Two Schemes for Combining Mixed Integer Programming with Constraint Programming

John Hooker

Carnegie Mellon University

IBM T. J. Watson Laboratory

September 2000

General Form of Conditional Model

When there are no x's

min	$f(x)$ [or $r(y)$]	objective function
s.t.	$p_i(y), \quad i \in I_1$	checkable constraints
	$g_i(x), \quad i \in I_2$	soluble constraints
	$q_i(y) \rightarrow h_i(x), \quad i \in I_3$	conditional constraints
	$d_i(x, y), \quad i \in I_4$	defined constraints
	$x \in X$	solution variables
	$y_j \in D_j, \quad \text{all } j$	search variables

Soluble constraint

Set of checkable constraints

Relaxation of Conditional Model at a Given Node of the Search Tree

$$
\begin{array}{ll}
\min & f(x) \\
\text{s.t.} & g_i(x), \quad i \in I_2 \\
& h_i(x), \quad \text{all } i \in I_3 \text{ for which } q_i(y) \text{ is true} \\
& \text{relaxation of } d_i(x, y), \quad i \in I_4 \\
& x \in X
\end{array}
$$

At each node of search tree:

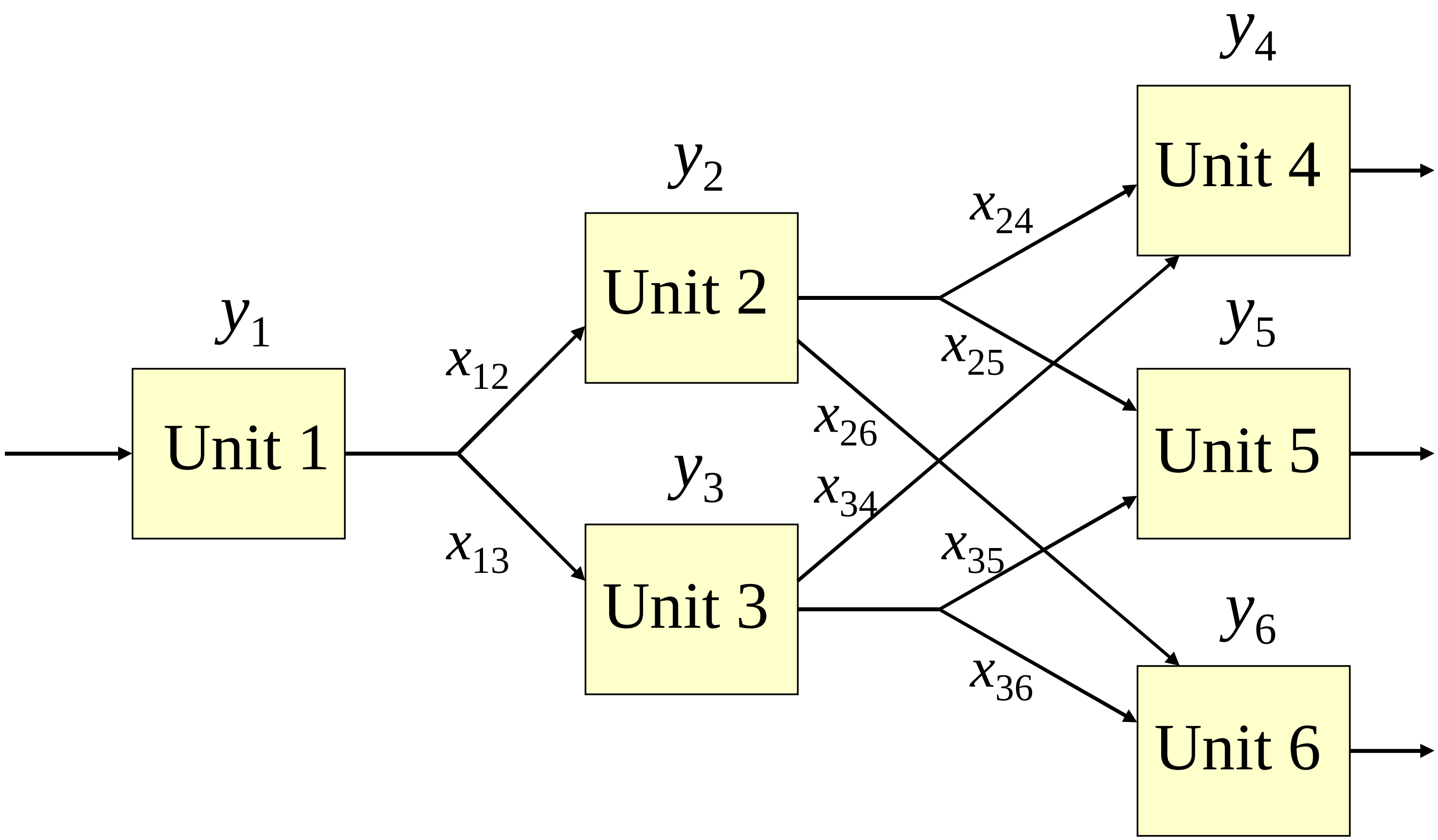
- Perform constrain propagation to reduce domains of search variables y_i and help determine truth or falsehood of $q_i(y)$'s.

