Progress in *clasp* series 3

Martin Gebser Roland Kaminski Benjamin Kaufmann
Javier Romero Torsten Schaub

University of Potsdam

Outline

1. Motivation
2. Disjunctive solving
3. Optimization
4. Heuristics
5. Configuration
6. Experiments
7. Summary

Potassco

Outline

1 Motivation
2 Disjunctive solving
3 Optimization
4 Heuristics
5 Configuration
6 Experiments
7 Summary

Potassco

Motivation

- DPLL *smodels*, *dlv*
  SAT *assat*, *cmodels*, *lp2sat*
  CDCL *clasp*, *wasp*

- Objective comprehensive description of *clasp*'s series 3
- Features of *clasp* series 3
  - parallel solving of disjunctive logic programs
  - parallel optimization with orthogonal strategies
  - declarative support for specifying domain heuristics
  - a portfolio of prefabricated expert configurations and
  - an application programming interface for library integration
- Empirical study contrasting them for solving optimization problems

Potassco

Motivation

- DPLL *smodels*, *dlv*
  SAT *assat*, *cmodels*, *lp2sat*
  CDCL *clasp*, *wasp*

- Objective comprehensive description of *clasp*'s series 3

- Features of *clasp* series 3
  - parallel solving of disjunctive logic programs
  - parallel optimization with orthogonal strategies
  - declarative support for specifying domain heuristics
  - a portfolio of prefabricated expert configurations and
  - an application programming interface for library integration

- Empirical study contrasting them for solving optimization problems

Potassco

Motivation

- DPLL *smodels*, *dlv*
  SAT *assat*, *cmodels*, *lp2sat*
  CDCL *clasp*, *wasp*
- Objective comprehensive description of *clasp*'s series 3
- Features of *clasp* series 3
  - parallel solving of disjunctive logic programs
  - parallel optimization with orthogonal strategies
  - declarative support for specifying domain heuristics
  - a portfolio of prefabricated expert configurations and
  - an application programming interface for library integration
- Empirical study contrasting them for solving optimization problems

Potassco

Motivation

- DPLL *smodels*, *dlv*
  SAT *assat*, *cmodels*, *lp2sat*
  CDCL *clasp*, *wasp*
- Objective comprehensive description of *clasp*'s series 3
- Features of *clasp* series 3
  - parallel solving of disjunctive logic programs
  - parallel optimization with orthogonal strategies
  - declarative support for specifying domain heuristics
  - a portfolio of prefabricated expert configurations and
  - an application programming interface for library integration
- Empirical study contrasting them for solving optimization problems

Potassco

Outline

1 Motivation

2 Disjunctive solving

3 Optimization

4 Heuristics

5 Configuration

6 Experiments

7 Summary

Potassco

Solving disjunctive logic programs

- Fact Solving DLPs leads to an elevated level of complexity
- Equitable interplay between generating and testing solvers
  - n generating and $k \times n$ testing solvers (given k head cycle components)

Potassco

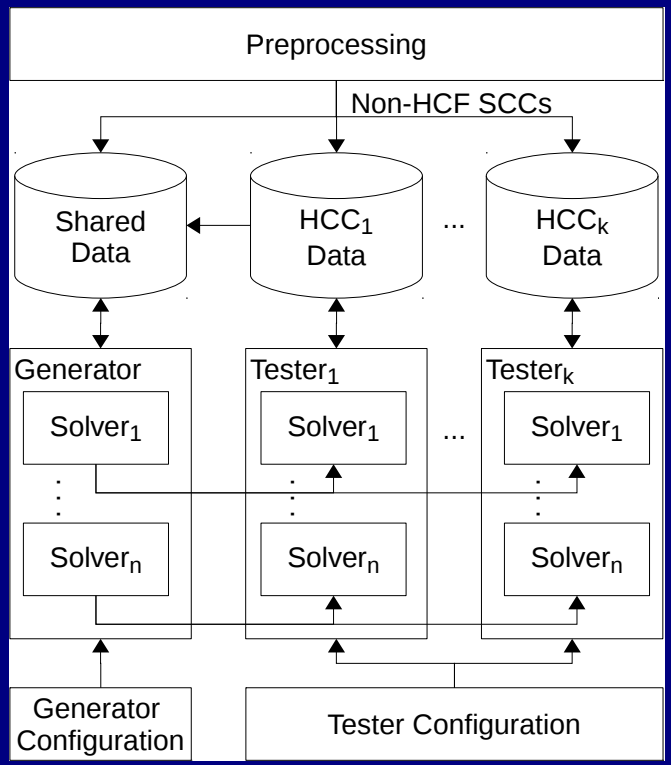
Solving disjunctive logic programs

- Fact Solving DLPs leads to an elevated level of complexity
- Equitable interplay between generating and testing solvers
  - n generating and $k \times n$ testing solvers
    (given k head cycle components)

Potassco

Solving disjunctive logic programs

- Fact Solving DLPs leads to an elevated level of complexity
- Equitable interplay between generating and testing solvers
  - n generating and $k \times n$ testing solvers (given k head cycle components)



Potassco

Solving disjunctive logic programs

- Fact Solving DLPs leads to an elevated level of complexity
- Equitable interplay between generating and testing solvers
  - n generating and $k \times n$ testing solvers
    (given k head cycle components)
  - frequency of expensive unfounded set checks is configurable
    (`--partial-check`)
  - testing solvers are configurable (`--tester`)

Potassco

Solving disjunctive logic programs

- Fact Solving DLPs leads to an elevated level of complexity
- Equitable interplay between generating and testing solvers
  - n generating and $k \times n$ testing solvers (given k head cycle components)
  - frequency of expensive unfounded set checks is configurable (`--partial-check`)
  - testing solvers are configurable (`--tester`)
- Preprocessing
  - `--pre` — Run preprocessing and exit
  - `gringo <file> | clasp --pre | cmodels`

Potassco

Solving disjunctive logic programs

- Fact Solving DLPs leads to an elevated level of complexity
- Equitable interplay between generating and testing solvers
  - n generating and $k \times n$ testing solvers (given k head cycle components)
  - frequency of expensive unfounded set checks is configurable (`--partial-check`)
  - testing solvers are configurable (`--tester`)
- Preprocessing
  - `--pre` — Run preprocessing and exit
  - `gringo <file> | clasp --pre | wasp`

Potassco

Solving disjunctive logic programs

- Fact Solving DLPs leads to an elevated level of complexity
- Equitable interplay between generating and testing solvers
  - n generating and $k \times n$ testing solvers (given k head cycle components)
  - frequency of expensive unfounded set checks is configurable (`--partial-check`)
  - testing solvers are configurable (`--tester`)
- Preprocessing
  - `--pre` — Run preprocessing and exit
  - `gringo <file> | clasp --pre | lp2sat | minisat`

Potassco

Outline

1 Motivation

2 Disjunctive solving

3 Optimization

4 Heuristics

5 Configuration

6 Experiments

7 Summary

Potassco

Optimization strategies

- Model-guided approach `--opt-strategy=bb,`*n*
  - classical branch-and-bound
  - SAT SAT...SAT UNSAT
- Core-guided approach `--opt-strategy=usc,`*n*
  - originated in MaxSAT community
  - UNSAT UNSAT...UNSAT SAT
- Combination via multi-threading
  - exchange of lower and upper bounds (in addition to nogoods)
- Enumeration of optimal models `--opt-mode=optN`
  - combinable with `--enum-mode`,
    eg. to compute intersection and union of optimal models

