Local Branching

TO FEASIBILITY ... AND BEYOND !!

Matteo Fischetti, DEI, University of Padova, fisch@dei.unipd.it

Andrea Lodi, DEIS, University of Bologna, alodi@deis.unibo.it

full paper available at www.dei.unipd.it/~fisch/locbra.ps

0-1 Mixed-Integer Programs

We consider generic **Mixed-Integer Linear Programming** problems (MIP's) with 0-1 variables

$$\min \ c^T x$$
$$A x \geq b$$
$$x_j \in \{0,1\} \quad , \forall j \in \mathcal{B} \neq \emptyset$$
$$x_j \text{ integer} \quad , \forall j \in \mathcal{G} \ (\supseteq \emptyset)$$

Relevant cases:

- 0-1 ILP's (generic or with a special structure)
- MIP's with no "general integer" variables
- MIP's with both general integer and binary variables, the latter being often used to activate/deactivate costs/constraints (possibly using BIG-M tricks…)

Assumption: once the binary variables have been fixed,
the problem becomes (relatively) easy to solve

Hard-to-solve 0-1 MIP's (in practice)

- In many practical cases, generic 0-1 MIP's can be solved in a satisfactory way by general-purpose commercial software which delivers:
  - Provably optimal solution
  - Heuristic solutions with a practically-acceptable error

Most MIPlib instances are of this type!

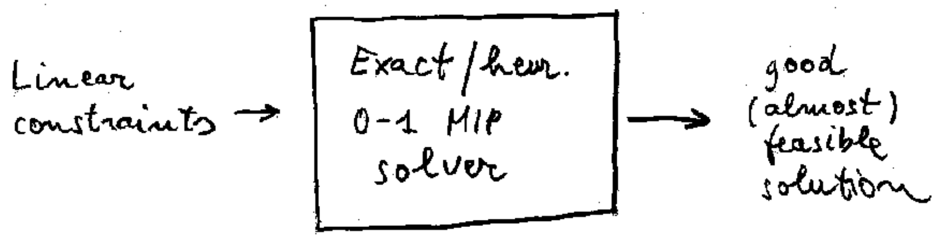
- Unfortunately, in other cases general-purpose software is not adequate and one has to:
  - Play with the MIP solver parameters ("emphasize integrality" etc.) so as to convince the **$#$#?@#** solver to deliver, at least, a good solution
  - Design and use ad-hoc heuristics—thus loosing the advantage of working in a generic MIP framework

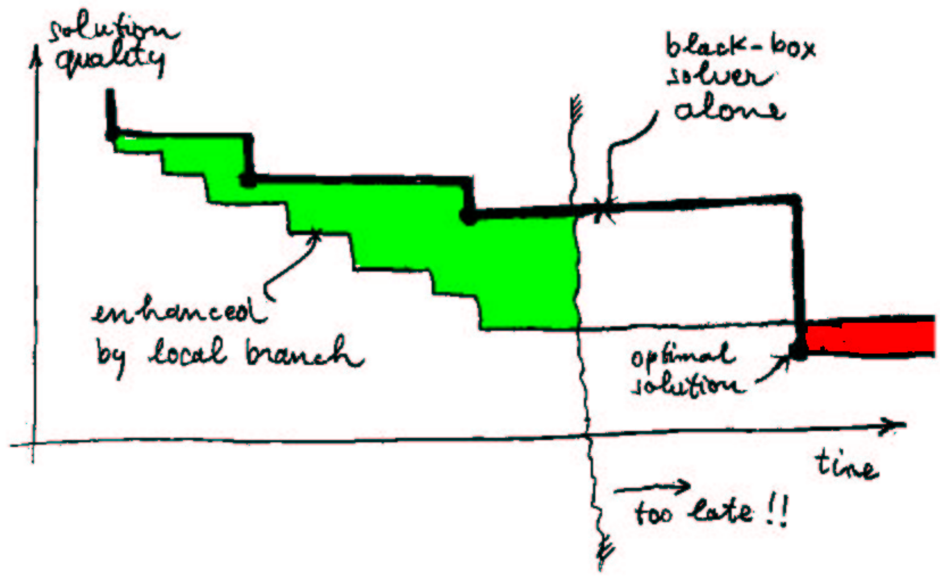
Many real-world instances are of this type!

