Lagrangian Decomposition for Optimal Cost Partitioning

Florian Pommerening[1] Gabriele Röger[1] Malte Helmert[1]
Hadrien Cambazard[2] Louis-Martin Rousseau[3]
Domenico Salvagnin[4]

[1]University of Basel, Switzerland

[2]Univ. Grenoble Alpes, CNRS, Grenoble INP*, G-SCOP, 38000 Grenoble, France
*Institute of Engineering Univ. Grenoble Alpes

[3]Polytechnique Montreal, Canada

[4]University of Padua, Italy

July 14, 2019

Structure

In this presentation

- context: cost partitioning in classical planning
- Lagrangian decomposition
  - simplified, specialized, ignoring assumptions
  - see paper for details
- relation to cost partitioning
- subgradient optimization
- algorithm to compute optimal cost partitioning
  without an LP solver

Lagrangian Decomposition

Starting with a Linear Program

Problem P

Min $c\,x$ s.t.

$$A_1\,x \geq b_1$$

$$\dots$$

$$A_k\,x \geq b_k$$

$$x \geq 0$$

Starting with a Linear Program

Problem P

$$\text{Min } c\,x \text{ s.t.}$$
$$A_1 x \geq b_1$$
$$\ldots$$
$$A_k x \geq b_k$$
$$x \geq 0$$

$\xrightarrow{\text{rewrite}}$

Problem P

$$\text{Min } c\,x \text{ s.t.}$$
$$A_i x_i \geq b_i \quad \forall i$$
$$x = x_i \quad \forall i$$
$$x, x_i \geq 0 \quad \forall i$$

Lagrangian Relaxation

Problem P

Min c x s.t.

$$A_i x_i \geq b_i \quad \forall i$$

$$x = x_i \quad \forall i$$

$$x, x_i \geq 0 \quad \forall i$$

Lagrangian Relaxation

Problem P

Min $c \; x$ s.t.

$$
\begin{aligned}
A_i \; x_i &\geq b_i && \forall i \\
x &= x_i && \forall i \\
x, \; x_i &\geq 0 && \forall i
\end{aligned}
$$

$\xrightarrow{\text{relax}}$

Problem $P(\lambda)$

Min $c \; x + \sum_i \lambda_i (x_i - x)$ s.t.

$$
\begin{aligned}
A_i \; x_i &\geq b_i && \forall i \\
x, \; x_i &\geq 0 && \forall i
\end{aligned}
$$

- Penalty term λ_i for violating $x = x_i$ called Lagrangian multiplier

Lagrangian Relaxation

Problem P

Min $c\, x$ s.t.

$$A_i\, x_i \geq b_i \quad \forall i$$

$$x = x_i \quad \forall i$$

$$x, x_i \geq 0 \quad \forall i$$

$\xrightarrow{\text{relax}}$

Problem $P(\lambda)$

Min $c\, x + \sum_i \lambda_i (x_i - x)$ s.t.

$$A_i\, x_i \geq b_i \quad \forall i$$

$$x, x_i \geq 0 \quad \forall i$$

- Penalty term λ_i for violating $x = x_i$ called Lagrangian multiplier
- for every choice of λ: value$(P(\lambda)) \leq$ value(P)
- Lagrangian dual problem: find λ that gives best lower bound
- best lower bound is perfect here: Max$_\lambda P(\lambda) =$ value(P)

Lagrangian Decomposition

Problem $P(\lambda)$

$$\text{Min } c\,x + \sum_i \lambda_i\,(x_i - x) \text{ s.t.}$$
$$A_i\,x_i \geq b_i \qquad \forall i$$
$$x,\ x_i \geq 0 \qquad \forall i$$

$P(\lambda)$ decomposes into independent subproblems $P(\lambda) = \sum_i P_i(\lambda)$

