Proof Complexity as a Computational Lens 2025/26 Dec 11, 2025

Lecture 14

Lecturer: Jakob Nordström *Scribe: Léo Perrin, Oliver Schwarz, Add-Scribe-Here*

1 Basics of Space Complexity for Resolution

Last time we started talking about how to measure the space complexity of proofs. We focused on the resolution proof system and the measure of *clause space*, which is, essentially, the maximal number of clauses you need to keep on the whiteboard during a presentation of the resolution proof if all resolution rule applications have to be done on clauses currently on the board, and if a clause is forgotten as soon as it is erased from the board, except that we have the input formula written on a slip of paper and can copy axiom clauses to the board at any point.

In this view, a resolution refutation can be represented as a sequence $\pi = (\mathbb{C}_0, \mathbb{C}_1, \ldots, \mathbb{C}_\tau)$ of sets of clauses $\mathbb{C}_t$, or *clause configurations*, such that $\mathbb{C}_0 = \emptyset$, $\bot \in \mathbb{C}_\tau$, and every $\mathbb{C}_t$ for $t \in [\tau]$ follows from $\mathbb{C}_{t-1}$ by

- adding an axiom clause $A \in F$ to get $\mathbb{C}_t = \mathbb{C}_{t-1} \cup \{A\}$ (*axiom download*),
- applying the resolution rule to two clauses $D_1, D_2 \in \mathbb{C}_{t-1}$ to derive D and add this clause to get $\mathbb{C}_t = \mathbb{C}_{t-1} \cup \{D\}$ (*inference*), or
- removing clauses so that $\mathbb{C}_t \subseteq \mathbb{C}_{t-1}$ (*erasure*).

Assuming this *configuration-style* definition of resolution, let us recall the definitions of the space measures we considered.

Definition 1.1 (Clause space, variable space, and total space). The *clause space* of a set of clauses, or *clause configuration*, $\mathbb{C}$ is $Sp(\mathbb{C}) = |\mathbb{C}|$, i.e., the number of clauses in $\mathbb{C}$. The *variable space* of $\mathbb{C}$ is defined as $VarSp(\mathbb{C}) = |Vars(\mathbb{C})|$, i.e., counting all variables of the configuration without repetitions, and the *total space* $TotSp(\mathbb{C}) = \sum_{C \in \mathbb{C}} W(\mathbb{C})$ counts also repetitions.

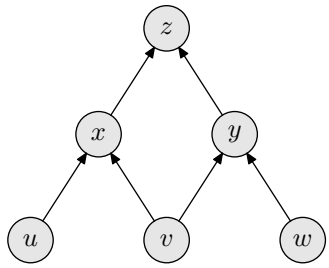
For any of these space measures M, the M-space of a resolution derivation $\pi = (\mathbb{C}_0, \mathbb{C}_1, \ldots, \mathbb{C}_\tau)$ is $M(\pi) = \max_{\mathbb{C} \in \pi}\{M(\mathbb{C})\}$ and the M-space of refuting a CNF formula F in resolution is $M_{\mathcal{R}}(F \vdash \bot) = \min_{\pi : F \vdash \bot}\{M(\pi)\}$.

As before, we omit the subscript specifying the proof system ($\mathcal{R}$ for resolution) when this is clear from context. Also, when we say just "space" below, we mean clause space, since this is the space measure that we are most interested in.

If for simplicity we fix F to be an unsatisfiable k-CNF formula over n variables, where $k = O(1)$, then the main results for resolution space that we proved last lecture can be stated as follows:

1. Refutation clause space is at most linear, i.e., $Sp(F \vdash \bot) \leq \min\{|F|, |Vars(F)|\} + O(1)$ [ET01].
2. Refutation clause space provides an upper bound on refutation width in that it always holds that $Sp(F \vdash \bot) \geq W(F \vdash \bot) + O(1)$ [AD08].
3. There are formulas such that $Sp(F \vdash \bot) = W(F \vdash \bot) = O(1)$ but where for any resolution refutation $\pi : F \vdash \bot$ it holds that $Sp(F \vdash \bot) \cdot W(F \vdash \bot) = \Omega(n/\log n)$ [Ben09].

The trade-off result for width versus clause space was obtained for formulas that encode a type of pebble games played on directed acyclic graphs (DAGs) with a single sink. In today's lecture, we will use a modified version of these formulas to investigate trade-offs between length and clause space. Let us remind ourselves how these so-called *pebbling contradictions* are defined.



(a) Pyramid graph Π_2 of height 2.

$$
\begin{aligned}
& u \\
\wedge\ & v \\
\wedge\ & w \\
\wedge\ & (\overline{u} \vee \overline{v} \vee x) \\
\wedge\ & (\overline{v} \vee \overline{w} \vee y) \\
\wedge\ & (\overline{x} \vee \overline{y} \vee z) \\
\wedge\ & \overline{z}
\end{aligned}
$$

(b) Pebbling contradiction Peb_{Π_2}.

Figure 1: Pebbling contradiction for the pyramid graph Π_2.

Definition 1.2 (Pebbling contradiction). Suppose that G is a directed acyclic graph (DAG) with a unique sink z and fan-in 2.[1] Identify every vertex $v \in V(G)$ with a propositional logic variable v. The *pebbling contradiction* (or *pebbling formula*) over G, denoted Peb_G, is the conjunction of the following clauses:

- for each source vertex s, a unit clause s (*source axioms*),
- for each non-source vertex w with immediate predecessors u and v, a clause $\overline{u} \vee \overline{v} \vee w$ (*pebbling axioms*),
- for the sink z, the unit clause $\overline{z}$ (*sink axiom*).

Note that the pebbling axioms are just encoding implications $(u \wedge v) \to w$ and thus propagate truth through the graph from the sources to the sink, which finally causes a conflict with the sink axiom $\overline{z}$. This shows that pebbling contradictions are indeed unsatisfiable, as their name suggests. For an example of a pebbling contradiction, see the CNF formula in Figure 1(b) defined in terms of the graph in Figure 1(a).

We argued last time that it is easy to see that if there is a black-pebbles-only pebbling of the DAG G in small time and space, then there is a resolution refutation that can simulate this pebbling to refute the formula Peb_G in length corresponding to the pebbling time and total space corresponding to the pebbling space. Note that this upper bound is on *total* space and thus is stronger than claiming the same bound for *clause* space. For the trade-off result in [Ben09] we needed the opposite direction, namely that if there is a resolution refutation of the formula Peb_G in small length and *variable* space, then there exists a black-white pebbling of G in small time and space. In more detail, if G is a DAG with constant fan-in and a single sink, then from any resolution refutation $\pi : Peb_G \vdash \bot$ of the pebbling contradiction over G one can extract a complete black-white pebbling $\mathcal{P}_\pi$ of G such that the pebbling time $\mathsf{time}(\mathcal{P}_\pi)$ is at most linear in the number of axiom downloads in π and the pebbling space $\mathsf{space}(\mathcal{P}_\pi)$ is at most linear in the variable space $VarSp(\pi)$.

We did not prove this theorem, since it is likely to end up on a problem set, but let us provide some intuition what the argument should be expected to go like. Given any configuration-style resolution refutation $\pi = (\mathbb{C}_0, \mathbb{C}_1, \ldots, \mathbb{C}_\tau)$, from every clause configuration $\mathbb{C}_t$ we extract a pebble configuration $\mathbb{P}_t = (B_t, W_t)$ as follows:

- If the positive literal v is present in some clause in $\mathbb{C}_t$, then place a black pebble on v, i.e., add v to B_t.
- Otherwise, if the negative literal $\overline{v}$ is present in $\mathbb{C}_t$, then place a white pebble on v, i.e., add v to W_t.

We claim that $\mathcal{P}_\pi = (\mathbb{P}_0, \mathbb{P}_1, \ldots, \mathbb{P}_\tau)$ constructed in this way is essentially a valid black-white pebbling, except for local adjustments that will not change the pebbling time or space by more than a (small) constant factor.

