CS 810: Complexity Theory	3/21/2007
Lecture 22: Proof Complexity	
Instructor: Dieter van Melkebeek	Scribe: Nathan Collins

In the last lecture we discussed quantum computation. Today we start the second part of the course, where we consider non-standard complexity measures. The standard complexity measures are time and space complexity. Today's complexity measure is the size of the minimum proof of a given theorem.

1 Issues

There are some issues to be resolved

- We all know intuitively what a proof is,[1] but we need a precise definition if we hope to talk about the complexity of proofs. Today we formalize proofs using the familiar notion of NP: A proof is something that is "easily" verified, where "easy" means in time polynomial in the length of the proof. This is like NP, in that the proof is a witness or certificate, but unlike NP, we don't require that the proof be of size polynomial in the size of the theorem in question. In fact, the central goal of proof complexity is to establish the *non*-existence of polynomial size proofs of propositional tautologies.
- We want our proof systems to have two properties:
  - Soundness: No contradiction can be proved.
  - Completeness: All true theorems have a proof.
- We need to decide what sort of things we want to prove.[2] The more the statements under consideration are restricted the more simple the systems will be and (hopefully) the more likely we will be able to prove things about them. Proof complexity considers systems in which we can establish whether or not a logical proposition is a propositional tautology. Since TAUT, the class of propositional tautologies, is co-NP complete, we can answer the NP vs co-NP question by understanding how "difficult" TAUT is. The consensus is that TAUT does not have poly size proofs, although this has only been proven in a few specific proof systems.

2 Definition and Central Theorem

Definition 1 (Propositional proof system)**.** *A propositional proof system (PPS) is a poly-time mapping from $\{0,1\}^*$ onto TAUT, s.t. if $f(x) = y$ then $x \vdash y \in TAUT$, where "$\vdash$" means "proves."*

[1]One interesting definition is a repeatable experiment in persuasion.

[2]Gödel's Incompleteness Theorems say something like soundness and completeness can't be proved in systems strong enough to express arithmetic. Here we allow other means than the system itself for proving soundness though.

Intuitively, given a tautology y, a PPS ensures there is a proof x that is efficiently checkable by f proving $y \in$ TAUT. We are assuming that we have some encoding of proofs as strings and when x isn't a valid proof we'll map it to a trivial tautology, e.g. T.

Notice that soundness and completeness are implicit in the definition

- Soundness: Since $f(\{0,1\}^*) \subseteq$ TAUT.
- Completeness: Since TAUT $\subseteq f(\{0,1\}^*)$.

Definition 2 (Polynomial bounded PPS)**.** *A PPS is* polynomially bounded *if each $y \in TAUT$ has a proof that is polynomial in length, i.e. there exists a constant c s.t.*

$$(\forall y \in TAUT)(\exists x \in \{0,1\}^{\leq |y|^c}) f(x) = y.$$

With these definitions we can precisely state the theorem (mentioned in the last section) that relates proof complexity to the NP vs co-NP question.

Theorem 1. *There exists a polynomially bounded PPS iff* NP $=$ *co*-NP*.*

Proof. The proof is very easy

$\Rightarrow$ A polynomially bounded PPS gives a NP machine that decides TAUT. Namely, the machine guesses the proof x and verifies that it is a valid proof by computing $f(x)$ and checking that it equals y.

$\Leftarrow$ co-NP is $\leq_m^p$-complete for TAUT.

$\square$

So, proof complexity tries to establish NP $\neq$ co-NP by establishing super-polynomial bounds on the proof complexity of TAUT. So far this has consisted of establishing such bounds in a few particular example systems.

3 Proof Systems

Many systems can be cast more easily as *refutation systems.* Since $y \in$ SAT iff $\neg y \in$ TAUT we think of a proof of membership in TAUT as refuting membership of the negation in SAT. DNFs are the natural form for TAUT, as CNFs are the natural form for SAT formulas. Without loss of generality, we focus on refuting CNFs – proving that CNF formulas are not in SAT. There is a general refutation strategy. To refute a formula φ we assume it is true and derive a contradiction. Derivations start from axioms and the assumption that φ is true, and then proceed using a set of sound derivation rules to arrive at a contradiction. The axioms and derivation rules are part of the proof system being used.

The most common refutation systems, and the ones we'll consider here are

- Logic-based: Manipulate boolean formulas.
- Geometry-based: Manipulate linear inequalities over $\mathbb{R}$.
- Algebra-based: Manipulate multivariate polynomials.

3.1 Logic-Based

The most trivial logic-based proof system just uses truth tables for proofs. Proofs in this system are clearly of exponential size. A more interesting class of examples are *Frege systems*, which are defined by a collection of axioms including

- Axioms in the form of easily recognizable tautologies.
- Derivation rules.

Example: A common axiom in Frege systems is the tautology $A \vee \overline{A}$. ⊠

Example: The *cut rule*

$$\frac{A \vee B \qquad \overline{A} \vee C}{B \vee C}$$

is a common derivation rule in Frege systems. The cut rule is read as "any assignment to the boolean variables that satisfies $A \vee B$ and $\overline{A} \vee C$ simultaneously satisfies $B \vee C$ too." Notice that the cut rule is intuitively valid. ⊠

A proof that a formula is not satisfiable in a logic based system is called a refutation.

