CS 2429 - Propositional Proof Complexity

Lecture #6: 17 October 2002

Lecturer: Toniann Pitassi

Scribe Notes by: Shannon Dalmao

In the previous lecture, we explored a lower bound on the length of Resolution proofs for random formulas and discussed some open problems related to Resolution lower bounds. We ended with a brief introduction to automatizability and proof search for Resolution.

In this lecture, we will introduce a broader notion of automatizability that allows us to derive weaker forms of automatizability for resolution. We will also introduce the notion of an interpolant. Finally, we will show how interpolants can be used to exploit known lower bounds for circuit classes in order to bound the complexity of proof systems.

1 $f(n, s)$-Automatizability and Resolution

Recall the definition of automatizable from the previous lecture:

Definition A proof system P is *automatizable* if there is a polynomial-time algorithm that approximates MLP_P (Minimum Length Proof) to within a polynomial factor.

What follows is a generalization of this notion of automatizability.

Definition Let $f : \mathrm{N} \times \mathrm{N} \longrightarrow \mathrm{N}$ be a function. A propositional proof system, V, is $f(n, S)$-*automatizable* if and only if there is an algorithm A_v such that given any unsatisfiable formula x, with $|x| = n$, A_v outputs a proof P (in V) in time at most $f(n, S)$ where S is the size of the shortest V-proof for x.

Note that our original definition of automatizable can be framed in this light by requiring that f be $n^{O(1)}S^{O(1)}$ (i.e. f is polynomial). Using this new notion of $f(n, S)$-automatizability, we now derive a couple of weak forms of automatizability for Resolution.

Theorem 1 *For kCNF formulas, Tree Resolution is $S^{O(\log n)}$-automatizable.*

Proof (sketch) Recall Theorem 3 from lecture #4 that states that every tree-like resolution proof of F of size S can be converted to one of width $\lceil \log_2 S \rceil + width(F)$. There are only $2^{\log S} \begin{pmatrix} n \\ \log S \end{pmatrix} = n^{O(\log S)} = S^{O(\log n)}$ clauses of size at most $\log S$. By Theorem 3 from lecture #4, we can convert these clauses into a tree proof. If we do not care about space, then we can run a breadth-first resolution only deriving clauses of width at most $\log S$. Alternatively, space requirements can be reduced to polynomial by making the search recursive.

Theorem 2 *For kCNF formulas, Resolution is* $2^{O(\sqrt{n \log S} \cdot \log n)}$ *automatizable.*

Proof The proof is similar to that of the previous theorem however it uses Theorem 5 from lecture #4 which states that every resolution proof of F of size S can be converted to one of width $\sqrt{2n \cdot \ln S} + width(F)$.

The fact that the above theorems hold for kCNF formulas is not much of a restriction since any formula can be converted into an equivalent 3CNF formula, by Cook's theorem. There are similar results for Polynomial Calculus with Resolution to those derived above for Resolution. Moreover, very weak systems such as truth tables are trivially automatizable. We conclude this section by noting, however, that there are few other known positive results on automatizability.

2 Interpolation

Although we present the following results in the context of propositional logic, the concept of interpolation was originally defined for First Order Logic. Let $A(\mathbf{p}, \mathbf{q})$ denote a formula over the vectors of variables $\mathbf{p}$ and $\mathbf{q}$. Similarly, let $B(\mathbf{p}, \mathbf{r})$ denote a formula over the vectors of variables $\mathbf{p}$ and $\mathbf{r}$. Finally, let $\mathbf{q} \cap \mathbf{r} = \phi$.

Definition If $A(\mathbf{p}, \mathbf{q}) \longrightarrow B(\mathbf{p}, \mathbf{r})$ is a tautology, then a *Craig interpolant* is any function C such that for any truth assignment α to $\mathbf{p}$,

1. $C(\alpha) = 0$ implies that $\neg A(\mathbf{p}, \mathbf{q})$ is a tautolgy, and
2. $C(\alpha) = 1$ implies that $B(\mathbf{p}, \mathbf{r})$ is a tautolgy.

The origin of the term interpolant is obviated when one notices that $A(\mathbf{p}, \mathbf{q}) \longrightarrow C(\mathbf{p})$ and $C(\mathbf{p}) \longrightarrow B(\mathbf{p}, \mathbf{r})$. There is also a dual definition of an interpolant for unsatisfiable CNF formulas, $A(\mathbf{p}, \mathbf{q}) \wedge B(\mathbf{p}, \mathbf{r})$, that says which one of $A(\mathbf{p}, \mathbf{q})$ and $B(\mathbf{p}, \mathbf{r})$ is unsatisfiable.