Problem $P_0(\lambda)$

$$\text{Min } \left(c - \sum_i \lambda_i\right) x \text{ s.t.}$$
$$x \geq 0$$

Problem $P_i(\lambda)$

$$\text{Min } \lambda_i\,x_i \text{ s.t.}$$
$$A_i\,x_i \geq b_i$$
$$x_i \geq 0$$

A closer look at $P_0(\lambda)$

Problem $P_0(\lambda)$

$$\text{Min } \left(c - \textstyle\sum_i \lambda_i \right) x \text{ s.t.}$$

$$x \geq 0$$

- if all objective coefficients non-negative: value$(P_0(\lambda)) = 0$
- otherwise $P_0(\lambda)$ is unbounded

Constraint encoded by $P_0(\lambda)$

$$\sum_i \lambda_i \leq c$$

Relation to Cost Partitioning

Summarizing Lagrangian Decomposition

Original Problem P

$$\text{Min } c\, x \text{ s.t.}$$
$$A_i\, x \geq b_i \quad \forall i$$
$$x \geq 0$$

Lagrangian Dual Problem

$$\text{Max } \textstyle\sum_i P_i(\lambda) \text{ s.t.}$$
$$\sum_i \lambda_i \leq c$$

Subproblem $P_i(\lambda)$

$$\text{Min } \lambda_i\, x_i \text{ s.t.}$$
$$A_i\, x_i \geq b_i$$
$$x_i \geq 0$$

Cost Partitioning of Operator-Counting Heuristics

Heuristic h

$$\text{Min } \mathit{cost}\, x \text{ s.t.}$$
$$A_i x \geq b_i \quad \forall i$$
$$x \geq 0$$

Optimal Cost Partitioning

$$\text{Max } \sum_i h_i(\mathit{cost}_i) \text{ s.t.}$$
$$\sum_i \mathit{cost}_i \leq \mathit{cost}$$

Heuristic $h_i(\mathit{cost}_i)$

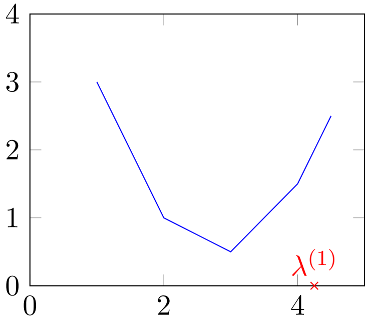
$$\text{Min } \mathit{cost}_i\, x_i \text{ s.t.}$$
$$A_i x_i \geq b_i$$
$$x_i \geq 0$$

How to Solve the Lagrangian Dual Problem

- Computing an optimal cost partitioning corresponds to solving the Lagrangian dual
- . . . but how can we solve it?
- $P(\lambda)$ is concave and we want to maximize it
  $\rightsquigarrow$ can use subgradient optimization

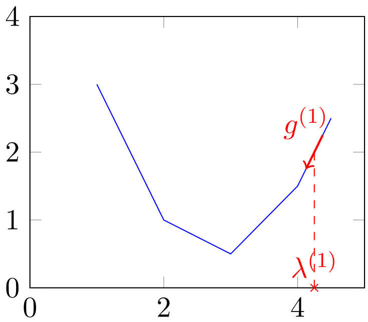
Subgradient Optimization

Subgradient Optimization



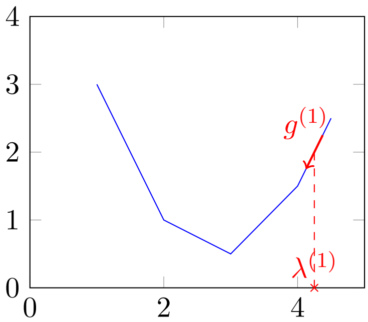
- choose point $\lambda^{(1)}$

Subgradient Optimization



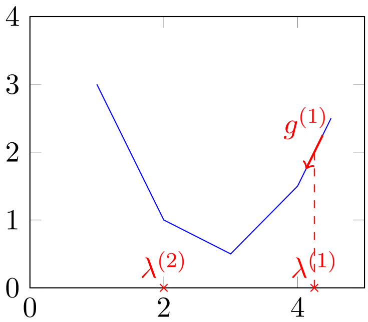
- choose point $\lambda^{(1)}$
- repeat for $t = 1, 2 \ldots$
  - find subgradient $g^{(t)}$ at $\lambda^{(t)}$