Better heuristics for general 0-1 MIP's strongly required!

A general heuristic framework

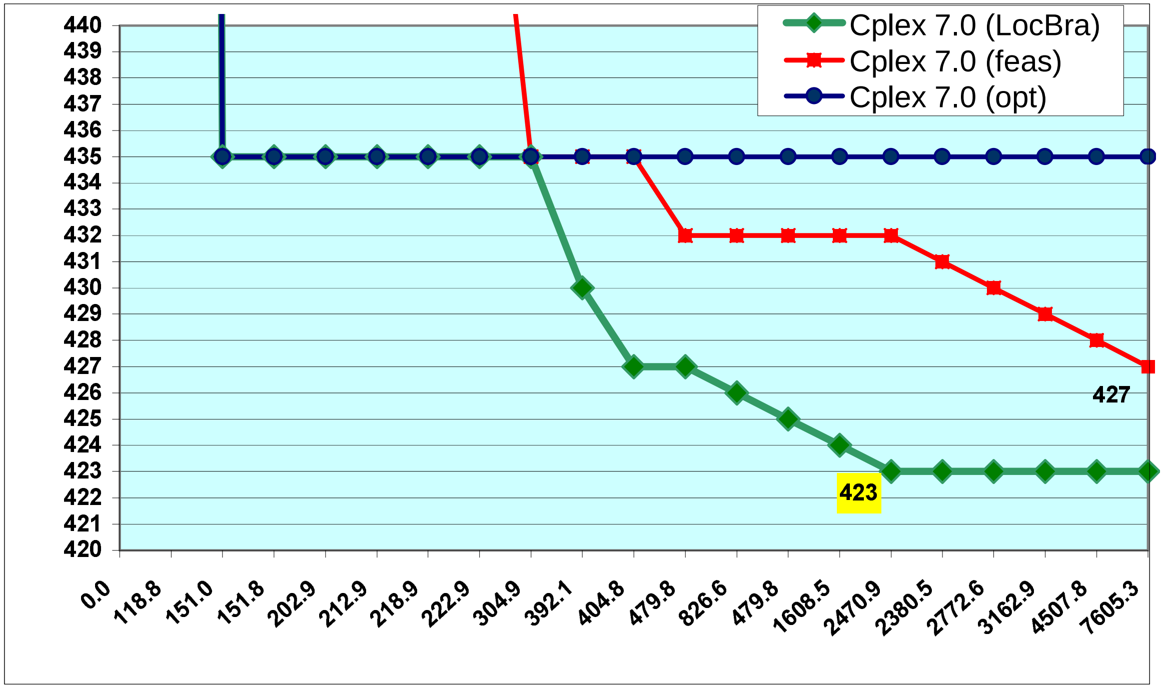
- We aim at embedding a **black-box** (general-purpose or specific) 0-1 MIP solver within an overall **heuristic framework** that "helps" the solver to deliver improved heuristic solutions





The desired "Italian flag"

An example: the hard MIPLIP problem seymour.lp



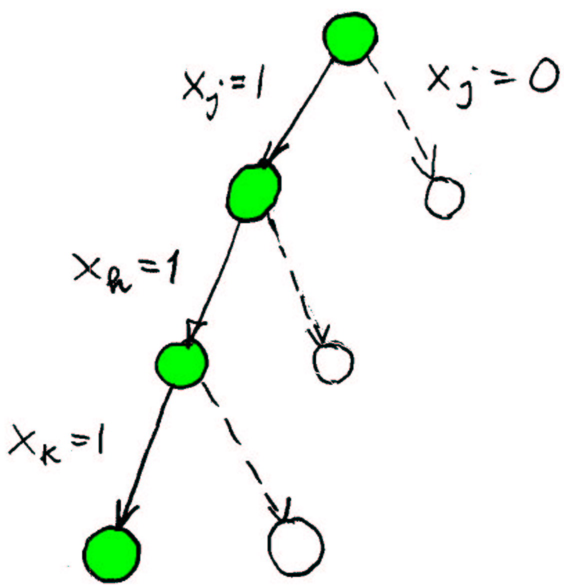
The Local Branch heuristic on a hard MIPLIP problem (seymour.lp)