- Formulate and solve relaxation, which consists of soluble constraints.

- If relaxation is feasible, try to find values for search variables that are consistent with solution variables. If this succeeds, one has feasible solution.

- Use optimal value of relaxation to prune tree if possible.

Processing Network Design



Processing Network Design

- The model uses search variables y_i to indicate the presence or absence of a unit.
- It uses conditional constraints to require that the fixed cost be incurred or the unit shut down.

$$0.6u_2 = x_{24} + x_{25}$$
$$0.4u_2 = x_{26}$$
$$0.7u_3 = x_{34}$$
$$0.3u_3 = x_{35} + x_{36}$$

$$\max \sum_i r_i u_i^{1/2} - \sum_i z_i$$

s.t.

$u = Ax$	flow thru units
$bu = Bx$	flow balance
$(y_i = \text{true}) \rightarrow (z_i = d_i)$, all i	unit is open
$(y_i = \text{false}) \rightarrow (u_i = 0)$, all i	unit is closed
$u \le c$	unit capacities
$u, x \ge 0$	

Processing Network Design

- Add don’t-be-stupid constraints to ensure that a unit is not opened unless downstream units are opened.

$$
\begin{array}{lll}
\max & \sum_i r_i u_i^{1/2} - \sum_i z_i & \\
\text{s.t.} & u = Ax & \text{flow thru units} \\
& bu = Bx & \text{flow balance} \\
& (y_i = \text{true}) \rightarrow (z_i = d_i), \text{ all } i & \text{unit is open} \\
& (y_i = \text{false}) \rightarrow (u_i = 0), \text{ all } i & \text{unit is closed} \\
& u \le c & \text{unit capacities} \\
& u, x \ge 0 & \\
& \left\{ \begin{array}{ll}
y_1 \rightarrow (y_2 \vee y_3) & y_3 \rightarrow y_4 \\
y_2 \rightarrow y_1 & y_3 \rightarrow (y_5 \vee y_6) \\
y_2 \rightarrow (y_4 \vee y_5) & y_4 \rightarrow (y_2 \vee y_3) \\
y_2 \rightarrow y_6 & y_5 \rightarrow (y_2 \vee y_3) \\
y_3 \rightarrow y_1 & y_6 \rightarrow (y_2 \vee y_3)
\end{array} \right\} & \text{don't - be - stupid}
\end{array}
$$

Processing Network Design

- Use an *inequality-or* global constraint to obtain good relaxation of disjunctive constraints.
- Use *cnf* global constraint to invoke resolution algorithm for don’t-be-stupid constraints.

