Hybrid CP/MIP and Benders Decomposition Methods

J. Christopher Beck
Department of Mechanical & Industrial Engineering
University of Toronto
Canada
jcb@mie.utoronto.ca

CPAIOR 2016 Master Class
May 29, 2016

University of Toronto
Mechanical & Industrial Engineering

Outline

- Learn Constraint Programming in 15 minutes or less!
- Why Hybridize?
- Three Decomposition Examples
- Final Comments

Constraint Programming

- Optimization technology built around tree search and inference
  - branch-and-infer
- Like MIP but:
  - No restriction on what a constraint is
- Just as MIP lives and dies depending on the relaxation, CP lives and dies depending on inference

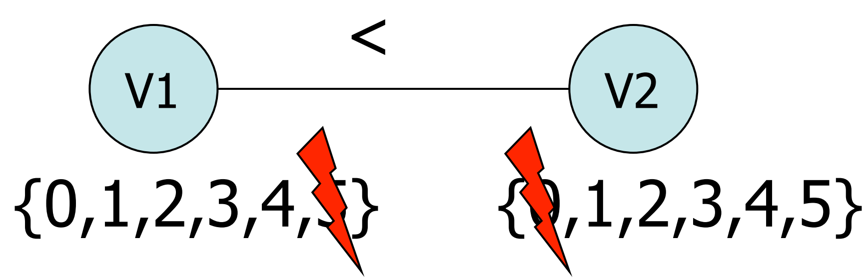
Implications

"Global Constraints"

- There is no general relaxation
- So how do you avoid enumerating the whole space?
  - Develop constraints that represent a common combinatorial sub-structure
  - Develop constraint-specific inference techniques that "prune" the search tree

Inference:
Domain Consistency (DC)

- Each value in the domain of each variable appears in at least one satisfying solution to the constraint
- Inference: remove values that do not meet the requirement



A constraint network is DC if all of its constraints are DC

Global Constraint

- An aggregate constraint over an arbitrary number of variables that:
  1. Represents some repeatedly occurring problem structure
  2. Allows for efficient inference that is stronger than can be achieved if a set of non-aggregated constraints is used to represent the structure

CP Model for a Nurse Scheduling Problem

$$
\begin{aligned}
\min \quad & \sigma \\
\text{s.t.} \quad & \texttt{spread}(\{W_1, \ldots, W_n\}, \mu, \sigma), \\
& \texttt{multiknapsack}(\{N_1, \ldots, N_m\}, \{A_1, ..., A_m\}, \{W_1, \ldots, W_n\}), \\
& \texttt{cardinality}(\{N_1, \ldots, N_m\}, \{1, \ldots, n\}, \{1, \ldots, MaxPatients\}), \\
& \texttt{pairwiseDisjoint}(\{Z_1, \ldots, Z_p\}), \\
& Z_k = \bigcup_{i \in P_k} N_i, & k = 1, \ldots, p \\
& W_j \in \{min\{A_i\}, \ldots, MaxAcuity\}, & j = 1, \ldots, n \\
& N_i \in \{1, \ldots, n\}. & i = 1, \ldots, m
\end{aligned}
$$

University of Toronto
Mechanical & Industrial Engineering

[Schaus et al. 2009] *CPAIOR*, 248-262, 2009.

DC for a Global Constraint

- Given: $c(v_1,\ldots, v_m)$
- c is *domain consistent* iff for all variables v_i, for all values $d_i \in D_i$ there exists a tuple of values

  $[d_j \in D_j], j \neq i$

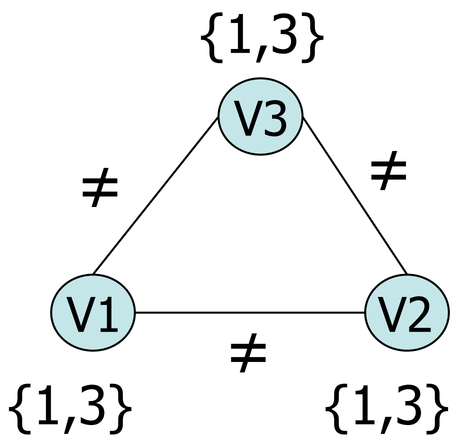
  such that

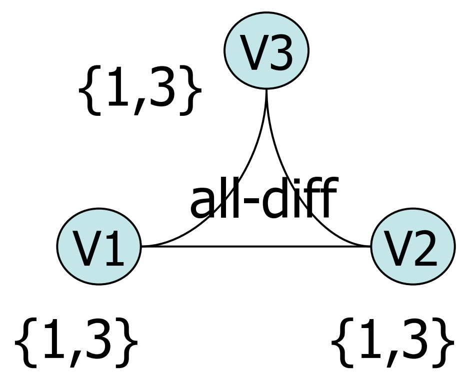
  $(v_i=d_i,[v_j=d_j]) \rightarrow T$

Need a "solution" to the constraint that supports $v_i=d_i$

All-Diff vs. Clique of ≠

- all-diff(v_1, v_2, …, v_n) $=_{def}$
  $v_i \neq v_j$ for $1 \leq i < j \leq n$

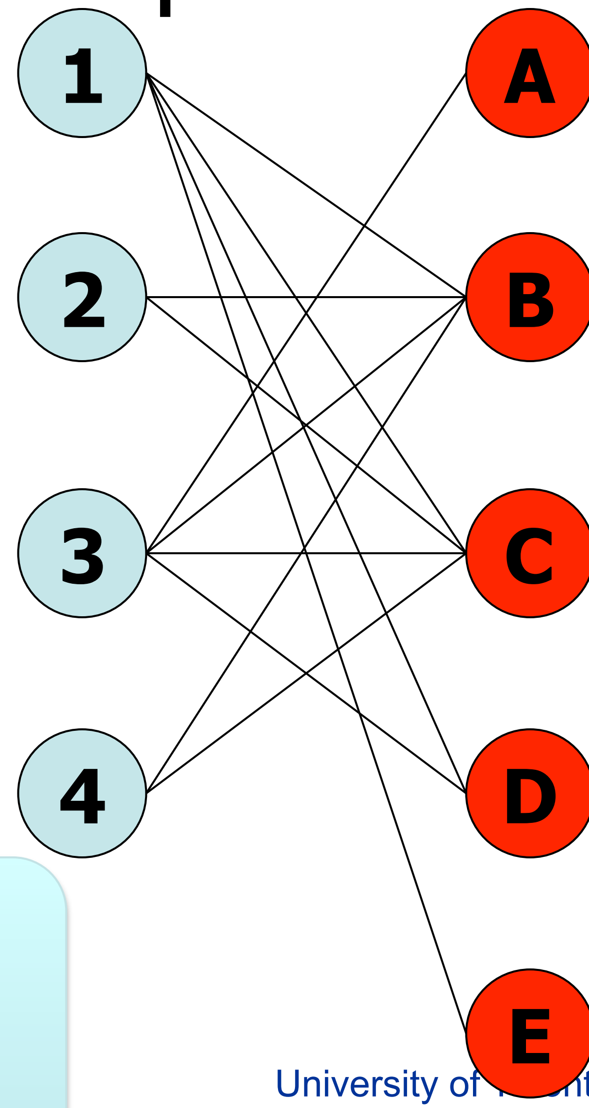




What is the complexity of making an all-diff DC?

All-different: Value Graph

- Example
  - $D_1 = \{B,C,D,E\}$,
    $D_2 = \{B,C\}$,
    $D_3 = \{A,B,C,D\}$,
    $D_4 = \{B,C\}$
  - all-diff($v_1,\ldots,v_4$)



A variable assignment is part of a solution to an all-diff constraint iff its corresponding edge is in a maximal matching [Regin 1994]

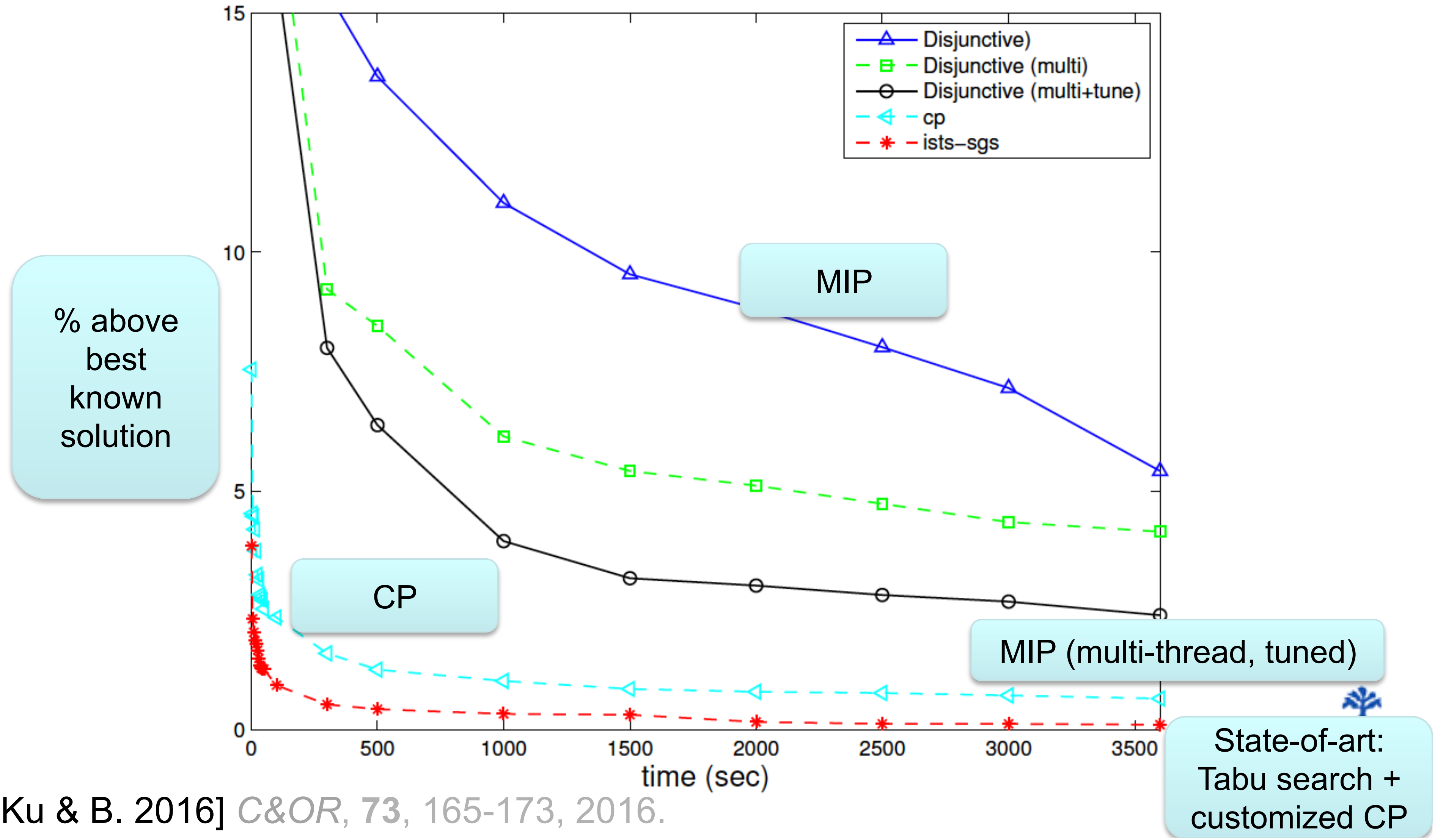
So …

- Over the past 25 years, CPers have developed a large number (400+) global constraints, accompanying inference algorithms, and complexity results
  - In practice, a smaller number of global constraints (~25) is commonly used
- Modeling is the “plugging together” of these global constraints

What is CP Good At?

