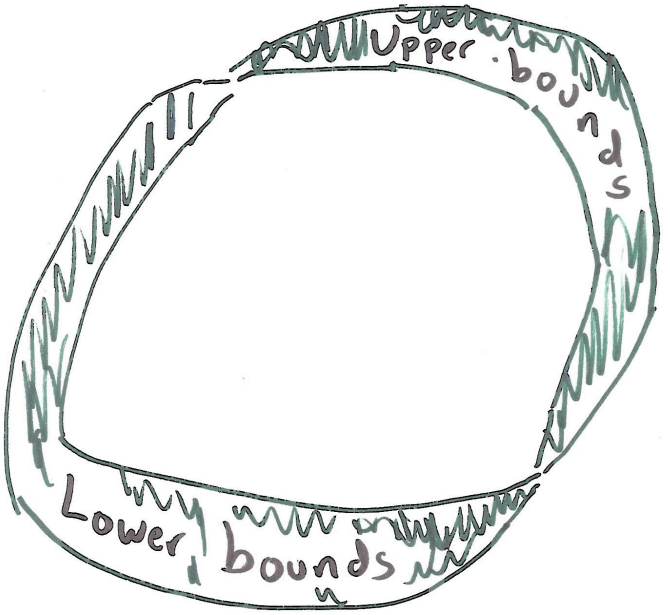
A Satisfiability Algorithm for AC^0



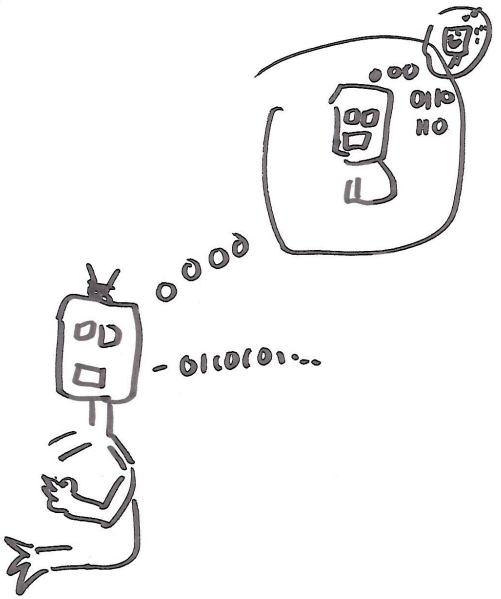
Russell Impagliazzo,
IAS + UCSD

William Matthews,
Google

Ramamohan Paturi,
UCSD

The proper study of
Mankind is Man

Alexander Pope



Is the proper study

of computers

computation?

Meta-computation:

Algorithmic problems where

the input is a description

of a computational device

or algorithm

Meta-computation problems

- Circuit SAT
  - Instance: Boolean Circuit C
  - Problem: Does there exist an x, C(x)=1?
- Circuit #-SAT
  - Instance: Boolean Circuit C
  - Problem: Count the number of x, C(x)=1

Meta-computing in derandomization

- Derandomizing PromiseBPP

  Instance: Boolean Circuit C

  Problem: Estimate Prob_x C(x)=1 to within small additive error
- Derandomize Polynomial Identity Testing

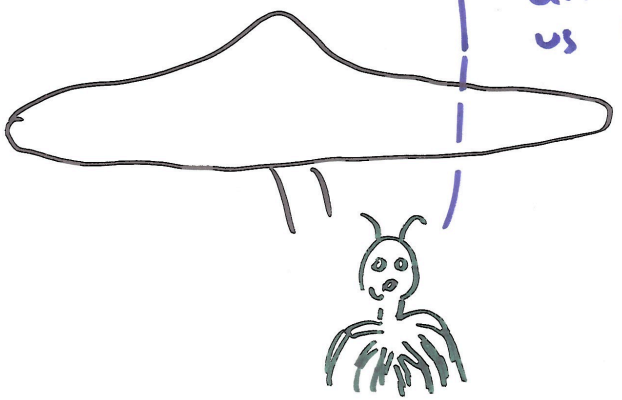
  Instance: Algebraic Circuit A, A(x) non-zero

  Problem: Find x so that $A(x) \neq 0$

Learning as meta-computing

- Instance: Black-box access to a function f computed by a circuit in concept class C, distribution D on inputs to C
- Problem: Find a description of a function f’ close to f on D

We come with a warning.
Your planet has serious problems
that must be dealt with.
It will require dedication
and intelligence. Take
us to your leaders!



Umm... You don't
know us very
well, do you?

To solve human problems,
you need to understand humans.
In particular, you need to
know what's not human.

“Zane’s Thesis”

- To solve meta-computing problems efficiently, we need to understand the computational model of possible inputs. In particular, we need to be able to prove lower bounds for that model.