[1] Actually, studying pebbling contradictions makes sense for graphs of arbitrary constant fan-in, and the definition of the formulas as such works for any (unbounded but finite) fan-in, but we will only use graphs with fan-in 2 in this course.

Note that strictly speaking we will have $\mathbb{P}_\tau = (\emptyset, \emptyset)$, which is not the desired final configuration $(\{z\}, \emptyset)$ with a single black pebble on the sink z and all other vertices being empty. However, since the pebbling formula is minimally unsatisfiable (which is straightforward to verify), the pebbling axiom for the sink z has to be downloaded at some point, and at that point in time we will place a black pebble on z. We can agree to just leave that black pebble there at all ensuing steps without changing the variable space by more than an additive constant 1.

In order for this construction to work, we also need a preprocessing step for the resolution refutation that removes any "obviously stupid" derivation steps. More formally, we need to show that any resolution refutation $\pi : Peb_G \vdash \bot$ can be made to satisfy the following conditions by just removing derivation steps if needed:

- Any clause in any configuration in π (other than the final empty clause) is resolved over at least once before it is deleted.
- Any clause is deleted immediately after having been used in a resolution inference step for the last time.

Given this, one can argue (which is not hard, but perhaps requires a little bit of care) that one can go from $\mathbb{P}_{t-1}$ to $\mathbb{P}_t$ for every $t \in [\tau]$ by making a constant number of valid pebble placements or removals. From the way we construct the pebble configurations $\mathbb{P}_t$, it is clear that the pebbling space is upper-bounded by the variable space of the resolution refutation and that the number of pebble placements is bounded by a constant times the number of axiom downloads.

2 Questions Regarding Clause Space, Length, and Width in Resolution

Our review of clause space in resolution last lecture raised a number of natural questions. Let us make these questions precise, since we will be interested in discussing them in quite some detail, and then also ask similar questions for polynomial calculus and cutting planes.

We know that if a k-CNF formula is refutable in small clause space, then it is refutable in small length as well. It is natural to ask whether this holds also in the other direction.

Question 1. *We know that if a CNF formula F is refutable in resolution in length L, then it is refutable in clause space $\mathrm{O}(L \log L)$. Are larger space savings possible? If F is easy with respect to length (i.e., refutable in polynomial length), is it easy with respect to clause space (i.e., refutable in logarithmic, or even constant, clause space) as well?*

A second question that came up during last lecture is whether, given that we know that a formula can be refuted in resolution in both small length and small clause space, we can find a resolution refutation that optimizes both measures simultaneously (possibly with some small blow-up in constant factors, say).

Question 2. *If F is refutable in length L and clause space s, can it be refuted in length $\mathrm{O}(L)$ and clause space $\mathrm{O}(s)$ simultaneously?*

Thinking more closely about Question 2, there is a natural way of relaxing a bit what we are asking for. After all, we know that any formula is refutable in linear clause space, and in some cases it might be that we would be happy to get a refutation in linear space if we just knew that the refutation is short enough. Therefore, regardless of the space complexity of a formula, it would be interesting to know whether we can optimize the length of a refutation while keeping the space at most linear in the formula size $S(F)$.

Question 3. *If F is refutable in length L, can it be refuted in length $\mathrm{poly}(L)$ (i.e., with at most a polynomial blow-up) and in linear clause space $\mathrm{O}(S(F))$ (i.e., the worst-case scenario for clause space) simultaneously?*

In view of that the clause space and width complexity measure coincide for most of the benchmark combinatorial formulas that we know and love in proof complexity, and in view of the result that clause space is an upper bound on width [AD08], it is natural to ask whether these two complexity measures are somehow the same (at least asymptotically speaking) or whether they can be separated.

Question 4. *Is it true for all unsatisfiable k-CNF formulas F that width and clause space coincide asymptotically, so that $Sp(F \vdash \bot) = \Theta(W(F \vdash \bot))$, or are there formulas such that $Sp(F \vdash \bot) = \omega(W(F \vdash \bot))$, showing that these two measure are fundamentally different?*

Now, before the suspense gets unbearable, let us briefly discuss what the answers to these questions are.

With respect to Question 1, the answer is "no," and in the strongest sense possible. It was shown in [BN08] that there are maximally easy formulas with respect to length (i.e., formulas of size $S(F)$ having resolution refutations in linear length $O(S(F))$) which exhibit almost worst-case hardness with respect to space; namely they require clause space $\Omega(S(F)/\log S(F))$. Intuitively speaking, what this says is that "space complexity and length complexity of formulas are (almost) completely unrelated." These formulas also answer Question 4 in almost the strongest sense possible, since they can be refuted in constant width.

For Question 2, the answer is again "no" in a very strong sense. In the worst case, it is not possible to get even close to optimal values for both length and space, as was proven in [BN11]. Again intuitively, this is saying that "it is impossible in general to do any meaningful simultaneous optimization of length and space." We will spend this lecture and the next discussing how these answers to Questions 1, 2, and 4 can be established. At this point, it is probably not too much of a spoiler alert to say that pebbling contradictions are going to play a major role in these results.

Since any graph can always be pebbled in linear time and linear space simultaneously, however, all the formulas we get from encoding pebble games will have refutations in simultaneous linear length and linear clause space, so we will not be able to shed any light on Question 3 in this way. For this, we will instead have to study variants of Tseitin formulas defined over the right kind of (non-expanding) graphs, as done in [BBI16, BNT13]. Doing so reveals that also for Question 3 the answer is a very strong "no." There are constant-width formulas which have short refutations in polynomial (and superlinear) clause space, but where even improving this polynomial space bound a little bit (to a smaller polynomial, but still superlinear) will incur a superpolynomial length blow-up.

3 Some Promising Pebbling Formulas

To prove the results in [BN08, BN11] on separations between length and space and on length-space trade-offs, we will use the following theorems about different families of pebbling formulas. We remark that these theorems follow immediately from the pebbling properties of the DAGs in terms of which the formulas are defined (as shown in [Ben09]), and these DAGs are described in the cited references. It should also be pointed out that all these graphs are explicitly constructible—i.e., there is an efficient algorithm for actually constructing and outputting explicit descriptions of the graphs—and hence this also holds for the formulas. While this explicitness is not strictly needed to prove the proof-complexity-theoretical results, it is a nice extra bonus that we can know exactly what the formulas in question look like.

Theorem 3.1 ([PTC77, GT78]). *There are 3-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ which have resolution refutations $\pi_n : F_n \vdash \bot$ in length $L(\pi_n) = O(n)$ and width $W(\pi_n) = O(1)$ but which require variable space $VarSp(F_n \vdash \bot) = \Omega\left(\frac{n}{\log n}\right)$.*

We remark that it is important, for this family of pebbling formulas as well as for the other formulas mentioned below, that the resolution refutations providing the upper bounds can all be carried out in constant width.

Theorem 3.2 ([CS80, CS82, Nor12]). *Let $g : \mathbb{N}^+ \to \mathbb{N}^+$ be any non-constant monotone function with $\omega(1) = g(n) = O\left(n^{1/7}\right)$ and fix any $\epsilon > 0$. Then there are 3-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ such that:*

- *There are resolution refutations* $\pi_n : F_n \vdash \bot$ *in width* $W(\pi_n) = \mathrm{O}(1)$ *and total space* $TotSp(\pi_n) = \mathrm{O}\big(g(n)\big)$ *(i.e., very small space).*
- *There are also refutations* $\pi_n : F_n \vdash \bot$ *in length* $L(\pi_n) = \mathrm{O}(n)$*, width* $W(\pi_n) = \mathrm{O}(1)$*, and total space* $TotSp(\pi_n) = \mathrm{O}\left(\left(n/(g(n))^2\right)^{1/3}\right)$ *(i.e., very short refutations, but in substantially larger space).*
- *Any refutation* $\pi_n : F_n \vdash \bot$ *in variable space* $VarSp(\pi_n) = \mathrm{O}\left(\left(n/(g(n))^2\right)^{1/3-\epsilon}\right)$ *has superpolynomial length* $L(\pi_n) = n^{\omega(1)}$ *(i.e., decreasing the variable space significantly compared to the short refutations above leads to a superpolynomial blow-up in length).*

Theorem 3.2 presents a trade-off between length and total space at the low end of the space range, saying that although the total space can be made very, very small, we cannot get even close to this small space without destroying the length properties. The next theorem deals with the other end of the spectrum, i.e., space close to the linear worst-case upper bound.

