

The Feasibility Pump



Matteo Fischetti – DEI, University of Padova

Domenico Salvagnin – DMPPA, University of Padova



CPAIOR, May 2009

Primal Heuristics

- Crucial when exact methods don't work
- Crucial for the effectiveness of exact methods (=B&C) for mixed integer programming (MIP)
- Need a first feasible solution to enable:
 - bounding!
 - reduced cost fixing
 - other improving heuristics (RINS, LB, etc...)

The Problem

We want to find a feasible solution to:

$$\min \{c^T x : Ax \geq b, x_j \text{ integer for } j \in J\}$$

Notation:

- polyhedron $P = \{x : Ax \geq b\}$
- index set of integer variables J

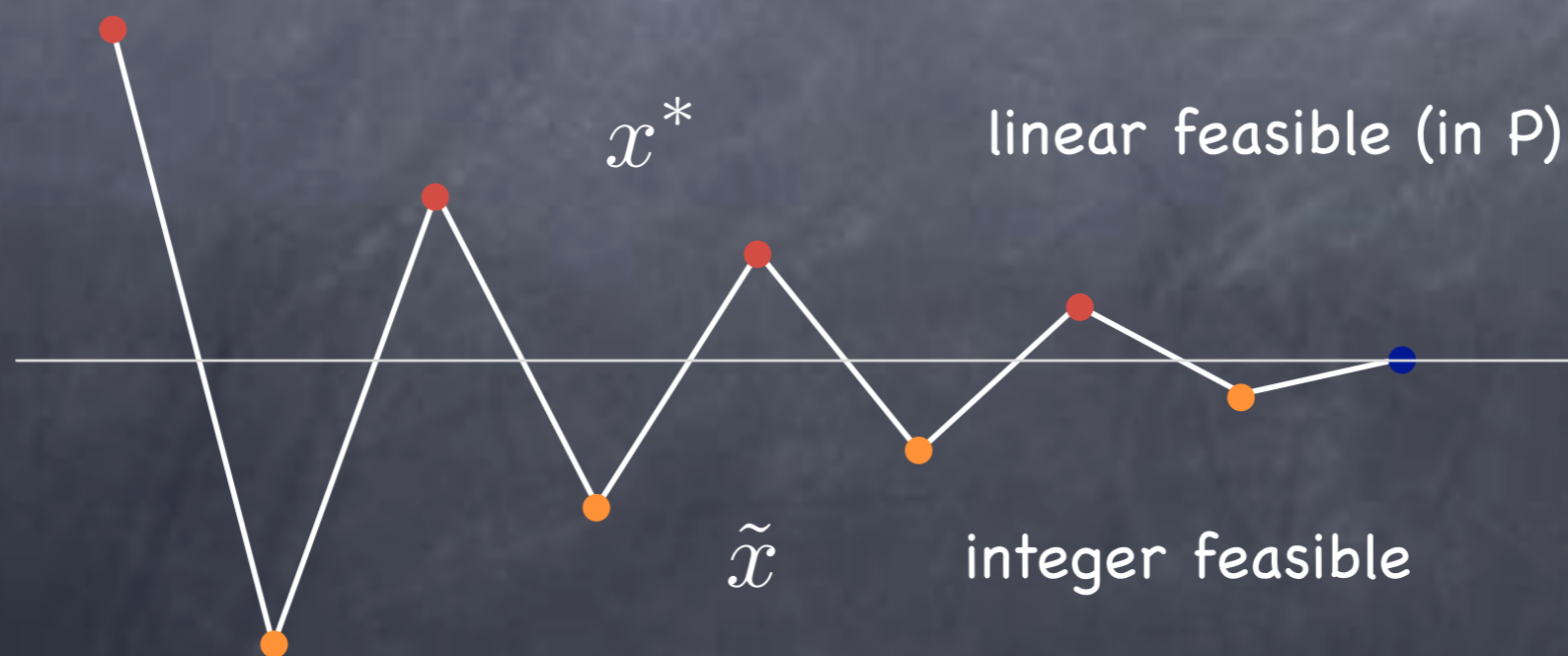
The Feasibility Pump 1.x

Fischetti, Glover, Lodi [2003]

Bertacco, Fischetti, Lodi [2005]

Achterberg, Berthold [2007]

- recent primal heuristic for MIPs
- generates two (hopefully convergent) trajectories of points x^* and \tilde{x} that satisfy feasibility in a complementary way



Basic Scheme I

How to get an integral point from a fractional one?