Definition 3 (Refutation)**.** *A* refutation *of a CNF φ is a sequence of formulas, ending in the empty clause, s.t. each formula is a clause from φ, an axiom, or follows, via derivation, from some previously derived formulas in the sequence. The empty clause is the basic contradiction in Frege Systems.*

Frege systems are usually further classified based on the complexity of the formulas allowed in the proof. The following are possible restrictions

- Every line of the proof is a clause from φ.
- Constant-depth Frege: Every line of the proof is computable by a constant depth circuit.
- Unrestricted Frege: Lines are arbitrary formulas. Earlier in the course we noted that boolean formulas are equivalent to circuits with fanout 1.
- Named sub-formulas: Allowed to have "lemmas" that can be instantiated multiple times. This can significantly decrease proof size. This can be viewed as allowing each line of the proof to be computed by circuits of fanout ¿ 1. Generally, the size and depth of a circuit can be significantly decreased by allowing fanout greater than 1.

3.1.1 Resolution

A commonly studied proof system consists of using only the cut rule.

Definition 4 (Resolution)**.** *A Frege system without axioms in which all proof lines are clauses and the cut-rule is the only derivation is called a* resolution *system.*

Notice that soundness is immediate in a resolution system, since cut can be proved valid, and any proof of the empty clause $\emptyset$ must end with the derivation

$$\frac{x \vee \emptyset \qquad \overline{x} \vee \emptyset}{\emptyset \vee \emptyset = \emptyset},$$

justifying our taking $\emptyset$ to be the basic contradiction. Completeness is harder, since we need to be able to derive the empty clause from any unsatisfiable φ.

Completeness follows from the Davis-Putnam procedure. Here we give an example of the procedure on the unsatisfiable formula

$$\begin{aligned}\varphi =& (x \vee y) \wedge (x \vee z) \wedge (y \vee z)\\ &\wedge (\overline{x} \vee \overline{y}) \wedge (\overline{x} \vee \overline{z}) \wedge (\overline{y} \vee \overline{z}).\end{aligned}$$

The formula φ is not satisfiable since the first line needs two true variables and the second line needs two false variables.

The Davis-Putnam procedure generates a resolution refutation demonstrating the unsatisfiability of φ using a binary-tree search on variable assignments. The tree for φ is given in Figure 1. The internal nodes of the tree are labeled by variables and a path in the tree corresponds to an assignment to the variables labeling nodes on the path. The leaves of the tree are labeled by clauses that are violated by the assignment corresponding to the path that reaches them. Once

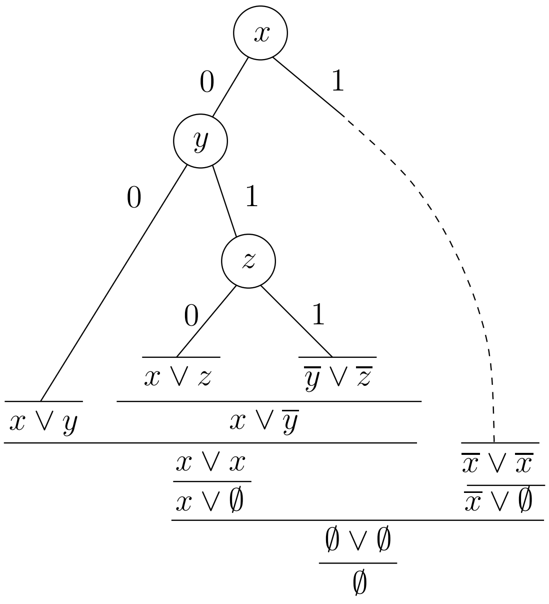

Figure 1: The right hand side of the tree, and corresponding derivations, are analogous to the left hand side and have been omitted. Notice that the derivations mirror the tree.

the tree has been constructed, a resolution refutation is constructed by working inward from the leaves, using the cut-rule on adjacent leaves to collapse the tree.

The Davis-Putnam procedure is used in many automated theorem provers, although it's easy to end up with exponential proofs when using Davis-Putnam to derive a resolution refutation.

3.2 Geometry-Based

The common geometric PPS is called *cutting planes*. A cutting planes system is a set of linear inequalities[3] with integer coefficients.