Definition If $A(\mathbf{p}, \mathbf{q}) \wedge B(\mathbf{p}, \mathbf{r})$ is an unsatisfiable CNF formula, then a *Craig interpolant* is any function C such that for any truth assignment α to $\mathbf{p}$,

1. $C(\alpha) = 0$ implies that $A(\mathbf{p}, \mathbf{q})$ is unsatisfiable, and
2. $C(\alpha) = 1$ implies that $B(\mathbf{p}, \mathbf{r})$ is unsatisfiable.

Theorem 3 *If for every unsatisfiable formula* $A(\boldsymbol{p}, \boldsymbol{q}) \wedge B(\boldsymbol{p}, \boldsymbol{r})$ *there exists a polynomial time computable interpolant, then* $NP \cap CoNP \subseteq P/poly$.

Proof (sketch) Fix some language $L \in NP \cap CoNP$. For each fixed length n, let $A(\mathbf{p}, \mathbf{q})$ code "$\mathbf{q}$ is a witness that $\mathbf{p}$ is in L," where p has length n, And let $B(\mathbf{p}, \mathbf{r})$ code "$\mathbf{r}$ is a witness that $\mathbf{p}$ is not in L". Such formulas A and B exist since L is in $NP \cap CoNP$. Now if there is a polynomial-time interpolant for $A(\mathbf{p}, \mathbf{q}) \wedge B(\mathbf{p}, \mathbf{r})$, then by definition, there is a polynomial-size circuit C that takes as input α, and determines whether or not $A(\alpha, \mathbf{q})$ is satisfiable. But $A(\alpha, \mathbf{q})$ is satisfiable if and only if α is in L, and thus this circuit circuit decides L on inputs of length n, and therefore L is in $P/poly$.

The above theorem implies that we cannot, in general, expect that an interpolant for a formula $A(\mathbf{p},\mathbf{q}) \wedge B(\mathbf{p},\mathbf{r})$ is going to be small. If however, $A(\mathbf{p},\mathbf{q}) \wedge B(\mathbf{p},\mathbf{r})$ is unsatisfiable and has a short resolution proof, then we can come up with a polynomial size circuit for computing a Craig interpolant.

2.1 Interpolation and Lower Bounds for Circuit Classes

Restricting various characteristics of the circuits that compute interpolants, such as size and monotonicity, allows us to refine the notion of interpolation in order to exploit the lower bounds that have been proven for various circuit classes. That is, if we have a proof system whose interpolants are in such a circuit class, then we can try to build a formula whose interpolant will be a circuit for a hard problem in the circuit class.

Definition Let V be a propositional proof system and let $f : \mathrm{N} \longrightarrow \mathrm{N}$ be any function. Then V has *f-interpolation* if and only if given an unsatisfiable formula $A(\mathbf{p},\mathbf{q}) \wedge B(\mathbf{p},\mathbf{r})$ with minimum proof size S in V, there exists a circuit of size at most $f(S)$ computing an interpolant C for $A(\mathbf{p},\mathbf{q}) \wedge B(\mathbf{p},\mathbf{r})$.

We say that V has *feasible interpolation* whenever f is polynomial. V has *monotone f-interpolation* if and only if whenever the variables $\mathbf{p}$ occur only positively in A (or only negatively in B) the circuit computing an interpolant is monotone. (Note that a monotone boolean function is a boolean function in which, if you flip any of the inputs in $\mathbf{p}$ from 0 to 1, the value of the function will not flip from 1 to 0. A monotone circuit on n boolean inputs is a circuit with AND and OR gates only, and no negations. Families of monotone circuits, one for each input length n, compute exactly the monotone boolean functions.)

Theorem 4 *If proof system V has feasible interpolation and $NP \not\subset P/poly$ then V is not polynomially bounded.*

Proof (sketch) Suppose, for the sake of contradiction, that V has feasible interpolation and is polynomially bounded (i.e. poly$(|x|)$) with bound p. Consider a formula $A(\mathbf{p},\mathbf{q}) \wedge B(\mathbf{p},\mathbf{r})$ where $\mathbf{p}$ codes a CNF formula, $A(\mathbf{p},\mathbf{q})$ says that assignment $\mathbf{q}$ satisfies $\mathbf{p}$, and $B(\mathbf{p},\mathbf{r})$ says that $\mathbf{r}$ of length $p(|x|)$ codes a V-proof of the unsatisfiability of $\mathbf{p}$.

