Using Bandit Algorithms for Adaptive Algorithmic Decisions in SCIP

Gregor Hendel

March 7, 2018, Aachen, Germany

Overview

Large Neighborhood Search in MIP

Adaptive Large Neighborhood Search

Computational Results

Extensions

Using Bandit Algorithms for Adaptive Algorithmic Decisions in SCIP

Large Neighborhood Search in MIP

Mixed Integer Programs

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \geq b \\ & l \leq x \leq u \\ & x \in \{0,1\}^{n_b} \times \mathbb{Z}^{n_i - n_b} \times \mathbb{Q}^{n - n_i} \end{aligned} \tag{MIP}$$

Solution method:

- typically solved with branch-and-cut
- primal heuristics support the solution process

Notation:

- $\mathcal{F}_P$ set of solutions of a MIP P
- x^{inc} incumbent solution, c^{dual} dual bound

LNS and the auxiliary MIP

Auxiliary MIP

Let P be a MIP with solution set $\mathcal{F}_P$. For a polyhedron $\mathcal{N} \subseteq \mathbb{Q}^n$ and objective coefficients $c_{\text{aux}} \in \mathbb{Q}^n$, a MIP P^{aux} defined as

$$\min \left\{ c_{\text{aux}}^T x \,|\, x \in \mathcal{F}_P \cap \mathcal{N} \right\}$$

is called an auxiliary MIP of P, and $\mathcal{N}$ is called neighborhood.

Large Neighborhood Search (LNS) heuristics solve auxiliary MIPs and can be distinguished by their respective neighborhoods.

Typical LNS neighborhoods

Let $\mathcal{M} \subseteq \{1, \ldots, n_i\}$, $x^* \in \mathbb{Q}^n$.

- fixing neighborhood

$$\mathcal{N}^{\text{fix}}(\mathcal{M}, x^*) := \left\{x \in \mathbb{Q}^n \,|\, x_j = x_j^* \;\forall j \in \mathcal{M}\right\}$$

Typical LNS neighborhoods

Let $\mathcal{M} \subseteq \{1, \ldots, n_i\}$, $x^* \in \mathbb{Q}^n$.

- fixing neighborhood

$$\mathcal{N}^{\text{fix}}(\mathcal{M}, x^*) := \left\{x \in \mathbb{Q}^n \,|\, x_j = x_j^* \;\forall j \in \mathcal{M}\right\}$$

- improvement neighborhood

$$\mathcal{N}^{\text{obj}}(\delta, x^{\text{inc}}) := \left\{x \in \mathbb{Q}^n \,|\, c^T x \leq (1 - \delta) \cdot c^T x^{\text{inc}} + \delta \cdot c^{\text{dual}}\right\}$$

Examples of LNS Heuristics

Relaxation Induced Neighborhood Search (RINS) [Danna et al., 2005]

$$\mathcal{N}_{\mathsf{RINS}} := \mathcal{N}^{\mathsf{fix}}\left(\mathcal{M}^{=}\left(\left\{x^{\mathsf{lp}}, x^{\mathsf{inc}}\right\}\right), x^{\mathsf{inc}}\right) \cap \mathcal{N}^{\mathsf{obj}}\left(\delta, x^{\mathsf{inc}}\right).$$

Examples of LNS Heuristics

Relaxation Induced Neighborhood Search (RINS) [Danna et al., 2005]

$$\mathcal{N}_{\text{RINS}} := \mathcal{N}^{\text{fix}}\left(\mathcal{M}^{=}\left(\left\{x^{\text{lp}}, x^{\text{inc}}\right\}\right), x^{\text{inc}}\right) \cap \mathcal{N}^{\text{obj}}\left(\delta, x^{\text{inc}}\right).$$

Local Branching [Fischetti and Lodi, 2003]

$$\mathcal{N}_{\text{LBranch}} := \left\{x \in \mathbb{Q}^{n} \mid \left\|x - x^{\text{inc}}\right\|_{b} \leq d_{\max}\right\} \cap \mathcal{N}^{\text{obj}}(\delta, x^{\text{inc}})$$

