Automated Configuration of MIP solvers

Frank Hutter, Holger Hoos, and Kevin Leyton-Brown

Department of Computer Science
University of British Columbia
Vancouver, Canada
{hutter,hoos,kevinlb}@cs.ubc.ca

CPAIOR 2010, June 16

Parameters in Algorithms

Most algorithms have parameters

- Decisions that are left open during algorithm design
  - numerical parameters (*e.g.*, real-valued thresholds)
  - categorical parameters (*e.g.*, which heuristic to use)
- Set to optimize empirical performance

Parameters in Algorithms

Most algorithms have parameters

- Decisions that are left open during algorithm design
  - numerical parameters (*e.g.*, real-valued thresholds)
  - categorical parameters (*e.g.*, which heuristic to use)
- Set to optimize empirical performance

Prominent parameters in MIP solvers

- Preprocessing
- Which type of cuts to apply
- MIP strategy parameters
- Details of underlying linear (or quadratic) programming solver

Example: IBM ILOG CPLEX

- 76 parameters that affect search trajectory

Example: IBM ILOG CPLEX

- 76 parameters that affect search trajectory

  "Integer programming problems are more sensitive to specific parameter settings, so **you may need to experiment with them**." [Cplex 12.1 user manual, page 235]

Example: IBM ILOG CPLEX

- 76 parameters that affect search trajectory

  "Integer programming problems are more sensitive to specific parameter settings, so **you may need to experiment with them**." [CPLEX 12.1 user manual, page 235]

- "Experiment with them"
  - Perform manual optimization in 76-dimensional space
  - Complex, unintuitive interactions between parameters

Example: IBM ILOG CPLEX

- 76 parameters that affect search trajectory

  "Integer programming problems are more sensitive to specific parameter settings, so **you may need to experiment with them**." [Cplex 12.1 user manual, page 235]

- "Experiment with them"
  - Perform manual optimization in 76-dimensional space
  - Complex, unintuitive interactions between parameters
  - **Humans are not good at that**

Example: IBM ILOG CPLEX

- 76 parameters that affect search trajectory

  "Integer programming problems are more sensitive to specific parameter settings, so **you may need to experiment with them**." [Cplex 12.1 user manual, page 235]

- "Experiment with them"
  - Perform manual optimization in 76-dimensional space
  - Complex, unintuitive interactions between parameters
  - **Humans are not good at that**

- Cplex automated tuning tool (since version 11)
  - Saves valuable human time
  - Improves performance

Our work: automated algorithm configuration

- Given:
  - Runnable algorithm A, its parameters and their domains
  - Benchmark set of instances Π
  - Performance metric m

Our work: automated algorithm configuration

- Given:
  - Runnable algorithm A, its parameters and their domains
  - Benchmark set of instances Π
  - Performance metric m
- Find:
  - Parameter setting ("configuration") of A optimizing m on Π

Our work: automated algorithm configuration

- Given:
  - Runnable algorithm A, its parameters and their domains
  - Benchmark set of instances Π
  - Performance metric m
- Find:
  - Parameter setting ("configuration") of A optimizing m on Π
- First to handle this with **many categorical parameters**
  - E.g. 51/76 CPLEX parameters are categorical
  - 10^{47} possible configurations $\rightsquigarrow$ **algorithm configuration**

Our work: automated algorithm configuration

- Given:
  - Runnable algorithm A, its parameters and their domains
  - Benchmark set of instances Π
  - Performance metric m
- Find:
  - Parameter setting ("configuration") of A optimizing m on Π
- First to handle this with **many categorical parameters**
  - E.g. 51/76 CPLEX parameters are categorical
  - 10^{47} possible configurations $\rightsquigarrow$ **algorithm configuration**

This paper: application study for MIP solvers

- Use existing algorithm configuration tool (PARAMILS)
- Use different MIP solvers (CPLEX, GUROBI, LPSOLVE)
- Use six different MIP benchmark sets
- Optimize different objectives (runtime to optimality/MIP gap)

Outline

1. Related work
2. Details about this study
3. Results
4. Conclusions

Outline

1. Related work
2. Details about this study
3. Results
4. Conclusions

Parameter Optimization Tools and Applications

- COMPOSER [Gratch & Dejong, '92; Gratch and Chien, '96]
  - Spacecraft communication scheduling
- CALIBRA [Diaz and Laguna, '06]
  - Optimized various metaheuristics
- F-RACE [Birattari et al., '04-present]
  - Iterated Local Search and Ant Colony Optimization
- PARAMILS [Hutter et al, '07-present]
  - SAT (tree & local search), time-tabling, protein folding, ...

