Scribe Notes

Proof Complexity

Math 267A - UCSD, Winter 2002

Instructor: Sam Buss

Contents

Math 267a - Propositional Proof Complexity

Lecture #1: 14 January 2002

Lecturer: Sam Buss

Scribe Notes by: Robert Ellis

1 Introduction to Propositional Logic

1.1 Symbols and Definitions

The language of propositional logic consists of connectives, propositional variables, parentheses and formulas, shown in the table below. The meanings of the symbols are given by their corresponding truth tables. The $\lor$ ("logical or"), $\land$ ("logical and") and $\rightarrow$ ("logical implication") operators are binary, while the $\neg$ ("logical negation") operator is unary. The truth tables for these operators are constructed by exhaustively assigning all possible truth values to the variables and listing the result defined by the operator.

connectives	$\lor, \land, \neg, \rightarrow$
variables	$x_1, x_2, \ldots$
parentheses	(,)
formulas	$(\phi \land \psi)$, $(\phi \lor \psi)$, $(\phi \rightarrow \psi)$, $(\neg\phi)$, etc.

Table 1: Components of Statements of Propositional Logic

x_1	x_2	$x_1 \lor x_2$
T	T	T
T	F	T
F	T	T
F	F	F

x_1	x_2	$x_1 \land x_2$
T	T	T
T	F	F
F	T	F
F	F	F

x_1	x_2	$x_1 \rightarrow x_2$
T	T	T
T	F	F
F	T	T
F	F	T

x_1	$\neg x_1$
T	F
F	T

A *truth assignment*

$$\tau : \{x_1, x_2, \ldots\} \rightarrow \{\mathrm{T}, \mathrm{F}\}$$

assigns logical truth values to the variables. A truth assignment τ extends to a truth assignment $\overline{\tau} : \{\phi\} \rightarrow \{\mathrm{T}, \mathrm{F}\}$ on the set of formulas $\{\phi\}$ by assigning truth values to variables in the formula and determining the truth of the formula by invoking the definitions of the connectives. The truth of a compound formula depends on the truth of its constituents. For example,

$$\overline{\tau}(\phi \rightarrow \psi) = \begin{cases} \mathrm{T} & \text{if } \overline{\tau}(\phi) = \mathrm{F} \text{ or } \overline{\tau}(\psi) = \mathrm{T} \\ \mathrm{F} & \text{otherwise.} \end{cases}$$

Two other common symbols, $\oplus$ and $|$, are the "exclusive or" and "nand" binary operators expressible in terms of the first four symbols. Specifically, $x_1|x_2 := \neg(x_1 \wedge x_2)$ and $x_1 \oplus x_2 := \neg((x_1 \wedge x_2) \vee (x_1|x_2))$.

1.2 SAT, TAUT, $\mathcal{P}$ and $\mathcal{NP}$

Statements in propositional logic, or *predicates*, which are of special interest are those for which there exists a truth assignment for which the predicate is true, and those for which the predicate is true for all truth assignments.

Definition The predicate ϕ is a *tautology* if for all truth assignments τ, $\overline{\tau}(\phi) = \mathrm{T}$.

Definition The predicate ϕ is *satisfiable* if there exists a truth assignment τ such that $\overline{\tau}(\phi) = \mathrm{T}$.

Based on their definitions, tautologies and satisfiable predicates are related as follows.

Fact If ϕ is a tautology then $(\neg\phi)$ is not satisfiable.

Definition *TAUT* is the set of all tautologies.

Definition *SAT* is the set of all satisfiable formulas.

The problem of recognizing a member of SAT is fundamental in computational complexity. $\mathcal{P}$ is defined to be the set of all predicates that are polynomial time recognizable. In this context, a predicate is a decision procedure yielding either 'T' or 'F'. *Polynomial time recognizable* means there exists an algorithm implementing the decision procedure which is bounded in the number of steps by a polynomial in the size of the input.

Example Let WFF, the set of well-formed formulas, be defined as the set of strings which are validly formed formulas build on connectives, parentheses and variables. Then the predicate WFF is a member of $\mathcal{P}$.

Definition $\mathcal{NP}$ is the set of predicates Q which can be expressed as

$$Q(x) = (\exists y, |y| < p(|x|))R(x, y),$$

where p is a polynomial, $|x|$ is the length of x viewed as a string, y is the "witness" to x, and R is a polynomial time algorithm which verifies that y is a witness to x. Thus $Q(x) = \mathrm{T}$ if and only if a witness y exists such that the time property $R(x, y)$ holds.

To show that SAT is in $\mathcal{NP}$, simply express it in the above form; e.g.,

$$SAT(X) \equiv (\exists y, (|y| \leq |x|))TRU(x, y),$$

where $TRU(x, y)$ is the algorithm which verifies that y encodes a truth assignment which satisfies x. We may take y to be weakly less than x in length, because it can simply encode truth values of the variables in the formula x as binary digits.

Homework 1 *Turn in a proof of $\mathcal{P} = \mathcal{NP}$ or $\mathcal{P} \neq \mathcal{NP}$ for \$1,000,000 (from the Clay Mathematics Institute) and an A+.*

2 Propositional Proof Systems

One purpose of propositional proof systems is to provide evidence of the membership of a formula ϕ in the set of tautologies TAUT. Of particular interest is the size of the proof of a tautology.

2.1 Truth Tables

Truth tables provide straightforward but lengthy verifications of tautologies. The proof of a tautology proceeds by exhaustively writing down all 2^n truth assignments for a predicate in n variables, and resolving the truth in each case by the logical rules. The written-out truth table is the actual proof. A proof of the tautology $A \to (B \to A)$ is given by the following table.

A	B	$B \to A$	$A \to (B \to A)$
T	T	T	T
T	F	T	T
F	T	F	T
F	F	T	T

We can easily obtain an estimate for the size of a truth table proof for a tautology ϕ. Let p be the number of distinct variables, and let m be the total number of unary or binary logical operators. The the truth table has approximately $m2^p = 2^{O(p)}$ entries. The size of such a table for $p = 100$ would be at least 1000 times the age of the universe in nanoseconds.

The exponential growth rate of the size of the truth table makes this kind of proof infeasible for even relatively small numbers of variables. Growth rates like 2^n, $2^{\epsilon n}$ and 2^{n^ϵ} are all "too big" in this sense. We prefer growth rates like n, $n \log n$, n^2 and n^3, which are feasible for large values of n.

2.2 Frege Proof Systems

A *Frege Proof System* consists of axioms, substitution rules and a rule of inference from which all tautologies can be proved in a much more tractable form that that of truth tables. Informally, a proof of a formula ϕ consists of a sequence of formulas in which each step is either an axiom, a substitution of a previous step, or derived by inference, and the last step is ϕ.

2.2.1 Proof System Components

We formalize these notions in the following definitions.

Definition The *schematic axiom* of a Frege proof system are a list of tautologies called *schematic tautologies* which may be used as the starting point of a Frege proof.

The list of schematic tautologies in a common Frege proof system, which allows any tautology to be proved, appears in Table 2. Here, A, B and C are variables, and for well-definedness of associativity we adopt the shorthand $\phi \to \psi \to \chi := \phi \to (\psi \to \chi)$.

$A \to (B \to A)$	$A \wedge B \to B$
$(A \to B) \to (A \to B \to C) \to (A \to C)$	$A \wedge B \to A$
$A \to A \vee B$	$A \to B \to A \wedge B$
$B \to A \vee B$	$(A \to B) \to (A \to \neg B) \to \neg A$
$(A \to C) \to (B \to C) \to (A \vee B \to C)$	$\neg\neg A \to A$

Table 2: Schematic Tautologies for a Frege proof system

Definition A *schematic substitution* σ is a mapping $\sigma : V \to WFF$ from a set $V \subseteq \{x_1, x_2, \ldots\}$ of variables to the set of well-formed formulas, written by

$$\sigma = (x_1/\phi_1, \ldots, x_k/\phi_k),$$

where $\phi_i = \sigma(x_i) = x_i\sigma$.

Informally, a schematic substitution σ replaces a variable x_i with some formula ϕ_i.

Definition $A\sigma$ is defined to be the result of replacing (simultaneously) each occurrence of x_i in the formula A by $x_i\sigma = \sigma(x_i)$.

Example Let $A = x_1 \to x_2$ and let $\sigma = (x_1/(x_1 \wedge x_2), x_2/x_1)$. Then

$$A\sigma = (x_1 \wedge x_2) \to x_1.$$

Fact If $\phi \in TAUT$, then for all σ, $\phi\sigma \in TAUT$.

Definition The *schematic rule of inference* called *modus ponens*, or MP, is represented by

$$\frac{x_1 \quad x_1 \to x_2}{x_2},$$

and is invoked by confirming that x_2 is true whenever x_1 is true and x_1 implies x_2.

Definition A *schematic inference*, denoted

$$\frac{A_1, \quad \ldots, \quad A_k}{B},$$

means that for all σ, if $A_1\sigma, \ldots, A_k\sigma$ have been proved, then $B\sigma$ can be inferred. Taking $k = 0$ yields a schematic axiom.

The above definitions are used to develop a schematic proof that some formula ϕ is a tautology. We formalize this type of Frege proof as follows.

Definition An $\mathcal{F}_0$-proof is a sequence of formulas $\phi_1, \phi_2, \ldots, \phi_k$ such that each ϕ_i is either an axiom or is inferred by *modus ponens* from some ϕ_j and ϕ_l, for $j, l < i$. It is called a *proof* of ϕ_k.

2.2.2 Proof System Characteristics

In the Frege proof system just described, the properties of soundness and completeness are highly desirable. *Soundness* means that no proof exists for a formula that is not a tautology. *Completeness* means that every tautology has a corresponding proof. We introduce the symbolism $\vdash_{\mathcal{F}_0} \phi$ to denote that $\mathcal{F}_0$ is a proof of ϕ, and use $\models \phi$ to denote $\phi \in TAUT$.

Theorem 1 (Soundness) *If* $\vdash_{\mathcal{F}_0} \phi$ *then* $\phi \in TAUT$.

Proof The proof is by induction on the number of steps in the $\mathcal{F}_0$ proof. The proof starts with a schematic axiom which is a tautology, every step in the proof by substitution is a tautology, and MP preserves tautologies, and so ϕ is a tautology.

Theorem 2 (Completeness) *If* $\phi \in TAUT$, *then* $\vdash_{\mathcal{F}_0} \phi$ *for some proof* $\mathcal{F}_0$.

Proof The proof of completeness for this Frege proof system is deferred until Lecture #2.

Generally, we require an expanded notion of soundness and completeness than what is outlined above. In particular, we may wish to prove a tautology in a proof system starting from an extra set of axiomatically tautological formulas.

Definition Let Γ be a set of formulas. Then $\Gamma \models \phi$ provided that for all truth assignments τ, if for all $\gamma \in \Gamma$ we have $\overline{\tau}(\gamma) = \mathrm{T}$, then $\overline{\tau}(\phi) = \mathrm{T}$. We call $\Gamma \models \phi$ a *tautological implication.*

The corresponding version of a $\mathcal{F}_0$-proof is as follows.

Definition Let Γ be a set of formulas. Then $\Gamma \vdash_{\mathcal{F}_0} \phi$ provided there exists a sequence $\phi_1, \ldots, \phi_k = \phi$ such that each ϕ_i is in Γ or is an (instance of an) axiom or is inferred by MP. We call the sequence an $\mathcal{F}_0$*-proof with hypotheses.*

We would certainly like to know how to deal with implicational tautologies when Γ is an infinite set. Fortunately, a compactness result simplifies things.

Theorem 3 (Compactness of tautological implication) *If* $\Gamma \models \phi$, *then there exists a finite subset* $\Gamma_0 \subseteq \Gamma$ *such that* $\Gamma_0 \models \phi$.

Proof The proof is left as an exercise.

Proposition 4 (Derived substitution rule)

1. *If* $\vdash_{\mathcal{F}_0} A$ *and* σ *is a substitution, then* $\vdash_{\mathcal{F}_0} A\sigma$.
2. *If* $\Gamma \vdash_{\mathcal{F}_0} \phi$, *then* $\Gamma\sigma \vdash_{\mathcal{F}_0} \phi\sigma$.

Proof Part 1 proceeds by applying σ to the whole $\mathcal{F}_0$-proof. Start with the sequence $\phi_1, \phi_2, \ldots, \phi_k = A$ and replace ϕ_i with $\phi_i\sigma$ to obtain $\phi_1\sigma, \phi_2\sigma, \ldots, \phi_k\sigma = A\sigma$. Part 2 proceeds similarly.

Note that a double substitution into a schematic axiom, such as ϕ_i replaced with $\phi_i\sigma$ can be simplified to a single substitution. Define

$$\begin{aligned} \sigma : x_i \mapsto x_i\sigma &= \phi_i, \qquad \text{and} \\ \tau : x_i \mapsto x_i\tau &= \psi_i. \end{aligned}$$

Then the double substitution $(A\sigma)\tau)$ can be written as $A(\sigma\tau)$ since

$$(\sigma\tau) : x_i \mapsto \phi_i\tau = x_i\sigma\tau.$$

The implicational soundness and completeness theorems are as follows.

Theorem 5 (Implicational soundness of $\mathcal{F}_0$) *If* $\Gamma \vdash_{\mathcal{F}_0} \phi$, *then* $\Gamma \models \phi$.

Proof The proof proceeds by induction on the number of steps in the $\mathcal{F}_0$-proof.

Theorem 6 (Implicational completeness of $\mathcal{F}_0$) *If* $\Gamma \models \phi$, *then* $\Gamma \vdash_{\mathcal{F}_0} \phi$.

Proof The proof is deferred until Lecture #2.

Math 267a - Propositional Proof Complexity

Lecture #2: 16 January 2002

Lecturer: Sam Buss

Scribe Notes by: Sashka Davis

3 Introduction to Frege Proof Systems

Last time we stated the Completeness and the Soundness theorems for the Frege Proof Systems, today we focus on the Completeness Theorem. The main point of the Completeness Theorem is that there exists a Frege Proof System which is complete.