Potassco

Optimization strategies

- Model-guided approach `--opt-strategy=bb,`*n*
  - classical branch-and-bound
  - SAT SAT...SAT UNSAT
- Core-guided approach `--opt-strategy=usc,`*n*
  - originated in MaxSAT community
  - UNSAT UNSAT...UNSAT SAT
- Combination via multi-threading
  - exchange of lower and upper bounds (in addition to nogoods)
- Enumeration of optimal models `--opt-mode=optN`
  - combinable with `--enum-mode`,
    eg. to compute intersection and union of optimal models

Potassco

Optimization strategies

- Model-guided approach `--opt-strategy=bb,`*n*
  - classical branch-and-bound
  - SAT SAT...SAT UNSAT
- Core-guided approach `--opt-strategy=usc,`*n*
  - originated in MaxSAT community
  - UNSAT UNSAT...UNSAT SAT
- Combination via multi-threading
  - exchange of lower and upper bounds (in addition to nogoods)
- Enumeration of optimal models `--opt-mode=optN`
  - combinable with `--enum-mode`,
    eg. to compute intersection and union of optimal models

Potassco

Optimization strategies

- Model-guided approach `--opt-strategy=bb,`*n*
  - classical branch-and-bound
  - SAT SAT...SAT UNSAT
- Core-guided approach `--opt-strategy=usc,`*n*
  - originated in MaxSAT community
  - UNSAT UNSAT...UNSAT SAT
- Combination via multi-threading
  - exchange of lower and upper bounds (in addition to nogoods)
- Enumeration of optimal models `--opt-mode=optN`
  - combinable with `--enum-mode`,
    eg. to compute intersection and union of optimal models

Potassco

Outline

1 Motivation

2 Disjunctive solving

3 Optimization

4 Heuristics

5 Configuration

6 Experiments

7 Summary

Potassco

Heuristic framework

- Change heuristic scores of atoms and signs `--heuristic=domain`
- Programmed heuristics expressed as a logic program
  - `_heuristic` (must be #shown)
  - `init`, `factor`, `level`, `sign`

```
_heuristic(occurs(A,T),factor,T) :- action(A), time(T).
```

- Structural heuristics invoked via command line options
  - `--dom-mod=`*m*`,`*p*
  - `level`, `sign`

Potassco

Heuristic framework

- Change heuristic scores of atoms and signs `--heuristic=domain`
- Programmed heuristics expressed as a logic program
  - Predicate `_heuristic` (must be #shown)
  - Modifiers `init`, `factor`, `level`, `sign`
  - Example

```
_heuristic(occurs(A,T),factor,T) :- action(A), time(T).
```

- Structural heuristics invoked via command line options
  - Option `--dom-mod=`m`,`p
  - Modifiers `level`, `sign`
  - Example

Potassco

Heuristic framework

- Change heuristic scores of atoms and signs `--heuristic=domain`
- Programmed heuristics expressed as a logic program
  - Predicate `_heuristic` (must be #shown)
  - Modifiers `init`, `factor`, `level`, `sign`
  - Example

```
_heuristic(occurs(A,T),factor,T) :- action(A), time(T).
```

- Structural heuristics invoked via command line options
  - Option `--dom-mod=`m`,`p
  - Modifiers `level`, `sign`
  - Example

Potassco

Heuristic framework

- Change heuristic scores of atoms and signs `--heuristic=domain`
- Programmed heuristics expressed as a logic program
  - Predicate `_heuristic` (must be #shown)
  - Modifiers `init`, `factor`, `level`, `sign`
  - Example

```
_heuristic(occurs(A,T),factor,T) :- action(A), time(T).
```

- Structural heuristics invoked via command line options
  - Option `--dom-mod=`m`,`p
  - Modifiers `level`, `sign`
  - Example

Potassco

Heuristic framework

- Change heuristic scores of atoms and signs `--heuristic=domain`
- Programmed heuristics expressed as a logic program
  - Predicate `_heuristic` (must be #shown)
  - Modifiers `init`, `factor`, `level`, `sign`
  - Example

```
_heuristic(occurs(A,T),factor,T) :- action(A), time(T).
```

- Structural heuristics invoked via command line options
  - Option `--dom-mod=`*m*`,`*p*
  - Modifiers `level`, `sign`
  - Example

Potassco

Heuristic framework

- Change heuristic scores of atoms and signs `--heuristic=domain`
- Programmed heuristics expressed as a logic program
  - Predicate `_heuristic` (must be #shown)
  - Modifiers `init`, `factor`, `level`, `sign`
  - Example

```
_heuristic(occurs(A,T),factor,T) :- action(A), time(T).
```

- Structural heuristics invoked via command line options
  - Option `--dom-mod=`*m*`,`*p*
  - Modifiers `level`, `sign`
  - Example

Potassco

Heuristic framework

- Change heuristic scores of atoms and signs `--heuristic=domain`
- Programmed heuristics expressed as a logic program
  - Predicate `_heuristic` (must be #shown)
  - Modifiers `init`, `factor`, `level`, `sign`
  - Example

```
_heuristic(occurs(mv,5),factor,5) :- action(mv), time(5).
```

- Structural heuristics invoked via command line options
  - Option `--dom-mod=`*m*`,`*p*
  - Modifiers `level`, `sign`
  - Example

Potassco

Heuristic framework

- Change heuristic scores of atoms and signs `--heuristic=domain`
- Programmed heuristics expressed as a logic program
  - Predicate `_heuristic` (must be #shown)
  - Modifiers `init`, `factor`, `level`, `sign`
  - Example

```
_heuristic(occurs(mv,5),factor,5) :- action(mv), time(5).
```

- Structural heuristics invoked via command line options
  - Option `--dom-mod=`*m*`,`*p*
  - Modifiers `level`, `sign`
  - Example `--dom-mod=4,8`
    - 4 negative `sign`
    - 8 atoms in #minimize statements

Potassco

Heuristic framework

- Change heuristic scores of atoms and signs `--heuristic=domain`
- Programmed heuristics expressed as a logic program
  - Predicate `_heuristic` (must be #shown)
  - Modifiers `init`, `factor`, `level`, `sign`
  - Example

```
_heuristic(occurs(mv,5),factor,5) :- action(mv), time(5).
```

- Structural heuristics invoked via command line options
  - Option `--dom-mod=`*m*`,`*p*
  - Modifiers `level`, `sign`
  - Example `--dom-mod=4,8`
    - 4 negative `sign`
    - 8 atoms in #minimize statements

    ➥ often boosts convergence to minimum

Potassco

Heuristic framework

- Change heuristic scores of atoms and signs `--heuristic=domain`
- Programmed heuristics expressed as a logic program
  - Predicate `_heuristic` (must be #shown)
  - Modifiers `init`, `factor`, `level`, `sign`
  - Example

```
_heuristic(occurs(mv,5),factor,5) :- action(mv), time(5).
```

- Structural heuristics invoked via command line options
  - Option `--dom-mod=`*m*`,`*p*
  - Modifiers `level`, `sign`
  - Example `--dom-mod=5,16`