Theorem 3.3 ([LT82]). *Let* κ *be a sufficiently large constant. Then there are* 3*-CNF formulas* $\{F_n\}_{n=1}^{\infty}$ *of size* $\Theta(n)$ *and a constant* $\kappa' \ll \kappa$ *such that:*

- *There are resolution refutations* $\pi_n : F_n \vdash \bot$ *in width* $W(\pi_n) = \mathrm{O}(1)$ *and total space* $TotSp(\pi_n) \leq \kappa' \frac{n}{\log n}$.
- *There are also refutations* $\pi_n : F_n \vdash \bot$ *in length* $L(\pi_n) = \mathrm{O}(n)$*, width* $W(\pi_n) = \mathrm{O}(1)$*, and linear total space* $TotSp(\pi_n) = \mathrm{O}(n)$.
- *Any refutation* $\pi_n : F_n \vdash \bot$ *in variable space* $VarSp(\pi_n) \leq \kappa \frac{n}{\log n}$ *must have exponential length* $L(\pi_n) = \exp\left(n^{\epsilon}\right)$ *for some* $\epsilon > 0$.

These theorems look promising in that they are the kind of results we have promised to prove in order to answer Questions 1 and 2. For example, the formulas of Theorem 3.1 seem like good candidates for the separation answering Question 1 in the negative. There is only one problem, however: *the results are for the wrong space measure!* We do not want trade-offs for the weak variable space measure, but for the much stronger clause space measure. That is, we would like to remove "Var" from "$VarSp$" in the statements of the theorems above to get bounds in terms of clause space "Sp."

The problem is that we cannot do this. We already know from last lecture that all pebbling formulas are in fact refutable in linear length and constant total space (and hence clause space) simultaneously. That is, pebbling formulas are super-easy with respect to length and total space simultaneously and there is no chance to get any nontrivial trade-offs this way!

4 Generalized Pebbling Contradictions

Since pebbling contradictions seemed so promising, we do not quite want to give up on them yet. We will try to make the formulas a little bit harder (but not too much harder) to get the kind of results that we want. It turns out that a good way of achieving this is to use some suitable Boolean function $f : \{0,1\}^d \to \{0,1\}$ of arity d and replace the variables in the formula by such functions. In other words, we substitute $f(x_1, \ldots, x_d)$ for each literal x and $\neg f(x_1, \ldots, x_d)$ for each literal $\overline{x}$, where $x_1, \ldots, x_d$ are new variables that do not appear in F. For brevity, we will sometimes use the shorthand $\vec{x} = x_1, \ldots, x_d$, so that $f_d(\vec{x})$ is just an equivalent way of writing $f_d(x_1, \ldots, x_d)$. After this substitution, the formula is no longer in conjunctive normal form, but since every function $f(\vec{x})$ is equivalent to a CNF formula over $x_1, \ldots, x_d$ with at most 2^d clauses, we can replace each formula with these clauses and then use De Morgan's laws to expand each original clause C to a set of clauses $C[f]$ over the new variables, and the conjunction of all of these clauses will be our new CNF formula $F[f]$.

It will be useful to formalize this concept of substitution for any CNF formula F and any Boolean function f. To this end, let f_d denote any (non-constant) Boolean function $f_d : \{0,1\}^d \to \{0,1\}$ of arity d.

Every function $f_d(x_1, \ldots, x_d)$ is equivalent to a CNF formula over $x_1, \ldots, x_d$ with at most 2^d clauses. Fix some canonical set of clauses $Cl(f_d(\vec{x}))$ representing f_d and let $Cl(\neg f_d(\vec{x}))$ denote the clauses in some chosen canonical representation of the negation of f_d. This canonical representation can be given by a formal definition (in terms of min- and maxterms), but we do not want to get too formal here and instead try to convey the intuition by providing a few examples. For instance, we have

$$Cl(\vee_2(\vec{x})) = \{x_1 \vee x_2\} \quad \text{and} \quad Cl(\neg\vee_2(\vec{x})) = \{\overline{x}_1, \overline{x}_2\} \tag{4.1}$$

for logical or of two variables and

$$Cl(\oplus_2(\vec{x})) = \{x_1 \vee x_2, \overline{x}_1 \vee \overline{x}_2\} \quad \text{and} \quad Cl(\neg\oplus_2(\vec{x})) = \{x_1 \vee \overline{x}_2, \overline{x}_1 \vee x_2\} \tag{4.2}$$

for exclusive or of two variables. If we let thr_d^k denote the threshold function saying that k out of d variables are true, then for thr_4^2 we have

$$Cl(thr_4^2(\vec{x})) = \begin{Bmatrix} x_1 \vee x_2 \vee x_3, \\ x_1 \vee x_2 \vee x_4, \\ x_1 \vee x_3 \vee x_4, \\ x_2 \vee x_3 \vee x_4 \end{Bmatrix} \quad \text{and} \quad Cl(\neg thr_4^2(\vec{x})) = \begin{Bmatrix} \overline{x}_1 \vee \overline{x}_2, \\ \overline{x}_1 \vee \overline{x}_3, \\ \overline{x}_1 \vee \overline{x}_4, \\ \overline{x}_2 \vee \overline{x}_3, \\ \overline{x}_2 \vee \overline{x}_4, \\ \overline{x}_3 \vee \overline{x}_4 \end{Bmatrix}. \tag{4.3}$$

The following observation is rather immediate, but nevertheless it might be helpful for later to state it explicitly.

Observation 4.1. *Suppose for any non-constant Boolean function f_d that $C \in Cl(f_d(x))$ and that ρ is any partial truth value assignment such that $\rho(C) = \bot$. Then for all $D \in Cl(\neg f_d(\vec{x}))$ it holds that $\rho(D) = \top$.*

Proof. If $\rho(C) = \bot$ this means that $\rho(f_d) = \bot$. Then clearly $\rho(\neg f_d) = \top$, so, in particular, ρ must fix all clauses $D \in Cl(\neg f_d(x))$ to true. □

We want to define formally what it means to substitute f_d for the variables $Vars(F)$ in a CNF formula F. For notational convenience, we assume that F only has variables x, y, z, et cetera, without subscripts, so that $x_1, \ldots, x_d, y_1, \ldots, y_d, z_1, \ldots, z_d, \ldots$ are new variables not occurring in F.

Definition 4.2 (Substitution formula). For a positive literal x and a non-constant Boolean function f_d, we define the *f_d-substitution* of x to be $x[f_d] = Cl(f_d(x))$, i.e., the canonical representation of $f_d(x_1, \ldots, x_d)$ as a CNF formula. For a negative literal $\neg y$, the f_d-substitution is $\neg y[f_d] = Cl(\neg f_d(y))$. The f_d-substitution of a clause $C = a_1 \vee \cdots \vee a_k$ is the CNF formula

$$C[f_d] = \bigwedge_{C_1 \in a_1[f_d]} \cdots \bigwedge_{C_k \in a_k[f_d]} \big(C_1 \vee \ldots \vee C_k\big) \tag{4.4}$$

and the f_d-substitution of a CNF formula F is $F[f_d] = \bigwedge_{C \in F} C[f_d]$.