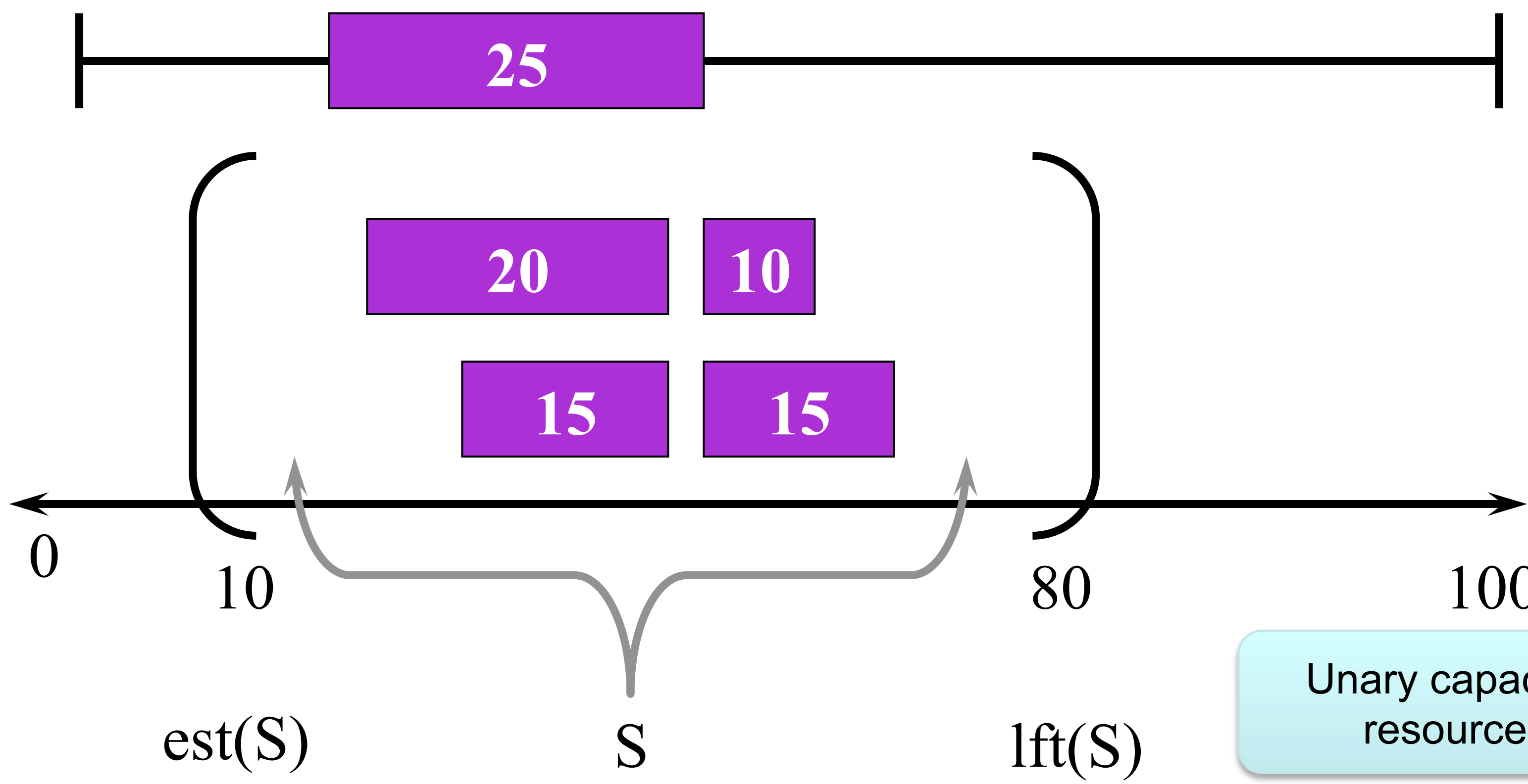
- CP wins or loses on inference!
- Problems with
  - interacting combinatorial structures that make it difficult to find a feasible solution
  - strong back-propagation from the cost function
- Scheduling is one of the most successful applications of CP

Job Shop Scheduling (10 instances: 20X20)



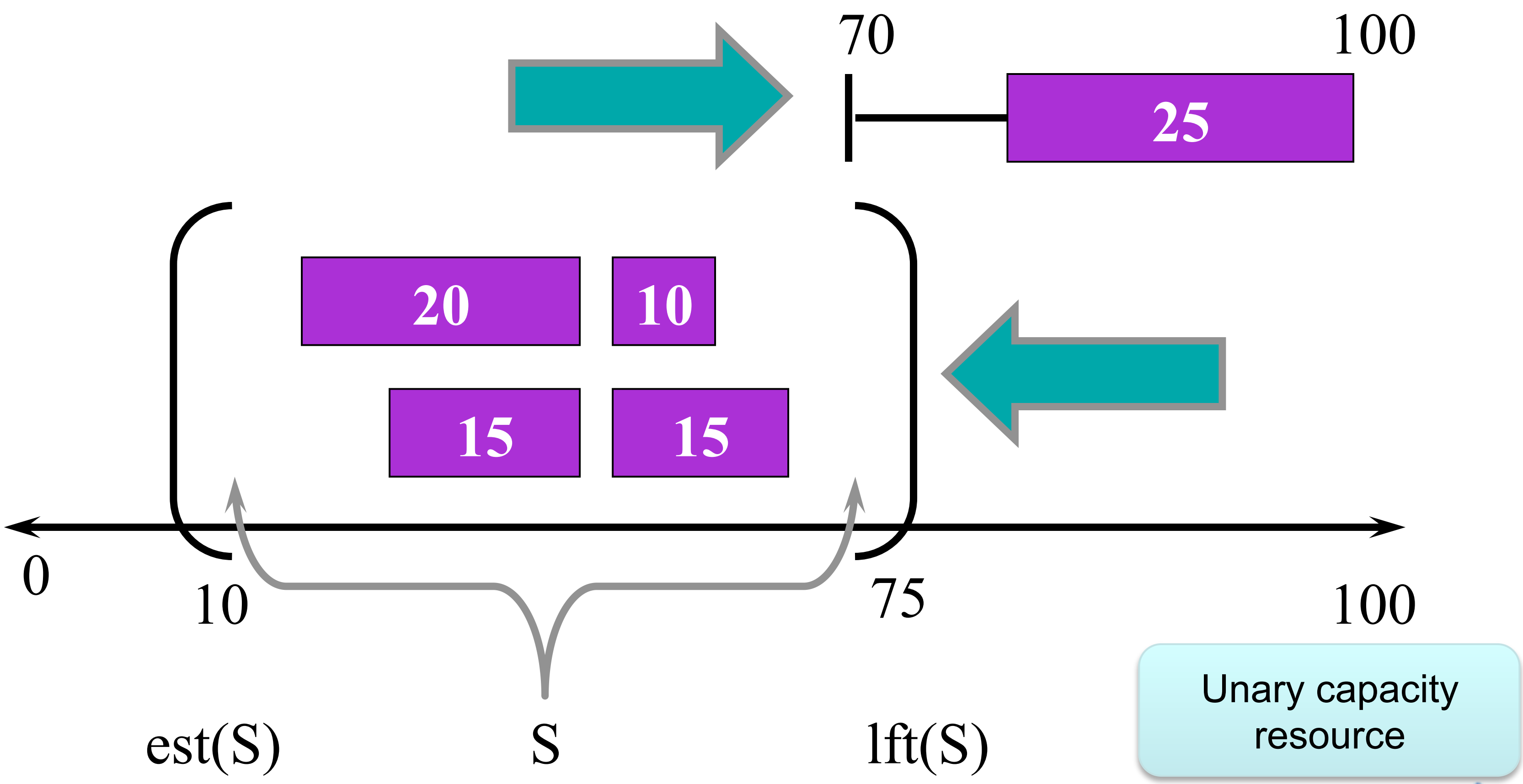
[Ku & B. 2016] *C&OR*, **73**, 165-173, 2016.

Scheduling Inference (Edge-Finding)



Unary capacity resource

Scheduling Inference (Edge-Finding)



University of Toronto
Mechanical & Industrial Engineering

CP Summary

- Very rich constraint language
  - Modeling is plugging together useful sub-structure (i.e., global constraints)
- Branch-and-infer
  - Tree search
  - At each node, run the inference algorithms in each constraint to reduce the search space
  - Inference in one constraint "propagates" to others

Outline

- Learn Constraint Programming in 15 minutes or less!
- Why Hybridize?
- Three Decomposition Examples
- Final Comments

Why Hybridize?

This question is different (and orthogonal) to the question of "Why decompose?"

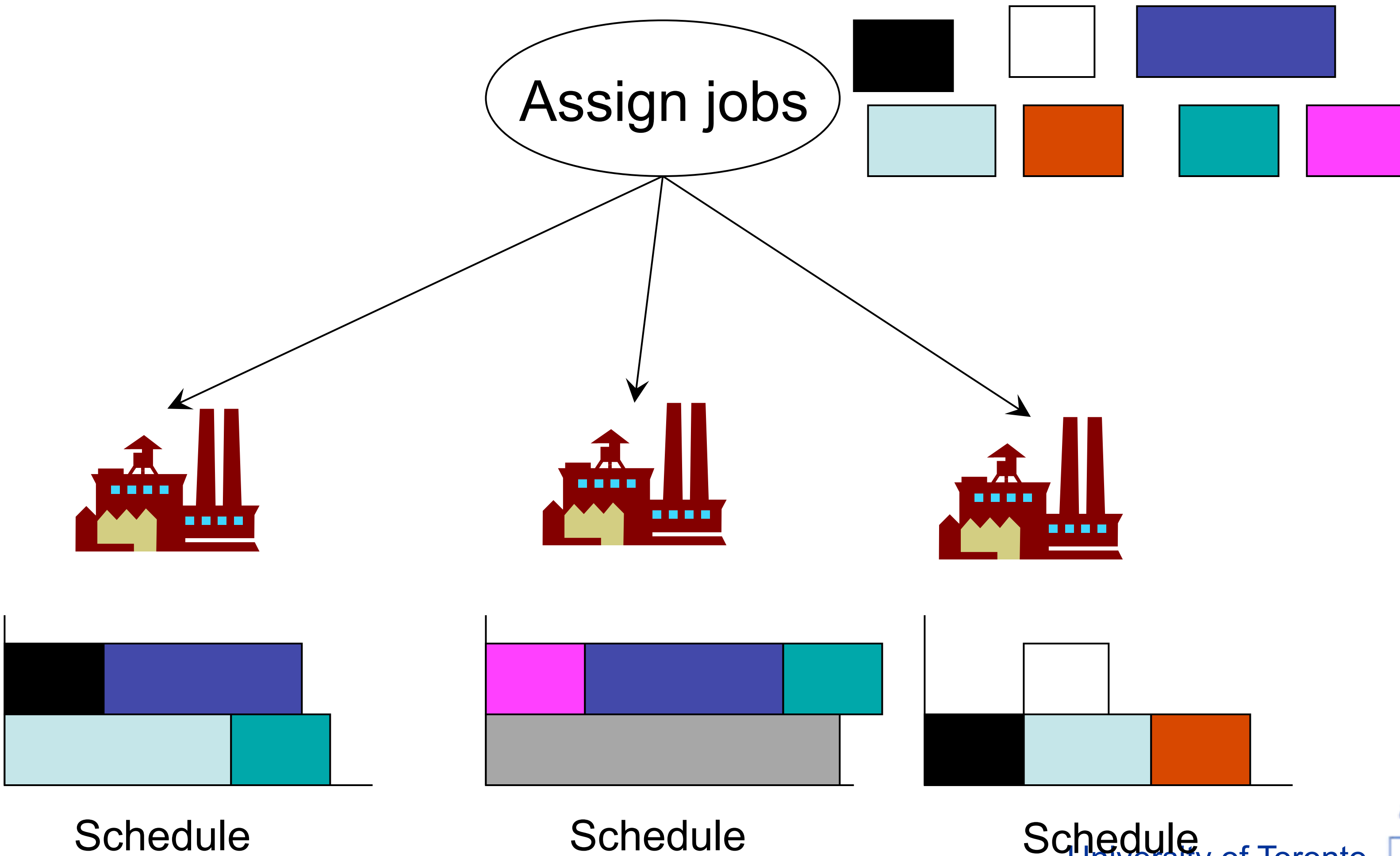
Decomposition-based CP-Hybrids

- Problems where CP brings something to the table, but doesn't have the whole answer
  - where there is a combination of mostly global cost-based reasoning and mostly local feasibility problems
  - where inference works well except for one problem characteristic
    - e.g., scheduling with alternatives

Problems with …

- "Cascading" decisions
  - some sort of assignment that activates or constrains other variables
    - assign jobs to resources/due dates then schedule
    - assign customers to open facilities then pack
    - decide # workers and then find policy
- Nice linear sub-problem relaxations and cuts
- A sub-problem where inference can perform strongly

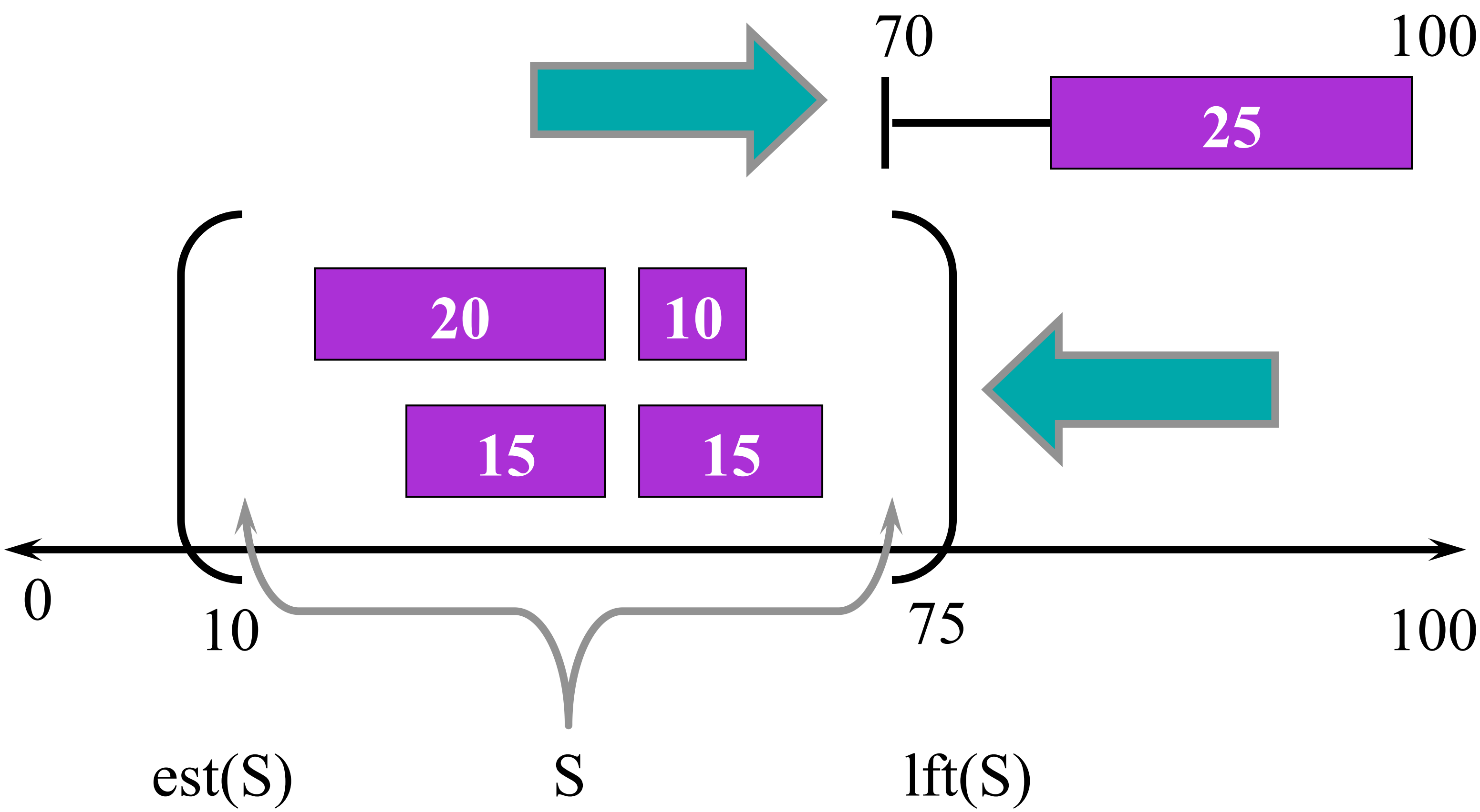
Resource Allocation & Scheduling



University of Toronto
Mechanical & Industrial Engineering

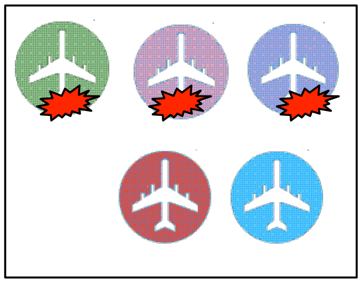
[Hooker 2005] *Constraints*, **10**, 385-401, 2005.

