Exponential lower bounds for the running time of DPLL algorithms for SAT on satisfiable formulas

Dmitry Itsykson

St. Petersburg State University, St. Petersburg, Russia

Based on the paper: M. Alekhnovich, E. A. Hirsch, D. Itsykson,
Exponential lower bounds for the running time of DPLL algorithms on satisfiable formulas.

ECCC Report TR04-041. ftp://ftp.eccc.uni-trier.de/pub/eccc/reports/2004/TR04-041/Paper.ps

SAT

Given a Boolean Formula in CNF

like $F[x_1, x_2, \ldots, x_n] = (x_1 \vee x_2) \wedge (\overline{x_1} \vee x_3 \vee \overline{x_4}) \wedge \ldots$

decide whether $\exists b_1, b_2, \ldots, b_n \in \{0,1\}$

such that $F[x_1 \leftarrow b_1, x_2 \leftarrow b_2, \ldots, x_n \leftarrow b_n] = 1$

$\mathcal{P} = \mathcal{NP}$ $\Leftrightarrow$ Exists polynomial-time algorithm for SAT

$\mathcal{NP} = \text{co-}\mathcal{NP}$ $\Leftrightarrow$ All unsatisfiable formulas have short refutations

Known results: exponential lower bounds for specific proof systems

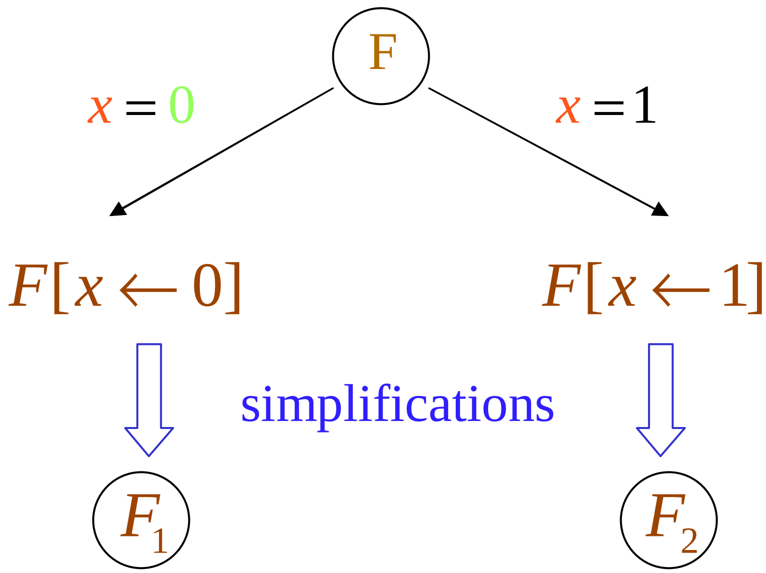
Motivation

- most classical algorithms originating from Davis, Putnam, Logemann, Loveland, 1960-62,
- almost all deterministic SAT solvers use DPLL
- satisfiable formulas are much easier for solvers
- known lower bounds for resolution imply bounds for DPLL on unsatisfiable formulas only.

GOAL: exponential lower bound for satisfiable formulas

DPLL – general scheme

Divide and conquer:



Specific algorithm:

- Heuristic A: chose variable x,
- Heuristic B: choose brunch to be examined first,
- Simplification rules.

Dead ends: trivial formulas.