Examples of LNS Heuristics

Relaxation Induced Neighborhood Search (RINS) [Danna et al., 2005]

$$\mathcal{N}_{\text{RINS}} := \mathcal{N}^{\text{fix}}\left(\mathcal{M}^{=}\left(\left\{x^{\text{lp}}, x^{\text{inc}}\right\}\right), x^{\text{inc}}\right) \cap \mathcal{N}^{\text{obj}}\left(\delta, x^{\text{inc}}\right).$$

Local Branching [Fischetti and Lodi, 2003]

$$\mathcal{N}_{\text{LBranch}} := \left\{x \in \mathbb{Q}^n \mid \left\|x - x^{\text{inc}}\right\|_b \leq d_{\max}\right\} \cap \mathcal{N}^{\text{obj}}(\delta, x^{\text{inc}})$$

- Crossover, Mutation [Rothberg, 2007]
- RENS [Berthold, 2014]
- Proximity [Fischetti and Monaci, 2014]
- DINS [Ghosh, 2007]
- Zeroobjective [in SCIP, Gurobi, XPress,. . .]
- Analytic Center Search [Berthold et al., 2017]
- . . .

Adaptive Large Neighborhood Search

Adaptive Large Neighborhood Search

- new primal heuristic plugin `heur_alns.c`
- controls 8 LNS heuristics called neighborhoods
- 3 important callbacks

```
/** callback to collect variable fixings of neighborhood */
#define DECL_VARFIXINGS(x) SCIP_RETCODE x ( ... )

/** callback for subproblem changes other than variable fixings
#define DECL_CHANGESUBSCIP(x) SCIP_RETCODE x (   ... )

/** callback function to return a feasible reference solution
 * for further fixings */
#define DECL_NHREFSOL(x) SCIP_RETCODE x ( ... )
```

- neighborhoods are called based on their reward
- further algorithmic steps: generic fixings, adaptive fixing rate
- released with SCIP 5.0

The Multi-Armed Bandit Problem

- Discrete time steps $t = 1, 2, \ldots$
- Finite set of actions $\mathcal{H}$

1. Choose $h_t \in \mathcal{H}$
2. Observe reward $g(h_t, t) \in [0, 1]$
3. Goal: Maximize $\sum_t g(h_t, t)$

The Multi-Armed Bandit Problem

- Discrete time steps $t = 1, 2, \ldots$
- Finite set of actions $\mathcal{H}$

1. Choose $h_t \in \mathcal{H}$
2. Observe reward $g(h_t, t) \in [0, 1]$
3. Goal: Maximize $\sum_t g(h_t, t)$

2 Scenarios:

- stochastic i.i.d. rewards for each action over time

The Multi-Armed Bandit Problem

- Discrete time steps $t = 1, 2, \ldots$
- Finite set of actions $\mathcal{H}$

1. Choose $h_t \in \mathcal{H}$
2. Observe reward $g(h_t, t) \in [0, 1]$
3. Goal: Maximize $\sum_t g(h_t, t)$

2 Scenarios:

- stochastic i.i.d. rewards for each action over time
- adversarial an opponent tries to maximize the player's regret.

Literature: [Bubeck and Cesa-Bianchi, 2012]

Upper Confidence Bound (UCB)

$$h_t \in \begin{cases} \underset{h \in \mathcal{H}}{\operatorname{argmax}} \left\{ \hat{r}_h(t-1) + \sqrt{\frac{\alpha \ln(1+t)}{T_h(t-1)}} \right\} & \text{if } t > |\mathcal{H}|, \\ \{H_t\} & \text{if } t \leq |\mathcal{H}|. \end{cases}$$

Upper Confidence Bound (UCB)

$$h_t \in \begin{cases} \underset{h \in \mathcal{H}}{\operatorname{argmax}} \left\{ \hat{r}_h(t-1) + \sqrt{\frac{\alpha \ln(1+t)}{T_h(t-1)}} \right\} & \text{if } t > |\mathcal{H}|, \\ \{H_t\} & \text{if } t \leq |\mathcal{H}|. \end{cases}$$

ε**-greedy**

Select heuristic at random with probability $\varepsilon_t = \varepsilon\sqrt{\frac{8}{t}}$, otherwise use best.

