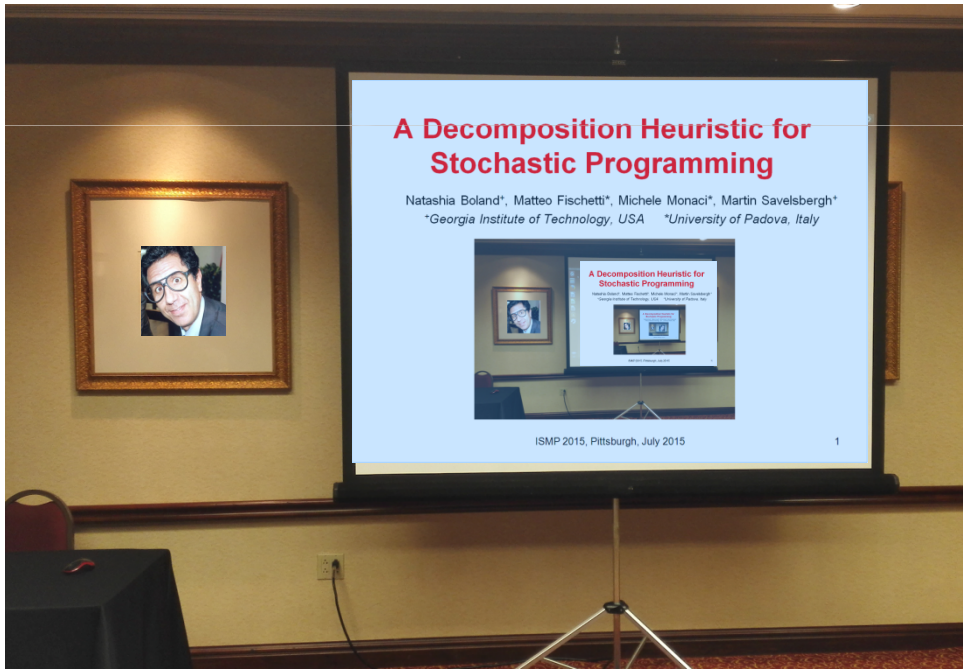
A Decomposition Heuristic for Stochastic Programming

Natashia Boland[+], Matteo Fischetti*, Michele Monaci*, Martin Savelsbergh[+]

[+]*Georgia Institute of Technology, USA* **University of Padova, Italy*



MIP heuristics

- We consider a Mixed-Integer convex 0-1 Problem (0-1 MIP, or just MIP)

$$\begin{aligned} \min\, & f(x) \\ & g(x) \leq 0 \\ x_j \in \{0,1\} \quad & \forall j \in J \end{aligned}$$

  where *f* and *g* are convex functions and $J \subseteq N := \{1, \cdots, n\}$
  → removing integrality leads to an easy-solvable continuous relaxation

- A **black-box** (exact or heuristic) MIP solver is available

- How to use the solver to **quickly** provide a **sequence of improved heuristic solutions** (time vs quality tradeoff)?

Large Neighborhood Search

- **Large Neighborhood Search** (LNS) paradigm:
  1. introduce **invalid constraints** into the MIP model to create a nontrivial sub-MIP "centered" at a given heuristic sol. $\tilde{x}$ (say)
  2. Apply the MIP solver to the sub-MIP for a while…

- Possible implementations:
  - **Local branching**: add the following linear cut to the MIP

$$\Delta(x, \tilde{x}) = \sum_{j \in J:\, \tilde{x}_j = 0} x_j + \sum_{j \in J:\, \tilde{x}_j = 1} (1 - x_j) \leq k$$

  - **RINS**: find an optimal solution x^* of the continuous relaxation, and fix all binary variables such that $x^*_j = \tilde{x}_j$
  - **Polish**: evolve a population of heuristic sol.s by using RINS to create offsprings, plus mutation etc.

Why should the subMIP be easier?

- What makes a (sub)MIP easy to solve?

  1. fixing many var.s reduces problem size & difficulty
  2. additional contr.s limit **branching's** scope
  3. **something else**?

