Propositional proof complexity I.

Jan Krajíček
Mathematical Institute
Academy of Sciences of the Czech Republic
Praha

This is a draft of a first part of lecture notes on Propositional proof complexity. It is roughly the content of my course at the Charles University in Spring'03. This first part is almost complete and there will be only a few additions. Most notably: infinitary criteria for lower bounds in tree-like and general resolution, on Ramsey theorem in resolution, the link between modular counting principles and algebraic proof systems, the separation between depth d and $d+1$ Frege systems, and the definition and few facts about the constant depth Frege systems with modular counting gates. There may be also missing references.

Next part will roughly correspond to a course I plan for Fall'04; it should include: links with bounded arithmetic, finitistic consistency statements and p-simulations, a construction of hard tautologies, NP-pairs and links to cryptography, automatizability of proof systems, some upper bounds (that can be proved via bounded arithmetic much more easily than directly), and a part on the current project of τ-formulas based on pseudo-random generators.

The eventual lecture notes will include also some topics not covered in either of the two courses. In particular, this should include auxiliary proof systems like algebraic proof systems (Nullstellensatz, polynomial calculus, a proof system based on a finitely presented group, etc.) or geometric proofs systems (cutting planes and their extensions to Lovasz-Schrijver system and to the 1st order theory of discretely ordered rings) or links with model theory (e.g. the notion of covering classes and Euler structures), and perhaps some other less familiar topics.

Some reference in the current text are just **??**; they refer to future parts.

21. 5. 2003

Contents

Chapter 1

Basic concepts and motivations

Propositional proof complexity studies the complexity of proving that a propositional formula is a tautology. For a definiteness we fix set $TAUT$ of tautologies in the DeMorgan language with constants 0, 1 (the truth values FALSE and TRUE) and propositional connectives: unary $\neg$ (the negation), and binary $\wedge$ and $\vee$ (the conjunction and the disjunction). (The language also contains various auxiliary symbols like brackets or commas.) The formulas are built, using the connectives, from the constants and from atoms $p_0, p_1, \ldots, p_n, \ldots$.

We consider all finite objects encoded in a finite alphabet and , in fact, in the binary alphabet $\{0,1\}$. In particular, we consider $TAUT$ as a subset of $\{0,1\}^*$ and so the length of a formula φ is denoted $|\varphi|$. A minor point to note (and then ignore) is that the length of an atom p_n is not 1 but $|p_n| \sim \log n$, as the index n has to be encoded in binary. But we shall ignore this as the logarithmic factor is irrelevant in our computations.

Consider any one of the usual text-book examples of propositional calculi working with DeMorgan formulas that is based on a finite number of axiom schemes (like $A \vee \neg A$, or similar) and a finite number of inference rules (like the modus ponens $A, \neg A \vee B / B$, or similar)[1]. Any such system is called a *Frege system* and denoted F. Two properties the system has are:

1. A formula τ has a proof in F iff $\tau \in TAUT$ (the if-direction is the completeness and the only-if-direction is the soundness of F).

[1]What the qualification *similar* means will be explained in Section 3.1.

2. The relation w *is an* F*-proof of* τ is a p-time decidable relation of w and τ.

These two properties lead to the following abstract definition of a proof system.

Definition 1.0.1 (Cook-Reckhow[14]) *A propositional proof system (a* pps, *shortly) is any p-time computable function* $P : \{0,1\}^* \to \{0,1\}^*$ *such that* $Rng(P) = TAUT$.

Any $w \in \{0,1\}^*$ *such that* $P(w) = \tau$ *is called a* P*-proof of* τ.

A pps P *is polynomially bounded if there exists a polynomial* $p(x) \in \mathbf{N}[x]$ *such that any* $\tau \in TAUT$ *has a* P*-proof* w *of size* $|w| \leq p(|\tau|)$.

It is easy to see that F can be seen as a pps in this abstract setting too. Just define a function P_F by:

$$P_F(w) = \begin{cases} \tau & \text{if } w \text{ is an } F\text{-proof of } \tau \\ 1 & \text{otherwise} \end{cases}$$

Any of the usual logic systems for propositional logic can be similarly represented, be it the sequent calculus, the natural deduction system, the first-order predicate logic or even first-order theories. For example, a less usual pps is:

$$P_{ZFC}(w) = \begin{cases} \tau & \text{if } w \text{ is a proof in set theory ZFC of the formalization} \\ & \text{of the statement } \tau \in TAUT \\ 1 & \text{otherwise} \end{cases}$$

based on set theory.

The following is the main theorem showing that proof complexity relates to computational complexity.

Theorem 1.0.2 (Cook-Reckhow[14]) *There exists a polynomially bounded pps iff* $\mathcal{NP} = co\mathcal{NP}$.

Proof :

If P is a p-bounded pps with the polynomial bound $p(x)$ then

$$\exists w(|w| \leq p(|x|)); P(w) = x$$

is an $\mathcal{NP}$-definition of $TAUT$, a $co\mathcal{NP}$-complete set.

On the other hand, if $\exists u(|u| \leq q(|x|)); A(u,x)$ is such a definition (with A a p-time relation) then the function

$$P(w) = \begin{cases} \tau & \text{if } w = (u,x) \text{ and } |u| \leq q(|\tau|) \wedge A(u,\tau) \text{ holds} \\ 1 & \text{otherwise} \end{cases}$$

is a polynomially bounded pps.

q.e.d.

Hence, if we believe that $\mathcal{NP} \neq co\mathcal{NP}$, no pps is p-bounded. A large part of proof complexity activity is centered around proving that particular pps' are not p-bounded (or even subexponentially bounded). The conjecture $\mathcal{NP} \neq co\mathcal{NP}$ itself would be unlikely proved in this incremental manner as a way to prove a universal statement is rarely proving all its instances. But we may hope to uncover hidden "computional hardness assumptions" in these lower bounds and thus to reduce the conjecture to some intuitively more rudimentary one. (More on this in the introductions to [29, 40] or in [30].)

However, there is another less illusory motivation for proving lower bounds for concrete pps' that I shall explain now.

Consider a first-order sentence in, say, the language of directed graphs: $=$, a binary relation $R(x,y)$ and a constant which we shall denote 0. As an example[2] I take the pigeonhole principle PHP:

$$\exists x \forall y, \neg R(x,y) \ \vee \ [\exists x_1, x_2, y; x_1 \neq x_2 \wedge R(x_1,y) \wedge R(x_2,y)] \ \vee$$

$$[\exists x, y_1, y_2; y_1 \neq y_2 \wedge R(x,y_1) \wedge R(x,y_2)] \ \vee \ \exists x; R(x,0) \ .$$

Assume that $R(x,y)$, a relation on some universe M, does not satisfy any of the first three disjuncts. Then it is a graph of an injective function $f : M \to M$. The last disjunct must then be true, i.e. 0 must be a value. In other words, PHP says that an injective function is surjective. The principle is valid for all finite M. For any $n \geq 1$ we can translate PHP into a propositional formula $\langle PHP \rangle_n$ as follows: Replace $\exists$ and $\forall$ by the disjunction and the conjunction respectively over all elements of $[n]$, leave the propositional connectives in place, replace true resp. false atomic sentences $i \neq j$ by 1 resp. by 0, and translate atomic sentences $R(i,j)$ by new atoms r_{ij}, one for

[2]This is not a random choice. We shall see that the PHP - in various forms - is the most important principle studied in proof complexity.

every pair $i, j \in [n]$. The formula $\langle PHP \rangle_n$, often denoted just PHP_n, is then:

$$\bigvee_i \bigwedge_j, \neg r_{ij} \ \vee \ [\bigvee_{i_1 < i_2, j} r_{i_1 j} \wedge r_{i_2 j}] \vee$$

$$[\bigvee_{i, j_1 < j_2} r_{i j_1} \wedge r_{i j_2}] \ \vee \ \bigvee_i r_{i0} \ .$$

Here it is already simplified a bit, deleting disjuncts which are 0 (like $(0 \wedge r_{ij} \wedge r_{ij}))$, and deleting also multiple occurences of some disjuncts (like $(r_{i_1 j} \wedge r_{i_2 j})$ and $(r_{i_2 j} \wedge r_{i_1 j}))$. In fact, PHP_n is usually simplified yet more. By allowing j to range only over $[n] \setminus \{0\}$ we get rid of the last disjunct in the formula:

$$\bigvee_i \bigwedge_j, \neg r_{ij} \ \vee \ [\bigvee_{i_1 < i_2} \bigvee_j r_{i_1 j} \wedge r_{i_2 j}] \ \vee \ [\bigvee_i \bigvee_{j_1 < j_2} r_{i j_1} \wedge r_{i j_2}] \ .$$

The truth assignments to r_{ij}' correspond to relations on $[n]$. As PHP is valid in all structures of size n, PHP_n is satisfied by all truth assignments, i.e. it is a tautology.

In general this translation can be defined for any Π^1_1 first order sentence Φ. If Φ is valid in all finite structures then the resulting sequence of formulas $\langle \Phi \rangle_n$, $n < \omega$, is a sequence of tautologies.

The second important motivation for studying lengths of proofs in particular pps' is the following fact: To any "usual"[3] first-order theory T it is possible to attach a pps P_T such that $\langle \Phi \rangle_n$, $n < \omega$, have short (usually polynomial or quasipolynomial size) P_T-proofs if T proves Φ. Hence a sufficiently strong lower bound to the length of such proofs implies the unprovability of Φ in T. A particular formula Φ to which the construction applies can be, for example, a consistency statement. Consistency statements are the most important formulas used in a calibration of the strength of theories (for very good proof-theoretic reasons).

Although we can describe P_T for T being Peano Arithmetic PA or set theory ZFC, nobody has a clue how to prove any lower bound for such P_T. However, for some theories of interest in logic (in particular, for the so called Bounded Arithmetic theories) the situation is much better and we have even exponential lower bounds for some of the P_T' arising in these cases.

We conclude the chapter by a natural notion of quasi-ordering of pps' by their strength.

[3]This topic will be studied in Chapter **??** where we define the qualification *usual*.

Definition 1.0.3 (Cook-Reckhow[14]) *Let P, Q be two pps'. Pps P p-simulates Q, $P \geq_p Q$ in symbols, iff there is a p-time computable function $g : \{0,1\}^* \to \{0,1\}^*$ such that for all $w \in \{0,1\}^*$:*

$$P(g(w)) = Q(w) \ .$$

In other words, g translates Q-proofs into P-proofs of the same formula. As g is p-time, the length of the P-proofs is at most polynomially longer than the length of the original Q-proofs.

Chapter 2

Resolution

We start our investigation of particular pps' with the resolution proof system R. It is the simplest pps for which it is not easy to prove a lower bound. The proof system has been introduced by Blake [9] and made prominent some thirty years later in its use in automated theorem proving, cf. Davis-Putnam [18] and Robinson [44].

2.1 Definition, soundness and completeness

Resolution is a proof system, denoted simply R, for proving formulas in a DNF form. In general, transforming a formula into an equivalent one in the DNF form may increase its size exponentially. However, we don't really need an equivalent formula, we only need that the original formula is a tautology iff the constructed DNF formula is too. This can be done by a simple trick, the so called *limited extension*, that is described in Exercise 2.9.1. A *literal* is an atom or its negation. A *clause* is a disjunction of literals $\ell_1 \vee \ldots \vee \ell_k$, possibly empty. As there are no other connectives or formulas in resolution, the clause is written simply as a set $\{\ell_1, \ldots, \ell_k\}$. The only inference rule in R is the *resolution rule*:

$$\frac{C \cup \{p_i\} \qquad D \cup \{\neg p_i\}}{C \cup D}$$

The atom p_i is called the resolved atom. There are no restriction on occurrences of p_i or $\neg p_i$ in C and D, but it is easy to see that we can assume w.l.o.g. that neither p_i nor $\neg p_i$ occur in $C \cup D$.

An assignment satisfies a clause if it makes true at least one literal in the clause. In particular, the empty clause cannot be satisfied. The resolution rule is sound: If both clauses in the hypothesis of an inference are satisfied by an assignment then the assignment satisfies the conclusion too.

Let A be a formula in a DNF form $\bigvee_{i \in I} B_i$, with $B_i = \bigwedge_{j \in J_i} \ell^i_j$ and ℓ^i_j literals. Define clauses $C_i := \{\neg \ell^i_j \mid j \in J_i\}$, for $i \in I$. A *resolution proof* of A is a sequence $D_1, \dots, D_t$ of clauses such that:

1. Each D_u is either one of *initial clauses* C_i, $i \in I$, or it is derived using the resolution rule from D_{v_1} and D_{v_2}, some $v_1, v_2 < u$.

2. The *end-sequent* D_t is the empty clause $\emptyset$.

The proof of A is also often called the *refutation* of $C_1, \dots, C_k$ as its existence certifies that C_i' are not simultaneously satisfiable.

Theorem 2.1.1 *A DNF formula is provable in R iff it is a tautology.*

Proof :

Let A be a DNF formula and let C_i's be the clauses obtained as above. Any truth assignment satisfying all C_i' would have to satisfy, by the soundness of the resolution rule, all clauses in any resolution refutation of $C_1, \dots, C_k$. In particular, also the end clause - the empty clause - would have to be satisfied. But that is impossible as there is nothing to satisfy in $\emptyset$. This proves the only-if part of the theorem.

For the opposite direction assume that $\mathcal{C} = \{C_1, \dots, C_k\}$ is unsatisfiable. Let $p_1, \dots, p_n, \neg p_1, \dots, \neg p_n$ be the literals appearing in $\mathcal{C}$. We shall prove by induction on n that for any such $\mathcal{C}$ there is a resolution refutation of $\mathcal{C}$.

If $n = 1$ there is nothing to prove: $\mathcal{C}$ must contain clauses $\{p_1\}$ and $\{\neg p_1\}$ and their resolvent is the empty clause. Assume $n > 1$, and partition $\mathcal{C}$ into four disjoint sets: $\mathcal{C}_{00} \cup \mathcal{C}_{01} \cup \mathcal{C}_{10} \cup \mathcal{C}_{11}$, of those clauses which contain no p_n and no $\neg p_n$, no p_n but do contain $\neg p_n$, do contain p_n but not $\neg p_n$ and contain both $p_n, \neg p_n$ respectively.

Now form new set of clauses $\mathcal{C}'$ by

1. Delete all clauses from $\mathcal{C}_{11}$.

2. Replace $\mathcal{C}_{01} \cup \mathcal{C}_{10}$ by the set of all clauses that are obtained by the resolution rule applied to all pairs of clauses $C_1 \cup \{\neg p_n\}$ from $\mathcal{C}_{01}$ and to $C_2 \cup \{p_n\}$ from $\mathcal{C}_{10}$.

Note that the new clauses introduced in the 2nd step do not contain either p_n or $\neg p_n$. More importantly, the new set of clauses $\mathcal{C}'$ is also unsatisfiable. This is because any assignment $\alpha' : \{p_1, \dots, p_{n-1}\} \to \{0,1\}$ satisfies either all clauses C_1 such that $C_1 \cup \{\neg p_n\} \in \mathcal{C}_{01}$, or all clauses C_2 such that $C_2 \cup \{p_n\} \in \mathcal{C}_{01}$ (otherwise we could find $C_1 \cup C_2 \in \mathcal{C}'$ not satisfied by α'). Hence α' can be extended, by giving a suitable value to p_n, to a truth assignment α satisfying $\mathcal{C}$, which is a contradiction.

q.e.d.

Obviously, the proof constructed in the completeness part of the argument can be sometimes exponentially long (see Exercise 2.9.2). However, this does not mean that there cannot be some other, much shorter, R-proofs. The first superpolynomial (and, in fact, exponential) lower bound for R-proofs has been proved only in 1985 by Haken [20]. We shall give, in the coming sections, several exponential lower bounds for R.

2.2 Tree-like resolution

An R-proof $\pi = (D_1, \dots, D_t)$ is *tree-like* iff each D_i is used at most once as a hypothesis of an inference in the proof. If one draws the *proof-graph* of π, a directed graph with nodes being the clauses and the edges going from the conclusion of an inference to the two hypothesis, then the condition tree-like precisely says that the graph is a tree (a *proof-tree*).

The proof system allowing exactly tree-like R-proofs is called *tree-like resolution* and denoted R^*. In this section we give an exponential lower bound on the size of R^*-proofs of PHP_n.

With an unsatisfiable set of clauses $\mathcal{C} = \{C_1, \dots, C_k\}$ we may associate the following *search problem*: Given a truth assignment α to the atoms of $\mathcal{C}$ find $C_i \in \mathcal{C}$ false under α. This search problem can be solved by a *branching program*, a simple concept from Boolean complexity.

A branching program is a directed acyclic graph with one in-degree 0 node (the source), and with all other nodes of out-degree either 2 (the inner nodes) or 0 (the leaves). The inner nodes are labelled by atoms and the two edges leaving a node are labelled by $0, 1$ respectively. The leaves are labelled by elements of a some set X. Any evaluation α of atoms determines a path through the branching program: The path starts at the source and in every node labelled by p_i uses the edge labelled 1 iff $\alpha(p_i) = 1$. In this way a branching program computes a function $f(p_1, \dots, p_n) : \{0,1\}^n \to X$

assigning to $\alpha \in \{0,1\}^n$ the label of the leaf on the path determined by α. *The size* of a branching program is the number of nodes.

An important special case of branching programs are *decision trees*, branching programs that are trees with the edges directed from the root towards the leaves. We speak about the *the height* of a decision tree, meaning the maximum length of a path through it.

Now back to our search problem of finding unsatisfied clauses from $\mathcal{C}$. Assume that we have an R^*-refutation π of $\mathcal{C}$. We shall use π as a decision tree for solving the search problem as follows. The underlying tree of the decision tree is the proof-tree of π. The source is the end-clause. A node corresponding to a clause D derived in π by resolving atom p_i is labeled by p_i. The edge from the node towards the node corresponding to a hypothesis of the inference is labelled by 1 (resp. by 0) iff the hypothesis contains $\neg p_i$ (resp. it contains p_i). The leaves of the tree correspond to initial clauses in π and they are labelled by the initial clauses themselves.

Lemma 2.2.1 *Assume π is an R^*-refutation of $\mathcal{C}$. Then the decision tree defined from π as above solves the search problem: Given a truth assignment α find an unsatisfied clauses in $\mathcal{C}$.*

In particular, the height of the decision tree is the same as the height of the proof tree of π.

Proof :

It is enough to observe that the clauses corresponding to the nodes on the path determined by an α are all falsified by α.

q.e.d.

Now we can prove our first, quite modest, lower bound.

Theorem 2.2.2 *Every R^*-proof of PHP_n must have the height at least $n-1$.*

Proof :

By Lemma 2.2.1 it suffices to show that any decision tree solving the search problem attached to PHP_n must have the height at least $n-1$.

The search problem can be interpreted as follows: Given a truth assignment α, which we may identify with a relation $\subseteq [n] \times ([n] \setminus \{0\})$, find one of:

1. A pigeon $i \in [n]$ that is mapped (by the function whose graph α is supposed to be) nowhere, i.e. i such that $\forall j \in [n] \setminus \{0\}; \neg\alpha(i, j)$.

2. Pigeons $i_1 < i_2$ and a hole j such that both i_1 and i_2 are mapped into j: $\alpha(i_1, j) \wedge \alpha(i_2, j)$.

3. A pigeon i and two holes $j_1 < j_2$ such that i is mapped to both the holes: $\alpha(i, j_1) \wedge \alpha(i, j_2)$.

Let $g :\subseteq [n] \to [n] \setminus \{0\}$ be a partial 1-to-1 map. The map g determines a partial truth assignment α_g by:

1. $\alpha_g(r_{ij}) = 1$ iff $g(i)$ is defined and equal to j.

2. $\alpha_g(r_{ij}) = 0$ iff $g(i)$ is defined but different from j, or for some $k \neq i$, $g(k) = j$.

3. $\alpha(r_{ij})$ is undefined in all other cases.

A partial truth assignment *forces* a clause true iff it assigns 1 to a literal in the clause, and it forces a clause false iff it assigns 0 to all literals in the clause. In particular, a partial assignment cannot force a clause false without giving a value to all literals occurring in it. The following is straightforward.

Claim: *Let $g :\subseteq [n] \to [n] \setminus \{0\}$ be a partial 1-to-1 map of cardinality $< n - 1$. Then the partial truth assignment α_g cannot force false any clause of $\neg PHP_n$, i.e. any initial clauses in an R^*-proof of PHP_n.*

Assume that we have a decision tree of the height h solving the search problem. We walk through the tree creating at step ℓ a partial 1-to-1 map $g_\ell : [n] \to [n] \setminus \{0\}$ such that $|g_\ell| \leq \ell$, and such that α_{g_ℓ} gives values to all atoms at the nodes of the path up to the ℓth step, and the values are consistent with the path.