    5 `level` and negative `sign`

    16 atoms in #show statements

Potassco

Heuristic framework

- Change heuristic scores of atoms and signs `--heuristic=domain`
- Programmed heuristics expressed as a logic program
  - Predicate `_heuristic` (must be #shown)
  - Modifiers `init`, `factor`, `level`, `sign`
  - Example

```
_heuristic(occurs(mv,5),factor,5) :- action(mv), time(5).
```

- Structural heuristics invoked via command line options
  - Option `--dom-mod=`*m*`,`*p*
  - Modifiers `level`, `sign`
  - Example `--dom-mod=5,16`

    `5` `level` and negative `sign`

    `16` atoms in #show statements

    ➥ compute $\subseteq$-minimal models wrt shown atoms

Potassco

Heuristic framework

- Change heuristic scores of atoms and signs `--heuristic=domain`
- Programmed heuristics expressed as a logic program
  - Predicate `_heuristic` (must be #shown)
  - Modifiers `init`, `factor`, `level`, `sign`
  - Example

```
_heuristic(occurs(mv,5),factor,5) :- action(mv), time(5).
```

- Structural heuristics invoked via command line options
  - Option `--dom-mod=`m`,`p
  - Modifiers `level`, `sign`
  - Example `--dom-mod=5,16 --enum-mod=domRec`

    `5` `level` and negative `sign`

    `16` atoms in statements

    ➥ enumerate $\subseteq$-minimal models wrt shown atoms

Potassco

Outline

1 Motivation

2 Disjunctive solving

3 Optimization

4 Heuristics

5 Configuration

6 Experiments

7 Summary

Potassco

Prefabricated configurations and portfolios

- Option `--configuration`

`frumpy`	Use conservative defaults as used in earlier *clasp* versions
`jumpy`	Use more aggressive defaults (than `frumpy`)
`tweety`	Use defaults geared towards typical ASP problems
`trendy`	Use defaults geared towards industrial problems
`crafty`	Use defaults geared towards crafted problems
`handy`	Use defaults geared towards large problems
`<file>`	Use configuration file to configure solver(s)

- Option `--print-portfolio`

Potassco

Prefabricated configurations and portfolios

- Option `--configuration`

`frumpy`	Use conservative defaults as used in earlier *clasp* versions
`jumpy`	Use more aggressive defaults (than `frumpy`)
`tweety`	Use defaults geared towards typical ASP problems
`trendy`	Use defaults geared towards industrial problems
`crafty`	Use defaults geared towards crafted problems
`handy`	Use defaults geared towards large problems
`<file>`	Use configuration file to configure solver(s)

- Option `--print-portfolio`

Potassco

Prefabricated configurations and portfolios

- Option `--configuration`

`frumpy`	Use conservative defaults as used in earlier *clasp* versions
`jumpy`	Use more aggressive defaults (than `frumpy`)
`tweety`	Use defaults geared towards typical ASP problems
`trendy`	Use defaults geared towards industrial problems
`crafty`	Use defaults geared towards crafted problems
`handy`	Use defaults geared towards large problems
`<file>`	Use configuration file to configure solver(s)

- Option `--print-portfolio`

Potassco

Outline

1 Motivation

2 Disjunctive solving

3 Optimization

4 Heuristics

5 Configuration

6 Experiments

7 Summary

Potassco

Experimental setup

- Objective Study the interplay of the various techniques
- Subjects Optimization problems
  - Great practical relevance
  - Algorithmic challenge due to multiple SAT and UNSAT problems
- Experimental series I Sum-based optimization
  - core- and model-guided strategies
  - domain heuristics
  - multi-threading
  - computation
- Experimental series II Inclusion-based optimization
  - saturation-based, disjunctive encodings
  - domain heuristics
  - computation and enumeration

Potassco

Experimental setup

- Objective Study the interplay of the various techniques
- Subjects Optimization problems
  - Great practical relevance
  - Algorithmic challenge due to multiple SAT and UNSAT problems
- Experimental series I Sum-based optimization
  - core- and model-guided strategies
  - domain heuristics
  - multi-threading
  - computation
- Experimental series II Inclusion-based optimization
  - saturation-based, disjunctive encodings
  - domain heuristics
  - computation and enumeration

Potassco

Experimental setup

- Objective Study the interplay of the various techniques
- Subjects Optimization problems
  - Great practical relevance
  - Algorithmic challenge due to multiple SAT and UNSAT problems
- Experimental series I Sum-based optimization
  - core- and model-guided strategies
  - domain heuristics
  - multi-threading
  - computation
- Experimental series II Inclusion-based optimization
  - saturation-based, disjunctive encodings
  - domain heuristics
  - computation and enumeration

Potassco

Experimental setup

- Objective Study the interplay of the various techniques
- Subjects Optimization problems
  - Great practical relevance
  - Algorithmic challenge due to multiple SAT and UNSAT problems
- Experimental series I Sum-based optimization
  - core- and model-guided strategies
  - domain heuristics
  - multi-threading
  - computation
- Experimental series II Inclusion-based optimization
  - saturation-based, disjunctive encodings
  - domain heuristics
  - computation and enumeration

Potassco

Experimental setup

- Objective Study the interplay of the various techniques
- Subjects Optimization problems
  - Great practical relevance
  - Algorithmic challenge due to multiple SAT and UNSAT problems
- Experimental series I Sum-based optimization
  - core- and model-guided strategies
  - domain heuristics
  - multi-threading
  - computation
- Experimental series II Inclusion-based optimization
  - saturation-based, disjunctive encodings
  - domain heuristics
  - computation and enumeration

Potassco

Experimental setup, series I

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations
  - tweety
  - trendy
  - crafty

```
--opt-strategy=bb  --config=tweety
--opt-strategy=usc --config=tweety
--dom-mod=5,8   --opt-strategy=bb  --config=tweety
--opt-strategy=bb,2  --config=trendy
--opt-strategy=usc,3 --config=crafty
--dom-mod=4,8 --opt-strategy=bb,1  --config=trendy
--config=myPortfolio4
```

Potassco

Experimental setup, series I

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
  - *model** `--opt-strategy=bb,2 --config=trendy`
  - *core** `--opt-strategy=usc,3 --config=crafty`
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
  - *multi* `--config=myPortfolio4`

Potassco

Experimental setup, series I

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
  - *model** `--opt-strategy=bb,2 --config=trendy`
  - *core** `--opt-strategy=usc,3 --config=crafty`
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
  - *multi* `--config=myPortfolio4`

Potassco

Experimental setup, series I

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
    - model-guided optimization strategy
    - heuristics preferring minimized atoms and assigning them to false
  - *model** `--opt-strategy=bb,2 --config=trendy`
  - *core** `--opt-strategy=usc,3 --config=crafty`
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
  - *multi* `--config=myPortfolio4`

Potassco

Experimental setup, series I

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
  - *model** `--opt-strategy=bb,2 --config=trendy`
  - *core** `--opt-strategy=usc,3 --config=crafty`
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
  - *multi* `--config=myPortfolio4`

