

Naming, optimised

Lemma. Let S be a set formulas, B a formula, and n a boolean variable not occurring in S, B . Let S' be a set of formulas obtained from S by replacing one or more **positive** (respectively, **negative**) occurrences of B in S by n . Then S is satisfiable if and only if so is $S' \cup \{n \rightarrow B\}$ (respectively, $S' \cup \{B \rightarrow n\}$).

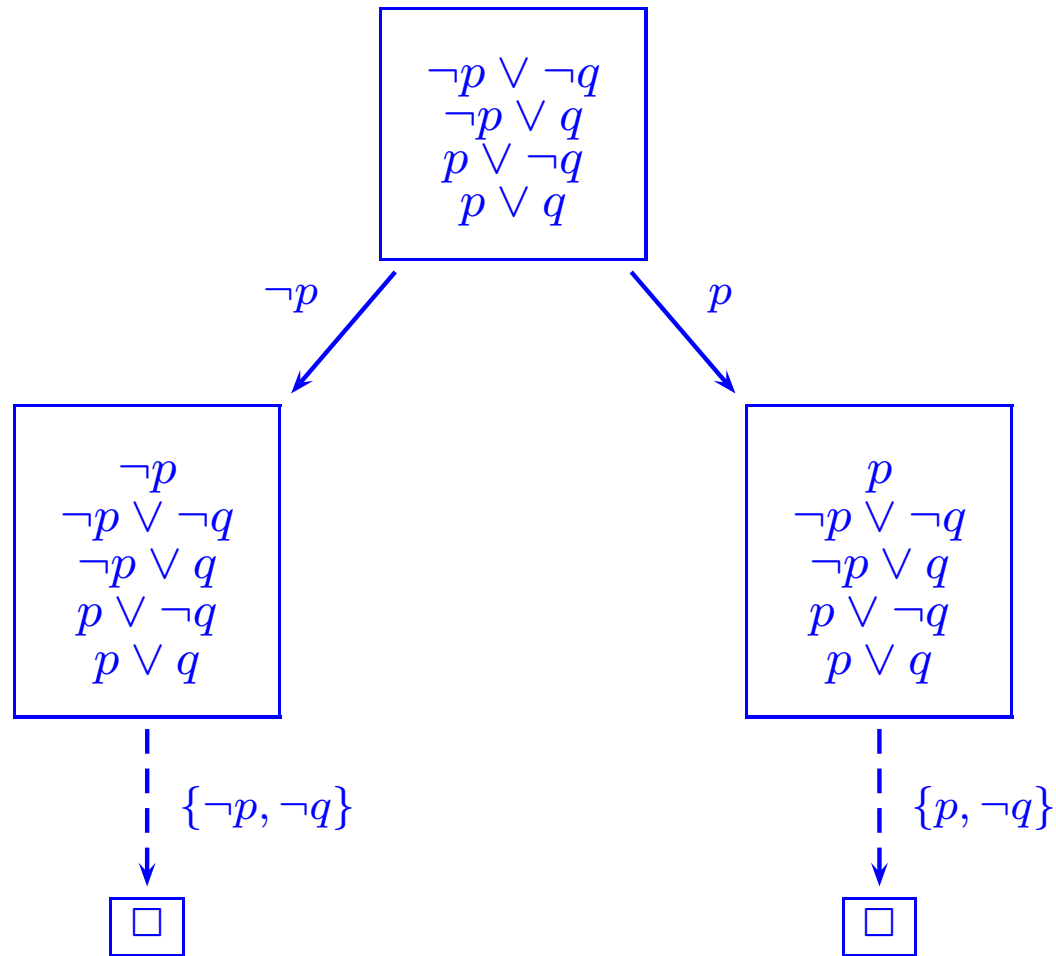
Example

name	subformula	polarity	formula to be transformed to CNF	clauses
				p_1
p_1	$\neg((p \rightarrow q) \wedge (p \wedge q \rightarrow r) \rightarrow (p \rightarrow r))$	+1	$p_1 \rightarrow \neg p_2$	$\neg p_1 \vee \neg p_2$
p_2	$(p \rightarrow q) \wedge (p \wedge q \rightarrow r) \rightarrow (p \rightarrow r)$	-1	$(p_3 \rightarrow p_7) \rightarrow p_2$	$p_3 \vee p_2$ $\neg p_7 \vee p_2$
p_3	$(p \rightarrow q) \wedge (p \wedge q \rightarrow r)$	+1	$p_3 \rightarrow (p_4 \wedge p_5)$	$\neg p_3 \vee p_4$ $\neg p_3 \vee p_5$
