Experimental Analysis on SUP

Yuliya Lierler

1 Brief Introduction to SUP

Answer set solvers Assat and Cmodels operate by computing a supported model M of the given logic program Π, and then continuing this process, if necessary, in the presence of additional constraints expressed by loop formulas. Computing M is accomplished by forming the completion of Π, clausifying it (at this step additional atoms can be introduced) and then invoking a SAT solver.

Solver Sup[1] [Lie08] operates in a similar way, by computing a sequence of supported models of Π, but it does not form the completion of Π. Instead, Sup applies Minisat (v1.12b) to Π itself (with each rule written as a clause), and the "non-clausal constraint" mechanism of Minisat is employed for expressing the supportedness restriction. The new solver can be described as "Cmodels without completion."

Computing a supported model of Π without forming the completion formula can be also accomplished by running the Atleast algorithm of Smodels. Computing supported models using Minisat with non-clausal constraints can be thought of as an alternative implementation of Atleast (enhanced by learning). In this sense, Sup combines Smodels and Cmodels algorithms.

2 Experiments

Here we present experimental analysis that compares performance of the system Sup versus Cmodels[2], Clasp[3], Smodels[4], and Smodels$_{cc}$[5] on tight and nontight logic programs.

In the experiments we used the following versions of the systems: Cmodels v. 3.77 using Minisat 1.12b, Clasp v. 1.0.5, Smodels$_{cc}$ v. 1.08 (implemented on top of Smodels v. 2.26), Smodels v. 2.32. Recall that system Sup extends Minisat 1.12b by implementing additional non-clausal constraints. Therefore, we compare Sup performance against Cmodels that uses Minisat 1.12b for its inference. System Sup0.1 stands for a version of Sup that implements "Unit propagate LP", "All Rules Cancelled", and "Backchain False" propagation rules,

[1] http://www.cs.utexas.edu/users/tag/sup .
[2] http://www.cs.utexas.edu/users/tag/cmodels .
[3] http://www.cs.uni-potsdam.de/clasp/ .
[4] http://www.tcs.hut.fi/Software/smodels .
[5] http://www.nku.edu/%7Ewardj1/Research/smodels_cc.html .

and does not implement "Backchain True" [Lie08]. System Sup, on the other hand, implements all these propagation rules that correspond to the inference of the Atleast algorithm of Smodels.

All considered solvers use preprocessor Lparse[4] to ground the problems. Therefore, the systems were run on identical ground instances. Grounding time is not accounted for in solving time. All times are reported in seconds. *tout* stands for the fact that a system did not terminate with a solution after 10 minutes.

All experiments were run on Intel(R) Pentium(R) D CPU 3.00GHz, 2 cpu cores, cache size 1024 KB, running Linux.

The first table presents the experiments run on tight problems.

Instance	Sup0.1	Sup	Cmodels	Clasp	Smodels_{cc}	Smodels
15 Puzzle: `http://www.cs.engr.uky.edu/ai/benchmarks.html`						
15puzzle.16	0.32	0.29	0.16	0.13	1.58	0.85
15puzzle.17	0.38	0.38	0.26	0.24	6.37	2.24
15puzzle.18	0.66	0.40	0.84	0.42	11.09	4.10
15puzzle.19	1.23	0.72	0.42	0.44	12.47	3.98
Blocks World: bw#blocks.#steps						
bw15.7	0.19	0.19	0.20	0.09	0.13	0.42
bw15.8	0.25	0.24	0.26	0.14	1.70	0.84
bw17.8	0.31	0.30	0.33	0.15	0.23	0.62
bw17.9	0.44	0.41	0.40	0.19	3.32	1.44
bw20.10	0.78	0.74	0.62	0.30	3.35	1.81
bw20.9	0.53	0.46	0.54	0.26	0.39	0.95
Towers of Hanoi: `http://asparagus.cs.uni-potsdam.de`						
towers-hanoi.31_6	9.68	9.20	8.79	4.82	21.23	16.88
towers-hanoi.32_6	3.97	15.52	19.95	8.42	26.22	22.80
towers-hanoi.33_6	8.28	36.69	22.84	14.19	40.11	31.88
towers-hanoi.34_6	19.95	10.02	10.07	12.74	49.93	44.05
towers-hanoi.35_6	43.68	92.96	74.03	23.86	115.83	62.19
towers-hanoi.36_6	183.05	70.71	117.56	38.98	112.44	86.01
towers-hanoi.37_6	35.71	30.75	290.65	34.25	84.75	120.53
towers-hanoi.38_6	243.41	233.82	37.03	70.94	99.87	168.88
towers-hanoi.39_6	96.59	tout	tout	123.40	384.24	237.90
towers-hanoi.40_6	tout	113.15	30.21	114.39	124.73	329.08
towers-hanoi.41_6	123.00	69.51	103.61	169.43	168.74	466.15
towers-hanoi.42_6	tout	389.88	91.28	182.10	tout	tout
towers-hanoi.43_6	tout	42.40	353.74	228.37	204.89	tout
towers-hanoi.44_6	501.80	438.78	498.89	tout	tout	tout
Pigeon Holes with 10 holes: pgh#pigeons						
pgh10	0.01	0.01	0.01	0.00	0.06	0.06