At the beginning put $g_0 := \emptyset$. Assume we have g_ℓ and the atom at the node we need to decide in the $(\ell+1)$st step is r_{ij}. If $j \neq 0$ and $g_\ell \cup \{(i, j)\}$ is a partial 1-to-1 map, define $g_{\ell+1} := g_\ell \cup \{(i, j)\}$. Otherwise put $g_{\ell+1} := g_\ell$. It is easy to verify that the maps g_ℓ' have the required properties.

By the claim, the last map must have the size at least $n - 1$. That is, the path has to continue for at least $n - 1$ steps, i.e. the height of the tree is at least $n - 1$.

q.e.d.

A binary tree of height n may have, if it is very unbalanced, the size just $2n+1$ and that gives a very poor lower bound (even the number of clauses in PHP_n is bigger: $O(n^3)$). Hence we need to modify the argument a bit in order to get a lower bound for the size of R^*-proofs of PHP_n. In fact, we will estimate from below the number of clauses in any such proof (that number is obviously a lower bound to the size).

First we prove a simple lemma about binary trees. We shall think of binary trees as ordered upwards from the root (the minimal element) up towards the leafs. Let us denote the ordering by a generic symbol $\geq$. For a binary tree T and a node a in T, denote by T^a the subtree of T consisting of nodes b such that $b \geq a$. By T_a denote the tree $(T \setminus T^a) \cup \{a\}$, i.e. it consists of nodes b such that $b \not\geq a$. By $|T|$ denote the size of a tree T.

Lemma 2.2.3 (Spira [45]) *There is a node $a \in T$ such that:*

$$(1/3)|T| \;\leq\; |T_a|, |T^a| \;\leq (2/3)|T| \;.$$

Proof :

Walk a path through T, starting at the root and always walking to the bigger subtree (if the two subtrees have the same size, choose arbitrarily one). The size s of a current subtree can decrease in one step only to $s' \geq \frac{s-1}{2}$.

Continue in this fashion until we reach the first node a such that the subtree T^a has the size $\leq (2/3)|T|$. The key observation is that then also $(1/3)|T| \leq |T^a|$. This is because the immediately previous subtree can have the size (by the bound to s' above) at most $s \leq 2|T^a|+1$: If it were $|T^a| < (1/3)|T|$ then the previous subtree had the size $\leq (2/3)|T|$ and the process should have stopped then.

As $|T_a| = |T| - |T^a| + 1$, the inequalities $(1/3)|T| \leq |T_a| \leq (2/3)|T|$ hold too.

q.e.d.

Recall the definition of a partial truth assignment α_g from the proof of Theorem 2.2.2.

Theorem 2.2.4 *Any R^*-proof of PHP_n must have the size at least $(3/2)^{n-2}$.*

Proof :

Let k be the number of clauses in some R^*-proof π of PHP_n. We shall construct a 4-tuple g_u, $\mathcal{D}_u$, E_u and S_u where:

1. $g_u :\subseteq [n] \to [n] \setminus \{0\}$ is a partial 1-to-1 map such that $|g_u| \leq u$.
2. $\mathcal{D}_u$ is a set of clauses (in the literals of PHP_n) each of which is forced true by α_{g_u}.
3. E_u is a clause forced false by α_{g_u}.
4. S_u is an R^*-proof of E_u from clauses of PHP_n and $\mathcal{D}_u$.
5. $|S_u| \leq (2/3)^u k$.

Put $g_0 := \emptyset$, $S_0 := \pi$, $\mathcal{D}_0 := \emptyset$ and $E_0 := \emptyset$. Assume we have g_u, $\mathcal{D}_u$, E_u, S_u. Find, using Lemma 2.2.3, a node $a \in S_u$ splitting S_u in the 1/3 - 2/3 fashion of the lemma. Let D be the clause at the node a. Consider two cases:

(a) D can be forced true by some $h \supseteq g_u$, a partial 1-to-1 map from $[n]$ into $[n] \setminus \{0\}$.

(b) There is no such h.

In Case (a) note that such h need to extend g_u by at most one pair (i, j); i.e. we may assume that $h \setminus g_u| \leq 1$. This is because to make a clause true it suffices to make one literal true. Take any such h and define:

- $g_{u+1} := h$.
- $\mathcal{D}_{u+1} := \mathcal{D}_u \cup \{D\}$.
- $E_{u+1} := E_u$.
- $S_{u+1} := (S_u)_a$, i.e. the nodes in S_u that are not $> a$.

In Case (b) put $g_{u+1} := g_u$, $\mathcal{D}_{u+1} := \mathcal{D}_u$, $E_{u+1} := D$ and $S_{u+1} := (S_u)^a$.

It is easy to verify that the properties 1. - 4. required from the 4-tuples are maintained in the construction.

Now assume that ℓ is so large that S_ℓ is just one clause E_ℓ, i.e. $|S_\ell| = 1$. By the construction E_ℓ is forced false by g_ℓ. Hence it cannot be a clause from $\mathcal{D}_\ell$ and must be from PHP_n. But then, identically as in the proof of Theorem 2.2.2, it must hold that $|g_\ell| \geq n - 1$, i.e. that $\ell \geq n - 1$.

The lower bound is obtained by combining this inequality with the estimate that $\ell \leq \lceil \log_{3/2}(k) \rceil$ is sufficient to enforce $|S_\ell| = 1$ (by $|S_\ell| \leq (2/3)^\ell k$):

$$\lceil \log_{3/2}(k) \rceil \geq n - 1 \quad \text{, so} \quad k \geq (3/2)^{n-2} \ .$$

q.e.d.

Although the construction looks formally different from the argument in Theorem 2.2.2, it is not really. We leave it as an Exercise 2.9.3 to turn the construction into a construction of an $\exists$-*decision tree*, a decision tree that branches according to the truth value of a clauses rather than of an atom.

Another direction to which it is possible to generalize this bound is to consider a proof system that operates not only with clauses formed from literals but with clauses formed from small conjunctions of literals (cf.[28]).We shall get back to this in **??**.

2.3 Effective interpolation: A general set-up

Assume that U and V are two disjoint $\mathcal{NP}$-sets (subsets of $\{0,1\}^*$). By the proof of the $\mathcal{NP}$-completeness of satisfiability there are sequences of propositional formulas $A_n(p_1,\ldots,p_n,q_1,\ldots,q_{t_n})$ and $B_n(p_1,\ldots,p_n,r_1,\ldots,r_{s_n})$ such that the size of A_n and B_n is $n^{O(1)}$ and such that

$$U_n := U \cap \{0,1\}^n \ = \ \{(\epsilon_1,\ldots,\epsilon_n) \in \{0,1\}^n \mid \exists \alpha_1,\ldots,\alpha_{t_n}\ A_n(\overline{\epsilon},\overline{\alpha}) \text{ holds}\}$$

and

$$V_n := V \cap \{0,1\}^n \ = \ \{(\epsilon_1,\ldots,\epsilon_n) \in \{0,1\}^n \mid \exists \beta_1,\ldots,\beta_{s_n}\ B_n(\overline{\epsilon},\overline{\beta}) \text{ holds}\}\ .$$

The assumption that $U \cap V = \emptyset$ is equivalent to the statement that the implications

$$A_n \longrightarrow \neg B_n$$

are all tautologies. By the Craig interpolation theorem [16, 17] (see Exercise 2.9.6) there is a formula $I_n(\overline{p})$ built only from atoms $\overline{p}$ such that both implications:

$$A_n \rightarrow I_n \ \text{ and } \ I_n \rightarrow \neg B_n$$

are tautologies. This means that the set defined by I_n:

$$W := \bigcup_n \{\overline{\epsilon} \in \{0,1\}^n \mid I_n(\overline{\epsilon}) \text{ holds }\}$$

separates U from V:

$$U \subseteq W \text{ and } W \cap V = \emptyset\ .$$

Hence a lower bound to a complexity of interpolating formulas is also a lower bound on the complexity of sets separating disjoint $\mathcal{NP}$-sets. We cannot really expect to polynomially bound the size of a formula or a circuit defining suitable W from the length of the implication $A_n \to \neg B_n$. This would immediately imply, as observed by Mundici [35, 36, 37], that $\mathcal{NP} \cap co\mathcal{NP} \subseteq \mathcal{NC}^1/poly$ or $\subseteq \mathcal{P}/poly$ (just take U and V two complementary $\mathcal{NP}$-sets).

The idea of effective interpolation (discussed first in Krajíček [24]) is more subtle: *For a given propositional proof system P, try to estimate the circuit-size of an interpolant of an implication in terms of the size of the shortest proof of the implication.*

Definition 2.3.1 *A pps P admits* effective interpolation[1] *iff there is a polynomial $p(x) \in \mathbf{N}[x]$ such that any implication with a P-proof of size m has an interpolant of a circuit size $\leq p(m)$.*

Exercise 2.9.7 shows why it is necessary to consider the circuit size and not just the formula size of the interpolant.

To start with, we have at least one example when this clearly works (we shall encounter LK in 3.4).

Example 2.3.2 *Cut-free propositional sequent calculus LK admits effective interpolation.*

The interpolating circuit is constructed by an obvious induction on the number of sequents in an LK-proof. (This is the base case in the usual proof-theoretic proof of Craig's interpolation theorem via cut-elimination, see, for example, [25, 4.3].)

The point of the effective interpolation method is that by establishing a good *upper bound* for a proof system P in the form of the effective interpolation we prove *lower bounds* on the size of P-proofs. Namely:

Theorem 2.3.3 *Assume that U and V are two disjoint $\mathcal{NP}$-sets such that U_n and V_n are inseparable by a set of circuit complexity $\leq s(n)$, all $n \geq 1$. Assume that P admits effective interpolation.*

Then the implications $A_n \to \neg B_n$ require P-proofs of size $\geq s(n)^\epsilon$, some $\epsilon > 0$.

---

[1] This is sometimes called *feasible interpolation.* I prefer the original name as in some applications the interpolant is not feasible (in the usual meaning of the term as beeing - uniform or nonuniform - p-time) but it is still in some sense *effective.*

Proof :

By the effective interpolation, a proof of the implication of size $\leq s$ yields an interpolant of circuit size $\leq p(s)$, some fixed polynomial. Pick $\epsilon > 0$ such that $p(s^\epsilon) \leq s$ for all $s \geq 1$.

q.e.d.

An apriori difficulty with this strategy how to get proof complexity lower bounds is that no non-trivial circuit lower bounds are known.

We shall overcome the difficulty by considering the *monotone* version of the effective interpolation. This will work because strong lower bounds to monotone circuits are known.

In the monotone version we consider separations of two $\mathcal{NP}$-sets U and V as earlier but now we assume that U is closed upwards:

$$u \in U_n \ \wedge \ u \leq u' \ \rightarrow \ u' \in U_n$$

where the ordering $u \leq u'$ on $\{0,1\}^n$ means that $u_i \leq u'_i$, for all bits $i \leq n$. If $U \cap V = \emptyset$ and U is closed upwards then U and V can be separated by W that is also closed upwards (e.g. by U itself). The same conclusion is true if we assume instead that V is closed downwards - we shall not discuss this dual case.

The propositional version of the monotone interpolation is the following statement.

Lemma 2.3.4 (Lyndon's theorem) *Assume that $A(\overline{p},\overline{q}) \rightarrow B(\overline{p},\overline{r})$ is a tautology, and that the atoms p_i' occur only positively (i.e. in the scope of an even number of negations) in A.*

Then there is a monotone interpolant $I(\overline{p})$ of the implication, an interpolant in which all p_i' also occur only positively.

Definition 2.3.5 *A pps P admits* monotone effective interpolation *iff there is a polynomial $p(x) \in \mathbf{N}[x]$ such that any implication with a P-proof of size m has a monotone interpolant of a monotone circuit size $\leq p(m)$.*

Similarly as Theorem 2.3.3 we get

Theorem 2.3.6 *Assume that U and V are two disjoint $\mathcal{NP}$-sets with U closed upwards. Assume that U_n and V_n are inseparable by a set closed upwards and of monotone circuit complexity $\leq s(n)$, all $n \geq 1$. Assume that P admits monotone effective interpolation.*

Then the implications $A_n \to \neg B_n$ require P-proofs of size $\geq s(n)^\epsilon$, some $\epsilon > 0$.

Now we give an example of two $\mathcal{NP}$-sets U and V, U closed upwards (and also, in fact, V closed downwards) for which it is known that any monotone separating set must be defined by a large monotone circuit.

In the next definition we denote the set of two-element subsets of $\{1, \ldots, n\}$ by the suggestive symbol $\binom{n}{2}$.

Definition 2.3.7 *Let $n, \omega, \xi \geq 1$. The set $Clique_{n,\omega}(p, q)$ is a set of the following clauses in the atoms p_{ij}, $\{i, j\} \in \binom{n}{2}$, and q_{ui}, $u = 1, \ldots, \omega$ and $i = 1, \ldots, n$:*

1. $\bigvee_{i \leq n} q_{ui}$, *all* $u \leq \omega$,
2. $\neg q_{ui} \vee \neg q_{vi}$, *all* $u < v \leq \omega$ *and* $i = 1, \ldots, n$,
3. $\neg q_{ui} \vee \neg q_{vj} \vee p_{ij}$, *all* $u < v \leq \omega$ *and* $\{i, j\} \in \binom{n}{2}$.

The set $Color_{n,\xi}(p, r)$ is the set of the following clauses in the atoms p_{ij}, $\{i, j\} \in \binom{n}{2}$, and r_{ia}, $i = 1, \ldots, n$ and $a = 1, \ldots, \xi$:

1. $\bigvee_{a \leq \xi} r_{ia}$, *all* $i \leq n$,
2. $\neg r_{ia} \vee \neg r_{ib}$, *all* $a < b \leq \xi$ *and* $i \leq n$,
3. *]* $\neg r_{ia} \vee \neg r_{ja} \vee \neg p_{ij}$, *all* $a \leq \xi$ *and* $\{i, j\} \in \binom{n}{2}$.

Truth assignments to atoms p_{ij} can be identified with undirected graphs with the vertex set $[n]$. Truth assignments to q_{ui} such that $Clique_{n,\omega}(p, q)$ is satisfied can be identified with 1-to-1 maps from the set $[\omega]$ onto a clique (i.e. a complete subgraph) in the graph determined by p, and truth assignments to r_{ia} such that $Color_{n,\xi}(p, r)$ is satisfied can be identified with colorings of the graph by ξ colors. The set

$$\{p \mid \exists q \; Clique_{n,\omega}(p, q)\}$$

is the set of graphs on $[n]$ with a clique of size $\geq \omega$, while the set

$$\{p \mid \exists r \; Color_{n,\xi}(p, r)\}$$

is the set of graphs on $[n]$ colorable by $\leq \xi$ colors.

Note that the atoms p_i' occur only positively in clauses in *Clique* and only negatively in *Color* and, indeed, the two sets are closed upwards and downwards respectively.

The implication

$$\bigwedge Clique_{n,\omega} \ \rightarrow\ \neg \bigwedge Color_{n,\xi}$$

is obviosuly a tautology if $\omega > \xi$.

The following theorem just restates the bound from [5][2].

Theorem 2.3.8 (Alon-Boppana[5]) *Assume that* $3 \leq \xi < \omega$ *and* $\sqrt{\xi}\omega \leq \frac{n}{8 \log n}$ *. Then the implication*

$$\bigwedge Clique_{n,\omega} \ \rightarrow\ \neg \bigwedge Color_{n,\xi}$$

has no interpolant of the monotone circuit-size smaller than:

$$2^{\Omega(\sqrt{\xi})} \ .$$

A suitable choice of parameters is $\xi := \lceil \sqrt{n} \rceil$ and $\omega := \xi + 1$. The lower bound provided by the theorem is then $2^{\Omega(n^{1/4})}$.

2.4 Communication complexity interlude

We shall prove in Section 2.5 that R admits both monotone and nonmonotone effective interpolation. First we need to recall, in this section, few notions and facts from communication complexity. This will be a base of a universal method for proving effective interpolation.

Let $U_n, V_n \subseteq \{0,1\}^n$ be two disjoint sets. *Karchmer-Wigderson* game on U_n, V_n (introduced in [21]) is played by two players A and B. Player A receives $u \in U$ while B receives $v \in V$. They communicate bits of information (following a protocol previously agreed on) until both players agree on the same $i \in [n]$ such that $u_i \neq v_i$. A measure of the complexity of the game is the minimum (over all protocols) of the number of bits they need to communicate in the worst case. This minimum is called the *communication complexity* of the game and it is denoted by $C(U_n, V_n)$.

[2]One needs to replace the class of graphs without a clique of size ξ used in [5] by the smaller class of ξ-colorable graphs. It is the bound to monotone circuits separating these two classes what is actually proved in [5].

Assume that we have a propositional formula (in the DeMorgan language) $\varphi(p_1, \ldots, p_n)$ that is constantly 1 and 0 on U_n and V_n respectively. We say that such φ separates U_n *from* V_n. Applying the DeMorgan rules if necessary, we may assume that the negations in φ are applied only to atoms.

The players can use such a formula as follows. They start at the top connective, i.e. at the whole formula, and will work down to smaller and smaller subformulas until reaching a literal. The property they will preserve is that the current subformula gives value 1 on u and 0 on v. This is true at the beginning, by the hypothesis. If the top connectiv is a conjunction the player B indicates to A, by sending one bit, which of the two subformulas yields value 0 on v. If the top conncetive is a disjunction, analogously A indicates to B which of the two subformulas is 1 on u. This argument proves a half of the following simple but important statement (for the other half see Exercise 2.9.8).

In the monotone version of the game U_n is assumed to be closed upwards, and the players search for i such that $u_i = 1 \wedge v_i = 0$ (and not just $u_i \neq v_i$). Any monotone formula separating U_n from V_n can be used by the players as a protocol, identically as above. Let $MC(U_n, V_n)$ be the monotone communication complexity of the game.

Theorem 2.4.1 (Karchmer-Wigderson[21]) *Let $U_n, V_n \subseteq \{0,1\}^n$ be two disjoint sets. Then $C(U_n, V_n)$ is equal to the minimal depth of a DeMorgan formula separating U_n from V_n.*

The same is true in the monotone case: $MC(U_n, V_n)$ is equal to the minimal depth of a monotone DeMorgan formula separating U_n from V_n.

If the players had a circuit C separating U_n from V_n instead of the formula φ they could use the same communication protocol. But the communication complexity would be still bounded only by the depth of C which really says nothing about the size of C. To capture the complexity of protocols comming from circuits we need to use a more general notion of protocol. The definition is a variant of a notion from [39] that used PLS-problems.

Definition 2.4.2 ([27]) *Let $U_n, V_n \subseteq \{0,1\}^n$ be two disjoint sets. A* protocol *for the Karchmer-Wigderson game on the pair (U_n, V_n) is a labelled directed graph G satisfying the following conditions:*

1. *G is acyclic and has one source denoted $\emptyset$.*

   The nodes with the out-degree 0 *are* leaves, *all other are* inner nodes.

2. *Leaves are labelled by one of the following formulas:*

$$u_i = 1 \wedge v_i = 0 \quad or \quad u_i = 0 \wedge v_i = 1$$

for some $i = 1, \ldots, n$.

3. *There is a function* $S(u, v, x)$ *(the* strategy*) such that* S *assigns to a node* x *and a pair* $(u, v) \in U_n \times V_n$ *an edge leaving from the node* x.

Fixing a pair $(u, v) \in U_n \times V_n$ *the strategy defines for every node* x *a directed path* $P^x_{uv} = x_1, \ldots, x_h$ *in* G*: Start at* x *and go towards a leaf* x_h*, always going from* x_i *using the edge* $S(u, v, x_i)$.

4. *For every* $(u, v) \times \in U_n \times V_n$ *there is a set* $F(u, v) \subseteq G$ *satisfying:*

   (a) $\emptyset \in F(u, v)$.
   (b) $x \in F(u, v) \to P^x_{u,v} \subseteq F(u, v)$.
   (c) The label of any leaf from $F(u, v)$ *is valid for* u, v.

   Such a set F *is called the* consistency condition.

A protocol is called monotone *iff every leaf in it is labelled by one of the formulas* $u_i = 1 \wedge v_i = 0$, $i = 1, \ldots, n$.

The communication complexity of G *is the minimal number* t *such that for every* $x \in G$ *the players (one knowing* u *and* x*, the other one* v *and* x*) decide whether* $x \in F(u, v)$ *and compute* $S(u, v, x)$ *with at most* t *bits exchanged in the worst case.*

See Exercise 2.9.9 about the consistency condition.