Variable-fixing strategy (hard version)

A commonly-used (often quite effective) **diving** heuristic framework:

- Let x^H be an (almost) feasible "target solution"

- **Heuristic depth-first search** of the branching tree:



- iteratively **fix to 1** certain "highly efficient" variables x_j such as $x_j^H = 1$ (**green nodes**)

- apply the **black-box module** to some **green nodes** only

- only limited **backtracking** allowed

Advantages:

- Problem size quickly reduced: the black-box solver can concentrate on smaller and smaller "**core problems**"
- The black-box solver is applied over and over on different subproblems (**diversification**)

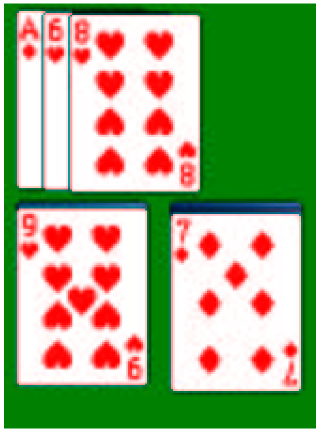
Disadvantages:

- How to choose the "highly efficient" variables to be fixed?
- Wrong choices at early levels are typically very difficult to detect, even when **lower bounds** are computed along the way…

How to reach a sufficiently-deep branching level with a good lower bound?

Example of a hard branching choice:

- select the "right cards" to hold in a poker game: for each card $j=1,\ldots,5$ in the "target solution", decide whether for fix $x_j = 1$ (hold the *j*-th card) or not
- a "creative strategy": keep all the 5 cards, declare that you'll change 3 cards, receive 3 new cards, and choose the 5 to keep only afterwards...



Variable-fixing strategy (soft version): local branching

General idea: don't decide the actual variables in S^H to be fixed (a difficult task!), but just their **number** $|S^H| - k$

Introduce the **local branching** constraint

$$\Delta(x, x^H) := \sum_{j \in B: x_j^H = 1} (1 - x_j) \leq k$$

or, more generally,

$$\Delta(x, x^H) := \sum_{j \in B: x_j^H = 0} x_j + \sum_{j \in B: x_j^H = 1} (1 - x_j) \leq k$$

so as to define a convenient **k-OPT neighbourhood** $N(x^H, k)$ of the target solution x^H

"Akin to k-OPT for TSP"

