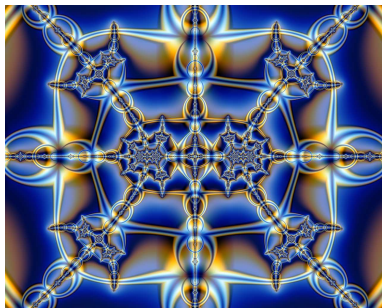


Symmetry Breaking in Local Search for Unsatisfiability

Fadi Aloul, American University of Sharjah, UAE
Inês Lynce, Technical University of Lisbon, Portugal
Steven Prestwich, 4C/University College Cork, Ireland

SymCon'07

Motivation



- *Symmetry breaking* is known for reducing the search space and therefore speeding up the search...
- *but* local search benefits from having many solutions.
- What about when local search is used for proving *unsatisfiability*?

Outline

- Boolean Satisfiability (SAT)
- Symmetry Breaking and SAT
- Local Search for Unsatisfiability
- Experimental Results
- Conclusions

Boolean Satisfiability (SAT)

- Boolean formula φ is defined over a set of propositional variables x_1, \dots, x_n , using the standard propositional connectives $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$, and parenthesis
 - The domain of propositional variables is $\{0, 1\}$
 - A formula φ in conjunctive normal form (CNF) is a conjunction of disjunctions (**clauses**) of **literals**, where a literal is a variable or its complement
- The Boolean satisfiability (SAT) problem:
 - Find an assignment to the variables x_1, \dots, x_n such that $\varphi(x_1, \dots, x_n) = 1$, or prove that no such assignment exists
- SAT is an **NP-complete** decision problem [Cook'71]
 - SAT was the first problem to be shown NP-complete
 - There are **no** known polynomial time algorithms for SAT

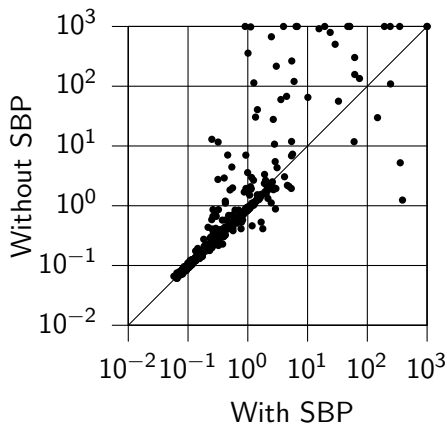
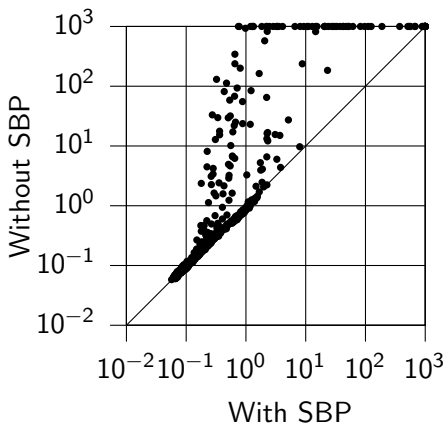
Algorithms for SAT

- Incomplete algorithms (i.e. **can only prove (un)satisfiability**):
 - Local search / hill-climbing
 - Genetic algorithms
 - Simulated annealing
 - ...
- Complete algorithms (i.e. **can prove both satisfiability and unsatisfiability**):
 - Proof system(s)
 - ▶ Resolution
 - ▶ Stalmarck's method
 - ▶ Recursive learning
 - ▶ ...
 - Binary Decision Diagrams (BDDs)
 - Backtrack search / DPLL
 - ▶ Conflict-Driven Clause Learning (CDCL)
 - ...

Symmetry Breaking and SAT

- Most work on symmetry breaking in SAT focuses on **instance-dependent** symmetries
 - Identification of permutable variables via graph automorphism
 - ▶ Shatter tool
 - These symmetries are *dependent* from each CNF formula: you do not have to know about the problem
 - ▶ Main drawback: number of symmetry breaking predicates can be exponential
- Little work on identification of **instance-independent** symmetries
 - Symmetries are broken according to specific properties
 - Approach seldom used in Constraint Programming

Symmetry Breaking and UNSAT/SAT



- 835 **UNSATISFIABLE**/**SATISFIABLE** bioinformatics problem instances, with symmetries broken according to specific properties
- Symmetry breaking is **more critical** for **unsatisfiable** instances

Local Search for Unsatisfiability

- Local search (LS) is widely applied to satisfiable problems, and sometimes beats DPLL
- Challenge by Selman, Kautz & McAllester in 1997: use LS instead to prove unsatisfiability
- RANGER: a greedy randomised resolution algorithm that will eventually discover proofs of any size while using bounded memory [SAT'06]
 - RANGER can refute some problems more quickly than backtrack search
 - Similar independent work with GUNSAT [IJCAI'07]

RANGER (i): main ideas

- RANGER (RANdomised GEneral Resolution) performs a randomised biased search in the space of multisets $\phi_i \subseteq \phi$
 - Each ϕ_i will be of the same constant size, and derived from ϕ_{i-1} by application of resolution or replacement of a clause by one taken from ϕ
- Greedily prefers resolvents that reduce the average resolvent size, but noisily allows some longer resolvents
- Allows any SAT-preserving transformation at each iteration
 - Subsumption and pure literal elimination applied randomly
- Termination occurs when the empty clause is derived
- Proved probabilistic approximate completeness (PAC): it eventually refutes any unsatisfiable problem

