A Decision-Making Procedure for Resolution-Based SAT-solvers

Eugene Goldberg

Cadence Research Labs (USA)

SAT-2008, Guangzhou, P.R. China

Summary

- Problem formulation and some background
- Decision making with a reference point (DMRP)
- Description of DMRP-SAT
- DMRP and autarkies
- Experimental results
- Directions for future research and conclusions

Local Search Algorithms

GSAT (1992), WalkSat (1994),….

Advantages:

- They know which clauses are currently falsified
- They look for a satisfying assignment incrementally (in terms of falsified clauses)

Flaws:

- Lack of Boolean Constraint Propagation (BCP)
- No clause learning.
- Work only for satisfiable formulas

DPLL-like procedures

SAT-algorithms based on the DPLL procedure (1962):
Rel-SAT, Grasp, SATO, Chaff, BerkMin, Siege, Minisat,….

Advantages:

- Use BCP
- Use clause learning
- Applicable to both satisfiable and unsatisfiable formulas

Flaws:

- Do not see all clauses that are falsified
- Try to satisfy all clauses at once

Can one combine the two approaches?

Ten challenges in propositional logic (Selman, Kautz, McAllester, 1997).

Challenge 7: (1-2 years). Demonstrate the successful combination of stochastic search and systematic search techniques, by the creation of a new algorithm that outperforms the best previous examples of both approaches.

Previous work: Mazur et al. 1998, Prestwich 2001, Habet at al 2002, Fang et al 2004, Hirsh et al. 2005.

Clause Satisfied Recursively

F	- CNF formula,
$\boldsymbol{p}$	- complete assignment (reference point)
$M(\boldsymbol{p})$	- the set of clauses of F that are falsified by $\boldsymbol{p}$
C	- a clause of $M(\boldsymbol{p})$

Point $\boldsymbol{p}'$ recursively satisfies a clause C falsified by reference point $\boldsymbol{p}$ if

- $\boldsymbol{p}'$ satisfies C
- $M(\boldsymbol{p}') \subset M(\boldsymbol{p})$.

This definition allows to extend the notion of the greedy heuristic of local search to the case when points $\boldsymbol{p}$ and $\boldsymbol{p}'$ can be arbitrarily far away from each other.

Main idea of decision-making with a reference point (DMRP) and DMRP-SAT

- generate an initial reference point $\boldsymbol{p}_1$
- build a sequence of points $\boldsymbol{p}_1,\ldots,\boldsymbol{p}_k$ such that $M(\boldsymbol{p}_1) \subset M(\boldsymbol{p}_2) \ldots \subset M(\boldsymbol{p}_k)$ where $M(\boldsymbol{p}_k)=\varnothing$.

Given $\boldsymbol{p}_i$, to find next reference point $\boldsymbol{p}_{i+1}$ we pick a clause C of $M(\boldsymbol{p}_i)$ and try find a complete assignment recursively satisfying C.

When looking for a new reference point we use a DPLL-like procedure with conflict clause learning. Once a new reference point is found, DMRP-SAT restarts.

DMRP-SAT is a regular DPLL-like SAT-solver with clause learning whose decision making is aimed at recursively satisfying a clause (generalized greedy heuristic of local search).

DMRP in More Detail

p – current reference point

C – clause falsified by ***p*** picked to be satisfied recursively.

y - current partial assignment

D(*C*,***p***,***y***) is equal to {*C*} if ***y*** is empty.

Otherwise a clause *C** of *F* is in *D*(*C*,***p***,***y***) if

a) a variable of *C** is assigned differently in ***y*** and ***p***

b) *C** is not satisfied by ***y***

DMRP-SAT updates *D*(*C*,***p***,***y***) every time an assignment is added to ***y*** (removed from ***y*** when backtracking).

If *D*(*C*,***p***,***y***)= ∅, then clause *C* is recursively satisfied by the following complete assignment ***p*´** (new reference point):

p*´**(x_i) = ***p(x_i), if x_i is not assigned in ***y***

p*´**(x_i) = ***y(x_i), otherwise

An Example

$F = C_1 \wedge .. \wedge C_5$ where $C_1 = x_1 \vee x_2 \vee x_3$, $C_2 = \sim x_1 \vee x_2 \vee x_4$, $C_3 = \sim x_1 \vee \sim x_3$, $C_4 = \sim x_1 \vee x_4$, $C_5 = \sim x_3 \vee \sim x_4 \vee \sim x_5$.

