GunSAT : A greedy Local Search Algorithm for Unsatisfiability

Gilles Audemard
Univ d'Artois, CRIL - CNRS, FRE2499,
Lens, F-62307
audemard@cril.univ-artois.fr

AND

Laurent Simon
Univ Paris-Sud, LRI - CNRS, UMR8623,
INRIA-Futurs, Orsay, F-91405
simon@lri.fr

Background

Local Search for SAT problems

Data: Σ **a CNF formula**
Result: **sat if a model is found, unknown otherwise**
begin
  for *i=1 to MaxTries* **do**
    Choose a random interpretation I **for** *j=1 to MaxFlips* **do**
      if *I is a model of* Σ **then** return SAT
      I = neighbour(I)
    end
  end
  return UNKNOWN
end

Top-5 reasons for this work

1. Because **there is none**
2. Local Search may be **more efficient** than complete methods
3. It's a **challenge**
4. It may be used to obtain **short proofs**
5. It may be used for **QBF solving**

Resolution

Resolution Rule for *producing* clauses [Robinson 1965]
Let $c_1 = (x \vee a_1 \vee a_2 \vee \dots a_n)$ and $c_2 = (\neg x \vee b_1 \vee b_2 \vee \dots b_m)$ be two clauses.
The clause $c = (a_1 \vee a_2 \vee \dots a_n \vee b_1 \vee \dots b_m)$ is called *the resolvent* of the clauses c_1 and c_2 (on the variable x).
We note $c = c_1 \otimes c_2$ this rule.

- **A Resolution Proof** is a series of clauses, each of them is obtained by the resolution rule with previous clauses, or is an initial clause.
- **Resolution limits**: Some known problems cannot be solved by polynomially-bounded general resolution proof (Pigeon-Hole, Urquhart, Random problems, ...). Thus, ***large* clauses must be produced** by resolution before producing $\bot$.

Extended resolution

Introduced in [Tseitin 1970]
If Σ is a formula, we can consider $\Sigma \wedge (\ e \Leftrightarrow l_1 \vee l_2)$ at any step, where e is a fresh variable and l_1 and l_2 are literals from Σ.

- **No hard examples are known for Extended Resolution.**
- Do not limit the number of new extended variables
- No Current Implementations of ER, but related to ROBDD, Multi-Resolution, Symmetries, ...

GunSAT algorithm

Data: Σ **a CNF formula**
Result: **unsat if a derivation of** $\bot$ **is found, unknown otherwise**
begin
  for *i=1 to* MaxTries **do**
    for *j=1 to* MaxFlips **do**
      if 2-Saturation(Σ) *returns* UNSAT **then**
        return UNSAT
      if $|\Sigma| >$ *MaxSize* **then**
        Remove-One-Clause(Σ)
      Add-One-Clause(Σ)
      Add-Extended-Variables(Σ)
      Simplify-Look-Ahead(Σ)
    end
    Replace Σ by all its *vital* clauses
  end
  return UNKNOWN
end

Clause scoring

Estimating how many models are explicitly filtered out

Measure at depth 1: Maintain the number of filtered models for each literals
A clause c_i of length $n_i > 1$ filter out 2^{n-n_i} of the models.
Assumption: All those filtered models are separately and equally distributed over all literals in the clause. Then the clause filter $w_1(n_i) = \frac{2^{n-n_i}}{n_i}$ of the models containing $\neg l$.

Deeper and deeper... Depth 2: For a clause c_i of length n_i, the 2^{n-n_i} filtered models are supposed as equally distributed over the $n_i.(n_i-1)/2$ pairs of literals occuring in c_i.

- Each pair (l_1, l_2) appearing in c_i is credited a weight of $w_2(n_i) = \frac{2^{n-n_i+1}}{n_i.(n_i-1)}$.
- Score of a pair (l_1, l_2): sum of its weights **in all clauses**
- Score $S(c)$ of a clause c: sum of the scores of all the pairs of literals it contains.

A remark on the scoring

How to link the clause scoring to its importance in the proof?
What if $S(c_i) \simeq n_i.(n_i-1).w_2(n_i)/2$**?**
c_i is nearly the only one that filter the models composed by the negation of its literals. Even if c_i is large, it should be kept.
What if $S(c_i) \gg n_i.(n_i-1).w_2(n_i)/2$**?**
There is a little hope that this clause is from great importance.

And now... the quadruplets

We need a step in our proof where l and $\neg l$ are in Σ. We have to find two literals l_1 and l_2 such that clauses $l_1 \vee l_2$, $\neg l_1 \vee l_2$, $l_1 \vee \neg l_2$ and $\neg l_1 \vee \neg l_2$ can be derived from Σ.
Improving quadruplets scores

$$S_q([x_1, x_2]) = S(l_1, l_2)^2 + S(\neg l_1, l_2)^2 + S(l_1, \neg l_2)^2 + S(\neg l_1, \neg l_2)^2$$

Any move that enhance the score of one of the best scored quadruplets is a greedy move.

Refinements

Binary Saturation

Binary clauses have always a high score. Whenever a binary clause is added, all new binary clauses that may be deduced from it are also added.

LookAhead Strategies

Enhance the power of gunsat

- Work on pairs of literals.
- Try to see what happens if values $(0,0), (0,1), (1,0), (1,1)$ are set to each pair of literals.
- Apply Unit Propagation

If any literal l of Σ is set to $\bot$ in all the four tries, then LH proved that $\Sigma \vdash \neg l$, and the unary clause $\neg l$ is added to Σ. May allow to discover equivalency literals, unit clauses, ...

Extended Resolution

Simple but efficient
Tricky increasing of pairs scores When we tried to increase a pair score too many-times without any success, use **extended resolution** to artificially increase the score of this pair of literals.
$e \Leftrightarrow l_1 \vee l_2$ is encoded by the three clauses
$(\neg e \vee l_1 \vee l_2)$,
$(e \vee \neg l_1)$
and $(e \vee \neg l_2)$.

Restarting: forget, but not too much

After MaxFlips**:** All clauses, except binary ones and the set of *vital* clauses are removed.
Σ may never be the same from restart to restart. Hopefully, Σ will only evolve to **a simpler and simpler formula**.
Deleting extended clauses: All clauses containing at least one extended variables are deleted after each restart, including binary ones.

Experiments

Structured Instances

	basic		LH		ER		LH + ER	
	% S	T (F)	% S	T (F)	% S	T (F)	% S	T (F)
aim-50 (8)	12	2.14 (26)	100	1.41 (146)	60	15.14 (1749)	100	1.58 (142)
aim-100 (8)	10	36.83 (3954)	55	11.93 (923)	27	139.74 (4998)	97	49.37 (1726)
aim-200 (8)	0	-	60	63.62 (1098)	5	739.00 (9099)	85	201.29 (2009)
jnh (33)	6	0.95 (0)	57	8.48 (276)	18	68.15 (986)	62	4.21 (687)
xor (39)	0	-		-	1.5	308.83 (6932)	11	31.91 (5197)

Random instances

		LH + ER	
V	R	% S	T (F)
50	4.25	58	60 (3880)
50	5.0	86	18 (1520)
50	6.0	97	5 (545)
60	4.25	35	126 (5785)
60	5.0	68	67 (3346)
60	6.0	92	16 (1094)
70	4.25	23	189 (6626)
70	5.0	51	187 (6193)
70	6.0	87	59 (2389)

More work to do...

Enhance performances

- Increase flip speed
- Lazy measure of pairs and quadruplets (Top-N)
- Unit Propagation smart data structures
- ...

... Try it on QBF ...