Now let us observe that this notion naturally formalizes protocols formed from a circuit (as described above). Assume that C is a circuit separating U_n from V_n. Reverse the edges in C, take for $F(u, v)$ those subcircuits differing in the value on u and v, and define the strategy and the labels of the leaves in an obvious way. This determines a protocol for the game on (U_n, V_n) whose communication complexity is 2. The next theorem says that there is a converse construction. The theorem reformulates a statement from [39] but we give it a new proof which then applies to generalizations in **??**.

Theorem 2.4.3 ([39]) *Let* $U_n, V_n \subseteq \{0, 1\}^n$ *be two disjoint sets. Let* G *be a protocol for the game on* U_n, V_n *which has* k *nodes and the communication complexity* t.

Then there is a circuit C of size $k2^{O(t)}$ separating U_n from V_n. Moreover, if G is monotone so is C.

On the other hand, any (monotone) circuit C of size s separating U_n from V_n determines a (monotone) protocol G with s nodes whose communication complexity is 2.

Proof :

The second part of the theorem was explained already, so let us concentrate on the first part. Let G be a protocol satisfying the hypothesis. For a node a and $w \in \{0,1\}^t$, let $R_{a,w}$ be the set of pairs $(u,v) \in U_n \times V_n$ such that the communication of the players deciding $a \in_? F(u,v)$ evolves according to w and ends with the affirmation of the membership. It is easy to see that $R_{a,w}$ is a rectangle, i.e. of the form $R_{a,w} = U_{a,w} \times V_{a,w}$ for some $U_{a,w} \subseteq U_n$ and $V_{a,w} \subseteq V_n$.

For a node a denote by k_a the number of nodes in G that can be reached from a by a (directed) path. So $k_a = 1$ for a a leaf, while $k_\emptyset = k$ for the source $\emptyset$.

Claim 1: *For all $a \in G$ and $w \in \{0,1\}^t$ there is a circuit $C_{a,w}$ separating $U_{a,w}$ from $V_{a,w}$ and of size $\leq k_a 2^{O(t)}$.*

(The constant in the $O(t)$ is independent of a.) This implies the theorem taking for a the source (which is in all $F(u,v)$).

The claim is proved by induction on k_a. If a is a leaf the statement is clear. Assume a is not a leaf and let $w \in \{0,1\}^t$. For $u \in U_{a,w}$ let $u^* \in \{0,1\}^{4^t}$ be a vector whose bits u^*_ω are parametrized by $\omega = (\omega_1, \omega_2) \in \{0,1\}^t \times \{0,1\}^t$ and such that $u^*_\omega = 1$ iff there is a $v \in V_{a,w}$ such that the communication of the players computing $S(u,v,a)$ evolves according to ω_1 and the computation of $S(u,v,a) \in_? F(u,v)$ evolves according to ω_2. Define $v^*_\omega \in \{0,1\}^{4^t}$ dually: $v^*_\omega = 0$ iff there is a $u \in U_{a,w}$ such that the communication of the players computing $S(u,v,a)$ evolves according to ω_1 and the computation of $S(u,v,a) \in_? F(u,v)$ evolves according to ω_2.

Let $U^*_{a,w}$ and $V^*_{a,w}$ be the sets of all these u^* and v^* respectively.

Claim 2: *There is a monotone formula $\varphi_{a,w}$ (in 4^t atoms) separating $U^*_{a,w}$ from $V^*_{a,w}$ of size $2^{O(t)}$.*

Claim 2 follows from Theorem 2.4.1 as there is an obvious way how the players can find a bit ω in which $u^*_\omega = 1$ and $v^*_\omega = 0$: They simply compute $S(u,v,a)$ (this gives them ω_1) and then decide $S(u,v,a) \in_? F(u,v)$ (this gives them ω_2).

Let us resume the proof of Claim 1. For $\omega_1 \in \{0,1\}^t$ let a_{ω_1} be the node $S(u,v,a)$ computed for some u,v with communication ω_1. Define a circuit:

$$C_{a,w} \; := \; \varphi_{a,w}(\ldots,\; y_{\omega_1,\omega_2}/C_{a_{\omega_1},\omega_2},\ldots)$$

that is, we substitute the circuit $C_{a_{\omega_1},\omega_2}$ in the position of the (ω_1,ω_2)-th variable in $\varphi_{a,w}$.

As $k_{a_{\omega_1}} < k_a$, the induction hypothesis implies that all $C_{a_{\omega_1},\omega_2}$ work correctly on all $U_{a_{\omega_1},\omega_2} \times V_{a_{\omega_1},\omega_2}$. The circuit $C_{a,w}$ works then correctly by the definition of the formula $\varphi_{a,w}$.

This concludes the proof of the general case. But the same proof gives also the monotone case (as $\varphi_{a,w}$ is monotone).

q.e.d.

2.5 Effective interpolation for resolution

In this section we prove the effective interpolation for resolution.

Theorem 2.5.1 (Krajíček[27]) *Assume that the set of clauses*

$$\{A_1,\ldots,A_m,B_1,\ldots,B_\ell\}$$

where:

1. $A_i \subseteq \{p_1,\neg p_1,\ldots,p_n,\neg p_n,q_1,\neg q_1,\ldots,q_s,\neg q_s\}$, *all* $i \leq m$
2. $B_j \subseteq \{p_1,\neg p_1,\ldots,p_n,\neg p_n,r_1,\neg r_1,\ldots,r_t,\neg r_t\}$, *all* $j \leq \ell$

has a resolution refutation with k *clauses.*

Then the implication:

$$\bigwedge_{i\leq m}(\bigvee A_i) \;\longrightarrow\; \neg\bigwedge_{j\leq \ell}(\bigvee B_j)$$

(where $\bigvee C$ *denotes the disjunction of the literals in a clause* C*) has an interpolant* $I(p)$ *whose circuit-size is* $kn^{O(1)}$.

Moreover, if all atoms p_i*' occur only positively in all* A_i *then there is a monotone interpolant whose monotone circuit-size is* $kn^{O(1)}$.

Before we prove the theorem let us note a corollary of the theorem and Theorems 2.3.6 and 2.3.8, our first exponential lower bound for R.

Corollary 2.5.2 ([27]) *There is a constant $c > 0$ such that whenever $3 \leq \xi < \omega$ and $\sqrt{\xi}\omega \leq \frac{n}{8 \log n}$ the following holds.*

Any R-proof of the implication

$$\bigwedge Clique_{n,\omega} \ \rightarrow \ \neg \bigwedge Color_{n,\xi}$$

must have at least $n^{-c} 2^{\xi^{1/2}}$ clauses.

In particular, if $n^{\Omega(1)} \leq \xi < \omega < n^{2/3}$ then any such refutation must have $2^{n^{\Omega(1)}}$ clauses.

Proof of Theorem 2.5.1:

Assume that π is an R-refutation with k clauses of $\{A_1, \ldots, A_m, B_1, \ldots, B_\ell\}$, a set of clauses satisfying the hypothesis of the theorem. Let U and V be the subsets of $\{0,1\}^n$ defined by

$$U \ := \ \{p \in \{0,1\}^n \mid \exists q \in \{0,1\}^s, \bigwedge_i \bigvee A_i\}$$

and by

$$V \ := \ \{p \in \{0,1\}^n \mid \exists r \in \{0,1\}^t, \bigwedge_j \bigvee B_j\}$$

respectively. Eventually we shall show how to transform π into a protocol for the Karchmer-Wigderson game on U, V, of size $k + 2n$ and of the communication complexity $O(\log n)$. But we start with a less formal argument.

Assume that $\pi = D_1, \ldots, D_k$. For D a clause let $\tilde{D}$ denote the set of all truth assignments satisfying D.

Assume player A gets $u \in U$ and player B gets $v \in V$. A fixes some $q^u \in \{0,1\}^s$ such that $\bigwedge \bigvee A_i(u, q^u)$ holds, and similarly B picks some $r^v \in \{0,1\}^t$, a witness of the membership of v in V.

The players will construct a path $P = S_0, \ldots, S_h$ through π, from the endsequent ($= S_0$) to one of the initial sequents. The property they will try to maintain is that the truth evaluations (u, q^u, r^v) and (v, q^u, r^v) do not satisfy the clauses on the path, i.e. are not in $\tilde{S}_a$, $a = 0, \ldots, h$.

Assume the players reach S_a which was deduced in π by the inference:

$$\frac{X \quad Y}{S_a} \ .$$

They first determine whether $(u, q^u, r^v) \in \tilde{X}$ and $(v, q^u, r^v) \in \tilde{X}$, and then continue depending on a possible outcome:

1. $(u, q^u, r^v) \in \tilde{X} \wedge (v, q^u, r^v) \in \tilde{X}$.

2. $(u, q^u, r^v) \notin \tilde{X} \wedge (v, q^u, r^v) \notin \tilde{X}$.

3. Exactly one of $(u, q^u, r^v), (v, q^u, r^v)$ is in $\tilde{X}$.

In the first case none of the two tuples can be in $\tilde{Y}$ and the players put $S_{a+1} := Y$. In the second case they take $S_{a+1} := X$. It is the third case which is most interesting: Necessarily $u \neq v$ and the players stop constructing the path and enter a protocol aimed at finding $i \leq n$ such that $u_i \neq v_i$.

As each initial sequent is satisfied by either (u, q^u, r^v) or by (v, q^u, r^v), the players must sooner or later enter the third possibility and thus find $i \leq n$ such that $u_i \neq v_i$.

For this to work we need to show that each of the three tasks:

1. Decide whether $(u, q^u, r^v) \in \tilde{D}$.

2. Decide whether $(v, q^u, r^v) \in \tilde{D}$.

3. If $(u, q^u, r^v) \in \tilde{D} \not\equiv (v, q^u, r^v) \in \tilde{D}$ find $i \leq n$ such that $u_i \neq v_i$.

where D is a clause, has small communication complexity. But this is easy: The first two can be decided by each player sending one bit (the truth value of the part of the clause he can evaluate), the third task needs $\log n$ bits by a binary search.

Let us now define the protocol G formally. G has $(k + 2n)$ nodes, the k clauses of π together $2n$ extra vertices. These extra vertices are labelled by formulas $u_i = 1 \wedge v_i = 0$ and $u_i = 0 \wedge v_i = 1$, $i = 1, \ldots, n$.

The consistency condition $F(u, v)$ is formed by those clauses D_j that are not satisfied by (v, q^u, r^v), and also by those of the extra $2n$ nodes whose label is valid for the pair u, v.

The strategy function $S(u, v, D_j)$ (for D_j derived from X and Y) is defined as follows:

1. If $(u, q^u, r^v) \notin \tilde{D}_j$ then
$$S(u, v, D_j) := \begin{cases} X & \text{if } (v, q^u, r^v) \notin \tilde{X} \\ Y & \text{if } (v, q^u, r^v) \in \tilde{X} \text{ (and hence } (v, q^u, r^v) \notin \tilde{Y}). \end{cases}$$

2. If $(u, q^u, r^v) \in \tilde{D}_j$ then the players use binary search for finding $i \leq n$ such that $u_i \neq v_i$. $S(u, v, D_j)$ is then the one of the two nodes labelled by $u_i = 1 \wedge v_i = 0$ and $u_i = 0 \wedge v_i = 1$ whose label is valid for the pair u, v.

Note that the strategy function $S(u, v, x)$ as well as the membership relation $x \in F(u, v)$ can be determined by the players exchanging at most $\log n$ bits. As G has $(k+2n)$ nodes, Theorem 2.4.3 yields a circuit separating U from V and having the size at most $(k + 2n) \cdot 2^{O(\log n)} = kn^{O(1)}$.

Now we turn to the monotone case, which requires a modification. Assume that the atoms p_j' occur only positively in all A_i's. Note that this means that U is closed upwards but even a bit more: If $u \in U$ and q^u is a witness for this, and $u \leq u'$, then q^u also witnesses the membership $u' \in U$.

The protocol in the monotone case will have only $(k + n)$ nodes, the k clauses of π plus n extra nodes labelled by formulas $u_i = 1 \wedge v_i = 0$, $i = 1, \ldots, n$. The consistency condition $F(u, v)$ is defined as before.

The strategy function changes a bit. In the third case of the construction of the path above assume that $(u, q^u, r^v) \in \tilde{X}$ while $(v, q^u, r^v) \notin \tilde{X}$. Then the players, instead of using the binary search for finding the bit in which u differs from v, they either find $i \leq n$ such that

$$u_i = 1 \wedge v_i = 0$$

or learn that there is some u' satisfying

$$u' \geq u \wedge (u', q^u, r^v) \notin \tilde{X}$$

This can be done by the player A only, in fact, and hence he just need to communicate $\log n$ bits identifying i to B.

Formally, in the first case they define

$$S(u, v, D_j) := \begin{cases} X & \text{if } (v, q^u, r^v) \notin \tilde{X} \\ Y & \text{if } (v, q^u, r^v) \in \tilde{X}. \end{cases}$$

In the second case $S(u, v, D_j)$ is simply the additional node with the label $u_i = 1 \wedge v_i = 0$.

By the monotonicity condition assumed about $A_1, \ldots, A_m$, for every u' occurring above it holds:

$$(u', q^u, r^v) \in \bigcap_{j \leq m} A_j \ .$$

This implies that the players again have to, sooner or later, enter the option leading to $i \leq n$ such that $u_i = 1 \wedge v_i = 0$.

So we get $(k + n) \cdot 2^{O(t)} = kn^{O(1)}$ bound to the size of a monotone separating circuit (by Theorem 2.4.3).

q.e.d.

2.6 Generalizations and limitations of effective interpolation

Note that the proof of Theorem 2.5.1 does not really use any particular information about the syntax of R; it works with the sets of satisfying assignments. This means that we can generalize effective interpolation to a more general situation which is grasped by the following concept.

Definition 2.6.1 ([27]) *Let $N \geq 1$.*

1. *The* semantic rule *allows to infer from two subsets $A, B \subseteq \{0,1\}^N$ a third one:*
$$\frac{A \quad B}{C}$$
iff $C \supseteq A \cap B$.

2. *A* semantic derivation *of the set $C \subseteq \{0,1\}^N$ from sets $A_1, \ldots, A_m \subseteq \{0,1\}^N$ is a sequence of sets $B_1, \ldots, B_k \subseteq \{0,1\}^N$ such that $B_k = C$, and such that each B_i is either one of A_j' or derived from two previous B_{i_1}, B_{i_2}, $i_1, i_2 < j$, by the semantic rule.*

3. *Let $\mathcal{X} \subseteq exp(\{0,1\}^N)$ be a family of subsets of $\{0,1\}^N$. A semantic derivation $B_1, \ldots, B_k$ is an $\mathcal{X}$-derivation iff all $B_i \in \mathcal{X}$.*

Derivability in semantic derivations, without a restriction to some $\mathcal{X}$, would be rather trivial: C is derivable from A_i's iff $C \supseteq \bigcap_i A_i$. But when the family $\mathcal{X}$ is not a filter on $\{0,1\}^N$, the notion of $\mathcal{X}$-derivability becomes non-trivial. For example, a family formed by the subsets of $\{0,1\}^N$ definable by a clause yields a non-trivial notion. The following technical definition abstracts a property of sets of truth assignments used in the proof of Theorem 2.5.1.

Definition 2.6.2 *Let $N = n + s + t$ be fixed and let $A \subseteq \{0,1\}^N$. Let $u, v \in \{0,1\}^n$, $q^u \in \{0,1\}^s$ and $r^v \in \{0,1\}^t$.*

The communication complexity *of A, $CC(A)$, is the minimal number of bits two players (one knowing u, q^u and the other one knowing v, r^v) need to exchange in the worst case in solving any of the following three tasks:*

1. *Decide whether $(u, q^u, r^v) \in A$.*

2. *Decide whether $(v, q^u, r^v) \in A$.*

3. *If $(u, q^u, r^v) \in A \not\equiv (v, q^u, r^v) \in A$ find $i \leq n$ such that $u_i \neq v_i$.*

The monotone communication complexity w.r.t. U *of* A, $MCC_U(A)$, *is the minimal* $t \geq CC(A)$ *such that the next task can be solved communicating* $\leq t$ *bits in the worst case.*

4. If $(u, q^u, r^v) \in A$ *and* $(v, q^u, r^v) \notin A$ *either find* $i \leq n$ *such that*

$$u_i = 1 \wedge v_i = 0$$

or learn that there is some u' *satisfying*

$$u' \geq u \wedge (u', q^u, r^v) \notin A$$

Note that proofs in any of the usual propositional calculi based on bounded arity inference rules translate into semantic derivations: Replace a clause, (a sequent, a formula, an equation, etc.) by the set of its satisfying truth assignments. The soudness of the inference rules implies that they translate into instances of the semantic rule.

The point of this generalization is that we can lift the effective interpolation from R to this context. Let $N = n + s + t$ be fixed for now. For $A \subseteq \{0,1\}^{n+s}$ define the set $\tilde{A}$ by:

$$\tilde{A} := \bigcup_{(a,b)\in A} \{(a,b,c) \mid c \in \{0,1\}^t\}$$

where a, b, c range over $\{0,1\}^n$, $\{0,1\}^s$ and $\{0,1\}^t$ respectively, and similarly for $B \subseteq \{0,1\}^{n+t}$ define $\tilde{B}$:

$$\tilde{B} := \bigcup_{(a,c)\in B} \{(a,b,c) \mid b \in \{0,1\}^s\} \ .$$

Theorem 2.6.3 *Let* $A_1, \ldots, A_m \subseteq \{0,1\}^{n+s}$ *and* $B_1, \ldots, B_\ell \subseteq \{0,1\}^{n+t}$. *Assume that there is a semantic derivation* $\pi = D_1, \ldots, D_k$ *of the empty set* $\emptyset = D_k$ *from the sets* $\tilde{A}_1, \ldots, \tilde{A}_m, \tilde{B}_1, \ldots, \tilde{B}_\ell$.

If the communication complexity of all D_i, $i \leq k$, *satisfies* $CC(D_i) \leq t$ *then the two sets*

$$U = \{u \in \{0,1\}^n \mid \exists q^u \in \{0,1\}^s; (u, q^u) \in \bigcap_{j\leq m} A_j\}$$

and

$$V = \{v \in \{0,1\}^n \mid \exists r^v \in \{0,1\}^t; (v, r^v) \in \bigcap_{j\leq \ell} B_j\}$$

can be separated by a circuit of size at most $(k+2n)2^{O(t)}$.

Further, if the sets $A_1, \ldots, A_m$ *satisfy the following monotonicity condition w.r.t.* U*:*

$$(u, q^u) \in \bigcap_{j \le m} A_j \wedge u \le u' \ \rightarrow\ (u', q^u) \in \bigcap_{j \le m} A_j$$

and $MCC_U(D_i) \le t$ *for all* $i \le k$*, then there is a monotone circuit separating* U *from* V *of size at most* $(k+n)2^{O(t)}$.

The proof is identical to the proof of Theorem 2.5.1.

Theorem 2.6.3 can be used to give exponential lower bounds for various proof systems of "geometric nature" (see Exercises 2.9.10, 2.9.11 and Chapter **??**). A particular proof system of this type that has been studied in the connections with the linear programming is the *cutting planes* proof system CP, introduced in [15]. This system operates with integer linear inequalities of the form $a_1x_1 + \ldots a_nx_n \ge b$, with x_i representing the truth values of atoms. CP has some obvious rules: adding two inequalities, multiplying an inequality by a positive constant, but also a less obvious one, the division rule:

$$\frac{a_1x_1 + \ldots a_nx_n \ge b}{\frac{a_1}{c}x_1 + \ldots \frac{a_n}{c}x_n \ge \lceil \frac{b}{c} \rceil}$$

provided that $c > 0$ and $c|a_i$, all i (the rounding up is what makes the system complete). CP has also two initial inequalities: $x \ge 0$, $-x \ge -1$. It is a refutation system which derives from an unsatisfiable system of inequalities the inequality $0 \ge 1$. The term *unsatisfiable* means that the system has no 0-1 solution. It is sound and complete and polynomially simulates resolution, see [15] or [25, 13.1].

We shall discuss effective interpolation for CP in Chapter **??**, together with a generalization of communication complexity from the Boolean framework to the so called real communication complexity. .

Finally, let us discuss an apriori limitation to the monotone effective interpolation method. Assume that

$$Clique_{n,\omega}\ \cup\ Color_{n,\xi}$$

were satisfiable. The satisfying assignment then defines a map from $[\omega]$ into $[\xi]$ that is 1-to-1 (composing the map from $[\omega]$ onto a clique with the coloring restricted to the clique). More formally, we can define propositional formulas E_{au} for $a \in [\omega]$ and $u \in [\xi]$ (built from the atoms p, q, r) and derive from $Clique_{n,\omega} \cup Color_{n,\xi}$ by a p-size R-proof that E_{au}'s define a graph of an injective function from $[\omega]$ into $[\xi]$. Hence whenever a proof system can prove the instance of the pigeonhole principle saying that no such map exists, it also shortly proves the unsatisfiability of $Clique_{n,\omega} \cup Color_{n,\xi}$ and hence cannot admit monotone effective interpolation. Such instances of the pigeonhole principle are provable in F and even in very weak subsystems of F (see Chapter **??**).

