Optimization in clasp 3*

Torsten Schaub
torsten@cs.uni-potsdam.de
http://potassco.sourceforge.net/videos.html

June 22, 2014

Contents

1 Encoding **2**
1.1 Systems . . . 2
1.2 Basic encoding . . . 2
1.3 Optimization phases . . . 2
1.4 clasp output . . . 3
1.5 More demanding instance . . . 3
1.6 Advanced encoding . . . 3
1.6.1 Encodings . . . 3
1.6.2 Explanation . . . 3
1.6.3 Solving . . . 4
1.6.4 Important note . . . 4
1.7 More demanding instance, continued . . . 4

2 Solving **4**
2.1 Options for optimization . . . 4
2.2 Progress saving . . . 5
2.3 Optimization strategies . . . 5
2.3.1 Multi-criteria optimization . . . 6
2.4 Optimization heuristics . . . 6
2.5 Structure-specific heuristics . . . 6
2.6 Domain-specific heuristics . . . 6
2.7 Parallel optimization . . . 6
2.8 Another example . . . 7

*BIG thanks to Martin Gebser for pimping the encodings!

1 Encoding

1.1 Systems

- commands gringo4 –version clasp3 –version

1.2 Basic encoding

- Example: Traveling Sales Person, Section 3.3 in [1]
  - **Note** [1] uses language of gringo 3
  - See also `http://en.wikipedia.org/wiki/Travelling_salesman_problem`
- commands
  - problem instance view-file graph.lp view-file costs.lp
  - problem encoding view-file tsp.lp4 [2]
  - problem solving gringo4 tsp.lp4 graph.lp costs.lp | clasp3

1.3 Optimization phases

- Branch and bound (top-down)
  1. converging to optimum (SAT ... SAT)
  2. prove optimiality (UNSAT)
- Unsatisfiable core driven (bottom-up; cf [4])
  1. identify and relax cores (UNSAT ... UNSAT)
  2. until consistency (SAT)

[1] M. Gebser, R. Kaminski, B. Kaufmann, and T. Schaub: Answer Set Solving in Practice. Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan and Claypool December 2012, 238 pages, `10.2200/S00457ED1V01Y201211AIM019`

[2] We use extension lp4 to indicate encodings for gringo 4 (along the ASP-Core-2 standard [3])

[3] F. Calimeri, W. Faber, M. Gebser, G. Ianni, R. Kaminski, T. Krennwallner, N. Leone, F. Ricca, and T. Schaub: ASP-Core-2: Input language format. 2012. Available at `https://www.mat.unical.it/aspcomp2013/files/ASP-CORE-2.0.pdf`.

[4] B. Andres, B. Kaufmann, O. Mattheis and T. Schaub: Unsatisfiability-based optimization in clasp. ICLP: 212-221, 2012. Available at `http://www.cs.uni-potsdam.de/wv/pdfformat/ankamasc12a.pdf`

1.4 clasp output

- option

  –quiet[=<m>,<o>],-q : Configure printing of models and optimize values <m>: print {0=all|1=last|2=no} models <o>: print {0=all|1=last|2=no} optimize values [<m>]

  - commands
    gringo4 tsp.lp4 graph.lp costs.lp | clasp3 –quiet=0,0 gringo4 tsp.lp4 graph.lp costs.lp | clasp3 –quiet=1,0 gringo4 tsp.lp4 graph.lp costs.lp | clasp3 –quiet=2,0

1.5 More demanding instance

- Example: Clumpy graphs [5]
- commands gringo4 clumpy-08x08$_{06}$.lp tsp.lp4 | clasp3 –quiet=2,0

1.6 Advanced encoding

- Example: Traveling Sales Person, Section 8.3 in [1]
  - **Note** [1] uses language of gringo 3

1.6.1 Encodings

- commands view-file tsp.lp4 [2] view-file tspA.lp4 [2]
  - **ATTENTION** Objective value is different (though optimum models remain the same)!

1.6.2 Explanation

Consider node 1:

```
edge(1,4). cost(1,4,1).            % <<< lowest cost edge
edge(1,2). cost(1,2,2).
edge(1,3). cost(1,3,3).
```

[5] J. Ward, J. Schlipf: Answer Set Programming with Clause Learning. LPNMR: 302-313, 2004.

```
order(1,1,2).
order(1,2,3).

              %  cycle(1,4)         no penalty
penalty(1,1,1) :- cycle(1,2).      %    penalty of one
penalty(1,2,1) :- cycle(1,3).      %    penalty of one ...
penalty(1,1,1) :- penalty(1,2,1).  %      ... plus one

#minimize{ 1,.. : penalty(1,1,1), 1,.. : penalty(1,2,1), ... }.
```

1.6.3 Solving

gringo4 tsp.lp4 graph.lp costs.lp | clasp3 gringo4 tspA.lp4 graph.lp costs.lp | clasp3

gringo4 tsp.lp4 clumpy-08x08$_{06}$.lp | clasp3 –quiet=2,0 gringo4 tspA.lp4 clumpy-08x08$_{06}$.lp | clasp3 –quiet=2,0

gringo4 tsp.lp4 clumpy-08x08$_{10}$.lp | clasp3 –quiet=2,0 gringo4 tspA.lp4 clumpy-08x08$_{10}$.lp | clasp3 –quiet=2,0

1.6.4 Important note

ALWAYS GET THE ENCODING RIGHT AT FIRST! YOU CAN NEVER RECOVER FROM A BAD ENCODING!