Parameter Optimization Tools and Applications

- COMPOSER [Gratch & Dejong, '92; Gratch and Chien, '96]
  - Spacecraft communication scheduling
- CALIBRA [Diaz and Laguna, '06]
  - Optimized various metaheuristics
- F-RACE [Birattari et al., '04-present]
  - Iterated Local Search and Ant Colony Optimization
- PARAMILS [Hutter et al, '07-present]
  - SAT (tree & local search), time-tabling, protein folding, ...
- STOP [Baz, Hunsaker, Brooks & Gosavi, '07 (Tech report)]
  [Baz, Hunsaker & Prokopyev, Comput Optim Appl, '09]
  - Optimized MIP solvers, including CPLEX
  - We only found this work $\approx$ 1 month ago

Parameter Optimization Tools and Applications

- COMPOSER [Gratch & Dejong, '92; Gratch and Chien, '96]
  - Spacecraft communication scheduling
- CALIBRA [Diaz and Laguna, '06]
  - Optimized various metaheuristics
- F-RACE [Birattari et al., '04-present]
  - Iterated Local Search and Ant Colony Optimization
- PARAMILS [Hutter et al, '07-present]
  - SAT (tree & local search), time-tabling, protein folding, ...
- STOP [Baz, Hunsaker, Brooks & Gosavi, '07 (Tech report)]
  [Baz, Hunsaker & Prokopyev, Comput Optim Appl, '09]
  - Optimized MIP solvers, including CPLEX
  - We only found this work $\approx$ 1 month ago
  - Main problem: only optimized performance for single instances
  - Only used small subset of 10 CPLEX parameters

Outline

1. Related work

2. Details about this study
   - The automated configuration tool: ParamILS
   - The MIP solvers: Cplex, Gurobi & lpsolve
   - Experimental Setup

3. Results

4. Conclusions

Outline

1. Related work

2. Details about this study
   - The automated configuration tool: ParamILS
   - The MIP solvers: Cplex, Gurobi & lpsolve
   - Experimental Setup

3. Results

4. Conclusions

Simple manual approach for configuration

Start with some parameter configuration

Simple manual approach for configuration

Start with some parameter configuration

Modify a single parameter

Simple manual approach for configuration

Start with some parameter configuration

Modify a single parameter
if *results on benchmark set improve* **then**
⌊ *keep new configuration*

Simple manual approach for configuration

```
Start with some parameter configuration
repeat
    Modify a single parameter
    if results on benchmark set improve then
        keep new configuration
until no more improvement possible (or "good enough")
```

Simple manual approach for configuration

Start with some parameter configuration
repeat
   Modify a single parameter
   if *results on benchmark set improve* **then**
      keep new configuration
until *no more improvement possible (or "good enough")*

⇝ Manually-executed **local search**

Simple manual approach for configuration

Start with some parameter configuration
repeat
  Modify a single parameter
  if *results on benchmark set improve* **then**
    keep new configuration
until *no more improvement possible (or "good enough")*

⇝ Manually-executed **local search**

ParamILS [Hutter et al., AAAI'07 & '09]:
Iterated local search: biased random walk over local optima

Instantiations of ParamILS Framework

How to evaluate each configuration?

- BasicILS(N): perform fixed number of N runs to evaluate a configuration θ
  - Variance reduction: use same N instances & seeds for each $\boldsymbol{\theta}$

Instantiations of ParamILS Framework

How to evaluate each configuration?

- BasicILS(N): perform fixed number of N runs to evaluate a configuration θ
  - Variance reduction: use same N instances & seeds for each $\boldsymbol{\theta}$
- FocusedILS: choose $N(\boldsymbol{\theta})$ adaptively
  - small $N(\boldsymbol{\theta})$ for poor configurations $\boldsymbol{\theta}$
  - large $N(\boldsymbol{\theta})$ only for good $\boldsymbol{\theta}$

Instantiations of ParamILS Framework

How to evaluate each configuration?