We begin with an example of $\mathcal{F}_0$-proof. The axioms of the Frege system we will use, are the Schematic Tautologies, defined in the previous lecture (**Ax1** denoting the first axiom, **Ax2** the second, etc.).

Example $(A \to A)$ has an $\mathcal{F}_0$-proof.

Proof
The following five lines form an $\mathcal{F}_0$-proof of $(A \to A)$.
$A \to (A \to A) \to A$, an instance of axiom **Ax1**
$A \to A \to A$, an instance of axiom **Ax1**
$(A \to (A \to A)) \to (A \to (A \to A) \to A)) \to (A \to A)$, an instance of axiom **Ax2**
$(A \to (A \to A) \to A) \to (A \to A)$, by **MP**
$A \to A$. □

Proof Complexity The $\mathcal{F}_\mathbf{0}$-proof has $O(1)$ lines and $O(|A|)$ symbols.

Theorem 7 (Deduction Theorem) $\Gamma \vdash_{\mathcal{F}_0} A \to B$ *iff* $\Gamma, A \vdash_{\mathcal{F}} B$.

Proof

1. $\Gamma \vdash_{\mathcal{F}_0} A \to B \Longrightarrow \Gamma, A \vdash_{\mathcal{F}} B$.
   Suppose an $\mathcal{F}_0$-proof of $\Gamma \vdash_{\mathcal{F}_0} A \to B$ is given by lines (1)-(3). We can form an $\mathcal{F}_0$ proof of $\Gamma, A \vdash_{\mathcal{F}} B$ by adding two lines as shown:

$$\mathcal{F}_0 : \quad \Gamma \tag{1}$$

$$\vdots \tag{2}$$

$$A \to B \tag{3}$$

$$A \tag{4}$$

$$B \tag{5}$$

Line (4) is the added new hypothesis. Line (5) is derived by MP. Thus we obtain $\mathcal{F}_0$-proof for $\Gamma, A \vdash_{\mathcal{F}} B$.

Proof Complexity $O(n)$ lines and $O(nm)$ symbols.

2. $\Gamma, A \vdash_{\mathcal{F}_0} B \Longrightarrow \Gamma \vdash_{\mathcal{F}} A \to B$.

   Proof Idea: Let the $\mathcal{F}_0$-proof of $\Gamma, A \vdash_{\mathcal{F}_0} B$ be a sequence $B = \varphi_1, \ldots, \varphi_n$. We shall use the substitution rule and replace each φ_i by $A \to \varphi_i$. The sequence of formulas $A \to \varphi_i$ is not a valid proof, but it can be converted into a valid proof as follows. Each φ_i either is A, or is inferred from A by MP, or is an axiom, or is a member of Γ. Thus to patch the proof we need to exhaust the following four cases.

   - Case 1: φ_i is A
     Then we re-use the 5-line proof of $A \to A$.
   - Case 2: φ_i is inferred by MP: $\dfrac{\varphi_j \quad \varphi_k = \varphi_j \to \varphi_i}{\varphi_i}$, $j, k < i$
     $A \to \varphi_j$
     $A \to \varphi_j \to \varphi_i$
     $(A \to \varphi_j) \to (A \to (\varphi_i \to \varphi_j)) \to (A \to \varphi_i)$, by **Ax2**
     $A \to \varphi_i$, by **MP**.
   - Case 3: φ_i is an axiom
     $\varphi_i \to (A \to \varphi_i)$, by **Ax1** .
     So φ_i can be replaced by the three line proof of $A \to \varphi_i$.
   - Case 4: $\varphi_i \in \Gamma$
     $\varphi_i \to (A \to \varphi_i)$, by **Ax1** .

   □

   Proof Complexity
   Each line in the original $\mathcal{F}_0$-proof becomes either three or five lines in the $\mathcal{F}_0$-proof of $\Gamma \vdash_{\mathcal{F}_0} A \to B$. The proof complexity remains $O(n)$ lines and $O(nm)$ symbols.

3.1 Usage of the Deduction Theorem

Let $\bigwedge_{i=1}^{k} A_i$, denotes any parenthesization of the conjunction of $A_1, \ldots, A_n$.

Example Given $\vdash_{\mathcal{F}_0} \bigwedge_{i=1}^{k} A_i \to A_{i0}$, with proof complexity $O(k)$ lines and $O(k|B|)$ symbols, where $B = \bigwedge_{i=1}^{k} A_i$, prove that $\bigwedge_{i=1}^{k} A_i \vdash_{\mathcal{F}_0} A_{i0}$.

Proof We follow the $\mathcal{F}_0$-proof. Begin with $\bigwedge_{i=1}^{k} A_i$. Repeatedly use the axioms $(A \wedge B \to B)$ and $(A \wedge B \to A)$ with MP. All the lines are well behaved. □

4 The Completeness and Implicational Completeness Theorems

Recall that the Completeness theorem states: $\phi \in TAUT \Longrightarrow\ \vdash_{\mathcal{F}_0} \phi$, for some proof $\mathcal{F}_0$. Now we state and prove the Implicational Completeness Theorem.

Theorem 8 (Implicational Completeness Theorem) *If* $\Gamma \models A$ *then* $\Gamma \overset{\mathcal{F}_0}{\vdash} A$.

Proof If $\Gamma \models A$, then there exists a finite Σ, $\Sigma \subset \Gamma$, s.t. $\Sigma \models A$. So w.l.o.g. Γ is finite. Let $\Gamma = \{B_1, \ldots B_k\}$, then $\models B_1 \to (B_2 \to (\ldots \to (B_k \to A)\ \ldots))$.
By the Completeness Theorem there exists an $\mathcal{F}_0$-proof of the tautology. By applying the Deduction Theorem k times we obtain $\Gamma \overset{\mathcal{F}_0}{\vdash} A$. $\Box$

Theorem 9 (Completeness Theorem) $A \in TAUT$*, then* $\vdash_{\mathcal{F}_0} A$.

Proof We mimic the method of the Truth Table proofs. We consider all possible truth assignments. Let $A = A(x_1, \ldots, x_k)$ and $A \in TAUT$. Let τ is a truth assignment, and

$$x_i^\tau = \begin{cases} x_i & \text{if } \tau(x_i) = \text{T} \\ \neg x_i & \text{if } \tau(x_i) = \text{F} \end{cases}$$

$$A^\tau = \begin{cases} A & \text{if } \tau(A) = \text{T} \\ \neg A & \text{if } \tau(A) = \text{F} \end{cases}$$

The proof follows from the following three claims:

Claim If $\vdash_{\mathcal{F}_0} \bigwedge\!\!\!\bigwedge_{i=1}^k x_i^\tau \to A^{(\tau)}$ then $\bigwedge\!\!\!\bigwedge_{i=1}^k x_i^\tau \vdash_{\mathcal{F}_0} A^{(\tau)}$

Proof The proof of the claim is based on the complexity of A.
Base case: If A is atomic then A is one of the x_i.
Suppose $A = B \bullet C$, where $\bullet$ is one of $\{\vee, \wedge, \to, \neg\}$, then $B^\tau, C^\tau \overset{\mathcal{F}_0}{\vdash} A^\tau$.
For each connective there are four cases for B and C. For example let ”$\bullet$“ = ”$\to$“ then:
$B, C \vdash_{\mathcal{F}} (B \to C)$
$B, \neg C \vdash_{\mathcal{F}} \neg(B \to C)$
$\neg B, C \vdash_{\mathcal{F}} (B \to C)$
$\neg B, \neg C \vdash_{\mathcal{F}} (B \to C)$

Proof Complexity The base case contributes $O(k)$ line, and for each connective the proof grows by finitely many lines, thus the total number of lines is $O(k + |A|) = O(|A|)$. Each line has $O(|A|)$ symbols, thus in total the proof has $O(|A|^2)$ symbols. However, we have to repeat this for all 2^k truth assignments to $x_1, x_2, \cdots, x_k$.

The following two claims would be used without a proof:

Claim $\vdash_{\mathcal{F}_0} ((Z \wedge C) \to D) \to ((\neg Z \wedge C) \to D) \to (C \to D)$

Claim $\vdash_{\mathcal{F}_0} ((Z \to A) \to (\neg Z \to A) \to A$

Now associating the conjunctions, $\bigwedge\!\!\!\bigwedge_{i=1}^{k} x_i^{\tau}$, from right to left w.l.o.g. we obtain: $\pm x_1 \wedge (\pm x_2 \wedge (\ldots (\pm x_{k-1} \wedge \pm x_k) \ldots) \to A$. We peel off all the variables one by one to obtain A. E.g. the last steps (using the second claim) are:

$$\left.\begin{array}{r} x_{k-1} \wedge x_k \to A \\ \neg x_{k-1} \wedge x_k \to A \end{array}\right\} x_k \to A$$

$$\left.\begin{array}{r} x_{k-1} \wedge \neg x_k \to A \\ \neg x_{k-1} \wedge \neg x_k \to A \end{array}\right\} \neg x_k \to A$$

thus $\vdash_{\mathcal{F}_0} x_k \to A$ and $\vdash_{\mathcal{F}_0} \neg x_k \to A$. From this and the third claim, $\vdash_{\mathcal{F}_0} A$.
□

Proof Complexity The last part of the proof contributes $O(2^k)$ new lines, with $O(|A|)$ symbols per line. Thus A has $\mathcal{F}_0$-proof of $O(|A|2^k)$ lines. The total number of symbols is $O(|A|^2 2^k)$, where k is the number of distinct variables in A.

5 Observations

The Completeness Theorem states that all valid tautologies can be proved. We observe that the size bounds of the $\mathcal{F}_0$-proofs are the same as the size bounds of the Truth Table Proofs (TTP). However, the $\mathcal{F}_0$ can be separated from the TTP. We demonstrate the separation by the following example.

Example $\phi = (A_1 \wedge \neg A_1) \vee (A_2 \wedge A_3 \wedge \cdots \wedge A_k)$.

ϕ has a short $\mathcal{F}_0$-proof and exponentially large TTP. Thus in the best case $\mathcal{F}_0$-proofs are better than TTP, but it is an open question whether they are better than TTP in the worst case.

A Proof System must be *sound* and the proofs ought to be *checkable efficiently* (in polynomial time). *Completeness* is another property which is nice and desirable, but not required.

6 P-simulate

The next theorem states that Truth Table Proofs (TTP) can be converted into $\mathcal{F}_0$ proofs by a polynomial time algorithm.

Theorem 10 (Simulation) *Frege Proof Systems p-simulate Truth Table Proofs.*

The converse does not hold as it can be seen from the example above. Thus TTP do not simulate Frege Proof System.

Definition An abstract propositional proof system over the propositional language $L = \{\vee, \wedge, \to, \neg\}$ is a polynomial time computable function f with domain strings of symbols and $range(f) \subset TAUT$.

Definition The function f is complete if the $range(f) = TAUT$.

Definition An f-proof of a formula φ is any x s.t. $f(x) = \varphi$.

Definition $\mathcal{F}_0$ as an abstract proof system is defined as:

$$f_{\mathcal{F}_0}(x) = \begin{cases} \varphi & \text{if } x \text{ codes a valid } \mathcal{F}_0\text{-proof of } \varphi \\ (x_1 \vee \neg x_1) & \text{otherwise} \end{cases}$$

This idea for constructing an abstract proof systems works for many other proof systems too. For example, let ZF be the usual theory of set theory, then

$$f_{ZF}(x) = \begin{cases} \varphi & \text{if } x \text{ codes a valid } ZF \text{ proof of "}\varphi \text{ is a tautology``} \\ (x_1 \vee \neg x_1) & \text{otherwise} \end{cases}$$

is an abstract proof system.

Math 267a - Propositional Proof Complexity

Lecture #3: 23 January 2002

Lecturer: Sam Buss

Scribe Notes by: Reid Andersen

7 p-Simulation

Definition Let f and g be proof systems in the same language. We say f *p-simulates* g if there exists a poly-time computable function $H(x)$ such that $\forall x$, $g(x) = f(H(x))$. We say f *simulates* g if there exists a polynomial $p(n)$ such that $\forall x \exists y$, $|y| \leq p(|x|)$ and $f(y) = g(x)$.

Definition A proof system f is *maximal* if f simulates g for any proof system g. A proof system f is *super* if there exists a polynomial $p(n)$ such that $\forall \varphi \in TAUT$, $\exists x$ such that $|x| \leq p(|\varphi|)$ and $f(x) = \varphi$. Note that any super proof system is maximal.

Open Question Is there a super or maximal proof system?

Theorem 11 *[3] [Cook] There exists a super proof system $\iff NP = co - NP$.*

Homework 2 *Prove the above theorem for a homework excercise.*

Definition A Frege system is a proof system given by a finite set of of schematic axioms and inference rules, and must be implicationally sound and implicationally complete.

Theorem 12 *[4] [Cook-Reckhow] If $\mathcal{F}_1$, $\mathcal{F}_2$ are Frege systems, then $\mathcal{F}_1$ p-simulates $\mathcal{F}_2$.*

Proof For the proof we will assume $\mathcal{F}_1$ and $\mathcal{F}_2$ have the same language, but the statement is true in general. Consider a rule of $\mathcal{F}_2$, $\frac{A_1 \ldots A_k}{B}$. $\mathcal{F}_1$ can prove $A_1 \ldots A_k \vdash B$ by the implicational completeness of Frege proof systems. Consider an $\mathcal{F}_2$-proof $\varphi_1 \ldots \varphi_n$. We convert to an $\mathcal{F}_1$-proof as follows: φ_i follows from an inference rule $\frac{A_1\sigma \ldots A_k\sigma}{B\sigma}$, where $A_1\sigma = \varphi_{i_1}$, $\ldots$, $A_k\sigma = \varphi_{i_k}$, with $i_1 \ldots i_k < i$, and $B\sigma = \varphi_i$. Assuming $\varphi_{i_1} \ldots \varphi_{i_k}$ already proved, use the substitution σ on the $\mathcal{F}_1$-proof $A_1 \ldots A_k \vdash B$ to get an $\mathcal{F}_1$-proof $\varphi_{i_1} \ldots \varphi_{i_k} \vdash \varphi_i$. Combining this proof and the proof of $\varphi_{i_1} \ldots \varphi_{i_k}$ yields an $\mathcal{F}_1$-proof of ϕ_i.