Subgradient Optimization



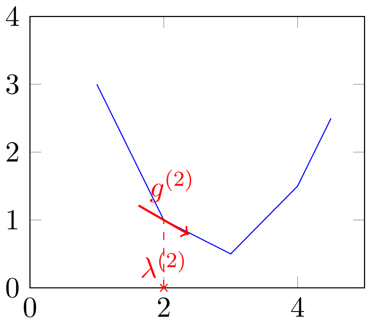
- choose point $\lambda^{(1)}$
- repeat for $t = 1, 2 \ldots$
  - find subgradient $g^{(t)}$ at $\lambda^{(t)}$
  - compute step length $\eta(t)$

Subgradient Optimization



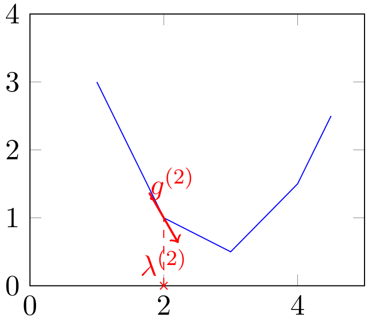
- choose point $\lambda^{(1)}$
- repeat for $t = 1, 2 \ldots$
  - find subgradient $g^{(t)}$ at $\lambda^{(t)}$
  - compute step length $\eta(t)$
  - set $\lambda^{(t+1)} = \lambda^{(t)} + \eta(t) g^{(t)}$

Subgradient Optimization



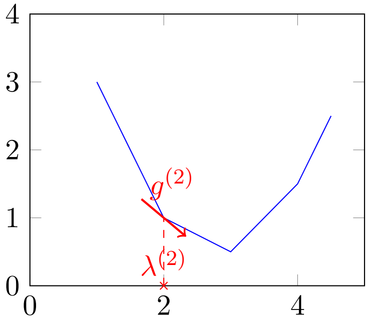
- choose point $\lambda^{(1)}$
- repeat for $t = 1, 2 \dots$
  - find subgradient $g^{(t)}$ at $\lambda^{(t)}$
  - compute step length $\eta(t)$
  - set $\lambda^{(t+1)} = \lambda^{(t)} + \eta(t) g^{(t)}$

Subgradient Optimization



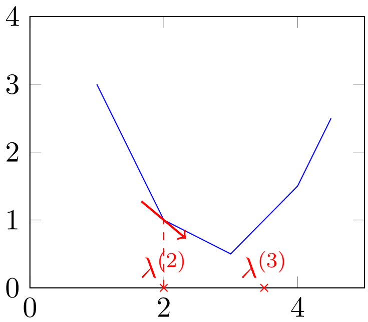
- choose point $\lambda^{(1)}$
- repeat for $t = 1, 2 \ldots$
  - find subgradient $g^{(t)}$ at $\lambda^{(t)}$
  - compute step length $\eta(t)$
  - set $\lambda^{(t+1)} = \lambda^{(t)} + \eta(t)g^{(t)}$

Subgradient Optimization



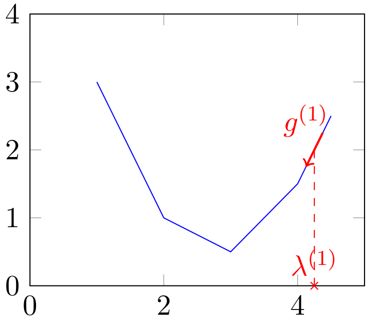
- choose point $\lambda^{(1)}$
- repeat for $t = 1, 2 \ldots$
  - find subgradient $g^{(t)}$ at $\lambda^{(t)}$
  - compute step length $\eta(t)$
  - set $\lambda^{(t+1)} = \lambda^{(t)} + \eta(t)g^{(t)}$