Example 4.3. Let f be the exclusive or function $f(x_1, x_2) = x_1 \oplus x_2$ of arity 2. Consider the clause

$$C = \overline{x} \vee y\,. \tag{4.5}$$

Substituting $\neg f(\vec{x})$ for $\overline{x}$ and $f(\vec{y})$ for y gives us

$$\neg(x_1 \oplus x_2) \vee (y_1 \oplus y_2)\,, \tag{4.6}$$

and expanding this to CNF we get the clause set

$$C[f] = \left\{ \begin{array}{l} x_1 \vee \overline{x}_2 \vee y_1 \vee y_2 \\ x_1 \vee \overline{x}_2 \vee \overline{y}_1 \vee \overline{y}_2 \\ \overline{x}_1 \vee x_2 \vee y_1 \vee y_2 \\ \overline{x}_1 \vee x_2 \vee \overline{y}_1 \vee \overline{y}_2 \end{array} \right\} . \tag{4.7}$$

It is straightforward to verify that (4.6) holds if and only if all clauses in (4.7) are satisified. If $\neg(x_1 \oplus x_2)$ is true, x_1 and x_2 are equal, satisfying all clauses of $C[f]$. Similarly, if $(y_1 \oplus y_2)$ is true, y_1 and y_2 have distinct truth values, satisfying all clauses of $C[f]$. If none of these conditions hold, i.e., $\neg(x_1 \oplus x_2) \vee (y_1 \oplus y_2)$ is false, then one of the clauses in $C[f]$ must be falsified.

Note that $F[f_d]$ is a CNF formula over $d \cdot |\mathit{Vars}(F)|$ variables containing strictly less than $|F| \cdot 2^{d \cdot W(F)}$ clauses. (Recall that we defined a CNF formula as a set of clauses, which means that $|F|$ is the number of clauses in F.) In particular, if F has constant width, then $F[f]$ will have constant width as well, and the size of $F[f]$ will be blown up by at most a constant factor.

Two examples of substituted version of the pebbling formula in Figure 1(b) are the substitution with logical or in Figure 2(a) and with exclusive or in Figure 2(b). As we shall see soon, these formulas play an important role in the line of research trying to understand proof space in resolution.

It is an easy exercise to show that $F[f_d]$ is unsatisfiable if and only if F is unsatisfiable. Since we will always use an unsatisfiable formula F as our starting point, the substituted CNF formula $F[f]$ is also unsatisfiable, and can hence be refuted by resolution. How can we carry out such a refutation of $F[f]$ in resolution? Perhaps the first thing that comes to mind is to simply mimic a resolution refutation $\pi : F \vdash \bot$ of the original formula line by line. That is, whenever π derives a clause C, our new proof π_f derives the corresponding set of clauses $C[f]$.

What properties does a resolution refutation π_f constructed in this way have? Obviously, the length does not decrease compared to the original refutation, i.e., $L(\pi_f) \geq L(\pi)$. Looking at space, however, it is not too hard to see that there is a terrible blow-up in that we get the bound $\mathit{Sp}(\pi_f) \geq \mathit{VarSp}(\pi)$. We leave the verification of this fact to the reader, but note that Example 4.3 above shows how the two variables in (4.5) blow up to four clauses in (4.7). Thus, if we want to refute $F[f]$ in a length- and space-efficient way, we surely want to do something smarter than this naive simulation. However, the next lemma (which is slightly informally stated now, and will be made more precise later) says that it is not possible to do better.

Lemma 4.4 ([BN11] (informal)). *Fix a k-CNF formula F and let $\pi : F \vdash \bot$ and $\pi_f : F[f] \vdash \bot$ be resolution refutations that are optimal with respect to clause space and variable space, respectively. Then if we have chosen f carefully enough, the above-mentioned blow-up in space $\mathit{Sp}(\pi_f) \geq \mathit{VarSp}(\pi)$ is unavoidable.*

Note that using this lemma together with the theorems in Section 3, we get exactly the lower bounds that we are looking for to answer Questions 1 and 2. Namely, Lemma 4.4 improves the "VarSp" lower bounds to "Sp" lower bounds, just as we wanted.

Let us now present a proof idea for the lemma that will not quite work, but that will give some intuition for the formal proofs that will follow.

Proof idea for Lemma 4.4. Intuitively, we want to argue that the only way to refute $F[f]$ is to simulate a resolution refutation of F. Thus, given a resolution refutation $\pi_f = (\mathbb{D}_0, \mathbb{D}_1, \ldots, \mathbb{D}_\tau)$ of $F[f]$ (i.e., with $\mathbb{D}_0 = \emptyset$ and $\bot \in \mathbb{D}_\tau$), we want to "extract" the refutation $\pi : F \vdash \bot$, which we will denote $\pi = (\mathbb{C}_0, \mathbb{C}_1, \ldots, \mathbb{C}_\tau)$, such that π_f is mimicking.

To this end, we will have to blackboards. On one blackboard, the refutation of $F[f]$ is given to us, step by step. On the other blackboard, we will extract a refutation of the original formula F "shadowing" the refutation of $F[f]$. For concreteness, let again the substitution function f be binary XOR.

Now we look at all blackboard configurations $\mathbb{D}_t$ step by step. For each $\mathbb{D}_t$, we check what disjunctions of XORs and negated XORs is implied by this configuration, and we write down the corresponding disjunctions over the original variables of F on our shadow blackboard. For instance, if $\mathbb{D}_t$ is the clause set

$$
\begin{aligned}
&(u_1 \vee u_2) && \wedge (\overline{v}_2 \vee \overline{w}_1 \vee y_1 \vee y_2)\\
\wedge\ &(v_1 \vee v_2) && \wedge (\overline{v}_2 \vee \overline{w}_2 \vee y_1 \vee y_2)\\
\wedge\ &(w_1 \vee w_2) && \wedge (\overline{x}_1 \vee \overline{y}_1 \vee z_1 \vee z_2)\\
\wedge\ &(\overline{u}_1 \vee \overline{v}_1 \vee x_1 \vee x_2) && \wedge (\overline{x}_1 \vee \overline{y}_2 \vee z_1 \vee z_2)\\
\wedge\ &(\overline{u}_1 \vee \overline{v}_2 \vee x_1 \vee x_2) && \wedge (\overline{x}_2 \vee \overline{y}_1 \vee z_1 \vee z_2)\\
\wedge\ &(\overline{u}_2 \vee \overline{v}_1 \vee x_1 \vee x_2) && \wedge (\overline{x}_2 \vee \overline{y}_2 \vee z_1 \vee z_2)\\
\wedge\ &(\overline{u}_2 \vee \overline{v}_2 \vee x_1 \vee x_2) && \wedge \overline{z}_1\\
\wedge\ &(\overline{v}_1 \vee \overline{w}_1 \vee y_1 \vee y_2) && \wedge \overline{z}_2\\
\wedge\ &(\overline{v}_1 \vee \overline{w}_2 \vee y_1 \vee y_2)
\end{aligned}
$$

(a) Substitution pebbling contradiction $Peb_{\Pi_2}[\vee_2]$ with respect to binary logical or.