One can also prove limitations to non-monotone effective interpolation but only modulo unproven cryptographical conjectures (like the security of RSA). More on this in **??**.

2.7 Width of resolution proofs

For a clause C, the *width* of C, denoted $w(C)$, is the number of literals in C. For a set $\mathcal{C}$ of clauses define $w(\mathcal{C}) := \max_{c \in \mathcal{C}} w(C)$. In particular, the width of a proof π, $w(\pi)$, is the maximal width of a clause in the proof.

Our aim in this section is to prove that a short R-proof can be transformed into a narrow proof. This will allow us to prove lower bounds for the size by proving sufficiently strong lower bounds on the width.

We shall use partial truth assignments called simply *restrictions*. The following notation will be handy. For ℓ a literal and $\epsilon \in \{0,1\}$ define:

$$\ell^\epsilon := \begin{cases} \ell & \text{if } \epsilon = 1 \\ \neg\ell & \text{if } \epsilon = 0 \end{cases}$$

Further, for ℓ and ϵ as above and C a clause define the restriction of C by $\ell = \epsilon$ to be the clause:

$$C \downarrow \ell = \epsilon := \begin{cases} C & \text{if neither } \ell \text{ nor } \neg\ell \text{ occur in } C \\ \{1\} & \text{if } \ell^\epsilon \in C \\ C \setminus \{\ell^{1-\epsilon}\} & \text{if } \ell^{1-\epsilon} \in C. \end{cases}$$

Similarly, for a set of clauses $\mathcal{C}$ put $\mathcal{C} \downarrow \ell = \epsilon := \{C \downarrow \ell = \epsilon \mid C \in \mathcal{C}\}$. Consider the effect a restriction, say $p = \epsilon$, has on a resolution inference:

$$\frac{X \cup \{q\} \qquad Y \cup \{\neg q\}}{X \cup Y}$$

If $p = q$ then the inference trasnforms into

$$\frac{X \qquad \{1\}}{X \cup Y} \quad \text{or} \quad \frac{Y \qquad \{1\}}{X \cup Y}$$

which is not a resolution inference. But it is an instance of a *weakening rule*:

$$\frac{Z_1}{Z_2} \quad \text{provided that} \quad Z_1 \subseteq Z_2$$

that is obviously sound. Moreover, a restriction of a weakening is again an instance of a weakening.

If $p \neq q$ and $p^\epsilon \in X \cup Y$ then the inference becomes

$$\frac{X \downarrow p = \epsilon \cup \{q\} \qquad Y \downarrow p = \epsilon \cup \{\neg q\}}{\{1\}}$$

which is again not a resolution inference. But we can simulate it by allowing $\{1\}$ as a new initial clause (axiom) in proofs.

Let R' be a proof system extending R by the weakening rule and by the new axiom. The point is that a restriction of an R'-proof is again an R'-proof (after transforming resolution inferences as described above). Clearly, lower bounds on R'-proofs apply, in particular, to R-proofs too.

The last piece of a usefull notation is $w(\mathcal{C} \vdash A)$, denoting the minimal width of an R'-derivation of a clause A from $\mathcal{C}$, and $\mathcal{C} \vdash_k A$ which stands for $k \geq w(\mathcal{C} \vdash A)$.

Lemma 2.7.1 *If $\mathcal{C} \downarrow p = 0 \vdash_k A$ then $\mathcal{C} \vdash_{k+1} A \cup \{p\}$.*
If $\mathcal{C} \downarrow p = 1 \vdash_k A$ then $\mathcal{C} \vdash_{k+1} A \cup \{\neg p\}$.

Proof :

We prove only the first part as the proof of the second part is identical. Assume that $\pi = D_1, \ldots, D_t$ is an R'-derivation of A from $\mathcal{C} \downarrow p = 0$ having the width k. Put $E_i := D_i \cup \{p\}$, for all $i \leq t$. We claim that $\pi' = E_1, \ldots, E_t$ is essentially an R'-derivation of $A \cup \{p\}$. The qualification essentially will be clear in a moment.

Assume first $D_i \in \mathcal{C} \downarrow p = 0$, say $D_i = C \downarrow p = 0$ for some $C \in \mathcal{C}$. Consider three cases:

1. $\neg p \in C$: Then $D_i = \{1\}$ and so $E_i = \{1, p\}$ can be derived from the axiom $\{1\}$ by a weakening.

2. $p \in C$: Then $D_i = C \setminus \{p\}$ and hence $E_i = C$ is an initial clause from $\mathcal{C}$.

3. $C \cap \{p, \neg p\} = \emptyset$: Then $D_i = C$ and so $E_i = C \cup \{p\}$ can be derived from C by a weakening.

Note that the extra line in the derivations of E_i' has the width bounded above by the width of clauses already in π'.

The case when D_i is derived in π by a resolution rule was already discussed when we motivated the extension of R to R'. The case when D_i is obtained by the weakening rule is trivial.

q.e.d.

Lemma 2.7.2 *For $\epsilon \in \{0,1\}$, assume that*

$$\mathcal{C} \downarrow p = \epsilon \vdash_{k-1} \emptyset \quad \textit{and} \quad \mathcal{C} \downarrow p = 1 - \epsilon \vdash_k \emptyset \ .$$

Then

$$w(\mathcal{C} \vdash \emptyset) \ \leq \ \max(k, w(\mathcal{C})) \ .$$

Proof :

By Lemma 2.7.1 the first part of the hypothesis implies $\mathcal{C} \vdash_k \{p^{1-\epsilon}\}$.

Resolve $\{p^{1-\epsilon}\}$ with all $C \in \mathcal{C}$ containing $\{p^{\epsilon}\}$; the width of all these inferences is bounded by $w(\mathcal{C})$. Therefore each clause $D \in \mathcal{C} \downarrow p = 1 - \epsilon$ has an R'-derivation from $\mathcal{C}$ of the width at most $\max(k, w(C))$.

This, together with the second part of the hypothesis of the lemma, concludes the proof.

q.e.d.

Theorem 2.7.3 (Ben-Sasson and Wigderson[8]) *Let $\mathcal{C}$ be an unsatisfiable set of clauses in literals $p_i, \neg p_i$, for $i \leq n$. Assume that $\mathcal{C}$ has a tree-like R'-refutation with $\leq 2^h$ clauses.*

Then:

$$w(\mathcal{C} \vdash \emptyset) \ \leq w(C) + h \ .$$

Proof :

We shall proceed by a double induction on n and h. If $n = 0$ or $h = 0$ then necessarily $\emptyset \in \mathcal{C}$ and there is nothing to prove. Assume that for $h_0 \geq 0$ the statement is true for all $h \leq h_0$ and for all $n \geq 0$. We shall prove that this is true also for $h_0 + 1$ by induction on n. By the above, we may assume that $n > 0$, hence there is a literal in $\mathcal{C}$.

Assume the last inference in a refutation π (having $\leq 2^{h_0+1}$ clauses) has been:

$$\frac{\{p\} \qquad\qquad \{\neg p\}}{\emptyset}$$

Hence one of the subproofs has the size $\leq 2^{h_0}$. Assume that it is the subproof π_0 ending with $\{p\}$. Restrict π_0 by $p = 0$; it becomes an R'-refutation of $\mathcal{C} \downarrow p = 0$. By the induction hypothesis for h_0:

$$w(\mathcal{C} \downarrow p = 0 \vdash \emptyset) \ \leq w(\mathcal{C} \downarrow p = 0) + h_0$$

Similarly, the restriction of the subproof π_1 ending with $\{\neg p\}$ by $p = 1$ becomes a refutation of $\mathcal{C} \downarrow p = 1$. It has, of course, $\leq 2^{h_0+1}$ clauses but it has $\leq n - 1$ atoms. So the induction hypothesis for $n - 1$ implies:

$$w(\mathcal{C} \downarrow p = 1 \vdash \emptyset) \ \leq w(\mathcal{C} \downarrow p = 1) + h_0 + 1$$

Applying Lemma 2.7.2 concludes the proof.

q.e.d.

Note that this immediately yields a lower bound to the size in terms of a lower bound to the width.

Corollary 2.7.4 *Every tree-like R' refutation of any $\mathcal{C}$ mus have the size*

$$\geq \ 2^{w(\mathcal{C} \vdash \emptyset) - w(\mathcal{C})}$$

Much more interesting is the following statement that shows that one can derive a lower bound to the size from one to the width even for general, not necessarily tree-like, R'-proofs.

Theorem 2.7.5 (Ben-Sasson and Wigderson [8]) *Let $\mathcal{C}$ be an unsatisfiable set of clauses in literals $p_i, \neg p_i$, for $i \leq n$.*

Then every R'-refutation must have the size at least

$$2^{\Omega(\frac{(w(\mathcal{C} \vdash \emptyset) - w(\mathcal{C}))^2}{n})} .$$

Proof :

Let k be the number of clauses in an R'-refutation π of $\mathcal{C}$. Let $h \geq 1$ be a parameter. Later we shall specify that $h := \lceil \sqrt{2n \log(k)} \rceil$ but this actual value is not used in the argument; it is only used at the end to optimize the bound.

We shall prove that

$$w(\mathcal{C} \vdash \emptyset) \ \leq \ w(\mathcal{C}) + O(\sqrt{n \log(k)}) \ .$$

If $n = 0$ then $\emptyset \in \mathcal{C}$ and there is nothing to prove.

Suppose $n > 0$. Call a clause C in π *wide* if $w(C) > h$. Let $s := (1 - \frac{h}{2n})^{-1}$.

By double induction on n and on t we prove that if the number of wide clauses in π is $< s^t$ then

$$w(\mathcal{C} \vdash \emptyset) \ \leq \ w(\mathcal{C}) + h + t \ .$$

Assume $t = 0$. Then there is no wide clause, i.e $w(\pi) \leq h \leq w(\mathcal{C}) + h$.

Now assume $t > 0$. One of the $2n$ literals, say ℓ, has to appear in at least $\frac{s^t h}{2n}$ wide clauses. Restrict π by $\ell = 1$. The clauses containing ℓ will be eliminated (they transform to $\{1\}$). Hence, in $\pi \downarrow \ell = 1$, a refutation of $\mathcal{C} \downarrow \ell = 1$, there remain less than

$$b \ - \ \frac{s^t h}{2n} \ = \ s^{t-1}$$

wide clauses. By the induction hypothesis for $t - 1$:

$$w(\mathcal{C} \downarrow \ell = 1 \vdash \emptyset) \ \leq \ w(\mathcal{C} \downarrow \ell = 1) + h + t - 1 \ .$$

Now apply to π the dual restriction $\ell = 0$. This produces a refutation $\pi \downarrow \ell = 0$ of $\mathcal{C} \downarrow \ell = 0$ where the number of wide clauses is still $< s^t$ but where the number of atoms is $n - 1$. Hence, by the induction hypothesis for $n - 1$:

$$w(\mathcal{C} \downarrow \ell = 0 \vdash \emptyset) \ \leq \ w(\mathcal{C} \downarrow \ell = 0) + h + t \ .$$

By applying Lemma 2.7.2 we get:

$$w(\mathcal{C} \vdash \emptyset) \ \leq \ w(\mathcal{C}) + h + t \ .$$

The particular value of the parameter h yields the wanted upper bound (using the estimate trivial $t \leq \log_s(k)$).

q.e.d.

In order to be able to prove via this theorem some lower bounds on the size of resolution proofs, we need unsatisfiable sets of clauses of small width (perhaps even constant) which require wide R-proofs. We shall construct such sets of clauses in Section 2.8.

2.8 Random sparse linear systems

Consider an $m \times n$ matrix A over the two-element field $\mathbf{F}_2$.. We call such a matrix ℓ-sparse iff each row contains at most ℓ non-zero entries.

Let $J_i := \{j \in [n] \mid A_{ij} = 1\}$ (hence $|J_i| \leq \ell$ if A is ℓ-sparse). The linear map from ${\mathbf{F}_2}^n$ into ${\mathbf{F}_2}^m$ defined by A is computed as:

$$(A \cdot x)_i = \sum_j A_{ij} x_j = \bigoplus_{j \in J_i} x_j \ .$$

Assume $m > n$. Hence $Rng(A)$ is a proper subset of ${\mathbf{F}_2}^m$. Let $b \in {\mathbf{F}_2}^m$ be any vector outside of the range of A. In other words, the linear system:

$$A \cdot x \ = \ b$$

has no solution (in $\mathbf{F}_2$). This unsolvability can be expressed as a tautology $\tau_b(A)$ in a DNF as follows (we skip A from the notation of τ_b as we always consider only one matrix at a time):

$$\tau_b \ := \ \bigvee_{i \in [m]} \ \bigvee_{\epsilon \in \{0,1\}^{|J_i|}, \bigoplus_{j \in J_i} \epsilon_j = 1 - b_i} \ \bigwedge_{j \in J_i} x_j^{\epsilon_j}$$

Here we use the notation x^ϵ from section 2.7. The formula says that there is some bit i such that the ith bits of $A \cdot x$ and b differ, which itself is expressed by saying that there is an evaluation ϵ to bits x_j of x that belong to J_i which determines the ith bit of $A \cdot x$ as $1 - b_i$, i.e. as different from b_i.

Note that the size of the formula is bounded above by $O(m 2^\ell \ell)$, and that the clauses of $\neg \tau_b$ have the width $\leq \ell$.

For the next definitions and statements let us fix parameters $1 \leq n < m$ and $\ell \leq m$, and an ℓ-sparse $m \times n$ matrix A. The next definition is a special case of a definition [1, Def.2.1].

Definition 2.8.1 *A boundary of a set of rows $I \subseteq [m]$, denoted $\partial_A(I)$, is the set of $j \in [n]$ such that exactly one entry A_{ij} equals 1 for $i \in I$.*

Let $1 \leq r \leq m$ and $\epsilon > 0$ be any parameters. Matrix A is an (r, ϵ)-expander iff for all $I \subseteq [m]$, $|I| \leq r$, $|\partial_A(I)| \geq \epsilon \ell |I|$.

Expanders simulate, in a sense, matrices with disjoint J_i's and of the maximal size ℓ. In such a case it would hold that $|\partial_A(I)| = \ell |I|$. An (r, ϵ)-expander achieves (as long as $|I| \leq r$) at least an ϵ-percentage of this maximal value.

We do not have any explicit matrix A that has suitable expansion properties but the existence of such a matrix can be proved by a probabilistic argument.

Consider the following random process yielding an ℓ-sparse matrix A. For every $i \in [m]$ and $u \leq \ell$ let $\mathbf{j}_{i,u}$ be chosen independently and uniformly at random from $[n]$. Let $J_i \subseteq [n]$ be the set of these values for fixed i, and define $A_{i,j} = 1$ iff $j \in J_i$.

The following theorem is a special case of [1, Thm.5.1].

Theorem 2.8.2 *For every $\delta > 0$ there is an $\ell \geq 1$ such that for all sufficiently large n the random ℓ-sparse $n^2 \times n$-matrix is an $(n^{1-\delta}, 3/4)$-expander with probability approaching* 1.

Proof :

Let $r \leq n^2$ and $\ell \geq 1$ be yet unspecified but fixed parameters; we shall specify the values later. Let A be an ℓ-sparse $n^2 \times n$-matrix constructed in the random process described above (so $m = n^2$). We want to show that

$$\mathbf{Prob}[A \text{ is not an } (r, 3/4)\text{-expander}] \longrightarrow 0 \ .$$

For $t \leq r$ let p_t be the probability that any one fixed set I of t rows in A falsifies the expansion property. Then

$$\mathbf{Prob}[A \text{ is not an } (r, 3/4)\text{-expander}] < \sum_{t=1}^{r} \binom{n^2}{t} p_t \leq \sum_{t=1}^{r} n^{2t} p_t \ .$$

Fix one such I, $|I| = t$. Then:

$$|\bigcup_{i \in I} J_i| \leq |\partial_A(I)| + 1/2[(\sum_{i \in I} |J_i|) - |\partial_A(I)|]$$

as any $j \in \bigcup_{i \in I} J_i \setminus \partial_A(I)$ belongs to at least two rows in I. The right hand side is bounded above by $1/2(|\partial_A(I)| + t\ell)$ and hence if it were $|\partial_A(I)| < (3/4)\ell t$ then also

$$|\bigcup_{i \in I} J_i| < (7/8)\ell t \ .$$

Consequently,

$$p_t \leq \mathbf{Prob}[|\bigcup_{i \in I} J_i| < (7/8)\ell t] \ .$$

The right hand side is simply the probability that in picking $t\ell$ elements of $[n]$ independently of each other we select less than $(7/8)t\ell$ elements. If this happens then there must be a set of $(1/8)t\ell$ steps among the $t\ell$ steps when we pick a point already selected; the later event has a probability bounded above by $\frac{t\ell}{n}$. Hence:

$$p_t \leq \mathbf{Prob}[|\bigcup_{i \in I} J_i| < (7/8)\ell t] \leq \binom{t\ell}{(1/8)t\ell}[\frac{t\ell}{n}]^{\frac{t\ell}{8}} \leq [O(\frac{\ell r}{n})^{\ell/8}]^t$$

Putting all these inequalities together we see that the probability that A is not an $(r, 3/4)$-expander is bounded above by a finite geometric sum

$$\sum_{t=1}^{r} n^{2t}[O(\frac{\ell r}{n})^{\ell/8}]^t = \sum_{t=1}^{r}[n^2 O(\frac{\ell r}{n})^{\ell/8}]^t$$

Substituting $n^{1-\delta}$ for r and taking $\ell \geq 1$ large enough constant (so that $\delta\ell/8 > 2$) the base of the progression $[n^2 O(\frac{\ell r}{n})^{\ell/8}]$ becomes bounded above by $n^{-\Omega(1)}$. Hence the sum approaches 0 as $n >> 0$.

q.e.d.

For the next few definitions and lemmas assume that A is an ℓ-sparse $m \times n$-matrix that is an $(r, \frac{3}{4})$-expander. For a set of rows $I \subseteq [m]$ let $J(I) := \bigcup_{i \in I} J_i$, and let A_I be the $(m - |I|) \times (n - |J(I)|)$-matrix obtained from A by deleting all rows in I and all columns in $J(I)$.

The next lemma slightly generalizes [1, L.4.6].

Lemma 2.8.3 *For any set of rows $I \subseteq [m]$ of size $|I| \leq r/2$ there is $\hat{I} \supseteq I$, $|\hat{I}| \leq 2|I|$, such that*

$(*)$ *For any $i \notin \hat{I}$, $|S_i \setminus \bigcup_{u \in \hat{I}} S_u| \geq \ell/2$.*

Moreover, for any $\hat{I}$ of size $|\hat{I}| \leq r$ having this property $(*)$, $A_{\hat{I}}$ is an $(r, \frac{1}{4})$-expander. Furthermore, there exists the smallest (w.r.t inclusion) such an $\hat{I}$.

Proof :

Assume $|I| \leq r/2$. Put $I_0 := I$. Add in consecutive steps $t = 0, \ldots$ to I_t any one row i as long as

$$(*) \qquad |J_i \cap \bigcup_{k \in I_t} J_k| > \ell/2 \ .$$

We claim that this process stops before t reaches $|I|$. Assume not, and let $I' = I_{|I|}$. Then, by $(*)$, it holds

$$\partial_A(I') < \ell|I| + (\ell/2)|I| = (3/4)\ell|I'|$$

This contradicts the expansion property of A, as $|I'| \leq r$.

Let $\hat{I}$ be the last I_t in the process, so $t < r/2$ and $|\hat{I}| \leq 2|I|$.

$\hat{I}$ clearly has property $(*)$. Thus we only need to verify the expansion property of $A_{\hat{I}}$. Let K be a set of $\leq r$ rows in $A_{\hat{I}}$. Then

$$\partial_{A_{\hat{I}}}(K) = \partial_A(K) \setminus \bigcup_{i \in \hat{I}} J_i(A)$$

As for all $i \in K \setminus \hat{I}$ we have $|J_i(A) \cap \bigcup_{k \in \hat{I}} J_k(A)| \leq \ell/2$, this equality implies that

$$|\partial_{A_{\hat{I}}}(K)| \geq |\partial_A(K)| - (\ell/2)|K| \geq (3/4)\ell|K| - (\ell/2)|K| \geq (1/4)|K| \ .$$

q.e.d.

The next definition and lemma are from [29].