p_4	$p \rightarrow q$	+1	$p_4 \rightarrow (p \rightarrow q)$	$\neg p_4 \vee \neg p \vee q$
p_5	$p \wedge q \rightarrow r$	+1	$p_5 \rightarrow (p_6 \rightarrow r)$	$\neg p_5 \vee \neg p_6 \vee r$
p_6	$p \wedge q$	-1	$(p \wedge q) \rightarrow p_6$	$\neg p \vee \neg q \vee p_6$
p_7	$p \rightarrow r$	-1	$(p \rightarrow r) \rightarrow p_7$	$p \vee p_7$ $\neg r \vee p_7$

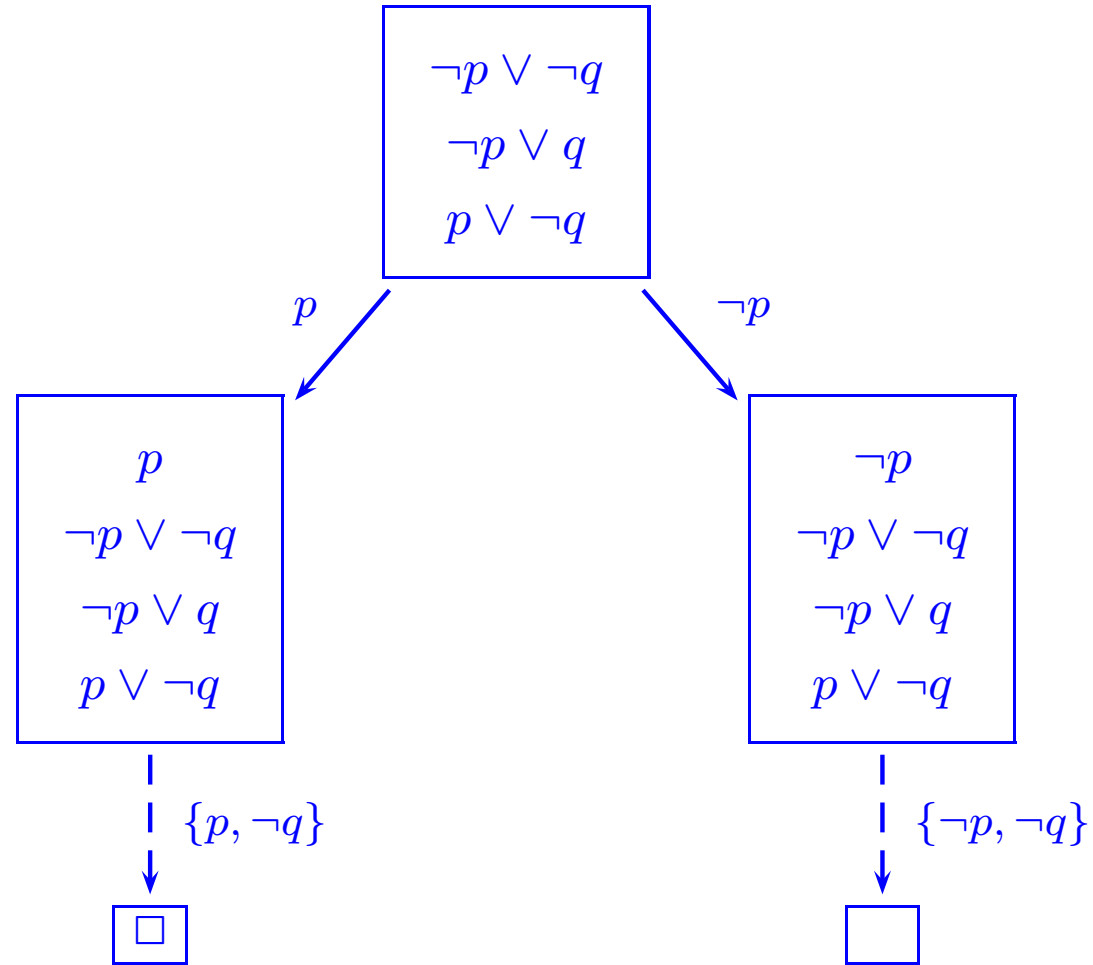
DLL = splitting + unit propagation

```
procedure DLL(S)  
input: set of clauses S  
output: satisfiable or unsatisfiable  
parameters: function select_literal  
begin  
  S := propagate(S)  
  if S is empty then return satisfiable  
  if S contains  $\square$  then return unsatisfiable  
  L := select_literal(S)  
  if DLL(S  $\cup$  {L}) = satisfiable  
    then return satisfiable  
    else return DLL(S  $\cup$  { $\tilde{L}$ })  
end
```

