CS 2429 - Propositional Proof Complexity

Lecture #4: October 3, 2002

Lecturer: Toniann Pitassi

Scribe Notes by: Mehrdad Sabetzadeh

1 Resolution Lower Bound for the Pigeon Hole Principle

In the previous lecture, we introduced the Pigeon Hole Principle problem, or *PHP* for short, and drew an outline of how we want to prove a lower bound on the length of Resolution proofs for *PHP*. We now carry out the proof in detail.

Roughly speaking, the proof of resolution lower bound for *PHP* proceeds as follows: first, we show that every proof of PHP^n_{n-1} contains a *medium complexity clause* and further that every medium complexity clause is *large*. Second, we show that a partial assignment to the variables, called a *restriction*, can be applied to every small proof so that

(a) every large clause disappears.

(b) the result is still a $PHP^{n'}_{n'-1}$ proof for some good size n'.

A proof by contradiction will then get us a lower bound on the proof size of PHP^n_{n-1}.

Proposition 1 (Number of CNF Clauses in PHP^n_{n-1}) *The number of clauses in PHP^n_{n-1} is $n + \binom{n}{2}(n-1) = \Theta(n^3)$.*

Notation: Any CNF clause in PHP^n_{n-1} problem can be visualized by an $n \times (n-1)$ table in which every cell c_{ij} in the table represents P_{ij}. The table that corresponds to a CNF clause has a "+" ("-") in cell c_{ij} iff P_{ij} ($\neg P_{ij}$) appears in the CNF clause. The table is interpreted as the *disjunction* of the literals. Figure 1 illustrates the tables for two PHP^4_3 clauses.

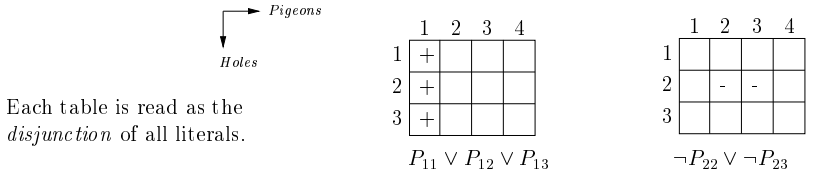

Figure 1: Tabular presentation of *PHP* clauses

Definition We say a truth assignment is i-critical if it matches all $n-1$ holes to all pigeons but pigeon i (see Figure 2.)

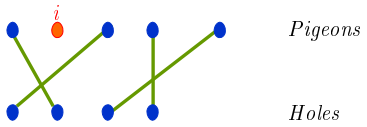

Figure 2: An i-critical truth assignment

Such an assignment is barely unsatisfying: it always satisfies all one-to-one, onto and function clauses and all *but* one of the clauses saying that f is total. The only clause it falsifies is $C_i = (P_{i1} \vee P_{i2} \vee \ldots \vee P_{i(n-1)})$ which says that pigeon i is mapped somewhere. We will only care about the properties of the clauses in the proof when evaluated on critical truth assignments. The properties of critical truth assignments make it convenient to convert each such clause C to a totally monotone clause $M(C)$ by replacing each occurrence of $\neg P_{ij}$ in C with $(P_{1j} \vee \ldots \vee P_{(i-1)j} \vee P_{(i+1)j} \vee \ldots \vee P_{nj})$. An example has been depicted in Figure 3.

Figure 3: Making clauses monotone

Taking the one-to-one and onto clauses of *PHP* into account, it is easy to check that the set of monotone clauses is equivalent to the original clauses with respect to critical truth assignments and therefore, inferences using these clauses are still *sound* with respect to critical truth assignments.

Tabular Resolution: Based on the definition of the pigeon hole principle and the argument given above, tabular resolution for monotone clauses can be defined as in Figure 4.