Edge-Finding

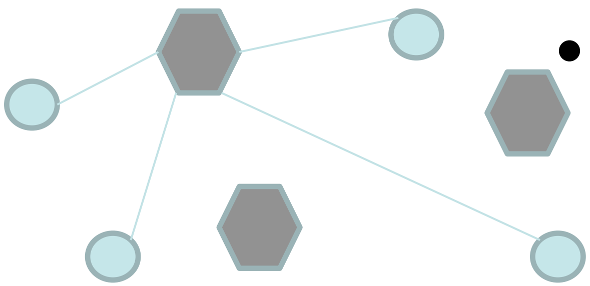


Problem: a reasonable number of resource assignments must be made before the strong inference techniques have any impact

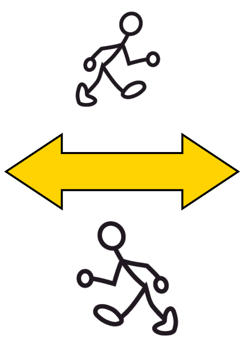
Three Examples



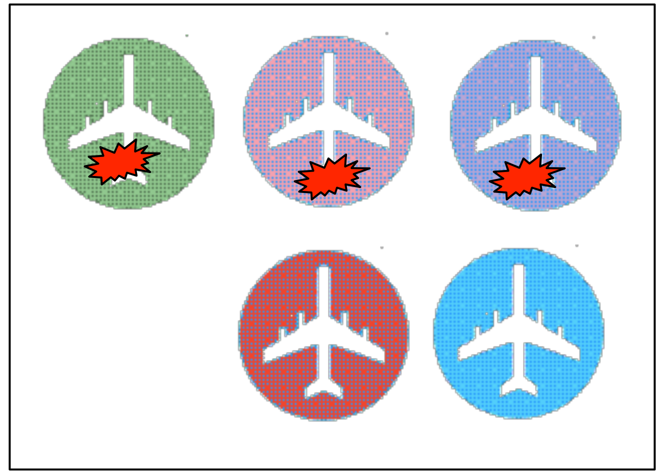
- Due date assignment and scheduling



- Facility location-allocation



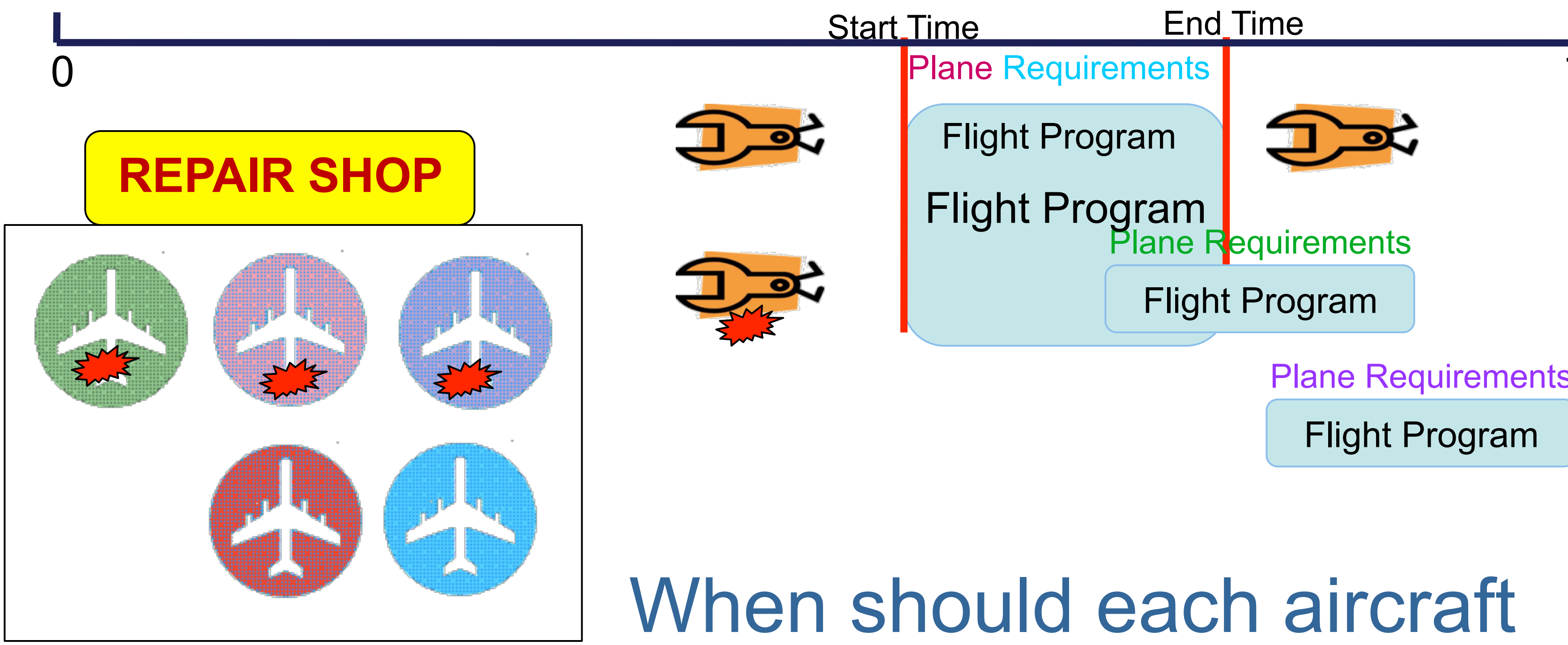
- Dynamic front-room/back-room service scheduling



An Aircraft Maintenance Scheduling Problem

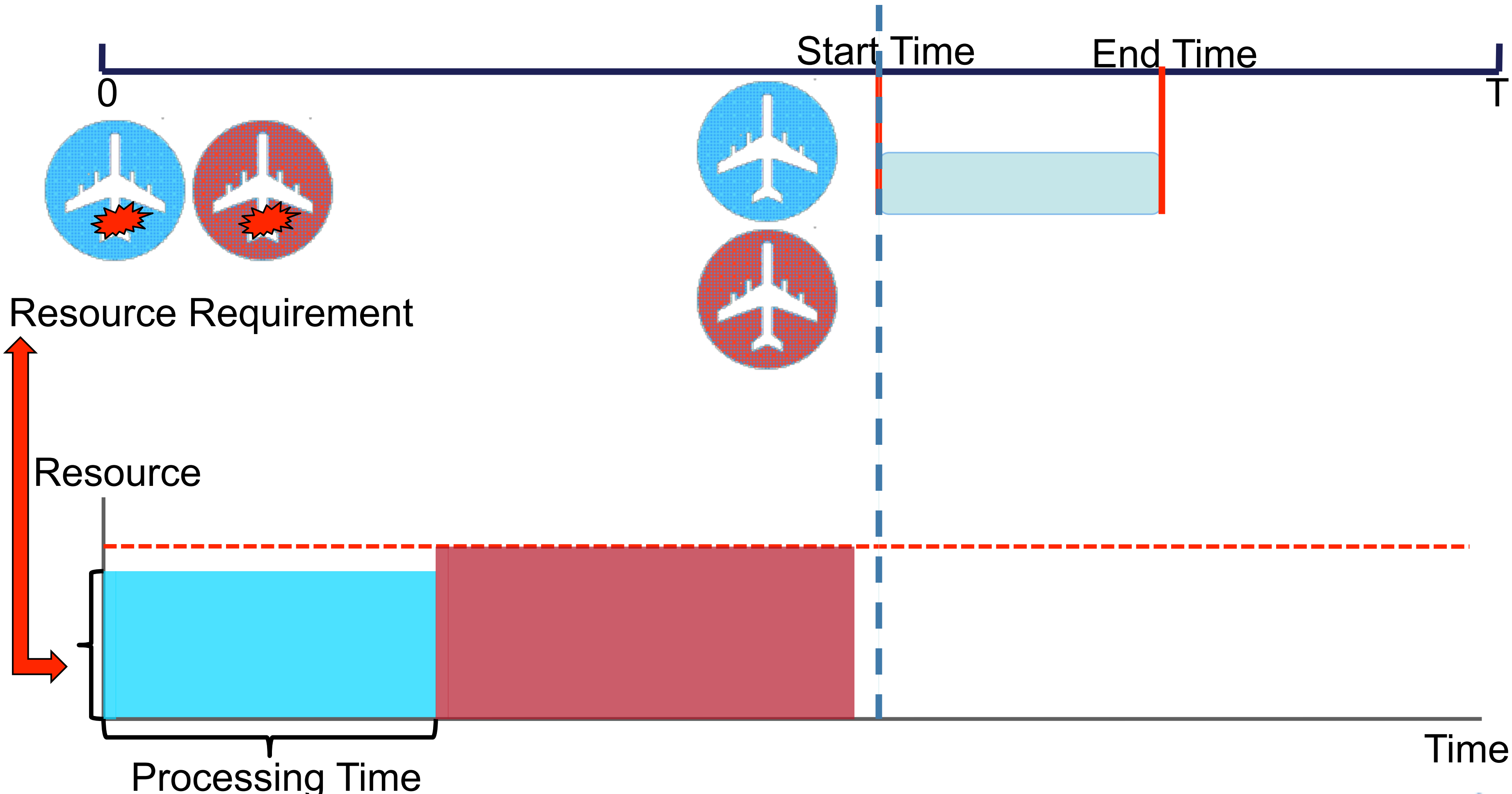
[Aramon Bajestani & B. 2013] *JAIR*, **47**, 35-70, 2013.

Aircraft Maintenance Scheduling Problem



When should each aircraft be ready?

Scheduling in the Repair Shop



University of Toronto
Mechanical & Industrial Engineering

Solution Approach: LBBD

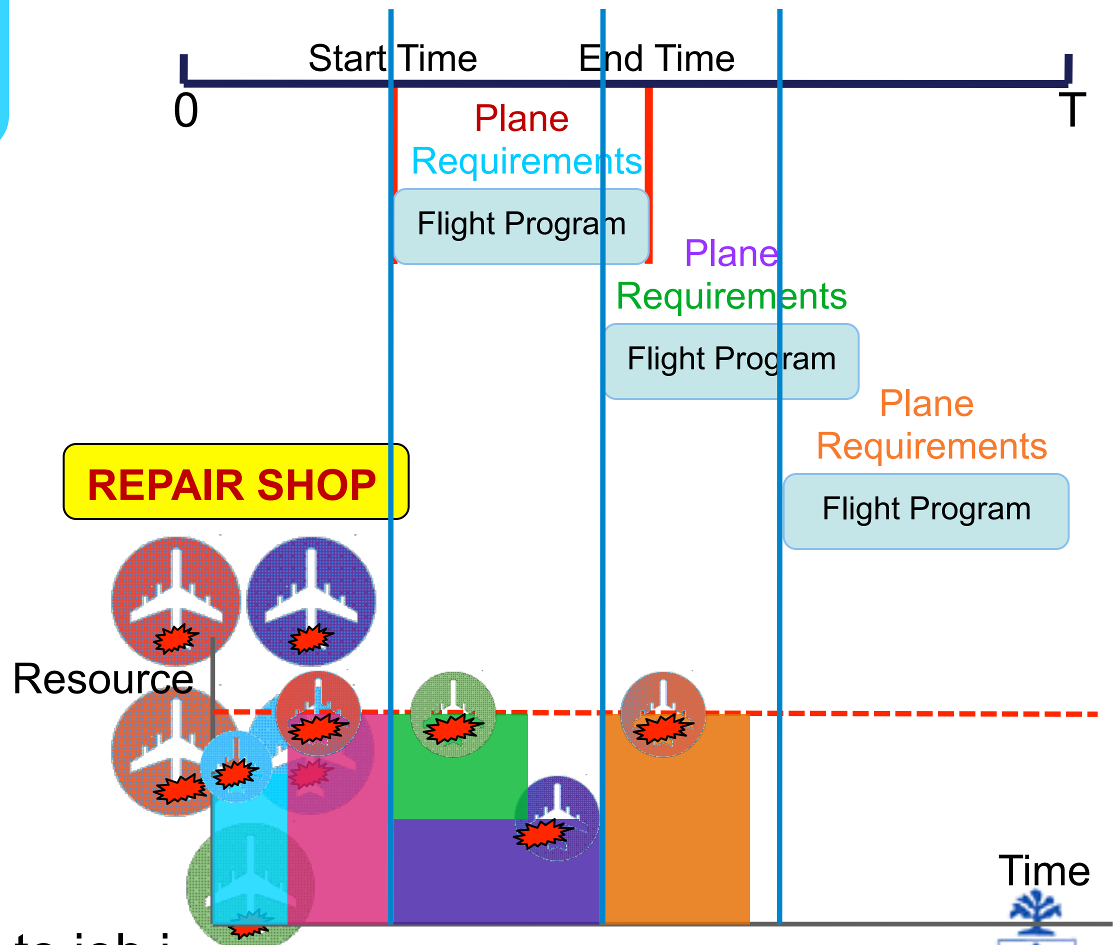
Master Problem
Assign aircraft to due dates to maximize flight coverage

No: Cut / **Solution**

Sub-problems
Is there a feasible repair shop schedule?