DLL. Example 1



DLL. Example 2



Model: $\{p \mapsto 0, q \mapsto 0\}$.

Two optimisations

Tautologies are clauses $p \vee \neg p \vee C$.

A literal L in S is called **pure** if S contains no clauses of the form $\tilde{L} \vee C$.

Tautologies and clauses with pure literals **can be removed**.

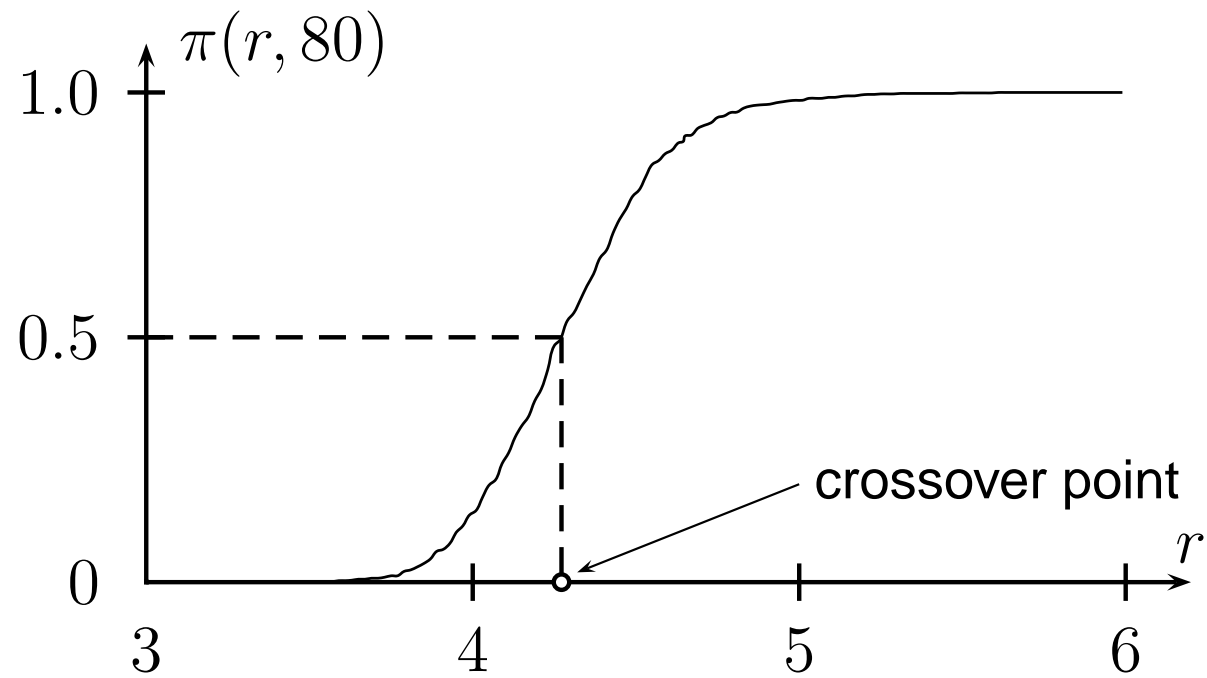
Random Clause Generation

Fix:

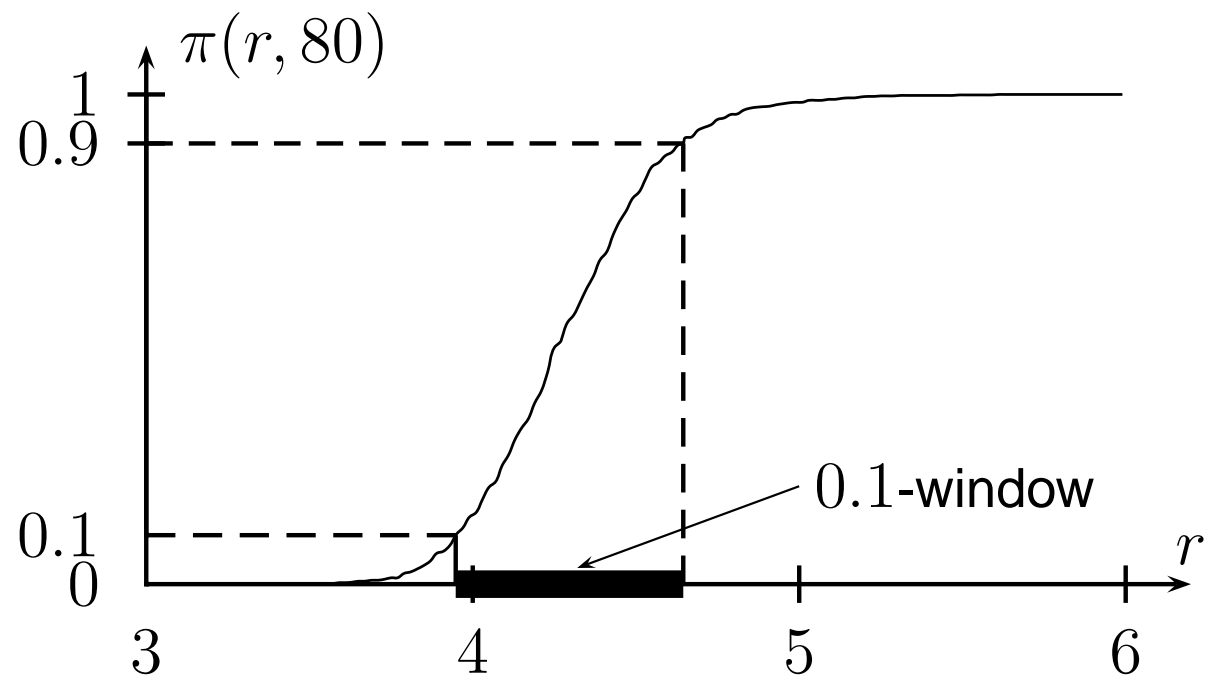
- ▶ Number k of **literals per clause**, so we will generate k -SAT instances;
- ▶ Number n of boolean variables;
- ▶ Real number r : **ratio of clauses per variable**.

Generate $[rn]$ clauses, each one has k literals randomly generated among $p_1, \dots, p_n, \neg p_1, \dots, \neg p_n$.