- BasicILS(N): perform fixed number of N runs to evaluate a configuration θ
  - Variance reduction: use same N instances & seeds for each $\boldsymbol{\theta}$
- FocusedILS: choose $N(\boldsymbol{\theta})$ adaptively
  - small $N(\boldsymbol{\theta})$ for poor configurations $\boldsymbol{\theta}$
  - large $N(\boldsymbol{\theta})$ only for good $\boldsymbol{\theta}$
  - typically outperforms BasicILS
  - used in this study

Adaptive Choice of Cutoff Time

- Evaluation of poor configurations takes especially long

Adaptive Choice of Cutoff Time

- Evaluation of poor configurations takes especially long
- Can terminate evaluations early
  - Incumbent solution provides bound
  - Can stop evaluation once bound is reached

Adaptive Choice of Cutoff Time

- Evaluation of poor configurations takes especially long
- Can terminate evaluations early
  - Incumbent solution provides bound
  - Can stop evaluation once bound is reached
- Results
  - Provably never hurts
  - Sometimes substantial speedups

    [Hutter et al., JAIR'09]

Outline

MIP Solvers & their parameters

- Commercial solvers: CPLEX 12.1 & GUROBI 2.0.1
- Open-source solver: LPSOLVE 5.5

MIP Solvers & their parameters

- Commercial solvers: CPLEX 12.1 & GUROBI 2.0.1
- Open-source solver: LPSOLVE 5.5

Algorithm	Parameter type	# params	# values	Total # configurations
CPLEX	Boolean	6	2	$1.90 \cdot 10^{47}$
	Categorical	45	3–7	
	Integer	18	discretized: 5–7	
	Continuous	7	discretized: 5–8	

MIP Solvers & their parameters

- Commercial solvers: CPLEX 12.1 & GUROBI 2.0.1
- Open-source solver: LPSOLVE 5.5

Algorithm	Parameter type	# params	# values	Total # configurations
CPLEX	Boolean	6	2	$1.90 \cdot 10^{47}$
	Categorical	45	3–7	
	Integer	18	discretized: 5–7	
	Continuous	7	discretized: 5–8	
GUROBI	Boolean	4	2	$3.84 \cdot 10^{14}$
	Categorical	16	3–5	
	Integer	3	discretized: 5	
	Continuous	2	discretized: 5	

MIP Solvers & their parameters

- Commercial solvers: CPLEX 12.1 & GUROBI 2.0.1
- Open-source solver: LPSOLVE 5.5

Algorithm	Parameter type	# params	# values	Total # configurations
CPLEX	Boolean	6	2	$1.90 \cdot 10^{47}$
	Categorical	45	3–7	
	Integer	18	discretized: 5–7	
	Continuous	7	discretized: 5–8	
GUROBI	Boolean	4	2	$3.84 \cdot 10^{14}$
	Categorical	16	3–5	
	Integer	3	discretized: 5	
	Continuous	2	discretized: 5	
LPSOLVE	Boolean	40	2	$1.22 \cdot 10^{15}$
	Categorical	7	3–8	

MIP Solvers & their parameters

- Commercial solvers: CPLEX 12.1 & GUROBI 2.0.1
- Open-source solver: LPSOLVE 5.5

Algorithm	Parameter type	# params	# values	Total # configurations
CPLEX	Boolean	6	2	$1.90 \cdot 10^{47}$
	Categorical	45	3–7	
	Integer	18	discretized: 5–7	
	Continuous	7	discretized: 5–8	
GUROBI	Boolean	4	2	$3.84 \cdot 10^{14}$
	Categorical	16	3–5	
	Integer	3	discretized: 5	
	Continuous	2	discretized: 5	
LPSOLVE	Boolean	40	2	$1.22 \cdot 10^{15}$
	Categorical	7	3–8	

Problems with some parameter configurations

- Segmentation faults & wrong results

MIP Solvers & their parameters

- Commercial solvers: CPLEX 12.1 & GUROBI 2.0.1
- Open-source solver: LPSOLVE 5.5