$$
\begin{array}{lll}
\max & \sum_i r_i u_i^{1/2} - \sum_i z_i & \\
\text{s.t.} & u = Ax & \text{flow thru units} \\
& bu = Bx & \text{flow balance} \\
& \text{inequality-or}\left(\begin{bmatrix} y_i \\ \neg y_i \end{bmatrix}, \begin{bmatrix} z_i \ge d_i \\ u_i = 0 \end{bmatrix} \right) & \text{global constraint} \\
& u \le c & \text{unit capacities} \\
& u, x \ge 0 & \\
& \text{cnf}\begin{pmatrix} y_1 \to (y_2 \vee y_3) & y_3 \to y_4 \\ y_2 \to y_1 & y_3 \to (y_5 \vee y_6) \\ y_2 \to (y_4 \vee y_5) & y_4 \to (y_2 \vee y_3) \\ y_2 \to y_6 & y_5 \to (y_2 \vee y_3) \\ y_3 \to y_1 & y_6 \to (y_2 \vee y_3) \end{pmatrix} & \text{global constraint}
\end{array}
$$

Knapsack Problem with All-different

Original problem

$$
\begin{array}{ll}
\min & 5y_1 + 8y_2 + 4y_3 \\
\text{s.t.} & 3y_1 + 5y_2 + 2y_3 \geq 30 \\
& \text{all-different}(y_1, y_2, y_3) \\
& y_j \in \{1,2,3,4\}, \quad \text{all } j
\end{array}
$$

Solved by branching and domain reduction only.

Knapsack Problem with All-different

The *continuous* predicate adds a continuous relaxation and any desired cutting planes.

$$\begin{array}{ll} \min & \text{continuous}(5y_1 + 8y_2 + 4y_3) \\ \text{s.t.} & \text{continuous}(3y_1 + 5y_2 + 2y_3 \geq 30) \\ & \text{all-different}(y_1, y_2, y_3) \\ & y_j \in \{1,2,3,4\}, \quad \text{all } j \end{array}$$

Replace objective function with $5x_1 + 8x_2 + 4x_3$

Add $3x_1 + 5x_2 + 2x_3 \geq 30$ and link(y_j,x_j) and possibly knapsack cuts

Knapsack Problem with All-different

The *cut* predicate generates cuts in the search variables so that domain reduction is applied to cuts. *Continous* adds continuous relaxation of problem and cuts.

$$
\begin{aligned}
\min\ & z \\
\text{s.t.}\ & \text{continuous}\left(\text{cut}\begin{pmatrix} z \geq 5y_1 + 8y_2 + 4y_3 \\ 3y_1 + 5y_2 + 2y_3 \geq 30 \end{pmatrix}\right) \\
& \text{all-different}(y_1, y_2, y_3) \\
& y_j \in \{1,2,3,4\}, \quad \text{all } j
\end{aligned}
$$

Cumulative Global Constraint

Ensures that total resources consumed by jobs at any one time do not exceed C.