Proof Complexity This is a polynomial time procedure. For each line of the $\mathcal{F}_2$-proof, there are $O(1)$ lines in the $\mathcal{F}_1$-proof. If the $\mathcal{F}_2$ proof has n lines and m total symbols, the $\mathcal{F}_1$ proof has $O(n)$ lines, and each line has $O(m)$ symbols. So the $\mathcal{F}_1$-proof contains $O(n)$ lines, and $O(mn)$ total symbols. Since $n \leq m$, the size of the $\mathcal{F}_1$-proof is bounded by a polynomial in the size of the $\mathcal{F}_2$-proof.

Open Question Can the bound of $O(mn)$ symbols in the preceeding proof be improved to $O(m)$? It can if we assume that $\mathcal{F}_1$ has modus ponens, but is it true in general?

Open Question Are Frege systems super? or maximal?

Open Question Is there a "natural" proof system stronger than Frege systems?

8 Extended Frege Sytems

Definition Here we define an extended Frege system, $e\mathcal{F}$. An $e\mathcal{F}_0$-proof is the same as an $\mathcal{F}_0$-proof, except the size of the proof is computed differently. The size of an extended Frege proof of A is (# of lines in the proof) + $|A|$.

Example In a previous lecture we saw that any formula $A \to A$ has an $\mathcal{F}_0$-proof of five lines. So there is an $e\mathcal{F}_0$-proof of $A \to A$ of size $5 + |A|$.

The catch is that an extended Frege system as defined above is not an abstract proof system, since an abstract proof system defines the size of a proof x to be the number of symbols in x. For this reason we will present an encoding where an $e\mathcal{F}_0$ proof with size n in the extended Frege sense can be encoded by a string of length $O(poly(n))$. We also present a polynomial time decoding algorithm to verify that a string encodes a valid $e\mathcal{F}_0$ proof. This decoding algorithm defines an abstract proof system with the notion of size that we desire, within a polynomial.

Encoding [7] [Parikh] Number the rules of inference. The axioms take values 0...9, and modus ponens takes 10. We represent an $e\mathcal{F}_0$-proof $\varphi_1, \ldots, \varphi_n = \varphi$ by a tuple $\langle e_1, \ldots, e_n, \varphi\rangle$ where if φ_i is an instance of axiom k then $e_i = k$, and if φ_i is inferred from $\varphi_{j_i}, \varphi_{k_i}$ by modus ponens, $e_i = \langle 10, j_i, k_i\rangle$.

The size of this proof skeleton is $O(nlogn + |\varphi|)$, where n is the size of the $e\mathcal{F}_0$ proof.

Claim There is a polynomial time algorithm to decide if an encoding corresponds to a valid $e\mathcal{F}_0$-proof.

Proof We convert the skeleton into a unification problem which has a solution iff the proof skeleton is valid. We create new "metavariables" $y_1 ... y_n$, and z^i_j, and search for a substitution $\sigma : y_i \mapsto \varphi_i$ which must satisfy the following equations:

1. $y_n \doteq \varphi$. (means $\sigma y_n = \varphi$)
2. if $e_i = \langle 10, j_i, k_i\rangle$, we require that $y_{k_i} \doteq (y_{j_i} \to y_i)$
3. for $0 \leq e_i \leq 9$ let A be the e_i-th axiom. Replace each x_j in A by z^i_j, and denote this instance of the axiom A by A^i. We require that $y_i \doteq A^i$.

A substitution σ that satisfies these requirements is called a unifier, and the encoding corresponds to a valid $e\mathcal{F}_0$-proof of φ if and only if such a σ exists. More on this next time.

Math 267a - Propositional Proof Complexity

Lecture #4: 28 January 2002

Lecturer: Sam Buss

Scribe Notes by: Alan Nash

9 The Unification Problem

9.1 The Problem

Last time we looked at how to solve the system: $x = \phi$, $\phi = \psi \rightarrow x$. In general, given:

- Variables: $x, y, z, \ldots$
- Function Symbols: $f, g, h, \ldots$, each with specified arity (including constants as 0-ary functions)
- Terms: built from variables and function symbols
- Finite sets of equations: $s_i \doteq t_i$ (where s_i, t_i are terms)

the *unification problem* is to find a substitution σ such that $s_i\sigma = t_i\sigma$ where equality here is equality of symbols.

Example Take the system with a single binary function symbol f and equations:

$$\begin{aligned} x_1 &\doteq f(x_2, x_2) \\ x_2 &\doteq f(x_3, x_3) \\ x_3 &\doteq f(x_4, x_4) \end{aligned}$$

a solution is given by σ such that:

$$\begin{aligned} \sigma(x_3) &= f(x_4, x_4) \\ \sigma(x_2) &= f(f(x_4, x_4), f(x_4, x_4)) \\ \sigma(x_1) &= f(f(f(x_4, x_4), f(x_4, x_4)), f(f(x_4, x_4), f(x_4, x_4))) \end{aligned}$$

As this example shows, in the worst case the solution's size is exponential in the size of the problem (total number of symbols to represent it).

Example The system with the single equation $f(x_1, x_2) = g(x_1, x_2)$ is unsolvable: the function symbols clash.

Example The system with equations $x \doteq g(y)$ and $y \doteq h(x)$ is unsolvable (the solution must be finite): it yields the derived equation $x \doteq g(h(x))$ ("occurs check").

We will see that these are the only two kinds of things that can go wrong.

9.2 An Algorithm for the Unification Problem

First, notice that terms can be represented by ordered directed acyclic diagrams (ODAGs). By 'ordered' we mean that there is a total ordering on each set of edges coming out of a vertex. When represented as ODAGs, the solutions are always polynomial in size.

To solve an unification problem, we first define an equivalence relation $\approx$ on terms, as follows:

1. $s \doteq t \rightarrow s \approx t$ (equation in system)
2. $s = t \rightarrow s \approx t$ (equal as strings)
3. $f(s_1, s_2, \ldots, s_k) \approx f(t_1, t_2, \ldots, t_k) \rightarrow \forall i (s_i \approx t_i)$
4. $s \approx t \rightarrow t \approx s$
5. $r \approx s \approx t \rightarrow r \approx t$

Example Given the system $f(x, g(y)) \doteq f(h(y), z)$ we have $x \approx h(y)$ and $z \approx g(y)$

Claim A unification problem is solvable iff (a) There are no $r \approx s$ so that r and s have different principal (outermost) function symbols and (b) The $\approx$-equivalence classes are well-ordered (i.e. no cycles) under the extension of the proper subterm relation to the equivalence classes (i.e., $[r] \prec [s]$ iff r is a proper subterm of s)

Proof: [$\Rightarrow$] suppose that σ is a solution. Then it is easy to show that $r \approx s$ implies $r\sigma = s\sigma$ holds, by induction on the cases defining $\approx$. Similarly, the relation "$r\sigma$ is a proper subterm of $s\sigma$" is a well-ordering that refines $r \prec s$.

[$\Leftarrow$] Define σ by induction along $\prec$ as follows. The base elements are of the form $[x]$, containing only variables and at most a single constant c. If $c \in [x]$ then define $\sigma(x) = c$. Otherwise, define $\sigma(x) = y$ where y is a new variable that depends only on $[x]$.

For the rest, i.e. $x \in [f(s_1, \ldots, s_k)]$ define $\sigma(x) = (f(s_1, \ldots, s_k))\sigma = f(s_1\sigma, \ldots, s_k\sigma)$.

Subclaim: $v \approx s$ implies $r\sigma = s\sigma$ (almost immediate).

Partial proof: Suppose $r = f(r_1, \ldots, r_k)$ and $s = f(s_1, \ldots, s_k)$. Then by the induction hypothesis, we have $r_i\sigma = s_i\sigma$.

What we have just described is a polynomial time algorithm. We can obtain the transitive closure using, for example, a breadth-first search or some other similar means (linear in size of graph). In fact, it is quadratic time. Notice that we have considered so far the decision problem; to provide an output in polynomial time we must output ODAGs.

supply reference: Paterson and Wegman provide a linear-time algorithm.

To unify a set of terms $\{r, s, t, \ldots\}$ means to unify the set $\{r \doteq s, s \doteq t, \ldots\}$. Conversely, to unify $\{r_i \doteq s_i : 1 \leq i \leq k\}$ is the same as to unify $\{f(r_1, r_2, \ldots, r_k) = f(s_1, s_2, \ldots, s_k)\}$

10 Extended Frege Systems (Again)

Now we look at an alternative definition of extended Frege systems ($e\mathcal{F}$-systems). An $e\mathcal{F}$-system is a Frege system ($\mathcal{F}$-system) augmented by *the extension rule.* That is, an $e\mathcal{F}$-proof consists of formulas $\phi_1, \ldots, \phi_n$ where each ϕ_i is:

- an axiom
- inferred by a rule from previous formulas
- is $z \leftrightarrow \psi$ where
  - z does not occur in ψ,
  - z does not occur in any previous line of the proof, and
  - z does not occur in ϕ_n.

To convert an $e\mathcal{F}$-proof into a $\mathcal{F}$-proof, proceed as follows:

- Replace z by ϕ wherever z occurs
- Replace $\psi \leftrightarrow \psi$ with a $\mathcal{F}$-proof of it ($O(1)$ lines).

Theorem 13 *(Statman [8]) An n-line $\mathcal{F}$-proof of ϕ can be converted into an $e\mathcal{F}$proof of ϕ with $O(n + |\phi|)$ lines and $O(n + |\phi|^2)$ symbols.*

Theorem 14 *Any two $e\mathcal{F}$-systems p-simulate each other*

Conjecture 1 *$\mathcal{F}$-systems do not simulate $e\mathcal{F}$-systems*

Notice that in polynomial-size $\mathcal{F}$-proofs, each line is a polynomial-size formula, while in polynomial-size $e\mathcal{F}$-proofs, each line is equivalent to a polynomial-size circuit. This converts to circuits; every use of resolution defines a circuit that is then used as input to subsequent circuits: $\psi(x_1, \ldots, x_k, z_1, \ldots, z_\ell)$ where $z_i \leftrightarrow \chi_i(x_1, \ldots, x_k, z_i, \ldots, z_{i-1})$.

Open Problem Polynomial size formulas have the same expressive power as polynomial circuits

Not only we do not know whether $\mathcal{F}$ systems simulate $e\mathcal{F}$ systems and whether p-size formulas simulate p-size circuits, we also do not know how to prove either one of this implications if the other one holds.

Proof	Nonuniform	Uniform	Bounded
System	Complexity	Complexity	Arithmetic
$\mathcal{F}$	p-size formulas	ALOGTIME	TNC_1
$e\mathcal{F}$	p-size circuits	P	PV/S_2^1

For references on this table, see Steve Cook's slides from the Edinburgh Complexity Workshop, October 2001, available at

`http://www.cs.toronto.edu/ sacook/edinburgh.ps`

Math 267a - Propositional Proof Complexity

Lecture #5: 29 January 2002

Lecturer: Sam Buss

Scribe Notes by: Tamsen Dunn

11 The Pigeon Hole Principle

Last time we finished our introduction to Frege Proof Systems. In this lecture we will give a propositional formulation of and a proof of the Pigeon Hole Principle. Its an interesting side note that this theorem was considered self evident until it was brought under the scrutiny of discrete mathematicians. Now the Pigeon Hole Principle is considered quite subtle. We will begin by analyzing the familiar form of the theorem.

11.1 The Familiar Form of the Pigeon Hole Principle

Theorem 15 (Pigeon Hole Principle A)

$$PHP_n^{n+1} : \forall n \in N \neg\exists f : \{1, 2, ..., n+1\} \rightarrow \{1, 2, ..., n\}$$

The above formulation of the Pigeon Hole Principle is sometimes taken in and of itself as the definition of n being finite.

11.2 The Pigeon Hole Principle by Induction

Now, we want to write the Pigeon Hole Principle as a family of propositional tautologies. Fix $n \geq 1$ and let our variables be $P_{i,j}$, where $1 \leq i \leq n+1$ and $1 \leq j \leq n$ and $P_{i,j}$ means $f(i) = j$. In this way, we no longer have a set of ordered pairs but a set of graphs.

Theorem 16 (Pigeon Hole Principle B)

$$PHP_n^{n+1} : \bigwedge_{i=1}^{n+1} \bigvee_{j=1}^{n} P_{i,j} \rightarrow \bigwedge_{i=1}^{n} \bigwedge_{j=i+1}^{n+1} \bigwedge_{m=1}^{n} (P_{i,m} \wedge P_{j,m})$$

This ought to be a tautology. Let's check:

Example $[PHP_1^2{:}]$ $(P_{1,1} \vee P_{1,2}) \rightarrow (P_{1,1} \vee P_{2,1}\ )$

So that works. Now for $n = 2$.

Example $[PHP^3_2:]$

Left hand side: $(P_{1,1} \vee P_{1,2}) \wedge (P_{2,1} \vee P_{2,2}) \wedge (P_{3,1} \vee P_{3,2})$
should imply
Right hand side: $(P_{1,1} \wedge P_{2,1}) \vee (P_{1,2} \wedge P_{2,2}) \vee (P_{1,1} \wedge P_{3,1}) \vee (P_{1,2} \wedge P_{3,2}) \vee (P_{2,1} \wedge P_{3,1}) \vee (P_{2,2} \wedge P_{3,2})$
and it does.

In the above formulation, the Pigeon Hole Principle has been reduced to a family of tautologies, each polynomial in size.

Proof Complexity On the right hand side we see that there are n^3 ways to select the mapping, so these formulas have $O(n^3)$ symbols.

Cook and Reckhow were the first to show that the Pigeon Hole Principle could be given a polynomial-size $e\mathcal{F}$-proof.