Algorithm	Parameter type	# params	# values	Total # configurations
CPLEX	Boolean	6	2	$1.90 \cdot 10^{47}$
	Categorical	45	3–7	
	Integer	18	discretized: 5–7	
	Continuous	7	discretized: 5–8	
GUROBI	Boolean	4	2	$3.84 \cdot 10^{14}$
	Categorical	16	3–5	
	Integer	3	discretized: 5	
	Continuous	2	discretized: 5	
LPSOLVE	Boolean	40	2	$1.22 \cdot 10^{15}$
	Categorical	7	3–8	

Problems with some parameter configurations

- Segmentation faults & wrong results
- Detect such runs online, give worst possible score
  - ⇝ Local search avoids problematic parameter configurations

MIP Solvers & their parameters

- Commercial solvers: CPLEX 12.1 & GUROBI 2.0.1
- Open-source solver: LPSOLVE 5.5

Algorithm	Parameter type	# params	# values	Total # configurations
CPLEX	Boolean	6	2	$1.90 \cdot 10^{47}$
	Categorical	45	3–7	
	Integer	18	discretized: 5–7	
	Continuous	7	discretized: 5–8	
GUROBI	Boolean	4	2	$3.84 \cdot 10^{14}$
	Categorical	16	3–5	
	Integer	3	discretized: 5	
	Continuous	2	discretized: 5	
LPSOLVE	Boolean	40	2	$1.22 \cdot 10^{15}$
	Categorical	7	3–8	

Problems with some parameter configurations

- Segmentation faults & wrong results
- Detect such runs online, give worst possible score
  - $\rightsquigarrow$ Local search avoids problematic parameter configurations
- Concise bug reports $\rightsquigarrow$ helped to fix 2 bugs in GUROBI (!)

Outline

Benchmark sets used

Domain	Type	#instances	Citation
Comp. sustainability (SUST)	MILP	2 000	[Gomes et al, '08]
Combinatorial auctions (WDP)	MILP	2 000	[Leyton-Brown et al., '00]
Mixed integer knapsack (MIK)	MILP	120	[Atamtürk, '03]
and 3 more ...			

Benchmark sets used

Domain	Type	#instances	Citation
Comp. sustainability (SUST)	MILP	2 000	[Gomes et al, '08]
Combinatorial auctions (WDP)	MILP	2 000	[Leyton-Brown et al., '00]
Mixed integer knapsack (MIK)	MILP	120	[Atamtürk, '03]
and 3 more ...			

Split benchmarks 50:50 into training and test sets

- Optimized parameters on the training set
- Reported performance on the test set
- Necessary to check for *over-tuning*

Setup of configuration experiments

Perform 10 independent runs of ParamILS

- Select configuration $\hat{\boldsymbol{\theta}}^*$ of run with best *training* performance

Setup of configuration experiments

Perform 10 independent runs of ParamILS

- Select configuration $\hat{\boldsymbol{\theta}}^*$ of run with best *training* performance

Compare test performance of:

- ParamILS's configuration $\hat{\boldsymbol{\theta}}^*$
- Default algorithm settings
- Cplex tuning tool
  - Gurobi and lpsolve: no tuning tool available

Outline

1. Related work

2. Details about this study

3. Results

4. Conclusions

Minimization of Runtime to Optimal Solution

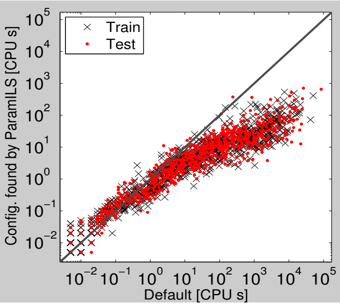
- "Optimal": relative optimality gap of 0.0001
  (Cplex and Gurobi default)

Minimization of Runtime to Optimal Solution

- "Optimal": relative optimality gap of 0.0001 (CPLEX and GUROBI default)
- Ran PARAMILS for 2 days on 10 machines