Upper Confidence Bound (UCB)

$$h_t \in \begin{cases} \operatorname*{argmax}_{h \in \mathcal{H}} \left\{ \hat{r}_h(t-1) + \sqrt{\frac{\alpha \ln(1+t)}{T_h(t-1)}} \right\} & \text{if } t > |\mathcal{H}|, \\ \{H_t\} & \text{if } t \leq |\mathcal{H}|. \end{cases}$$

ε-greedy

Select heuristic at random with probability $\varepsilon_t = \varepsilon \sqrt{\frac{8}{t}}$, otherwise use best.

Exp.3

$$p_{h,t} = (1-\gamma) \cdot \frac{\exp(w_{h,t})}{\sum_{h'} \exp(w_{h',t})} + \gamma \cdot \frac{1}{8}$$

Upper Confidence Bound (UCB)

$$h_t \in \begin{cases} \underset{h \in \mathcal{H}}{\operatorname{argmax}} \left\{ \hat{r}_h(t-1) + \sqrt{\frac{\alpha \ln(1+t)}{T_h(t-1)}} \right\} & \text{if } t > |\mathcal{H}|, \\ \{H_t\} & \text{if } t \leq |\mathcal{H}|. \end{cases}$$

ε-greedy

Select heuristic at random with probability $\varepsilon_t = \varepsilon\sqrt{\frac{8}{t}}$, otherwise use best.

Exp.3

$$p_{h,t} = (1-\gamma) \cdot \frac{\exp(w_{h,t})}{\sum_{h'} \exp(w_{h',t})} + \gamma \cdot \frac{1}{8}$$

Individual parameters $\alpha, \varepsilon, \gamma \geq 0$ must be calibrated.

Bandit Selection in SCIP API

Since SCIP 5.0, all 3 bandit algorithms are available in the public API.

1. Bandit creation

```
SCIP_BANDIT* bandit;

SCIPcreateBanditUcb(scip, &bandit, priorities,
               alpha, nactions, seed);
SCIPcreateBanditExp3(scip, &bandit, priorities,
               gamma, ..., nactions, seed);
SCIPcreateBanditEpsgreedy(scip, &bandit, priorities,
               epsilon, nactions, seed);
```

2. Selection

```
SCIPbanditSelect(bandit, *selection);
```

3. Update

```
SCIPbanditUpdate(bandit, selection, reward);
```

http://scip.zib.de/doc-5.0.1/html/group__PublicBanditMethods.php

Rewarding Neighborhoods

Goal A suitable reward function $r^{\text{alns}}(h_t, t) \in [0, 1]$

Rewarding Neighborhoods

Goal A suitable reward function $r^{\text{alns}}(h_t, t) \in [0, 1]$

Solution Reward

$$r^{\text{sol}}(h_t, t) = \begin{cases} 1 & \text{, if } x^{\text{old}} \neq x^{\text{new}} \\ 0 & \text{, else} \end{cases}$$

Rewarding Neighborhoods

Goal A suitable reward function $r^{\text{alns}}(h_t, t) \in [0, 1]$

Solution Reward

$$r^{\text{sol}}(h_t, t) = \begin{cases} 1 & , \text{ if } x^{\text{old}} \neq x^{\text{new}} \\ 0 & , \text{ else} \end{cases}$$

Gap Reward

$$r^{\text{gap}}(h_t, t) = \frac{c^T x^{\text{old}} - c^T x^{\text{new}}}{c^T x^{\text{old}} - c^{\text{dual}}}$$

Goal A suitable reward function $r^{\text{alns}}(h_t, t) \in [0, 1]$

Solution Reward

$$r^{\text{sol}}(h_t, t) = \begin{cases} 1 & \text{, if } x^{\text{old}} \neq x^{\text{new}} \\ 0 & \text{, else} \end{cases}$$

Gap Reward

$$r^{\text{gap}}(h_t, t) = \frac{c^T x^{\text{old}} - c^T x^{\text{new}}}{c^T x^{\text{old}} - c^{\text{dual}}}$$