Idea of Proof: Given a mapping $f : [n+1] \rightarrow [n]$ where $[n]$ and $[n+1]$ are sets such that $[n] = \{1, 2, ..., n\}$, we want to show that this mapping causes a contradiction.

Define $f^n = f$ and

$$f^m(i) = \begin{cases} f^{m+1}(i) & \text{when } f^{m+1}(i) < m+1 \\ f^{m+1}(m+2) & \text{otherwise} \end{cases}$$

So, given an f^{m+1} we want to find an f^m. Suppose the mapping is as follows:

$$domain - - - - - range$$

$$\begin{array}{ccc} m+2 & & --- \\ m+1 & \searrow & m+1 \\ m & & m \\ \vdots & & \vdots \\ 2 & \nearrow & 2 \\ 1 & & 1 \end{array}$$

The idea of this proof is to successively use induction to prove $f^m : [m+1] \rightarrow [m]$ is a one-to-one mapping. At each inductive step drop a pair, such as $m+2$ from the domain and $m+1$ from the range, and reconnect the respective arrows to which ever empty slots are available.

Assuming $f^m + 1 : [m+2] \rightarrow [m+1]$, we use the inductive claims that

(1) f^{m+1} is one-to-one $\rightarrow$ f^m is one-to-one, and

(2) $f^{m+1} : [m+1] \rightarrow [m]$

The induction will finally stop at the bottom, at $m = 1$ where

$$f^1 : [2] \rightarrow [1]$$

It is reasonable to claim this is impossible, a contradiction by definition.

Now, let's translate this into an $e\mathcal{F}$-proof:

11.3 $e\mathcal{F}$-Proof of the Pigeon Hole Principle

Proof

Idea of Proof: We begin by presuming the hypothesis $\neg PHP_n^{n+1}$. Then we derive some instance of $\neg PHP_1^2$ which can be disproved, and allow that to negate the hypothesis.

First introduce some new variables: For $\neg PHP_n^{n+1}$, we need to have at least the set of $\{P_{i,j}\}$ variables.

Define $\neg PHP_m^{m+1}(q^m)$ to have new variables $q_{i,j}^m$ for $m = n, ..., 2, 1$. Let $q_{i,j}^n \leftrightarrow P_{i,j}$. Now the extension rule is used to introduce $q_{i,j}^m$ by

$$q_{i,j}^m \leftrightarrow (q_{i,j}^{m+1} \vee (q_{i,m+1}^{m+1} \wedge q_{m+2,j}^{m+1}))$$

where $1 \leq i \leq m+1$, and $1 \leq j \leq m$. Since each q^m is so defined by the previous q^{m+1}, they are allowed by the extension rule.

Now we claim that $e\mathcal{F}$ has polynomial size proofs of $\neg PHP_{m+1}^{m+2} \rightarrow \neg PHP_m^{m+1}$. Proving this claim below is equivalent to proving the main theorem.

$$\neg PHP_{m+1}^{m+2}(q^{m+1}) \rightarrow \neg PHP_m^{m+1}(q^m)$$

To finish, we need to give an $e\mathcal{F}$-proof of the conjuncts of $\neg PHP_m^{m+1}(q^m)$ from the conjuncts of $\neg PHP_{m+1}^{m+2}(q^{m+1})$. This is basically a brute-force case analysis. The idea of the case analysis comes back to the picture we drew earlier. Its simply not possible to map two (or more) elements of the domain to a single slot in the range. Any time it did would violate one-to-oneness. The full proof can be found in Cook and Reckhow, JSL 1979.

Proof Complexity All formulas given in this $e\mathcal{F}$-proof are of $O(n^3)$ since in the PHP there are n^3 steps for that many conjuncts. We step down from n, so that gives us $O(n^4)$ lines, and each line had $O(n^3)$ symbols.

Suppose we were instead thinking of an ordinary $\mathcal{F}$-proof. Straightforward conversion fails with exponential blow up. We have to re-express every $q_{i,j}$ in terms of its original variables, thus eliminating all the extension variables from $m = n$ to $m = 1$. That means we would have roughly 3 times as many symbols. For example, q^m uses three q^{m+1} 's, so by straightforward replacement substitution the formulas increase in size by a factor of 3^n.

For a different $\mathcal{F}$-proof which is polynomial in size, see Buss, JSL 1978. Buss's proof rests on a counting argument: "The idea behind the proof is that one can't [not have the PHP] because then one set would have more than the other."

We have very few examples of $e\mathcal{F}$ and $\mathcal{F}$ size comparisons. However, we still tend to believe the separation is maintained from arguments made in circuit theory. We will come back to this.

12 Tree-Like versus Non-Tree-Like Proofs

Definition A $\mathcal{F}$ or $e\mathcal{F}$-proof is *tree-like* if each formula in the proof is used at most once as a hypothesis of an inference. Other proofs may be dag-like or sequence-like.

Theorem 17 (Krajíček) *Tree-like (extended) $\mathcal{F}$-proofs p-simulate ordinary (extended) $\mathcal{F}$-proofs.*

Intuitively, you might expect the transition to tree-like proofs would cause an an exponential blow up. But that is not the case.

Proof Given a sequence-like proof, $\phi_1, \phi_2, ..., \phi_n = \phi$ one wants to create a tree-like proof from $\psi_1, \psi_2, ..., \psi_n$ where $\psi_i = \bigwedge_{j=1}^{i} \phi_i$. We hope that if multiple formulas are necessary we can use conjunctions for it once and only once. Now, we claim that the sequence $\psi_1, \psi_2, ..., \psi_n$ can be "patched up" to be a tree-like proof. The patching is done by cases:

Case (1) : ϕ_i is an axiom.

Assume ψ_{i-1} has already been derived. We have the axiom ϕ_i. We can derive

$$\psi_{i-1} \to \phi_i \to (\psi_{i-1} \wedge \phi_i)$$

as an instance of $A \to B \to (A \wedge B)$. So MP twice gave us

$$\psi_{i-1} \wedge \phi_i$$

which is equivalent to ψ_i. And case (1) is proven.

Case (2) :

ϕ_i is inferred from ϕ_j and ϕ_k by MP. ϕ_k is $(\phi_j \to \phi_i)$. By assuming we have a proof of ψ_{i-1}, we can complete the rest of the proof by a straightforward unwinding of conjunctions. See the example below to see how this will work:

Example The following are tautologies: $A \wedge B \to A$ and $A \wedge B \to B$, $(A \to B) \wedge (B \to C) \to (A \to B)$. Given an instance

$$(\alpha \wedge \beta) \wedge \gamma \to (\alpha \wedge \beta) \to \alpha.$$

$$(\alpha \wedge \beta) \wedge \gamma \to (\alpha \wedge \beta) \to ((\alpha \wedge \beta) \to \alpha) \to ((\alpha \wedge \beta) \wedge \gamma) \to \alpha.$$

Using MP twice, for two conjuncts we have

$$((\alpha \wedge \beta) \wedge \gamma) \to \alpha.$$

Substituting the above equations into our proof, we now have the desired

$$\psi_{i-1} \to \phi_j$$

and

$$\psi_{i-1} \to \phi_k$$

Each of the above is tree like. Now, using a substitution instance of the following tautology:

$$A \to (A \to B) \to (A \to (B \to C)) \to (A \wedge C)$$

we can derive from the 3 MP's that

$$\psi_{i-1} \to (\psi_{i-1} \to \phi_j) \to (\phi_{j-1} \to \phi_k) \to (\psi_{i-1} \wedge \phi_i)$$

We are done with case (2) because

$$\psi_{i-1} \wedge \phi = \psi_i$$

Therefore, in either case, each ψ_i can be created for the tree-like proof. That concludes our proof and this lecture.

Math 267a - Propositional Proof Complexity

Lecture #6: 4 February 2002

Lecturer: Sam Buss

Scribe Notes by: Rosalie Iemhoff

13 Substitution Frege systems

In a previous lecture we encountered certain proof systems for which it is *conjectured* that they are stronger than Frege systems (in the sense that Frege systems do not p-simulate them). These were the extended Frege systems. Here we will define other such proof systems: the substitution Frege systems. We will see that these systems are as strong as extended Frege systems; they p-simulate extended Frege systems and vice versa. As in the case of extended Frege systems, it is an open question whether Frege systems p-simulate substitution Frege systems or not.

Definition A *substitution Frege system*, $s\mathcal{F}$, is a Frege system augmented with a substitution rule $A/A\sigma$. Thus a proof in a substitution Frege system $s\mathcal{F}$ is a sequence $\varphi_1, \ldots, \varphi_n$, where every φ_i is an axiom of $\mathcal{F}$, or inferred from earlier φ_j by a rule of $\mathcal{F}$, or $\varphi_i = \varphi_j\sigma$, for some $j < i$ and some substitution σ.

Note that substitution Frege systems are indeed complete proof systems; it is not difficult to see that they are sound and complete (the substitution rule is clearly sound, and completeness follows from the fact that Frege systems are complete). Note however that they are not implicationally sound. This is not a problem since implicational soundness is not required for a proof system.

Theorem 18 *$s\mathcal{F}$-systems and $e\mathcal{F}$-systems p-simulate each other.*

Proof We will only show that $s\mathcal{F}$-systems p-simulate $e\mathcal{F}$-systems, the other direction can be found in [6] or [5] (Lemma 4.5.5). For this proof we consider extended Frege systems of the second type, i.e. with an extension rule. Let $\varphi_1, \ldots, \varphi_n$ be a proof in an extended Frege system $e\mathcal{F}$ of size k. Thus $k = \mathcal{O}(n + |\varphi_n|)$. W.l.o.g. we can assume that the first m lines in the proof are extension rules and that the others are not: for $i \leq m$, $\varphi_i = z_i \leftrightarrow \chi_i(\bar{x}, z_1, \ldots, z_{i-1})$, i.e. φ_i is an instance of the extension rule, and for $i > m$, φ_i is not an instance of the extension rule. Clearly, we have

$$\varphi_1, \ldots, \varphi_m \vdash_{\mathcal{F}} \varphi_n.$$

Observe that this Frege proof has size at most k. By the Deduction Theorem (Lecture #2) we have

$$\vdash_{\mathcal{F}} \left(\bigwedge\!\!\!\bigwedge_{i=1}^{m} (z_i \leftrightarrow \chi_i)\right) \to \varphi_n, \tag{6}$$

where the proof has size polynomial in k. W.l.o.g. we associate the brackets in the big conjunction to the left. From (6) we get

$$\vdash_{\mathcal{F}} \left(\bigwedge\!\!\!\bigwedge_{i=1}^{m-1}(z_i \leftrightarrow \chi_i) \wedge (z_m \leftrightarrow \chi_m)\right) \rightarrow \varphi_n. \tag{7}$$

Note that the variable z_m only occurs at one place in the whole formula. Now we hit the formula in (7) with a substitution $\sigma(z_m) = \chi_m$ and obtain

$$\vdash_{\mathcal{F}} \left(\bigwedge\!\!\!\bigwedge_{i=1}^{m-1}(z_i \leftrightarrow \chi_i) \wedge (\chi_m \leftrightarrow \chi_m)\right) \rightarrow \varphi_n.$$

From this we get

$$\vdash_{\mathcal{F}} \left(\bigwedge\!\!\!\bigwedge_{i=1}^{m-1}(z_i \leftrightarrow \chi_i)\right) \rightarrow \varphi_n.$$

Then we repeat this procedure for $m-1$, then for $m-2$, etc., so that we finally end up with

$$\vdash_{\mathcal{F}} \varphi_n.$$

About the size of the proof: for (6) we have already observed that the proof is of size polynomial in k. From this it is easy to conclude that the other proofs have size polynomial in k as well.

It is interesting to ask whether one can restrict the substitutions in the substitution rule and still get systems that are as strong as $s\mathcal{F}$ systems. Renaming Frege systems are an example of this. A substitution is a *renaming substitution* if its range is contained in the set of propositional variables. A *renaming Frege system*, $r\mathcal{F}$, is a substitution Frege system for which in the substitution rule only renaming substitutions are allowed. A renaming substitutions σ replaces the variables in A by (possibly) other variables (hence the name renaming). At first sight renaming Frege systems may seem weaker than substitution Frege systems, but it turns out that both systems p-simulate each other (a proof can be found in [2]). Hence extended Frege systems and renaming Frege systems p-simulate each other as well.

One could try to restrict the substitutions that are allowed in the substitution rule even further than in renaming Frege systems. For example, one could require that the substitutions map variables only to variables that occur in the last line of the proof, or one could require that the renaming substitutions are injective. For these systems it is no longer known whether they are as strong as extended Frege systems or not.

A propositional proof systems that is considered stronger than extended Frege systems, is quantified propositional logic QBF. QBF is an extension of propositional logic by propositional quantifiers

$$\forall x \text{ (intended meaning } A(x/\top) \wedge A(x/\bot))$$

$$\exists x \text{ (intended meaning } A(x/\top) \vee A(x/\bot))$$

It is known that QBF p-simulates extended Frege systems [5]. It is conjectured that the converse does not hold, but this is open.

14 The best known lower bounds on proof lengths

In this section we will discuss the best known lower bounds on Frege and extended Frege proofs. For Frege proofs we have a linear lower bound on the number of lines and a quadratic lower bound on the size. Hence for extended Frege proofs we have a linear lower bound on the size of proofs. Since the best known upper bounds on Frege proofs are exponential, there is still a large gap between the lower bounds and the upper bounds. To prove the lower bounds we need some terminology.

Definition Given a Frege proof P, a formula A is called *active* in P if it occurs in P as a subformula in an inference that explicitly uses the principal connective of A. We tacitly assume here that there is no ambiguity as to what rule is applied in a certain inference; one can always label the rules to avoid ambiguity of this kind.

Example In the rule Modus Ponens

$$\frac{A \qquad A \to B}{B}$$

the formula $A \to B$ is the only formula that is made active by this inference. The axiom $A \to (B \to A)$ makes the formulas $A \to (B \to A)$ and $(B \to A)$ active.