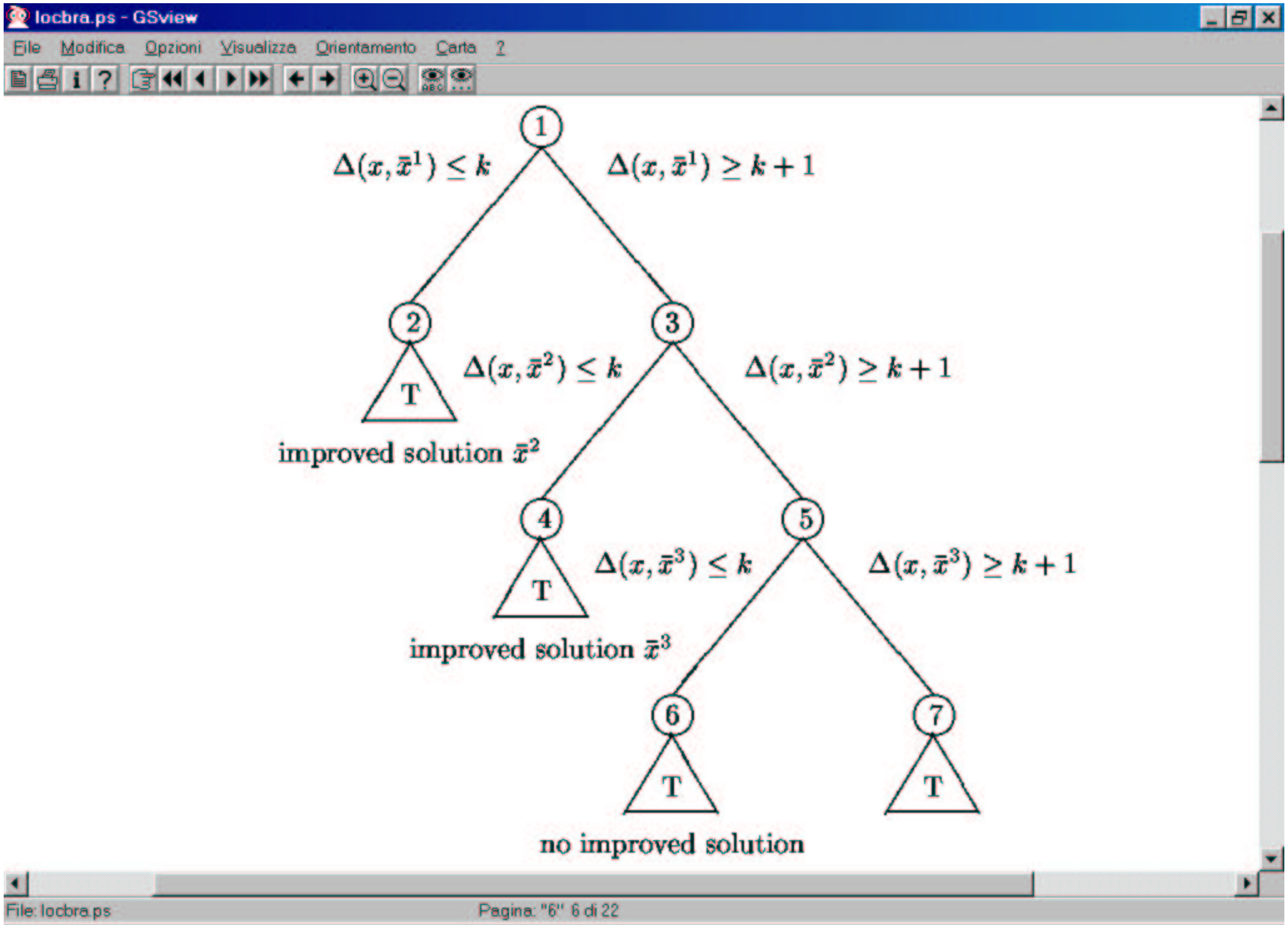
Local branching in an exact solution framework

Alternate between **strategic** and **tactical** branching decisions:

- **STRATEGIC (high level) branching phase:**
  - concentrate on a convenient target solution and/or a certain neighbourhood size k

- **TACTICAL (fine grain) branching phase:**
  - search $N(x^H, k)$ by means of the black-box module (e.g. a general-purpose MIP code using branching on variables…)

Conjecture: a small value of k drives the black-box solver towards integrality as effectively as fixing a large number of variables, but with a **much larger degree of freedom** ➔ better solutions can be found at early branching levels…



T = tactical branching (within the black-box solver)