Plain rounding of the components in J to the nearest integer

$$\tilde{x}_j = \begin{cases} [x_j^*] & j \in J \\ x_j^* & j \notin J \end{cases}$$

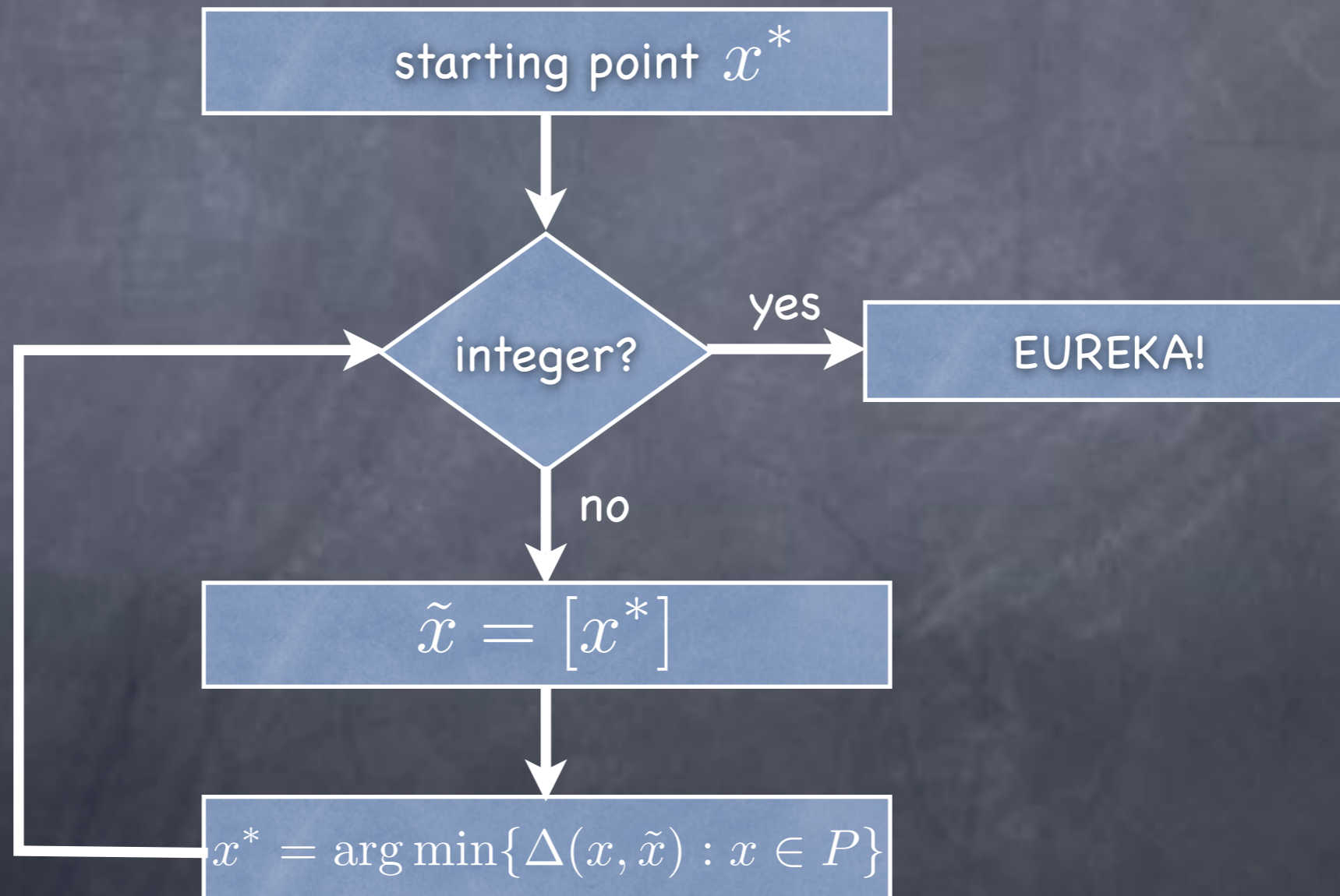
How to get an LP-feasible point from an integral one?