Definition 2.8.4 1. *Any I satisfying the condition $(*)$ from Lemma 2.8.3 is called a safe set of rows.*

2. *A partial assignment $\rho :\subseteq \{x_1, \ldots, x_n\} \rightarrow \{0,1\}$ is called safe iff $dom(\rho) = \bigcup_{i \in I} J_i$, for some safe I.*

   We pick any such I and call it the support of ρ, denoted $supp(\rho)$.

3. *Let $b \in \{0,1\}^m$. A safe partial assignment ρ with support I is a safe partial solution of $A \cdot x = b$ iff for all $J_i \subseteq \bigcup_{u \in I} J_u$, $\bigoplus_{j \in J_i} \rho(x_j) = b_i$.*

4. *For ρ a safe partial solution with support I, b^ρ is an $(m - |I|)$-vector with the ith coordinate being $b_i \oplus \bigoplus_{j \in J_i \cap dom(\rho)} \rho(x_j)$, for i such that $J_i \not\subseteq dom(\rho)$.*

   Vector x_I consists of those variables not in $J(I)$.

Note that if ρ is a safe solution with support I, and ξ is a solution of $A_I \cdot x_I = b^\rho$, then $\rho \cup \xi$ is a solution of $A \cdot x = b$.

Lemma 2.8.5 *Let $I \subseteq I' \subseteq [m]$ be two safe systems, with $|I' \setminus I| \leq r$. Assume that ρ is a safe assignment with support I. Let $c_i \in \{0,1\}$, $i \in I' \setminus I$, be arbitrary.*

Then ρ can be extended to a safe assignment ρ' with support I' such that $\bigoplus_{j \in J_i} \rho'(x_j) = c_i$, for all $i \in I' \setminus I$.

Proof :

By Lemma 2.8.3, A_I is an $(r, \frac{1}{4})$-expander. By the expansion property, every subset of $I' \setminus I$ has a non-empty border in A_I and hence, in particular, cannot constitute a linearly dependent set of rows of A_I. Thus the linear system

$$\bigoplus_{j \in J_i \setminus dom(\rho)} x_j = c_i \oplus \bigoplus_{j \in J_i \cap dom(\rho)} \rho(x_j)$$

has a solution ξ. Put $\rho' := \rho \cup \xi$.

q.e.d.

Theorem 2.8.6 (Krajíček[29]) *Assume that A is an ℓ-sparse $m \times n$ matrix that is an $(r, 3/4)$-expander. Let $b \notin Rng(A)$.*

Then every R-proof of $\tau_b(A)$ must have the width at least $\geq r/4$.

Proof :

Let π be a resolution refutation of $A \cdot x = b$, i.e. a proof of $\tau_b(A)$. Let w denote the width of π.

We shall construct a sequence of clauses $C_0, \ldots, C_e$ occurring in π and a sequence of partial safe assignments $\alpha_t :\subseteq \{x_1, \ldots, x_n\} \to \{0,1\}$ for $t = 0, \ldots, e$, such that the following conditions are satisfied:

1. $C_0 := \emptyset$ is the end clause of π. Each C_{t+1} is a hypothesis of an inference in π yielding C_t, and C_e is an initial clause.

2. If x_j occurs in C_t then $x_j \in dom(\alpha_t)$.

3. C_t is false under the assignment α_t.

4. $|supp(\alpha_t)| \leq 2w$.

Put $\alpha_0 := \emptyset$. Assume we have C_t and α_t, and that C_t has been inferred in π by:

$$\frac{D_1 \cup \{x_j\} \qquad D_2 \cup \{\neg x_j\}}{C_t (= D_1 \cup D_2)}$$

Let $I' \supseteq supp(\alpha_t)$ be a minimal safe set with some row containing j. It exists, by Lemma 2.8.3, as long as $|supp(\alpha_t)| + 1 \leq r/2$; as $|supp(\alpha_t)| \leq 2w$ this inequality follows if $w < r/4$.

By Lemma 2.8.5 there is a partial safe solution $\rho' \supseteq \alpha_t$. Take for $\alpha_{t+1} \subseteq \rho'$ a minimal safe assignment obeying conditions 2. Finally, take for C_{t+1} the clause among $D_1 \cup \{x_j\}$, $D_2 \cup \{\neg x_j\}$ made false by α_{t+1}.

Now note that conditions on C_e and α_e lead to a contradiction. C_e is an initial clause and so α_e makes true its negation which is one of the conjunctions $\bigwedge_{j \in J_i} x_j^{\epsilon_j}$ in τ_b. In particular, $\oplus_{j \in J_i} \epsilon_j = 1 - b_i$. But that violates the assumption that α_e satisfies all equations of $A \cdot x = b$ evaluated by α_e.

We have constructed the sequence under the assumption that $w < r/4$. Hence $w \geq r/4$.

q.e.d.

Corollary 2.8.7 ([29]) *Assume that A is an ℓ-sparse $m \times n$ matrix that is an $(r, 3/4)$-expander. Let $b \notin Rng(A)$. Then every R-proof of $\tau_b(A)$ must have the size at least $\geq 2^{\Omega(\frac{(r/4-\ell)^2}{n})}$.*

In particular, for every $\delta > 0$ there is an $\ell \geq 1$ such that for all sufficiently large n there exists an ℓ-sparse $n^2 \times n$-matrix A such that $\tau_b(A)$ requires R-proofs of size at least $\geq 2^{\Omega(n^{1-\delta})}$.

Proof :

Apply Theorem 2.8.2 for $\delta/2$, to get $\ell \geq 1$ and an ℓ-sparse $n^2 \times n$-matrix A which is an $(n^{1-\delta/2}, 3/4)$-expander. By Theorem 2.8.6 every R-proof of $\tau_b(A)$ must have the width at least $\Omega(n^{1-\delta/2})$.

The width-size relation given in Theorem 2.7.5 it follows that the size of any such proof must be at least $\exp(\Omega(\frac{(n^{1-\delta/2}-\ell)^2}{n}))$, as ℓ bounds the width of the initial clauses. This is $2^{\Omega(n^{1-\delta})}$.

q.e.d.

2.9 Exercises

Exercise 2.9.1 Limited extension *is a way how to translate formulas into DNF formulas and preserving (un)satisfiability. It is analogous to the reduction of the general satisfiability problem to the satisfiability of sets of clauses. Let A be any formula built from atoms $p_1, \ldots, p_n$. Introduce for each subformula B of A (including A itself) a new atom q_B. Let $Ext(A)$ of all clauses of the form:*

1. $\{q_B, \neg p_i\}$, $\{\neg q_B, p_i\}$, *if B is atom p_i.*
2. $\{q_B, q_C\}$, $\{\neg q_B, \neg q_C\}$ *if $B = \neg C$.*
3. $\{\neg q_B, q_{C_1}, q_{C_2}\}$, $\{q_B, \neg q_{C_1}\}, \{q_B, \neg q_{C_2}\}$ *if $B = C_1 \vee C_2$.*
4. $\{\neg q_B, q_{C_1}\}$, $\{\neg q_B,\ q_{C_2}\}$, $\{q_B, \neg q_{C_1}, \neg q_{C_2}\}$ *if $B = C_1 \wedge C_2$*

Compute the total length of all formulas in $Ext(A)$ and prove that $Ext(A) \cup \{q_A\}$ is satisfiable if and only if A is satisfiable.

Exercise 2.9.2 *Analyze the argument in Theorem 2.1.1, and give an upper bound on the number of clauses in a resolution refutation of any unsatisfiable set of k clauses formed from literals build from n atoms.*

Exercise 2.9.3 *Let an $\exists$-decision tree be a decision tree branching according to the truth value of a clause. Transform the proof of Theorem 2.2.4 into a construction of an $\exists$-decision tree (from π) and a lower bound to the height of such trees solving the search problem associated with PHP_n.*

Exercise 2.9.4 *Show that Lemma 2.2.1 can be reversed: Turning a decision tree upside down gives,* essentially, *an R^*-refutation of $\mathcal{C}$.*

Show that a general, non-tree-like, R-refutation of $\mathcal{C}$ yields a branching program solving the search problem, but not vice versa.

Prove the following theorem, showing that even in the case of non-tree-like proofs we can get, under special conditions, a correspondence between branching programs and R-proofs.

Theorem 2.9.5 ([34]) *The minimal number of clauses in a* regular *resolution refutation of $\mathcal{C}$, (where "regular" means that on every path through the refutation every atom is resolved at most once) equals to the minimal number of nodes in a* read-once *branching program solving the search problem associated with $\mathcal{C}$ (where "read-once " means that on every path through the branching program every atom occurs at most once as a label of an node).*

A proof can be found in [25, Chpt.4] but let me give a sketch. For the hard direction (from a program to a proof) associate with every node v in the read-once program a clause C_v having the property that every assignment determining a path going through v falsifies C_v. If v is a leaf then C_v is the clause from $\mathcal{C}$ labelling v in the program. Assume that the node v is labelled by atom p_i and the edge (v, v_1) is labelled by 1, and (v, v_0) by 0.

We claim that C_{v_1} does not contain p_i and C_{v_0} does not contain $\neg p_i$. This is because σ is read-once and so no path reaching v (and at least one path does reach v as s is minimal possible) determines the value of p_i. Hence we could prolong such path by giving to p_i value 1 if $p_i \in C_{v_1}$ or value 0 if $\neg p_i \in C_{v_0}$. This new path would satisfy C_{v_0} or C_{v_1} respectively, contradicting the assumption above.

It follows that either one of the clauses C_{v_1}, C_{v_0} contains none of $p_i, \neg p_i$, or that C_{v_0} contains p_i but not $\neg p_i$ and C_{v_1} contains $\neg p_i$ but nor p_i. In the former case define C_v to be the clause containing none of $p_i, \neg p_i$, and in the latter case define C_v to be the resolution of clauses C_{v_1} and C_{v_0} w.r.t. atom p_i.

It is easy to verify (using an argument similar to the one above) that no path through v satisfies C_v.

The root of σ has to be assigned the empty clause as all paths go through it. Hence the constructed object is a regular resolution refutation.

Exercise 2.9.6 *Prove the Craig interpolation theorem for propositional logic, as well as its monotone version (Lyndon theorem).*

Exercise 2.9.7 *Given a circuit C of size s formalize the statement that C* has a unique computation on an input $\overline{p}$. *The formalization is a family of implications (one for each output bit). Show that each of these implications has a resolution proof of size $O(s)$.*

Exercise 2.9.8 *Prove Theorem 2.4.1.*

Exercise 2.9.9 *Show that in order for Theorem 2.4.3 to hold we cannot replace the consistency condition in Definition 2.4.2 by a simpler one: For all u, v the label of the leaf in $P^{\emptyset}_{u,v}$ is valid for u, v.*

Exercise 2.9.10 *Define a* linear equational calculus *(LEC) to be a proof system working with linear equations*

$$a_1 x_1 + \ldots + a_n x_n = b$$

over a finite field. The rules allow to add two equations and to multiply an equation by an element of the field. An LEC-refutation of equations $E_1, \ldots, E_m$ *is an LEC-derivation of the equation* $0 = 1$ *from* $E_1, \ldots, E_m$. *Let the "size" of an equation be the number of non-zero coefficients. LEC is sound and complete (by Gauss elimination), if by completeness we mean that every system of equations unsolvable in* F *is refutable. When completeness is considered only w.r.t. the systems with no* 0-1 *solution then LEC is complete only for the two-element field* $\mathbf{F}_2$. *But not even all Boolean functions can be represented by a conjunction of linear equations and so LEC cannot be considered, even for* $\mathbf{F}_2$, *as a full propositional proof system in the sense of [14].*

Prove all these facts and prove the effective interpolation for LEC.

Exercise 2.9.11 *Prove a bound to the interpolantion for* CP. *Express the bound in terms of* n *and* M, *a bound to the absolute values of coefficients occuring in a derivation.*

Chapter 3

Frege systems and stronger systems

In this chapter we depart from resolution towards particular stronger systems (general systems will be studied in Chapter **??**). The most important among them are Frege systems F and Extended Frege systems EF. We shall also discuss in this chapter the Substitution Frege systems SF and the Quantified propositional calculus G.

3.1 Frege systems

The notion of a *Frege system* formalizes the usual calculus for propositional logic everybody learns at school. It has a language complete for propositional logic and is based on finitely many axiom schemes (like $A \vee \neg A$) and inference rules (like modus ponens $A,\ A \rightarrow B\ /\ B$).

Definition 3.1.1 (Cook-Reckhow[14]) *Let L be any fixed finite language complete for propositional logic (that is, all boolean functions can be defined in L).*

A Frege rule *(tacitly in L) is a $k+1$-tuple of formulas $A_0, \ldots, A_k$ in atoms $p_1, \ldots, p_n$ written as:*

$$\frac{A_0, \ldots, A_{k-1}}{A_k}\ ,$$

such that any truth assignment $\alpha : \{p_1, \ldots, p_n\} \rightarrow \{0, 1\}$ satisfying all formulas $A_0, \ldots, A_{k-1}$ satisfies also A_k.

A Frege rule in which $k = 0$ is called a Frege axiom scheme.

An instance of the rule is obtained by a simultaneous substitution of arbitrary formulas B_i for all p_i.

The condition posed on a Frege rule in the definition is the soudness of the rule.

Definition 3.1.2 (Cook-Reckhow[14]) *Let F be a finite collection of Frege rules.*

1. *A* Frege proof *(an F-proof briefly) of formula ξ from formulas $\eta_1, \ldots, \eta_u$ is a finite sequence $\theta_1, \ldots, \theta_k$ of formulas such that $\theta_k = \xi$, and such that every θ_i is either one of $\eta_1, \ldots, \eta_u$, or is inferred from some earlier θ_j's ($j < i$) by a rule of F.*

2. *F is implicationally complete if and only if any ξ can be F-proved from any set $\{\eta_1, \ldots, \eta_u\}$ if every truth assignment satisfying all η_i's satisfies also ξ (i.e. ξ is a semantical consequence of η_i's).*

3. *F is a* Frege proof system *if and only if it is implicationally complete.*

One of the main features of Frege system is the robustness of the definition. We may vary the language, the proof format (tree-like or sequence-like), and even pass to natural deduction or sequent calculus formalizations, and we always get a polynomially-equivalent (in the sense of polynomial simulation) proof system. I shall not discuss the p-equivalence to sequent calculus or natural deduction as we do not discuss the formalizations much in this chapter (see Exercise 3.5.1 or [14]). The first two statements made above are the content of the following two theorems.

In the next theorem we shall confine ourselves to Frege systems whose language contains the DeMorgan language. The reason is that we have defined in Definition 1.0.1 proof systems using the set $TAUT$ of DeMorgan tautologies. If the language of a system does not contain the DeMorgan language we would have to specify a particular translation of DeMorgan tautologies into the language and this is just obscures things.

Theorem 3.1.3 (Reckhow [41]) *Assume that F and F' are two Frege systems and that the languages of both contain the DeMorgan language.*

Then F and F' polynomially simulates each other.

Moreover, the p-simulations can be choosen so that both the number of steps and the size of proofs increase at most proportionally and the depth increases by a constant.

The only full published proof of the theorem I am aware of is in [25, Chpt.4]. I shall not repeat the proof here but I shall outline the main difficulty and the main idea how to overcome it.

Obviously it is enough to prove that any Frege system F in the DeMorgan language p-simulates any Frege system F' in a language L containing the DeMorgan language. If L were in fact just the DeMorgan langauge then the p-simulation could be done easily. In any F'-rule $\eta_0, \dots, \eta_k$ the formula η_k semantically follows from $\eta_0, \dots, \eta_{k-1}$. By the implicational completeness of F there is an F-proof π of η_k from $\eta_0, \dots, \eta_{k-1}$. Thus whenever we would see an application of the rule in an F'-proof we could simulate it in F by (an instance of) proof π. It is easy to computate that this increases the size as well as the number of steps only proportionally.

When the langueg of F' is bigger than the DeMorgan language the natural approach would be to first represent all connectives in L by DeMorgan formulas and then proceed as before. However, a difficulty may arise. Assume that L contains the equivalence connective $\equiv$. In the DeMorgan language we may define $p \equiv q$ by $(p \wedge q) \vee (\neg p \wedge \neg q)$. If we translate inthis way the formula

$$p_1 \equiv (p_2 \equiv (p_3 \equiv \dots (p_{n-1} \equiv p_n) \dots)$$

we obtain a formula of size $\Omega(2^n)$.

The way how to overcome this difficulty is the following. Note that if the nesting of $\equiv$'s in a formula is k then the translation will have size $O(2^k)$. Hence if we manage first to modify the original F'-proof (that we attempt to p-simulate) so that every formula in it has only logarithmic depth then the translation will work. In fact, this can be done. See [25, Lemma 4.4.14] for a detailed proof.

Definition 3.1.4 *A Frege proof $\theta_1, \dots, \theta_k$ is* tree-like *if and only if every step θ_i is a hypothesis of at most one inference in the proof.*

Frege proof system F using only tree-like proofs is denoted F^*.

Theorem 3.1.5 (Krajíček[24]) *F^* p-simulates F. In fact, any F-proof of size m, with k steps, of depth d can be transformed into a tree-like proof of the same formula that has size $O(mk \log(k))$, $O(k \log(k))$ steps and the depth $d + O(1)$.*

Proof :

Let $\theta_1, \ldots, \theta_k$ be an F-proof of τ. Derive consecutively (in a tree-like fashion) formulas $\phi_i := \theta_1 \wedge \ldots \wedge \theta_i$ (brackets balancing the conjunction into a binary tree of depth at most $O(log(i))$).

We claim that ϕ_{i+1} has a tree-like proof from ϕ_i with a $O(i \cdot log(i))$ number of steps, size $O(i \cdot |\phi_i|)$ and of the depth $d + O(1)$. Obviously, the following is sufficient:

Claim For $j \leq i$, *any* θ_j *can be proved from* ϕ_i *by a tree-like proof with* $O(i \cdot log(i))$ *steps, size* $O(log(i) \cdot |\phi_i|)$ *and depth* $dp(\phi_i) + O(1)$.

q.e.d.

No strong lower bounds are known for Frege systems. The following is the best one.

Theorem 3.1.6 (Krajíček[23]) *Any* F*-proof of* $\neg\neg \ldots 1$*, the negation occuring* $2n$*-times, must have the size at least* $\Omega(n^2)$ *(the constant implicit in* Ω *depends on the particular system* F*).*

This theorem is a simple corollary of a general statement about the structure of proofs, even in predicate logic, from [22]. We state it only for Frege systems.

Theorem 3.1.7 (Krajíček[22]) *For every Frege system* F *there is a constant* $c > 0$ *such that the following holds.*

If A *has an* F*-proof* $\pi = B_1, \ldots, B_k$ *with* k *steps there is another* F*-proof* $C_1, \ldots, C_k$ *such that:*

1. *The logical depth of formulas* C_i *is bounded by* $c \cdot k$*, all* $i \leq k$*.*

2. *There is a substitution* σ *of formulas for atoms in* C_i*'s such that:*

$$\sigma(C_i) = B_i$$

all $i \leq k$*.*

3.2 Substitution Frege systems

Instance of Frege rules are obtained by substitutions but the substitution itself is not a valid inference rule in Frege systems. Substitution Frege systems extend Frege systems by allowing the rule.

Definition 3.2.1 *The* substitution rule *allows to substitute simultaneously formulas for atoms:*

$$\frac{A(p_1, \ldots, p_n)}{A(B_1, \ldots, B_n)}.$$

A Frege system F augmented by the substitution rule is denoted SF.

We can eliminate an application of the substitution rule by repeating the part of the proof before the inference, with B_i's substituted everywhere for p_is. In such a transformation these repetitions can be nested and the proof may grow exponentially.

In fact, this exponential increase in the number of steps is necessary. This fact is due to Tseitin-Cubarjan [46]. A simpler example than their original one is provided by the following statement.

Lemma 3.2.2 ([23]) *Let F and SF be a frege and a Substitution Frege systems respectively.*

The formula $\neg^{(2^n)}(1)$ has an SF-proof with $O(n)$ steps but every F-proof requires $\Omega(2^n)$ steps.

Proof :

Define $A_n := \neg^{(2^n)}(1)$ with $\neg^{(k)}$ denoting k occurrences of $\neg$. Let $B_k = p \to (\neg)^{2^k}(p)$.

SF-derives B_k from B_{k-1} ina constant number of steps utilizing the substitution rule: Substitute $(\neg)^{2^{k-1}}(p)$ for p in B_{k-1} and apply modus ponens. B_0 has a constant size proof, so every B_k has an SF-proof with $O(k)$ steps.

For the second part of the statement assume that A_n has an F-proof with k steps. By Theorem 3.1.7 there is an F-proof of some formula B such that, in particular, the logical depth of B is $O(k)$ and A_n is a substitution instance of B.