Subgradient Optimization



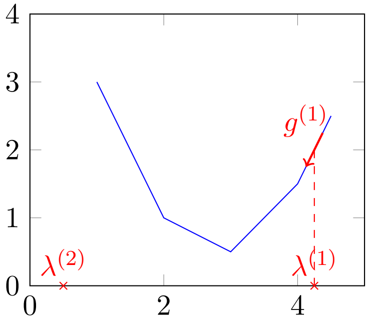
- choose point $\lambda^{(1)}$
- repeat for $t = 1, 2 \ldots$
  - find subgradient $g^{(t)}$ at $\lambda^{(t)}$
  - compute step length $\eta(t)$
  - set $\lambda^{(t+1)} = \lambda^{(t)} + \eta(t) g^{(t)}$

Projected Subgradient Optimization



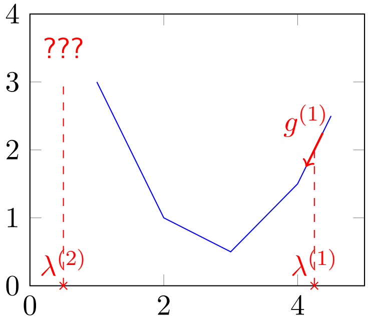
- choose point $\lambda^{(1)}$
- repeat for $t = 1, 2 \ldots$
  - find subgradient $g^{(t)}$ at $\lambda^{(t)}$
  - compute step length $\eta(t)$
  - set $\lambda^{(t+1)} = \lambda^{(t)} + \eta(t) g^{(t)}$

Projected Subgradient Optimization



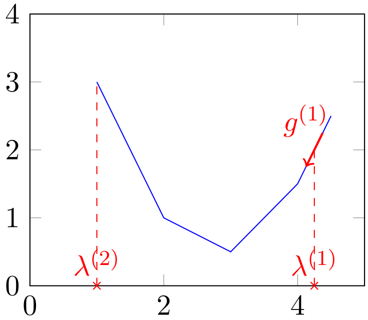
- choose point $\lambda^{(1)}$
- repeat for $t = 1, 2 \ldots$
  - find subgradient $g^{(t)}$ at $\lambda^{(t)}$
  - compute step length $\eta(t)$
  - set $\lambda^{(t+1)} = \lambda^{(t)} + \eta(t) g^{(t)}$

Projected Subgradient Optimization



- choose point $\lambda^{(1)}$
- repeat for $t = 1, 2 \ldots$
  - find subgradient $g^{(t)}$ at $\lambda^{(t)}$
  - compute step length $\eta(t)$
  - set $\lambda^{(t+1)} = \lambda^{(t)} + \eta(t)g^{(t)}$

Projected Subgradient Optimization



- choose point $\lambda^{(1)}$
- repeat for $t = 1, 2 \ldots$
  - find subgradient $g^{(t)}$ at $\lambda^{(t)}$
  - compute step length $\eta(t)$
  - set $\lambda^{(t+1)} = \text{proj}((\lambda^{(t)} + \eta(t) g^{(t)})$

Application to Cost Partitioning over Abstractions

Subgradient Optimization for Cost Partitioning

Analogies in cost partitioning

- current point $\lambda^{(t)}$
- subgradient $g^{(t)}$
- projection

Subgradient Optimization for Cost Partitioning

Analogies in cost partitioning

- current point $\lambda^{(t)}$
  - current cost functions $\mathit{cost}_1, \ldots, \mathit{cost}_k$
- subgradient $g^{(t)}$
- projection

Subgradient Optimization for Cost Partitioning

Analogies in cost partitioning

- current point $\lambda^{(t)}$
  - current cost functions $\mathit{cost}_1, \dots, \mathit{cost}_k$
- subgradient $g^{(t)}$
  - optimal solutions of subproblems $P_i(\lambda^{(t)})$
  - if subproblems are abstraction heuristics: shortest paths in abstractions
- projection