The intuition here is that in a proof the inactive formulas can be changed while the proof remains valid. This is the content of the next claim, of which we omit the proof.

Claim If A is not active in a proof P, then if A is everywhere replaced by B, the result is still a valid proof.

Claim Let c be the maximum number of connectives shown in any inference rule or axiom of a Frege system $\mathcal{F}$ (for $\mathcal{F}_0$, $c = 6$), and let $A_1, \ldots, A_k$ be all the distinct active formulas in an $\mathcal{F}$-proof P, then $|P| \geq \frac{1}{c}(\sum_{i=1}^{k} |A_i|)$.

Proof Observe that any inference activates at most c formulas, and that if a subformula of a formula is active, then so is the formula of which it is a subformula. Therefore, given a symbol in the proof P, it lies in at most c activated occurrences of $A_1, \ldots, A_k$. Moreover, every A_i is activated somewhere. This implies that $c \cdot |P| \geq \sum_{i=1}^{k} |A_i|$. Hence $|P| \geq \frac{1}{c}(\sum_{i=1}^{k} |A_i|)$.

Let φ_n be the formula

$$\bot \vee (\bot \vee (\ldots (\bot \vee \top) \ldots)$$

in which n $\bot$'s occur. $\top$ denotes “true”: $\top = x \vee \neg x$. $\bot$ denotes “falsum”: $\bot = x \wedge \neg x$ (sometimes these symbols are added to the language of propositional logic).

Claim In any proof of φ_n all the n subformulas of φ_n that are distinct from $\bot$ and $\top$ are active.

Proof If not, by Claim 14 we could replace such an inactive subformula by $\bot$ and obtain a valid formula as well. This cannot be, as such a formula would not be a tautology.

By the previous two claims, any Frege proof of φ_n has at least $\sum_{i=1}^{n} i$ symbols. Thus any Frege proof of φ_n has $\Omega(n^2)$ symbols.

Claim Let c be the maximum number of connectives shown in any inference rule or axiom of a Frege system $\mathcal{F}$. If an $\mathcal{F}$-proof P has k distinct active subformulas, then P has at least $\frac{k}{c}$ lines.

Thus any Frege proof of φ_n has $\Omega(n)$ lines. Hence any extended Frege proof of φ_n has size $\Omega(n)$.

15 Resolution

Resolution is an algorithm to prove formulas that are of a certain syntactic form. It arose in the 50's when people were looking for efficient theorem provers. The algorithm is simple; it has only one rule, the so-called Resolution Rule. The drawback is that certain formulas have long Resolution proofs compared to their Frege proofs. The Pigeonhole Principle is an example of this. In one of the next lectures we will see that it has no polynomial size Resolution proof, while it has polynomial size Frege proofs [1]. As said, Resolution can only be applied to formulas of a special kind. Namely, the formulas in Conjunctive Normal Form (CNF). A formula is said to be in CNF if it is the conjunction of disjunctions of variables and negated variables, i.e. if it is of the form

$$\bigwedge_i \left(\bigvee_j A_{ij} \right)$$

where the A_{ij}'s are of the form x or $\neg x$, for some variable x. Note that every formula can be written in CNF. For example,

$$\begin{array}{lcl} (x \rightarrow y) & \leftrightarrow & (\neg x \vee y) \\ (\neg(x \wedge y) \wedge z) & \leftrightarrow & ((\neg x \vee \neg y) \wedge z) \\ (x \vee (y \wedge z)) & \leftrightarrow & ((x \vee y) \wedge (x \vee z)) \end{array}$$

where the formulas at the right side are in CNF. Observe that in going to CNF the size of a formula can increase exponentially.

Definition We define what a Resolution Refutation is.
Syntax: variables $x_1, x_2, \ldots$
Literals: x_i, $\bar{x}_i$ (the intended meaning of $\bar{x}_i$ is $\neg x_i$). If x is a literal, then $\bar{x}$ is defined so that $\bar{\bar{x}} = x$: $\bar{x} = y$, when $x = \bar{y}$, and $\bar{x} = \bar{x}$ otherwise.
A *Clause* is a set of literals. The intended meaning of a clause is the disjunction of its members. We call a set of clauses Γ satisfiable if there exists a truth-assignment that satisfies all clauses in Γ.

Example

- $\{\bar{x}, y\}$ means $\neg x \vee y$.
- $\{x, \bar{x}\}$ is always valid.
- $\{\} = \emptyset$ is the unsatisfiable clause.

The *Resolution Rule*: (C and D denote clauses)

$$\frac{C \cup \{x\} \quad D \cup \{\bar{x}\}}{C \cup D}$$

The clause $C \cup D$ is called the *resolvent* of the rule. A *Resolution Refutation* of a set of clauses Γ consists of a sequence of clauses $C_1, \ldots, C_n$, where $C_n = \emptyset$, and for each $i \leq n$ either

1. $C_i \in \Gamma$
2. C_i is inferred by the Resolution Rule from C_j and C_h, for some $j, h < i$.

The idea behind a Resolution Refutation is the following. Assuming that Γ is satisfiable we infer other clauses that are also satisfiable till we end up with the empty clause. Since the empty clause is not satisfiable, the assumption that Γ is satisfiable is refuted. Thus Γ is not satisfiable. The following claims make this precise. W.l.o.g. we can disallow having both x and $\bar{x}$ in a clause in a proof.

Claim If the truth-assignment τ satisfies the clauses $C \cup \{x\}$ and $D \cup \{\bar{x}\}$, then τ satisfies the resolvent $C \cup D$.

Theorem 19 (Soundness Theorem) *If there exists a Resolution Refutation for Γ, then Γ is unsatisfiable.*

Proof Suppose Γ has a Resolution Refutation $C_1, \ldots, C_n$. If there would exist a truth-assignment τ that satisfies Γ, then by the previous claim τ would satisfy all C_i. Hence τ would satisfy the unsatisfiable empty clause C_n, quod non. Thus Γ is unsatisfiable.

Math 267a - Propositional Proof Complexity

Lecture #7: 6 February 2002

Lecturer: Sam Buss

Scribe Notes by: Dan Curtis

16 Completeness and Soundness of Resolution Proofs

16.1 Definition of a Resolution Proof

Recall the resolution rule:

$$\frac{C \cup \{x\} \;\; D \cup \{\bar{x}\}}{C \cup D}.$$

Definition A set of literals $\{x_1, \ldots, x_n\}$, with x_i in P_k or $\bar{P}_k$, is called a clause.

Definition Resolution refutes a set of clauses if and only all the clauses cannot be simultaneously satisfied.

A clause is a disjunction of literals and a set of clauses is a conjuction of clauses, which can be thought of as a conjuctive normal form formula. We can view resolution as proving disjunctive normal form formulas. For right now, resolution can prove tautologies that are in Disjunctive Normal Form.

Example The Pigeon hole principle (PHP_n^m) can be written as

$$\bigwedge_{i=1}^{m} \bigvee_{j=1}^{n} p_{ij} \rightarrow \bigvee_{i=1}^{m-1} \bigvee_{j=i+1}^{m} \bigvee_{k=1}^{n} (p_{ik} \wedge p_{jk}).$$

The negation of this ($\neg PHP_n^m$) is

$$\bigwedge_{i=1}^{m} \bigvee_{j=1}^{n} p_{ij} \wedge \bigwedge_{i=1}^{m-1} \bigwedge_{j=i+1}^{n} \bigwedge_{k=1}^{n} (\bar{p}_{ik} \wedge \bar{p}_{jk}).$$

which is in conjunctive normal form.

Written as a set of clauses:

$$\begin{array}{lll} \{p_{i,1}, \ldots, p_{i,n}\}, & i = 1, \ldots, m & \leftarrow m \text{ clauses} \\ \{\bar{p}_{ik}, \bar{p}_{jk}\}, & i = 1, \ldots, m-1;\ j = i+1, \ldots, n;\ k = 1, \ldots n & \leftarrow\approx m^2 \text{ clauses} \end{array}$$

A resolution "proof" of PHP means a refutation of this set of clauses.

16.2 Completeness Theorem

Theorem 20 *(Completeness Theorem) If $\mathcal{C}$ is an unsatisfiable set of clauses, then $\mathcal{C}$ has a resolution refutation.*

Proof Using induction on the number of variables in $\mathcal{C}$, assume $\mathcal{C}$ has zero variables. Then either $\mathcal{C} = \{\emptyset\}$, in which case it contains the refutation $\emptyset$, or $\mathcal{C} = \emptyset$ which is satisfiable. Thus the hypothesis holds for any clause with zero variables.

Now, let $\mathcal{C}$ be an unsatisfiable set of clauses and let x be a variable in some clause in $\mathcal{C}$. Define

$$\begin{aligned} \mathcal{C}_x &= \{\text{the set of clauses in } \mathcal{C} \text{ that contain } x\} \\ \mathcal{C}_{\bar{x}} &= \{\text{the set of clauses in } \mathcal{C} \text{ that contain } \bar{x}\} \\ \mathcal{C}' &= \mathcal{C} - (\mathcal{C}_x \cup \mathcal{C}_{\bar{x}}). \end{aligned}$$

Then resolve all $\mathcal{C}_x$ clauses with all $\mathcal{C}_{\bar{x}}$ clauses by

$$\frac{D \cup \{x\} \quad E \cup \{\bar{x}\}}{D \cup E}$$

Let $\mathcal{D} = \mathcal{C}' \cup \{$all resolvents of the form $D \cup E$, where $D \cup \{x\} \in \mathcal{C}_x$ and $E \cup \{\bar{x}\} \in \mathcal{C}_{\bar{x}}\}$

Since $\mathcal{D}$ has fewer variables than $\mathcal{C}$, then by the induction hypothesis, if $\mathcal{D}$ is unsatisfiable, then $\mathcal{D}$ has a refutation. Also, from the construction of $\mathcal{D}$, if $\mathcal{D}$ has a refutation, then $\mathcal{C}$ has a refutation. Thus, if we can show that $\mathcal{D}$ is unsatisfiable, then $\mathcal{C}$ has a refutation.

Suppose $\mathcal{D}$ is satisfiable and τ is a truth assignment that satisfies $\mathcal{D}$. Define τ^+ to be the same as τ with the addition that $\tau(x) = T$, and define τ^- to be the same as τ with the addition that $\tau(x) = F$.

Suppose τ^+ does not satisfy $\mathcal{C}$. Then there is a $E \cup \{\bar{x}\} \in \mathcal{C}$ such that τ^+ does not satisfy $E \cup \{\bar{x}\}$. But then τ does not satisfy E. Similarly, if τ^- does not satisfy $\mathcal{C}$, then there is a $D \cup \{x\}$ such that τ does not satisfy D. However, since τ satisfies $\mathcal{D}$, τ satisfies $D \cup E$. So, either τ^+ or τ^- satisfies $\mathcal{C}$.

16.3 Size of a Resolution Proof

The size of a resolution proof can be measured in two ways:

a) Total number of literals in all clauses.

b) Number of clauses.

Clearly, (b) $\leq$ (a) $\leq$ (b)$\cdot$(number of distinct variables), so a polynomial size bound on b) implies a polynomial size bound on a).

16.4 Subsumption Rule

Definition The subsumption rule (weakening rule), for any two clauses C and D with $C \subseteq D$, is given by

$$\frac{C}{D}.$$

Theorem 21 *A resolution and subsumption refutation of a set $\mathcal{C}$ of clauses can be converted into a smaller resolution refutation of $\mathcal{C}$.*

In practice, a theorem prover has $C_1, \ldots, C_k$ as input clauses and generates clauses with resolution. At some point, if it has clauses D and E with $E \subseteq D$, then it is alright to discard D without any negative consequences.

Proof Let $\phi_1, \ldots, \phi_k = \emptyset$ be a refutation using resolution and subsumption. A new refutation $\psi_1, \ldots, \psi_k = \emptyset$, built recursively in the following way using only resolution, will have the property that $\psi_i \subseteq \phi_i$ for each $i \leq k$.

For each $i \leq k$, define ψ_i as follows:

1) If $\phi_i \in \mathcal{C}$, then set $\psi_i = \phi_i$. In this case, clearly $\psi_i \subseteq \phi_i$.

2) If ϕ_i is inferred by subsumption $\frac{\phi_l}{\phi_i}$ for some $l \leq i$, with $\phi_l \subseteq \phi_i$, then set $\psi_i = \psi_l$. Here, we have $\psi_i = \psi_l \subseteq \phi_l \subseteq \phi_i$.

3) If ϕ_i is inferred by resolution, for some $j, l \leq i$,

$$\frac{\phi_j \ \phi_l}{\phi_i}$$

resolving on $x \in \phi_j$ and $\bar{x} \in \phi_l$, do the following:

   a) If $x \notin \psi_j$, set $\psi_i = \psi_j \subseteq \phi_i$.
   b) If $\bar{x} \notin \psi_l$, set $\psi_i = \psi_l \subseteq \phi_i$.
   c) Otherwise, set $\psi_i = \mathrm{res}_x(\psi_j, \psi_l)$, where res_x is defined to be the resolvent obtained by the resolution using the literal x. Since $\psi_j \subseteq \phi_j$ and $\psi_l \subseteq \phi_l$, then $\psi_i \subseteq \phi_i$.

Clearly, $\psi_k = \emptyset$, since $\psi_k \subseteq \phi_k = \emptyset$. Finally, erase any duplicate ψ_i's.

16.5 Refutation Proof of the Pigeon Hole Principle

As a point of notation, throughout this proof, we will use $[k]$ to denote the set $\{1, \ldots, k\}$.