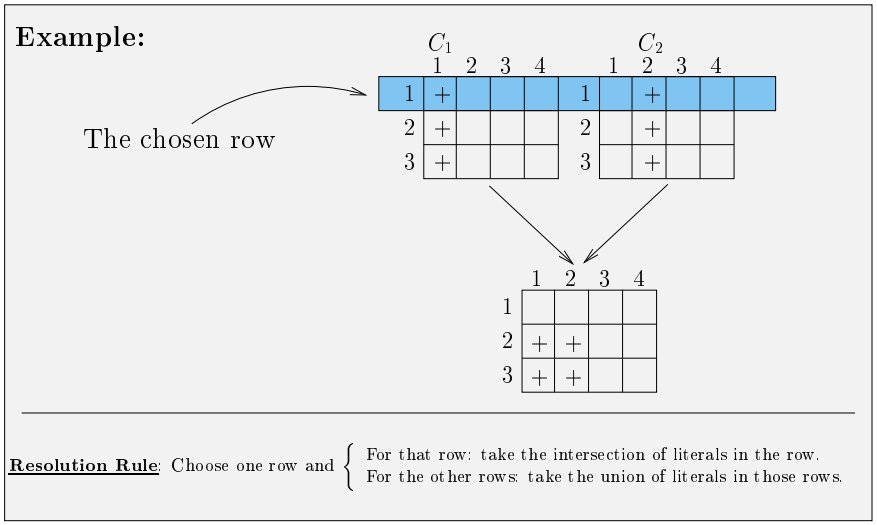

Figure 4: Tabular Resolution

Definition Given a clause C, let $badpigions(C) = \{i|$ there is some i-critical assignment α falsifying $C\}$. We define the complexity of C, $comp(C) = |badpigions(C)|$. Figure 5 illustrates three examples: the first two tables represent two initial clauses and the third one represents Λ, the final clause.

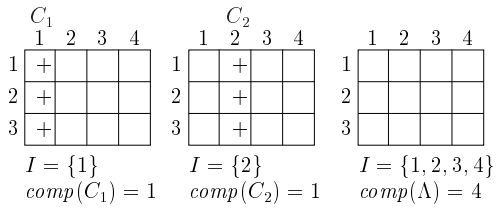

Figure 5: Clause complexity examples

The complexity of initial pigeon clauses is at most one and the complexity of the final clause, Λ, is n. Moreover, by soundness, we know that the complexity of a resolvent is at most the sum of the complexities of the two clauses from which it was derived, so if clause A and B imply a clause C, then $comp(C) \leq comp(A) + comp(B)$. Therefore, if C is the first clause in the proof with $comp(C) > n/3$, we must have $n/3 < comp(C) \leq 2n/3$ because we cannot do more than double the complexity of the resolvent in each step. Therefore, we couldn't have jumped over $(n/3, 2n/3]$ region. Now, it remains to show that $M(C)$ contains a large number of variables.

For $comp(C) = t$, we claim that $M(C)$ has at least $(n-t)t \geq 2n^2/9$ distinct literals mentioned. Fix some $i \in badpigeons(C)$, and let α_i be an i-critical truth assignment with $C(\alpha_i) = \text{FALSE}$. For each $j \notin badpigeons(C)$, define the j-critical assignment α_{ij}, obtained from α_i by toggling i and j, that is if α_{ij} maps i to hole k, then j was mapped to k in α_i (see Figure 6.)

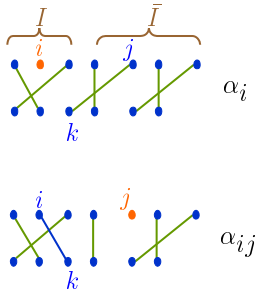

Figure 6: Toggle operation

Now $C(\alpha_{ij}) = \text{TRUE}$ since $j \notin badpigeons(C)$ and further, α_i and α_{ij} differ only in that α_{ij} maps i to k rather than j to k. Since C and $M(C)$ agree on all critical assignments and $M(C)$ is positive, it must contain the variable P_{ik}. This argument may be applied for every $i \in badpigeons(C)$ and $j \notin badpigeons(C)$, yielding the size bound on $M(C)$.

Finally, we describe the restriction argument that gets us the desired result. Restrictions in this case are partial assignments that map certain pigeons to certain holes. To map a pigeon i to hole j, we set P_{ij} to TRUE and set all other P_{ik} or P_{kj} to FALSE (see Figure 7.) Note that we must be very careful not to falsify any clause when applying restriction.

This reduces PHP^n_{n-1} to $PHP^{n'}_{n'-1}$, where $n' = n - 1$. To complete the proof, let us call a positive clause *large* iff it has at least $n^2/10$ literals. Assume, for a proof by contradiction, that

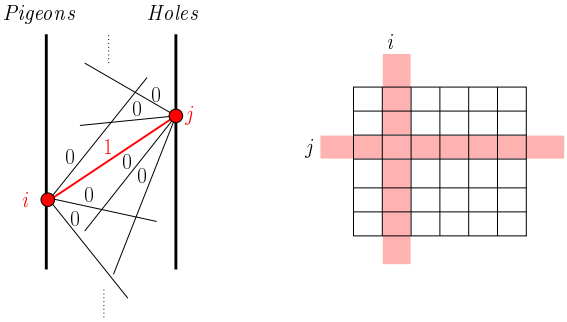

Figure 7: Restriction

there is some resolution proof of PHP^n_{n-1} with at most $S < 2^{n/20}$ clauses C such that $M(C)$ is large.