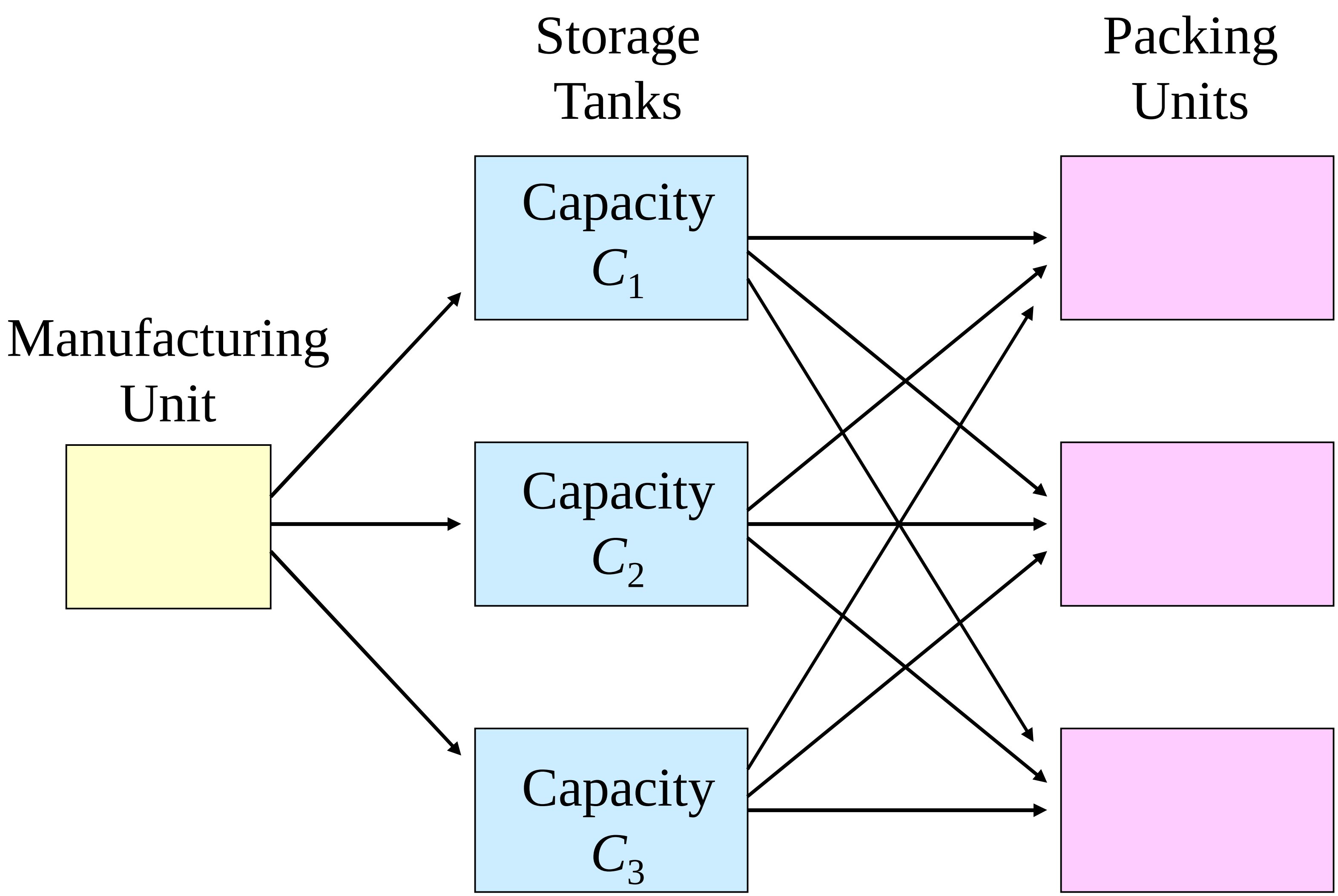
$$\text{cumulative}\big((t_1,\ldots,t_n),(d_1,\ldots,d_n),(r_1,\ldots,r_n),C\big)$$