Recall that the negation of the Pigeon Hole Principle can be written as:

$$\bigwedge_{i=1}^{m} \bigvee_{j=1}^{n} p_{ij} \wedge \bigwedge_{i=1}^{m-1} \bigwedge_{j=i+1}^{n} \bigwedge_{k=1}^{n} (p_{ik} \wedge p_{jk}).$$

For this proof, we will prove the special case PHP_n^{n+1} (i.e. $m = n+1$). Writing this as a set of clauses, we get

$$\mathcal{C} = \{\{P_{i,1}, \ldots, P_{i,n}\}, 1 \leq i \leq n\} \cup \{\{\bar{P}_{i,k}, \bar{P}_j, k\}, 1 \leq i \leq j \leq m; 1 \leq k \leq n\}$$

Proof The refutation will proceed in a series of stages, $s = n, n-1, \ldots, 0$. At stage s, we have the following clauses: For each injective map $\pi : \{1, \ldots, s\} \to \{1, \ldots, n\}$ we have the clause $\{\bar{P}_{1,\pi(1)}, \bar{P}_{2,\pi(2)}, \ldots, \bar{P}_{s,\pi(s)}\}$.

At stage $s = 0$, the only map is $\pi : \emptyset \to [n]$ and the clause is $\emptyset$.

At stage $s = n$, for any injective map $\pi : [n] \to [n]$, start with the initial clause $\{P_{n+1,1}, \ldots, P_{n+1,n}\}$ and resolve with the initial clauses $\{\bar{P}_{i,\pi(i)}, \bar{P}_{n+1,\pi(i)}\}$ for each $1 \leq i \leq n$.

For the induction step, assume we have the stage $s+1$ clauses. Given any injective map $\pi : [s] \to [n]$ we need to derive $\{\bar{P}_{1,\pi(1)}, \bar{P}_{2,\pi(2)}, \ldots, \bar{P}_{s,\pi(s)}\}$. For $j \notin \text{Range}(\pi)$, define π_j to be $\pi \cup \{(s+1) \mapsto j\}$. Since $\pi_j : [s+1] \to [n]$, then from stage $s+1$ we already have

$$(*_j) \qquad \{\bar{P}_{1,\pi(1)}, \bar{P}_{2,\pi(2)}, \ldots, \bar{P}_{s,\pi(s)}, \bar{P}_{s+1,j}\}.$$

To derive the stage s clauses, start with the initial clause $\{P_{s+1,1}, \ldots, P_{s+1,n}\}$ and resolve with the initial clauses $\{\bar{P}_{i,\pi(i)}, \bar{P}_{s+1,\pi(i)}\}$ for each $1 \leq i \leq s$. After resolving with each of the s clauses, we get

$$\{\bar{P}_{1,\pi(1)}, \bar{P}_{2,\pi(2)}, \ldots, \bar{P}_{s,\pi(s)}, P_{s+1,j_1}, \ldots, P_{s+1,j_{n-s}}\}$$

where $[n] - \text{Range}(\pi) = \{j_1, \ldots, j_{n-s}\}$. Finally, resolve with the $(*_j)$ clauses for $j = j_1, \ldots, j_{n-s}$ and we get $\{\bar{P}_{1,\pi(1)}, \bar{P}_{2,\pi(2)}, \ldots, \bar{P}_{s,\pi(s)}\}$ as desired.

16.6 Size of Proof of Pigeon Hole Principle

There are n stages for this proof of PHP_n^{n+1}. At each stage, there are on the order of $O(n^s)$ injective maps $\pi : [s] \to [n]$. Also, there are n steps required to derive each clause. Thus, the size of this proof is on the order of $O(n \cdot n \cdot n^n) = 2^{O(n \log n)}$ total number of clauses.

However, a more honest measure of the size of the proof is in terms of the number of variables $v = \Omega(n^2)$. In terms of v, the size of the proof is on the order $2^{O(\sqrt{v} \log \sqrt{V})} = 2^{O(\sqrt{v} \log V)}$.

16.7 Soundness Theorem

Theorem 22 *(Soundness Theorem) If $\mathcal{C}$ is a set of clauses with a refutation, then $\mathcal{C}$ is unsatisfiable.*

Proof Proof of the soundness theorem is deferred until the next lecture.

Math 267a - Propositional Proof Complexity

Lecture #8: 11 February 2002

Lecturer: Sam Buss

Scribe Notes by: Bryant Forsgren

17 Last Time

In Lecture 7 we proved the "strong" Pigeon Hole Principle (PHP_n^{n+1}) by giving a resolution refutation of its negation. The refutation was tree-like and had size $2^{O(n \log n)}$. We claim without proof that a non-tree-like refutation of size $2^{O(n)}$ exists. Today our goal is to prove exponential lower bounds ($2^{\Omega(n)}$) on the size of any refutation of the $\neg PHP_n^{n+1}$ clauses.

18 Views of Resolution Refutations

18.1 Resolution Proof as a Decision dag

Any resolution proof starts with a set of initial clauses $C_1, C_2, \ldots C_k$, and ends with the empty clause $\emptyset$. Clearly all of the variables need to be eliminated to reach this conclusion. For example, the variable x can be eliminated from two clauses of the form $D \cup \{x\}$ and $E \cup \{\overline{x}\}$ to derive $D \cup E$. In particular, there exists some variable y such that the empty clause $\emptyset$ is derived from $\{y\}$ and $\{\overline{y}\}$. We can view this pictorially as follows:

$$\begin{array}{c}
C_1 \qquad \ldots \qquad C_k \\
\ddots \quad \vdots \quad \iddots \\
\dfrac{D \cup \{x\} \quad E \cup \{\overline{x}\}}{D \cup E} \\
\ddots \quad \vdots \quad \iddots \\
\dfrac{\{y\} \ \{\overline{y}\}}{\emptyset}
\end{array}$$

Given any such refutation and a truth assignment τ, it is clear that there exists an initial clause C_i such that τ falsifies C_i. We wish to use the refutation as a decision dag to find such a C_i. We start with $\emptyset$, and work toward the initial clauses, making a decision at each clause we encounter. Suppose we are at the clause $D \cup E$ in the diagram. We do the following:

```
If $\tau(x) = \top$
    go to $E \cup \{\overline{x}\}$
Else
    go to $D \cup \{x\}$
```

Our invariant is the following: we are always at a clause C which is falsified by τ. Furthermore, we are guaranteed to eventually reach one of the initial clauses C_i. By our easily verified invariant, C_i is an initial clause which is falsified by τ.

18.2 Resolution Proof as Guiding a Game

The game is played between a Prover and an Adversary . The Prover wishes to find a clause that is false, and the adversary wishes to prevent this from happening. A round of the game is played as follows:

1. Prover asks a query "y?".
2. Adversary answers *True* ($\top$) or *False* ($\bot$).
3. Prover remembers the answer (but is allowed to forget later).

Claim There is an exact correspondence between resolution proofs and winning strategies for the Prover.

This is true because at any particular point in the game, the Prover and Adversary are at some clause in the refutation. This clause contains exactly those literals $\overline{y}$ such that the Prover knows y holds.

19 Exponential Lower Bounds on Refutation Proofs of the Pigeon Hole Principle

19.1 The "weak" Pigeon Hole Principle

We now define the "weak" Pigeon Hole Principle. Intuitively, it states that

$$\forall m > n \nexists f : [m] \overset{1-1}{\to} [n]$$

over the natural numbers.

Definition Let $m > n$; $m, n \in \mathbb{N}$

$$PHP_n^m : \bigwedge_{i=1}^{m} \bigvee_{j=1}^{n} P_{i,j} \to \bigvee_{i=1}^{m-1} \bigvee_{j=i+1}^{m} \bigvee_{k=1}^{n} (P_{i,k} \wedge P_{j,k})$$

The clauses of $\neg PHP_n^m$ are as follows:

$$\{P_{i,1}, \ldots, P_{i,n}\} \quad \text{for } i = 1, \ldots, m$$

$$\{\overline{P_{i,k}}, \overline{P_{j,k}}\} \qquad \text{for } 1 \leq i < j \leq m, 1 \leq k \leq n$$

Note that it is often easier to prove PHP_n^m for $m >> n$, than for $m = n + 1$.

Definition The *width* of a refutation R is $\max\{|C| : C \text{ is a clause in } R\}$.

Note that the refutation of $\neg PHP_n^{n+1}$ from the previous lecture had width $O(n)$.

Theorem 23 (Dantchev 2002) *Let $m > n >> 0$. Then any resolution refutation of $\neg PHP_n^m$ of width $\leq \frac{n^2}{32}$ has size $\geq 2^{\frac{n}{8}}$ (where size is understood to mean the number of clauses in the proof).*

Proof Suppose we have a refutation R of width $\leq \frac{n^2}{32}$ and size $< 2^{\frac{n}{8}}$, for "large enough" n. Let $H_1 = \{1, \ldots, \lfloor \frac{n}{2} \rfloor\}$, $H_2 = \{\lfloor \frac{n}{2} \rfloor + 1, \ldots, n\}$. Now fix a π which maps each pigeon i to either H_1 or H_2. Denote this by $i \in H_{\pi(i)}$.

Definition Pigeon i is *busy* if either:

(1) The Prover knows $P_{i,j} = \top$ for some $j \in H_{\pi(i)}$. (call this case busy$_1$)

(2) The Prover knows $P_{i,j} = \bot$ for $\geq \frac{n}{4}$ many $j \in H_{\pi(i)}$. (call this case busy$_2$)

As described above, the Prover views R as a decision dag and chooses the queries accordingly. When the Prover queries a variable $P_{i,j}$, the Adversary responds as follows:

(1) If $j \notin H_{\pi(i)}$, Adversary answers "$\bot$".

(2) If $j \in H_{\pi(i)}$ and i is not busy, Adversary answers "$\bot$".

(3) Otherwise ($j \in H_{\pi(i)}$ and i is busy), the Adversary chooses an unassigned hole $k \in H_{\pi(i)}$ for which $P_{i,k}$ is not known and assigns pigeon i to that hole. The Adversary then answers accordingly, and remembers this assignment until (if ever) pigeon i becomes unbusy.

Claim The Adversary can keep going as long as there are $< \frac{n}{4}$ busy pigeons.

The game stops when there are $\geq \frac{n}{4}$ busy pigeons at some clause C_π. By assumption, C_π has width $\leq \frac{n^2}{32}$ and has $\frac{n}{4}$ busy pigeons.

Notice that each pigeon of type busy$_2$ contributes $\frac{n}{4}$ literals into C_π. Suppose C_π has $> \frac{n}{8}$ pigeons which are busy$_2$. Then C_π has width $> \frac{n^2}{32}$ which is a contradiction. Therefore, at most $\frac{n}{8}$ of the $\frac{n}{4}$ busy pigeons can be busy$_2$.

So at least $\frac{n}{8}$ i's in C_π are of type busy$_1$. In other words, for at least $\frac{n}{8}$ i's there exists a $j \in H_{\pi(i)}$ such that $\overline{P_{i,j}} \in C_\pi$. We wish to address the following question: "For how many π's can *this* clause be C_π?" But this is only possible for $\leq 2^{(m - \frac{n}{8})}$ many π's. So there are $\geq 2^{\frac{n}{8}}$ distinct C_π's, contradicting the assumption that size $< 2^{\frac{n}{8}}$.

19.2 The "strong" Pigeon Hole Principle

Definition A *restriction* is a partial truth assignment that maps some variables to $\{\top, \bot\}$, leaving other variables unassigned ($*$). A restriction can be expressed in the following way:

$$\rho(x) = \begin{cases} \top & \text{if } Cond_A(x) \\ \bot & \text{if } Cond_B(x) \\ * & \text{if } Cond_C(x) \end{cases}$$

Where each $Cond_i$ is an arbitrary condition.

Definition If Σ is a set of clauses, $\Sigma_{\restriction\rho}$ is the set of clauses constructed as follows:

```
    **Foreach** $C = \{x_1, \ldots, x_k\} \in \Sigma$
        **If** $\exists i$ such that $\rho(x_i) = \top$
            discard $C$
        **Else**
            put $\{x_i : \rho(x_i) = *\}$ into $\Sigma_{\restriction\rho}$
```

Theorem 24 *If R is a refutation of Σ, then $R_{\restriction\rho}$ is a refutation of $\Sigma_{\restriction\rho}$ (to be precise, it is a resolution and subsumption refutation).*

What this means is that size and width do not increase under restrictions.

Theorem 25 *For any $\alpha \in (0, \frac{n}{8})$, any refutation of $\Sigma = \neg PHP_n^{n+1}$ has size $\geq 2^{\epsilon n}$ where $\epsilon = \frac{1}{8} - \alpha$ (for large enough n).*

Proof Assume there is a refutation R of size $< 2^{\epsilon n}$. We construct a restriction ρ as follows:

```
    Fix $\beta \in (0,1)$
        (note that $\alpha$ and $\beta$ satisfy this relationship: $\beta = 1 - 8\alpha$)
    **Foreach** pigeon $i$
        pick $i$ with probability $1 - \beta$
        **If** pigeon $i$ is picked
            map it to a unique, randomly selected hole $j_i$
            set $\rho(P_{i,j_i}) = \top$
            **Foreach** $k \neq j_i$
                set $\rho(P_{i,k}) = \bot$
            **Foreach** $k \neq i$
                set $\rho(P_{k,j_i}) = \bot$
```

We apply this restriction to Σ, yielding $\Sigma_{\restriction\rho}$. The expected number of holes in $\Sigma_{\restriction\rho}$ is

$$n - (1-\beta)(n+1) = \beta n - 1 + \beta$$

So with some fixed non-zero probability, the number of remaining holes is at least βn. We also apply ρ to the refutation R which yields $R_{\restriction\rho}$, a refutation of $\neg PHP_{\lceil\beta n\rceil}^{\lceil\beta n\rceil+1}$ of size $\leq 2^{\epsilon n}$.

Claim $R_{\restriction\rho}$ has width $\leq \frac{(\beta n)^2}{32}$ with probability approaching 1 as $n \to \infty$. This will be a contradiction, provided $\epsilon > \frac{\beta}{8}$.

This last claim will be proved next time, finishing the proof of Theorem 3. The idea is that any clause in R will get mapped to $\top$ by ρ and vanish from $R_{\restriction\rho}$.

Math 267a - Propositional Proof Complexity

Lecture #9: 13 February 2002

Lecturer: Sam Buss

Scribe Notes by: Nathan Segerlind

20 Last Time

In this lecture, we prove exponential an lower bound on the sizes of resolution refutations of PHP_n^{n+1}. We also discuss recent results on the size needed to prove certain formulations of circuit lower bounds in resolution.