Minimization of Runtime to Optimal Solution

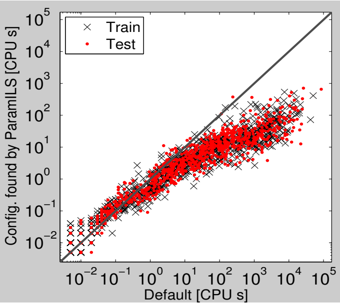
- "Optimal": relative optimality gap of 0.0001 (CPLEX and GUROBI default)
- Ran PARAMILS for 2 days on 10 machines
- Mean speedup (on test instances)
  - CPLEX 2x to 50x



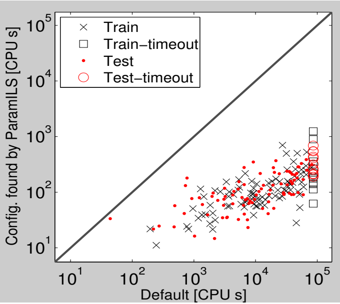
CPLEX on SUST instances (50x)

Minimization of Runtime to Optimal Solution

- "Optimal": relative optimality gap of 0.0001 (CPLEX and GUROBI default)
- Ran PARAMILS for 2 days on 10 machines
- Mean speedup (on test instances)
  - CPLEX 2x to 50x
  - LPSOLVE 1x (no speedup) to 150x



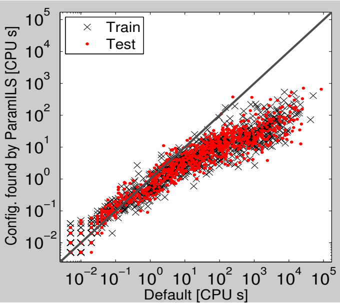
CPLEX on SUST instances (50x)



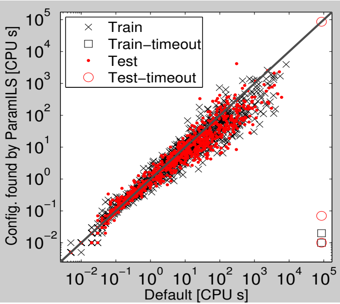
LPSOLVE on WDP instances (150x)

Minimization of Runtime to Optimal Solution

- "Optimal": relative optimality gap of 0.0001 (CPLEX and GUROBI default)
- Ran PARAMILS for 2 days on 10 machines
- Mean speedup (on test instances)
  - CPLEX 2x to 50x
  - LPSOLVE 1x (no speedup) to 150x
  - GUROBI 1.2x to 2.3x



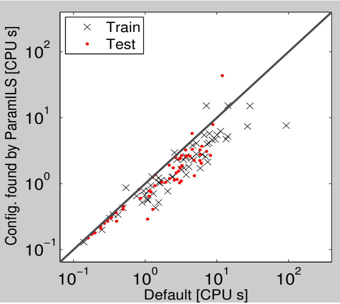
CPLEX on SUST instances (50x)



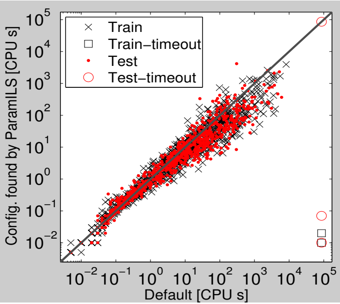
GUROBI on SUST instances (2.3x)

Minimization of Runtime to Optimal Solution

- "Optimal": relative optimality gap of 0.0001 (CPLEX and GUROBI default)
- Ran PARAMILS for 2 days on 10 machines
- Mean speedup (on test instances)
  - CPLEX 2x to 50x
  - LPSOLVE 1x (no speedup) to 150x
  - GUROBI 1.2x to 2.3x



GUROBI on MIK instances (1.2x)



GUROBI on SUST instances (2.3x)

Comparison to Cplex tuning tool

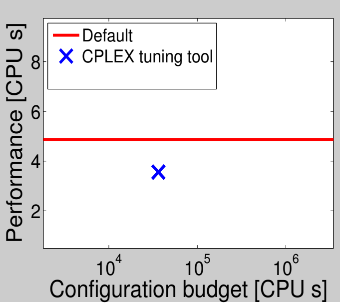
- CPLEX tuning tool
  - Evaluates predefined good configurations, returns best one
  - Required runtime varies (from $< 1h$ to weeks)