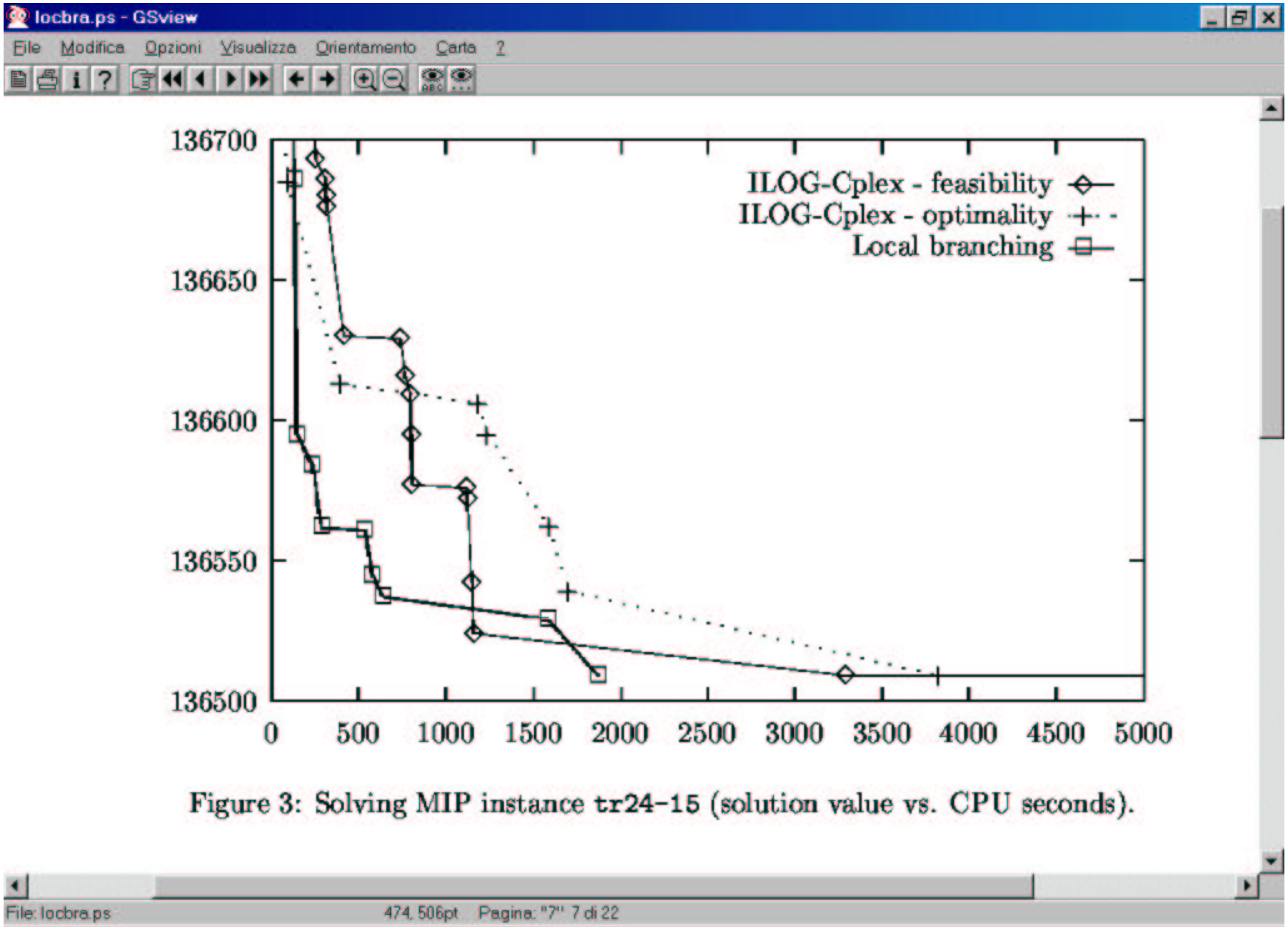


Figure 3: Solving MIP instance tr24-15 (solution value vs. CPU seconds).

full paper available at www.dei.unipd.it/~fisch/locbra.ps

Local branching in a heuristic solution framework

- Easy adaptation of the previous framework: in case of stalling, use a **diversification** mechanism to find a (worse) solution x^{h+1} to replace the current-best solution x^h, and continue.

- Diversification by **Variable Neighbourhood Search** (Hansen & Mladenovic, 1998):

  Find a solution x^{h+1} close enough to x^h, but outside the current k-OPT neighbourhood, e.g.

$$x^{h+1} \in N(x^h, k + k/2) \setminus N(x^h, k)$$

- **Implementation:** run the black-box solver (initial upper bound = $+\infty$) to find the first feasible solution x^{h+1} of the current problem amended by the **diversification constraint**

$$k + 1 \le \Delta(x, x^h) \le k + k/2$$

"Akin to a random 3-OPT move after several 2-OPT moves for TSP"