- In Branch-and-**Bound** methods, the quality of the root-node relaxation is of paramount importance as the method is **driven by** the relaxation solution found at each node

- Quality in terms of **integrality gap** …
- … but also in term of "**similarity**" of the root node solution to the optimal integer solution (the "more integer" the better…)

Relaxation grip

- Effect of **local branching** constr. for various values of the neighborhood radius *k* on MIPLIB2010 instance *ramos3.mps* (root node relaxation)

x-range	$k=0$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=10$	$k=20$	$k=30$	$k=50$	$k=99$	$k=+\infty$
$=0$	1920	1919	1919	1919	1919	1919	1920	1619	1619	1606	1562	672
$(0.0, 0.1]$	0	0	0	0	0	0	0	303	301	302	276	849
$(0.1, 0.2]$	0	1	0	0	0	5	0	0	2	14	73	551
$(0.2, 0.3]$	0	0	0	0	0	0	4	0	0	4	16	108
$(0.3, 0.4]$	0	0	1	0	5	0	0	0	3	2	7	7
$(0.4, 0.5]$	0	0	0	6	0	0	0	8	5	2	17	0
$(0.5, 0.6]$	0	0	0	0	0	0	2	5	5	9	18	0
$(0.6, 0.7]$	0	0	5	0	1	0	0	0	0	6	40	0
$(0.7, 0.8]$	0	0	0	0	0	0	9	0	3	17	86	0
$(0.8, 0.9]$	0	5	0	0	0	1	0	0	14	81	67	0
$(0.9, 1.0)$	0	0	0	0	0	0	0	249	232	142	24	0
$=1$	267	262	262	262	262	262	252	3	3	2	1	0
time (sec.s)	0.01	0.08	0.12	0.14	0.16	0.13	0.31	0.55	0.61	0.73	1.40	98.18
# LP-iter.s	0	827	1033	1145	1214	1095	1930	2897	3101	3476	4971	23870
LP-bound	267.00	266.33	265.66	265.00	264.33	263.66	260.88	255.70	250.62	240.47	215.97	145.80

Changing the subMIP objective

- Altering the MIP objective function can have a big impact on

  - **time to get the optimal solution of the continuous relax.**

    working with a simplified/different objective can lead to huge speedups (orders of magnitude)

  - **success of the internal heuristics (diving, rounding, …)**

    the original objective might interfere with heuristics (no sol. found even for trivial set covering probl.s) and sometimes is reset to zero

  - **search path towards the integer optimum**

    search is trapped in the upper part of the tree (where the lower bounds are better), with frequent divings to grasp far-away (in terms of lower bound) solutions

Proximity Search

- A variant of Local Branching/ Feasibility Pump introduced in

  M. Fischetti, M. Monaci, Proximity search for 0-1 mixed-integer convex programming, Journal of Heuristics 20 (6), 709-731, 2014

- We want to be free to work with a modified objective function that has a better "**relaxation grip**" and hopefully allows the black-box solver to quickly improve the incumbent solution

- **Step 1.** Add an explicit **cutoff** constraint $f(x) \leq f(\tilde{x}) - \theta$

- **Step 2**. Replace the objective $f(x)$ by **the proximity function**

$$\Delta(x, \tilde{x}) := \sum_{j \in J: \tilde{x}_j = 0} x_j + \sum_{j \in J: \tilde{x}_j = 1} (1 - x_j)$$

Relaxation grip

- Effect of the cutoff constr. for various values of parameter θ on MIPLIB2010 instance **ramos3** (root node relaxation)

x-range	$\theta = 0$	$\theta = 1$	$\theta = 2$	$\theta = 3$	$\theta = 4$	$\theta = 5$	$\theta = 10$	$\theta = 20$	$\theta = 30$	$\theta = 50$	$\theta = 99$	$\theta = 121$
$= 0$	1920	1919	1919	1919	1924	1920	1619	1619	1600	1565	1276	682
$(0.0, 0.1]$	0	0	0	0	0	0	303	297	293	281	420	926
$(0.1, 0.2]$	0	0	0	0	0	4	0	6	26	65	194	380
$(0.2, 0.3]$	0	1	0	5	0	0	0	3	7	15	64	169
$(0.3, 0.4]$	0	0	0	0	0	0	0	1	2	8	75	29
$(0.4, 0.5]$	0	0	6	0	0	0	8	4	3	16	91	0
$(0.5, 0.6]$	0	0	0	0	0	0	5	5	9	19	47	1
$(0.6, 0.7]$	0	0	0	0	0	0	0	2	9	35	17	0
$(0.7, 0.8]$	0	5	0	1	0	1	0	10	25	88	3	0
$(0.8, 0.9]$	0	0	0	0	0	11	0	28	101	68	0	0
$(0.9, 1.0)$	0	0	0	0	0	0	249	209	110	26	0	0
$= 1$	267	262	262	262	263	251	3	3	2	1	0	0
time (sec.s)	0.00	0.04	0.03	0.03	0.04	0.21	0.45	0.54	0.57	0.90	4.77	30.91
# LP-iter.s	0	352	341	357	358	1180	2164	2543	2637	3627	6829	11508
Δ-distance	0.00	1.50	3.00	4.50	6.00	7.88	17.45	37.13	56.86	96.90	208.71	292.67