Potassco

Experimental setup, series I

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
  - *model** `--opt-strategy=bb,2 --config=trendy`
  - *core** `--opt-strategy=usc,3 --config=crafty`
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
    - * = best configuration for respective optimization strategy
  - *multi* `--config=myPortfolio4`

Potassco

Experimental setup, series I

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
  - *model** `--opt-strategy=bb,2 --config=trendy`
  - *core** `--opt-strategy=usc,3 --config=crafty`
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
  - *multi* `--config=myPortfolio4`

Potassco

Results for sum-based optimization

Benchmark		*model*	*core*	*heuristic*	*model**	*core**	*heuristic**	*multi*
15-puzzle	(16)	260/ 5/ 90	45/ 0/ 100	425/ 9/ 62	266/ 5/ 83	21/ 0/ 100	249/ 5/ 88	9/ 0
Fastfood ^{w}	(29)	9/ 0/ 100	290/ 13/ 55	30/ 0/ 100	22/ 0/ 100	290/ 14/ 67	10/ 0/ 100	7/ 0
Labyrinth	(29)	445/ 18/ 75	299/ 11/ 62	365/ 14/ 84	395/ 15/ 79	250/ 10/ 66	442/ 19/ 58	229/ 9
Sokoban	(28)	1/ 0/ 100	1/ 0/ 100	1/ 0/ 100	1/ 0/ 100	1/ 0/ 100	1/ 0/ 100	0/ 0
Tsp ^{w}	(29)	600/ 29/ 57	600/ 29/ 0	600/ 29/ 100	600/ 29/ 70	600/ 29/ 32	600/ 29/ 73	600/ 29
Wbds	(29)	600/ 29/ 70	421/ 19/ 34	600/ 29/ 82	600/ 29/ 31	394/ 17/ 67	600/ 29/ 72	397/ 17
Abstract ^{m2}	(30)	19/ 0/ 100	99/ 0/ 100	311/ 13/ 57	20/ 0/ 100	73/ 2/ 94	21/ 0/ 100	6/ 0
Connected	(26)	513/ 22/ 75	476/ 20/ 23	513/ 22/ 89	531/ 23/ 52	474/ 20/ 51	514/ 22/ 93	479/ 20
Crossing	(30)	372/ 16/ 78	177/ 5/ 83	451/ 20/ 66	381/ 17/ 61	174/ 6/ 88	367/ 16/ 86	162/ 5
MaxClique	(30)	593/ 29/ 20	50/ 0/ 100	528/ 23/ 61	370/ 13/ 75	23/ 0/ 100	313/ 8/ 91	21/ 0
Valves ^{w}	(30)	508/ 24/ 79	543/ 27/ 10	561/ 28/ 7	515/ 25/ 87	561/ 28/ 55	513/ 25/ 92	518/ 25
Aspeed ^{m2,w}	(30)	57/ 0/ 100	540/ 27/ 38	490/ 21/ 42	89/ 1/ 99	470/ 23/ 54	64/ 0/ 100	65/ 0
Expansion	(30)	103/ 3/ 92	1/ 0/ 100	40/ 0/ 100	63/ 2/ 96	1/ 0/ 100	30/ 0/ 100	0/ 0
Repair	(30)	113/ 1/ 97	0/ 0/ 100	10/ 0/ 100	32/ 0/ 100	1/ 0/ 100	44/ 0/ 100	1/ 0
Iscas85	(30)	129/ 4/ 96	0/ 0/ 100	158/ 7/ 88	134/ 4/ 92	0/ 0/ 100	306/ 13/ 71	0/ 0
Paranoid ^{m2}	(30)	377/ 8/ 79	1/ 0/ 100	103/ 4/ 92	80/ 3/ 94	1/ 0/ 100	59/ 2/ 98	1/ 0
Trendy ^{m4,w}	(30)	485/ 19/ 47	4/ 0/ 100	241/ 11/ 80	254/ 11/ 82	6/ 0/ 100	219/ 10/ 87	6/ 0
Metro ^{w}	(30)	42/ 0/ 100	237/ 7/ 77	325/ 14/ 59	45/ 0/ 100	162/ 4/ 93	29/ 0/ 100	21/ 0
PartnerUnits	(30)	234/ 5/ 94	111/ 2/ 93	150/ 4/ 87	225/ 8/ 82	103/ 1/ 97	251/ 9/ 83	97/ 0
Ricochet	(30)	86/ 0/ 100	85/ 0/ 100	97/ 0/ 100	167/ 2/ 95	88/ 0/ 100	136/ 1/ 97	21/ 0
ShiftDesign ^{m3}	(30)	600/ 30/ 19	23/ 0/ 100	105/ 5/ 86	436/ 16/ 67	44/ 1/ 99	351/ 13/ 80	29/ 0
Timetabling ^{w}	(30)	407/ 17/ 63	8/ 0/ 100	205/ 10/ 84	208/ 10/ 84	31/ 1/ 97	280/ 11/ 73	4/ 0
SUM	(636)	6553/259/1731	4011/160/1676	6307/263/1724	5435/213/1829	3768/**156**/1859	5397/212/**1942**	*2674/105*
AVG		298/ 12/ 79	182/ 7/ 76	287/ 12/ 78	247/ 10/ 83	171/ 7/ 85	245/ 10/ 88	122/ 5

Potassco

Results for sum-based optimization

Benchmark		*model*			*core*			*heuristic*			*model**			*core**			*heuristic**			*multi*	
15-puzzle	(16)	260/	5/	90	45/	0/	100	425/	9/	62	266/	5/	83	21/	0/	100	249/	5/	88	9/	0
Fastfood ^{w}	(29)	9/	0/	100	290/	13/	55	30/	0/	100	22/	0/	100	290/	14/	67	10/	0/	100	7/	0
Labyrinth	(29)	445/	18/	75	299/	11/	62	365/	14/	84	395/	15/	79	250/	10/	66	442/	19/	58	229/	9
Sokoban	(28)	1/	0/	100	1/	0/	100	1/	0/	100	1/	0/	100	1/	0/	100	1/	0/	100	0/	0
Tsp ^{w}	(29)	600/	29/	57	600/	29/	0	600/	29/	100	600/	29/	70	600/	29/	32	600/	29/	73	600/	29
Wbds	(29)	600/	29/	70	421/	19/	34	600/	29/	82	600/	29/	31	394/	17/	67	600/	29/	72	397/	17
Abstract ^{m2}	(30)	19/	0/	100	99/	0/	100	311/	13/	57	20/	0/	100	73/	2/	94	21/	0/	100	6/	0
Connected	(26)	513/	22/	75	476/	20/	23	513/	22/	89	531/	23/	52	474/	20/	51	514/	22/	93	479/	20
Crossing	(30)	372/	16/	78	177/	5/	83	451/	20/	66	381/	17/	61	174/	6/	88	367/	16/	86	162/	5
MaxClique	(30)	593/	29/	20	50/	0/	100	528/	23/	61	370/	13/	75	23/	0/	100	313/	8/	91	21/	0
Valves ^{w}	(30)	508/	24/	79	543/	27/	10	561/	28/	7	515/	25/	87	561/	28/	55	513/	25/	92	518/	25
Aspeed ^{m2,w}	(30)	57/	0/	100	540/	27/	38	490/	21/	42	89/	1/	99	470/	23/	54	64/	0/	100	65/	0
Expansion	(30)	103/	3/	92	1/	0/	100	40/	0/	100	63/	2/	96	1/	0/	100	30/	0/	100	0/	0
Repair	(30)	113/	1/	97	0/	0/	100	10/	0/	100	32/	0/	100	1/	0/	100	44/	0/	100	1/	0
Iscas85	(30)	129/	4/	96	0/	0/	100	158/	7/	88	134/	4/	92	0/	0/	100	306/	13/	71	0/	0
Paranoid ^{m2}	(30)	377/	8/	79	1/	0/	100	103/	4/	92	80/	3/	94	1/	0/	100	59/	2/	98	1/	0
Trendy ^{m4,w}	(30)	485/	19/	47	4/	0/	100	241/	11/	80	254/	11/	82	6/	0/	100	219/	10/	87	6/	0
Metro ^{w}	(30)	42/	0/	100	237/	7/	77	325/	14/	59	45/	0/	100	162/	4/	93	29/	0/	100	21/	0
PartnerUnits	(30)	234/	5/	94	111/	2/	93	150/	4/	87	225/	8/	82	103/	1/	97	251/	9/	83	97/	0
Ricochet	(30)	86/	0/	100	85/	0/	100	97/	0/	100	167/	2/	95	88/	0/	100	136/	1/	97	21/	0
ShiftDesign ^{m3}	(30)	600/	30/	19	23/	0/	100	105/	5/	86	436/	16/	67	44/	1/	99	351/	13/	80	29/	0
Timetabling ^{w}	(30)	407/	17/	63	8/	0/	100	205/	10/	84	208/	10/	84	31/	1/	97	280/	11/	73	4/	0
SUM	(636)	6553/	259/	1731	4011/	160/	1676	6307/	263/	1724	5435/	213/	1829	3768/	**156**/	1859	5397/	212/	**1942**	*2674*/	*105*
AVG		298/	12/	79	182/	7/	76	287/	12/	78	247/	10/	83	171/	7/	85	245/	10/	88	122/	5