- Axioms: Unlike the Frege systems above, which had universal axioms, in cutting planes the axioms are specific to the formula in question. More precisely, if $\varphi = \bigwedge_i C_i$ is the CNF we are trying to refute, then there will be one axiom for each clause C_i in φ. For example, for the clause $x \vee \overline{y} \vee z$ we get the inequality $x + (1 - y) + z \geq 1$, which is satisfied by values x, y, and z in $\{0, 1\}$ only when $(x \geq 1) \vee (y \leq 0) \vee (z \geq 1)$.
- Derivation rules:
  - Addition of "same direction" inequalities:
$$\frac{f \geq a \qquad g \geq b}{f + g \geq a + b}$$
  - Multiplication by a positive integer:
$$\frac{f \geq a \qquad c \in \mathbb{N}}{cf \geq ca}$$
  - Rounding:
$$\frac{(f = \sum_i a_i x_i) \geq a \qquad (\forall i) c \mid a_i}{\frac{f}{c} \geq \lceil \frac{a}{c} \rceil}$$
    Notice that rounding is *not* valid if we allow non integer values for the x_i.
- The basic contradiction is $0 \geq 1$.

Soundness is easy like with Frege systems, since the axioms are again "true." Completeness is again non-obvious, and we won't argue it here.

3.3 Algebra-Based

The *polynomial calculus* is a popular algebra-based PPS. In the polynomial calculus the lines in the proofs are multivariate polynomials. The polynomial calculus is based on the following theorem of Hilbert

Theorem 2 (Hilbert's Nullstellensatz)**.** *In any algebraically closed field* [4] *a set $P_1, \ldots, P_n$ of (multivariate) polynomials has no common root iff there exist another set $Q_1, \ldots, Q_n$ of polynomials s.t.*
$$\sum_i P_i Q_i \equiv 1.$$

[3] A linear inequality determines a hyper-half-plane that "cuts" space into two regions, one admissible and one not.
[4] All polynomials over the field have a root in the field. $\mathbb{C}$ is the most common example.

One direction of the proof is easy: If $\sum_i P_i Q_i \equiv 1$ then the P_i certainly don't share a root. The other direction is more difficult.

In the polynomial calculus we arithmetize clauses by producing polynomials that have roots corresponding to satisfying assignments for each clause. For example, for the clause $x \vee \overline{y} \vee z$ we considered above, we get the polynomial $(1-x)y(1-z)$. A refutation resolution consists of finding the polynomials Q_i. This accomplished by starting with polynomials corresponding to the formula, and using a series of multiplications of polynomials to arrive at the constant 1.

- Axioms: One polynomial $x(1-x)$ for each variable x, to force the allowable values taken by the variables to be zero and one.
- Derivation rules: If f and g are polynomials
  - If a and b are polynomials then
  $$\frac{f \qquad g}{af + bg}$$
  - If x is a variable then
  $$\frac{f}{xf}$$
- The constant polynomial 1 is the basic contradiction.

For the logical and geometric systems above the lines in the proofs are inherently short. In the polynomial calculus we can multiply polynomials though, which can lead to exponential size lines $(\Pi_i(x_i + y_i))$. In the polynomial calculus we let the degree be the measure of a polynomial's size.

4 Lower Bounds

We mentioned earlier that the goal of proof complexity is to establish super-polynomial lower bounds on establishing propositional tautologies. For the systems we have mentioned the best known lower bounds are

- Cutting planes: Exponential.
- Polynomial calculus: Linear degree.
- Unrestricted Frege: Unknown.
- Frege resolution: Exponential.

5 Unsatisfiable CNFs

The following families of unsatisfiable CNFs have proved useful in establishing lower bounds in propositional proof complexity.

5.1 Pigeonhole Principle

- PHP^m_n: The pigeonhole principle for m pigeons and n holes ($m > n$). The CNF has
  - Variables x_{ij} corresponding to the ith pigeon being assigned to the jth hole.
  - Clauses
    - Every pigeon assigned to a hole
$$\bigwedge_i \bigvee_j x_{ij}$$
    - No hole has two pigeons assigned to it
$$\bigwedge_k \bigwedge_{i \neq j} \overline{x}_{ik} \vee \overline{x}_{jk}$$

In the Frege resolution system PHP^{n+1}_n has size $2^{\Omega(n)}$ proofs. When $m = n + 1$ the pigeon surplus is minimized and hence this is the case with the highest proof complexity. In general, for arbitrary $m > n$, PHP^m_n has proof complexity of $2^{\Omega(\sqrt[3]{n})}$ in Frege resolution.

In cutting planes, PHP has only polynomial proof complexity.

5.2 Tseitin Formulas

Given a graph $G = (V, E)$ we choose a "charge" function $\sigma : V \to \{0, 1\}$ on the vertices. Once σ has been chosen we ask if there exists a charge assignment γ for the edges s.t. the σ-charge of each vertex is equal to the parity of the γ-charges of the edges incident that vertex. In symbols, we are asking if, given σ,

$$(\exists \gamma : E \to \{0, 1\})(\forall v \in V)(\sigma(v) = \bigoplus_{e \in E \text{ s.t. } v \in e} \gamma(e))$$

Exercise 1. *Show that the sum of the vertex charges, $\sum_v \sigma(v)$, must be even if there exists a satisfactory choice γ of edge charges.*

The strongest lower bounds for Tseitin formulas arise when the graphs considered are expanders. Lower bounds of size $2^{\Omega(n)}$ for resolution are achieved by certain expander families.

6 Next Time

Next time we'll talk about more powerful proof systems that are allowed to use randomness.