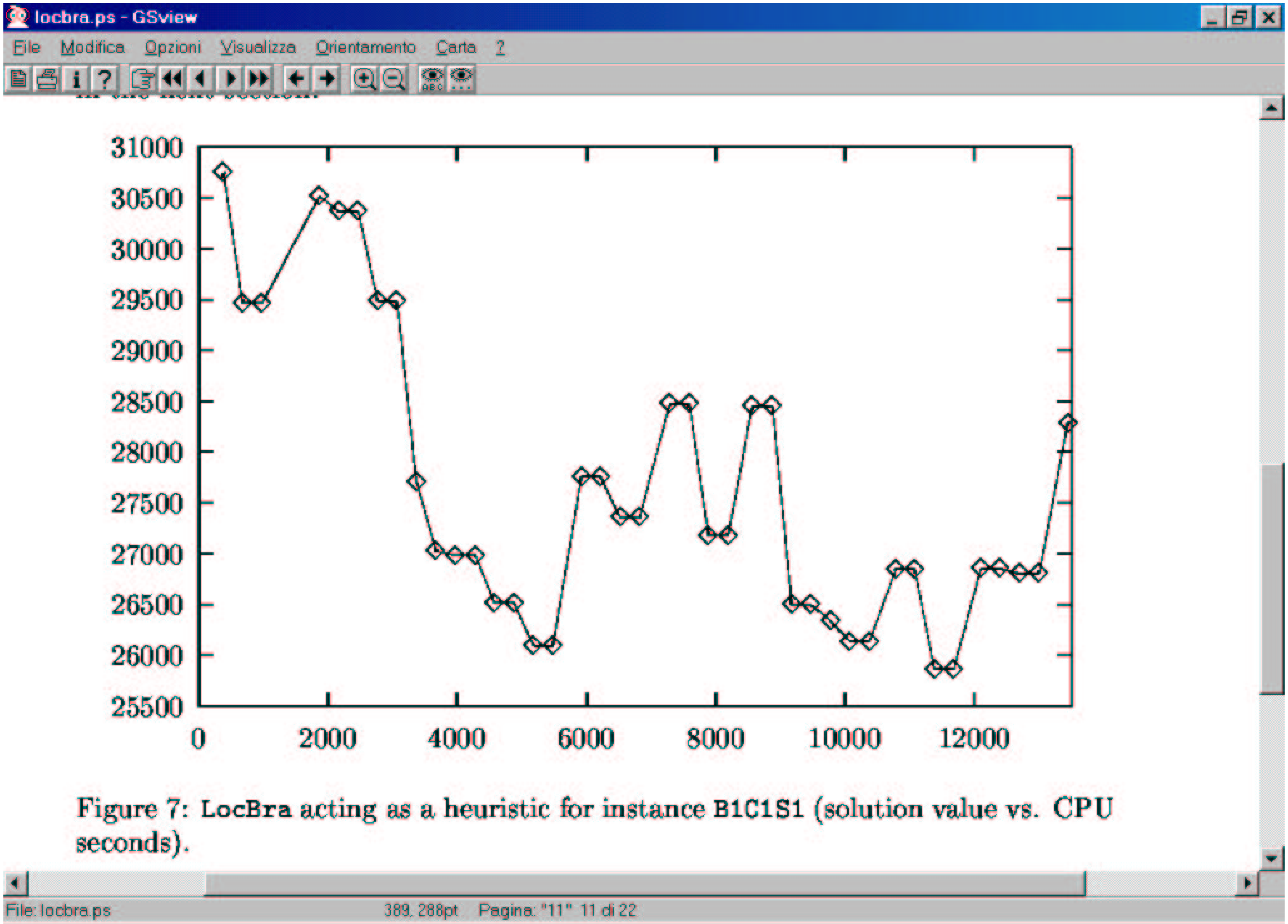


Figure 7: LocBra acting as a heuristic for instance B1C1S1 (solution value vs. CPU seconds).

full paper available at www.dei.unipd.it/~fisch/locbra.ps

Computational results (towards optimality)