Potassco

Results for sum-based optimization

Benchmark		*model*			*core*			*heuristic*			*model**			*core**			*heuristic**			*multi*	
15-puzzle	(16)	260/	5/	90	45/	0/	100	425/	9/	62	266/	5/	83	21/	0/	100	249/	5/	88	9/	0
Fastfood ^{w}	(29)	9/	0/	100	290/	13/	55	30/	0/	100	22/	0/	100	290/	14/	67	10/	0/	100	7/	0
Labyrinth	(29)	445/	18/	75	299/	11/	62	365/	14/	84	395/	15/	79	250/	10/	66	442/	19/	58	229/	9
Sokoban	(28)	1/	0/	100	1/	0/	100	1/	0/	100	1/	0/	100	1/	0/	100	1/	0/	100	0/	0
Tsp ^{w}	(29)	600/	29/	57	600/	29/	0	600/	29/	100	600/	29/	70	600/	29/	32	600/	29/	73	600/	29
Wbds	(29)	600/	29/	70	421/	19/	34	600/	29/	82	600/	29/	31	394/	17/	67	600/	29/	72	397/	17
Abstract ^{m2}	(30)	19/	0/	100	99/	0/	100	311/	13/	57	20/	0/	100	73/	2/	94	21/	0/	100	6/	0
Connected	(26)	513/	22/	75	476/	20/	23	513/	22/	89	531/	23/	52	474/	20/	51	514/	22/	93	479/	20
Crossing	(30)	372/	16/	78	177/	5/	83	451/	20/	66	381/	17/	61	174/	6/	88	367/	16/	86	162/	5
MaxClique	(30)	593/	29/	20	50/	0/	100	528/	23/	61	370/	13/	75	23/	0/	100	313/	8/	91	21/	0
Valves ^{w}	(30)	508/	24/	79	543/	27/	10	561/	28/	7	515/	25/	87	561/	28/	55	513/	25/	92	518/	25
Aspeed ^{m2,w}	(30)	57/	0/	100	540/	27/	38	490/	21/	42	89/	1/	99	470/	23/	54	64/	0/	100	65/	0
Expansion	(30)	103/	3/	92	1/	0/	100	40/	0/	100	63/	2/	96	1/	0/	100	30/	0/	100	0/	0
Repair	(30)	113/	1/	97	0/	0/	100	10/	0/	100	32/	0/	100	1/	0/	100	44/	0/	100	1/	0
Iscas85	(30)	129/	4/	96	0/	0/	100	158/	7/	88	134/	4/	92	0/	0/	100	306/	13/	71	0/	0
Paranoid ^{m2}	(30)	377/	8/	79	1/	0/	100	103/	4/	92	80/	3/	94	1/	0/	100	59/	2/	98	1/	0
Trendy ^{m4,w}	(30)	485/	19/	47	4/	0/	100	241/	11/	80	254/	11/	82	6/	0/	100	219/	10/	87	6/	0
Metro ^{w}	(30)	42/	0/	100	237/	7/	77	325/	14/	59	45/	0/	100	162/	4/	93	29/	0/	100	21/	0
PartnerUnits	(30)	234/	5/	94	111/	2/	93	150/	4/	87	225/	8/	82	103/	1/	97	251/	9/	83	97/	0
Ricochet	(30)	86/	0/	100	85/	0/	100	97/	0/	100	167/	2/	95	88/	0/	100	136/	1/	97	21/	0
ShiftDesign ^{m3}	(30)	600/	30/	19	23/	0/	100	105/	5/	86	436/	16/	67	44/	1/	99	351/	13/	80	29/	0
Timetabling ^{w}	(30)	407/	17/	63	8/	0/	100	205/	10/	84	208/	10/	84	31/	1/	97	280/	11/	73	4/	0
SUM	(636)	6553/	259/	1731	4011/	160/	1676	6307/	263/	1724	5435/	213/	1829	3768/	**156**/	1859	5397/	212/	**1942**	*2674*/	*105*
AVG		298/	12/	79	182/	7/	76	287/	12/	78	247/	10/	83	171/	7/	85	245/	10/	88	122/	5