21 A Size Lower Bound for Resolution Refutations of PHP_n^{n+1}

The main result of this lecture is the proof that resolution refutations of PHP_n^{n+1} require size $2^{\Omega(n)}$. We show that clauses of high width are likely to be satisfied by a random restriction selected according to the distribution given in lecture 8. These restrictions transform the refutation into a refutation of a slightly smaller instance of the pigeonhole principle, and by the results of lecture 8, this new refutation must have either large width or large size. Because the restricted refutation has small width, it must have large size. Therefore, the original refutation must also have large size.

Recall that the width of a clause is the number of variables appearing in the clause, and the width of a resolution refutation is the maximum width of a clause in the refutation. We also assume that in a resolution refutation, there are no clauses that contain both a variable and its negation.

First, we show that clauses of high width must have many pigeons that appear in many literals.

Definition Let $s > 0$ be given. Let C be a clause. Let $i \in [n]$ be a pigeon. We say that i is an s-heavy pigeon of C if $|\{j \in [n] | X_{i,j} \in C \vee \neg X_{i,j} \in C\}| \geq s$.

Lemma 26 *Let n be an integer strictly greater than 1. For any clause C and any $\gamma > 0$, if C has width at least γn^2, then C contains at least $\frac{\gamma n}{2}$ many $\frac{\gamma n}{2}$-heavy pigeons.*

Proof Let h be the number of $\frac{\gamma n}{2}$-heavy pigeons in C.

First, we show that $h \geq 1$. Because each pigeon that is not $\frac{\gamma n}{2}$-heavy can contribute at most $\frac{\gamma n}{2}$ literals to C, if every pigeon were not $\frac{\gamma n}{2}$-heavy, then there would be at most $(n+1)\frac{\gamma n}{2} = \frac{\gamma n^2}{2} + \frac{\gamma n}{2}$ many literals in C. Because $n > 1$, this is quantity is less than $\gamma n^2/2$ and we would have a contradiction to the fact C has width at least γn^2.

Moreover, each $\frac{\gamma n}{2}$-heavy pigeon can contribute at most n literals to C, so we have the following inequalities.

$$\gamma n^2 \leq (n+1-h)\frac{\gamma n}{2} + hn$$
$$\gamma n^2 \leq \frac{\gamma n^2}{2} + hn$$
$$\frac{\gamma n^2}{2} \leq hn$$
$$\frac{\gamma n}{2} \leq h$$

Recall the definition on partial assignments given in lecture 8: for a fixed parameter $\beta \in (0,1)$, we choose to match each pigeon from $[n]$ with independent probability $1-\beta$. Then, we uniformly choose a matching between the selected pigeons and the holes. In the sequel of this lecture, this distribution will be referred to as $\mathcal{R}_{n,\beta}$.

We now show that a clause that has many heavy pigeons is very likely to be satisfied by a restriction chosen according to $\mathcal{R}_{n,\beta}$.

Lemma 27 *If C is a clause that contains t many αn-heavy pigeons, then*

$$Pr_{\rho\in\mathcal{R}_{n,\beta}}\left[C\restriction_\rho \neq 1\right] \leq (1-(1-\beta)\alpha/2)^{t-1}$$

Proof Let F be the set of pigeons from $[1,n]$ that are αn-heavy in C. Notice that $|F| \geq t-1$. For each $i \in F$, let H_i be the set of holes so that the variable $X_{i,j}$ occurs in C. For each $i \in F$, $j \in H_i$ notice that exactly one of $X_{i,j}, \neg X_{i,j}$ occurs in C: let $l_{i,j}$ be this literal. In this notation, C can be written as $\bigvee_{i\in F}\bigvee_{j\in H_i} l_{i,j} \vee C'$. where C' is a clause that contains only literals whose pigeons are not in F. Furthermore, we may assume without loss of generality that F consists of the pigeons 1 through $|F|$ because permuting the first n pigeons does not change the distribution $\mathcal{R}_{n,p}$.

For each $i \in F$, let E_i be the event that $\left(\bigvee_{\substack{k\in F\\k<i}}\bigvee_{j\in H_k} l_{k,j}\right)\restriction_\rho \neq 1$.

Now we bound, for each $i \in F$, the probability that, conditioned on E_i, $\rho \in \mathcal{R}_{n,\beta}$ satisfies $\bigvee_{j\in H_i} l_{i,j}$.

First of all, if $\bigvee_{j\in H_i} l_{i,j}$ contains two or more negative literals, then ρ satisfies $\bigvee_{j\in H_i} l_{i,j}$ if and only if ρ matches the pigeon i to some hole, and this occurs with probability $1-\beta$.

If $\bigvee_{j\in H_i} l_{i,j}$ contains no negative literals, then at worst the preceding elements of F were each matched to an element of H_i, and the chance of satisfying $\bigvee_{j\in H_i} l_{i,j}$ is at most $(1-\beta)\left(\frac{|H_i|-i+1}{n-i+1}\right)$. Because $|H_i| \geq \alpha n$ and $t \leq \alpha n/2$, we have that this probability exceeds $(1-\beta)\alpha/2$.

If $\bigvee_{j\in H_i} l_{i,j}$ contains exactly one negative literal, then $\bigvee_{j\in H_i} l_{i,j}$ is satisfied with the probability that ρ matches i to some hole besides the forbidden hole. At the very worst, ρ did not match any of the preceding pigeons of F to the forbidden hole, so the probability of satisfaction is at least $(1-\beta)(1-1/(n-i+1))$. This is equal to $(1-\beta)(n-i)/(n-i+1)$ which is at least $(1-\beta)\alpha/2$.

Therefore, for any $i \in F$, we have the following inequalities:

$$\Pr_{\rho\in\mathcal{R}_{n,\beta}}\left[(\bigvee_{j\in H_i} l_{i,j})\restriction_\rho = 1 \mid E_i\right] \geq (1-\beta)\alpha/2$$

$$\mathrm{Pr}_{\rho\in\mathcal{R}_{n,\beta}}\left[(\bigvee_{j\in H_i} l_{i,j})\restriction_\rho \neq 1 \mid E_i\right] \leq 1-(1-\beta)\alpha/2$$

Examination of the conditional probabilities of satisfying the literals involved with each heavy pigeon reveals the following.

$$\mathrm{Pr}_{\rho\in\mathcal{R}_{n,\beta}}\left[C\restriction_\rho \neq 1\right] \leq \mathrm{Pr}_{\rho\in\mathcal{R}_{n,\beta}}\left[(\bigvee_{i\in F}\bigvee_{j\in H_i} l_{i,j})\restriction_\rho \neq 1\right] = \prod_{i\in F}\mathrm{Pr}_{\rho\in\mathcal{R}_{n,\beta}}\left[(\bigvee_{j\in H_i} l_{i,j})\restriction_\rho \neq 1 \mid E_i\right]$$

$$\leq \prod_{i\in F}(1-(1-\beta)\alpha/2) = (1-(1-\beta)\alpha/2)^{t-1}$$

We now show that a random restriction will almost certainly satisfy every wide clause of a small proof.

Lemma 28 *Let $\epsilon,\beta\in(0,1)$ be a constants so that $\epsilon < -(\beta^2/64)\log_2(1-(1-\beta)\beta^2/128)$. For n sufficiently large, if R is resolution refutation of PHP_n^{n+1} of size at most $2^{\epsilon n}$, then the following inequality holds.*

$$Pr_{\rho\in\mathcal{R}_{n,\beta}}\left[w(R\restriction_\rho)\geq \beta^2 n^2/32\right] = o(1)$$

Proof For each clause C of R that has width at least $\beta^2n^2/32$, by lemma 26, C contains at least $\beta^2 n/64$ many $\beta^2 n/64$-heavy pigeons. Therefore, by lemma 27, each clause C of R of width at least $\beta^2n^2/32$ is not satisfied with probability at most $(1-(1-\beta)\beta^2/128)^{(\beta^2n/64)-1}$. Therefore, by an application of the union bound, we have the following inequality.

$$\mathrm{Pr}_{\rho\in\mathcal{R}_{n,\beta}}\left[w(R\restriction_\rho)\geq\beta^2n^2/32\right] \leq 2^{\epsilon n}(1-(1-\beta)\beta^2/128)^{(\beta^2n/64)-1} = 2^{\epsilon n}2^{((\beta^2n/64)-1)\log_2(1-(1-\beta)\beta^2/128)}$$

$$= 2^{n(\epsilon+((\beta^2/64)-1/n)\log_2(1-(1-\beta)\beta^2/128))}$$

Because ϵ and β are constants with $\epsilon < -(\beta^2/64)\log_2(1-(1-\beta)\beta^2/128)$, this probability is $o(1)$ as n tends to infinity.

We now combine these lemmas with Danchev's theorem to prove the size lower bounds for resolution refutations of PHP_n^{n+1}.

Theorem 29 *There exists an $\epsilon > 0$ so that every resolution refutation of PHP_n^{n+1} has size at least $2^{\epsilon n}$.*

Proof

We will show that for constants $\epsilon,\beta\in(0,1)$ with $\epsilon<\beta/8$ and $\epsilon < -(\beta^2/64)\log_2(1-(1-\beta)\beta^2/128)$ there is no resolution refutation of PHP_n^{n+1} of size $2^{\epsilon n}$. There are constants satisfying these bounds because for any $\beta\in(0,1)$, $-\log_2\left(1-(1-\beta)\beta^2/128\right)>0$ and therefore we can take ϵ to be the minimum of $\beta/8$ and $-(\beta^2/64)\log_2(1-(1-\beta)\beta^2/128)$, and we will have that $\epsilon\in(0,1)$.

For the sake of contradiction, assume that R is a resolution refutation of PHP_n^{n+1} of size less than $2^{\epsilon n}$.

Let $\rho \in \mathcal{R}_{n,\beta}$ be given. Let M be the set of pigeons matched by ρ, and let $m = |M|$. Notice that for the set of clauses PHP_n^{n+1}, the set of clauses $PHP_n^{n+1} \restriction_\rho$ is is just a renaming of PHP_{n-m}^{n-m+1}.

Because the number of pigeons matched by $\rho \in \mathcal{R}_{n,\beta}$ is distributed according to a binomial distribution, the expected number of pigeons matched by ρ is $(1-\beta)n$. By the central limit theorem as n tends to infinity, the probability that the number of matched pigeons exceeds $(1-\beta)n$ tends to $1/2$. Therefore, for sufficiently large n, the probability that ρ leaves at least $\beta n + 1$ many pigeons unmatched is at least $1/4$.

Lemma 28 tells us that $\Pr_{\rho \in \mathcal{R}_{n,\beta}} \left[w(R \restriction_\rho) \geq \beta^2 n^2/32 \right]$ is $o(1)$

Therefore, for sufficiently large n, we may choose $\rho \in \mathcal{R}_{n,\beta}$ so that $w(R \restriction_\rho) < \beta^2 n^2/32$ and ρ leaves at least βn many pigeons unmatched.

Because the restriction of a resolution refutation is a resolution refutation (see lecture 8), $R \restriction_\rho$ is a resolution refutation of $PHP_n^{n+1} \restriction_\rho$. Therefore, up to renaming the variables, $R \restriction_\rho$ is a resolution refutation of $PHP_{\beta n}^{\beta n+1}$ with each clause of width strictly less than $\beta^2 n^2/32$ and of size at most $2^{\epsilon n}$. By Danchev's theorem, every resolution refutation of $PHP_{\beta n}^{\beta n+1}$ requires width at least $\beta n^2/32$ or size at least $2^{\beta n/8}$, but because $\epsilon < \beta/8$, we have obtained a contradiction.

22 Lower Bounds for Resolution Proofs of Circuit Lower Bounds

Recently, Razborov has shown that certain formulations of circuit lower bounds require exponential size refutations in resolution. Given the truth-table of a boolean function, $f_n : \{0,1\}^n \to \{0,1\}$, and a a parameter $t \leq 2^n$, a CNF $Circuit_t(f_n)$ is constructed which is satisfiable if and only if f_n can be computed by a circuit of size t. If f_n is a function which is not computed by any circuit of size $\leq t$, then this formula is unsatisfiable. The formula is constructed in a brute force way, with $O(t)$ many variables encoding the circuit, and with $O(t2^n)$ many variables representing the value of each gate on each assignment to the inputs. The clauses state that the output of each gate is consistent with the output of its inputs.

Razborov's result shows that for any function f_n and size t, $Circuit_t(f_n)$ has no resolution refutation of size less than $2^{\Omega(t/n^3)}$. The proof works by reducing the onto-functional weak pigeonhole principle to the principle $Circuit_t(f_n)$.

Math 267a - Propositional Proof Complexity

Lecture #11: 27 February 2002

Lecturer: Sam Buss

Scribe Notes by: Liz Arentzen

23 Monotone Craig Interpolation

Monotone Craig Interpolation provides another way to obtain exponential lower bounds on Resolution proofs.

23.1 Propositional Case

Theorem 30 *Let $\phi = \phi(\vec{p}, \vec{q})$ be a formula in which the $\vec{p}$ variables occur only positively. Also, suppose that $\models \phi \to \psi$ where $\psi(\vec{p}, \vec{r})$. Then there exists an interpolant $C(\vec{p})$ such that the $\vec{p}$ variables occur only positively in C.*

Formulae are built with $\land$, $\lor$, $\neg$.

Definition An occurrence is positive if and only if it is under the scope of an even number of $\neg$ signs.

Note that application of De Morgan's Laws or the distributive laws does not affect whether a particular occurrence is positive.

Lemma 31 *Let the variables in $\vec{p}$ be $p_1, p_2, \ldots, p_k$ and let the variables $\vec{p}\,'$ be $p'_1, p'_2, \ldots, p'_k$. If $p_i \to p'_i$ is true $\forall i$ for some truth assignment, then $C(\vec{p})$ is true $\Rightarrow C(\vec{p}\,')$ is true, assuming that the variables of $\vec{p}$ occur only positively in C. (This last property is called* monotonicity.*)*

Proof By induction on size of C.