Probability of obtaining an unsatisfiable set

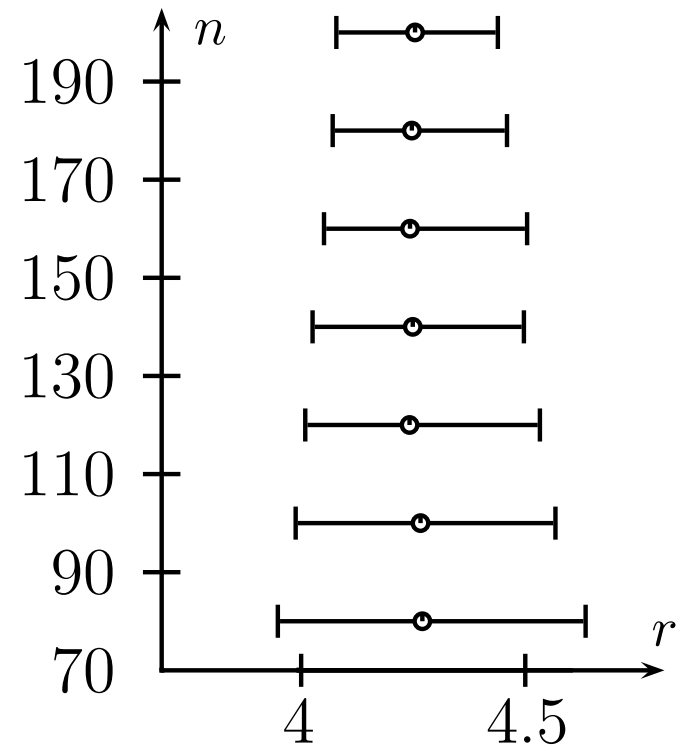
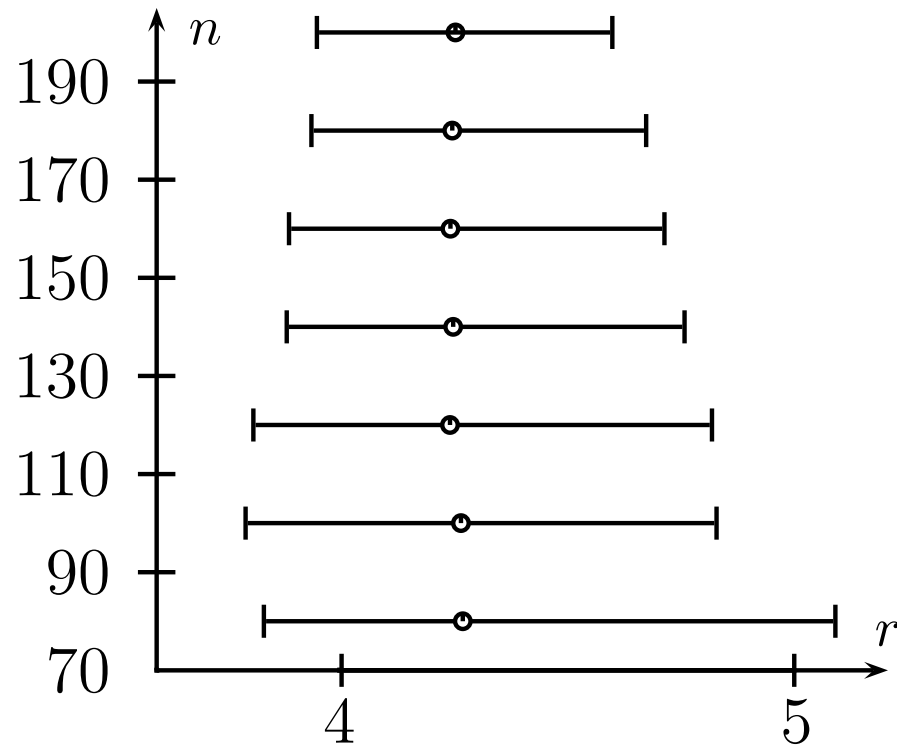