Failure Penalty

$$r^{\text{fail}}(h_t, t) = \begin{cases} 1, & \text{if } x^{\text{old}} \neq x^{\text{new}} \\ 1 - \phi(h_t, t)\frac{n(h_t)}{n^{\text{lim}}} & \end{cases}$$

Rewarding Neighborhoods

Goal A suitable reward function $r^{\text{alns}}(h_t, t) \in [0, 1]$

Solution Reward

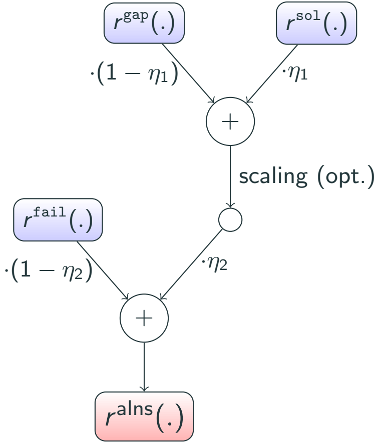
$$r^{\text{sol}}(h_t, t) = \begin{cases} 1 & \text{, if } x^{\text{old}} \neq x^{\text{new}} \\ 0 & \text{, else} \end{cases}$$

Gap Reward

$$r^{\text{gap}}(h_t, t) = \frac{c^T x^{\text{old}} - c^T x^{\text{new}}}{c^T x^{\text{old}} - c^{\text{dual}}}$$

Failure Penalty

$$r^{\text{fail}}(h_t, t) = \begin{cases} 1, & \text{if } x^{\text{old}} \neq x^{\text{new}} \\ 1 - \phi(h_t, t)^{\frac{n(h_t)}{n^{\text{lim}}}} & \end{cases}$$



Default settings in ALNS: $\eta_1 = 0.8$, $\eta_2 = 0.5$

Fixing at a Target Rate

If a neighborhood provides a reference solution x^{ref} (neighborhood callback)

Additional variables are fixed in ascending order based on

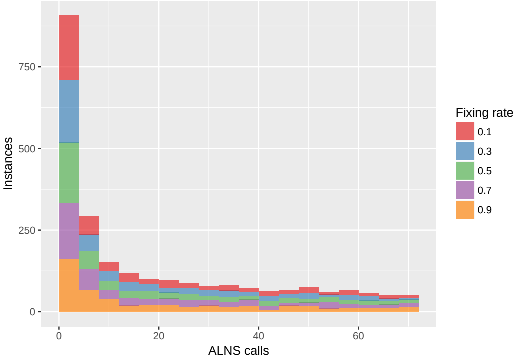
1. Proximity to already fixed variables in the variable constraint graph
2. High root reduced cost score of the fixing
3. High pseudo cost score of the fixing
4. Randomly

A similar logic is applied to unfix variables.

Computational Results

Simulation

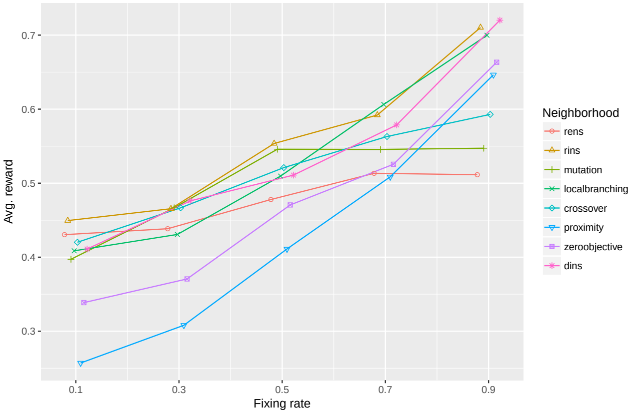
- Always execute all 8 neighborhoods with ALNS (disable old LNS heuristics)
- Disable solution transfer
- Record each reward
- Fixing rates 0.1 – 0.9