Name	set	m	n	$\|B\|$	bestVal	context	ref.
arki001	A	1048	1388	415	7,581,034.85	MIPLIB 3.0	[4]
seymour	A	4944	1372	1372	424.00	MIPLIB 3.0	[4]
net12	B	14021	14115	1603	255.00	network design	[3]
biella1	C	1203	7328	6110	3,070,810.15	crew scheduling	[22]
NSR8K	C	6284	38356	32040	21,520,487.01	crew scheduling	[22]
rail507	D	509	63019	63009	175.00	railway crew scheduling	[5]
rail2536c	D	2539	15293	15284	691.00	railway crew scheduling	[5]
rail2586c	D	2589	13226	13215	957.00	railway crew scheduling	[5]
rail4284c	D	4287	21714	21705	1078.00	railway crew scheduling	[5]
rail4872c	D	4875	24656	24645	1556.00	railway crew scheduling	[5]
glass4	E	396	322	302	1,587,515,737.50	nesting	[17]
UMTS	F	4465	2947	2802	30,160,547.00	telecomm. network	[21]
van	F	27331	12481	192	5.09	telecomm. network	[18]
roll3000	G	2295	1166	246	13,065.00	railway rolling stock	[13]
nsrand_ipx	G	735	6621	6620	51,520.00	railway line planning	[13]
A1C1S1	H	3312	3648	192	11,834.02	lot-sizing	[23]
A2C1S1	H	3312	3648	192	11,251.10	lot-sizing	[23]
B1C1S1	H	3904	3872	288	25,869.15	lot-sizing	[23]
B2C1S1	H	3904	3872	288	26,297.63	lot-sizing	[23]
tr12-30	H	750	1080	360	130,596.00	lot-sizing	[23]
sp97ar	I	1761	14101	14101	667,735,390.40	railway line planning	[10]
sp97ic	I	1033	12497	12497	436,984,606.56	railway line planning	[10]
sp98ar	I	1435	15085	15085	531,942,554.88	railway line planning	[10]
sp98ic	I	825	10894	10894	449,915,159.36	railway line planning	[10]

Table 1: The hard MIP instances in the test bed.

- Improved results w.r.t. ILOG Cplex 7.0 in 20 out of the 24 cases in the test-bed

Computational results (towards feasibility)

- Instances for which even finding a first **feasible** solution is extremely hard in practice, hence the local branching framework (as stated) cannot be initialized in a proper way…
    - **[Relaxed model]**: **relax** the MIP model by introducing **artificial variables** (with big-M coefficients in the objective function) so as to make the trivial solution (0,0,…0) feasible.

- The **"to feasibility and beyond"** solution approach:

1. choose an **infeasible** solution x^H, e.g., for each integer x_j
    - **[Trivial target]**: set $x_j^H = 0$
    - **[Rounded LP target]**: set $x_j^H = round(x_j^*)$, where x^* is an optimal LP solution
    - **[CPX callback target]**: take the less-infeasible sol. found at the root node by the black-box MIP heuristics, if available
2. **relax** the MIP model by introducing an **artificial variable** (with big-M coefficient in the objective function) for each constraint violated by x^H
3. apply the standard **local branching** framework starting from x^H