Subgradient Optimization for Cost Partitioning

Analogies in cost partitioning

- current point $\lambda^{(t)}$
  - current cost functions $\mathit{cost}_1, \ldots, \mathit{cost}_k$
- subgradient $g^{(t)}$
  - optimal solutions of subproblems $P_i(\lambda^{(t)})$
  - if subproblems are abstraction heuristics: shortest paths in abstractions
- projection
  - project arbitrary set of cost functions to cost partitioning

Subgradient Optimization for Cost Partitioning

Anytime algorithm

- choose any cost partitioning $\mathit{cost}^{(1)}$
- repeat for $t = 1, 2 \ldots$
  - for each abstraction i
    - find optimal solution π^* under $\mathit{cost}_i^{(t)}$
    - set $\mathit{cost}_i^{(t+1)}(o) = \mathit{cost}_i^{(t)}(o) + \eta(t)\mathit{occurrences}(o, \pi^*)$
  - project $\mathit{cost}^{(t+1)}$ to a cost partitioning

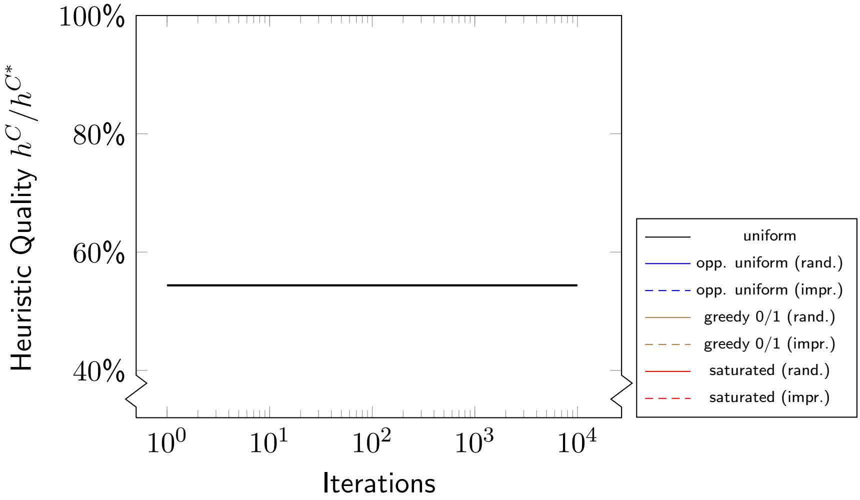
Experiments

Experiment Setup

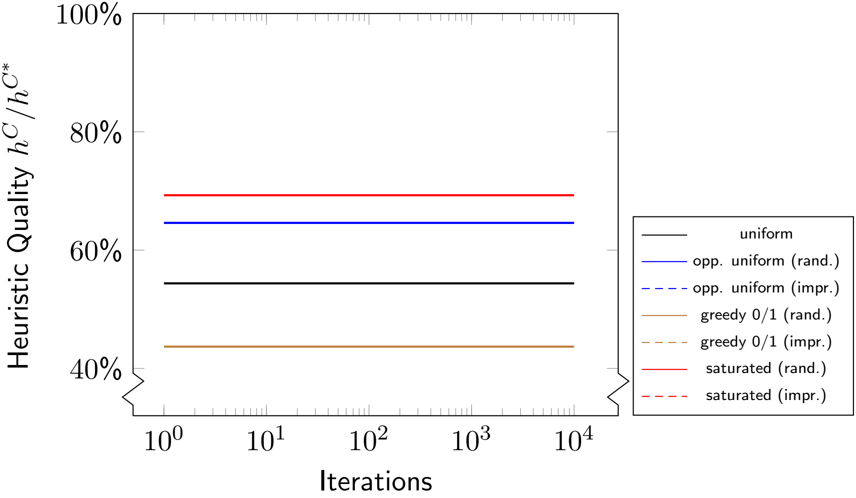
Experiment setup

- IPC instances from optimal tracks (1998–2018)
- projections to all interesting patterns up to size 2 (and 3)
- non-negative cost partitioning
  - no good way to project to general cost partitioning
- 300 s time limit, 2 GB memory limit
- heuristic values of initial states
- seeded with different cost partitioning methods
  - uniform
  - opportunistic uniform (random/improved order)
  - greedy zero-one (random/improved order)
  - saturated (random/improved order)