Job start times

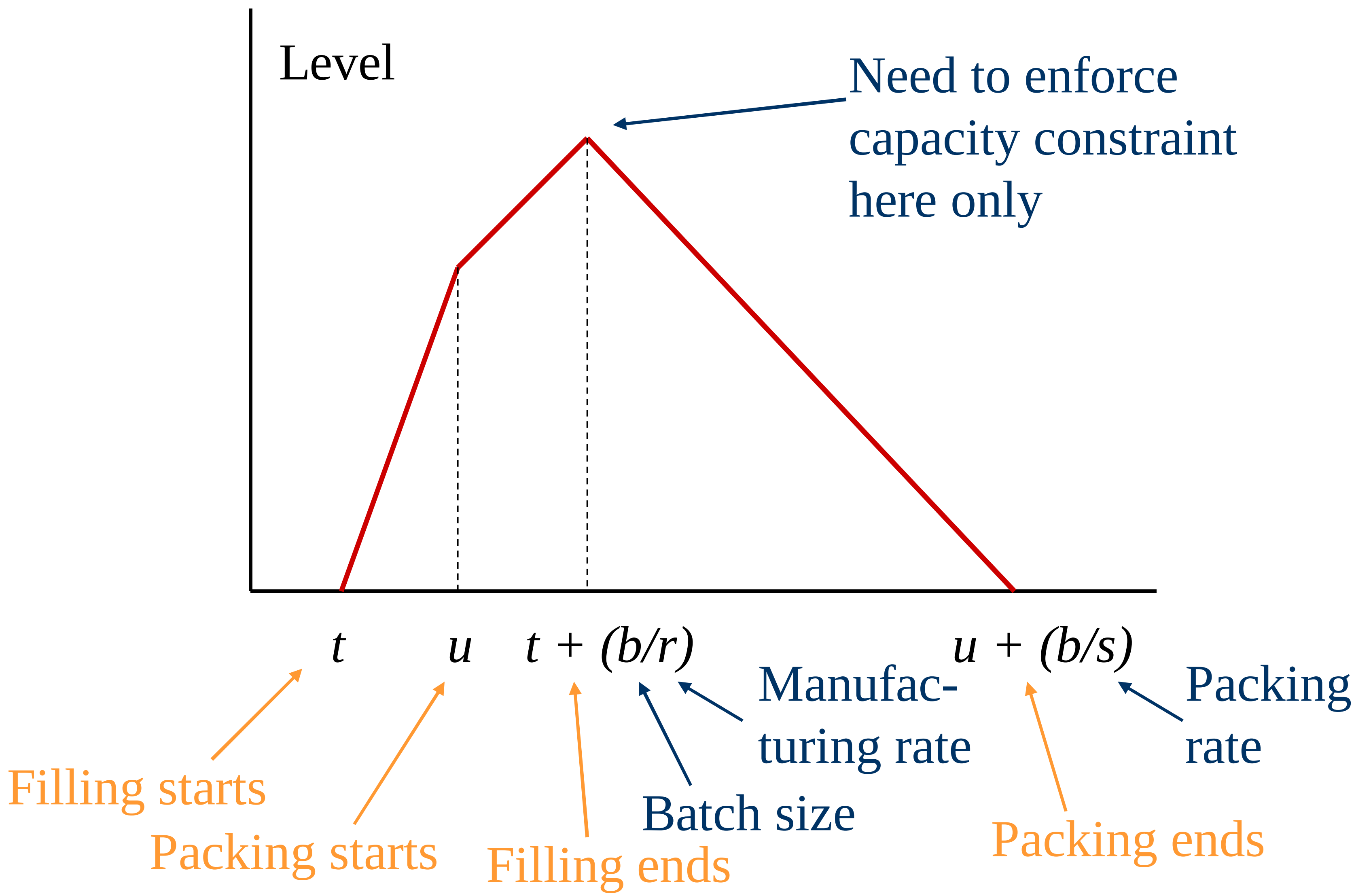
Job durations

Job resource requirements

Production Scheduling



Filling of StorageTank



$$\min\ T \quad \leftarrow \text{Makespan}$$

$$\text{s.t.}\quad T \ge u_j + \frac{b_j}{s_j}, \quad \text{all } j$$

$$t_j \ge R_j, \quad \text{all } j \quad \leftarrow \text{Job release time}$$

$$\text{cumulative}\big(t, v, (1,\ldots,1), m\big) \quad \leftarrow m \text{ storage tanks}$$

$$v_i = u_i + \frac{b_i}{s_i} - t_i, \quad \text{all } i \quad \leftarrow \text{Job duration}$$

$$b_i\left(1 - \frac{s_i}{r_i}\right) + s_i u_i \le C_i, \quad \text{all } i \quad \leftarrow \text{Tank capacity}$$

$$\text{cumulative}\left(u, \left(\frac{b_1}{s_1}, \ldots, \frac{b_n}{s_n}\right), e, p\right) \quad \leftarrow p \text{ packing units}$$

$$u_j \ge t_j \ge 0$$

Logic-Based Benders Model

- Master problem contains optimization problem, such as MILP.
- Subproblem contains constraint satisfaction problem.
- Conditional constraints determine which constraints go into the subproblem.

General Form of Benders Model

$$
\begin{aligned}
\min \quad & f(y) \\
\text{s.t.} \quad & p_i(y), \quad i \in I_1 \\
& g_i(x), \quad i \in I_2 \\
& q_i(y) \rightarrow h_i(x), \quad i \in I_3 \\
& y \in Y \\
& x_j \in D_j, \quad \text{all } j
\end{aligned}
$$