RANGER (ii): implementation

```
RANGER( $\phi, p_i, p_t, p_g, w, k$ ):  
   $i \leftarrow 1$  and  $\phi_1 \leftarrow \{\text{any } k \text{ clauses from } \phi\}$   
  while  $\phi_i$  does not contain the empty clause  
    with probability  $p_i$   
      replace a random  $\phi_i$  clause by a random  $\phi$  clause  
    otherwise  
      resolve random  $\phi_i$  clauses  $c, c'$  giving  $r$   
      if  $r$  is non-tautologous and  $|r| \leq w$   
        with probability  $p_g$   
          if  $|r| \leq \max(|c|, |c'|)$  replace the longer of  $c, c'$  by  $r$   
        otherwise  
          replace a random  $\phi_i$  clause by  $r$   
    with probability  $p_t$   
      apply any sat-preserving transformation to  $\phi, \phi_i$   
   $i \leftarrow i + 1$  and  $\phi_{i+1} \leftarrow \{\text{the new formula}\}$   
  return UNSATISFIABLE
```

RANGER (iii): an example

Original CNF formula with
3 variables and 5 clauses

$$\begin{aligned}\omega_1 &= (a \vee b \vee c) \\ \omega_2 &= (\neg a \vee b \vee c) \\ \omega_3 &= (a \vee \neg c) \\ \omega_4 &= (\neg a \vee \neg c) \\ \omega_5 &= (\neg b \vee c)\end{aligned}$$

$$k \geq 3 + 1$$

4 clauses

$$\begin{aligned}\omega_1 &= (a \vee b \vee c) \\ \omega_2 &= (\neg a \vee b \vee c) \\ \omega_4 &= (\neg a \vee \neg c) \\ \omega_5 &= (\neg b \vee c)\end{aligned}$$

$\omega_6 = \text{res}(\omega_1, \omega_2, a)$
Replace ω_4 by ω_6

$$\begin{aligned}\omega_3 &= (a \vee \neg c) \\ \omega_4 &= (\neg a \vee \neg c) \\ \omega_5 &= (\neg b \vee c) \\ \omega_6 &= (b \vee c)\end{aligned}$$

Delete ω_1 and ω_2
(subsumed by ω_6)

Pick ω_3 and ω_4
to have 4 clauses

$$\begin{aligned}\omega_1 &= (a \vee b \vee c) \\ \omega_2 &= (\neg a \vee b \vee c) \\ \omega_5 &= (\neg b \vee c) \\ \omega_6 &= (b \vee c)\end{aligned}$$

$$\square = \text{res}(\text{res}(\omega_3, \omega_4, a), \text{res}(\omega_5, \omega_6, b), c)$$

Experimental Setup

- Results are medians over 10 runs
- Problem instances from **different domains**:
 - Circuits
 - Genetics
 - Graphs
 - Pigeon hole
- Symmetries broken using Shatter:
 - Detection of graph automorphisms

Experimental Results I

without symmetry breaking					
instance	#V	#C	#Lit	#Steps	#Time
chnl10_11	220	1122	2420	n/a	>1000
chnl10_12	240	1344	2880	n/a	>1000
chnl10_13	260	1586	3380	n/a	>1000
chnl11_12	264	1476	3168	n/a	>1000
chnl11_13	286	1742	3718	n/a	>1000
chnl11_20	440	4220	8800	n/a	>1000
hole7	56	204	448	n/a	>1000
hole8	72	297	648	n/a	>1000
hole9	90	415	900	n/a	>1000
hole10	110	561	1210	n/a	>1000
hole11	132	738	1584	n/a	>1000
hole12	156	949	2028	n/a	>1000
Urq3_5	46	470	2912	n/a	>1000
x1.1_16	46	122	364	n/a	>1000
2pipe	892	6695	18637	n/a	>1000
1dlx_c_mc_ex_bp_f	776	3725	10045	n/a	>1000
simp-b2ar_5.01	110	428	992	n/a	>1000
simp-ace_5.07	187	643	1584	n/a	>1000
simp-nonuniform-10_50.06	156	522	1141	n/a	>1000
simp-hapmap_chr21_HCB_30	223	880	2363	n/a	>1000

Experimental Results II

with symmetry breaking					
instance	#V	#C	#Lit	#Steps	#Time
chnl10_11	728	3077	9103	2700218	11.605
chnl10_12	796	3487	10213	2999725	18.605
chnl10_13	864	3917	11363	3326896	27.67
chnl11_12	878	3847	11291	4417899	33.85
chnl11_13	953	4321	12561	5155214	50.905
chnl11_20	1478	8255	22683	948902	1.55
hole7	153	567	1663	241347	0.35
hole8	199	776	2261	352256	0.535
hole9	251	1026	2967	626528	1.015
hole10	309	1320	3787	948902	1.55
hole11	373	1661	4727	1153560	2.1
hole12	443	2052	5793	1784522	3.825
Urq3_5	46	500	2941	n/a	>1000
x1.1_16	48	142	385	n/a	>1000
2pipe	1246	8137	23571	n/a	>1000
1dlx_c_mc_ex_bp_f	780	3746	10073	n/a	>1000
simp-b2ar_5.01	120	449	1022	n/a	>1000
simp-ace_5.07	205	694	1670	n/a	>1000
simp-nonuniform-10_50.06	160	531	1153	n/a	>1000
simp-hapmap_chr21_HCB_30	223	913	2395	n/a	>1000

Conclusions

- Evidence that symmetry breaking **speeds up** local search for unsatisfiability
 - No evidence that that symmetry breaking harms RANGER's performance
- Possible explanations:
 - Symmetry breaking clauses allow **smaller refutations**
 - Additional variables introduced by Shatter
 - ▶ Similar effect to **extended resolution** that can lead to exponentially smaller proofs
- More experiments **needed!**