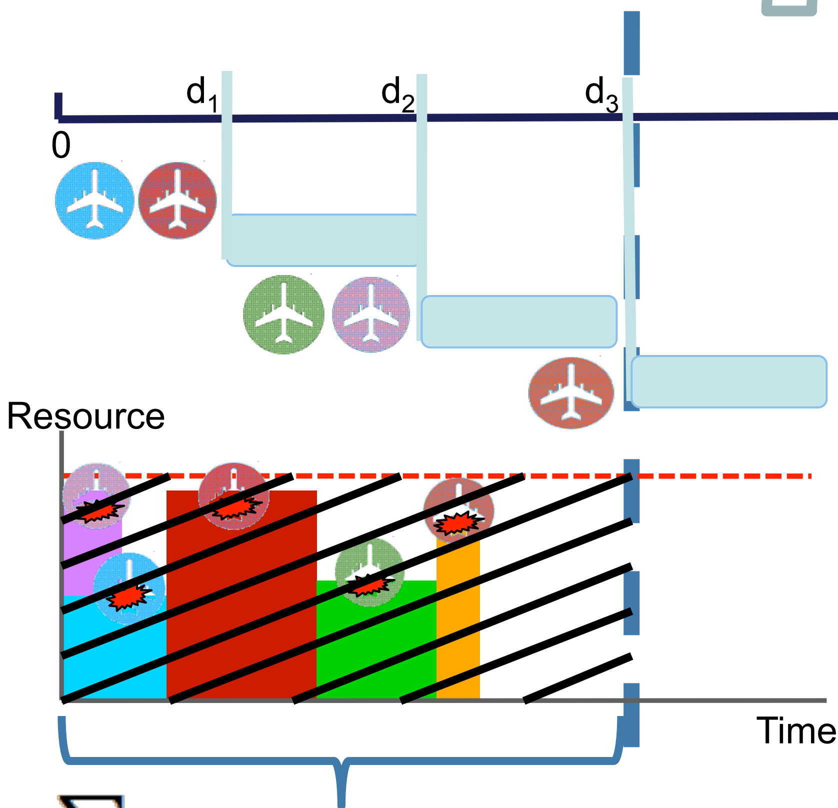
Yes

Optimal Solution

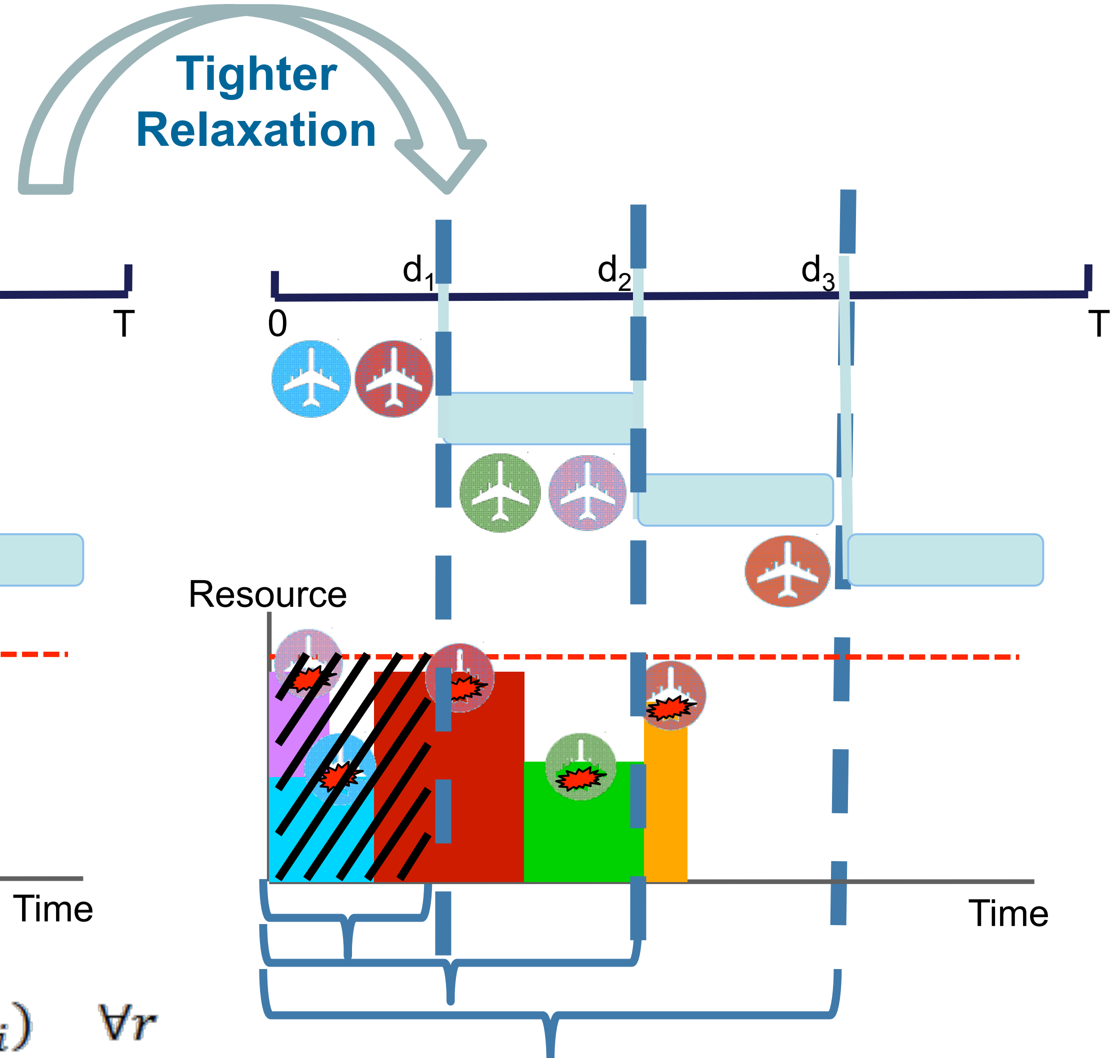


$x_{ij}=\begin{cases} 1 \text{ if due date i is assigned to job j} \\ 0 \text{ otherwise} \end{cases}$

Relaxation

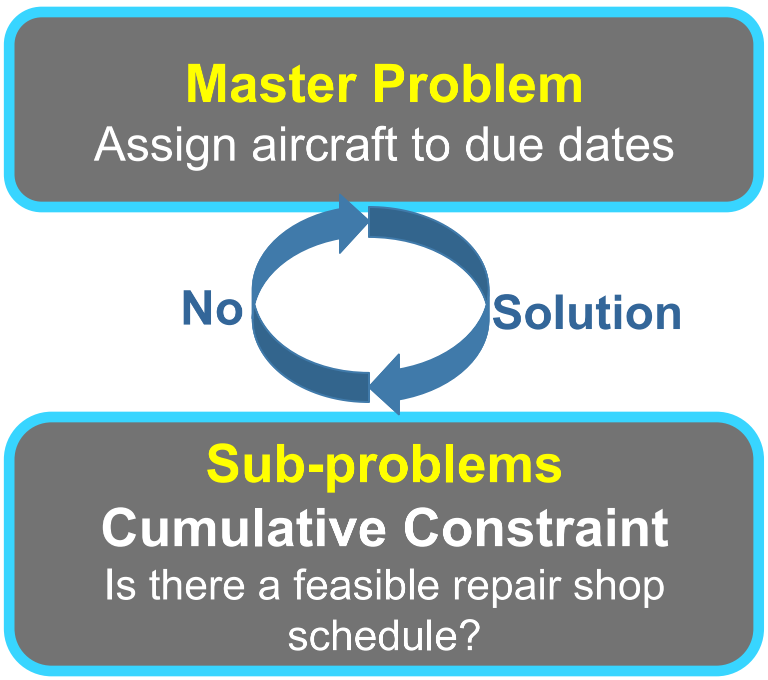


$$\sum_{j \in M_r} c_{jr}\, p_{jr} \leq C_r \max_{j \in M_r, i \in D} (x_{ij} d_i) \quad \forall r$$

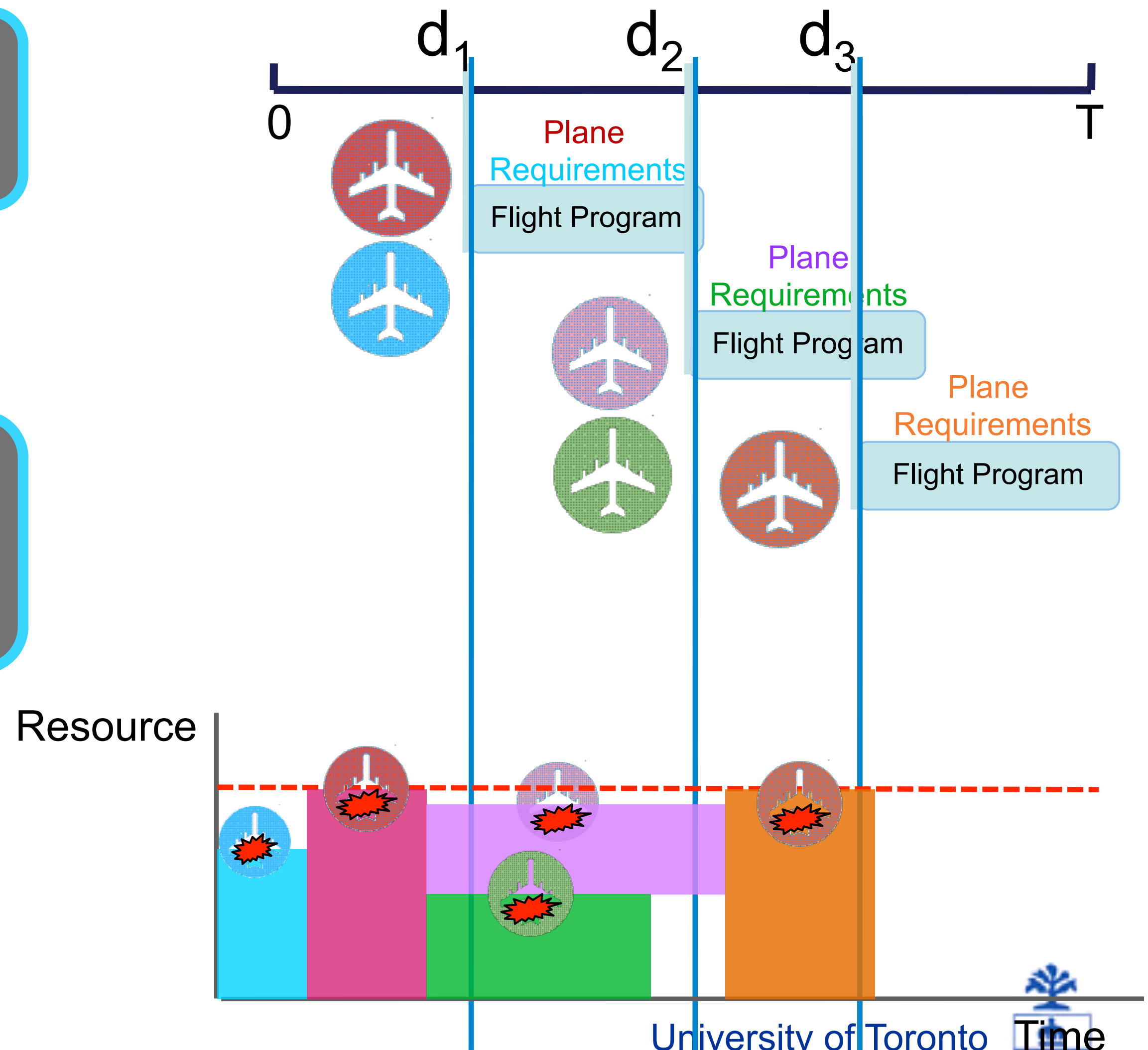


$$\sum_{j \in M_r, \sum_{i=1}^{|D|} x_{ij} d_i \leq ST_w} c_{jr}\, p_{jr} \leq C_r\, ST_w \quad \forall r, w$$

Cut

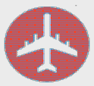
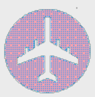
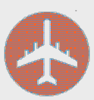