A Feasible interpolant for this formula gives a polynomial size circuit that, for each CNF formula (encoded by $\mathbf{p}$), tells which of $A(\mathbf{p},\mathbf{q})$ and $B(\mathbf{p},\mathbf{r})$ is unsatisfiable. That is, we have a polynomial size circuit for deciding satisfiability. This implies that $NP \subset P/poly$. Note that the inequality is strict because P/poly is known to contain undecidable languages that are not in NP.

In the case that the proof system has monotone feasible interpolation, we can do better than the above theorem by using cliques and co-cliques.

Theorem 5 *If V has monotone feasible interpolation, then V is not polynomially bounded.*

Proof

In what follows, we will derive a contradiction to the Razborov/Alon-Boppana Boolean circuit lower bounds for the k-clique problem:

Theorem 6 (Razborov, Alon-Boppana) *There exists an ε such that for sufficiently large n, and $m = \frac{n}{10}$, any monotone circuit which outputs a 1 on all m-cliques, and a 0 on all $(m-1)$co-cliques requires size 2^{n^ε}.*

Consider the formula $A(\mathbf{p},\mathbf{q}) \wedge B(\mathbf{p},\mathbf{r})$ where $\mathbf{p}$ encodes an undirected graph G over n vertices, $A(\mathbf{p},\mathbf{q})$ says that G contains a clique of size m, and $B(\mathbf{p},\mathbf{r})$ says that G is a co-clique of size (m-1). Recall that an m-clique is a set of m fully connected vertices. Recall also that an (m-1) co-clique is a graph whose vertices are partitioned into m-1 groups. An edge in the (m-1) co-clique connects any pair of vertices if and only if the two vertices are contained in different groups of the partition. Below, we give the specifics of such an encoding.

- $\mathbf{p}$ is used to encode the undirected graph G over n vertices. For $1 \leq i < j \leq n$, variable $p_{i,j}$ is 1 iff (i,j) is an edge in G.
- $\mathbf{q}$ is used to describe a clique of size m in G. For $1 \leq i \leq m$ and $1 \leq j \leq n$, variable $q_{i,j}$ states that vertex j is the i^{th} vertex in the clique. The following clauses are necessary to state that G contains an m-clique.
  1. $q_{i,1} \vee \ldots, \vee q_{i,n}$ for $1 \leq i \leq m$. These clauses state that some vertex is the i^{th} vertex in the clique.
  2. $\overline{q_{i,j}} \vee \overline{q_{i',j}}$ for $i \neq i'$and $1 \leq j \leq n$. These clauses ensure that we do not place any vertex in the clique twice.
  3. $p_{i,j} \vee \overline{q_{i,j}} \vee \overline{q_{i',j'}}$ for $i \neq i'$and $1 \leq j < j' \leq n$. These clauses ensure that any two vertices in the clique are actually connected by an edge in G.
- $\mathbf{r}$ is used to state that G is an (m-1) co-clique. For $1 \leq i \leq n$ and $1 \leq j \leq m-1$, variable $r_{i,j}$ is 1 iff vertex i is in the j^{th} group of the partition. The following clauses are necessary to state that G is an $(m-1)$ co-clique.
  1. $r_{i,1} \vee \ldots \vee r_{i,m-1}$ for $1 \leq i \leq n$. These clauses ensure that every vertex gets placed in at least one group of the partition.
  2. $\overline{r_{i,j}} \vee \overline{r_{i',j}} \vee \overline{p_{i,i'}}$ for $1 \leq i < i' \leq n$ and $1 \leq j \leq m-1$. These clauses ensure that any two vertices in the same group are not connected by an edge in G.

Assume, for the sake of contradiction, that V has monotone feasible interpolation and V is polynomially bounded. Now consider the family of formulas $A(\mathbf{p},\mathbf{q}) \wedge B(\mathbf{p},\mathbf{r})_n$, for $n = 1 \ldots \infty$ with $m = \frac{n}{10}$. Since V is polynomially bounded, this family of formulas has polynomial size V-proofs. Moreover, since the variables $p_{i,j}$ appear only positively in A (and only negatively in B) and V has a monotone feasible interpolant, we have that for all n, there is a monotone circuit computing an interpolant for $A(\mathbf{p},\mathbf{q}) \wedge B(\mathbf{p},\mathbf{r})_n$. This monotone circuit in particular outputs 0 on all (m-1) co-cliques, and 1 on all m-cliques, thus contradicting the Razborov/Alon-Boppana theorem.