Find the point in P closest w.r.t the L_1 norm Δ

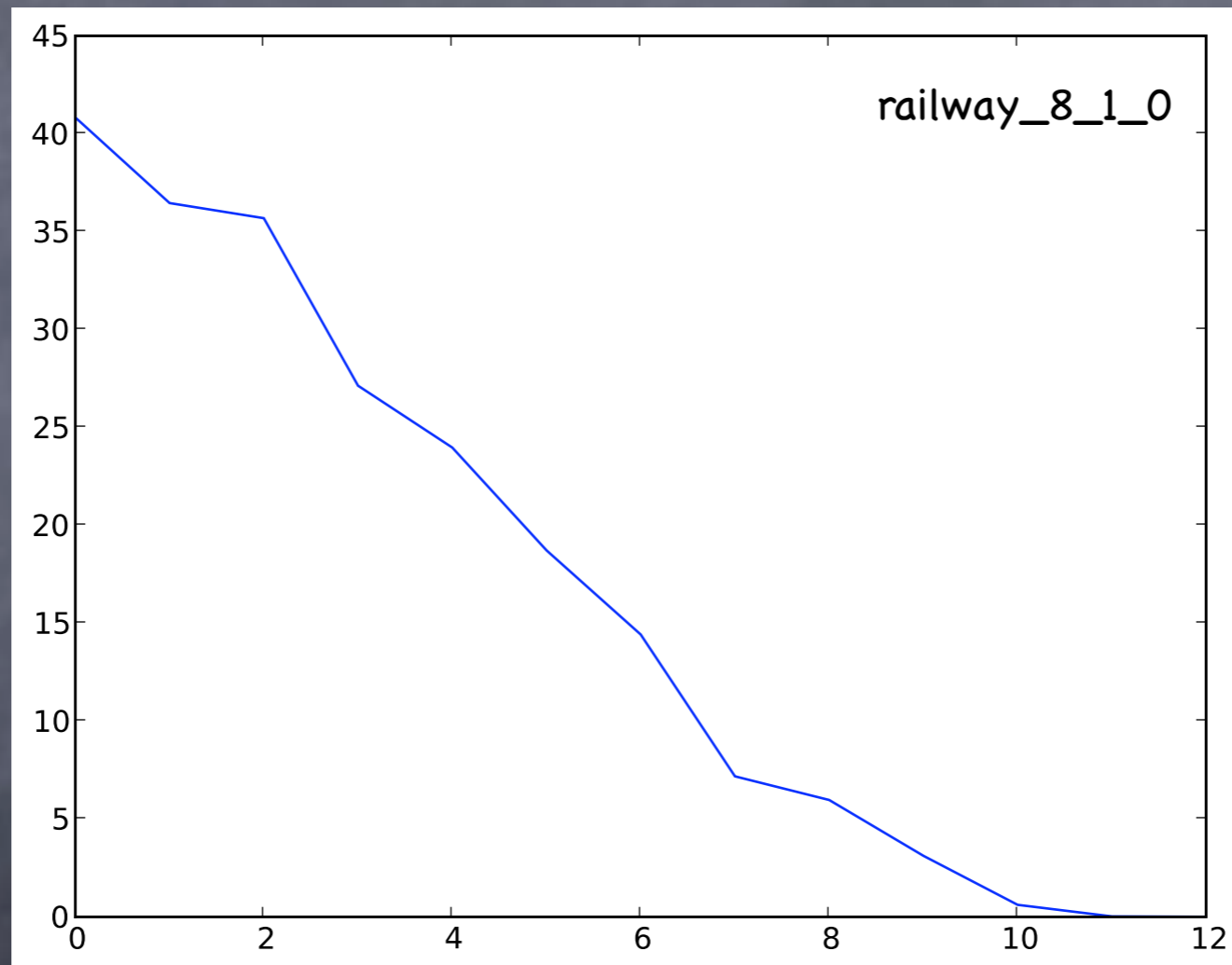
$$x^* = \arg \min \{ \Delta(x, \tilde{x}) : x \in P \}$$

Basic Scheme II

A simplified algorithm may look like this:

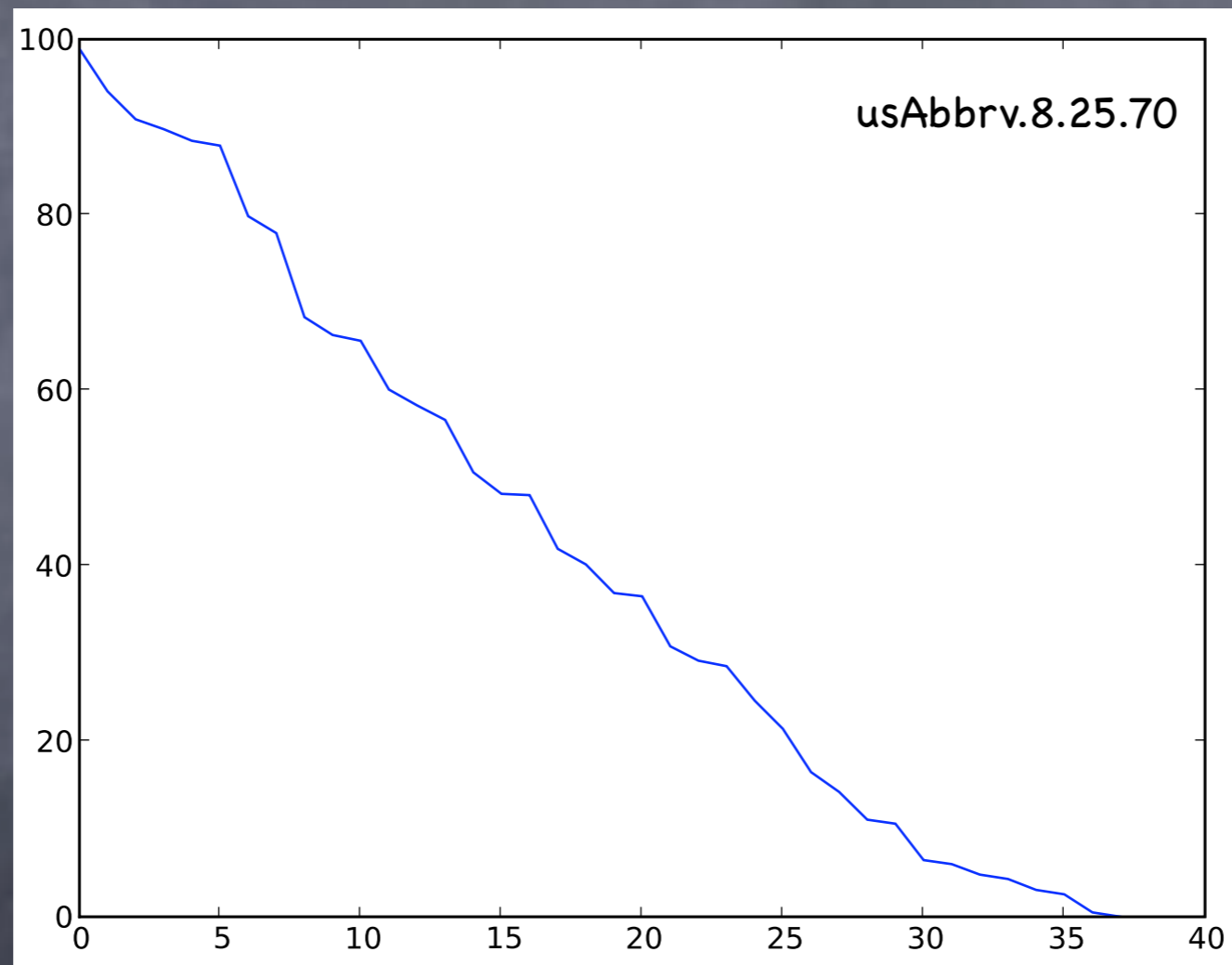


Real Examples



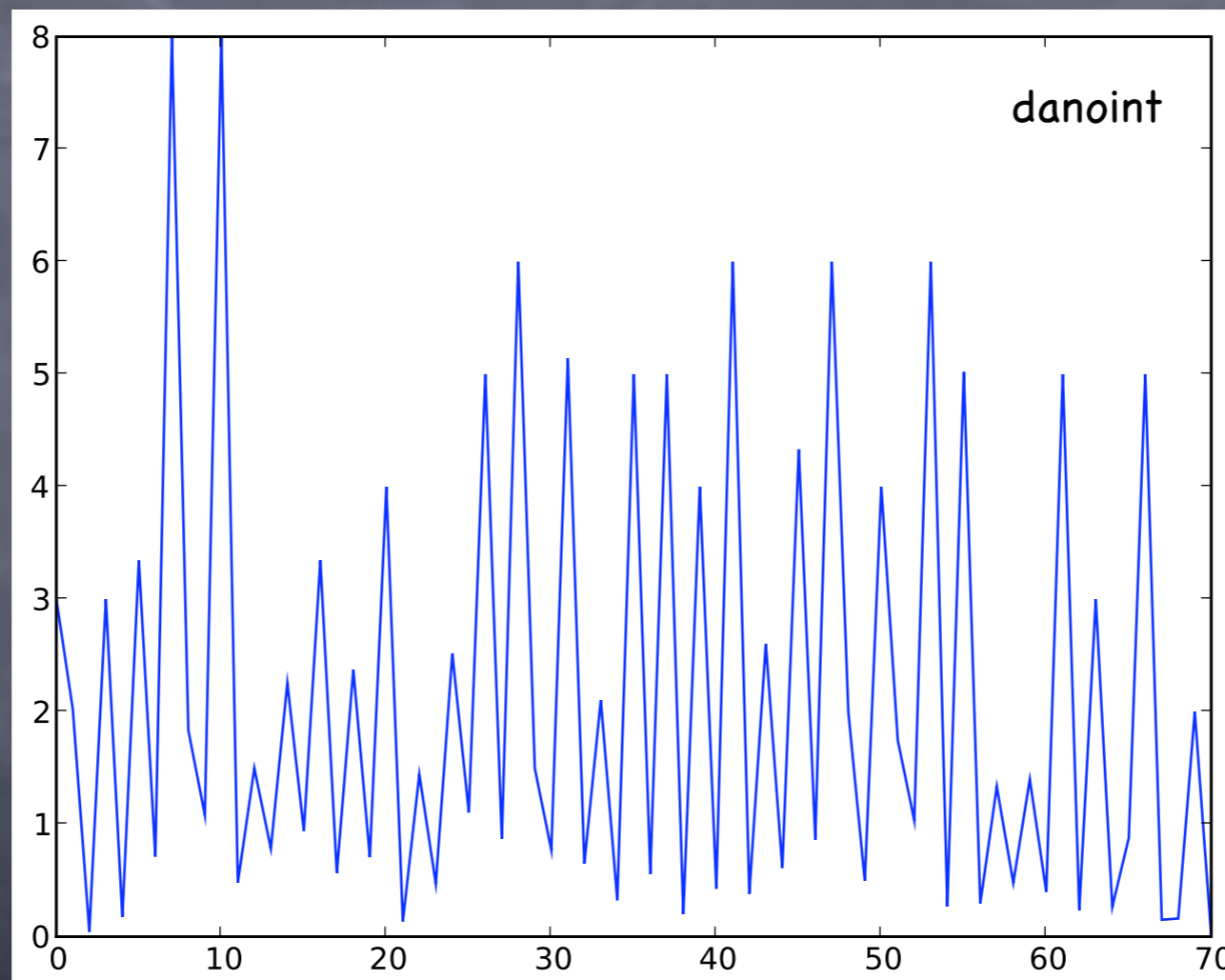
Good

Real Examples



Fair

Real Examples



Bad!

Objective FP

Achterberg, Berthold [2007]

- FP is often not satisfactory from the solution quality point of view
- No surprise: objective function is used ONLY in the very first iteration!
- Solution: Objective FP
 - use a convex combination of original objective and distance function when solving the LPs
 - gradually decrease objective weight

Rounding: are we serious?

Many advantages:

- + extremely simple and fast
- + $[x]$ is the nearest integer point to x
- + convergence **IN ABSENCE** of cycles

BUT:

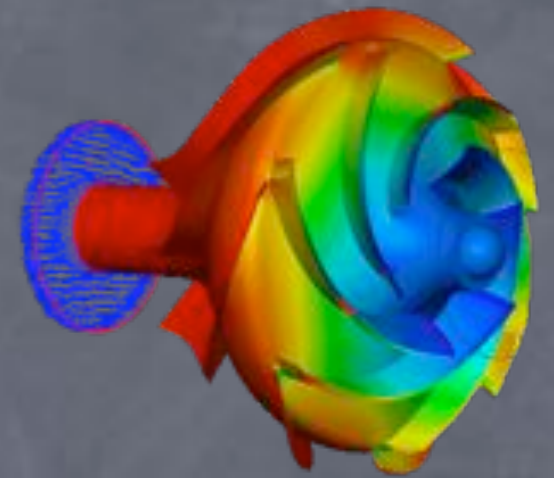
- prone to cycling (many different continuous x may map to same integral $[x]$)
- completely forgets p