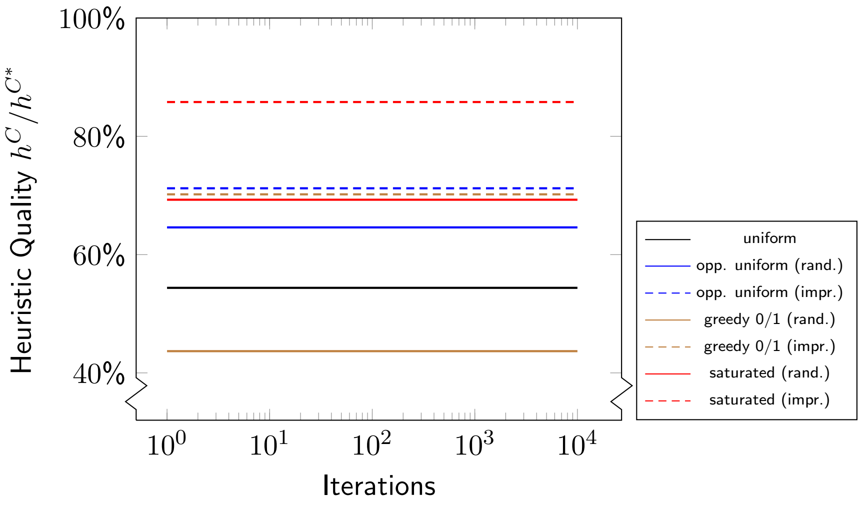
Heuristic Quality



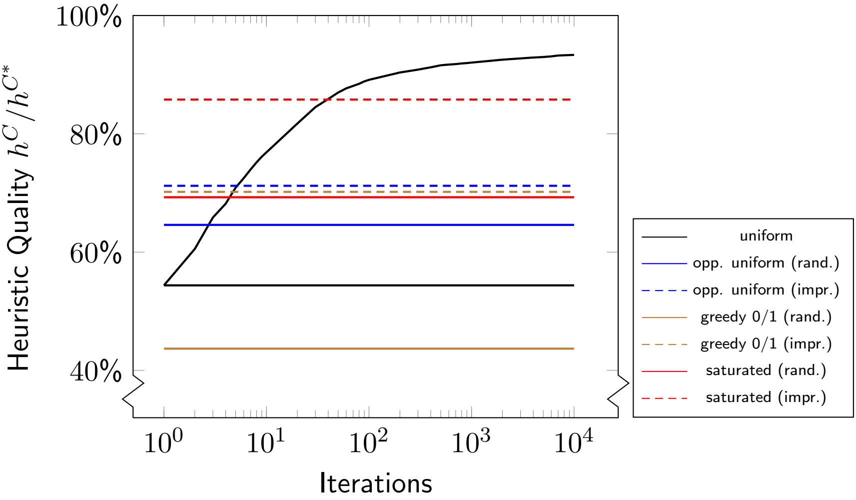
Heuristic Quality



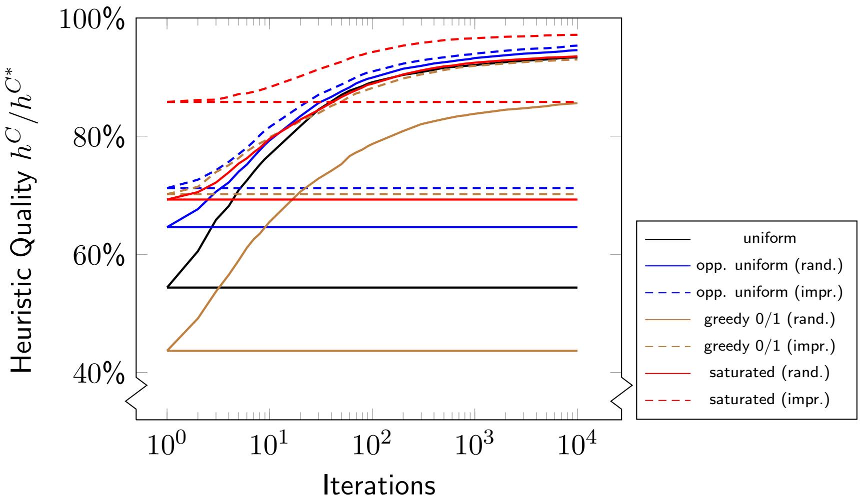
Heuristic Quality



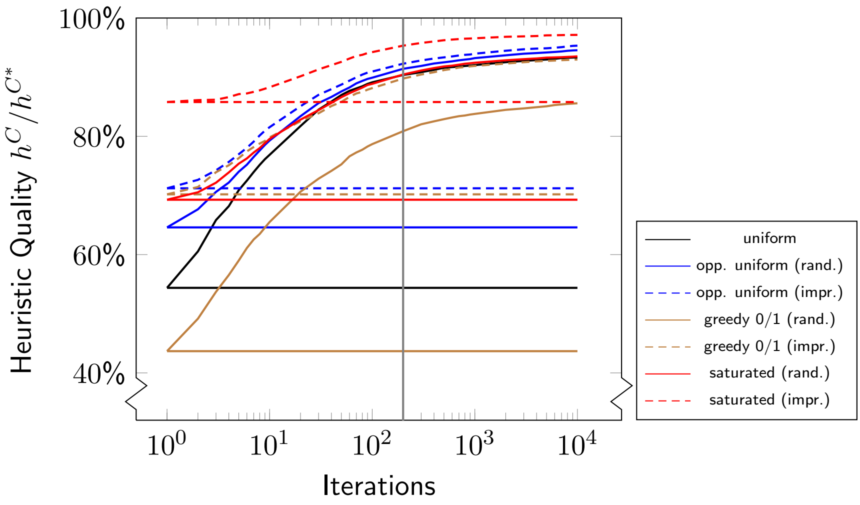
Heuristic Quality



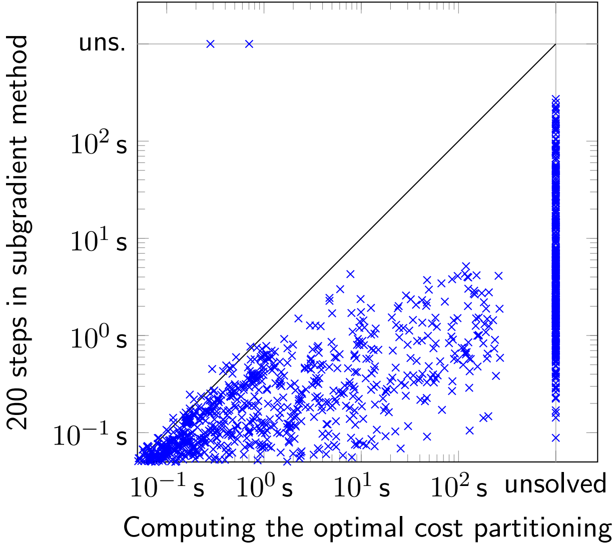
Heuristic Quality



Heuristic Quality



Runtime


uns.
10^2 s
10^1 s
10^0 s
10^{-1} s
200 steps in subgradient method
10^{-1} s
10^0 s
10^1 s
10^2 s
unsolved
Computing the optimal cost partitioning

Conclusion

Conclusion

Contributions to Cost Partitioning

- new interpretation as Lagrangian decomposition
- interesting relation to subgradient optimization
- anytime algorithm for suboptimal cost partitioning

Future Work

- techniques from subgradient optimization
  - better stopping conditions
  - dynamic step length functions
  - improved updates
- open questions
  - projection for general cost partitioning
  - consider highly different operator costs