As B is a tautology, necessarily $B = A_n$. Hence $\Omega(2^n) \leq k$.

q.e.d.

Note that this statement does not exponentially separate F from SF; the point is that the speed-up it achieved on a formula that has itself exponential size. In fact, no lower bounds at all are known for SF.

3.3 Extended Frege systems

There is another way how to augment Frege systems to apparently stronger proof systems. The idea is to allow the proof system to abbreviate (possibly large) formulas by new atoms.

Definition 3.3.1 (Cook-Reckhow[14]) *Let F be a Frege system. An* extended Frege proof *is a sequence of formulas $A_1, \ldots, A_k$ such that every A_i is either obtained from some previous A_j's by an F- rule or has the form:*

$$q \equiv B$$

with the following conditions satisfied:

1. *Atom q does appear neither in B, nor in any A_j for $j < i$.*
2. *Atom q does not appear in A_k.*

(If $\equiv$ is not in the language of F we use any fixed formula defining it.) A formula of this form is called an extension axiom, *q is called an* extension atom.

An extended Frege system *EF is the proof system whose proofs are extended Frege proofs.*

The possibility to introduce extension axiom in an extended Frege proof is sometimes called the "Extension rule" although it is not a rule in the earlier sense.

Similarly as with the Substitution rule we can eliminate the extension rule by consecutively replacing all extension atoms by their defining formulas. However, extension atoms may occur in defining formulas of other extension atoms (introduced later in the proof) and this nesting can cause an exponential increase in size in this transformation. But if we have the substitution rule this works well.

Lemma 3.3.2 *A Substitution Frege system SF polynomially simulates any Extended Frege system EF.*

Proof :

Let $q_1 \equiv B_1, \ldots, q_r \equiv B_r$ be the extension axioms introduced in an EF-proof in this order. In fact, we may clearly assume that these r extension axioms form the first r steps of the proof.

Now transform the original proof with steps A's into a new proof with steps

$$q_r \equiv B_r \to (q_{r-1} \equiv B_{r-1} \to (\dots (q_1 \equiv B_1) \dots) \to A \ .$$

This transformation uses only Frege rules.

Next apply to the last formula of this form the substitution rule by substituting B_r for q_r, then B_{r-1} for q_{r-1} etc.. This eliminates (applying modus ponens with formulas of the form $C \equiv C$) the part

$$q_r \equiv B_r \to (q_{r-1} \equiv B_{r-1} \to (\dots (q_1 \equiv B_1) \dots) \to \dots$$

It is easy to compute that the size of the original proof increases at most quadratically in this process.

q.e.d.

Considerably more difficult is the opposite simulation. We shall give its proof in Chapter **??** using bounded arithmetic. A direct combinatorial p-simulation can be found in [31] or in [25, Sec.4.5].

Theorem 3.3.3 ([19, 31]) *Any extended Frege system EF polynomially simulates any Substitution Frege system SF.*

The following four facts summarize further elementary but important properties of Extended frege systems (see Exercises 3.5):

1. Extended Frege systems satisfy the analogue of Reckhow's Theorem 3.1.3.

2. There is no difference in measuring the complexity of EF-proofs by the size or by the number of steps: Any formula A having an EF-proof with k steps has also an EF-proof of size $O(k + |A|)$.

3. The minimal numbers of steps in a F-proof and in an EF-proof of a formula are proportional to each other.

4. Allowing the extension rule (see Exercise 3.5.4 for a precise formulation) in resolution creates a proof system p-equivalent to EF.

5. EF is p-equivalent to "Frege systems operating with circuits".

I left to the end of the section the sad issue of lower bouds for EF: No lower bounds, even super-linear, are known. Cook and Reckhow [14] originally suggested that the pigeonhole principle PHP_n (see Chapter 1) may separate EF from F. However, this is not true, the principle has polynomial size proofs in both EF and F. The upper bound in EF is simple and the proof just formalizes a straightforward proof by induction on n. The upper bound in F is much harder and requires to show that Frege system "can count". We give here only the proof of the first upper bound; the proof of the second will be given in Chapter **??** via bounded arithmetic.

Theorem 3.3.4 (Cook-Reckhow[14]) *The pigeonhole principle PHP_n has an EF-proof of size polynomial in n.*

Proof :
Let p_{ij} be the atoms of PHP_n; $i \in [n]$ and $j \in [n-1]$. Define, using the extension rule, new atoms q_{uv}, for $u \in [n-1]$ and $v \in [n-2]$ by:

$$q_{uv} \;:=\; p_{uv} \vee [p_{nv} \wedge p_{u(n-1)}]$$

It is easy to see that there is a size $n^{O(1)}$ EF-derivation of $\neg PHP_{n-1}$ expressed in atoms q_{uv} from $\neg PHP_n$ (expressed in atoms p_{ij}).

Iterating this proces deduces $\neg PHP_2$ from $\neg PHP_n$ by p-size EF-proof. But $\neg PHP_2$ has a refutation (of a constant size).

q.e.d.

Theorem 3.3.5 (Buss[10]) *The pigeonhole principle PHP_n has an F-proof of size polynomial in n.*

The issue of which tautologies form plausible candidates as being hard for EF will be discussed in Chapter **??** (see also [25, 26, 29]).

3.4 Quantified propositional calculus

It is most convenient to define the quantified propositional logic G over Gentzen's sequent calculus LK (we consider only its propositional fragment here).

The lines in a sequent calculus proof are not formulas but *sequents*, an ordered pair of two finite (possibly empty) sequences of formulas written as:

$$A_1, \ldots, A_u \longrightarrow B_1, \ldots, B_v \;.$$

Formulas $A_1, \dots, A_u$ form the *antecedent* and formulas $B_1, \dots, B_v$ form the *succedent* of the sequent. Letters $\Gamma, \Delta, \Pi, \dots$ will denote finite sequences of formulas, called also *cedents.*

The truth definition is extended from formulas to sequents as follows: A truth assignment α to the atoms in a sequent $\Gamma \longrightarrow \Delta$ satisfies the sequent if and only if α either satisfies a formula from the succedent Δ or it satisfies the negation of a formula from the antecedent Γ.

Note that, in particular, the empty sequent $\emptyset \longrightarrow \emptyset$ (written also simply $\longrightarrow$) cannot be satisfied. The empty sequent plays in LK the role of the empty clause in R.

Definition 3.4.1 *An LK-proof is a sequence of sequents in which every sequent is either an* initial sequent, *a sequent having one of the forms:*

$$p \longrightarrow p, \quad 0 \longrightarrow, \quad \longrightarrow 1$$

with p an atom, or is derived from previous sequents in the proof by one of the following rules:

1. weakening rules
$$\textbf{left}\ \frac{\Gamma \longrightarrow \Delta}{A, \Gamma \longrightarrow \Delta} \quad \textit{and} \quad \textbf{right}\ \frac{\Gamma \longrightarrow \Delta}{\Gamma \longrightarrow \Delta, A}$$

2. exchange rules
$$\textbf{left}\ \frac{\Gamma_1, A, B, \Gamma_2 \longrightarrow \Delta}{\Gamma_1, B, A, \Gamma_2 \longrightarrow \Delta} \quad \textit{and} \quad \textbf{right}\ \frac{\Gamma \longrightarrow \Delta_1, A, B, \Delta_2}{\Gamma \longrightarrow \Delta_1, B, A, \Delta_2}$$

3. contraction rules
$$\textbf{left}\ \frac{\Gamma_1, A, A, \Gamma_2 \longrightarrow \Delta}{\Gamma_1, A, \Gamma_2 \longrightarrow \Delta} \quad \textit{and} \quad \textbf{right}\ \frac{\Gamma \longrightarrow \Delta_1, A, A, \Delta_2}{\Gamma \longrightarrow \Delta_1, A, \Delta_2}$$

4. $\neg$: introduction rules
$$\textbf{left}\ \frac{\Gamma \longrightarrow \Delta, A}{\neg A, \Gamma \longrightarrow \Delta} \quad \textit{and} \quad \textbf{right}\ \frac{A, \Gamma \longrightarrow \Delta}{\Gamma \longrightarrow \Delta, \neg A}$$

5. $\wedge$: introduction rules
$$\textbf{left}\ \frac{A, \Gamma \longrightarrow \Delta}{A \wedge B, \Gamma \longrightarrow \Delta} \quad \textit{and} \quad \frac{A, \Gamma \longrightarrow \Delta}{B \wedge A, \Gamma \longrightarrow \Delta}$$
$$\textit{and}\ \textbf{right}\ \frac{\Gamma \longrightarrow \Delta, A \quad \Gamma \longrightarrow \Delta, B}{\Gamma \longrightarrow \Delta, A \wedge B}$$

6. $\vee$: introduction rules

$$\textbf{left}\ \frac{A, \Gamma \longrightarrow \Delta \qquad B, \Gamma \longrightarrow \Delta}{A \vee B, \Gamma \longrightarrow \Delta} \quad \textit{and}$$

$$\textbf{right}\ \frac{\Gamma \longrightarrow \Delta, A}{\Gamma \longrightarrow \Delta, A \vee B} \quad \textit{and} \quad \frac{\Gamma \longrightarrow \Delta, A}{\Gamma \longrightarrow \Delta, B \vee A}$$

7. cut rule

$$\frac{\Gamma \longrightarrow \Delta, A \qquad A, \Gamma \longrightarrow \Delta}{\Gamma \longrightarrow \Delta}$$

Every rule except the cut rule introduces a new formula; such a formula is called the principal formula *of the rule and theformulas from which it is inferred are called the* minor formulas *of the rule. All other formulas in the rule are called the* side formulas.

For a formula in Δ or Γ in the lower sequent of a rule, the same occurrence in the upper sequent(s) is called the immediate ancestor *of the formula. The immediate ancestor(s) of a principal formula of a rule are the minor formulas of the rule.*

An ancestor *of a formula is any formula obtained by repeating the immediate ancestor step.*

The following is well-known (and left as an Exercise 3.5.7).

Theorem 3.4.2 *The system LK is sound and complete. That is, all provable sequents are satisfied by all truth assignments and whenever a sequent $\Gamma \longrightarrow \Delta$ is satisfied by all truth assignments then it has an LK-proof. Moreover, this proof does not need to use the cut-rule.*

Quantified propositional calculus is formed from the sequent calculus LK by introduction of propositional quantifiers: $\forall x A(\overline{p}, x)$ (meaning $A(\overline{p}, 0) \wedge A(\overline{p}, 1)$, and $\exists x A(\overline{p}, x)$ (meaning $A(\overline{p}, 0) \vee A(\overline{p}, 1)$.

Of course, any quantified propositional formula can be equivalently written without the quantifiers. However, the quantifier-free formula may be exponentially longer. For example, $\bigvee_{\overline{\epsilon}} A(\overline{\epsilon})$ with $\overline{\epsilon}$ ranging over $\{0,1\}^n$ has size $\Omega(2^n|A|)$ but an equivalent quantified formula $\exists x_1 \ldots \exists x_n A(\overline{x})$ has size only $O(n) + |A|$.

Definition 3.4.3 *Quantified propositional calculus G extends LK by allowing quantified propositional formulas in sequents and by augmenting LK by the following four quantifier rules:*

1. $\forall$:**introduction**

$$\textbf{left}\ \frac{A(B), \Gamma \longrightarrow \Delta}{\forall x A(x), \Gamma \longrightarrow \Delta} \quad \textit{and} \quad \textbf{right}\ \frac{\Gamma \longrightarrow \Delta, A(p)}{\Gamma \longrightarrow \Delta, \forall x A(x)}$$

2. $\exists$:**introduction**

$$\textbf{left}\ \frac{A(p), \Gamma \longrightarrow \Delta}{\exists x A(x), \Gamma \longrightarrow \Delta} \quad \textit{and} \quad \textbf{right}\ \frac{\Gamma \longrightarrow \Delta, A(B)}{\Gamma \longrightarrow \Delta, \exists x A(x)}$$

*where B is any formula. The atom p must not occur in the lower sequents of $\forall$:**right** and $\exists$:**left**.*

Proof system G p-simulates SF and is presumably strictly stronger. But we do not know how to prove this.

Lemma 3.4.4 *G p-simulates SF.*

Proof :

The SF-proof is transformed into a G-proof line by line. A line consisting of a formula A is represented by the sequent $\longrightarrow A$.

We shall only show how G simulates an application of the substitution rule:

$$\frac{A(p_1, \ldots, p_n)}{A(B_1, \ldots, B_n)}.$$

the rest being obvious (cf. Exercise 3.5.8).

To $\longrightarrow \theta(p_1, \ldots, p_n)$ apply n-times $\forall$:**right** to derive

$$\longrightarrow \forall x_1 \ldots \forall x_n A(x_1, \ldots, x_n) \ .$$

The sequent

$$A(B_1, \ldots, B_n) \longrightarrow A(B_1, \ldots, B_n)$$

has a short G -proof. Hence

$$\forall x_1 \ldots \forall x_n A(x_1, \ldots, x_n) \longrightarrow A(B_1, \ldots, B_n)$$

follows by n applications of $\forall$:**left**. Then infer, via cut-rule, the wanted sequent:

$$\longrightarrow A(B_1, \ldots, B_n) \ .$$

q.e.d.

The proof system G can be stratified into subsystems $G_1^*, G_1, G_2^*, G_2, \ldots$ and interesting relations between the subsystems can be proved. This will be done in Chapter **??** using bounded arithmetic.

3.5 Exercises

The next five exercises ask for proofs of five statements, all from [14].

Exercise 3.5.1 *Prove that F and LK (Definition 3.4.1) p-simulate each other.*

Exercise 3.5.2 *Prove the analogue of Theorem 3.1.3 for Extended Frege systems.*

Exercise 3.5.3 *Prove that any formula A having an EF-proof with k steps has also an EF-proof of size $O(k + |A|)$.*

Exercise 3.5.4 *Define* Extended resolution *ER as the resolution proof system R augmented by all clauses from $Ext(\phi)$, for all formulas ϕ, as extra initial clauses (cf. Exercise 2.9.1).*

Prove that ER and EF polynomially simulate each other.

Exercise 3.5.5 *Prove that the minimal numbers of steps in a F-proof and in an EF-proof of a formula are proportional to each other. Hence measuring the size of EF-proofs is the same as measiring the number of steps in F-proofs.*

Exercise 3.5.6 *Define a notion of "a Frege system operating with circuits" and prove thatit is p-equivalent with EF.*

Circuit Frege systems are defined in [**?**]. Somewhat different formalization is in [29].

Exercise 3.5.7 *Prove that the sequent calculus defined in 3.4.1 is sound and complete (even without the cut-rule).*

Exercise 3.5.8 *Complete the details in the proof of Lemma 3.4.4.*

Chapter 4

Constant depth Frege systems

Constant depth Frege systems are natural subsystems of Frege systems. Resolution can be seen as depth 0 or 1 (depending on the formulation) Frege system. The interest in these systems is two-fold. First, these are the most interesting and, essentially, the strongest proof systems for which we can prove strog lower bounds. Secondly, formulas $\langle\Phi\rangle_n$ produced in the translation of a first-order principle Φ (cf. Chapter 1) have the depth bounded by a constant. In fact, proofs in some theories of bounded arithmetic of such Φ yield a family of constant-depth Frege proofs for $\langle\Phi\rangle_n$'s (cf.Chapter **??**). Thus lower bounds for constant depth Frege proofs imply independence results for such theories.

4.1 Definition of the systems and the PHP lower bound

We shall consider a Frege system F in the language 0, 1, $\neg$ and $\vee$. The depth of a formula is the maximum number of blocks of disjunctions and of negations when going from the formula to atomic subformulas. The inductive definition is as follows.

Definition 4.1.1 *The depth of a formula A, denoted $dp(A)$, is defined by the following conditions:*

1. $dp(0) = dp(1) = dp(p) = 0$, *for any atom p.*

2. *$dp(\neg A) := dp(A)$, if A starts with $\neg$, and $dp(\neg A) := 1 + dp(A)$ otherwise.*

3.

$$dp(A \vee B) = \begin{cases} \max(dp(A), dp(B)) & \textit{if both } A \textit{ and } B \textit{ start with } \vee \\ 1 + \max(dp(A), dp(B)) & \textit{if both } A \textit{ and } B \textit{ start with } \neg \\ \max(1 + dp(A), dp(B)) & \textit{if } B \textit{ starts with } \vee \textit{ and } B \textit{ does not} \\ \max(dp(A), 1 + dp(B)) & \textit{if } A \textit{ starts with } \vee \textit{ and } B \textit{ does not} \end{cases}$$

A subsystem of F using only formulas of depth at most d is denoted F_d.

Recall the pigeonhole principle formulas PHP_n from Chapter 1. We shall consider it in the form saying that a relation cannot be a graph of a bijection between $n+1$ and n. It will be convenient to consider instead of proofs of PHP_n refutations of the set $\neg PHP_n$ of the following formulas where i's ranger over $[n+1]$ while j's range over $[n]$:

- $\bigvee_j p_{ij}$, one for each i.
- $\bigvee_i p_{ij}$, one for each j.
- $\neg p_{i_1 j} \vee \neg p_{i_2 j}$, one for each triple $i_1 < i_2$ and j.
- $\neg p_{i j_1} \vee \neg p_{i j_2}$, one for each triple i and $j_1 < j_2$.

We are ready to state a major lower bound in proof complexity, perhaps the most important of all. We give the proof of the theorem at the end of Section 4.4 after developing some machinery.

Theorem 4.1.2 ([33, 38]) *Let $d \geq 2$ and $0 < \delta < 5^{-d}$ be arbitrary. Then for sufficiently large $n \geq 1$, in any F_d-refutation of $\neg PHP_n$ must occur at least 2^{n^δ} different formulas as subformulas. In particular, any such proof must have the size at least 2^{n^δ}.*

It was M. Ajtai[2] who first proved that there are no polynomial size F_d proofs PHP_n. The first exponential lower bound for F_d's have been actually proved for different formulas in Krajíček [24] (cf. Section **??**). Subsequently Ajtai's lower bound have been strenghten to the exponential one by independent proofs in [33] and [38].

4.2 PHP-decision trees

Let $\phi(x_1, \ldots, x_n)$ be a propositional formula. Its truth value on a given truth assignment can be determined by a decision tree (cf. Section 2.2). A decision tree branches at a node according to the truth value of a variable. Hence as we travel in the tree from the root to a leaf we collect bigger and bigger information about the assignment until the truth value of ϕ is determined. In principle the depth of such a tree must be n, the number of all variables (cf.Exercise 4.8.1).

We want to somehow simulate a non-existing truth assignment satisfying $\neg PHP_n$, and we will do it using a modification of decision trees. These new trees will not have labels attached to leaves.

Definition 4.2.1 *Let $D \subseteq [n+1]$ and $R \subseteq [n]$. A PHP-tree over D, R is inductively defined as follows:*

1. *A single node, a root, is a PHP-tree over any D, R.*

2. *For every $i \in D$ the following is a PHP-tree over D, R: The tree branches at the root according to all $j \in R$, and at a son of the root at the branch j continues by a PHP-tree over $D \setminus \{i\}, R \setminus \{j\}$.*

3. *For every $j \in R$ the following is a PHP-tree over D, R: The tree branches at the root according to all $i \in D$, and at a son of the root at the branch i continues by a PHP-tree over $D \setminus \{i\}, R \setminus \{j\}$.*

A PHP-tree is a PHP-tree over $[n+1], [n]$. (We shall often say just "a tree" instead of "a PHP-tree".) The height of a tree T is denoted $||T||$. A tree of the height $\leq k$ is called also a k-tree.

We think of the tree as branching according to queries $f(i) =?$ and $f^{(-1)}(j) =?$, where f is a name for a (non-existing) bijection between $[n+1]$ and $[n]$. Every path in a tree determines a partial 1-to-1 map between $[n+1]$ and $[n]$; we identify the path with the partial map and the tree with the set of all such maps corresponding to all paths.

Consider the simplest example; a tree of depth 1 branching according to all answers to $f(i) =?$. A formula $\bigvee_j p_{ij}$, an axiom of $\neg PHP_n$, is intuitively true at every leaf of the tree because at any leaf one p_{ij} is made true. On the other hand, if we think of f as everywhere defined (and, in particular, as $f(i)$ being defined) then the tree describes all possibilities. Hence the formula

$\bigvee_j p_{ij}$ is "true" in the sense that it holds in all possibilities described by the tree.

Now consider formula $\neg p_{ij} \vee \neg p_{ik}$ for $j \neq k$, another axiom of $\neg PHP_n$. A suitable tree to use for the formula is a tree branching first according to $f^{(-1)}(j) =?$ and then, at a branch corresponding to any $u \in [n+1]$, according to $f^{(-1)}(k) =?$ with answers from $[n+1] \setminus \{u\}$. At every path through the tree either $f(i) \neq j$ or $f(i) \neq k$ and hence the formula is satisfied. As before, thinking of f as an injective map that is onto, the branching of the tree describes all possibilities. Hence again the formula is "true" in the sense of being true in all situations described by a tree.