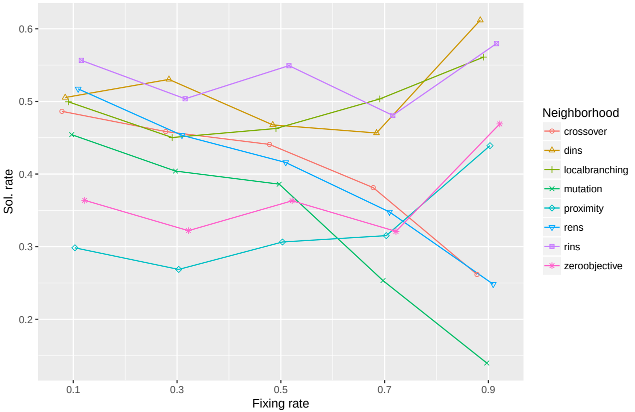
Test Set

665 instances from the test sets MIPLIB3, MIPLIB2003, MIPLIB2010, Cor@l, 5h time limit.

Rewards by Fixing Rate

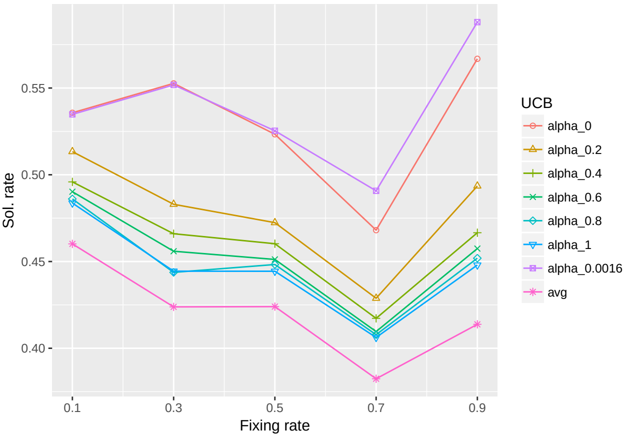


Solution Rate

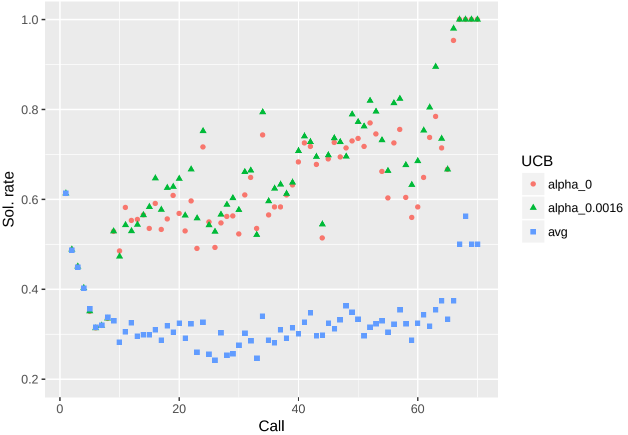


UCB Calibration

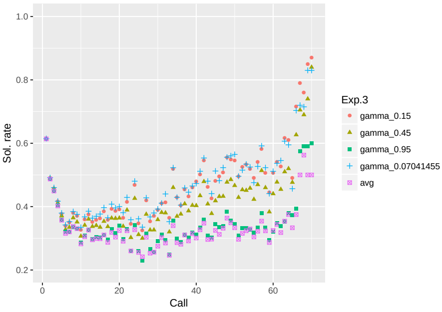
Simulate 100 repetitions of UCB, Exp.3, and ϵ-greedy on the data