$$
\begin{aligned}
&(u_1 \vee u_2) && \wedge (v_1 \vee \overline{v}_2 \vee \overline{w}_1 \vee w_2 \vee y_1 \vee y_2)\\
\wedge\ &(\overline{u}_1 \vee \overline{u}_2) && \wedge (v_1 \vee \overline{v}_2 \vee \overline{w}_1 \vee w_2 \vee \overline{y}_1 \vee \overline{y}_2)\\
\wedge\ &(v_1 \vee v_2) && \wedge (\overline{v}_1 \vee v_2 \vee w_1 \vee \overline{w}_2 \vee y_1 \vee y_2)\\
\wedge\ &(\overline{v}_1 \vee \overline{v}_2) && \wedge (\overline{v}_1 \vee v_2 \vee w_1 \vee \overline{w}_2 \vee \overline{y}_1 \vee \overline{y}_2)\\
\wedge\ &(w_1 \vee w_2) && \wedge (\overline{v}_1 \vee v_2 \vee \overline{w}_1 \vee w_2 \vee y_1 \vee y_2)\\
\wedge\ &(\overline{w}_1 \vee \overline{w}_2) && \wedge (\overline{v}_1 \vee v_2 \vee \overline{w}_1 \vee w_2 \vee \overline{y}_1 \vee \overline{y}_2)\\
\wedge\ &(u_1 \vee \overline{u}_2 \vee v_1 \vee \overline{v}_2 \vee x_1 \vee x_2) && \wedge (x_1 \vee \overline{x}_2 \vee y_1 \vee \overline{y}_2 \vee z_1 \vee z_2)\\
\wedge\ &(u_1 \vee \overline{u}_2 \vee v_1 \vee \overline{v}_2 \vee \overline{x}_1 \vee \overline{x}_2) && \wedge (x_1 \vee \overline{x}_2 \vee y_1 \vee \overline{y}_2 \vee \overline{z}_1 \vee \overline{z}_2)\\
\wedge\ &(u_1 \vee \overline{u}_2 \vee \overline{v}_1 \vee v_2 \vee x_1 \vee x_2) && \wedge (x_1 \vee \overline{x}_2 \vee \overline{y}_1 \vee y_2 \vee z_1 \vee z_2)\\
\wedge\ &(u_1 \vee \overline{u}_2 \vee \overline{v}_1 \vee v_2 \vee \overline{x}_1 \vee \overline{x}_2) && \wedge (x_1 \vee \overline{x}_2 \vee \overline{y}_1 \vee y_2 \vee \overline{z}_1 \vee \overline{z}_2)\\
\wedge\ &(\overline{u}_1 \vee u_2 \vee v_1 \vee \overline{v}_2 \vee x_1 \vee x_2) && \wedge (\overline{x}_1 \vee x_2 \vee y_1 \vee \overline{y}_2 \vee z_1 \vee z_2)\\
\wedge\ &(\overline{u}_1 \vee u_2 \vee v_1 \vee \overline{v}_2 \vee \overline{x}_1 \vee \overline{x}_2) && \wedge (\overline{x}_1 \vee x_2 \vee y_1 \vee \overline{y}_2 \vee \overline{z}_1 \vee \overline{z}_2)\\
\wedge\ &(\overline{u}_1 \vee u_2 \vee \overline{v}_1 \vee v_2 \vee x_1 \vee x_2) && \wedge (\overline{x}_1 \vee x_2 \vee \overline{y}_1 \vee y_2 \vee z_1 \vee z_2)\\
\wedge\ &(\overline{u}_1 \vee u_2 \vee \overline{v}_1 \vee v_2 \vee \overline{x}_1 \vee \overline{x}_2) && \wedge (\overline{x}_1 \vee x_2 \vee \overline{y}_1 \vee y_2 \vee \overline{z}_1 \vee \overline{z}_2)\\
\wedge\ &(v_1 \vee \overline{v}_2 \vee w_1 \vee \overline{w}_2 \vee y_1 \vee y_2) && \wedge (z_1 \vee \overline{z}_2)\\
\wedge\ &(v_1 \vee \overline{v}_2 \vee w_1 \vee \overline{w}_2 \vee \overline{y}_1 \vee \overline{y}_2) && \wedge (\overline{z}_1 \vee z_2)
\end{aligned}
$$

(b) Substitution pebbling contradiction $Peb_{\Pi_2}[\oplus_2]$ with respect to binary exclusive or.

Figure 2: Examples of substitution pebbling formulas for the pyramid graph Π_2.

in (4.7), then $\mathbb{D}_t$ clearly implies $\neg(x_1 \oplus x_2) \vee (y_1 \oplus y_2)$ and so we write down $\overline{x} \vee y$ on our shadow blackboard. We do this for all such impliciations, and in this way we translate $\mathbb{D}_t$ into a clause configuration $\mathbb{C}_t$ over the original variables.

We want to argue that if we translate each configuration $\mathbb{D}_t$ derived from $F[f]$ in this way into a clause configuration $\mathbb{C}_t$ over the variables of F, then $\pi = (\mathbb{C}_0, \mathbb{C}_1, \ldots, \mathbb{C}_\tau)$ is a resolution refutation of F and that the length and variable space of the refutation π on the shadow blackboard is upper-bounded by the length and clause space of the given refutation π_f, respectively. □

If this would pan out, the lemma would follow. Unfortunately, this does not quite work, but we will prove something similar that will give us the same result in the end. This will require us to set up some machinery, however. Before we even start doing this, we also need to show that the substituted formulas are not *too* hard. That is, we need to show that the upper bounds in Section 3 still hold for these formulas. This is the next lemma.

Lemma 4.5 ([BN11]). *Suppose F is an unsatisfiable CNF formula and $f : \{0,1\}^d \to \{0,1\}$ is a non-constant Boolean function. If there is a resolution refutation $\pi : F \vdash \bot$ in length $L(\pi) = L$, total space $TotSp(\pi) = s$, and width $W(\pi) = w$, then there is also a resolution refutation $\pi_f : F[f] \vdash \bot$ of the substituted formula $F[f]$ in length $L(\pi_f) = L \cdot \exp(\mathrm{O}(dw))$, total space $TotSp(\pi_f) = s \cdot \exp(\mathrm{O}(dw))$, and width $W(\pi_f) = \mathrm{O}(dw)$.*

In particular, if the refutation $\pi : F \vdash \bot$ has constant width, then it is possible to refute $F[f]$ with only a constant factor blow-up in length and space as compared to π (where this constant depends on $W(\pi)$ and f). We remark that the same statement holds true for any sequential proof system that can simulate resolution proofs sufficiently efficiently line by line, such as, for instance, cutting planes or PCR.)

Proof sketch for Lemma 4.5. The proof is straighforward, but somewhat tedious, and we will only give the general outline. We simulate π step by step in "the obvious way", making sure that if the current configuration in π is $\mathbb{C}$, then π_f has derived the clauses $\{C[f] \mid C \in \mathbb{C}\}$. If π downloads an axiom C, we let π_f download all axioms in $C[f]$. Those are at most $\exp(\mathrm{O}(d \cdot W(C)))$ many clauses. If π resolves $C_1 \vee x$ and $C_2 \vee \overline{x}$ to derive $C_1 \vee C_2$, we let π_f derive $(C_1 \vee C_2)[f]$ from $(C_1 \vee x)[f]$ and $(C_2 \vee \overline{x})[f]$ (which can be done by the implicational completeness of resolution). When a clause C is erased from the board by π, then π_f erases all the clauses in $C[f]$. The details can easily be verified by the reader, or can be looked up in [BN11]. □

5 Projections

Let us now return to the proof of Lemma 4.4. Our idea is that we want to extract a refutation of F from any refutation of $F[f]$. In what follows, we will change this terminology slightly and think of a refutation $\pi_f : F[f] \vdash \bot$ as "projecting" a refutation $\pi : F \vdash \bot$ of the original formula, where we want to "project" any clause configuration $\mathbb{D} \in \pi_f$ derived from $F[f]$ to a clause configuration $\mathbb{C}$ derived from F. As we described before, our intuition for projections is that if, for instance, $\mathbb{D}$ implies $\neg f(\vec{x}) \vee f(\vec{y})$, then this should project the clause $\overline{x} \vee y$. It will be useful, however, to relax this requirement a bit in order to not over-specify and allow other definitions of projections as well as long as these definitions are "in the same spirit." In the next definition, we specify which formal properties a projection must satisfy in order for our approach to work.

As a technical note, let us remark that in what follows we do not distinguish between the set of clauses $C[f]$ (which is just a syntactic object) and the Boolean function that is encoded by the conjunction of all clauses in $C[f]$ (which is a mathematical function). Also, it will be convenient to define some notation. For sets of clauses $\mathbb{C}$ and $\mathbb{D}$ (which, as usual, are identified with the CNF formulas consisting of the conjunctions of all the clauses), we define $\mathbb{C} \vee \mathbb{D} = \{C \vee D \mid C \in \mathbb{C},\ D \in \mathbb{D}\}$. Then for substitution in clauses it holds that $(C \vee a)[f] = C[f] \vee a[f]$ (this is easy to verify just on a syntactical level). For a set of variables $V = \{x, y, z, \ldots\}$, we let

$$Vars^d(V) = \{x_1, x_2, \ldots, x_d, y_1, y_2, \ldots, y_d, z_1, z_2, \ldots, z_d, \ldots\} \tag{5.1}$$

denote the variables after substitution (which we assume are disjoint from the variables in V). Now we can define what we mean by a *projection*.