Comparison to Cplex tuning tool

- CPLEX tuning tool
  - Evaluates predefined good configurations, returns best one
  - Required runtime varies (from $< 1h$ to weeks)
- PARAMILS: anytime algorithm
  - At each time step, keeps track of its incumbent

Comparison to Cplex tuning tool

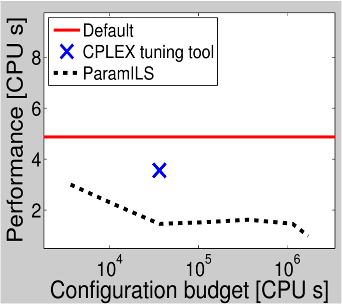
- CPLEX tuning tool
  - Evaluates predefined good configurations, returns best one
  - Required runtime varies (from $< 1h$ to weeks)
- PARAMILS: anytime algorithm
  - At each time step, keeps track of its incumbent



CPLEX on MIK instances

Comparison to Cplex tuning tool

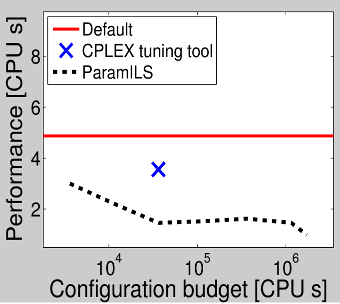
- CPLEX tuning tool
  - Evaluates predefined good configurations, returns best one
  - Required runtime varies (from $< 1h$ to weeks)
- PARAMILS: anytime algorithm
  - At each time step, keeps track of its incumbent



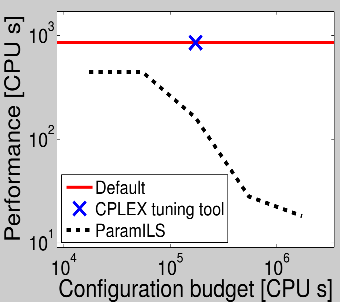
CPLEX on MIK instances

Comparison to Cplex tuning tool

- CPLEX tuning tool
  - Evaluates predefined good configurations, returns best one
  - Required runtime varies (from $< 1h$ to weeks)
- PARAMILS: anytime algorithm
  - At each time step, keeps track of its incumbent



CPLEX on MIK instances



CPLEX on SUST instances

Minimization of Optimality Gap

- Objective: minimal optimality gap within 10 seconds runtime

Minimization of Optimality Gap

- Objective: minimal optimality gap within 10 seconds runtime
- Ran ParamILS for 5 hours on 10 machines

Minimization of Optimality Gap

- Objective: minimal optimality gap within 10 seconds runtime
- Ran ParamILS for 5 hours on 10 machines
- Reduction factors of average optimality gap (on test set)
  - Cplex 1.3x to 8.6x
  - lpsolve 1x (no reduction) to 46x
  - Gurobi 1.1x to 2.2x

Outline

1. Related work
2. Details about this study
3. Results
4. Conclusions

Conclusions

MIP solvers can be configured automatically

- Configuration tool ParamILS available online:
  - http://www.cs.ubc.ca/labs/beta/Projects/ParamILS/
  - off-the-shelf tool (knows nothing about MIP or MIP solvers!)
- Sometimes substantial improvements
- Saves valuable human time

Conclusions

MIP solvers can be configured automatically

- Configuration tool ParamILS available online:
  - `http://www.cs.ubc.ca/labs/beta/Projects/ParamILS/`
  - off-the-shelf tool (knows nothing about MIP or MIP solvers!)
- Sometimes substantial improvements
- Saves valuable human time

Requirements

- Representative instance set
  - 100 instances sometimes not enough
  - If you generate instances, please make more (*e.g.*, 2000)!