The basic #Proxy heuristic

`Proximity Search`:

1. let $\tilde{x}$ be the initial heuristic feasible solution to refine;
   repeat
2. explicitly add the *cutoff constraint* $f(x) \leq f(\tilde{x}) - \theta$ to the MIP model;
3. replace $f(x)$ by the "proximity" objective function $\Delta(x, \tilde{x})$;
4. run the MIP solver on the new model until a termination condition is reached, and let x^* be the best feasible solution found (x^* empty if none);
   if x^* is nonempty and $J \subset N$ **then**
5. refine x^* by solving the convex program
   $$x^* := \text{argmin}\{f(x) : g(x) \leq 0,\ x_j = x_j^*\ \forall j \in J\}$$
   end
6. recenter $\Delta(x, \cdot)$ by setting $\tilde{x} := x^*$, and/or update θ
   until *an overall termination condition is reached*;

Proximity search with incumbent

- Basic Proximity Search works **without an incumbent** (as soon a better integer sol. is found, we cut it off) → powerful internal tools of the black-box solver (including RINS) are never activated

- Easy workaround: **soft** cutoff constraint (slack *z* with BIGM penalty)

$$\begin{aligned} \textit{min} \quad & \Delta(x, \tilde{x}) + M z \\ & f(x) \leq f(\tilde{x}) - \theta + z \\ & \dots \end{aligned}$$

- Warm-started the subMIP with the (high-cost but) feasible integer sol. $\widetilde{x}$

Stochastic Programming 0-1 MILP's

- A 0-1 MILP with a decomposition structure (**Stochastic Programming**)

$$
\begin{array}{lllll}
\min c_0^T y & +\sum_{k=1}^{K} \mu_k & & & \\
 & c_k^T x_k & = \mu_k & & \\
A_0 y & +A_k x_k & \geq b_k & k = 1, \ldots, K & \\
y & & \in Y & & \\
 & x_k & \in P_k & k = 1, \ldots, K &
\end{array}
$$

where $Y = P_0 \cap \{0,1\}^m$ and each set P_k $(k = 0, \ldots, K)$ is a polyhedron.

- For fixed *y* and μ_K's, each x_K can be determined by solving an individual MILP → **Benders' decomposition** (master on *y* and μ_K, *K* slaves on x_K)

Computational conjecture

Known drawbacks of Benders decomposition

- Benders can be **slow** as the master solution has little incentive to be feasible ← lot of Benders cuts needed to "cure" master's **natural reluctance to become feasible**

- Long sequence of **super-optimal** infeasible sol.s provided by the master, feasibility eventually reached at the last iteration #**BadForHeuristics**

Our computational conjecture

Changing the master objective will improve relaxation grip

→ master sol.s almost feasible "by empathy" and not "by cuts"

→ faster convergence of Benders' scheme

→ a natural setting for Proximity Search

#ProximityBenders

- Just apply Proximity Search (PS) **on top** of Benders' decomposition

Given a feasible solution $(\widetilde{y}, \widetilde{x}, \widetilde{\mu})$, add the cutoff constraint

$$c_0^T y + \sum_{k=1}^{K} \mu_k \leq U - \theta$$

where $U = c_0^T \widetilde{y} + \sum_{k=1}^{K} \widetilde{\mu}_k$ and $\theta > 0$ is a given tolerance,
and replace the objective function with the Hamming distance

$$\Delta(y, \widetilde{y}) = \sum_{j:\widetilde{y}_j=0} y_j + \sum_{j:\widetilde{y}_j=1} (1 - y_j)$$

- **Note: Benders as a black-box inside PS, not vice-versa as in**

Rei, W., Cordeau, J.-F., Gendreau, M., and Soriano, P. (2009). Accelerating Benders decomposition by local branching. INFORMS Journal on Computing, 21(2):333–345.

Computational experiments

- Three classes of Stochastic Programs from the literature
  1. Stochastic **Capacitated Facility Location**
  2. Stochastic **Network Interdiction**
  3. Stochastic **Fixed-Charge Multi-Commodity Network Design**

- Extensive **computational results** on benchmark instances from the literature

- **Outcome**:
  - Proximity Benders can be very effective to quickly find high-quality solutions to **very large instances of Stochastic Programs**
  - Very useful when solving the **root node LP relaxation** is already **computationally prohibitive**

Example

Solution value over time for a hard instance from

Bodur, M., Dash, S., Gunluck, O., and Luedtke, J. (2014). Strengthened Benders cuts for stochastic integer programs with continuous recourse. Optimization Online 2014- 03-4263.