Definition 5.1 (Projection). Assume that $f : \{0,1\}^d \to \{0,1\}$ is a fixed Boolean function, $\mathcal{P}$ is a sequential proof system, $\mathbb{D}$ denotes some arbitrary set of Boolean functions over $Vars^d(V)$ of the syntactic form specified by $\mathcal{P}$, and $\mathbb{C}$ denotes arbitrary sets of disjunctive clauses over V. Then the function proj_f mapping sets of Boolean functions $\mathbb{D}$ over $Vars^d(V)$ to clauses $\mathbb{C}$ over V is an *f-projection* if it is:

Complete: If $\mathbb{D} \models C[f]$, then there is a $C' \subseteq C$ such that $C' \in \mathsf{proj}_f(\mathbb{D})$ (i.e., the clause C either is in $\mathsf{proj}_f(\mathbb{D})$ or is derivable from $\mathsf{proj}_f(\mathbb{D})$ by weakening).

Nontrivial: If $\mathbb{D} = \emptyset$, then $\mathsf{proj}_f(\mathbb{D}) = \emptyset$.

Monotone: If $\mathbb{D}' \models \mathbb{D}$ and $C \in \mathsf{proj}_f(\mathbb{D})$, then there is a $C' \subseteq C$ such that $C' \in \mathsf{proj}_f(\mathbb{D}')$.

Incrementally sound: Let A be a clause over V and let L_A be the encoding of some clause in $A[f]$ as a Boolean function of the type prescribed by $\mathcal{P}$. Then if $C \in \mathsf{proj}_f(\mathbb{D} \cup \{L_A\})$, it holds for all literals $a \in Lit(A) \setminus Lit(C)$ that $\overline{a} \vee C \supseteq C_a \in \mathsf{proj}_f(\mathbb{D})$.

Notice that we have kept the definition general enough to work for any sequential proof system. For any such proof system, we can show that any projection in the sense of Definition 5.1 can be used to extract resolution refutations from $\mathcal{P}$-refutations in a sense that is made precise in the following lemma.

Lemma 5.2. *Let $\mathcal{P}$ be a sequential proof system and $f : \{0,1\}^d \to \{0,1\}$ a Boolean function, and suppose that* proj *is an f-projection. Then for any CNF formula F it holds that if $\pi_f = (\mathbb{D}_0, \mathbb{D}_1, \ldots, \mathbb{D}_\tau)$ is a $\mathcal{P}$-refutation of the substitution formula $F[f]$, then the sequence $\big(\mathsf{proj}(\mathbb{D}_0), \mathsf{proj}(\mathbb{D}_1), \ldots, \mathsf{proj}(\mathbb{D}_\tau)\big)$ of sets of projected clauses form the "backbone" of a resolution refutation π of F in the following sense:*

1. *We start with an empty blackboard:* $\mathsf{proj}(\mathbb{D}_0) = \emptyset$.
2. *We end with contradiction:* $\bot \in \mathsf{proj}(\mathbb{D}_\tau)$.
3. *All transitions from* $\mathsf{proj}(\mathbb{D}_{t-1})$ *to* $\mathsf{proj}(\mathbb{D}_t)$ *for time $t \in [\tau]$ can be accomplished in resolution in such a fashion that $VarSp(\pi) = \mathrm{O}\big(\max_{\mathbb{D} \in \pi_f}\{VarSp(\mathsf{proj}(\mathbb{D}))\}\big)$.*
4. *The length of π is upper-bounded by π_f in the sense that the only time π does a download of $C \in F$ is when π_f downloads some axiom $L_C \in C[f]$ from $F[f]$.*

Intuitively speaking, Lemma 5.2 says that we can consider the projected clause configurations $\mathsf{proj}(\mathbb{D}_i)$ as "snapshots" of a resolution refutation π of F. These snapshots capture the "interesting part" of the refutation π in that "nothing important" happens in between the snapshots, in particular with regards to variable space. This goes a long way towards proving what we need to get our length-space separation and trade-offs. However, there is one key component missing, namely the connection between the space measures of π_f and π in Lemma 4.4.

6 How to Use Projections to Prove Space Lower Bounds

Note that Lemma 5.2 as stated holds for any sequential proof system $\mathcal{P}$. We will soon have to restrict our attention to $\mathcal{P}$ being resolution for the proofs to work, but for as long as it is possible we will try to keep the discussion general (because we would like to extend this framework to stronger proof systems such as polynomial calculus resolution).

In order for Lemma 5.2 to be really useful, we would like the projection to somehow preserve space when it is projecting derivations in the proof system $\mathcal{P}$ to resolution derivations. These considerations lead us to the concept of *space-faithfulness* as defined next.

Definition 6.1 (Space-faithful projection). Consider a sequential proof system $\mathcal{P}$ with space measure $Sp_{\mathcal{P}}(\cdot)$. Let $f : \{0,1\}^d \to \{0,1\}$ be a fixed Boolean function, and suppose that proj_f is an f-projection. Then we say that proj_f is *space-faithful of degree K with respect to $\mathcal{P}$* if there is a polynomial Q of degree at most K such that for any set of Boolean functions $\mathbb{D}$ over $Vars^d(V)$ on the form prescribed by $\mathcal{P}$, it holds that $Q(Sp_{\mathcal{P}}(\mathbb{D})) \geq \left| Vars(\mathsf{proj}_f(\mathbb{D})) \right|$. We say that proj_f is *linearly space-faithful* if Q has degree 1, and that proj_f is *exactly space-faithful* if we can choose $Q(n) = n$.

The way to read Definition 6.1 is that the smaller the degree K is, the tighter the reduction between the proof system $\mathcal{P}$ and resolution will be with respect to space.

Before proving Lemma 5.2, let us see how it can be used to prove trade-offs provided that we can construct space-faithful projections. As stated in the introduction, proving such trade-offs is the very reason we set up all this mathematical machinery.

Theorem 6.2. *Let $\mathcal{P}$ be a sequential proof system with space measure $Sp_{\mathcal{P}}(\cdot)$. Suppose that the Boolean function $f : \{0,1\}^d \to \{0,1\}$ is such that there exists an f-projection which is space-faithful of degree K with respect to $\mathcal{P}$. Then if F is any unsatisfiable CNF formula and π_f is any $\mathcal{P}$-refutation of the substitution formula $F[f]$, there is a resolution refutation π of F such that:*

- *The length of π is upper-bounded by π_f in the sense that π makes at most as many axiom downloads as π_f.*
- *The space of π is upper-bounded by π_f in the sense that $VarSp(\pi) = \mathrm{O}\big(Sp_{\mathcal{P}}(\pi_f)^K\big)$.*

In particular, if there is no resolution refutation of F in variable space $\mathrm{O}(s)$ that does $\mathrm{O}(L)$ axioms downloads, then there is no $\mathcal{P}$-refutation of $F[f]$ in simultaneous length $\mathrm{O}(L)$ and $\mathcal{P}$-space $\mathrm{O}\big(\sqrt[K]{s}\big)$.

Proof. First, note that the first bullet in the theorem is a direct consequence of Lemma 5.2 (namely part 4 of this lemma) . Thus, we only need to prove the second bullet here.

Let π_f be a $\mathcal{P}$-refutation of $F[f]$, and let π be the resolution refutation we obtain by applying the projection proj on π_f as in Lemma 5.2. By part 3 of the lemma we have

$$VarSp(\pi) = \mathrm{O}\left(\max_{\mathbb{D}\in\pi_f}\{\, VarSp(\mathsf{proj}(\mathbb{D}))\}\right) . \tag{6.1}$$

Fix some $\mathcal{P}$-configuration $\mathbb{D}$ maximizing the right-hand side of (6.1). For this $\mathbb{D} \in \pi$ we have

$$VarSp(\mathsf{proj}(\mathbb{D})) = \mathrm{O}\big(Sp_{\mathcal{P}}(\mathbb{D})^K\big) \tag{6.2}$$

according to Definition 6.1, and since obviously $Sp(\mathbb{D}) \leq \max_{\mathbb{D}\in\pi_f}\{Sp(\mathbb{D})\} = Sp\big(\pi_f\big)$ we get by combining (6.1) and (6.2) that

$$VarSp(\pi) \leq \mathrm{O}\left(Sp\big(\pi_f\big)^K\right) \tag{6.3}$$

and the theorem follows. □

We now turn to the proof of Lemma 5.2. We will give a fairly detailed outline of the proof, but for the complete details we refer to Appendix A.