pgh5	0.01	0.00	0.00	0.00	0.08	0.07
pgh7	0.18	0.22	0.14	0.10	20.98	1.67
pgh8	0.75	1.10	0.68	1.00	tout	8.87
pgh9	7.60	9.30	4.03	5.73	tout	47.31
Queens Normal Encoding: q.lp.#queens						
q.lp.18	0.14	0.14	0.14	0.08	155.62	7.84
q.lp.22	0.28	0.27	0.30	0.14	tout	tout
q.lp.24	0.35	0.35	0.38	0.20	tout	tout
q.lp.30	0.70	0.71	0.75	0.44	tout	tout
Queens Cardinality Constraint Encoding: q.lp2.#queens						
q.lp2.18	0.06	0.05	0.06	0.04	147.88	2.24
q.lp2.22	0.11	0.10	0.12	0.08	tout	191.76
q.lp2.24	0.15	0.14	0.17	0.11	tout	267.27
q.lp2.30	0.30	0.28	0.32	0.21	tout	tout
TOAST: `http://asparagus.cs.uni-potsdam.de`						
sequence1-ss1-Plain	0.80	0.82	0.86	0.39	1.08	3.45
sequence2-ss1-Plain	2.62	2.62	2.77	1.42	2.81	11.57
sequence2-ss2-Plain	5.86	3.28	3.56	2.43	40.37	21.85
sequence3-ss1-Plain	5.89	5.87	6.13	3.46	5.65	26.01
sequence3-ss2-Plain	5.36	5.25	5.32	3.21	5.25	25.14
sequence3-ss3-Plain	138.08	49.69	78.45	10.51	444.41	tout
sequence4-ss1-Plain	12.58	11.45	11.55	6.99	10.40	46.50
sequence4-ss2-Plain	14.86	14.68	24.50	9.56	99.85	71.42
sequence4-ss3-Plain	408.99	468.25	tout	294.27	tout	tout
sequence4-ss4-Plain	tout	tout	tout	tout	tout	tout
sequence1-ss1	0.81	0.82	0.84	0.39	1.08	3.42
sequence2-ss1	2.56	2.64	2.68	1.44	2.76	11.46
sequence2-ss2	5.87	3.17	3.65	2.43	40.23	21.93
sequence3-ss1	7.08	6.43	6.60	3.54	6.02	25.35
sequence3-ss2	8.11	9.00	8.34	5.75	79.23	46.07
sequence3-ss3	137.37	49.84	78.45	10.39	444.16	tout
sequence4-ss1	13.75	12.55	12.67	7.06	10.61	46.20
sequence4-ss2	16.13	16.88	13.38	8.96	102.98	74.85
sequence4-ss3	103.33	207.21	16.74	233.09	tout	tout
sequence4-ss4	tout	tout	tout	tout	tout	tout
Vertex Cover vcx.# minimum size vertex cover: `http://www.cs.engr.uky.edu/ai/benchmarks.html`						
vc1.53	8.30	3.23	19.81	tout	tout	218.59
vc2.50	11.55	6.76	13.32	94.43	tout	11.73
vc3.55	0.65	4.09	64.44	512.56	tout	10.59
vc4.54	6.49	10.19	23.05	tout	tout	tout