Master problem: $f(y)$; $p_i(y),\ i \in I_1$; $y \in Y$

Subproblem: $g_i(x),\ i \in I_2$; $x_j \in D_j,\ \text{all } j$

Conditional constraints: $q_i(y) \rightarrow h_i(x),\ i \in I_3$

Fixing y to $\bar{y}$ defines the subproblem (a feasibility problem):

$$
\begin{aligned}
& g_i(x), \quad i \in I_2 \\
& h_i(x), \quad \text{all } i \in I_1 \text{ for which } q_i(\bar{y}) \text{ is true} \\
& x_j \in D_j, \quad \text{all } j
\end{aligned}
$$

This produces a Benders cut $B_{\bar{y}}(y)$

The master problem is

$$
\begin{aligned}
\min \quad & f(y) \\
\text{s.t.} \quad & p_i(y), \quad i \in I_1 \\
& B_{y^k}(y), \quad k = 1, \cdots, K \\
& y \in Y
\end{aligned}
$$

Machine Scheduling
(Jain & Grossmann)

- Schedule jobs on parallel machines to minimize production cost.
- It costs C_{ij} to process job j on machine i.
- Assign jobs to machine in master problem (MILP).
- Schedule jobs on each machine in subproblem (constraint satisfaction).

Machine assigned to job j

Total cost

$$\min \sum_j C_{y_j j}$$

Release times

$$\text{s.t.} \quad t_j \geq R_j, \quad \text{all } j$$

Deadlines

$$t_j + D_{y_j j} \leq S_j, \quad \text{all } j$$

$$\text{cumulative}\Big((t_j | y_j = i), (D_{ij} \mid y_j = i), (1, \ldots, 1), 1\Big)$$

Start times of jobs assigned to machine i

Schedule jobs assigned to machine i

Problem in Benders Form

$$
\begin{aligned}
\min\ & \sum_j C_{y_j j} \\
\text{s.t.}\ & t_j \geq R_j, \quad \text{all } j \\
& t_j + D_{y_j j} \leq S_j, \quad \text{all } j \\
& \text{cumulative}\big((t_j | y_j = i), (D_{ij} \mid y_j = i), (1,\ldots,1), 1\big) \\
& t'_j \geq R_j, \quad \text{all } j \\
& t'_j + D_{y_j j} \leq S_j, \quad \text{all } j \\
& \text{link}\big(t'_j, t_j\big)
\end{aligned}
$$

Finite domain variable linked to t_j

The subproblem decomposes into a scheduling problem for each machine *i*:

$$t'_j \ge R_j, \quad \text{all } j \text{ with } \bar{y}_j = i$$
$$t'_j + D_{\bar{y}_j j} \le S_j, \quad \text{all } j \text{ with } \bar{y}_j = i$$
$$\text{cumulative}\Big((t'_j | \bar{y}_j = i), (D_{ij} \mid \bar{y}_j = i), (1,\ldots,1), 1\Big)$$

If this problem is infeasible, then at least one joib assigned machine *i* must be assigned to some other machine. This gives the Benders cut,

Disjunction ⟶ $\bigcup_{\substack{j \\ \bar{y}_j = i}} (y_j \ne i)$

The master problem becomes,

$$
\begin{aligned}
\min \quad & \sum_j C_{y_j j} \\
\text{s.t.} \quad & t_j \geq R_j, \quad \text{all } j \\
& t_j + D_{y_j j} \leq S_j, \quad \text{all } j \\
& \bigcup_{\substack{j \\ y_j^k = i}} \left(y_j \neq i \right), \quad i \in I^k,\ k = 1, \ldots, K
\end{aligned}
$$

This is easily converted to an MILP model,

$$
\begin{aligned}
\min \quad & \sum_{ij} C_{ij} x_{ij} \\
\text{s.t.} \quad & t_j \ge R_j, \quad \text{all } j \\
& t_j + \sum_i D_{ij} x_{ij} \le S_j, \quad \text{all } j \\
& \sum_{\substack{j \\ y_j^k = i}} (1 - x_{ij}) \ge 1, \quad i \in I^k,\ k = 1, \ldots, K \\
& \sum_j D_{ij} x_{ij} \le \max_j \{S_j\} - \min_j \{R_j\}, \quad \text{all } i \\
& x_{ij} \in \{0,1\}
\end{aligned}
$$

Strengthens continuous relaxation