

# CS 2429 - Propositional Proof Complexity

## Lecture #11?: 28 November 2002

Lecturer: Toniann Pitassi

Scribe Notes by: Matei David

### 1 Lower bound for bounded-depth Frege proofs of $PHP_n^{n+1}$

In this lecture we will continue the proof of the following theorem.

**Theorem 1** *Any bounded-depth Frege proof of  $PHP_n^{n+1}$  requires exponential size.*

We have seen in the previous lectures the definitions of matching restrictions, matching disjunctions, matching decision trees.

#### 1.1 Overview

We will prove the theorem by contradiction. Assuming there is a short proof  $\mathcal{P}$  of  $PHP_n^{n+1}$  in which all formulas have depth at most  $d$ , we will apply matching restrictions in order to turn the formulas into matching decision trees. The assignment of matching decision trees to formulas is a  $k$ -evaluation. We consider the formulas in  $\mathcal{P}$  in order of increasing depth (recall that depth is defined as the maximum number of alternations of quantifiers).

If  $S = \neg A$  is a formula in our proof, assume we have assigned a matching decision tree  $T(A)$  to formula  $A$ . We assign it a decision tree  $T(S)$  by turning all leaf labels in  $T(A)$  from 0 to 1 and from 1 to 0.

If  $S = A_1 \vee \dots \vee A_k$  is a disjunction in  $\mathcal{P}$ , we construct  $T(S)$  by taking the OR path of 1 leaves in all  $T(A_i)$ , applying a nice restriction to that DNF formula, and building a canonical matching decision tree for that formula. A switching Lemma will guarantee that nice restrictions exist.

The contradiction will come in the following manner.

1. Axioms of the Frege system will be turned into 1-trees (ie, trees which have only leaves labelled by 1).
2. The rules of the Frege system preserve 1-trees.
3. However, any formula in  $PHP_n^{n+1}$  is transformed into a 0-tree.

## 1.2 Analogy

The assignment of trees to formulas creates an analogy with the proof that bounded-depth circuit computing Parity requires super-polynomial size. However, the analogy is broken in the sense that the trees in there compute the exact function, while the matching decision trees in the  $k$ -evaluations used for the proof of Theorem 1 do not, not even on restrictions compatible with that tree. That is, assuming  $\rho = \rho_1 \dots \rho_k$  is a “good” restriction, compare  $f$  only on assignments which extend  $\rho$  to  $T$ .

The tree is equivalent to the formula for only one level. However, when  $S = A_1 \vee \dots \vee A_k$ , rewriting the matching decision trees as matching disjunctions will not preserve the equivalence. Consider  $\sigma$  a partial matching. Even if there exists one path in all trees  $T(A_i)$  consistent with  $\sigma$ , the trees might have nothing in common. Each one is querying only *some* pigeons and we are trying to build something about *all* pigeons. Eg, in a tree which starts by quering  $P_{1,1}$  and  $\sigma$  sends pigeon 2 to hole 5, there might be many paths consistent with  $\sigma$ .

TONI: I didn't quite get the argument above.

## 1.3 $PHP_n^{n+1}$ consists of 0-trees

$PHP$  is the disjunction of the following formulas:

1.  $\neg(\neg P_{i,k} \vee \neg P_{j,k}), \forall i \neq j \leq n+1, \forall k \leq n$
2.  $\neg(P_{i,1} \vee \dots \vee P_{i,n}), \forall i \leq n+1$

Restrictions reduce  $PHP$  to fewer pigeons and holes. After the second block of  $\vee$ , the tree is no longer equivalent to the formula.

Consider formulas of the first type. In order to show that  $T(\neg(\neg P_{i,k} \vee \neg P_{j,k}))$  is a 0-tree, it's enough to show that  $T(\neg P_{i,k} \vee \neg P_{j,k})$  is a 1-tree. By definition

$$T(\neg P_{i,k} \vee \neg P_{j,k}) = T(Disj(T^c(P_{i,k}) \vee Disj(T^c(P_{j,k}))))$$

$T^c(P_{i,k})$  is a tree of size 2 which has 1's for all assignments where  $i$  and  $k$  are mapped to something, and only one 0 corresponding to mapping pigeon  $i$  to hole  $k$  [picture?].

The DNF  $Disj(T^c(P_{i,k}) \vee Disj(T^c(P_{j,k})))$  will contain all terms where  $i, j$  and  $k$  are mapped to something, because, eg, mapping  $k$  to  $i$  is always a leaf labelled with 1 in  $T^c(P_{j,k})$ . [picture?]

For formulas of the second kind, it is enough to show that  $T(P_{i,1} \vee \dots \vee P_{i,n})$  is a 1-tree. By definition,

$$T(P_{i,1} \vee \dots \vee P_{i,n}) = T(\vee_{j=1}^n Disj(T(P_{i,j})))$$

But each  $Disj(T(P_{i,j}))$  contains only one term, namely  $P_{i,j}$ . Then the DNF is  $P_{i,1} \vee \dots \vee P_{i,n}$  and its associated tree starts by querying pigeon  $i$  and will have all leaves labelled with 1 at one level below root, as the formula is true no matter where this pigeon is mapped.

### 1.4 All formulas in a bounded-depth Frege proofs get assigned 1-trees

This is Lemma 5.1 in the paper. The Frege system we are considering has axiom  $A \vee \neg A$ , and rules

$$\frac{A}{A \vee B}, \frac{A \vee A}{A}, \frac{A \vee (B \vee C)}{(A \vee B) \vee C}, \frac{A \vee B, \neg A \vee C}{B \vee C}$$

The proof is in the paper, using as parameter the maximum number of subformulas in each rule.

**Theorem 2 (Lemma 5.1)** *Let  $f$  be the maximum number of subformulas appearing in a rule (this is a constant, 7?). Let  $\mathcal{P}$  be a proof of  $PHP_n^{n+1}$ ,  $T$  a  $k$ -evaluation for all subformulas in  $\mathcal{P}$  and  $k < n/f$ , then any formula occurring as a line in  $\mathcal{P}$  gets converted to a 1-tree.*

The proof is by induction on the number of lines, if we start with axioms and keep applying sound rules (as the ones above), all formulas convert to 1-trees.

After applying a restriction the number of variables we are left is  $n' = n^\epsilon$ . Since we might be applying  $d$  restrictions (the bound on the depth of formulas), we want  $k \ll n^{\epsilon^d}$ .

TONI: Here you argued that the proof works for two of the rules, the axiom and  $\frac{A}{A \vee B}$  but I didn't understand the argument for either.

TONI: Next you quickly considered how the parameters look like. What I have is very vague.

The entire argument also works for onto- $PHP$  or func- $PHP$  because they also convert to 0-trees.