Our preliminary strategy is thus the following. We assign to all formulas a tree and a subset of (the set of paths in) the tree where the formula is true.

A difficulty arises: As there is no bijection f, no tree can decide the truth of all atoms. This implies that formulas may have different trees attached to them and we need a way how to compare them. Explaining more informally would rather obfuscate things so we launch into a formal treatment.

Definition 4.2.2 *1. $\mathcal{M}$ is the set of all partial bijections between $[n+1]$ and $[n]$. Maps from $\mathcal{M}$ are denoted $\alpha, \beta, \gamma, \ldots$. The size of α is the size of its domain and it is denoted $|\alpha|$.*

2. α and β are incompatible, $\alpha \perp \beta$ in symbols, iff $\alpha \cup \beta \notin \mathcal{M}$. The fact that α and β are compatible will be denoted $\alpha || \beta$.

3. Let $H \subseteq \mathcal{M}$ and let T be a tree (tacitly a PHP-tree). Tree T refines set H, $H \lhd T$ in symbols, iff for all $\alpha \in T$ either $\forall \beta \in H, \alpha \perp \beta$ or $\exists \gamma \in H, \gamma \subseteq \alpha$.

4. For T, S trees, $T \times S := \{\alpha \cup \beta \mid \alpha \in T, \beta \in S\}$. It is called a common refinement of T and S.

5. For $H \subseteq \mathcal{M}$ and S a tree, the projection of H on S is the set $S(H) := \{\alpha \in S \mid \exists \gamma \in H, \gamma \subseteq \alpha\}$.

We shall often use the definition of refinment in the following form: $H \lhd T$ iff whenever an $\alpha \in T$ is compatible with some $\beta \in H$ then it contains some $\gamma \in H$.

Throughtout this section letters $H, K, \ldots$ will denote subsets of $\mathcal{M}$ while letters S and T are reserved for trees. As stated earlier, Greek letters $\alpha, \beta, \ldots$ denote elements of $\mathcal{M}$.

Lemma 4.2.3 *If* $|\delta| \leq n - ||S||$ *then* $\exists \gamma \in S, \gamma || \delta$.

Proof :

Walk through the tree S answering queries according to δ whenever it applies, and arbitrarily but consistently with δ otherwise. The assumption that $|\delta| \leq n - ||S||$ implies that we do not run into a contradiction before reaching a leaf of S. Map γ is the map determined by the particular path.

q.e.d.

Lemma 4.2.4 *Assume* $||S|| + ||T|| \leq n$ *and* $H \lhd S \lhd T$. *Then also* $H \lhd T$.

Proof :

Assume $\delta \in T$ is compatible with some $\alpha \in H$. We want to show that δ contains some element of H.

By Lemma 4.2.3 $\exists \gamma' \in S, \gamma' || \delta$. By this, and by $S \lhd T$, $\exists \gamma \in S, \gamma \subseteq \delta$. Such γ is necessarily compatible with α and hence, by $H \lhd S$, $\exists \alpha' \in H, \alpha' \subseteq \gamma$. Hence $\alpha' \subseteq \delta$ too.

q.e.d.

Lemma 4.2.5 *Assume* $||S|| + ||T|| \leq n$. *Then* $S \times T$ *is a* PHP*-tree such that* $||S \times T|| \leq ||S|| + ||T||$, *and such that* $S \lhd S \times T$ *and also* $T \lhd S \times T$.

Proof :

The bound to the height of $S \times T$ is obvious. We prove that $S \lhd S \times T$, the second statement is proved identically.

Assume that $\beta \cup \gamma \in S \times T$, with $\beta \in S$ and $\gamma \in T$, is compatible with some $\alpha \in S$. Then necessarily $\alpha = \beta$, i.e. $\beta \cup \gamma$ contains an element of S.

q.e.d.

Lemma 4.2.6 *Assume* $||S|| + ||T|| \leq n$ *and* $H \lhd S \lhd T$. *Then*

1. $T(S(H)) = T(H)$.

2. $T(S) = T$.

3. $S(H) = S$ *iff* $T(H) = T$.

Proof :

The inclusion $T(S(H)) \subseteq T(H)$ follows from the definition. For the opposite inclusion assume that $\beta \in T(H)$ because $\beta \supseteq \gamma$ for some $\gamma \in H$. Using Lemma 4.2.3 $S \triangleleft T$ implies that $\exists \alpha \in S, \alpha \subseteq \beta$. Such β is then compatible with γ and hence, as $H \triangleleft S$, $\exists \gamma' \in H, \gamma' \subseteq \alpha$. So we have $\gamma' \subseteq \alpha \subseteq \beta$ and so $\beta \in T(S(H))$. This proves part 1.

Part 2. follows from part 1. by taking $H := \{\emptyset\}$. For part 3. assume first $S(H) = S$. By parts 2. and 1.: $T(S) = T$, and $T(S(H)) = T(H)$. So $T = T(H)$.

Finally, assume that $T(H) = T$. Let $\alpha \in S$. By Lemma 4.2.3 there is $\beta \in T$ compatible with α. By the assumption also $\beta \in T(H)$ and so $\exists \gamma \in H, \gamma \subseteq \beta$. But such γ is compatible with α and hence, by $H \triangleleft S$, $\exists \gamma' \in H, \gamma' \subseteq \alpha$. So $\alpha \in S(H)$ as we wanted to show.

q.e.d.

Lemma 4.2.7 *1.* $S(\bigcup_i H_i) = \bigcup_i S(H_i)$.

2. If $H_0, H_1 \subseteq T$ *and* $H_0 \cap H_1 = \emptyset$ *then* $T(H_0) \cap T(H_1) = \emptyset$.

3. If $S \triangleleft T$, $||S|| + ||T|| \leq n$ *and* $H \subseteq S$ *then* $T(S \setminus H) = T \setminus T(H)$.

Proof :

The first two propositions follow directly from definitions. By Lemma 4.2.6 $T(S) = T$, hence the last proposition follows from the first two.

q.e.d.

4.3 k-evaluations

We continue with some fixed (and large enough) $n \geq 1$ and we shall also fix a parameter $1 \leq k \leq n$. Let Γ be a set of formulas in the atoms of PHP_n that is closed under subformulas.

Definition 4.3.1 *A k-evaluation of Γ is a map*

$$\phi \in \Gamma \longrightarrow H_\phi \subseteq S_\phi$$

assigning to a formula $\phi \in \Gamma$ a k-tree S_ϕ and its subset H_ϕ, such that the following four conditions are satisfied:

1. *$S_0 := S_1 := \{\emptyset\}$, i.e. the tree consisting of the root only. Further $H_0 := \emptyset$ and $H_1 := S_1$.*

2. *$S_{p_{ij}}$ is the depth 2 tree that first branches according to $f(i) =?$ and then according to $f^{(-1}(j) =?$. $H_{p_{ij}} := \{(i, j)\}$, the only path in $S_{p_{ij}}$ of length 1.*

3. *$S_{\neg\phi} := S_\phi$ and $H_{\neg\phi} := S_\phi \setminus H_\phi$, whenever $\neg\phi \in \Gamma$.*

4. *Assume $\phi = \bigvee_i \phi_i$ is in Γ (where the big disjunction abbreviates arbitrarily bracketet binary disjunctions). Then*

$$\bigcup_i H_{\phi_i} \lhd S_\phi \quad \textit{and} \quad H_\phi := S_\phi(\bigcup_i H_{\phi_i})$$

If $H_\phi = S_\phi$ we say that ϕ is "true" w.r.t. to the k-evaluation, (or simply that it is "true" if the evaluation is fixed).

Lemma 4.3.2 *Assume that (H, S) is a k-evaluation of all formulas occuring as subsformulas in an axiom of $\neg PHP_n$, and that $k \leq n - 2$.*

Then the axiom is "true" with respect to the evauation.

Proof :

Consider an axiom of the form $\phi = \bigvee_j p_{ij}$ for some fixed $i \in [n + 1]$. By Definition 4.3.1 $H_{p_{ij}} = \{(i, j)\}$ and S_ϕ must refine the set $H = \{(i, j) \mid j \in [n]\}$. Note that H itself is a 1-tree and that $H(H) = H$.

Hence $T(H) = T$ holds also in the common refinment of H and S_ϕ by Lemma 4.2.6, and by the same lemma again also $S_\phi = S_\phi(H) = H_\phi$.

We leave the other axioms to Exercise 4.8.2.

q.e.d.

Now we prove that Frege rules are sound even for the notion of "true" w.r.t. a k-evaluation.

Lemma 4.3.3 *There exists a constant $c_F \geq 1$ such that if (H, S) is k-evaluation of all formulas occuring as subformulas in an instance of an F-rule, $k \leq n/c_F$, and all hypotheses of the instance of the rule are "true" w.r.t. the evaluation then also the conclusion of the rule is "true".*

The constant c_F depends only on the particular rules in F.

Proof :

Consider an F-rule of the form

$$\frac{A_1(q_1,\ldots,q_t), \qquad \ldots \qquad , A_s(q_1,\ldots,q_t)}{A_{s+1}(q_1,\ldots,q_t)}$$

Let r be a number bigger than the number of subformulas in the rule.

Assume that (H,S) is a k-evaluation of formulas occuring in some instance

$$\frac{A_1(B_1,\ldots,B_t), \qquad \ldots \qquad , A_s(B_1,\ldots,B_t)}{A_{s+1}(B_1,\ldots,B_t)}$$

of the rule, and such that $k \leq n/r$. We also assume that all

$$A_1(B_1,\ldots,B_t),\ldots,A_s(B_1,\ldots,B_t)$$

are "true" with respect to the evaluation, i.e.

$$H_{A_i(B_1,\ldots,B_t)} = S_{A_i(B_1,\ldots,B_t)}, \quad \text{for } 1 \leq i \leq s \ .$$

Let Γ be all formulas occuring in the instance, and Γ_0 its subset consisting of formulas C of the form $A'(B_1,\ldots,B_t)$ where A' is a subformula of some A_i, $i \leq s+1$.

By the choice of r, $|\Gamma_0| < r$, and so there is a common refinement T of all S_C, and $||T|| \leq \frac{r-1}{r}n$ (by Lemma 4.2.5). In particular, $||T|| + ||S_C|| \leq n$ for all $C \in \Gamma_0$.

Claim: *The map defined by $C \in \Gamma_0 \to T(H_C)$ is a map of formulas in Γ_0 into the Boolean algebra of subsetes of T such that:*

(a) The negation corresponds to the complement: $T(H_{\neg C}) = T \setminus T(H_C)$.

(b) The disjunction correspinds to the union: $T(H_{C\vee D}) = T(H_C) \cup T(H_D)$.

(c) All hypotheses $C = A_i(B_1,\ldots,B_t)$, $i \leq s$, of the instance of the rule get the value 1 in the Boolean algebra: $T(H_C) = T$.

For part (a): If $\neg C \in \Gamma_0$, $H_{\neg C} = S_C \setminus H_C$, and hence $T(H_{\neg C}) = T \setminus T(H_C)$ by Lemma 4.2.7.

For part (b) let $C \vee D \in \Gamma_0$. We need to consider cases distinguished by the form of C and D; we shall treat only the hardest case when both C and D are themselves disjunctions. Assume $C = \bigvee_u C_u$ and $D = \bigvee_v D_v$. By Lemma 4.2.7:

$$H_{C\vee D} = S_{C\vee D}(\bigcup_u H_{C_u}) \cup S_{C\vee D}(\bigcup_v H_{D_v})$$

hence by Lemmas 4.2.6 and 4.2.7:

$$T(H_{C\vee D}) = T(S_{C\vee D}(\bigcup_u H_{C_u})) \cup T(S_{C\vee D}(\bigcup_v H_{D_v})) =$$

$$T(\bigcup_u H_{C_u}) \cup T(\bigcup_v H_{D_v}) =$$

$$T(S_C(\bigcup_u H_{C_u})) \cup T(S_D(\bigcup_v H_{D_v})) =$$

$$T(H_C) \cup T(H_D) \ .$$

Part (c) follows by Lemma 4.2.6:

$$T(H_{A_i(\overline{B})}) = T(S_{A_i(\overline{B})}) = T$$

for $i \leq s$.

The lemma follows noting that any Frege rule is valid in any Boolean algebra (cf.Exercise 4.8.3).

q.e.d.

Our strategy for proving Theorem 4.1.2 is now clear. Having an alleged F_d-refutation of $\neg PHP_n$ we take a k-evaluation (with small enough k) of the set of all formulas occuring in the refutation. This would lead to contradiction by Lemmas 4.3.2 and 4.3.3. Hence if we manage to construct a k-evaluation of any small set of formulas we can conclude that no F_d-refutation of $\neg PHP_n$ can be small.

4.4 The existence of k-evaluations

This section is devoted to the construction of k-evaluations of small sets of formulas. The qualification small will mean of size at most 2^{n^δ}, for suitable $\delta > 0$.

It is quite easy to find small sets which have no k-evaluation with $k < n$, cf. Exercise 4.8.4, and that is insufficient for the key Lemmas 4.3.2 and 4.3.3. This forces us to employ a simplification procedure before trying to find a k-evaluation with small k. The simplification will be done by a partial truth assignment.

We shall think of the set $\mathcal{M}$ as of the set of partial bijections between a subset of domain D and range R. $D = [n+1]$ and $R = [n]$ at the beginning, as earlier.

Definition 4.4.1 *Let $\alpha, \rho \in \mathcal{M}$. Define the restriction of α by ρ to be:*

$$\alpha^\rho = \begin{cases} \alpha \setminus \rho & \text{if } \alpha || \rho \\ undefined & \text{if } \alpha \perp \rho \end{cases}$$

Further define:

1. $H^\rho := \{\alpha^\rho \mid \alpha \in H\}$.
2. $D^\rho := D \setminus dom(\rho)$.
3. $R^\rho := R \setminus rng(\rho)$.
4. $n_\rho := |R^\rho| (= n - |\rho|)$.

Our strategy in the construction of a k-evaluation of a set Γ will be the following. We construct the evaluation in steps. We start by defining the evaluation for atoms and constants in Γ: that is canonical by Definition 4.3.1. At every step we extend the k-evaluation to negations and to disjunctions of formulas for which it is already defined (hence the number of steps is bounded by the maximal depth of a formula in Γ). The case of negations is again canonical and only the case of disjunction will cause us a problem. To extend the definition to disjunctions we will need to apply a restriction by some ρ. The following lemma essentialy says that the part of the evaluation already constructed will still work after the restriction.

We continue using the convention that $S, T, \ldots$ denote PHP-trees.

Lemma 4.4.2 *Let $\rho \in \mathcal{M}$ be arbitrary. Then:*

1. *If $H \lhd S$ then $H^\rho \lhd T^\rho$.*
2. *If $|\rho| + ||S|| \leq n$ then S^ρ is a PHP-tree over D^ρ and R^ρ.*
3. *If $H \lhd S$ then $S^\rho(H^\rho) = (S(H))^\rho$.*

We leave the proof to the Exercise 4.8.5. The next lemma is the key technical step in the construction of k-evaluations.

Lemma 4.4.3 *Let $0 < \delta < \epsilon < 1/5$. Let $H_i \subseteq \mathcal{M}$, for $i \leq s$. Assume that $||H_i|| \leq k$ for all $i \leq s$. Assume that*

$$k \leq n^\delta \quad and \quad s \leq 2^k$$

and that n is large enough. Then there exists $\rho \in \mathcal{M}$ such that $n_\rho = n^\epsilon$ and such that there exist PHP-trees S_i, $i \leq s$, over D^ρ and R^ρ, satisfying

1. $H_i^\rho \lhd S_i$,
2. $||S_i|| \leq k$.

Proof :

Assume first that we have just one H; we shall consider the case of having s sets H_i at the end.

We shall describe a game played by two players with the set H. In the proof it will be played with H^ρ actually but we consider only H first not to complicate the notation.

At the beginning player I pick an $h_1 \in H$. Player II replies my some $\delta_1 \in \mathcal{M}$ such that $dom(h_1) \subseteq dom(\delta_1)$, $rng(h_1) \subseteq rng(\delta_1)$, and such that no proper submap of δ_1 has this property. It may be that $\delta_1 = h_1$ or that at least $\delta_1 \supseteq h$, some $h \in H$: In that case the game ends. Otherwise necessarily $\delta_1 \perp h_1$ and the game moves to the next round. Generally, before round $t \geq 2$, the players have constructed sequences $h_1, \ldots, h_{t-1}$ (the moves of I) and $\delta_1 \subseteq \ldots \subseteq \delta_{t-1}$ (the moves of II). At the t-th steps player I picks some $h_t \in H$ compatible with δ_{t-1}; if no such h_t exists the game stops. Player II then extends δ_{t-1} to some $\delta_t \in \mathcal{M}$ such that $dom(h_t) \subseteq dom(\delta_t)$, $rng(h_t) \subseteq rng(\delta_t)$, and such that no proper submap of δ_t containing δ_{t-1} has this property. If δ_t contains some $h \in H$ then the game stops, otherwise the players move to the next round.

The use of this game is described in the following claim which follows immediately from the definition when the game stops.

Claim 1: *For any fixed strategy of the player I consider the set*

$$S := \{\delta_t \mid \delta_1 \subseteq \ldots \subseteq \delta_t \ \text{ is a finished play in some strategy of } II \ \}$$

Then the set S is a PHP-tree and $H \lhd S$.

To simplify things we shall fix one strategy of I: We fix an ordering $h^1, h^2, \ldots$ of H and player I always picks in his move the first h in the ordering compatible with the previous move of II. We shall call player I using this startegy I_{fix}.

Let us call the set of all pairs (i, j) in all $h_\ell \setminus \delta_{\ell-1}$ the critical pairs of the play. These are exactky the pairs for which II is required to specify $f(i)$ and $f^{(-1)}(j)$. If the number of critical pairs in all finished games against I_{fix} is bounded by r then clearly $||S|| \leq 2r$. Hence we would like to show that the number of critical pairs is bounded by $k/2$. However, it is easy to construct a set of small maps from $\mathcal{M}$ such that any finished game must contain $\geq n/2$ critical pairs (cf. Exercise 4.8.6). This is the place where we employ a restriction by suitable ρ.

Assume we fix $\rho \in \mathcal{M}$ and restrict first H by ρ, and play the game on H^ρ (we continue to use the same ordering of elements of H for I_{fix}). This is the same as if we defined $\delta_0 := \rho$ and required h_1 and δ_1 to contain δ_0.

Claim 2: *There exists $\rho \in \mathcal{M}$, $n_\rho = n^\epsilon$, such that every play (tacitly against I_{fix}) on H^ρ contains at most $k/2$ critical pairs.*

We shall prove the claim by contradiction. Assume that there is no such ρ. Hence for every ρ there is a play, resulting in the moves $\delta_1 \subseteq \ldots \subseteq \delta_t$ of II, that contains at least $k/2+1$ critical pairs. In fact, we will truncate the play when it reaches the $(k/2+1)$-st critical pair, so we shall assume that there are exactly $k/2$ critical pairs (this is only for a simplification of a computation). Fix one such play for each ρ.

Now concentrate on one fixed ρ and the associated fixed play. Note that all critical pairs are disjoint, and are also also disjoint from ρ. Hence the set τ containing ρ and all critical pairs is actually an element of $\mathcal{M}$, and $|\tau| = |\rho| + k/2$.

Having τ we cannot a priori determine ρ but we can determine the first move h_1^ρ of I_{fix}: It is the first $h^\rho \in H^\rho$ that is compatible with τ.

Now note that we can actually encode by a small information the critical pairs in h_1^ρ and the first move δ_1 of II: Critical pairs from h_1^ρ form one of its $\leq 2^k$ subsets (here we use that $||H|| \leq k$), and the move of II is determined by giving a value (resp. inverse value) of f for every i (resp. j) occuring in the critical pairs in h_1^ρ. There is $\leq 2(k/2) = k$ such i's and j's, and at most n^ϵ values to choose from: This is because the values II chooses must be outside the domain (resp. the range) of τ and $n - |\tau| \leq n - |\rho| = n^\epsilon$. Hence there are at most $(n^\epsilon)^k$ possibilities of II's action on the critical pairs. All together, we can encode II's first move δ_1 by a number $\leq (2n^\epsilon)^k$.

Once we know δ_1 we replace in τ by δ_1 all critical pairs in h_1^ρ, getting some τ'. But know we can reconstruct also the second move h_2^ρ of I_{fix}: It is the first $h^\rho \in H^\rho$ compatible with τ'. Hence we proceed as before: Encode the II's second move by a number $\leq (2n^\epsilon)^k$, and replace in τ' all critical pairs in h_2^ρ by δ_2, etc.