$\boldsymbol{p} = (x_1=0, x_2=0, x_3=0, x_4=0, x_5=0)$ is a reference point

$M(\boldsymbol{p}) = \{C_1\}$, The process of satisfying C_1 recursively is shown below.

$\boldsymbol{y} = \emptyset$, $D(C_1,\boldsymbol{p},\boldsymbol{y}) = \{C_1\}$

Decision assignment: $\boldsymbol{y} = (\{x_1=1\})$, $D(C_1,\boldsymbol{p},\boldsymbol{y}) = \{C_2,C_3,C_4\}$,

BCP: $\boldsymbol{y} = (x_1=1,\{x_3=0\})$, $D(C_1,\boldsymbol{p},\boldsymbol{y}) = \{C_2,C_4\}$,

$\boldsymbol{y} = (x_1=1,x_3=0,\{x_4=1\})$, $D(C_1,\boldsymbol{p},\boldsymbol{y}) = \{C_2\}$.

Decision assignment: $y = (x_1=1,x_3=0,x_4=1,\{x_2=1\})$, $D(C_1,\boldsymbol{p},\boldsymbol{y}) = \emptyset$

Point $\boldsymbol{p}' = (x_1=1,x_2=1,x_3=0,x_4=1,x_5=0)$ is a new reference point (in this particular case, $\boldsymbol{p}'$ is a satisfying assignment as well)

Decision Making Heuristics of DMRP-SAT

We use a combination of two heuristics:

1) Try to satisfy the clauses of $D(C,\boldsymbol{p},\boldsymbol{y})$ as fast as possible.

2) Give preference to most recently derived conflict clauses.

- When picking a clause C (where $C(\boldsymbol{p}) = 0$) to be recursively satisfied we choose a conflict clause derived most recently (if any).

- When making a decision assignment, we pick a conflict clause C^* of $D(C,\boldsymbol{p},\boldsymbol{y})$ derived most recently (if any) and make an assignment to a varaible of C^* satisfying the largest number of clauses of $D(C,\boldsymbol{p},\boldsymbol{y})$.

- If $D(C,\boldsymbol{p},\boldsymbol{y})$ does not contain conflict clauses we make a decision assignment to a variable of a clause of $D(C,\boldsymbol{p},\boldsymbol{y})$ that satisfies the largest number of clauses of $D(C,\boldsymbol{p},\boldsymbol{y})$.

DMRP and Autarkies

There is a strong similarity between the notion of an autarky and that of a recursively satisfied clause.

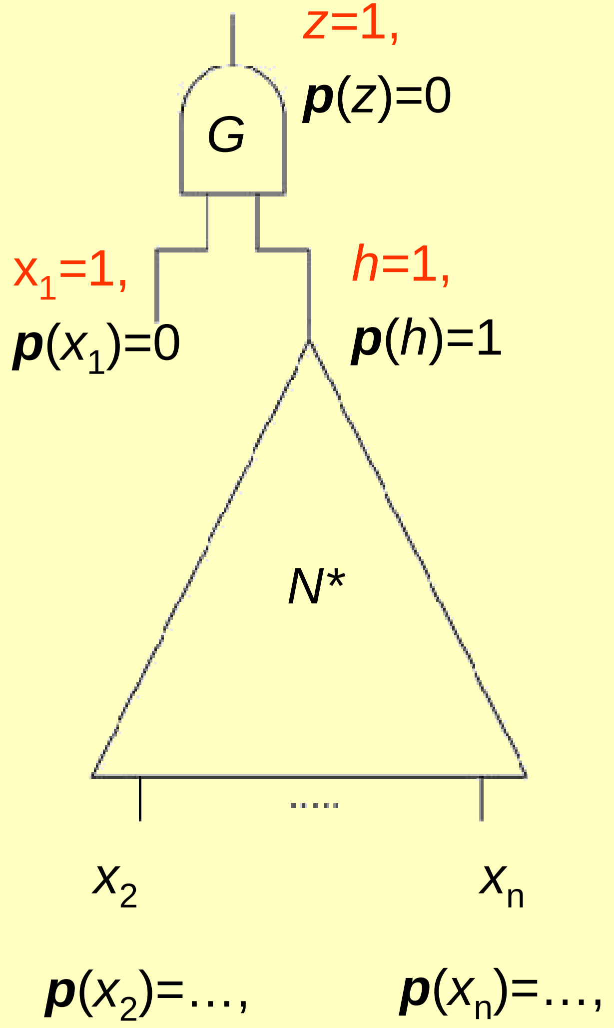
To find an autarky for a CNF formula F, one has to satisfy each clause of F that is