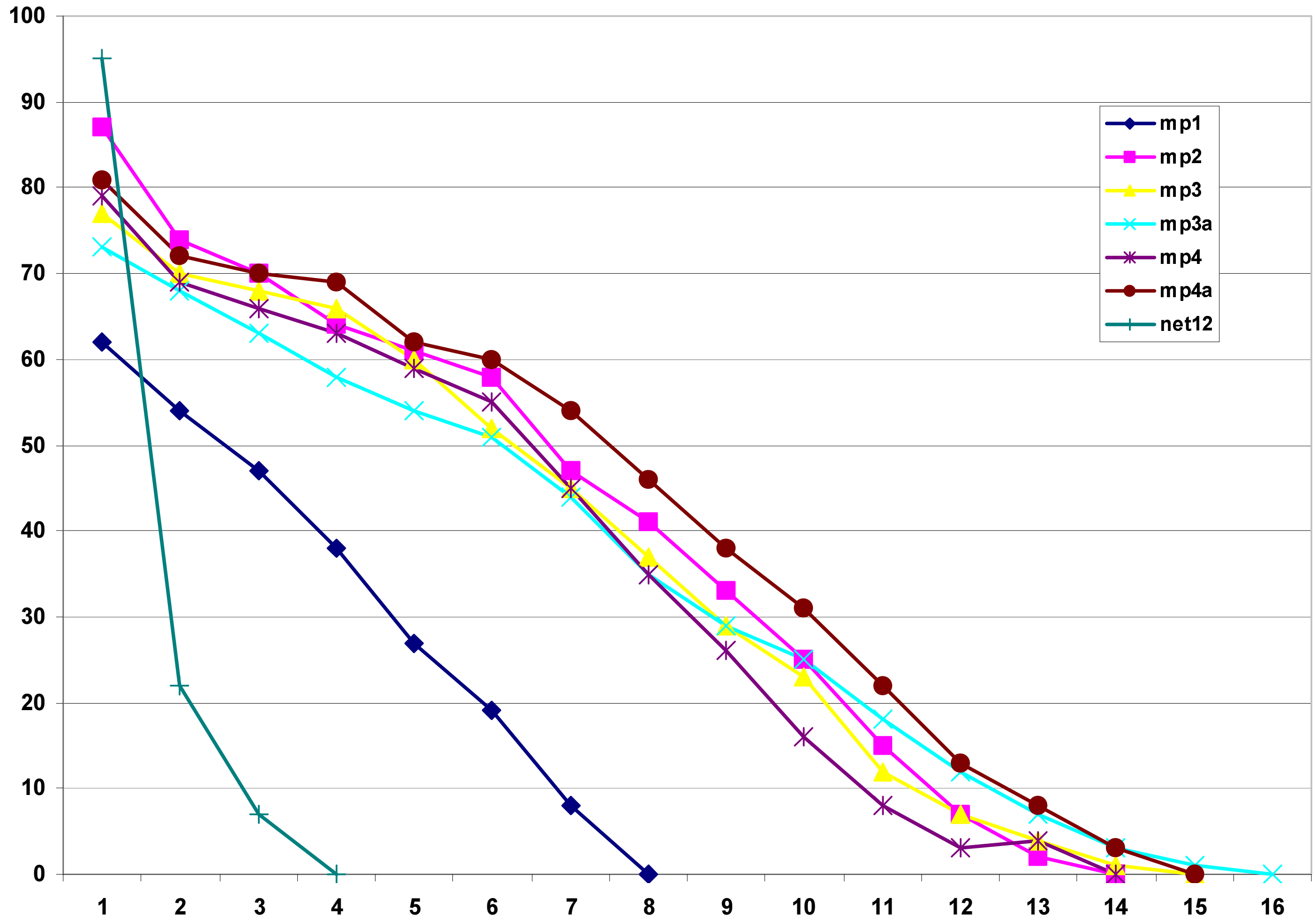
name	*n*	*m*	**Cplex 8.1**	**LOCAL BRANCHING**							
				Relaxed Model		**Trivial target**		**Rounded LP target**		**CPX callback**	
			time	*inf*	*time*	*inf*	*time*	*inf*	*time*	*inf*	*time*
mp1	10565	21199	11686.3	-	397.5	2444	1063.6	106	**51.4**	62	259.1
mp2	10009	23881	N/A	4	7727.6	2489	3193.1	131	**254.2**	87	1299.2
mp3	10009	23915	N/A	9	9943.6	2348	5352.4	285	**346.0**	77	2950.3
mp3a	10009	23865	N/A	6	14217.7	2516	4898.8	264	**178.0**	73	3218.7
mp4	10009	23914	N/A	4	5837.5	2411	7374.5	259	**176.4**	79	3278.0
mp4a	10009	23866	N/A	N/A	N/A	2515	3390.8	214	**135.8**	81	1947.9
net12	14115	14021	N/A	30	6472.5	92	1750.8	398	1259.1	95	**832.9**

*mp** = hard shift scheduling (manpower) instances provided by ILOG Cplex.

time = computing time in Digital Alpha 533 MHz seconds; 5-hour time limit (18,000 sec.s)

inf = n. of violated constraints in the initial target solution

Infeasibility reduction starting from the CPX callback target



Infeasibility reduction starting from the rounded LP target