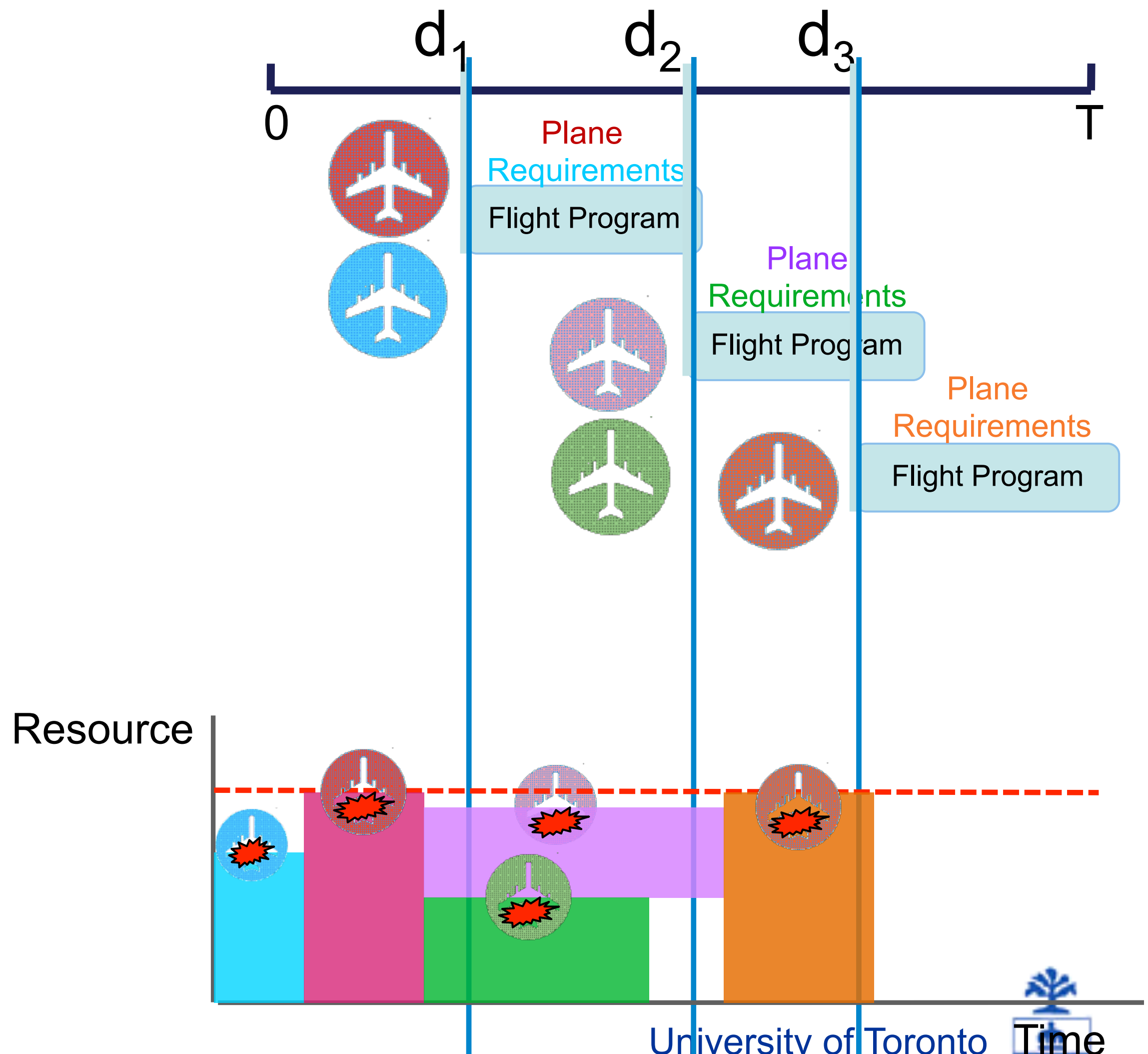
Master Problem
Assign aircraft to due dates
No
Solution
Sub-problems
Cumulative Constraint
Is there a feasible repair shop schedule?



d1
d2
d3
0
T
Plane
Requirements
Flight Program
Plane
Requirements
Flight Program
Plane
Requirements
Flight Program
Resource
Time

Cut

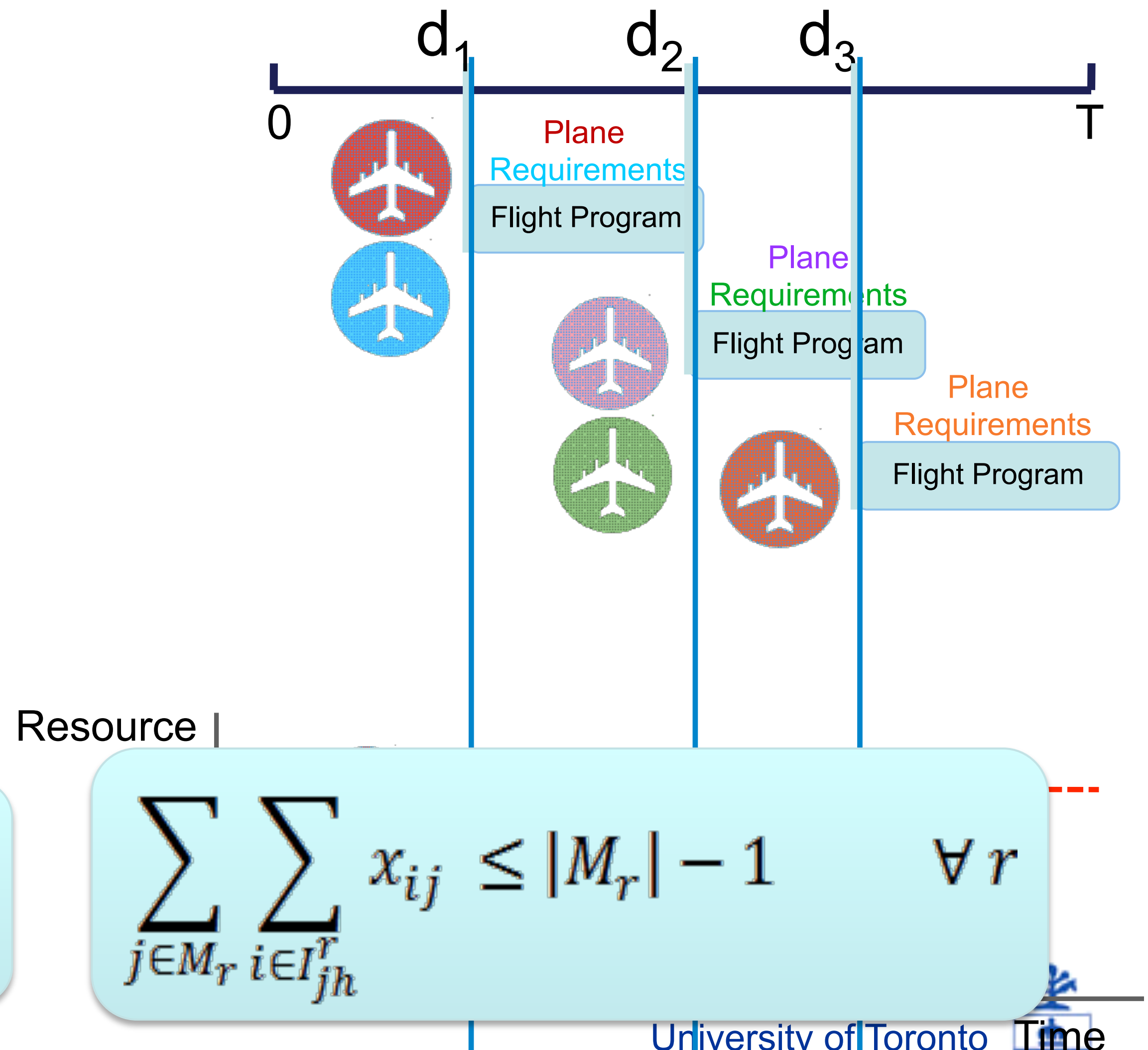
x_{ij}	d_1	d_2	d_3
	1	0	0
	1	0	0
	0	1	0
	0	1	0
	0	0	1



Cut

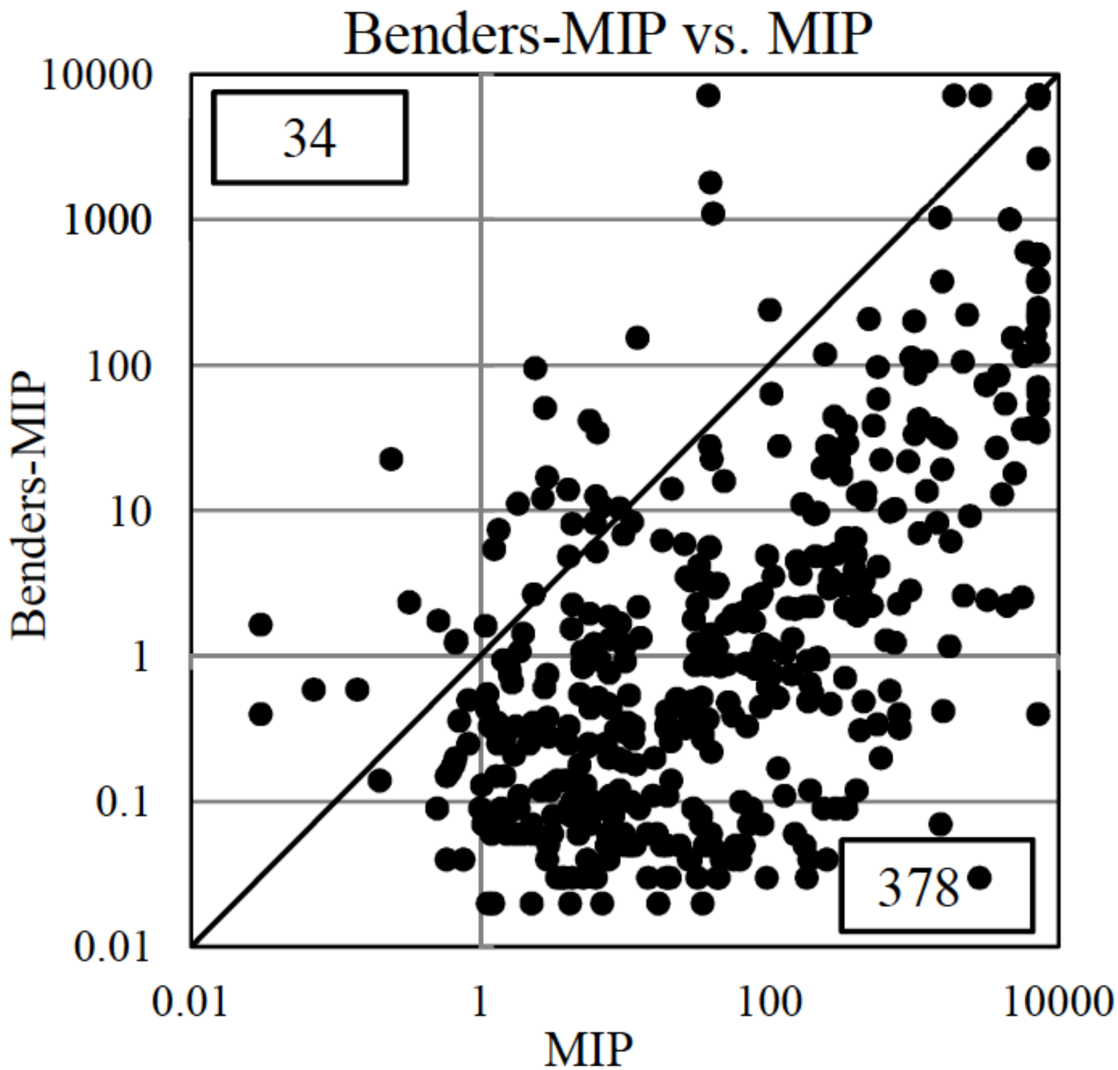
x_{ij}	d_1	d_2	d_3
	1	0	0
	1	0	0
	0	1	0
	0	1	0
	0	0	1

$$\sum x_{ij} \leq (5\text{-}1)$$



$$\sum_{j \in M_r} \sum_{i \in I_{jh}^r} x_{ij} \leq |M_r| - 1 \qquad \forall r$$

Computational Results



Computational Results

Method	Mean Time (s)	Mean Iterations (Median)	Mean % MP Time (Median)	Mean % SP Time (Median)	% Solved to Opt.
Benders-MIP-T	213	66.4 (8.0)	52% (54%)	48% (46%)	98%
Benders-MIP	227	64.7 (8.0)	62% (67%)	38% (33%)	98%
MIP	837	-	-	-	94%
Dispatch Rule	≈ 0	-	-	-	10%*
CP	6857	-	-	-	5%

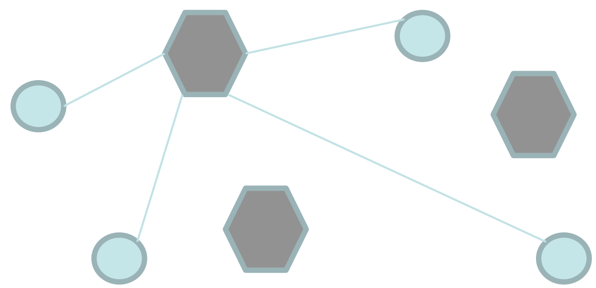
- 420 problem instances
- 7200-second time limit
- IBM ILOG CPLEX & CPO 12.3

Decomposition-based CP-Hybrids

- Problems where CP brings something to the table, but doesn't have the whole answer
  - where there is a combination of mostly global cost-based reasoning and mostly local feasibility problems
  - where inference works well except for one problem characteristic
    - e.g., scheduling with alternatives