Conclusions

MIP solvers can be configured automatically

- Configuration tool ParamILS available online:
  - http://www.cs.ubc.ca/labs/beta/Projects/ParamILS/
  - off-the-shelf tool (knows nothing about MIP or MIP solvers!)
- Sometimes substantial improvements
- Saves valuable human time

Requirements

- Representative instance set
  - 100 instances sometimes not enough
  - If you generate instances, please make more (*e.g.*, 2000)!
- CPU time (here: 10×2 days per domain)

Future Work

- Model-based techniques
  - Fit a model that predicts performance of a given configuration on a given instance

Future Work

- Model-based techniques
  - Fit a model that predicts performance of a given configuration on a given instance
  - Use that model to quantify
    - Importance of each parameter
    - Interaction of parameters
    - Interaction of parameters and instance characteristics

Future Work

- Model-based techniques
  - Fit a model that predicts performance of a given configuration on a given instance
  - Use that model to quantify
    - Importance of each parameter
    - Interaction of parameters
    - Interaction of parameters and instance characteristics
- Per-instance approaches for heterogeneous benchmarks
  - Given a new unseen instance:
    - Compute instance characteristics (fast)
    - Use parameter config. predicted to be best for the instance

Thanks to:

- Providers of instance benchmark sets
  - Louis-Martin Rousseau
  - Bistra Dilkina
  - Berkeley Computational Optimization Lab
- Commercial MIP solvers for free full academic license
  - IBM (CPLEX)
  - GUROBI
- LPSOLVE developers for their solver
- Compute clusters
  - Westgrid
  - CFI-funded arrow cluster
- Funding agencies
  - Postdoc fellowship from CBIE
  - MITACS
  - NSERC

Backup slides

Differences to STOP [Baz et al, '09]

Baz et al optimized for single instances

"In practice, users would typically be tuning for a family of related instances rather than for an individual instance"

- Generalization to *sets* of instances is nontrivial
  - Cannot afford to run all instances for each configuration
  - $\rightsquigarrow$ FocusedILS adapts # runs per configuration

Further differences

- Baz et al used older Cplex version (9.0)
  - defaults improved in newer Cplex versions
- Baz et al considered (only) 10 Cplex parameters
  - and also not all possible values for each parameter
  - in order to improve Stop's performance
  - $\rightsquigarrow$ requires domain knowledge

Configuration of MIP Solvers: Optimality Gap

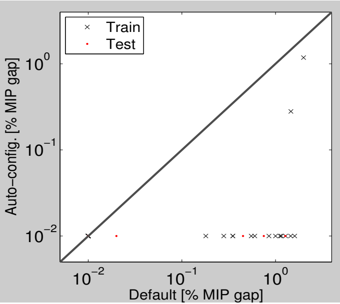
- Objective: minimal optimality gap within 10 seconds runtime

Configuration of MIP Solvers: Optimality Gap

- Objective: minimal optimality gap within 10 seconds runtime
- Ran ParamILS for 5 hours on 10 machines

Configuration of MIP Solvers: Optimality Gap

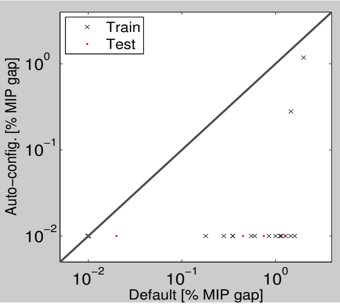
- Objective: minimal optimality gap within 10 seconds runtime
- Ran ParamILS for 5 hours on 10 machines
- Reduction factors of average optimality gap (on test inst.)
  - Cplex 1.3x to 8.6x
  - lpsolve 1x (no reduction) to 46x
  - Gurobi 1.1x to 2.2x



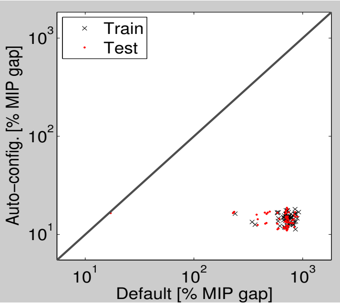
Cplex on MIK instances (8.6x)

Configuration of MIP Solvers: Optimality Gap

- Objective: minimal optimality gap within 10 seconds runtime
- Ran ParamILS for 5 hours on 10 machines
- Reduction factors of average optimality gap (on test inst.)
  - Cplex 1.3x to 8.6x
  - lpsolve 1x (no reduction) to 46x
  - Gurobi 1.1x to 2.2x



Cplex on MIK instances (8.6x)



lpsolve on MIK instances (46x)