- touched by a previous assignment
- is not satisfied yet

The difference is as follows. When looking for an autarky, an unsatisfied clause C^* is considered as touched if a literal of C^* is set to 0.

When satisfying a clause recursively, an unsatisfied clause C^* is consi-dered as touched only if a literal of C^* is set to 0 by an assignment in $\boldsymbol{y}$ and this assignment is different in $\boldsymbol{y}$ and the reference point $\boldsymbol{p}$.

Example



$F = F(G) \wedge F(N^*) \wedge z$

> $z = 1 \;\rightarrow\; (x_1=1,\ h=1)$
> $F\ (z=1, x_1=1, h=1)$ is not an autarky
> (clauses of $F(N^*)$ are touched by $h=1$.)

Suppose $\boldsymbol{p}$ is a reference point ($z=0, x_1=0, h=1, \ldots$) that falsifies only unit clause z.

Point $\boldsymbol{p}$ can be obtained by making assignments to input variables of $x_1,..,x_n$ and then computing the rest of the values by BCP.

> After making assignments $z=1, x_1=1, h=1$ the unit clause z is satisfied recursively (assignment $h=1$ is the same as in reference point $\boldsymbol{p}$, so clauses of $F(N^*)$ are not touched.

Experimental Results (1)

BMC formulas

sat/unsat (#form.)	BerkMin (a version that is not publicly avail.)		Minisat (v1.13)		DMRP-SAT	
	#cnfl. $\times 10^3$	total time (sec.)	#cnfl. $\times 10^3$	total time (sec.)	#cnfl. $\times 10^3$	total time (sec.)
sat (28)	2,546	44,814	3,457	58,319	**333**	**9,569**
unsat(51)	2,156	28,594	1,355	**14,507**	**791**	15,160
total(79)	4,702	73,408	4,812	72,826	**1,124**	**24,725**

Experimental Results (2)

A sample of BMC formulas (satisfiable* and unsatisfiable)

Name	#vars $\times 10^6$	#clauses $\times 10^6$	Minisat		DMRP-SAT	
			#cnfl. $\times 10^3$	total time (sec.)	#cnfl. $\times 10^3$	total time (sec.)
sched*	1.0	2.7	24	386	**0.07**	**2.6**
ipt*	1.2	3.5	108	3,029	**4.8**	**205**
sdl*	0.4	1.2	149	**472**	**75**	1,659
always	0.2	0.8	21	213	**5.0**	**38**
page	0.2	0.8	19	**151**	**14**	425
cmcnt	1.2	3.6	**2.5**	**68**	3.0	134

Experimental Results (3)

Statistics on recursively satisfied clauses (DMRP-SAT)

Name	#conflicts	size of initial $M(\boldsymbol{p})$	#cases of rec. sat. clauses	#longest chain	$\lvert\boldsymbol{y}\rvert \,/\, \lvert Vars(F)\rvert$ when $M(\boldsymbol{p})=0$ %
sched*	67	1	1	1	18
byteen*	8,824	543	255	255	3.5
data*	15,521	1,034	212	114	77
prop3*	77,127	44	29	6	76
SUN-443*	2,010	3,999	2,000	2,000	1.6

Conclusions

We describe a DPLL-like procedure whose decision making is based on a generalization of the greedy heuristic of local search

Experiments show that this decision making procedure is viable and merits further research.

Directions for Future Research

Best way to choose next clause to be recursively satisfied

Best way to generate a reference point (in case DMRP-SAT fails to satisfy a chosen clause within a threshold)

Developing a version of DMRP-SAT where a set of clauses (rather than one clause) is recursively satisfied

Approximate computation of the set $D(C,\boldsymbol{p},\boldsymbol{y})$ (currently DMRP is more expensive than conflict-driven decision making of Chaff)