There are at most $k/2$ moves before we get $k/2$ critical pairs. Hence the whole (truncated) play can be encoded by τ togerther with a $k/2$-tuple of numbers $\leq (2n^\epsilon)^k$, i.e. by a number $\leq (2n^\epsilon)^{k^2/2}$.

Because τ together with the auxiliari information determines ρ, the numbers

$$a := \text{ the number of different } \rho \text{ of size } n - n^\epsilon \; = \binom{n+1}{n^\epsilon}\binom{n}{n^\epsilon}(n-n^\epsilon)!$$

and

$$b := \text{ the number of different } \tau \text{ of size } n - n^\epsilon - k/2 \; =$$
$$\binom{n+1}{n^\epsilon - k/2}\binom{n}{n^\epsilon - k/2}(n - n^\epsilon - k/2)!$$

must satisfy the inequality:

$$a \leq b \cdot (2n^\epsilon)^{k^2/2}$$

All this argument was for one set H. However, if we had s of them we just encode by a number $\leq s$ which of the sets is the one in which we have, for a given ρ, a play with at least $k/2$ critical pairs. Hence, if no suitable ρ existed, we would have to have

$$a \leq s \cdot b \cdot (2n^\epsilon)^{k^2/2}$$

It is not difficult to compute that this inequality does not hold if the parameters satisfy the hypotheses of the lemma.

q.e.d.

Now we are going to use a restriction ρ in order to construct a k-evaluation. We will need a notion of a formula restricted by ρ defined as follows.

$$p_{ij}^\rho = \begin{cases} 1 & i \in dom(\rho) \wedge \rho(i) = j \\ 0 & \{(i,j)\} \perp \rho \\ p_{ij} & \text{otherwise} \end{cases}$$

and then take for ϕ^ρ the formula ϕ with all atoms p_{ij} replaced by p_{ij}^ρ.

Lemma 4.4.4 *Let $0 < \delta < \epsilon < 5^{-d}$. Then for sufficiently large $n \geq 1$ every set Γ of size at most 2^{n^δ} and closed under subformulas there exists a map ρ, $|\rho| = n - n^\epsilon$, and an n^δ-evaluation of Γ^ρ.*

Proof :

Let $s = 2^{n^\delta}$ and $k = n^\delta$. Assume that $|\Gamma| \leq s$. Pick $\epsilon_0 > 0$ such that $0 < \delta < \epsilon_0^d < \epsilon_0 < 5^{-d}$. We shall construct the restriction ρ and the k-evaluation of Γ^ρ in d steps.

Put $\rho_0 := \emptyset$ and let ν_0 be the canonical (by Definition 4.3.1) 2-evaluation of the depth 0 formulas in Γ, i.e. of the constants and the atoms. In step $1 \leq t \leq d$ we assume that we already have restrictions $\rho_0 \subseteq \ldots \subseteq \rho_{t-1}$ with $n_{\rho_\ell} = n^{\epsilon_0^\ell}$ and a k-evaluation ν_{t-1} of all depth $\leq t-1$ formulas in $\Gamma^{\rho_{t-1}}$.

To extend the evaluation to depth t formulas we apply Lemma 4.4.3 with $n := n_{\rho_{t-1}}$ and the parameters δ and ϵ_0 fixed earlier. This will give us a restriction on the universe $[n+1] \setminus dom(\rho_{t-1})$, $[n] \setminus rng(\rho_{t-1})$, i.e. a restriction $\rho_t \supseteq \rho_{t-1}$ on $[n+1]$, $[n]$. By Lemma 4.4.2, $\nu_{t-1}^{\rho_t}$ will still work for the depth $\leq t-1$ formulas and this evaluation is extended to an evaluation ν_t of depth $\leq t$ formulas in Γ^{ρ_t} by the virtue of Lemma 4.4.3.

The final $\rho := \rho_d$ and $\nu := \nu_d$ satisfy the requirements of the lemma with $\epsilon := \epsilon_0^d$.

q.e.d.

Proof of Theorem 4.1.2:

We are now ready to prove the theorem. For the sake of contradiction assume that π is an F_d-refutation of $\neg PHP_n$ with less than 2^{n^δ} different formulas. Let Γ be the set of all formulas occuring in π as subformulas.

Take the ρ and the k-evaluation (with $k := n^\delta$) of Γ^ρ provided by Lemma 4.4.4. For large enough n it holds that $n^\delta < n/c_F$, where c_F is the constant from Lemma 4.3.3. By Lemmas 4.3.2 and 4.4.2, the axioms of $(\neg PHP_n)^\rho = \neg PHP_{n_\rho}$ are "true" w.r.t. the evaluation. By Lemma 4.3.3 all steps in π are "true" too. But the last formula, the constant 0, is not "true". That is a contradiction.

4.5 Counting principles

The PHP -principle says that there is no pairing between sets of sizes differing by 1. More general principles can be considered. Fix $m \geq 2$. The *counting modulo m principle* says that a set with n elements cannot be partitioned into m-element blocks unless its size is divisible by m.

The propositional formulation of the principle will use atoms q_e, one for each m-element subset e of $[n]$. The set of m-element subsetes of $[n]$ will be denoted simply $\binom{[n]}{m}$.

Definition 4.5.1 *The axioms of the* $\neg Count^n_m$ *are:*

1. $\neg q_e \vee \neg q_f$, *whever* $e, f \in \binom{[n]}{m}$ *are incompatible (denoted* $e \perp f$*);* $e \neq f \wedge e \cap f \neq \emptyset$.

2. $\bigvee_{e:i \in e} q_e$, *for all* $i \in [n]$.

$Count^n_m$ *is the disjunction of the negations of all axioms of* $\neg Count^n_m$.

We shall leave it as an advanced Exercise (see 4.8.7) for the reader to modify the machinery of PHP-trees and k-evaluations to $Count^n_m$. In particular, $Count_m$-trees over $[n]$ branch accoring to queries $i \in ?$, each branch corresponding to one $e \in \binom{[n]}{m}$ containing i and consistent with blocks on the path to the node. Everything will then work analouglsy as in the proof of Theorem 4.1.2 and we get the following lower bound.

Theorem 4.5.2 *For any* $m \geq 2$ *and* $d \geq 3$ *there is* $\delta > 0$ *such that for all sufficiently large* n *not divisible by* m*, in any* F_d*-refutation of* $\neg Count^n_m$ *must occur at least* 2^{n^δ} *different subformulas. In particular, any such refutation must have the size at least* 2^{n^δ}.

4.6 Relation of PHP and $Count_m$ principles

By Theorems 4.1.2 and 4.5.2 neither PHP principle nor $Count_m$ principles have subexponential F_d-proofs. It is thus natural to study the strength of F_d when augmented by all instances (of a priori bounded depth) of either PHP or $Count_m$ as extra axioms.

Lemma 4.6.1 *For any* $m \geq 2$ *there are* $d \geq 2$ *and* $c \geq 1$ *such that for all* $n \geq 1$ *there are* F_d*-proofs of size* n^c *of* PHP_n *from instances of the* $Count_m$ *principle.*

Proof :

Consider the set N consisting of disjoint copies of $[n+1]$ and $[n]$, and further $m-2$ disjoint copies of $[n]$. Hence $|N| = m \cdot n + 1$. Assume f is a bijection between $[n+1]$ and $[n]$. Then the set of all blocks of the form

$$\{i, f(i), \ldots, f(i)\}$$

with $i \in [n+1]$ and $f(i)$' taken from all $m-1$ copies of $[n]$, form a partition of $[N]$ into m-element blocks. This violates an instance of the $Count_m^N$-principle.

This informal argument can be made formal quite easily (cf. **??** or Exercise 4.8.8).

q.e.d.

The opposite direction is much more interesting.

Theorem 4.6.2 ([42, 7]) *Let $m \geq 2$ be fixed. For any $d \geq 2$ there exists $\delta > 0$ such that for all $n \geq m$ large enough and not divisible by m the following holds:*

In any F_d-proof of $Count_m^n$ from instances of PHP must occur at least 2^{n^δ} different subformulas. In particular, any such proof must have the size at least 2^{n^δ}.

Proof :

First it is easy to see that several instances of PHP are equivalent, over F_d by short proofs, to just one instances: Just define the instance by definiton by cases; it is the first instance in the list for which PHP-fails, or something trivial otherwise.

Let this one instance be the instance for PHP_N for formulas ψ_{ij} (built from the atoms of $Count_m^n$) replacing the atoms p_{ij} of PHP_N. In particular, $i \in [N+1]$ and $j \in [N]$.

Let Γ be all formulas occuring in an F_d-proof of $Count_m^n$ from the instance of PHP_N. If Γ were small there would be ρ (a partial m-partition of $[n]$) and a k-evaluation (H, S) of all formulas in Γ^ρ making all axioms of $\neg Count_m^{n_\rho}$ "true". Hence also the $\neg PHP_N(\psi_{ij})^\rho$ is "true".

Let T be a $Count_m$-tree refining all trees $S_{\psi_{ij}^\rho}$, and define the map:

$$(i,j) \in [N+1] \times [N] \to A_{ij} := T(S_{\psi_{ij}^\rho})$$

We think of A_{ij} simply as of sets of partial m-partitions of n_ρ.

Claim: *The following identities hold: $\bigcup_j A_{ij} = T$, $\bigcup_i A_{ij} = T$, $A_{i_1 j} \cap A_{i_2 j} = \emptyset$ if $i_1 \neq i_2$, and $A_{ij_1} \cap A_{ij_2} = \emptyset$ if $j_1 \neq j_2$.*

The claim follows from the fact that the instances of the PHP is "true" w.r.t. the k-evaluation.

The claim leads to a contradiction as counting the size of $\bigcup_{ij} A_{ij}$ first by rows or by columns leads to two different values: $(N+1) \cdot |T|$ and $N \cdot |T|$. Hence no such k-evaluatin can exists and consequently the proof cannot contain only a small number of formulas. The particular values of parameters are the same as in Theorems 4.1.2 or 4.5.2.

q.e.d.

The mutual relation of counting principles with different moduli m is more complicated.

4.7 Mutual relations of counting principles

We shall look at mutual relations between counting principles in this section. The first statement simplifies a bit what moduli we need to consider.

Lemma 4.7.1 *Let $m \geq 2$ and let $p_1, \dots, p_k$ be all prime divisors of m. There there are $d \geq 2$ and $c \geq 1$ such that*

1. *$Count^n_m$, n not divisible by m, can be derived by an F_d-proof of size n^c from instances of $Count_{p_i}$, all $i \leq k$.*
2. *Any $Count^n_{p_i}$, n not divisible by p_i, can be derived by an F_d-proof of size n^c from an instance of $Count_m$.*

We shall not prove the lemma here,as it is much easier to formalize via bounded arithmtic, cf. **??**.

The lemma means that when studying the mutual relation we can concentrate just on counting principles with moduli that are primes. The following theorem has be first prove inthe form of the non-existence of polynomial upper bound in [4, 6, 43].

Theorem 4.7.2 ([11]) *Let $p, q \geq 2$ be two fixed different primes. For any $d \geq 2$ there exists $\delta > 0$ such that for all $n \geq q$ large enough and not divisible by q the following holds:*

In any F_d-proof of $Count^n_q$ from instances of $Count_p$ must occur at least 2^{n^δ} different subformulas. In particular, any such proof must have the size at least 2^{n^δ}.

The proof will not be given in this draft.

4.8 Exercises

Exercise 4.8.1 *Show that any formula ϕ decidable by a decision tree of depth k is equivalent to a k-DNF, i.e. a formula which is a disjunction of conjunctions, each of size at most k, and that the same holds for $\neg\phi$.*

On the other hand, show that if both ϕ and $\neg\phi$ are expressible as k-DNF then ϕ can be decided by a decision tree of depth $\leq k^2$.

Exercise 4.8.2 *Prove that all axioms of $\neg PHP_n$ are "true" w.r.t. a k-evaluation, as long as $k \leq n-2$. (cf. Lemma 4.3.2)*

Exercise 4.8.3 *Prove that any Frege rule is sound in any Boolean algebra $\mathcal{B}$: If hypotheses of an instance of the rule get value $1_{\mathcal{B}}$ then also the conclusion of the rule gets value $1_{\mathcal{B}}$.*

Exercise 4.8.4 *Find small sets, say of size $n^{O(1)}$, of formulas that have no k-evaluation with $k < n$.*

Exercise 4.8.5 *Prove Lemma 4.4.2.*

Exercise 4.8.6 *Construct a set of constant size maps from $\mathcal{M}$ such that any finished game must contain $\geq n/2$ critical pairs (cf. Lemma 4.8.3).*

Exercise 4.8.7 *Define the notion of $Count_m$-tree and the corresponding notion of k-evaluations, and prove Theorem 4.5.2.*

Exercise 4.8.8 *Prove Lemma 4.6.1.*

Bibliography

[1] Alekhnovich, M., Ben-Sasson, E., Razborov, A. A., and Wigderson, A., Pseudorandom generators in propositional proof complexity, *Electronic Colloquium on Computational Complexity*, Rep. No.**23**, (2000). Ext. abstract in: *Proc. of the 41^{st} Annual Symp. on Foundation of Computer Science*, (2000), pp.43-53.

[2] Ajtai, M. (1988) The complexity of the pigeonhole principle, in: *Proc. IEEE 29^{th} Annual Symp. on Foundation of Computer Science*, pp. 346-355.

[3] —— (1990) Parity and the pigeonhole principle, in: *Feasible Mathematics*, Eds. S.R.Buss and P.J.Scott, pp.1-24. Birkhauser.

[4] —— (1994) The independence of the modulo p counting principles, in: *Proceedings of the 26th Annual ACM Symposium on Theory of Computing*, pp.402-411. ACM Press.

[5] Alon, N., and Boppana, R. (1987) The monotone circuit complexity of Boolean functions, *Combinatorica*, **7(1)** : 1-22.

[6] Beame, P., Impagliazzo, R., Krajíček, J., Pitassi, T., and Pudlák, P. (1994) Lower bounds on Hilbert's Nullstellensatz and propositional proofs, submitted.

[7] Beame, P., and Pitassi, T. (1993) Exponential separation between the matching principles and the pigeonhole principle, preprint.

[8] Ben-Sasson, E., and Wigderson, A. (1999) Short proofs are narrow - resolution made simple, in: *Proc. of the 31^{st} ACM Symp. on Theory of Computation*, pp. 517-526.

[9] Blake, A. (1937) Canonical expressions in boolean algebra, PhD. Thesis, University of Chicago.

[10] Buss, S. R. (1987) The propositional pigeonhole principle has polynomial size Frege proofs, *J. Symbolic Logic*, **52**: 916-927.

[11] Buss, R. S. Impagliazzo, R., Krajíček, J., Pudlák, P., Razborov, A. A., and Sgall, J., Proof complexity in algebraic systems and bounded depth Frege systems with modular counting, *Computational Complexity*, **6(3)**, (1996/1997), pp.256-298.

[12] Cook, S A. (1971) The complexity of theorem proving procedures, in: *Proc. 3^{rd} Annual ACM Symp. on Theory of Computing*, pp. 151-158. ACM Press.

[13] —— (1975) Feasibly constructive proofs and the propositional calculus, in: *Proc. 7^{th} Annual ACM Symp. on Theory of Computing*, pp. 83-97. ACM Press.

[14] Cook, S. A., and Reckhow, A. R. (1979) The relative efficiency of propositional proof systems, *J. Symbolic Logic*, **44(1)**:36-50.

[15] Cook, W., Coullard, C. R., and Turán, G. (1987) On the complexity of cutting plane proofs, *Discrete Applied Mathematics*, **18**:25-38.

[16] Craig, W. (1957) Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory, *Journal of Symbolic Logic*, **22(3)**:269-285.

[17] —— (1957) Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory, *Journal of Symbolic Logic*, **22(3)**: 269-285.

[18] Davis, M., and Putnam, H. (1960) A computing procedure for quantification theory, *Journal of the ACM*, **7(3)**, pp.210-215.

[19] Dowd, M. (1979) Propositional representations of arithmetic proofs, *PhD Thesis*, *University of Toronto*.

[20] Haken,A. (1985) The intractability of resolution, *Theoretical Computer Science*, **39**:297-308.

[21] Karchmer, M., and Wigderson, A. (1988) Monotone circuits for connectivity require super - logarithmic depth, in: *Proc.* 20^{th} *Annual ACM Symp. on Theory of Computing*, pp.539-550. ACM Press.

[22] Krajíček, J. (1989a) On the number of steps in proofs, *Annals of Pure and Applied Logic*, **41**:153-178.

[23] Krajíček, J. (1989b) Speed-up for propositional Frege systems via generalizations of proofs, *Commentationes Mathematicae Universitas Carolinae*, **30(1)**:137-140.

[24] Krajíček, J. (1994) Lower bounds to the size of constant-depth propositional proofs, *Journal of Symbolic Logic*, **59(1)**: 73-86.

[25] Krajíček, J. (1995) *Bounded arithmetic, propositional logic, and complexity theory*, Encyclopedia of Mathematics and Its Applications, Vol. **60**, Cambridge University Press.

[26] J. Krajíček, On Frege and Extended Frege Proof Systems. in: "Feasible Mathematics II", eds. P. Clote and J. Remmel, Birkhauser, (1995), pp. 284-319.

[27] Krajíček, j. (1997) Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic, *J. Symbolic Logic*, **62(2)**, pp. 457-486.

[28] Krajíček, J. (2001) On the weak pigeonhole principle, *Fundamenta Mathematicae*, Vol.**170(1-3)**, pp.123-140.

[29] Krajíček, J. (2002) Dual weak pigeonhole principle, pseudo-surjective functions, and provability of circuit lower bounds, submited, (preprint Nov. 2002).

[30] Krajíček, J. (2002) Hardness assumptions in the foundations of theoretical computer science, preprint in the ITI series (Jan.'03).

[31] Krajíček, J., and Pudlák, P. (1989a) Propositional proof systems, the consistency of first order theories and the complexity of computations, *J. Symbolic Logic*, **54(3)**:1063-1079

[32] Krajíček, J., and Pudlák, P. (1990a) Quantified propositional calculi and fragments of bounded arithmetic, *Zeitschrift f. Mathematikal Logik u. Grundlagen d. Mathematik*, **36**:29-46.

[33] Krajíček, J.,Pudlák, P. and Woods, A. (1991) Exponential lower bound to the size of bounded depth Frege proofs of the pigeonhole principle, submitted.

[34] Lovász, L., Naor, M., Newman, I, and Wigderson, A. (1991) Search problems in the decision tree model, in: *Proceedings of the 32nd IEEE Symposium on Foundations of Computer Science*, IEEE Computer Society, pp. 576-585.

[35] Mundici, D. (1983) A lower bound for the complexity of Craig's interpolants in sentential logic, *Archiv fur Math. Logik*, **23** : 27-36.

[36] Mundici, D. (1984) Tautologies with a unique Craig interpolant, uniform vs. non-uniform complexity, *Annals of Pure and Applied Logic*, **27**, pp.265-273.

[37] Mundici, D. (1984) NP and Craig's interpolation theorem, Proc. *Logic Colloquium 1982*, North-Holland, pp. 345-358.

[38] Pitassi, T., Beame, P., and Impagliazzo, R. (1993) Exponential lower bounds for the pigeonhole principle, *Computational complexity*, **3**, pp.97-308.

[39] Razborov, A. A. —— (1995) Unprovability of lower bounds on the circuit size in certain fragments of bounded arithmetic, *Izvestiya of the R.A.N.*, **59(1)**, pp.201-224.

[40] —— (2002) Pseudorandom generators hard for k-DNF resolution and polynomial calculus resolution, preprint, (Dec.'02).

[41] Reckhow, R. A. (1976) On the lengths of proofs in the propositional calculus, PhD.Thesis, Dept. of CS, University of Toronto. Technical Report No.87.

[42] —— (1993) Independence in bounded arithmetic, PhD. Thesis, Oxford University.

[43] —— (1994) $Count(q)$ does not imply $Count(p)$, preprint.

[44] Robinson, J., A. (1965) A machine-oriented logic based on the resolution principle, *Journal of the ACM*, **12(1)**, pp.23-41.

[45] Spira, P. M. (1971) On time-hardware complexity of tradeoffs for Boolean functions, in: *Proc. 4^{th} Hawaii Symp. System Sciences*, pp. 525-527.North Hollywood, Western Periodicals Co..

[46] Tseitin, G. C., and Choubarian, A. A. (1975) On some bounds to the lengths of logical proofs in classical propositional calculus" (Russian), *Trudy Vyčisl Centra AN Arm SSR i Erevanskovo Univ.*, **8**:57-64.