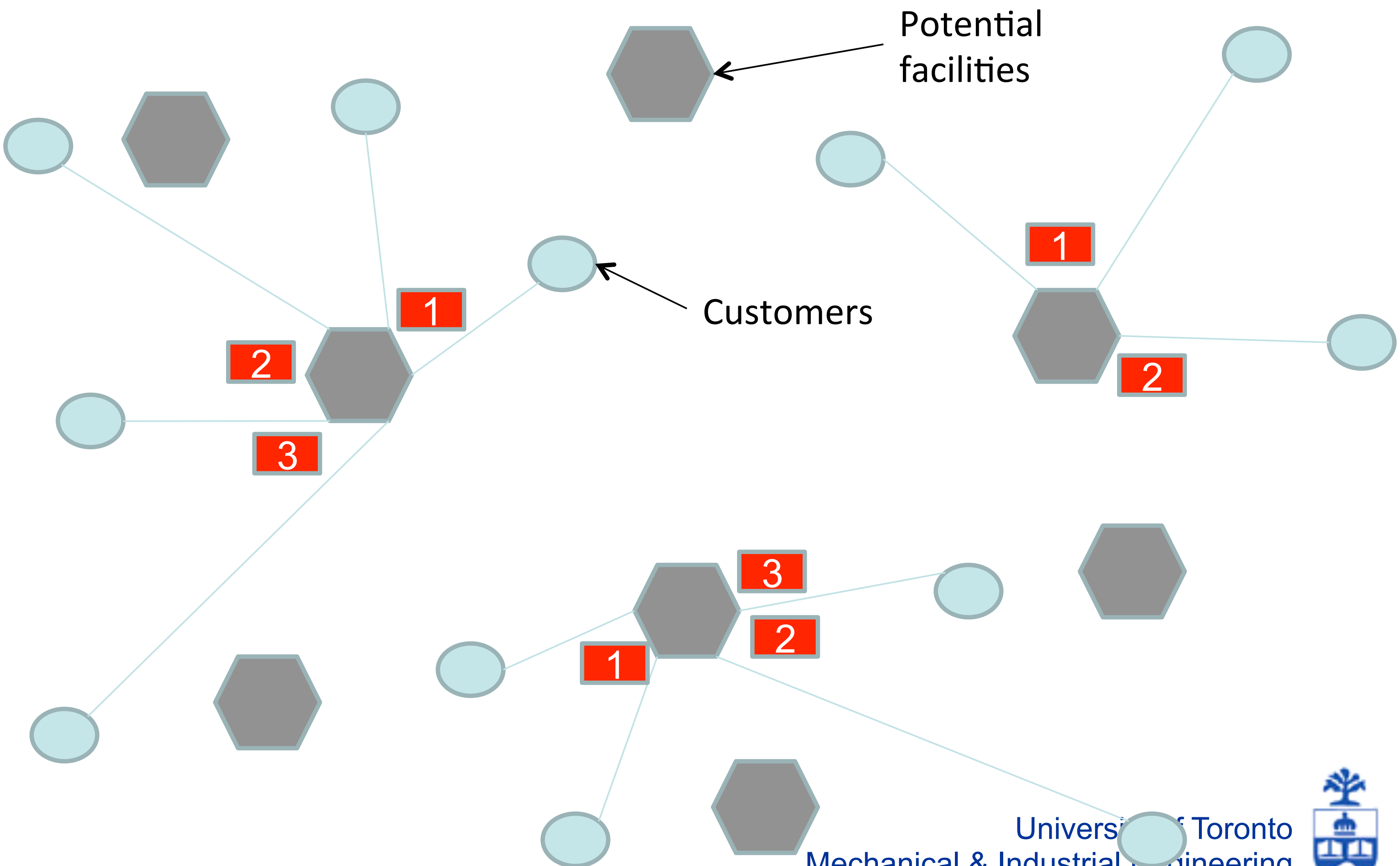
Due date assignment

A Location- Allocation Problem



[Fazel-Zarandi & B. 2013] *IJOC*, **24**, 399-415, 2012.

A Location-Allocation Problem



Problem

- Choose facilities to open ($p_j = 1$), given fixed facility cost (f_j)
- Assign customers to facilities (x_{ij}) given service cost (c_{ij})
- Assign customers to trucks ($truck_i$) given cost per truck (u) and maximum travel distance for each truck (ℓ)

[Alberada-Sambola, et al. 2009] *C&OR*, **36**(2), 597-611, 2009.

LBBD Model

Master Problem

Open facilities, assign customers, and assign # of trucks to each facility

Cut

Solution

Sub-problems

At each facility, pack trips onto allocated trucks

Master Problem

$$\text{minimize} \quad \sum_{j\in J} f_j p_j + \sum_{i\in I}\sum_{j\in J} c_{ij}x_{ij} + u\sum_{j\in J} V_j$$

opening + service + truck costs

$$\text{s.t.} \quad \sum_{j\in J} x_{ij} = 1 \quad i \in I,$$

customers assign to one facility

$$\sum_{i\in I} d_i x_{ij} \le b_j p_j \quad j \in J,$$

facility capacity

$$t_{ij}x_{ij} \le l \quad i \in I, j \in J,$$

truck distance

$$V_j \ge \frac{\sum_{i\in I} t_{ij}x_{ij}}{l} \quad j \in J,$$

sub-problem relaxation

$$cuts,$$

$$x_{ij} \le p_j \quad i \in I, j \in J,$$

only assign customer to open facilities

$$x_{ij}, p_j \in \{0,1\}, V_j \in \{0,\dots,\bar{k}\} \quad i \in I, j \in J,$$

Sub-problems

The CP formulation of the TASP is as follows:

$$\begin{aligned} \min \quad & V_j^{\mathrm{BP}} \\ \text{s.t.} \quad & \mathrm{pack}(load, truck, dist), \\ & V_j \leq V_j^{\mathrm{BP}} \leq V_j^{\mathrm{FFD}}, \end{aligned}$$

Series of feasibility problems

Cut

Variable in master

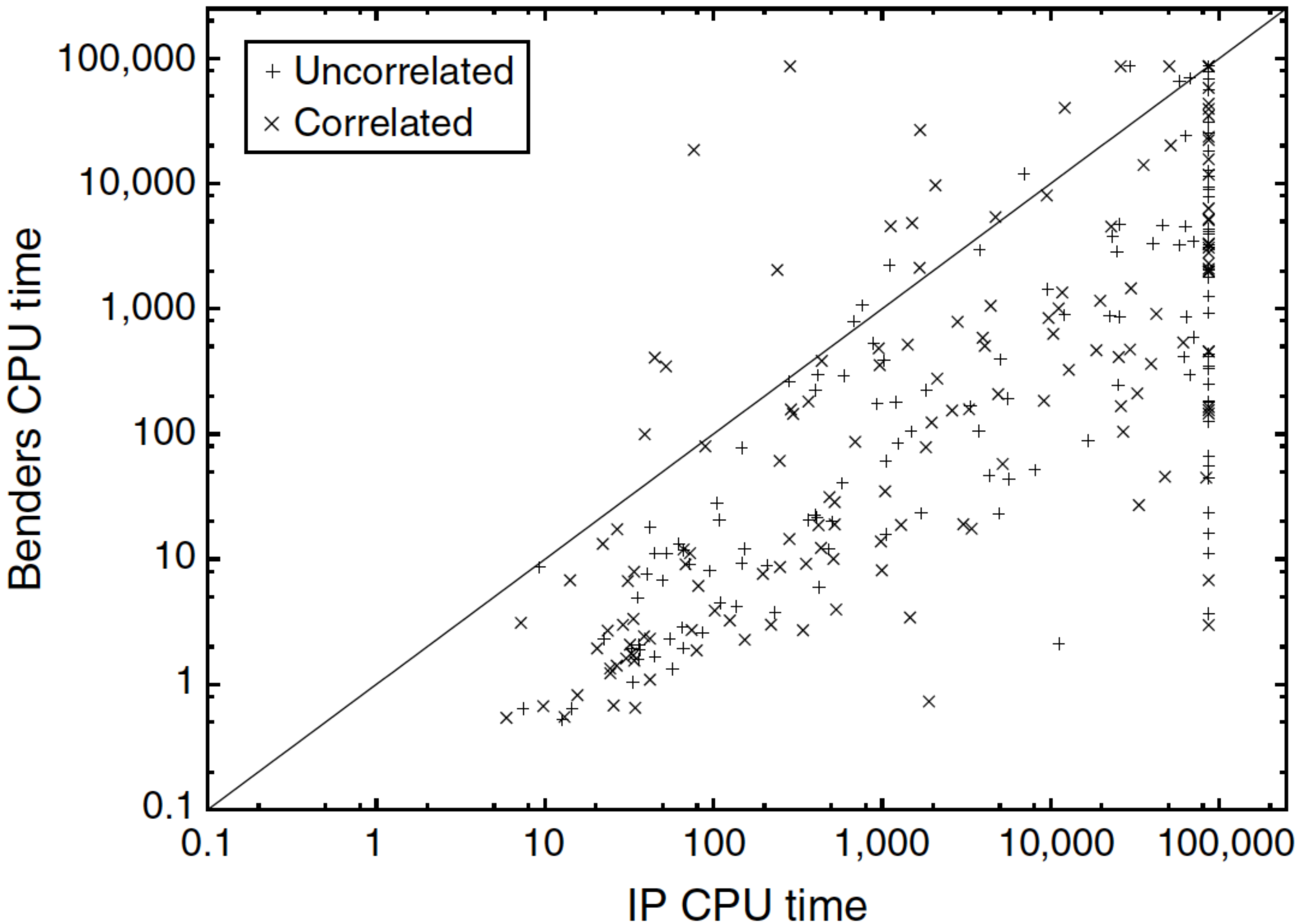
UB on truck reduction if one visit is removed

$$V_j \geq V_{jh}^* - \sum_{i \in I_{jh}} (1 - x_{ij}), \quad j \in J_h,$$

Optimal value from TASP

Results

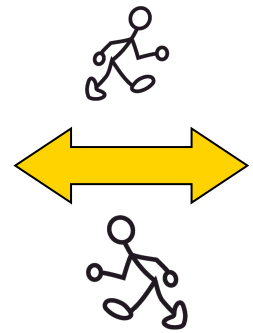
IBM CPLEX 11.0
IBM ILOG Solver 6.5
Time-out: 24 hours



Decomposition-based CP-Hybrids

- Problems where CP brings something to the table, but doesn't have the whole answer
  - where there is a combination of mostly global cost-based reasoning and mostly local feasibility problems
  - where inference w[...] problem characte[...]
    - e.g., scheduling with alternatives