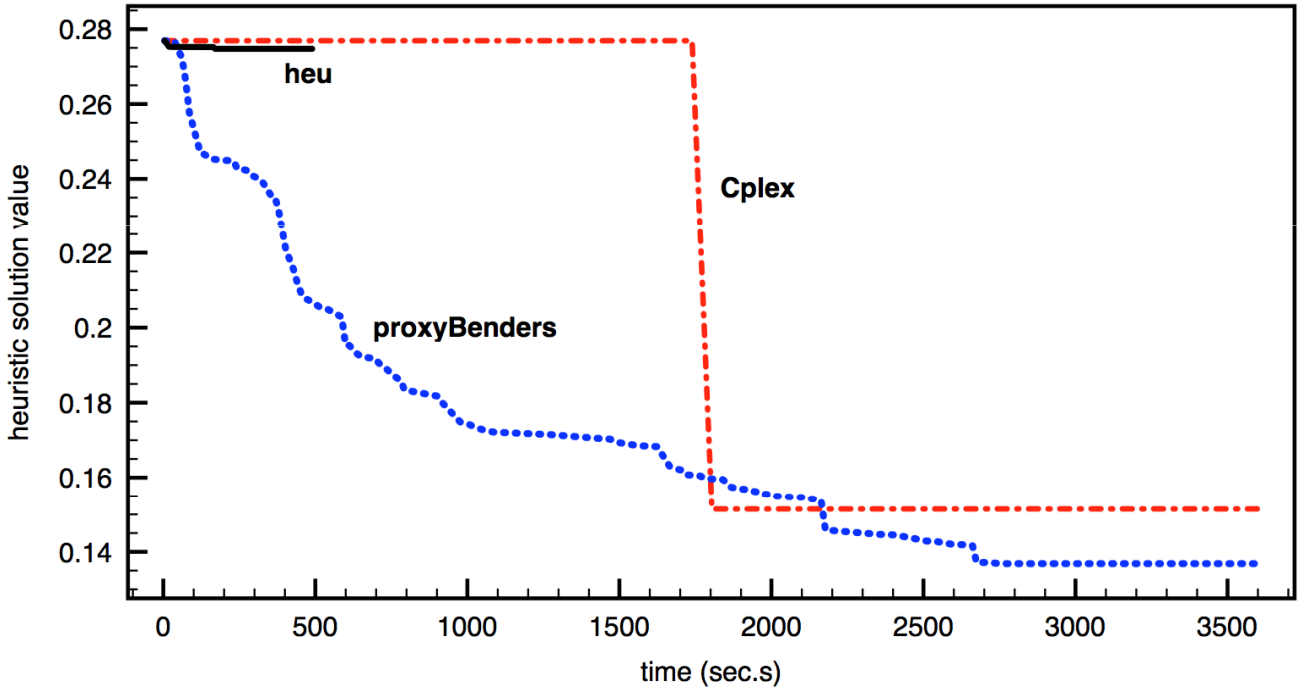
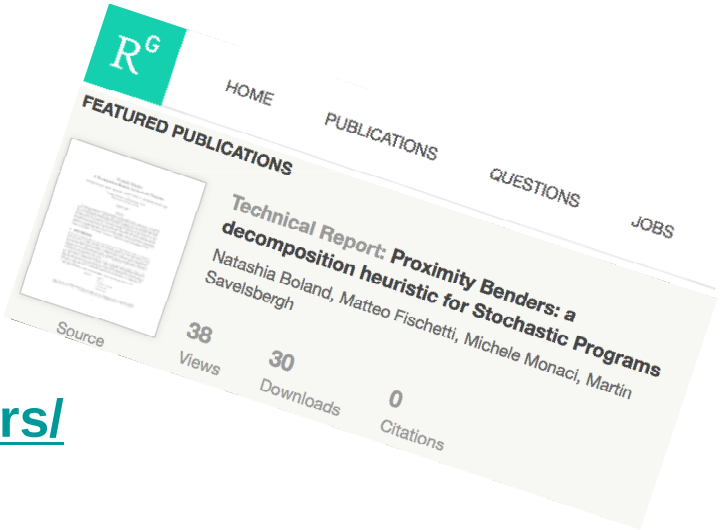


Figure 1: The best known solution value over time for the three heuristics.

Thank you for your attention

- **Full paper**

  N. Boland, M. Fischetti, M. Monaci, M. Savelsbergh, “Proximity Benders: a decomposition heuristic for Stochastic Programs", Technical Report DEI, 2015.



and **slides** available at

http://www.dei.unipd.it/~fisch/papers/

http://www.dei.unipd.it/~fisch/papers/slides/