Proof sketch for Lemma 5.2. Suppose that $\pi_f = (\mathbb{D}_0, \mathbb{D}_1, \ldots, \mathbb{D}_\tau)$ is a $\mathcal{P}$-refutation of $F[f]$. In this proof, for the sake of conciseness let us write $\mathbb{C}_t$ to denote $\mathsf{proj}(\mathbb{D}_t)$. We prove the parts of the lemma one by one.

1. The fact that $\mathbb{C}_0 = \emptyset$ is a direct consequence of the nontriviality of the projection.
2. $\bot \in \mathbb{C}_\tau$ is also implied immediately by the definition of a projection, namely by the completeness condition.

3. So far, not too much interesting stuff happened in the proof, but this part is more interesting. We need to show that if we can derive $\mathbb{D}_t$ from $\mathbb{D}_{t-1}$ using our favorite proof system $\mathcal{P}$, then we can derive $\mathbb{C}_t$ from $\mathbb{C}_{t-1}$ in resolution. Note that we only need to worry about new clauses in $\mathbb{C}_t \setminus \mathbb{C}_{t-1}$ here—clauses in $\mathbb{C}_{t-1} \setminus \mathbb{C}_t$ that disappear at time t can just be erased and will not cause us any problems. We have three cases for the derivation step at time t:

   Inference: Any $C \in \mathbb{C}_t \setminus \mathbb{C}_{t-1}$ can be derived from $\mathbb{C}_{t-1}$ by weakening as proj is monotone.

   Erasure: Again, monotonicity allows us to derive any new clauses in $\mathbb{C}_t$ from $\mathbb{C}_{t-1}$ by weakening.

   Axiom download: Let C be a clause in $\mathbb{C}_t \setminus \mathbb{C}_{t-1}$ for $\mathbb{C}_t = \mathsf{proj}_f(\mathbb{D}_{t-1} \cup \{L_A\})$, where L_A is an encoding in $\mathcal{P}$ of some clause in $A[f]$ downloaded at this time step. The incremental soundness of projections tells us that for all literals $a \in Lit(A) \setminus Lit(C)$ we have some $C_a \in \mathbb{C}_{t-1}$ such that $C_a \subseteq \overline{a} \vee C$. This means that we can derive $\overline{a} \vee C$ by weakening from $\mathbb{C}_{t-1}$ for all $a \in Lit(A) \setminus Lit(C)$. Then we download the axiom clause A in our resolution derivation and resolve with all these clauses one by one to derive C. □

This proof is only a sketch because some things were swept under the rug. For instance, we did not look at the variable space. While we know its values for the projected clause configurations $\mathbb{C}_t$, what happens in between these configurations? Who knows, maybe the space treacherously increases a lot after $\mathbb{C}_t$ before decreasing again to reach the known value at $\mathbb{C}_{t+1}$, pretending nothing happened? Actually, as shown in Appendix A, this cannot happen, but the proof sketch provided above does not argue why this is so.

7 Designing Projections for Resolution

What Theorem 6.2 says is the following: suppose we have a family of CNF formulas with lower bounds for refutation variable space in resolution, or with trade-offs between refutation length (or, more precisely, number of axiom downloads) and refutation variable space (such as for instance pebbling contradictions over suitable graphs, as we discussed last lecture). Then we can amplify these lower bounds or trade-offs to stronger proof complexity measures in a potentially stronger proof system $\mathcal{P}$, provided that we can find a Boolean function f and an f-projection proj that is space-faithful with respect to $\mathcal{P}$.

Thus, at this point we can in principle forget everything about proof complexity. If we want to prove space lower bounds or time-space trade-offs for some proof system $\mathcal{P}$, we can focus on studying Boolean functions of the form used by $\mathcal{P}$ and trying to devise space-faithful projections for such functions. For the rest of this lecture, we will let $\mathcal{P}$ be resolution. What will be said below works in slightly greater generality, however, namely for the stronger family of k-DNF resolution proof systems. Although we did not have time to cover these results in class, we discuss them briefly in Appendix B.

To describe the projections we will study, we need a definition.

Definition 7.1 (Precise implication [BN11]). Let f be a Boolean function of arity d, let $\mathbb{D}$ be a set of Boolean functions over $Vars^d(V)$, and let C be a disjunctive clause over V. If

$$\mathbb{D} \models C[f] \tag{7.1a}$$

but for all strict subclauses $C' \subsetneqq C$ it holds that

$$\mathbb{D} \not\models C'[f]\,, \tag{7.1b}$$

we say that the clause set $\mathbb{D}$ implies $C[f]$ *precisely* and write

$$\mathbb{D} \rhd C[f]\,. \tag{7.2}$$

Informally speaking, a precise implication of some Boolean function means that precisely this function is implied but nothing stronger. We want to project clauses that correspond to such precise implications as described next.

Definition 7.2 (Resolution projection [BN11]). Let f denote a Boolean function of arity d and let $\mathbb{D}$ be any set of Boolean functions over $Vars^d(V)$. Then we define the set of clauses

$$\mathsf{Rproj}(\mathbb{D}) = \{C \mid \mathbb{D} \rhd C[f]\} \tag{7.3}$$

to be the *resolution projection* of $\mathbb{D}$.

What we want to prove now is that for $\mathcal{P}$ being resolution, and for the right choice of Boolean function f, Rproj is a linearly space-faithful projection. Then we can apply Theorem 6.2 to the CNF formula families we discussed at the beginning of last lecture, and this will give us the results we are after. Let us start by showing that Rproj is a projection.

Lemma 7.3. *The mapping* Rproj *is an f-projection (for any sequential proof system $\mathcal{P}$).*

Proof. The proof of this lemma is very straightforward. All we have to do is to check each condition Rproj must satisfy in order to be an f-projection.

Nontriviality is obvious. An empty $\mathcal{P}$-configuration evaluates to true under all assignments, and so cannot project any clauses according to Definition 7.2. Thus, $\mathsf{Rproj}(\emptyset) = \emptyset$.

Suppose $\mathbb{D} \rhd C[f]$ and $\mathbb{D}' \vDash \mathbb{D}$. Then we can remove literals from C one by one until we get a new clause C' which is minimal with respect to the property that $\mathbb{D}' \vDash C'[f]$. By definition, we then have that $\mathbb{D}' \rhd C'[f]$, i.e., $C' \in \mathsf{Rproj}(\mathbb{D}')$. This shows that Rproj satisfies both *completeness* and *monotonicity*.

It remains to consider the *incremental soundness*. Suppose, using the notation introduced above, that $\mathbb{D} \cup \{L_A\} \vDash C[f]$. Consider a truth value assignment α such that $\alpha(\mathbb{D}) = 1$ and $\alpha(C[f]) = 0$. If no such α exists, then simply by parsing what semantic implication $\vDash$ means we get that $\mathbb{D} \vDash C[f]$ and we have already seen how C can be derived from $\mathsf{Rproj}(\mathbb{D})$ by weakening in this case. It would be great if this were always the case, because if so we would now be done with the proof, but as conscientious mathematicians we must also consider the other case.