FP 1.x: quick summary

- simple primal heuristics
- convergent in absence of cycles
- cycles are a big problem
- often feasible solutions are found because of frequent perturbations rather than by design
- simple rounding is quite blind and may fail on trivial instances (e.g. knapsacks, set covering)



FP 2.0: Inference!



- Rounding a variable means fixing it temporarily to a value
- Propagate the rounding of an integer variable before rounding the remaining ones
- Advantages:
 - use information from the linear constraints
 - hopefully still fast
 - akin to diving, but without solving LPs

Bound Strengthening

Savelsbergh [1994] Maros [2003] Hooker [2006] Achterberg [2007]

Given a linear constraint with positive coefficients:

$$LB \leq \sum a_j x_j \leq UB$$

and original bounds $[l_j, u_j]$ on the variables, we can compute the minimum and maximum activities:

$$L_{\min} = \sum a_j l_j \quad L_{\max} = \sum a_j u_j$$

and update variable bounds:

$$x_j \leq l_j + \frac{UB - L_{\min}}{a_j} \quad x_j \geq u_j + \frac{LB - L_{\max}}{a_j}$$

(can be rounded for integer variables and generalized to constraints with negative coefficients)

Constraint Propagation

Schulte [2000] Actherberg[2007]

- How do we organize constraint propagation?
- We use a simplified propagator approach
- Basic scheme:

K : set of variables

C_i : constraint i

$C_i \supset j$: C_i involves variable j

Q : set of constraints to propagate

$Q \leftarrow \{C_i: i = 1, \dots, m\}$

while Q not empty:

$C_i \leftarrow \text{pop}(Q)$

$K \leftarrow \text{propagate}(C_i)$

$Q \leftarrow Q \cup \{C_i: C_i \supset j \text{ for some } j \text{ in } K\}$

Constraint Propagation II

- Actual implementation is more involved due to optimizations (incremental propagation and specific constraint structure exploitation)
- What's the complexity of all of this?
 - polynomial for pure binary problems
 - pseudo-polynomial for general integer
 - may not converge in a finite number of propagations for continuous variables!

NEED TO STOP PROPAGATIONS AT SOME POINT!

FP 2.0: some remarks...

- propagation can be time consuming, but is typically fast enough
- the final result depends on:
 - how we choose the next variable to round
 - the order of constraint propagators
- no dominance exists between the two versions of the FP, but FP 2.0 is strictly better on a single feasible shot

Results: testbed

- Testbed: 78 instances from MIPLIB 2003
- Turned into feasibility problems by adding bounds on the optimal value, as a relative gap [10%, 100%, None] from the best known solution
- variables sorted by increasing fractionality before propagation (+ some noise)
- 10 different seeds for random perturbation
- two scenarios: alone and embed

Computational Results

UB	Figure	alone			embed		
		std	prop	$\Delta\%$	std	prop	$\Delta\%$
None	succ.	87	90	3	49	68	39
	itr	27	10	62	9	5	44
	time	1.05	0.82	21	0.39	0.39	0
100%	succ.	85	88	5	32	55	72
	itr	87	24	72	12	7	43
	time	2.22	1.46	34	0.42	0.42	0
10%	succ.	50	59	18	17	21	23
	itr	285	201	30	17	14	16
	time	5.80	7.75	-34	0.47	0.56	-19

Computational Results

Solution Quality

Figure	FP		objFP	
	std	prop	std	prop
gap%	77.2	57.6	52.4	35.5
itr	27	9	84	42
time	1.0	0.8	1.3	1.0

Conclusions

- Propagation can be very effective if embedded into the FP scheme (both for binary and general integer instances, also when embedded inside the objective version)
 - reduced number of iterations
 - increased success rate and solution quality, in a comparable amount of time
- What's next?
 - exploit higher level modeling tools when available
 - turn attention to the LPs (maybe FP 3.0...)



Paper available at:
<http://www.math.unipd.it/~salvagni/>