Examples

A: choose the most frequent variable,

choose a variable from the shortest clause, …

B: choose the most frequent sign, …

Simplifications:

unit clause elimination: $x \wedge G \quad \Longrightarrow \quad G[x \leftarrow 1]$

pure literal rule: $\bar{x}$ does not appear in $F \Rightarrow$

$$F \quad \Longrightarrow \quad F[x \leftarrow 1]$$

Known facts

- exponential lower bounds for resolution refutations of unsatisfiable formulas translate to DPLL [Tseytin, 1968],…,[Puldak, Impagliazzo, 2000]
- exponential lower bounds on satisfiable formulas for specific DPLL algorithms:
  [Nikolenko, 2002]: Greedy+Unit Clause+Randomization
  [Achilioptas, Beame, Molloy, 2003]: Greedy+Unit Clause, Ordered DLL (conditional bounds)
  [Achilioptas, Beame, Molloy, 2004]: Ordered DLL: exponential time with constant probability

Ultimate goal: Every DPLL algorithm takes exponential time with prob. $1-2^{-\Omega(n)}$ on satisfiable $F_1, F_2, \ldots, F_n, \ldots$

Generalized myopic algorithms

A, B : read $n^{1-\varepsilon}$ clauses,

read other clauses without negations,

query the number of occurrence of a literal

Simplifications: unit clauses, pure literals.

Drunk algorithms

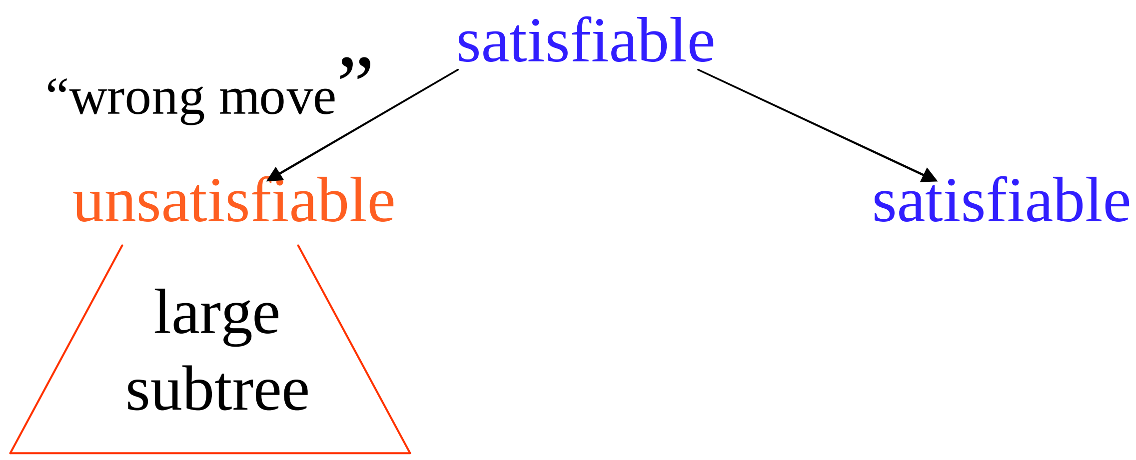
A : any !

B : random 50 : 50 .

Simplifications: unit clauses, pure literals, subsumption.

Theorem: $\exists$ sequence of (polynomial-size) satisfiables formulas F_n (*resp.*, G_n) such that $\forall$ polynomial-time randomize generalized myopic (*resp.*, *drunk*) algorithm errs with probability $1-2^{-\Omega(n)}$.

Proof strategy: show that w.h.p. a DPLL algorithm obtain a hard unsatisfiable formula:



Construction for drunk algorithms

Take any hard unsatisfiable F

$F[x_i \leftarrow 0]$ remains hard

$(F \vee x_1) \wedge (F \vee x_2) \wedge \ldots \wedge (F \vee x_n) = G$

first move is wrong with probability $\frac{1}{2}$.

Take n renamed copies

$$G^{(1)} \wedge G^{(2)} \wedge \ldots \wedge G^{(n)}$$

$\Rightarrow$ wrong move with probability $1 - \frac{1}{2^n}$.

Construction for generalized myopic algorithms

1) construct $\mathbf{n}\times\mathbf{n}$ 0/1- matrix **A** that
   - is non-degenerate,
   - has 3 nonzero entries per row,
   - has certain expansion properties.