Potassco

Results for sum-based optimization

Benchmark		*model*			*core*			*heuristic*			*model**			*core**			*heuristic**			*multi*		
15-puzzle	(16)	260/	5/	90	45/	0/	100	425/	9/	62	266/	5/	83	21/	0/	100	249/	5/	88	9/	0	
Fastfood ^{w}	(29)	9/	0/	100	290/	13/	55	30/	0/	100	22/	0/	100	290/	14/	67	10/	0/	100	7/	0	
Labyrinth	(29)	445/	18/	75	299/	11/	62	365/	14/	84	395/	15/	79	250/	10/	66	442/	19/	58	229/	9	
Sokoban	(28)	1/	0/	100	1/	0/	100	1/	0/	100	1/	0/	100	1/	0/	100	1/	0/	100	0/	0	
Tsp ^{w}	(29)	600/	29/	57	600/	29/	0	600/	29/	100	600/	29/	70	600/	29/	32	600/	29/	73	600/	29	
Wbus	(29)	600/	29/	76	421/	19/	34	600/	29/	82	600/	29/	31	394/	17/	67	600/	29/	72	397/	17	
Abstract ^{m2}	(30)	19/	0/	100	99/	0/	100	311/	13/	57	20/	0/	100	73/	2/	94	21/	0/	100	6/	0	
Connected	(26)	513/	22/	75	476/	20/	23	513/	22/	89	531/	23/	52	474/	20/	51	514/	22/	93	479/	20	
Crossing	(30)	372/	16/	78	177/	5/	83	451/	20/	66	381/	17/	61	174/	6/	88	367/	16/	86	162/	5	
MaxClique	(30)	593/	29/	20	50/	0/	100	528/	23/	61	370/	13/	75	23/	0/	100	313/	8/	91	21/	0	
Valves ^{w}	(30)	508/	24/	79	543/	27/	10	561/	28/	7	515/	25/	87	561/	28/	55	513/	25/	92	518/	25	
Aspeed ^{m2,w}	(30)	57/	0/	100	540/	27/	38	490/	21/	42	89/	1/	99	470/	23/	54	64/	0/	100	65/	0	
Expansion	(30)	103/	3/	92	1/	0/	100	40/	0/	100	63/	2/	96	1/	0/	100	30/	0/	100	0/	0	
Repair	(30)	113/	1/	97	0/	0/	100	10/	0/	100	32/	0/	100	1/	0/	100	44/	0/	100	1/	0	
Iscas85	(30)	129/	4/	96	0/	0/	100	158/	7/	88	134/	4/	92	0/	0/	100	306/	13/	71	0/	0	
Paranoid ^{m2}	(30)	377/	8/	79	1/	0/	100	103/	4/	92	80/	3/	94	1/	0/	100	59/	2/	98	1/	0	
Trendy ^{m4,w}	(30)	485/	19/	47	4/	0/	100	241/	11/	80	254/	11/	82	6/	0/	100	219/	10/	87	6/	0	
Metro ^{w}	(30)	42/	0/	100	237/	7/	77	325/	14/	59	45/	0/	100	162/	4/	93	29/	0/	100	21/	0	
PartnerUnits	(30)	234/	5/	94	111/	2/	93	150/	4/	87	225/	8/	82	103/	1/	97	251/	9/	83	97/	0	
Ricochet	(30)	86/	0/	100	85/	0/	100	97/	0/	100	167/	2/	95	88/	0/	100	136/	1/	97	21/	0	
ShiftDesign ^{m3}	(30)	600/	30/	19	23/	0/	100	105/	5/	86	436/	16/	67	44/	1/	99	351/	13/	80	29/	0	
Timetabling ^{w}	(30)	407/	17/	63	8/	0/	100	205/	10/	84	208/	10/	84	31/	1/	97	280/	11/	73	4/	0	
SUM	(636)	6553/	259/	1731	4011/	160/	1676	6307/	263/	1724	5435/	213/	1829	3768/	**156**/	1859	5397/	212/	**1942**	*2674*/	*105*	
AVG		298/	12/	79	182/	7/	76	287/	12/	78	247/	10/	83	171/	7/	85	245/	10/	88	122/	5	

Potassco

Results for sum-based optimization

Benchmark	*model*	*core*	*heuristic*	*model**	*core**	*heuristic**	*multi*
SUM (636)	6553/259/1731	4011/160/1676	6307/263/1724	5435/213/1829	3768/156/1859	5397/212/1942	2674/105
AVG	298/ 12/ 79	182/ 7/ 76	287/ 12/ 78	247/ 10/ 83	171/ 7/ 85	245/ 10/ 88	122/ 5

- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
  - *model** `--opt-strategy=bb,2 --config=trendy`
  - *core** `--opt-strategy=usc,3 --config=crafty`
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
  - *multi* `--config=myPortfolio4`

Potassco

Results for sum-based optimization

Benchmark	*model*	*core*	*heuristic*	*model**	*core**	*heuristic**	*multi*
SUM (636)	6553/259/1731	4011/160/1676	6307/263/1724	5435/213/1829	3768/156/1859	5397/212/1942	2674/105
AVG	298/ 12/ 79	182/ 7/ 76	287/ 12/ 78	247/ 10/ 83	171/ 7/ 85	245/ 10/ 88	122/ 5

- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
  - *model** `--opt-strategy=bb,2 --config=trendy`
    - model-guided optimization with exponentially increasing steps
    - configuration `trendy`
  - *core** `--opt-strategy=usc,3 --config=crafty`
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
  - *multi* `--config=myPortfolio4`

Potassco

Results for sum-based optimization

Benchmark	*model*	*core*	*heuristic*	*model**	*core**	*heuristic**	*multi*
SUM (636)	6553/259/1731	4011/160/1676	6307/263/1724	5435/213/1829	3768/156/1859	5397/212/1942	2674/105
AVG	298/ 12/ 79	182/ 7/ 76	287/ 12/ 78	247/ 10/ 83	171/ 7/ 85	245/ 10/ 88	122/ 5

- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
  - *model** `--opt-strategy=bb,2 --config=trendy`
    - model-guided optimization with exponentially increasing steps
    - configuration `trendy`
  - *core** `--opt-strategy=usc,3 --config=crafty`
    - core-guided optimization with algorithm *oll*
    - configuration `crafty`
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
  - *multi* `--config=myPortfolio4`

Potassco

Results for sum-based optimization

Benchmark	*model*	*core*	*heuristic*	*model**	*core**	*heuristic**	*multi*
SUM (636)	6553/259/1731	4011/160/1676	6307/263/1724	5435/213/1829	3768/156/1859	5397/212/1942	2674/105
AVG	298/ 12/ 79	182/ 7/ 76	287/ 12/ 78	247/ 10/ 83	171/ 7/ 85	245/ 10/ 88	122/ 5

- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
  - *model** `--opt-strategy=bb,2 --config=trendy`
    - model-guided optimization with exponentially increasing steps
    - configuration `trendy` — *frequent restarts*
  - *core** `--opt-strategy=usc,3 --config=crafty`
    - core-guided optimization with algorithm *oll*
    - configuration `crafty` — *infrequent restarts*
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
  - *multi* `--config=myPortfolio4`

Potassco

Results for sum-based optimization

Benchmark	*model*	*core*	*heuristic*	*model**	*core**	*heuristic**	*multi*
SUM (636)	6553/259/1731	4011/160/1676	6307/263/1724	5435/213/1829	3768/156/1859	5397/212/1942	2674/105
AVG	298/ 12/ 79	182/ 7/ 76	287/ 12/ 78	247/ 10/ 83	171/ 7/ 85	245/ 10/ 88	122/ 5

- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
  - *model** `--opt-strategy=bb,2 --config=trendy`
  - *core** `--opt-strategy=usc,3 --config=crafty`
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
    - hierarchic model-guided optimization
    - heuristics preferring to assign false to minimized atoms
  - *multi* `--config=myPortfolio4`

Potassco

Results for sum-based optimization

Benchmark	*model*	*core*	*heuristic*	*model**	*core**	*heuristic**	*multi*
SUM (636)	6553/259/1731	4011/160/1676	6307/263/1724	5435/213/1829	3768/**156**/1859	5397/212/**1942**	2674/105
AVG	298/ 12/ 79	182/ 7/ 76	287/ 12/ 78	247/ 10/ 83	171/ 7/ 85	245/ 10/ 88	122/ 5

- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
  - *model** `--opt-strategy=bb,2 --config=trendy`
  - *core** `--opt-strategy=usc,3 --config=crafty`
    - *most problems solved* (among single-threaded strategies)
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
    - *best anytime behaviour* (among single-threaded strategies)
  - *multi* `--config=myPortfolio4`

Potassco

Results for sum-based optimization

Benchmark	*model*	*core*	*heuristic*	*model**	*core**	*heuristic**	*multi*
SUM (636)	6553/259/1731	4011/160/1676	6307/263/1724	5435/213/1829	3768/156/1859	5397/212/1942	**2674**/**105**
AVG	298/ 12/ 79	182/ 7/ 76	287/ 12/ 78	247/ 10/ 83	171/ 7/ 85	245/ 10/ 88	122/ 5

- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
  - *model** `--opt-strategy=bb,2 --config=trendy`
  - *core** `--opt-strategy=usc,3 --config=crafty`
    - *most problems solved* (among single-threaded strategies)
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
    - *best anytime behaviour* (among single-threaded strategies)
  - *multi* `--config=myPortfolio4`
    - *faster and more problems solved*

Potassco

Results for sum-based optimization

Benchmark	*model*	*core*	*heuristic*	*model**	*core**	*heuristic**	*multi*
SUM (636)	6553/259/1731	4011/160/1676	6307/263/1724	5435/213/1829	3768/156/1859	5397/212/1942	**2674**/**105**
AVG	298/ 12/ 79	182/ 7/ 76	287/ 12/ 78	247/ 10/ 83	171/ 7/ 85	245/ 10/ 88	122/ 5

- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
  - *model** `--opt-strategy=bb,2 --config=trendy`
  - *core** `--opt-strategy=usc,3 --config=crafty`
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
  - *multi* `--config=myPortfolio4`
    - *faster and more problems solved*
    - improves over the virtually best single-threaded configuration

Potassco

Results for sum-based optimization

Benchmark	*model*	*core*	*heuristic*	*model**	*core**	*heuristic**	*multi*
SUM (636)	6553/259/1731	4011/160/1676	6307/263/1724	5435/213/1829	3768/156/1859	5397/212/1942	**2674**/**105**
AVG	298/ 12/ 79	182/ 7/ 76	287/ 12/ 78	247/ 10/ 83	171/ 7/ 85	245/ 10/ 88	122/ 5

- Configurations
  - *model* `--opt-strategy=bb --config=tweety`
  - *core* `--opt-strategy=usc --config=tweety`
  - *heuristic* `--dom-mod=5,8 --opt-strategy=bb --config=tweety`
  - *model** `--opt-strategy=bb,2 --config=trendy`
  - *core** `--opt-strategy=usc,3 --config=crafty`
  - *heuristic** `--dom-mod=4,8 --opt-strategy=bb,1 --config=trendy`
  - *multi* `--config=myPortfolio4`
    - *faster and more problems solved*
    - improves over the virtually best single-threaded configuration

    ☞ *¡SYNERGY AT WORK!*

Potassco

Experimental setup, series II

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations `--configuration=tweety`
  - tweety
  - frumpy
  - crafty
  - jumpy
  - handy
  - trendy
  - many

(enumeration only)

Potassco

Experimental setup, series II

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations `--configuration=tweety`
  - *meta*
  - *heuristic*
  - *meta-heuristic*
  - *meta-heuristic-recording* (enumeration only)

Potassco

Experimental setup, series II

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations (Computation) `--configuration=tweety`
  - *meta*
  - *heuristic*
  - *meta-heuristic*
  - *meta-heuristic-recording* (enumeration only)

Potassco

Experimental setup, series II

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations (Computation) `--configuration=tweety`
  - *meta*
    - saturation-based, disjunctive encodings generated via *metasp*
  - *heuristic*
  - *meta-heuristic*
  - *meta-heuristic-recording* (enumeration only)

Potassco

Experimental setup, series II

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations (Computation) `--configuration=tweety`
  - *meta*
    - saturation-based, disjunctive encodings generated via *metasp*
  - *heuristic* `--dom-mod=5,16`
    - heuristics preferring shown atoms and assigning them to false
  - *meta-heuristic*
  - *meta-heuristic-recording* (enumeration only)

Potassco

Experimental setup, series II

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations (Computation) `--configuration=tweety`
  - *meta*
    - saturation-based, disjunctive encodings generated via *metasp*
  - *heuristic* `--dom-mod=5,16`
    - heuristics preferring shown atoms and assigning them to false
  - *meta-heuristic*
    - use heuristics to reduce number of invalid solution candidates
  - *meta-heuristic-recording* (enumeration only)

Potassco

Experimental setup, series II

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations (Enumeration) `--configuration=tweety`
  - *meta*
    - saturation-based, disjunctive encodings generated via *metasp*
  - *heuristic*
    - heuristics preferring shown atoms and assigning them to false
  - *meta-heuristic*
  - *meta-heuristic-recording* (enumeration only)

Potassco

Experimental setup, series II

- Limits 600 seconds wall-clock time and 6 GB of memory per run
- Measurements
  - Average time (timeout accounts for 600 seconds)
  - Number of timeouts
  - Relative quality (score similar to that of ASP'14)
- Configurations (Enumeration) `--configuration=tweety`
  - *meta*
    - saturation-based, disjunctive encodings generated via *metasp*
    - enumeration in polynomial space
  - *heuristic*
    - heuristics preferring shown atoms and assigning them to false
    - enumeration in exponential space
  - *meta-heuristic*
  - *meta-heuristic-recording* (enumeration only)