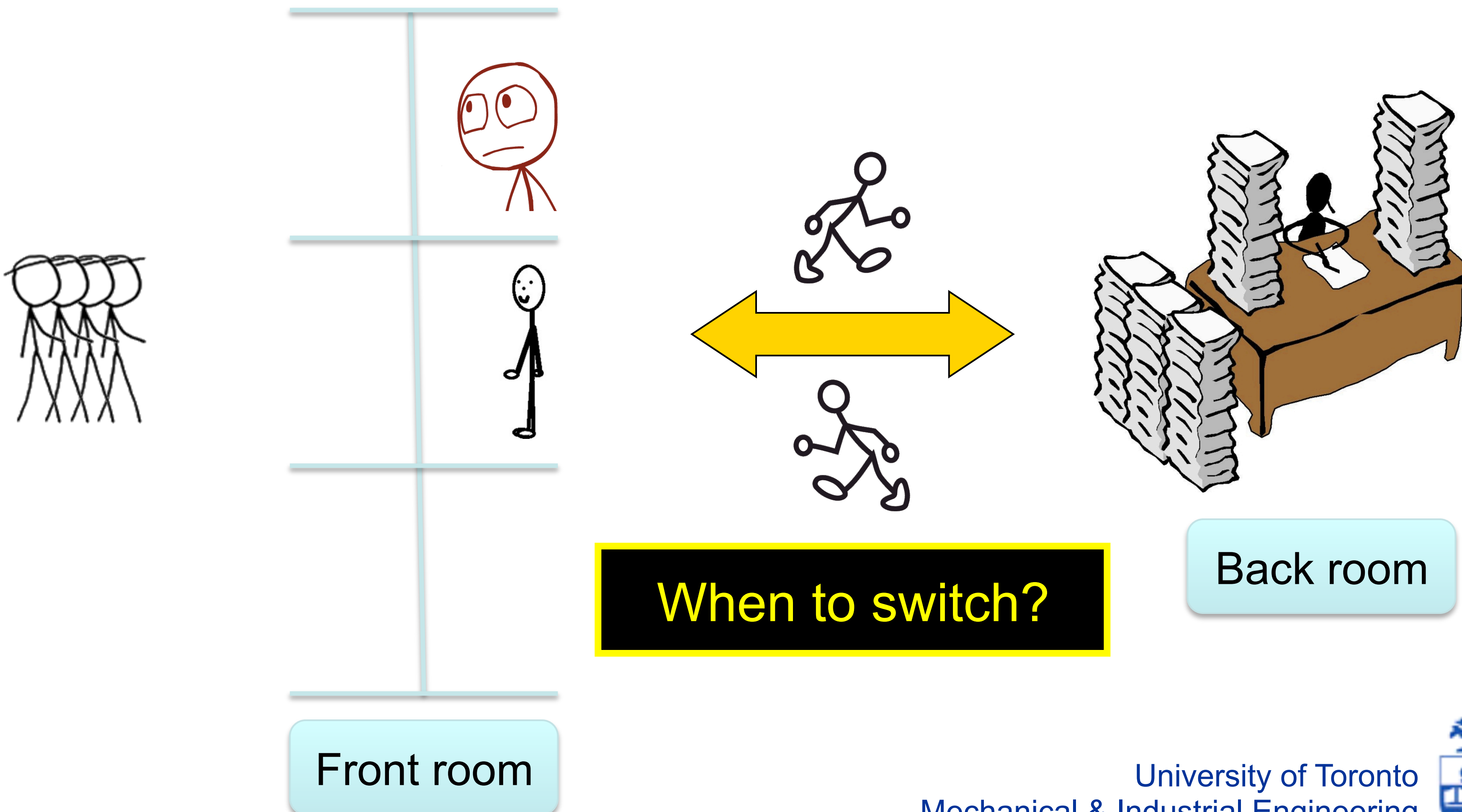
Global assignments, local packing



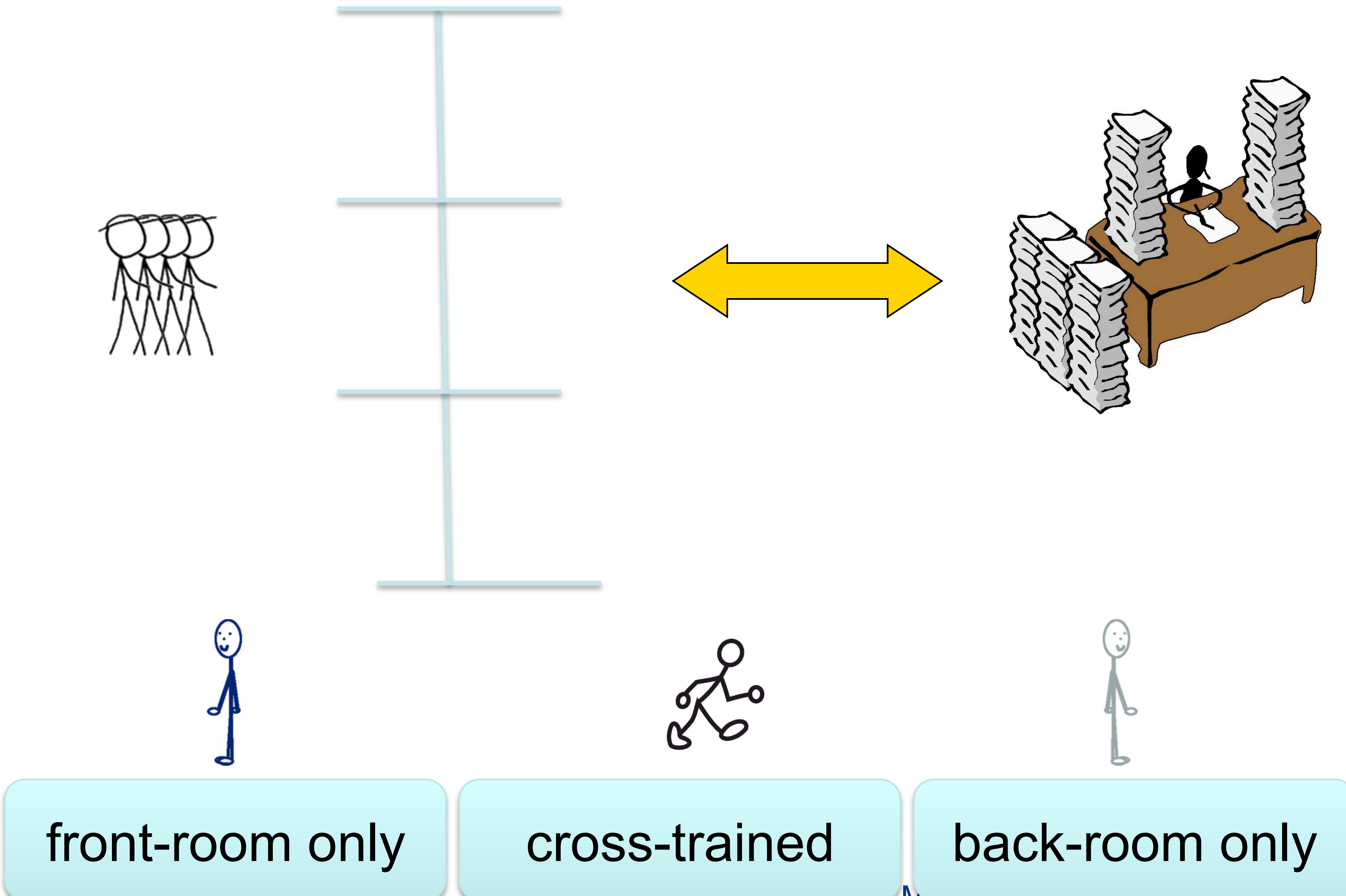
Dynamic Front-Room/Back-Room Service Scheduling

[Terekhov, B., & Brown 2009] *IJOC*, **21**(4), 549-561, 2009.

A Front-Room/Back-Room Problem



Problem Description



Problem Description

- Determine the number of front-room, back-room, and cross-trained workers to hire and a policy for switching workers that:
  - Minimizes total cost
  - Meets a bound on the maximum expected customer waiting time
  - Ensures all the work in the back-room is done

Problem Description

Cost of front-room workers

# of front-room workers

$$\text{minimize } c_f f + c_b b + c_x x$$

$$\text{s.t. } W_q \leq W_u,$$

$$B \geq B_l.$$

[Terekhov, B., & Brown 2009] *IJOC*, **21(4)**, 549-561, 2009.

Cost Cases

(1) $c_b > c_x > c_f$,

(2) $c_f > c_x > c_b$,

(3) $c_x \geqslant c_f + c_b$,

(4) $c_x \leqslant c_f$ and $c_x \leqslant c_b$,

(5) $c_x \leqslant c_b + c_f$, $c_x \geqslant c_f$ and $c_x \geqslant c_b$.

Cross-trained workers cost more than single skill workers but less than two of them

[Terekhov & B. 2009] *EJOR*, **198**, 223-231, 2009.

LBBD Model

Master Problem

Assign # of workers: *f, x, b*

Cut ⟳ **Solution**

Sub-problem

Find a switching policy that meets bounds on front-room waiting and gets back-room work done

Master Problem

$$\text{minimize} \quad \text{cost} = c_f f + c_b b + c_x x$$

$$\text{s.t.} \quad f + x \geq F_{\text{total}},$$

$$b + x \geq B_{\text{total}},$$

$$0 \leq f \leq F_{\text{total}} - 1,$$

$$0 \leq b \leq B_{\text{total}} - 1,$$

$$1 \leq x \leq F_{\text{total}} + B_{\text{total}} - 1,$$

$$f + b + x \geq \max(F_{\text{total}}, B_{\text{total}}),$$

$$\max(c_f F_{\text{total}}, c_b B_{\text{total}}) \leq \text{cost} \leq c_f F_{\text{total}} + c_b B_{\text{total}},$$

cuts

front-room workers if $x = 0$

back-room workers if $x = 0$

All these constraints are really a relaxation of the sub-problem

Cut

$$(x > x' \vee f > f' \vee b > b')$$

Value of *x* in last master solution

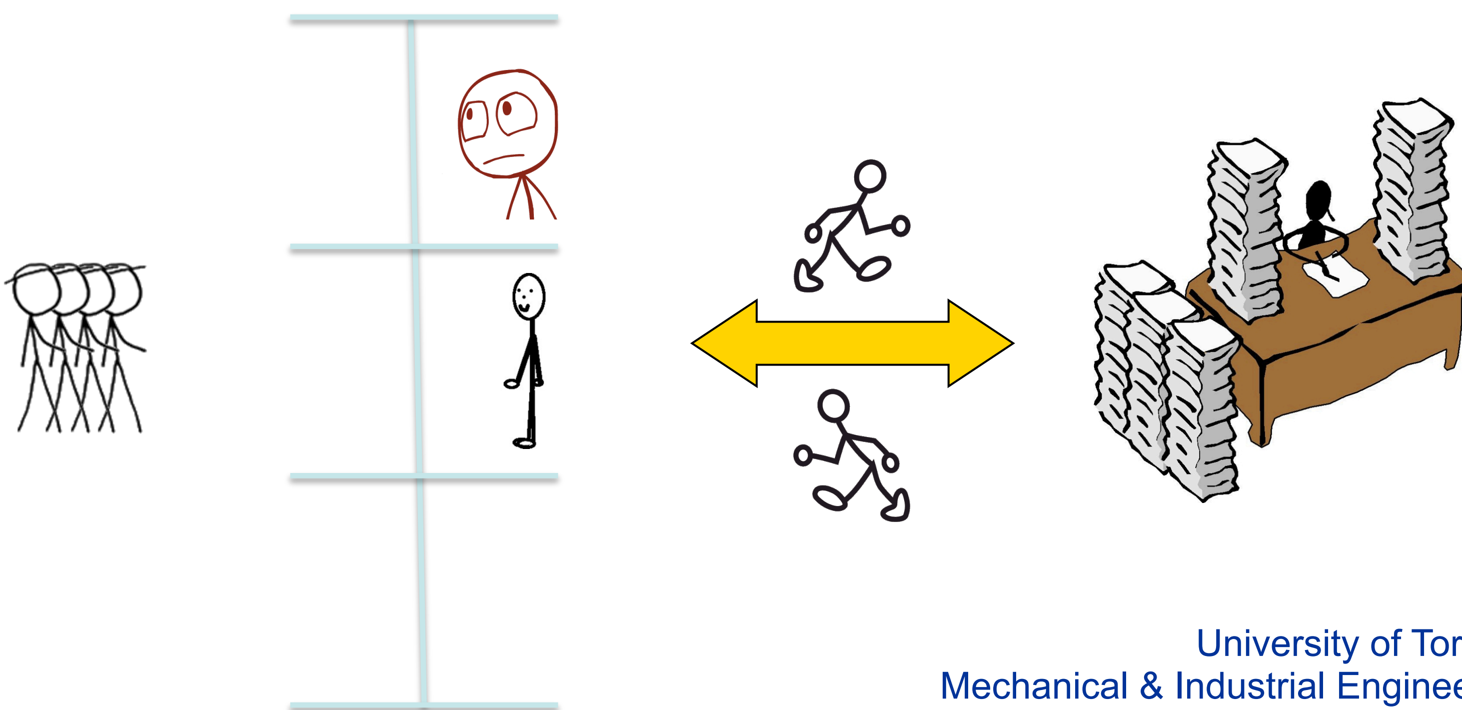
Value of *f* in last master solution

Value of *b* in last master solution

- If sub-problem is infeasible, we need at least one more worker
- Nogood cut

Sub-problem

- Find a policy for switching workers that:
  - Satisfies expected customer waiting time
  - Ensures all the work in the back-room is done



Problem Formulation

- Max # of customers – S
- # of workers – N
- Customers arrive according to Poisson process with rate λ
- Service times follow exponential distribution with rate μ

University of Toronto
Mechanical & Industrial Engineering

[Berman et al. 2005] *EJOR*, **167**(2), 349-369, 2005.

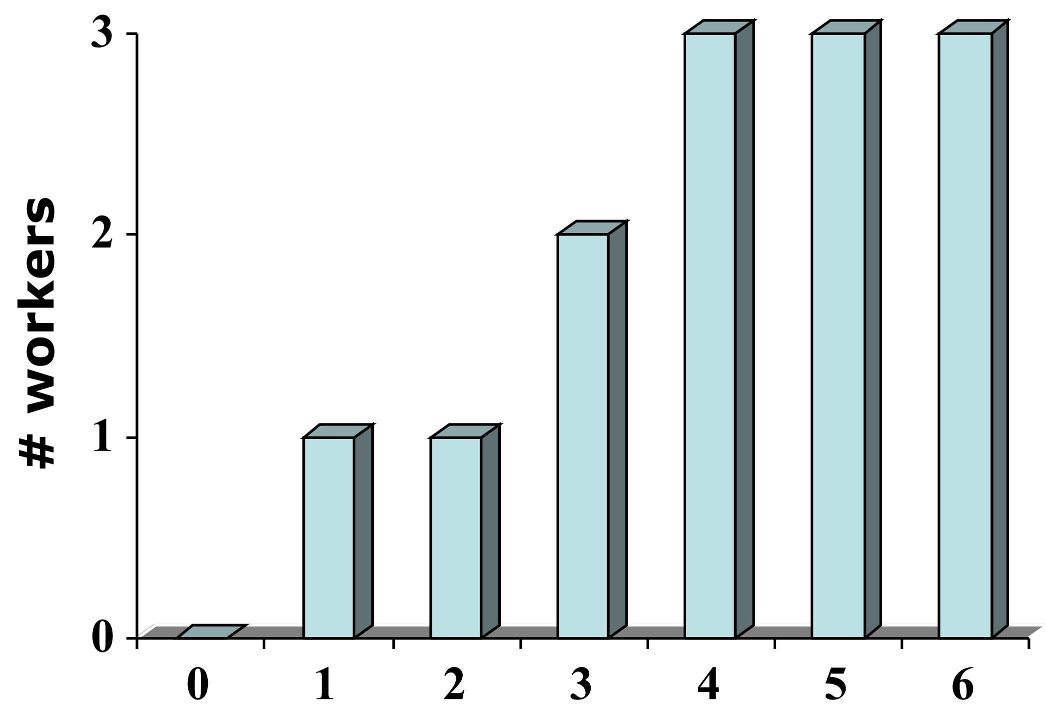
Switching Policy

$(k_0, k_1, k_2, \ldots, k_N)$

$k_0 \quad k_1 \quad k_2 \quad k_3$

$(0, 2, 3, 6)$

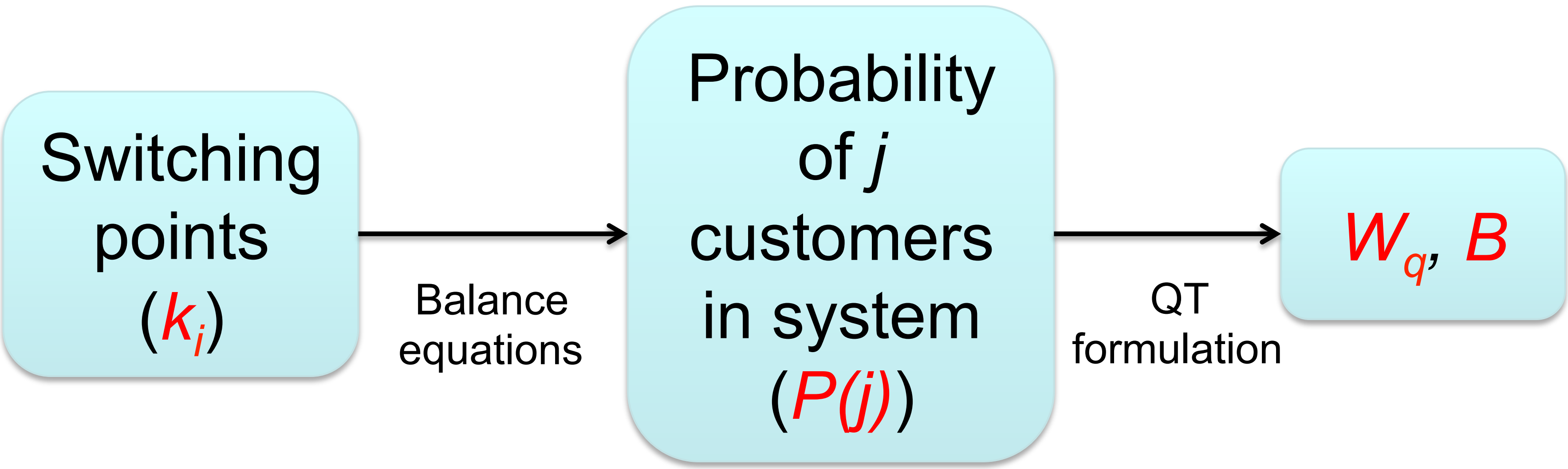
Should have $i+f$ workers in the front-room when there are between $k_{i-1}+1$ and k_i customers



Obs: 1) $k_{i-1} < k_i$ 2) lower k_i less waiting

[Berman et al. 2005] *EJOR*, **167**(2), 349-369. 2005.