ϵ -window

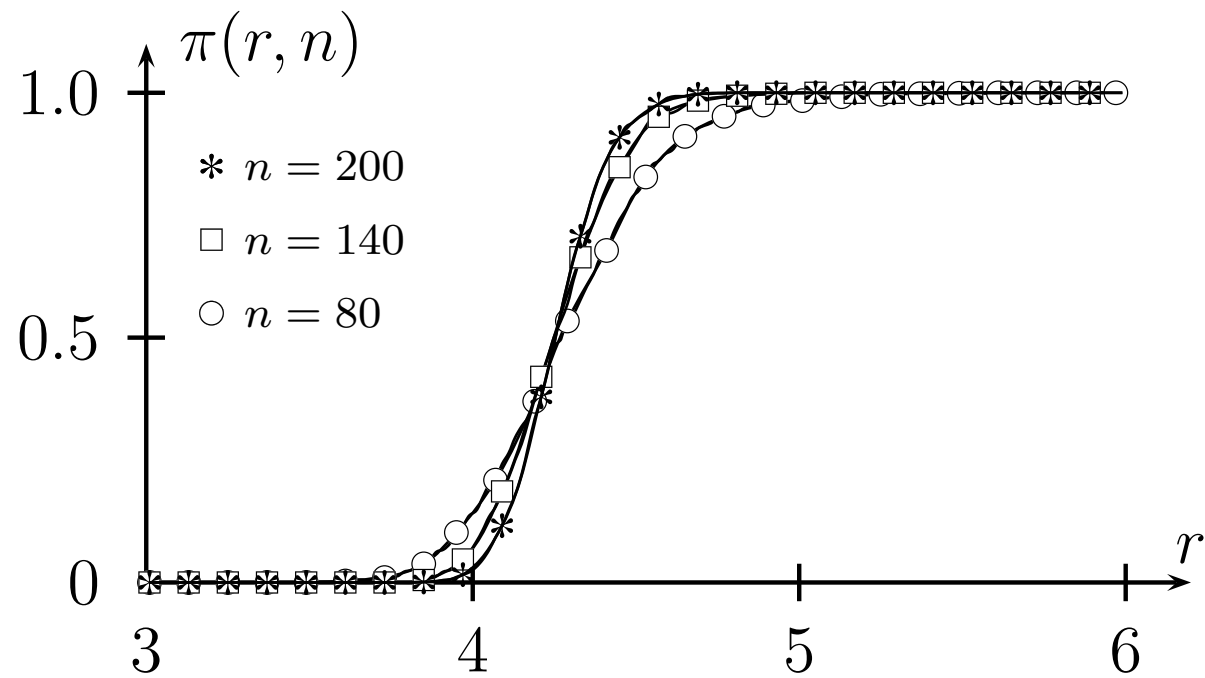


Scaling Window Effect

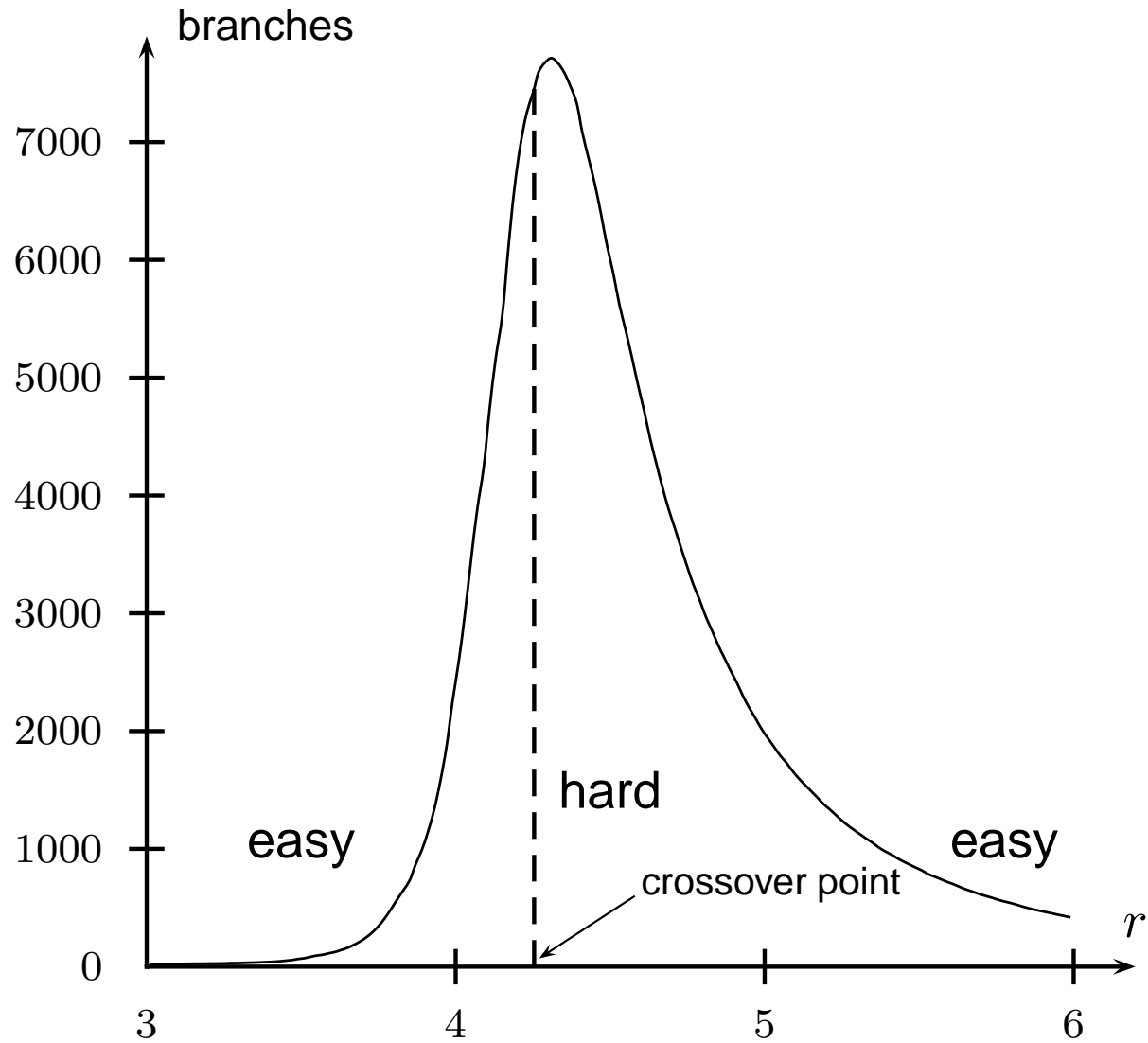
0.01-window and 0.1-window:



Sharp Phase Transition



Easy-Hard-Easy Pattern



Satisfiability Algorithm that Cannot Establish Unsatisfiability

procedure *CHAOS*(*S*)

input: set of clauses *S*

output: interpretation *I* such that $I \models S$ or *don't know*

parameters: positive integer *MAX-TRIES*

begin

repeat *MAX-TRIES* times

I := random interpretation

if $I \models S$ then return *I*

return *don't know*

end

Satisfiability has **short witnesses**: interpretations.

Randomised Algorithms for SAT

- ▶ Choose a **random interpretation**.
- ▶ If this interpretation is not a model, repeatedly choose a variable and **change its value in the interpretation** (**flip** the variable).

The flipped variables are chosen using heuristics or randomly, or both.

Flipping a Variable

$$\mathit{flip}(I, p)(q) = \begin{cases} I(q), & \text{if } p \neq q; \\ 1, & \text{if } p = q \text{ and } I(p) = 0; \\ 0, & \text{if } p = q \text{ and } I(p) = 1. \end{cases}$$

In other words, the interpretation $\mathit{flip}(I, p)$ is obtained from I by changing its value on p .