Potassco

Results for inclusion-based optimization

Benchmark		*meta*	*heuristic*	*meta-heur.*	*meta*	*heuristic*	*meta-heuristic*	*meta-heur.-rec*
15-puzzle	(16)	25/ 0	14/ 0	23/ 0	321/ 7/ 91	408/ 9/ 75	354/ 7/ 69	444/ 9/ 38
Fastfood	(29)	1/ 0	0/ 0	0/ 0	356/ 14/ 59	210/ 9/ 100	348/ 14/ 65	268/ 10/ 71
Labyrinth	(29)	356/ 16	84/ 3	347/ 15	600/ 29/ 72	600/ 29/ 91	600/ 29/ 73	600/ 29/ 61
Sokoban	(28)	22/ 0	1/ 0	12/ 0	22/ 0/ 95	1/ 0/ 100	23/ 0/ 96	12/ 0/ 98
Tsp	(29)	7/ 0	0/ 0	7/ 0	600/ 29/ 48	600/ 29/ 100	600/ 29/ 58	600/ 29/ 44
Wbds	(29)	219/ 7	23/ 1	38/ 1	600/ 29/ 53	600/ 29/ 82	600/ 29/ 72	600/ 29/ 49
Connected	(26)	109/ 3	0/ 0	61/ 2	532/ 23/ 35	532/ 23/ 100	532/ 23/ 60	532/ 23/ 70
Crossing	(30)	98/ 1	14/ 0	14/ 0	600/ 30/ 32	600/ 30/ 99	600/ 30/ 42	600/ 30/ 76
MaxClique	(30)	189/ 3	0/ 0	3/ 0	600/ 30/ 25	600/ 30/ 100	600/ 30/ 50	600/ 30/ 75
Valves	(30)	600/ 30	560/ 28	600/ 30	600/ 30/ 98	560/ 28/ 100	600/ 30/ 98	600/ 30/ 98
Aspeed	(30)	600/ 30	4/ 0	581/ 29	600/ 30/ 73	600/ 30/ 100	600/ 30/ 74	600/ 30/ 75
Expansion	(30)	600/ 30	0/ 0	600/ 30	600/ 30/ 75	298/ 14/ 100	600/ 30/ 75	600/ 30/ 75
Repair	(30)	552/ 26	0/ 0	5/ 0	595/ 29/ 25	438/ 20/ 100	589/ 29/ 50	481/ 21/ 77
Iscas85	(30)	60/ 3	0/ 0	0/ 0	600/ 30/ 25	600/ 30/ 100	600/ 30/ 50	600/ 30/ 75
Paranoid	(30)	191/ 6	1/ 0	16/ 0	600/ 30/ 25	600/ 30/ 100	600/ 30/ 50	600/ 30/ 75
Trendy	(30)	411/ 18	3/ 0	133/ 0	581/ 29/ 27	580/ 29/ 100	581/ 29/ 51	581/ 29/ 75
Metro	(30)	126/ 5	54/ 1	33/ 1	571/ 27/ 42	576/ 28/ 70	581/ 28/ 65	573/ 27/ 78
PartnerUnits	(30)	600/ 30	168/ 4	507/ 9	600/ 30/ 42	168/ 4/ 98	596/ 29/ 61	501/ 9/ 78
Ricochet	(30)	405/ 16	57/ 0	266/ 10	388/ 14/ 46	56/ 0/ 100	285/ 11/ 77	264/ 10/ 83
Timetabling	(30)	600/ 30	16/ 0	85/ 1	600/ 30/ 27	283/ 14/ 98	600/ 30/ 51	336/ 15/ 82
SUM	(576)	5773/254	999/ **37**	3332/ 128	10568/500/1013	8908/**415**/**1913**	10490/497/1285	9991/450/1453
AVG		289/ 13	50/ 2	167/ 6	528/ 25/ 51	445/ 21/ 96	525/ 25/ 64	500/ 22/ 73

Potassco

Results for inclusion-based optimization

Benchmark	*meta*	*heuristic*	*meta-heur.*	*meta*	*heuristic*	*meta-heuristic*	*meta-heur.-rec*
SUM (576)	5773/254	999/ 37	3332/ 128	10568/500/1013	8908/415/1913	10490/497/1285	9991/450/1453
AVG	289/ 13	50/ 2	167/ 6	528/ 25/ 51	445/ 21/ 96	525/ 25/ 64	500/ 22/ 73

- Configurations (Computation)
  - *meta*
  - *heuristic*
  - *meta-heuristic*
  - *meta-heuristic-recording*

Potassco

Results for inclusion-based optimization

Benchmark	*meta*	*heuristic*	*meta-heur.*	*meta*	*heuristic*	*meta-heuristic*	*meta-heur.-rec*
SUM (576)	5773/254	**999**/ **37**	3332/ 128	10568/500/1013	8908/415/1913	10490/497/1285	9991/450/1453
AVG	289/ 13	50/ 2	167/ 6	528/ 25/ 51	445/ 21/ 96	525/ 25/ 64	500/ 22/ 73

- Configurations (Computation)
  - *meta*
  - *heuristic*
    - *faster and more problems solved*
  - *meta-heuristic*
  - *meta-heuristic-recording*

Potassco

Results for inclusion-based optimization

Benchmark	*meta*	*heuristic*	*meta-heur.*	*meta*	*heuristic*	*meta-heuristic*	*meta-heur.-rec*
SUM (576)	5773/254	999/ 37	3332/ 128	10568/500/1013	**8908**/**415**/**1913**	10490/497/1285	9991/450/1453
AVG	289/ 13	50/ 2	167/ 6	528/ 25/ 51	445/ 21/ 96	525/ 25/ 64	500/ 22/ 73

- Configurations (Enumeration)
  - *meta*
  - *heuristic*
    - *faster and more problems solved*
    - *more models enumerated*
  - *meta-heuristic*
  - *meta-heuristic-recording*

Potassco

Results for inclusion-based optimization

Benchmark	*meta*	*heuristic*	*meta-heur.*	*meta*	*heuristic*	*meta-heuristic*	*meta-heur.-rec*
SUM (576)	5773/254	999/ 37	3332/ 128	10568/500/1013	8908/415/1913	10490/497/1285	9991/450/1453
AVG	289/ 13	50/ 2	167/ 6	528/ 25/ 51	445/ 21/ 96	525/ 25/ 64	500/ 22/ 73

- Configurations (Enumeration)
  - *meta*
    - worst performance
  - *heuristic*
  - *meta-heuristic*
  - *meta-heuristic-recording*

Potassco

Results for inclusion-based optimization

Benchmark	*meta*	*heuristic*	*meta-heur.*	*meta*	*heuristic*	*meta-heuristic*	*meta-heur.-rec*
SUM (576)	5773/254	999/ 37	3332/ 128	10568/500/1013	8908/415/1913	10490/497/1285	9991/450/1453
AVG	289/ 13	50/ 2	167/ 6	528/ 25/ 51	445/ 21/ 96	525/ 25/ 64	500/ 22/ 73

- Configurations (Enumeration)
  - *meta*
    - worst performance
  - *heuristic*
  - *meta-heuristic*
    - better performance
  - *meta-heuristic-recording*

Potassco

Results for inclusion-based optimization

Benchmark	*meta*	*heuristic*	*meta-heur.*	*meta*	*heuristic*	*meta-heuristic*	*meta-heur.-rec*
SUM (576)	5773/254	999/ 37	3332/ 128	10568/500/1013	8908/415/1913	10490/497/1285	9991/450/1453
AVG	289/ 13	50/ 2	167/ 6	528/ 25/ 51	445/ 21/ 96	525/ 25/ 64	500/ 22/ 73

- Configurations (Enumeration)
  - *meta*
    - worst performance
  - *heuristic*
  - *meta-heuristic*
    - better performance
  - *meta-heuristic-recording*
    - even better performance

(enumeration only)

Potassco

Outline

1 Motivation

2 Disjunctive solving

3 Optimization

4 Heuristics

5 Configuration

6 Experiments

7 Summary

Potassco

Summary

- Various ASP solving techniques
  - Disjunctive solving
  - Optimization
  - Heuristics
  - Configuration
- Empirical study of their impact on optimization problems
- Paper
  - Multi-threading
  - C++ library
- `http://potassco.sourceforge.net`

Potassco

Summary

- Various ASP solving techniques
  - Disjunctive solving
  - Optimization
  - Heuristics
  - Configuration
- Empirical study of their impact on optimization problems
- Paper
  - Multi-threading
  - C++ library
- `http://potassco.sourceforge.net`

Potassco

Summary

- Various ASP solving techniques
  - Disjunctive solving
  - Optimization
  - Heuristics
  - Configuration
- Empirical study of their impact on optimization problems
- Paper
  - Multi-threading
  - C++ library
- `http://potassco.sourceforge.net`

Potassco