The second table presents the experiments run on nontight problems. A flag $-t$ abbreviates a flag $-temp$ that allows Sup to forget learnt clauses due to loop formulas (by default Sup adds these clauses into permanent clause database). A flag $-a$ abbreviates a flag $-atomreason$ that forces Sup or Cmodels to add only a clause implied by some loop formula and unsatisfied by a current model rather than the complete loop formula unsatisfied by the model (by default Sup and Cmodels add a complete loop formula unsatisfied by the model). *Cm, Cl,* Sm_{cc}, and *Sm* stand for *Cmodels, Clasp,* $Smodels_{cc}$, and *Smodels*, respectively.

Instance	Sup0.1	Sup	Sup -t	Sup -a	Sup -a-t	Cm	Cm -a	Cl	Sm_{cc}	Sm
Deterministic Automaton: `http://www.fmi.uni-stuttgart.de/szs/research/projects/synthesis/benchmarks030923.html`										
mutex2IDFD	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mutex2Morin	0.05	0.04	0.04	0.04	0.04	0.04	0.05	0.02	0.08	0.22
mutex3IDFD	0.01	0.01	0.01	0.01	0.01	0.02	0.01	0.01	0.07	0.06
mutex3Morin	15.12	15.28	15.21	15.30	15.25	15.68	15.68	15.45	306.30	153.60
mutex4IDFD	1.33	0.73	0.76	0.80	0.74	0.66	0.64	0.80	35.53	13.46
phi2IDFD	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
phi2Morin	0.01	0.00	0.01	0.00	0.01	0.00	0.01	0.00	0.01	0.04
phi3IDFD	0.01	0.00	0.01	0.01	0.01	0.00	0.01	0.00	0.02	0.02
phi3Morin	0.90	0.70	0.56	0.96	0.95	0.62	0.72	0.37	2.50	2.80
phi4IDFD	7.22	0.61	0.78	12.11	14.85	0.39	2.98	0.02	0.31	0.16
phi4Morin	tout	104.29	100.55	tout	448.82	105.54	tout	95.50	tout	tout
phi5IDFD	tout	384.82	tout	tout	tout	67.14	tout	138.56	tout	tout
Bound Model Checking: `http://www.tcs.hut.fi/~kepa/experiments/boundsmodels/`										
dp10.i.O2.b12	48.17	7.93	6.26	30.88	3.51	10.14	22.29	0.20	tout	63.11
dp10.s.O2.b9	1.17	0.65	0.87	0.54	0.42	0.78	0.68	0.14	26.18	13.04
dp12.i.O2.b14	tout	tout	558.27	254.79	88.18	188.89	272.55	7.17	tout	tout
dp12.s.O2.b10	1.00	1.70	2.09	5.89	2.01	0.76	2.05	0.72	tout	337.28
dp6.i.O2.b8	0.23	0.21	0.16	0.20	0.20	0.18	0.16	0.02	0.93	0.32
dp8.i.O2.b10	1.97	0.99	1.09	0.98	0.77	1.53	2.60	0.03	12.97	4.12
dp8.s.O2.b8	0.24	0.22	0.22	0.12	0.20	0.15	0.10	0.02	2.58	1.18
Hamiltonian Cycle: `http://www.cs.engr.uky.edu/ai/benchmarks.html`										
hc_1S	0.20	1.02	tout	tout	2.82	tout	tout	tout	tout	tout
hc_2S	tout	6.43	tout	0.29	5.37	13.50	8.60	0.38	153.44	tout
hc_3S	4.05	52.01	tout	1.28	8.15	5.94	3.10	tout	tout	tout
hc_4S	tout	tout	tout	7.08	2.81	tout	0.94	2.18	14.92	tout
hc_5U	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.02	0.08
hc_6U	tout	tout	tout	tout	tout	tout	tout	tout	tout	tout

hc_7U	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.02	0.08
hc_8U	0.01	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.02	0.08

References

[Lie08] Yuliya Lierler. Abstract answer set solvers. http://www.cs.utexas.edu/users/yuliya/papers/aasp-full.ps, 2008.