On average, restricting a P_{ij} to TRUE will satisfy $S/10$ of all large clauses because large clauses each have $1/10$ of all variables. Choose a P_{ij} that satisfies the most large clauses. This restriction decreases the number of large clauses by a factor of $9/10$. Now repeat such a restriction $\log_{9/10} S < 0.329n$ times. The remaining proof proves $PHP^{n'}_{n'-1}$ for some n' such that $2(n')^2/9 > n^2/10$ and does not have any large clauses. This is a contradiction because such a refutation, from what we saw earlier, must have a clause of size at least $2(n')^2/9$ which qualifies as a large clause even for PHP^n_{n-1}.

Theorem 2 (Resolution size lower bound) *For sufficiently large n, any proof of PHP^n_{n-1} requires size at least $2^{n/20}$.*

2 Width versus Size of Resolution Proofs

Let F be a set of clauses over variables $\{x_1, \ldots, x_n\}$ and $width(F)$ be the number of literals in the largest clause in F. If P is a resolution proof of F, $width(P)$ is the number of literals in the largest clause in P. Let $proofwidth(F)$ denote the minimum of all proofs P of F of $width(P)$.

Theorem 3 *Every Davis-Putnam (DLL)/tree-like resolution proof of F of size S can be converted to one of width $\lceil \log_2 S \rceil + width(F)$.*

Proof We show this by induction on the size of the resolution proof. Clearly, the claim holds for $S = 1$. Assume that for all sets F' of clauses with a tree-like resolution refutation of size $S' < S$, there is a tree-like resolution proof P' of F' with $width(P') \leq \lceil \log_2 S' \rceil + width(F')$.

Now consider a tree-like resolution refutation of size S of a set F of clauses and let x be the last variable resolved to derive the empty clause Λ. Clearly, one of the two subtrees at the top has size at most $S/2$ and the other has size strictly smaller than S. Without loss of generality, let these be the left and the right subtree, respectively. Also, assume $\bar{x}$ comes from the left subtree and x from the right as in Figure 8 (left).

Since we can prove $\bar{x}$ from F in size at most $S/2$, we can also prove Λ from $F|_{x \leftarrow 1}$ in size at most $S/2$. The induction hypotheses now implies that we can also derive from $F|_{x \leftarrow 1}$ in width at most $w - 1 = \lceil \log_2(S/2) \rceil + width(F) = \lceil \log_2(S) \rceil + width(F) - 1$. Adding $\bar{x}$ to each of the clauses

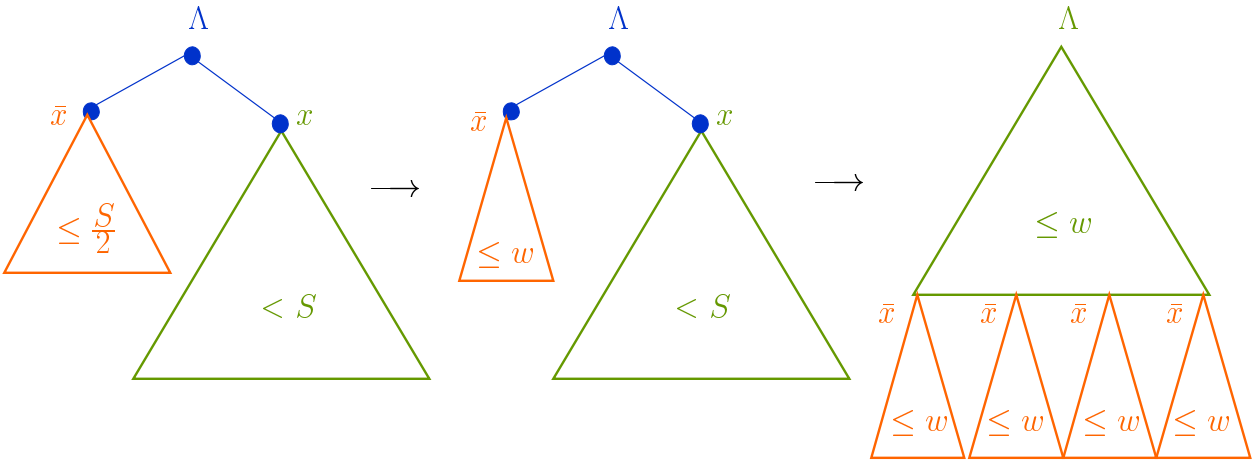

Figure 8: Converting a small size proof to one with small width

in this proof lets us derive $\bar{x}$ from F in width $w = \lceil \log_2(S) \rceil + width(F)$. In a similar way, starting with the right subtree, which is of size strictly smaller than S, we can derive Λ from $F|_{x \leftarrow 0}$ in width at most $w = \lceil \log_2(S) \rceil + width(F)$.