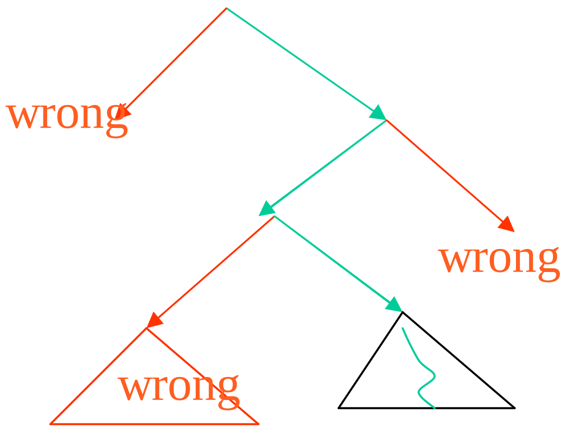
   (solution: take a larger matrix at random, select n linearly independent rows)

2) take 0/1 vector **b** at random

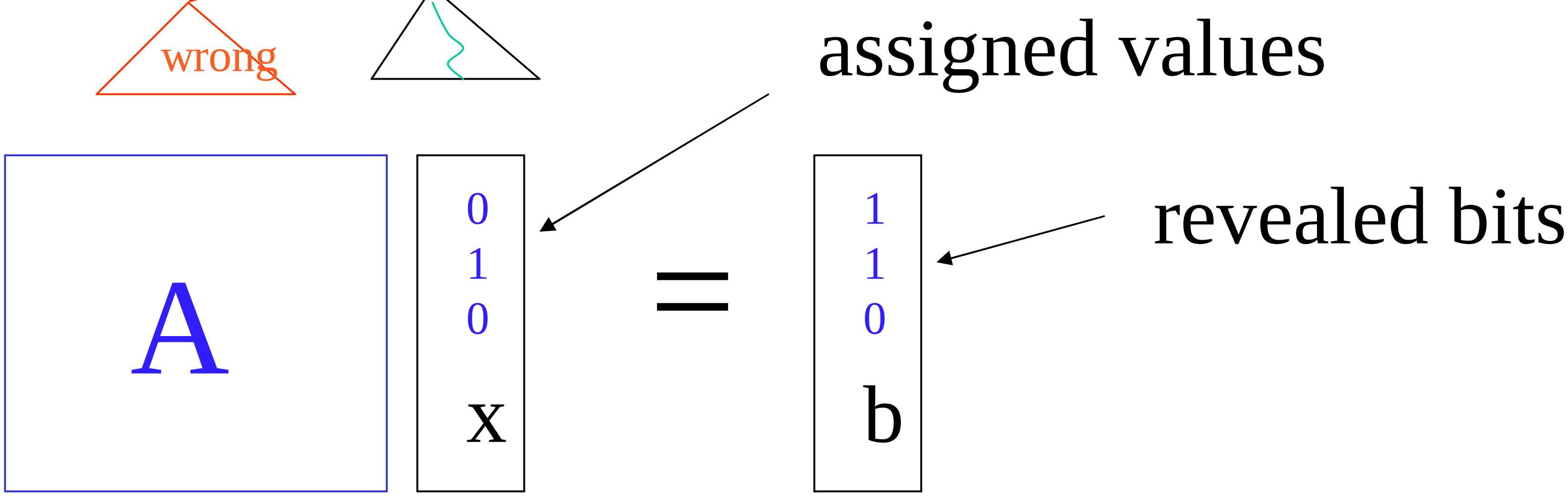
3) convert **Ax**=**b** into 3-CNF (with unique satisfying assignment)

$$x+y+z=1 \Leftrightarrow (x\vee y\vee z)\wedge(\bar{x}\vee\bar{y}\vee z)\wedge(\bar{x}\vee y\vee\bar{z})\wedge(x\vee\bar{y}\vee\bar{z})$$

Myopic: idea of the proof



One wrong move
and you are lost!



If b and b' are similar solution may differ much.

Open question:

Generalize the model !

- myopic simplifications rule,
- oracle access to the input.