Formalizations of Zane's Thesis

- IKW : Derandomizing PromiseBPP implies NEXP is not contained in P/poly
- KI: Derandomizing PIT implies that either NEXP is not contained in P/poly or Perm is not in AlgP/poly
- FK: Deterministic Learning implies new circuit lower bounds

Williams' Theorem

- Williams '10, '11

  If Circuit-SAT is solvable in co-non-deterministic time that is less than 2^n by a super-polynomial factor, then NEXP is not in P/poly

Why this is so great!

- Even minute savings over exhaustive search prove lower bounds!
- Can be translated down to smaller classes of circuits, giving a general upper bound-lower bound duality!
- Actually was used to prove new lower bounds!

  Williams ‘11 proved NEXP not in ACC_0 by giving non trivial ACC_0-SAT algorithm.

Translating down

- Assume C-SAT is in co-non-deterministic time T(n), for circuit class C.
- If P is not contained in C, we have a lower bound already. So assume P is contained in C
- C-TAUT is in non-deterministic time T, i.e., small proofs of C-tautologies.

- We combine these to give size T*poly(n) proofs of arbitrary tautologies

Proving arbitrary tautologies

- Let A be an arbitrary circuit, expressing a tautology
- For each gate in A, g_i, it is defined as g_i=op_i(g_j,g_k) , and output gate is g_out
- For each gate i, proof contains B_i, a C-circuit equivalent to the sub-circuit at g_i
- For each gate, prove the C-tautology B_i=pp_i(B_j,B_k)
- Prove the C-tautology B_out
- Greater Expressive Power of a class means harder SAT problem

Structure of SAT-problems

- For which classes of circuits C do improved C-SAT algorithms exist?
- How much improvement over exhaustive search is possible?
- How does the power of C-circuits affect the complexity of C-SAT?

Better SAT algorithms would be useful

- SAT-solvers used in practice, do surprisingly well for many applications
- But really only work well on k-CNFs for small k
- Reductions to k-SAT can blow up number of variables hugely
- Performance very sensitive to details of reduction

Do ``higher order'' SAT heuristics work well?

- Can we directly solve SAT for more complex formulas without reducing to k-SAT?
- Can we get similar savings for more sophisticated goals, like counting number of satisfying assignments? (Bayesian reasoning)

Generic cryptanalysis

- Block ciphers are designed somewhat ad hoc
- Ideal efficiency, for hardware implementation
  - Small number of logic gates, e..g, linear
  - Time is parallel time, so could be constant time=constant depth
  - Key size chosen to be as small as possible, so want best algorithm for breaking code to be close to exhaustive search

Can we show these goals are incompatible?

- Known message attack: Solve E_K (M)= C for K given M, C
- Reduces to SAT for whatever circuits compute E, with n = key size
- So better than exhaustive search algorithm for

  SAT of linear-size, constant depth formulas means generic cryptanalysis of ``overly optimistic'' ciphers.

How to compare SAT algorithms

- Improved algorithm:

  Runs in time 2^{n(1-s(n,m))} poly (m),

  where m is the size of the circuit, n the number of variables
- s(n,m) = ``savings'', bigger better
- s(n,m)=0 : exhaustive search
- s(n,m)=1 : polynomial-time algorithm
- Sometimes, only save for small m

For which classes of circuits are improved SAT algorithms known?

- k-CNF's

  k-SAT:

  s(n,m) = c/k for constant c.

  Many algorithms achieve this: PPZ, Schoning, PPSZ, this work, etc.

- k-#SAT: s(n,m) = exp (-k) [ILP]

  s(n,m) = c/k (this work)

CNF SAT

- SAT: s(n,m) = O(1/log (m/n)) [Shuler]
- #SAT s(n,m) = O(1/log (m/n)) [This work, Shuler?]

Constant depth (AC_0) circuits

- AC_0-SAT :

  CIP : s(n,m) = 1/(m/n)^{2^d} (Only good when m linear in n)

  Williams : s(n,m)= 1/n^{1-2^-d}

  BIS : s(n,m)=1/n^a, any a > 0, m quasi-polynomial

Our main result

- Zero-error probabilistic AC_0-SAT and #-SAT algorithms with

  s(n,m) = 1/(log (m/n))^{d-1}

- Matches best bounds for CNF-SAT
- Any improvement shows NEXP not contained in NC_1

Connection to lower bounds

- Uses standard lower bound technique: switching lemma
- Extends by proving joint switching lemma for families of CNF's
- Gives new circuit lower bound:

  Correlation with parity at most 2^{-n/(log (m/n))^{d-1}} (Improves on Ajtai, BIS 2^{-n^{1-a}}, independently discovered by Hastad 11)

Hastad Switching Lemma

- Let C be a k-CNF. Let rho be a restriction that assigns variables * with probability p, and otherwise assigns them independent random bits. Then the probability that the canonical decision tree for C|rho has depth at least h

  is at most (cpk)^h for some absolute constant c.