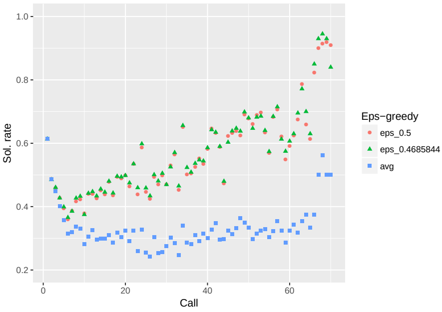


Learning Curve of UCB

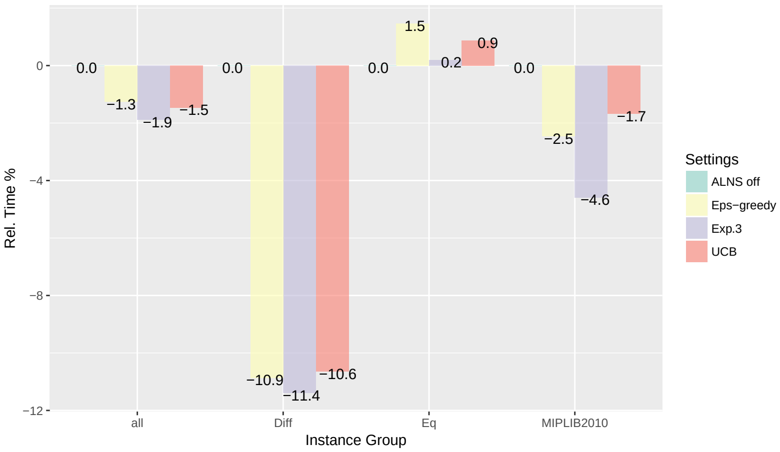


More Learning Curves





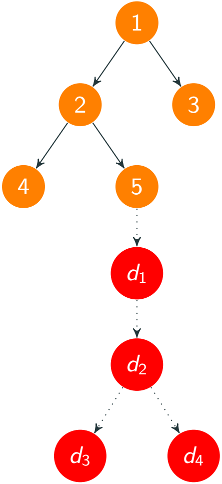
Performance of the ALNS framework



Extensions

Diving Heuristics (joint work with Jakob Witzig)

8 different diving heuristics explore an auxiliary tree in probing mode.



Goal of Selection

Improving solutions and relevant search information

Possible Reward functions

- minimum avg. depth
- minimum backtracks/conflict ratio
- minimum avg. probing nodes
- minimum avg. LP iterations

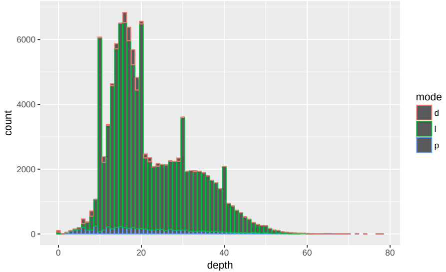
LP Pricing (joint work with Matthias Miltenberger)

Pricings

- Devex
- Steepest Edge
- Quick Start Steepest Edge

Goal of the Selection

Maximize LP throughput



LP counts in diving, probing, and normal lp mode for `timtab1`.

Challenge

Calibration of a deterministic timing approximation across instances.

Conclusion

- ALNS framework to unify existing LNS heuristics as neighborhoods
- Bandit selection algorithms available in SCIP.
- A suitable reward function for LNS heuristics from which the bandits can "learn" even in a short amount of time.
- Started on applications to other selection problems within SCIP.

Next steps

- finish transformation of the classic LNS plugins.
- better communication of presolving/propagation/history information between SCIP and sub-SCIP.

Berthold, T. (2014).
Rens - the optimal rounding.
Mathematical Programming Computation, 6(1):33–54.

Berthold, T., Perregaard, M., and Meszaros, C. (2017).
Four good reasons to use an interior point solver within a mip solver.

Bubeck, S. and Cesa-Bianchi, N. (2012).
Regret analysis of stochastic and nonstochastic multi-armed bandit problems.
CoRR, abs/1204.5721.

Danna, E., Rothberg, E., and Pape, C. L. (2005).
Exploring relaxation induced neighborhoods to improve MIP solutions.
Mathematical Programming, 102(1):71–90.

Fischetti, M. and Lodi, A. (2003).
Local branching.
Mathematical Programming, 98(1-3):23–47.

Fischetti, M. and Monaci, M. (2014).
Proximity search for 0-1 mixed-integer convex programming.
Technical report, DEI - Università di Padova.

Ghosh, S. (2007).
DINS, a MIP Improvement Heuristic.
In Fischetti, M. and Williamson, D. P., editors, *Integer Programming and Combinatorial Optimization: 12th International IPCO Conference, Ithaca, NY, USA, June 25-27, 2007. Proceedings*, pages 310–323, Berlin, Heidelberg. Springer Berlin Heidelberg.

Rothberg, E. (2007).
An Evolutionary Algorithm for Polishing Mixed Integer Programming Solutions.
INFORMS Journal on Computing, 19(4):534–541.