuts

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

\[ \text{min} \quad m \]

subject to

\[
\begin{align*}
\{ & m \geq t_j + d_{ij}, \quad \text{all } j \\
& \text{cumulative } \begin{cases} 
(t_j | \bar{x}_{ij} = 1) \\
(p_{ij} | \bar{x}_{ij} = 1) \\
(c_{ij} | \bar{x}_{ij} = 1)
\end{cases} \\
& 0 \leq t_j \leq d_0, \quad \text{all } j 
\} \text{, all } i
\]

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

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 \( m^* = \min \text{ makespan for an } n\text{-job problem on machine } i \)

\( m' = \min \text{ makespan when jobs } 1, \ldots, s \text{ are removed.} \)

Then

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

Idea: Consider solution of problem with jobs 1, \ldots, s removed. Obtain a solution for the original problem by adding jobs 1, \ldots, 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

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

Relaxation
Benders cuts
Makespan on machine \(i\) in iteration \(h\)
Minimize Tardiness: Logic-Based Benders

\textit{Master Problem: Assign jobs to machines}

\[
\begin{align*}
\text{min} & \quad T \\
\text{s.t.} & \quad \sum_{i} x_{ij} = 1, \quad \text{all } j \\
& \text{relaxation of subproblem} \\
& \text{Benders cuts}
\end{align*}
\]
Relaxation of subproblem

**Lemma.** Consider a min tardiness problem that schedules jobs 1, ..., \( n \) on machine \( i \), where \( d_1 \leq ... \leq d_n \). 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 \( \pi \) is a permutation of 1, ..., \( n \) such that

\[
p_{\pi_i(1)} c_{\pi_i(1)} \leq \cdots \leq p_{\pi_i(n)} c_{\pi_i(n)}
\]
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 $i$
\[ T' = \min \text{ tardiness when jobs 1,\ldots,s are removed.} \]

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

Idea: Consider solution of problem with jobs 1,\ldots,s removed. Obtain a feasible solution for the original problem by adding jobs 1,\ldots,s sequentially at the beginning and pushing the other jobs forward.
From the lemma, we have for each iteration $h$ the Benders cut

$$T \geq \sum_{i} T_{hi}$$

$$T_{hi} \geq T_{hi}^* - |J_{hi}| \sum_{j \in J_{hi}} (1 - x_{ij}) p_{ij}, \quad \text{all } i$$

$$T_{hi} \geq 0$$

Min tardiness on machine $i$ in subproblem
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

<table>
<thead>
<tr>
<th>Jobs</th>
<th>MILP*</th>
<th>CP</th>
<th>Benders</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1.9</td>
<td>0.14</td>
<td>0.09</td>
</tr>
<tr>
<td>12</td>
<td>199</td>
<td>2.2</td>
<td>0.06</td>
</tr>
<tr>
<td>14</td>
<td>1441</td>
<td>79</td>
<td>0.04</td>
</tr>
<tr>
<td>16</td>
<td>3604+</td>
<td>1511</td>
<td>1.1</td>
</tr>
<tr>
<td>18</td>
<td>7200+</td>
<td>7.0</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
<td></td>
<td>85</td>
</tr>
</tbody>
</table>

*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

<table>
<thead>
<tr>
<th>Jobs</th>
<th>MILP*</th>
<th>CP</th>
<th>Benders</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.9</td>
<td>0.13</td>
<td>0.37</td>
</tr>
<tr>
<td>12</td>
<td>797</td>
<td>2.6</td>
<td>0.55</td>
</tr>
<tr>
<td>14</td>
<td>114</td>
<td>35</td>
<td>0.34</td>
</tr>
<tr>
<td>16</td>
<td>678*</td>
<td>1929</td>
<td>4.5</td>
</tr>
<tr>
<td>18</td>
<td>7200+</td>
<td></td>
<td>14.6</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td></td>
<td>2.9</td>
</tr>
<tr>
<td>22</td>
<td></td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td></td>
<td>53</td>
<td></td>
</tr>
</tbody>
</table>

*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**

<table>
<thead>
<tr>
<th>Jobs</th>
<th>MILP*</th>
<th>CP</th>
<th>Benders</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2.0</td>
<td>0.10</td>
<td>0.6</td>
</tr>
<tr>
<td>12</td>
<td>7.2</td>
<td>1.4</td>
<td>4.0</td>
</tr>
<tr>
<td>14</td>
<td>158</td>
<td>72</td>
<td>2.8</td>
</tr>
<tr>
<td>16</td>
<td>906*</td>
<td>344</td>
<td>0.8</td>
</tr>
<tr>
<td>18</td>
<td>6343+</td>
<td></td>
<td>5.2</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td></td>
<td>2.6</td>
</tr>
<tr>
<td>22</td>
<td></td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td></td>
<td>114</td>
<td></td>
</tr>
</tbody>
</table>

*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

<table>
<thead>
<tr>
<th>Jobs</th>
<th>CP</th>
<th>Benders</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.8</td>
<td>0.08</td>
</tr>
<tr>
<td>12</td>
<td>4.0</td>
<td>0.4</td>
</tr>
<tr>
<td>14</td>
<td>299</td>
<td>7.8</td>
</tr>
<tr>
<td>16</td>
<td>3737</td>
<td>30</td>
</tr>
<tr>
<td>18</td>
<td>7200+</td>
<td>461</td>
</tr>
</tbody>
</table>

+ At least one problem in the 5 exceeded 7200 sec (2 hours)
### Min makespan, 3 machines

Average (sec) of 5 instances shown

<table>
<thead>
<tr>
<th>Jobs</th>
<th>CP</th>
<th>Benders</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.9</td>
<td>0.06</td>
</tr>
<tr>
<td>12</td>
<td>7.5</td>
<td>0.3</td>
</tr>
<tr>
<td>14</td>
<td>981</td>
<td>0.7</td>
</tr>
<tr>
<td>16</td>
<td>4414</td>
<td>6.5</td>
</tr>
<tr>
<td>18</td>
<td>7200+</td>
<td>13.3</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>34</td>
</tr>
<tr>
<td>22</td>
<td></td>
<td>1509</td>
</tr>
</tbody>
</table>

+ At least one problem in the 5 exceeded 7200 sec (2 hours)
### Min makespan, 4 machines
Average (sec) of 5 instances shown

<table>
<thead>
<tr>
<th>Jobs</th>
<th>CP</th>
<th>Benders</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.07</td>
<td>0.09</td>
</tr>
<tr>
<td>12</td>
<td>1.9</td>
<td>0.09</td>
</tr>
<tr>
<td>14</td>
<td>524</td>
<td>0.8</td>
</tr>
<tr>
<td>16</td>
<td>3898</td>
<td>0.9</td>
</tr>
<tr>
<td>18</td>
<td>7200+</td>
<td>13.9</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>22</td>
<td></td>
<td>472</td>
</tr>
</tbody>
</table>

+ 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*

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

*Terminated at 600 sec*
<table>
<thead>
<tr>
<th>Jobs</th>
<th>Min cost</th>
<th>Min makespan</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>15</td>
<td>0.7</td>
<td>1.6</td>
</tr>
<tr>
<td>20</td>
<td>50</td>
<td>13</td>
</tr>
<tr>
<td>25</td>
<td>2.9</td>
<td>213</td>
</tr>
<tr>
<td>30</td>
<td>4.8</td>
<td>2075</td>
</tr>
<tr>
<td>35</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>976</td>
<td></td>
</tr>
</tbody>
</table>
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