GSAT

procedure $GSAT(S)$

input: set of clauses S

output: interpretation I such that $I \models S$ or *don't know*

parameters: integers $MAX-TRIES$, $MAX-FLIPS$

begin

repeat $MAX-TRIES$ times

$I :=$ random interpretation

if $I \models S$ then return I

repeat $MAX-FLIPS$ times

$p :=$ an atom such that $flip(I, p)$ satisfies

the maximal number of clauses in S

$I = flip(I, p)$

if $I \models S$ then return I

return *don't know*

end

GSAT example

$p_1 \vee \neg p_2 \vee p_3, \neg p_2 \vee \neg p_3, \neg p_1 \vee \neg p_3, \neg p_1 \vee p_2, p_1 \vee p_2.$

flip	interpretation			satisfied clauses			candidates for flipping	flipped atom
no.	p_1	p_2	p_3	p_1	p_2	p_3		
1	0	0	1	4				

GSAT example

$p_1 \vee \neg p_2 \vee p_3, \quad \neg p_2 \vee \neg p_3, \quad \neg p_1 \vee \neg p_3, \quad \neg p_1 \vee p_2, \quad p_1 \vee p_2.$

flip	interpretation			satisfied clauses			candidates for flipping	flipped atom	
no.	p_1	p_2	p_3	p_1	p_2	p_3			
1	0	0	1	4	3	4	4	p_2, p_3	p_2

GSAT example

$p_1 \vee \neg p_2 \vee p_3, \quad \neg p_2 \vee \neg p_3, \quad \neg p_1 \vee \neg p_3, \quad \neg p_1 \vee p_2, \quad p_1 \vee p_2.$

flip no.	interpretation			satisfied clauses			candidates for flipping	flipped atom	
	p_1	p_2	p_3	p_1	p_2	p_3			
1	0	0	1	4	3	4	4	p_2, p_3	p_2
2	0	1	1	4					

GSAT example

$p_1 \vee \neg p_2 \vee p_3, \quad \neg p_2 \vee \neg p_3, \quad \neg p_1 \vee \neg p_3, \quad \neg p_1 \vee p_2, \quad p_1 \vee p_2.$

flip no.	interpretation			satisfied clauses				candidates for flipping	flipped atom
	p_1	p_2	p_3	p_1	p_2	p_3			
1	0	0	1	4	3	4	4	p_2, p_3	p_2
2	0	1	1	4	3	4	4	p_2, p_3	p_3

GSAT example

$p_1 \vee \neg p_2 \vee p_3, \neg p_2 \vee \neg p_3, \neg p_1 \vee \neg p_3, \neg p_1 \vee p_2, p_1 \vee p_2.$

flip no.	interpretation			satisfied clauses			candidates for flipping	flipped atom	
	p_1	p_2	p_3		p_1	p_2			p_3
1	0	0	1	4	3	4	4	p_2, p_3	p_2
2	0	1	1	4	3	4	4	p_2, p_3	p_3
3	0	1	0	4					

GSAT example

$p_1 \vee \neg p_2 \vee p_3, \quad \neg p_2 \vee \neg p_3, \quad \neg p_1 \vee \neg p_3, \quad \neg p_1 \vee p_2, \quad p_1 \vee p_2.$

flip no.	interpretation			satisfied clauses				candidates for flipping	flipped atom
	p_1	p_2	p_3		p_1	p_2	p_3		
1	0	0	1	4	3	4	4	p_2, p_3	p_2
2	0	1	1	4	3	4	4	p_2, p_3	p_3
3	0	1	0	4	5	4	4	p_1	p_1

GSAT example

$p_1 \vee \neg p_2 \vee p_3, \neg p_2 \vee \neg p_3, \neg p_1 \vee \neg p_3, \neg p_1 \vee p_2, p_1 \vee p_2.$

flip no.	interpretation			satisfied clauses				candidates for flipping	flipped atom
	p_1	p_2	p_3		p_1	p_2	p_3		
1	0	0	1	4	3	4	4	p_2, p_3	p_2
2	0	1	1	4	3	4	4	p_2, p_3	p_3
3	0	1	0	4	5	4	4	p_1	p_1
	1	1	0	5					