What Are We Trying To Do?



- Construct a CP model with switching points (k_i's) as decision variables

Formally …

$W_q \leq W_u,$

waiting time below bound

$B \geq B_l.$

back-room work gets done

$\sum_{j=k_0}^{S} P(j) = 1$

$$P(j)\lambda = P(j+1)i\mu \qquad j = k_{i-1}, k_{i-1}+1, \ldots, k_i - 1$$

balance equations

$$F = \sum_{i=1}^{N} \sum_{j=k_{i-1}+1}^{k_i} iP(j)$$

expected # of workers in front-room

$$B = N - F$$

expected # of workers in back-room

$$L = \sum_{j=k_0}^{S} jP(j)$$

expected queue length

$$W_q = \frac{L}{\lambda(1 - P(k_N))} - \frac{1}{\mu}$$

expected waiting time

[Terekhov & B. 2008] *JAIR*, **32**, 123-167, 2008.

Sub-problem Results?

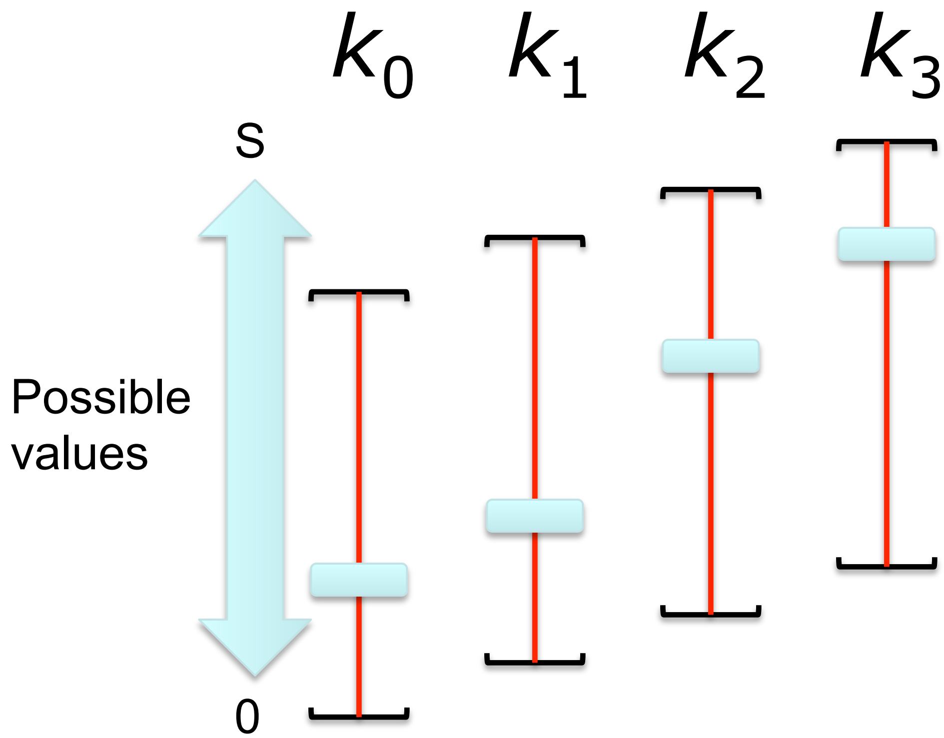
- 30 instances each for S = {10, 20, …, 100}
- Other parameters randomly generated

If-Then	105
PSums	126

Problem Instances (out of 300) solved and proved optimal in 10 minutes.

[Terekhov & B. 2008] *JAIR*, **32**, 123-167, 2008.

Exploiting the Policy Structure

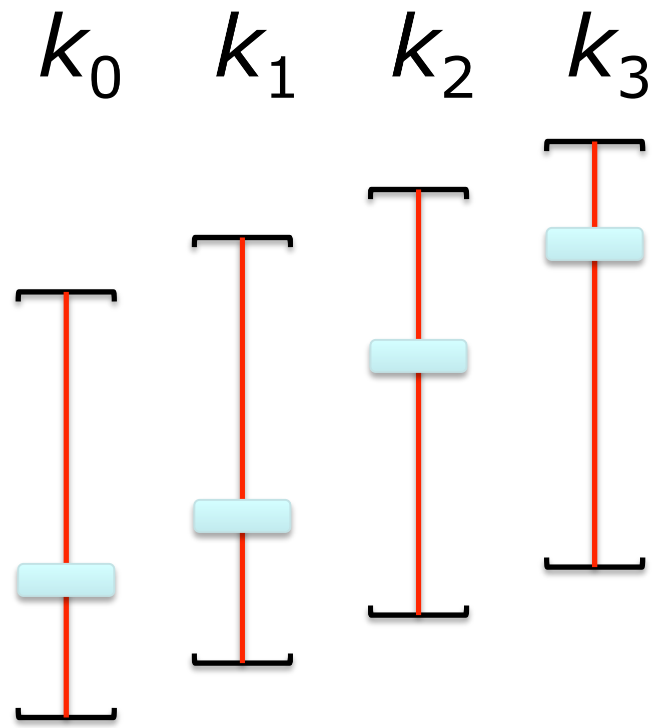


- When k_i's at UB, W_q maximized
- When k_i's at LB, W_q is minimized

Obs: 1) $k_{i-1} < k_i$ 2) lower k_i less waiting

Exploiting the Policy Structure

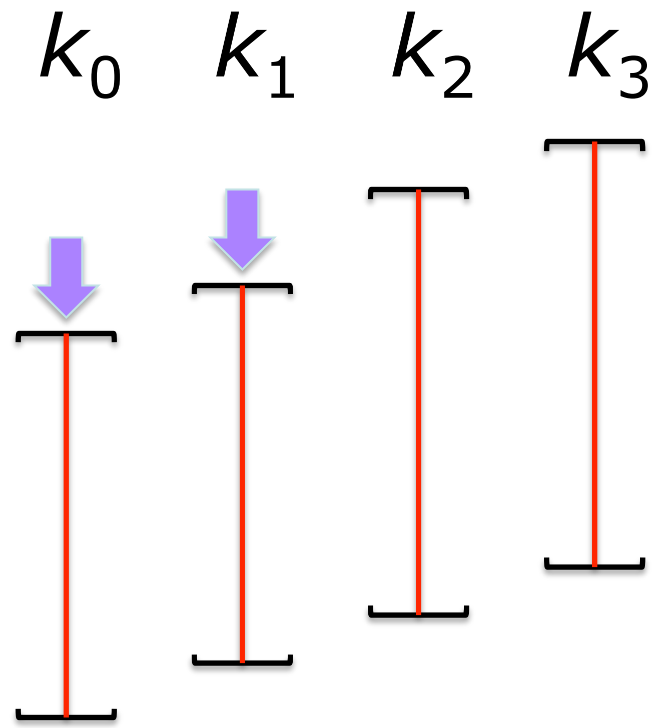
- Idea
  - Set k_i at its UB, set k_j, $j \neq i$ at LB
  - If $W_q > W_u$, remove UB from domain of k_i
- Symmetric reasoning for B



Obs: 1) $k_{i-1} < k_i$ 2) lower k_i less waiting

Exploiting the Policy Structure

- Idea
  - Set k_i at its UB, set k_j, $j \neq i$ at LB
  - If $W_q > W_u$, remove UB from domain of k_i
- Symmetric reasoning for B



"Shaving"

- Idea
  - Set an (integer) variable x at its LB (UB) and propagate
  - If infeasible, then the LB (UB) of x can be tightened
- Similar to Singleton Arc Consistency

[Martin & Shmoys 1996] *IPCO*, 389-403, 1996.

Sub-problem Results with Shaving

	No Shaving	Shaving before search and at each new incumbent
If-Then	105	234
PSums	126	238

Problem Instances (out of 300) solved and proved optimal in 10 minutes.

[Terekhov & B. 2008] *JAIR*, **32**, 123-167, 2008.

Global Results

Max # number of customers (300 instances)

Means

Statistic/ value of *S*	10	20	30	40	50	60	70	80	90	100
CPU time (seconds)	0.04	0.70	2.59	0.28	0.72	0.55	0.33	93.27	5.25	6.43
No. of iterations	4.57	7.77	16.07	6.13	8.00	7.23	8.23	34.73	27.60	23.83
Total no. of workers	4.07	6.33	9.17	4.80	5.40	5.60	5.27	15.33	8.83	8.93
Difference compared to spec.-only (%)	15.40	4.17	0.48	5.21	5.54	1.28	2.91	0.09	0	0
Difference compared to cross.-only (%)	2.11	2.95	3.71	4.43	4.08	5.72	6.10	5.73	5.90	6.00

Decomposition-based CP-Hybrids

- Problems where CP brings something to the table, but doesn't have the whole answer
  - where there is a combination of mostly global cost-based reasoning and mostly local feasibility problems
  - where inference works well except for one problem characteristic
    - e.g., scheduling with alternatives

MP defines # variables, not clear how to model sub-problem without CP

I confess…

- The master problem is not actually solved with MIP – we used CP
  - So this isn't really a MIP/CP hybrid, but it could be

Outline

- Learn Constraint Programming in 15 minutes or less!
- Why Hybridize?
- Three Decomposition Examples
- Final Comments

Would MIP/CP LBBD
work for my problem?

Decomposition-based CP-Hybrids

- Problems where CP brings something to the table, but doesn't have the whole answer
  - where there is a combination of mostly global cost-based reasoning and mostly local feasibility problems
  - where inference works well except for one problem characteristic
    - e.g., scheduling with alternatives

Problems with …

- “Cascading” decisions
  - some sort of assignment that activates or constrains other variables
    - assign jobs to resources/due dates then schedule
    - customers to open facilities then pack
    - decide # workers and then find policy
- Nice linear sub-problem relaxations and cuts
- A sub-problem where inference can perform strongly

What about a CP master and a MIP sub-problem?

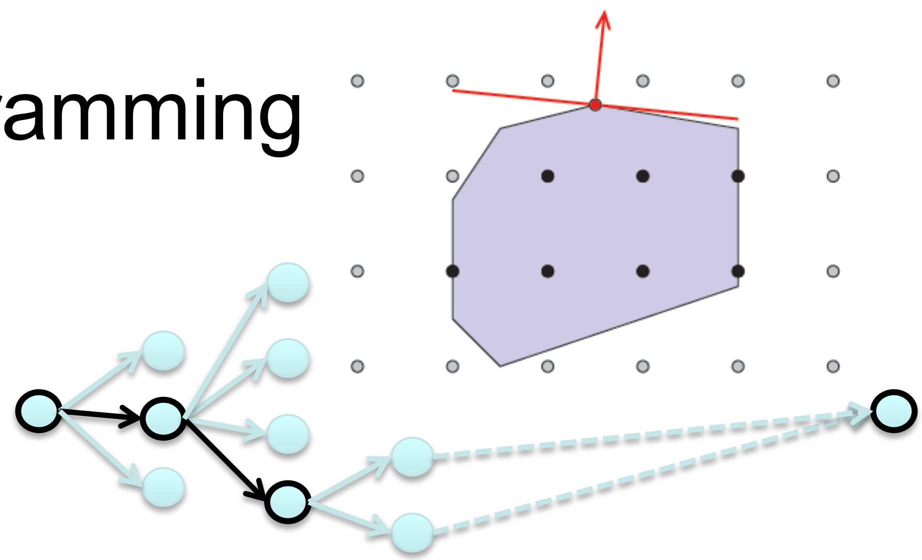
CP then MIP?

- There are examples in the literature but it is less developed
  - Relaxations and cuts are both better understood in MIP
  - Optimization master favours MIP and feasibility sub-problems favour CP (not uncommon)


When is he going to
stop talking?

Post-Doctoral Position

- AI Planning and Mathematical Programming
  - PhD in OR or CS
  - Strong math and software skills
  - Publication record
  - Deadline: July 1, 2016



```
tidel.mie.utoronto.ca/AI_MP_PostDoc2016.pdf
jcb@mie.utoronto.ca
```

Hybrid CP/MIP and Benders Decomposition Methods

J. Christopher Beck
Department of Mechanical & Industrial Engineering
University of Toronto
Canada
jcb@mie.utoronto.ca

CPAIOR 2016 Master Class
May 29, 2016

University of Toronto
Mechanical & Industrial Engineering