Now use a copy of the left sub-tree to resolve with each leaf clause of the right subtree that contains an x as in Figure 8 (right). This allows us to eliminate x at the very bottom of the right subtree, and we are effectively left with $F|_{x \leftarrow 0}$. From what we said before, we can now derive Λ from this in width $\lceil \log_2(S) \rceil + width(F)$. This completes the proof.

Corollary 4 *Any Davis-Putnam (DLL)/tree-like resolution proof of F requires size at least $2^{(proofwidth(F) - width(F))}$.*

In an almost similar way, we can prove the following theorem and its corollary:

Theorem 5 *Every resolution proof of F of size S can be converted to one of width $\sqrt{2n \ln S} + width(F)$.*

Corollary 6 *Any resolution proof of F requires size at least $\mathbf{e}^{(proofwidth(F) - width(F))^2/(2n)}$.*

3 Resolution Proofs Based on the Width-Size Relationship

Given F, a set of unsatisfiable clauses, let $s(F)$ be the size of the minimum subset of F that is unsatisfiable. Define the boundary δF of F to be the set of variables appearing in exactly one clause of F. Let the *sub-critical* expansion of F be

$$e(F) = \max_{\frac{s(F)}{3} \leq s \leq \frac{2s(F)}{3}} \min \{|\delta G| : G \subseteq F, |G| = s\}$$

The following lemma (depicted in Figure 9) relates the width of a proof F to its sub-critical expansion:

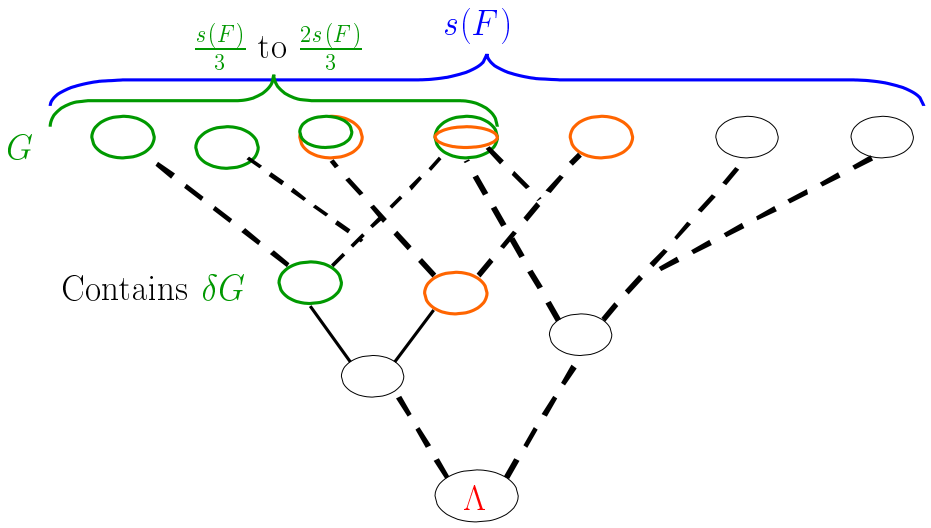

Figure 9: Relating proof width to sub-critical expansion

Lemma 7 *If P is a resolution proof of F, then $width(P) \geq e(F)$.*

Proof Given a clause C in P and a collection of clauses $G \subseteq F$, write $G \Rightarrow_P C$ if all the clauses in G are used in P to derive C. In a resolution proof, if a literal appears in a clause C then the only way it can be removed from clauses derived using C is if the literal is resolved with its negation. Therefore, if $G \Rightarrow_P C$ then every variable in δG appears in C and so $width(C) \geq |\delta G|$.

Define the complexity, $comp_P(C)$, of a clause C in P to be the size of the set $G \subseteq F$ such that $G \Rightarrow_P C$. By definition $comp_P(\Lambda) \geq s(F)$ and $comp_P(C) = 1$ for any clause in F. Furthermore, $comp_P$ is sub-additive, $comp_P(C) \leq comp_P(A) + comp_P(B)$ if C is a resolvent of A and B. Any clause C in P such that $s(F)/3 \leq comp_P(C) \leq 2s(F)/3$ satisfies $width(C) \geq |\delta G|$ for some set $G \subseteq F$ with $s(F)/3 \leq |G| \leq 2s(F)/3$. Maximizing over all choices of s, $s(F)/3 \leq s \leq 2s(F)/3$ where $|G| = s$, we get $width(P) \geq e(F)$.

Corollary 8 *Any Davis-Putnam/DLL proof of F requires size at least $2^{e(F)}$ and any resolution proof requires size at least $2^{\Omega(e^2(F)/n)}$.*