Proof (of Theorem 1) $\phi(\vec{p}, \vec{q}) \to \psi(\vec{p}, \vec{r})$ is the same as $\exists \vec{q}\ \phi(\vec{p}, \vec{q}) \to \forall \vec{r}\ \psi(\vec{p}, \vec{r})$. Then let

$$C(\vec{p}) \doteq \bigvee_{\substack{\text{all T/F settings} \\ \text{of the variables in } \vec{q}}} \phi(\vec{p}, \vec{q}). \tag{8}$$

The p's occur only positively so we can see that this interpolant has the proper form.

Alternately, it can be shown that a fitting interpolant is

$$C(\vec{p}) \doteq \bigwedge_{\substack{\text{all T/F settings} \\ \text{of the variables in } \vec{r}}} \psi(\vec{p}, \vec{r}). \tag{9}$$

23.2 Resolution Case

Theorem 32 *Let $\Gamma = \Gamma(\vec{p}, \vec{q})$ be a set of clauses, and let $\Delta = \Delta(\vec{p}, \vec{r})$ be a set of clauses.*

Assume that the $\vec{p}$ variables occur only positively in Δ (that is, there are no $\neg p_i$'s in clauses in Δ), or assume that the $\vec{p}$ variables occur only negatively in Γ (that is, there are no p_i's in clauses in Γ). Also assume $\Gamma \cup \Delta$ is unsatisfiable (i.e. has a refutation). Then there is an interpolant $C(\vec{p})$ such that $\forall$ truth assignments τ

if $\bar{\tau}(C(\vec{p})) = T$, then $\exists$ clause $C \in \Gamma$ such that $\tau(C) =$False,

if $\bar{\tau}(C(\vec{p})) = F$, then $\exists$ clause $C \in \Delta$ such that $\tau(C) =$False,

and such that the $\vec{p}$ variables occur only positively in C.

Proof

Let

$$C(\vec{p}) \doteq \bigwedge_{\substack{\text{all T/F settings}\\ \text{of the variables in } \vec{q}}} \bigvee_{C \in \Gamma} \neg C(\vec{p}, \vec{q}). \tag{10}$$

That $C(\vec{p})$ will be a fitting interpolant is immediate from the fact that if $\Delta \cup \Gamma$ is unsatisfiable

$$\bigwedge_{C \in \Delta} C \Rightarrow \bigvee_{C \in \Gamma} \neg C. \tag{11}$$

Alternately we may let

$$C(\vec{p}) \doteq \bigvee_{\substack{\text{all T/F settings}\\ \text{of the variables in } \vec{r}}} \bigwedge_{C \in \Delta} C(\vec{p}, \vec{r}). \tag{12}$$

Theorem 33 *Let R be a refutation of $\Gamma \cup \Delta$ of size s. Then $C(\vec{p})$ can be written with monotone circuit size $O(s)$. If R is tree-like, $C(\vec{p})$ has monotone formula size $O(s)$.*

Theorem 34 *(Restatement of part of Theorem 33) Let Γ consist of clauses containing $\vec{q}$'s and negative occurrences of $\vec{p}$'s. Let Δ consist of clauses containing $\vec{r}$'s and $\vec{p}$'s. (Note that we make no assumption on whether these $\vec{p}$ occur positively or negatively in the clauses of Δ.) If R is a refutation of $\Gamma \cup \Delta$, then there exists an interpolant ϕ, such that the size of ϕ is $O($number of steps in $R)$.*

Definition For C a clause in R, we let ϕ_C be defined by

1. $\phi_C = T$ if $C \in \Gamma$
2. $\phi_C = F$ if $C \in \Delta$
3. $\phi_C = \phi_{C_1 \cup \{p_i\}} \vee (p_i \wedge \phi_{C_2 \cup \{\bar{p}_i\}})$ if C is such that
$$\frac{C_1 \cup \{p_i\} \quad C_2 \cup \{\bar{p}_i\}}{C = C_1 \cup C_2}$$
4. $\phi_C = \phi_{C_1 \cup \{q_i\}} \wedge \phi_{C_2 \cup \{\bar{q}_i\}}$ if C is such that
$$\frac{C_1 \cup \{q_i\} \quad C_2 \cup \{\bar{q}_i\}}{C = C_1 \cup C_2}$$

5. $\phi_C = \phi_{C_1 \cup \{r_i\}} \vee \phi_{C_2 \cup \{\bar{r}_i\}}$ if C is such that

$$\frac{C_1 \cup \{r_i\} \quad C_2 \cup \{\bar{r}_i\}}{C = C_1 \cup C_2}$$

Definition For C a clause, let $C^\Gamma = C \cap \{q_i, \bar{q}_i, \bar{p}_i : i \geq 0\}$ and let $C^\Delta = C \cap \{p_i, \bar{p}_i, r_i, \bar{r}_i : i \geq 0\}$.

Claim For all τ, $\forall C \in R$,

A. if $\tau \not\models C^\Gamma$ and $\tau(\phi_C) = T$, then $\exists D \in \Gamma$ such that $\tau \not\models D$.

B. if $\tau \not\models C^\Delta$ and $\tau(\phi_C) = F$, then $\exists D \in \Delta$ such that $\tau \not\models D$.

Proof of this claim will imply that $\phi_\emptyset$ works as an interpolant, since $\emptyset^\Gamma = \emptyset = \emptyset^\Delta$, which is not satisfied by any τ.

Proof (of Claim)
Proof is by induction on inferences in R.

1. $C \in \Gamma$. Then $\phi_C = T$. $C \in \Gamma \Rightarrow C^\Gamma = C \cap \{q_i, \bar{q}_i, \bar{p}_i : i \geq 0\} = C$. Then if $\tau \not\models C^\Gamma$, $\tau \not\models C$, and so trivially $\exists C \in \Gamma$ such that $\tau \not\models C$.

2. $C \in \Delta$. Then $\phi_C = F$. $C \in \Delta \Rightarrow C^\Delta = C$. Hence if $\tau \not\models C^\Delta$, trivially $\exists C \in \Delta$ such that $\tau \not\models C$.

3. $C = C_1 \cup C_2$ with $\frac{C_1 \cup \{p_i\} \quad C_2 \cup \{\bar{p}_i\}}{C = C_1 \cup C_2}$. Then $\phi_C = \phi_{C_1 \cup \{p_i\}} \vee (p_i \wedge \phi_{C_2 \cup \{\bar{p}_i\}})$.

   (a) Suppose $\tau \not\models C^\Gamma$ and $\tau(\phi_C) = T$. Note that $(C_1 \cup \{p_i\})^\Gamma = C_1^\Gamma$. Then $\tau \not\models (C_1 \cup \{p_i\})^\Gamma$, since $C_1^\Gamma \subseteq C^\Gamma$ and $\tau \not\models C^\Gamma$ imply that $\tau \not\models C_1^\Gamma$.

      i. If $\tau(\phi_{C_1 \cup \{p_i\}}) = T$, then since we have $\tau \not\models (C_1 \cup \{p_i\})^\Gamma$, by the induction hypothesis we are done.

      ii. Otherwise, $\tau(p_i \wedge \phi_{C_2 \cup \{\bar{p}_i\}}) = T$. Thus $\tau(\phi_{C_2 \cup \{\bar{p}_i\}}) = T$. Also $\tau(p_i) = T$, so $\tau \not\models \{\bar{p}_i\}$. Note $\{\bar{p}_i\} = \{\bar{p}_i\}^\Gamma$, so $\tau \not\models \{\bar{p}_i\}^\Gamma$. Also $C_2^\Gamma \subseteq C^\Gamma$ and $\tau \not\models C^\Gamma$ imply $\tau \not\models C_2^\Gamma$. Thus $\tau \not\models C_2^\Gamma \cup \{\bar{p}_i\}^\Gamma$ and hence as $C_2^\Gamma \cup \{\bar{p}_i\}^\Gamma = (C_2 \cup \{\bar{p}_i\})^\Gamma$, we get $\tau \not\models (C_2 \cup \{\bar{p}_i\})^\Gamma$. Since we have $\tau(\phi_{C_2 \cup \{\bar{p}_i\}}) = T$ and $\tau \not\models (C_2 \cup \{\bar{p}_i\})^\Gamma$, the induction hypothesis applies.

   (b) Suppose $\tau \not\models C^\Delta$ and $\tau(\phi_C) = F$

      i. If $\tau(p_i) = T$, then $\tau \not\models (C_2 \cup \{\bar{p}_i\})^\Delta$. Also $\tau(\phi_{C_2 \cup \{\bar{p}_i\}}) = F$. Then the induction hypothesis applies.

      ii. If $\tau(p_i) = F$, then $\tau \not\models (C_1 \cup \{p_i\})^\Delta$ and $\tau(\phi_{C_1 \cup \{p_i\}}) = F$. Then the induction hypothesis applies.

4. $C = C_1 \cup C_2$ with $\frac{C_1 \cup \{q_i\} \quad C_2 \cup \{\bar{q}_i\}}{C = C_1 \cup C_2}$. Then $\phi_C = \phi_{C_1 \cup \{q_i\}} \wedge \phi_{C_2 \cup \{\bar{q}_i\}}$.

   (a) Suppose $\tau \not\models C^\Gamma$ and $\tau(\phi_C) = T$. $\tau(\phi_C) = T$ implies that $\tau(\phi_{C_1 \cup \{q_i\}}) = \tau(\phi_{C_2 \cup \{\bar{q}_i\}}) = T$. Either $\tau \not\models (C_1 \cup \{q_i\})^\Gamma$ (if $\tau(q_i) = F$) or $\tau \not\models (C_2 \cup \{\bar{q}_i\})^\Gamma$ (if $\tau(q_i) = T$). In either case, the induction hypothesis applies.

(b) Suppose $\tau \not\models C^{\Delta}$ and $\tau(\phi_C) = F$. Note that $(C_1 \cup \{q_i\})^{\Delta} = C_1^{\Delta}$ and $(C_2 \cup \{\bar{q}_i\})^{\Delta} = C_2^{\Delta}$. So $\tau \not\models (C_1 \cup \{q_i\})^{\Delta}$ and $\tau \not\models (C_2 \cup \{\bar{q}_i\})^{\Delta}$. Also, since $\tau(\phi_C) = F$, either $\tau(\phi_{C_1 \cup \{q_i\}}) = F$ or $\tau(\phi_{C_2 \cup \{\bar{q}_i\}}) = F$ so in either case the induction hypothesis applies.

5. $C = C_1 \cup C_2$ with $\frac{C_1 \cup \{r_i\} \quad C_2 \cup \{\bar{r}_i\}}{C = C_1 \cup C_2}$. Then $\phi_C = \phi_{C_1 \cup \{r_i\}} \vee \phi_{C_2 \cup \{\bar{r}_i\}}$.

   (a) Suppose $\tau \not\models C^{\Gamma}$ and $\tau(\phi_C) = T$. Note that $(C_1 \cup \{r_i\})^{\Gamma} = C_1^{\Gamma}$, and $(C_2 \cup \{\bar{r}_i\})^{\Gamma} = C_2^{\Gamma}$. Hence $\tau \not\models (C_1 \cup \{r_i\})^{\Gamma}$ and $\tau \not\models (C_2 \cup \{\bar{r}_i\})^{\Gamma}$. Also, $\tau(\phi_{C_1 \cup \{r_i\}}) = T$ or $\tau(\phi_{C_2 \cup \{\bar{r}_i\}}) = T$. Then the induction hypothesis applies to whichever one equals T.

   (b) Suppose $\tau \not\models C^{\Delta}$ and $\tau(\phi_C) = F$. $\tau(\phi_C) = F$ implies $\tau(\phi_{C_1 \cup \{r_i\}}) = \tau(\phi_{C_2 \cup \{\bar{r}_i\}}) = F$. Either $\tau \not\models (C_1 \cup \{r_i\})^{\Delta}$ (if $\tau(r_i) = F$) or $\tau \not\models (C_2 \cup \{\bar{r}_i\})^{\Delta}$ (if $\tau(r_i) = T$). In either case, the induction hypothesis applies.

Notice that ϕ is monotone in the p_i's.

23.3 Exponential Lower Bounds on Resolution Proofs for Clique and Coloring

Definition A k-coloring of a graph is an assignment of k colors to vertices such that no adjacent vertices have the same color.

Definition A k-clique in a graph is a subset consisting of k nodes such that for any pair of nodes in the subset there is an edge in the graph which joins them.

Theorem 35 *A graph cannot have both a $k+1$ clique and a k-coloring.*

References

[1] S. R. Buss, *Polynomial size proofs of the propositional pigeonhole principle*, Journal of Symbolic Logic, 52 (1987), pp. 916–927.

[2] ——, *Some remarks on lengths of propositional proofs*, Archive for Mathematical Logic, 34 (1995), pp. 377–394.

[3] S. A. Cook, *Feasibly constructive proofs and the propositional calculus*, in Proceedings of the Seventh Annual ACM Symposium on Theory of Computing, 1975, pp. 83–97.

[4] S. A. Cook and R. A. Reckhow, *The relative efficiency of propositional proof systems*, Journal of Symbolic Logic, 44 (1979), pp. 36–50.

[5] J. Krajíček, *Bounded Arithmetic, Propositional Calculus and Complexity Theory*, Cambridge University Press, Heidelberg, 1995.

[6] J. Krajíček and P. Pudlák, *Propositional proof systems, the consistency of first-order theories and the complexity of computations*, Journal of Symbolic Logic, 54 (1989), pp. 1063–1079.

[7] R. J. Parikh, *Some results on the lengths of proofs*, Transactions of the American Mathematical Society, 177 (1973), pp. 29–36.

[8] R. Statman, *Complexity of derivations from quantifier-free Horn formulae, mechanical introduction of explicit definitions, and refinement of completeness theorems*, in Logic Colloquium '76, R. Gandy and M. Hyland, eds., Amsterdam, 1977, North-Holland, pp. 505–517.