1.7 More demanding instance, continued

- commands gringo4 clumpy-08x08$_{10}$.lp tspA.lp4 > tspA10 clasp3 tspA10 –quiet=2,0

2 Solving

2.1 Options for optimization

- commands (use less on shell ;) clasp3 –help=3 > helper
- view-file helper

2.2 Progress saving

- See [6]
- view-file helper
- commands

  clasp3 tspA10 –quiet=2,0 –save-progress=0 clasp3 tspA10 –quiet=2,0 –save-progress=1
- view-file ham.lp4
- view-file hamO.lp4
- commands

  gringo4 hamO.lp4 clumpy-08x08$_{10}$.lp | clasp3 –quiet=2,0 –save-progress=1
  gringo4 hamO.lp4 clumpy-08x08$_{10}$.lp | clasp3 –quiet=2,0 –save-progress=0
  gringo4 hamO.lp4 clumpy-08x08$_{10}$.lp | clasp3 –quiet=2,0 –save-progress=0 –restart-on-model

2.3 Optimization strategies

- view-file helper
- commands

  clasp3 tspA10 –quiet=2,0 –opt-strategy=0 clasp3 tspA10 –quiet=2,0 –opt-strategy=2 clasp3 tspA10 –quiet=2,0 –opt-strategy=3 clasp3 tspA10 –quiet=2,0 –opt-strategy=4 [4] clasp3 tspA10 –quiet=2,0 –opt-strategy=5
- commands

  gringo4 ham.lp4 clumpy-08x08$_{10}$.lp | clasp3 –quiet=2,0

  gringo4 hamO.lp4 clumpy-08x08$_{10}$.lp | clasp3 –quiet=2,0 –opt-strategy=0
  gringo4 hamO.lp4 clumpy-08x08$_{10}$.lp | clasp3 –quiet=2,0 –opt-strategy=2
  gringo4 hamO.lp4 clumpy-08x08$_{10}$.lp | clasp3 –quiet=2,0 –opt-strategy=3
  gringo4 hamO.lp4 clumpy-08x08$_{10}$.lp | clasp3 –quiet=2,0 –opt-strategy=4

  gringo4 hamO.lp4 clumpy-16x16$_{10}$.lp | clasp3 –quiet=2,0 –opt-strategy=0
  gringo4 hamO.lp4 clumpy-16x16$_{10}$.lp | clasp3 –quiet=2,0 –opt-strategy=4

[6] K. Pipatsrisawat, A. Darwiche: A lightweight component caching scheme for satisfiability solvers. SAT: 294-299, 2007.

2.3.1 Multi-criteria optimization

- view-file tspMO.lp4 (see hamO.lp4)
- commands

  gringo4 tspMO.lp4 clumpy-08x08$_{10}$.lp | clasp3 –quiet=2,0 –opt-strategy=0
  gringo4 tspMO.lp4 clumpy-08x08$_{10}$.lp | clasp3 –quiet=2,0 –opt-strategy=1

2.4 Optimization heuristics

- view-file helper
- commands clasp3 tspA10 –quiet=2,0 –opt-heuristic=1 clasp3 tspA10 –quiet=2,0 –opt-heuristic=2 clasp3 tspA10 –quiet=2,0 –opt-heuristic=3

2.5 Structure-specific heuristics

- See [7]
- view-file helper
- commands clasp3 tspA10 –quiet=2,0 –heuristic=domain –dom-pref=16 –dom-mod=4 clasp3 tspA10 –quiet=2,0 –heuristic=domain –dom-pref=16 –dom-mod=5

2.6 Domain-specific heuristics

- See [7]
- commands gringo4 clumpy-08x08$_{10}$.lp tspA.lp4 | clasp3 –heuristic=domain –quiet=2,0 gringo4 clumpy-08x08$_{10}$.lp tspA.lp4 tspH.lp4 | clasp3 –heuristic=domain –quiet=2,0

2.7 Parallel optimization

- view-file helper
- commands
  - auto configuration clasp3 –print-portfolio clasp3 tspA10 –quiet=2,0 –parallel-mode=4,compete

[7] M. Gebser, B. Kaufmann, R. Otero, J. Romero, T. Schaub and P. Wanko: Domain-specific Heuristics in Answer Set Programming. AAAI: 350-356, 2013. Available at `http://www.cs.uni-potsdam.de/wv/pdfformat/gekaotroscwa13a.pdf`

- homogeneous configuration clasp3 tspA10 –quiet=2,0 –configuration=tweety –opt-strategy=0 –parallel-mode=4,compete
- customized configuration view-file optfolio-heterogeneous clasp3 tspA10 –quiet=2,0 –configuration=optfolio-heterogeneous –parallel-mode=4,compete

2.8 Another example

- Example: Ricochet Robots [8] See also `http://en.wikipedia.org/wiki/Ricochet_Robot` Fix horizon to 15 and try to find a minimum number of moves to reach target position (viz -c goal=4)

- commands view-file RR/robotsN.lp4 [2]

  gringo4 RR/board16-1.lp RR/robots.lp RR/goals16-1.lp RR/robotsN.lp4 -c horizon=15 -c goal=4 > rico16hor15goal4 clasp3 rico16hor15goal4 –quiet=2,0 –opt-strategy=0 clasp3 rico16hor15goal4 –quiet=2,0 –opt-strategy=2 clasp3 rico16hor15goal4 –quiet=2,0 –opt-strategy=3 clasp3 rico16hor15goal4 –quiet=2,0 –opt-strategy=4 clasp3 rico16hor15goal4 –quiet=2,0 –opt-strategy=5

[8] M. Gebser, H. Jost, R. Kaminski, P. Obermeier, O. Sabuncu, T. Schaub and M. Schneider: Ricochet Robots: A transverse ASP benchmark. LPNMR: 348-360, 2013. Available at `http://www.cs.uni-potsdam.de/wv/pdfformat/gejokaobsascsc13a.pdf`