Thus, suppose there exists such an α. This α must falsify $L_A \in A[f]$ or we immediately get a contradiction. If we write $A = \bigvee_{i=1}^{k} a_i$, we have that $A[f] = a_1[f] \vee a_2[f] \vee \cdots \vee a_k[f]$ and α must falsify every $a_i[f]$ in order to falsify L_A. Turning the tables, this means that $\alpha\big(\overline{a}_i[f]\big) = 1$ for all $i = 1, \ldots, k$. From this we can deduce that if α satisfies $\mathbb{D}$, then this assignment must also satisfy $\overline{a_i}[f] \vee C[f]$ for all $i = 1, \ldots, k$. The incremental soundness follows. □

In order to get a space-faithful projection Rproj, we pick a Boolean function f as in the next definition.

Definition 7.4 (Non-authoritarian function [BN11]). We say that a Boolean function $f(x_1, \ldots, x)$ is *k-non-authoritarian*[2] if no restriction to $\{x_1, \ldots, x_d\}$ of size k can fix the value of f. In other words, for every restriction ρ to $\{x_1, \ldots, x_d\}$ with $|\rho| \leq k$ there exists two assignments $\alpha_0, \alpha_1 \supset \rho$ such that $f(\alpha_0) = 0$ and $f(\alpha_1) = 1$. If this does not hold, f is *k-authoritarian*. A 1-(non-)authoritarian function is called just *(non-)authoritarian*.

Observe that a function on d variables can be k-non-authoritarian only if $k < d$. Perhaps the most natural example of a k-non-authoritarian function is exclusive or $\oplus_{k+1}$ of $k+1$ variables. Indeed, even if you fix any k variables in this function to any values, you can still flip the function to both true and false by choosing the right value of the final $(k+1)$st variable. One could say that in such a function, every variable has its say in deciding the final result. That is, as Boolean functions go, it seems to be somewhat democratic, or at least not authoritarian. Hence the terminology in Definition 7.4.

Now we are finally ready to state the main technical result, which will allow us to prove the space lower bounds and trade-offs that we are after: picking f non-authoritarian is enough to guarantee that Rproj is linearly space-faithful for resolution.

Theorem 7.5 ([BN11]). *If f is a non-authoritarian Boolean function, then the projection* Rproj *is linearly space-faithful with respect to the resolution proof system.*

[2] Such functions have also been referred to as $(k+1)$-*robust* in [ABRW02].

Proof. In the case of resolution, the set of Boolean functions $\mathbb{D}$ just consists of disjunctive clauses over $Vars^d(V)$. Fix an arbitrary such clause set $\mathbb{D}$ and let $V^* = Vars\big(\mathsf{Rproj}(\mathbb{D})\big)$. What we want to prove is that $|\mathbb{D}| \geq |V^*|$. We assume without loss of generality that $\mathbb{D}$ is a minimal clause set such that $Vars\big(\mathsf{Rproj}(\mathbb{D})\big) = V^*$, since otherwise we can just remove clauses from $\mathbb{D}$ until we get such a set.

Consider a bipartite graph with the vertices on the left labelled by clauses $D \in \mathbb{D}$ and the vertices on the right labelled by variables $x \in V^*$. We draw an edge between D and x if some variable x_i from $Vars^d(x)$ appears in D. Let $N(\mathbb{D}')$ denote the neighbours on the right of a clause set $\mathbb{D}'$. We claim without proof that $N(\mathbb{D}) = V^*$, i.e., that all $x \in V^*$ have incoming edges from $\mathbb{D}$ (establishing this claim is left as a useful but not too hard exercise in juggling with the definitions in this lecture).

Fix some $\mathbb{D}_1 \subseteq \mathbb{D}$ of maximal size with neighbour set $V_1^* = N(\mathbb{D}_1)$ such that $|\mathbb{D}_1| \geq |V_1^*|$ (and note that such a set always exist, since $\mathbb{D}_1 = \emptyset$ is allowed). If $\mathbb{D}_1 = \mathbb{D}$ we are done, since if so $|\mathbb{D}| \geq |V^*|$ which is exactly what we want to prove. Suppose therefore that $\mathbb{D}_1 \neq \mathbb{D}$ and let us argue by contradiction. Let $\mathbb{D}_2 = \mathbb{D} \setminus \mathbb{D}_1 \neq \emptyset$ and $V_2^* = V^* \setminus V_1^*$. For all $\mathbb{D}' \subseteq \mathbb{D}_2$ we must have $|\mathbb{D}'| \leq |N(\mathbb{D}') \setminus V_1^*| = |N(\mathbb{D}') \cap V_2^*|$, since otherwise we could have added $\mathbb{D}'$ to $\mathbb{D}_1$ and so this latter set would not have been chosen of maximal size. This in turns implies by Hall's theorem that there is a matching M from $\mathbb{D}_2$ into V_2^*.

Consider some clause $C \in \mathsf{Rproj}(\mathbb{D}) \setminus \mathsf{Rproj}(\mathbb{D}_1)$ such that $\mathbb{D}_1$ is "too weak" to project C (such a clause exists by the minimality of $\mathbb{D}$). Let C_i be the part of C that mentions variables from V_i^* for $i = 1, 2$. Then by Definitions 7.1 and 7.2 it holds that $\mathbb{D}_1 \cup \mathbb{D}_2 \models C_1[f] \vee C_2[f]$ but $\mathbb{D}_1 \not\models C_1[f]$. This means that there is a truth value assignment α_1 to $Vars^d(V_1^*)$ satisfying $\mathbb{D}_1$ but falsifying $C_1[f]$. Observe that $Vars(\mathbb{D}_1) \subseteq Vars^d(V_1^*)$ by construction and that

Using the matching M, we can find another partial truth value assignment α_2 to $Vars^d(V_2^*)$ that satisfies all clauses $D \in \mathbb{D}_2$ by setting at most one variable x_i for every $x \in V_2^*$. In particular, this means that α_2 does not determine the truth value of $C_2[f]$ since f is non-authoritarian, and this in turn means that we can extend α_2 to a full assignment over $Vars^d(V_2^*)$ such that $C_2[f]$ is falsified.

Note now that $\alpha_1 \cup \alpha_2$ is a legal truth value assignment, since α_1 and α_2 are assigning two disjoint sets of variables, namely $Vars^d(V_1^*)$ and $Vars^d(V_2^*)$. Therefore, they cannot "overlap" in such a way that there is some variable assigned to 1 by one assignment and to 0 by the other. It follows from what was argued above that this assignment $\alpha_1 \cup \alpha_2$ satisfies $\mathbb{D}_1 \cup \mathbb{D}_2$ but falsifies $C_1[f] \vee C_2[f]$, which is a contradiction. □

8 Space Lower Bounds and Length-Space Trade-offs for Resolution

Instructor's comment 1: *Note that our lower bounds and trade-offs holds for* semantic resolution*, where anything implied from $\mathbb{C}_t$ can be derived and added in a single step. In this model, any CNF formula F is refutable in simultaneous length and clause space $|F| + \mathrm{O}(1)$. This means that the trade-off results are really strong! But this is also a weakness of the technique, because it shows that we could never prove trade-offs where clause space becomes superlinear.*

Now the three theorems we discussed at the beginning of the lecture follow with clause space instead of variable space in the bounds, just as we wanted. Let us conclude by stating these theorems (where for concreteness we choose the substitution function to be binary exclusive or, which after substitution gives us 6-CNF formulas).

Theorem 8.1 ([BN08]). *There is a family of explicitly constructible 6-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ which have resolution refutation length $L_{\mathcal{R}}(F_n \vdash \bot) = \mathrm{O}(n)$ and refutation width $W_{\mathcal{R}}(F_n \vdash \bot) = \mathrm{O}(1)$ but require clause space $Sp_{\mathcal{R}}(F_n \vdash \bot) = \Omega\big(\frac{n}{\log n}\big)$.*

That is, Theorem 8.1 tells us that although small space complexity implies that a formula is easy also with respect to length and width in resolution, the opposite does not hold: a formula can be maximally easy with respect to both length and width but still have essentially worst-case clause space complexity. As we have discussed before, by combining [ET01] and [HPV77] it is straightforward to show that if a formula is refutable in length $\mathrm{O}(n)$, then it is also refutable in clause space $\mathrm{O}(n/\log n)$, so the bound in Theorem 8.1 is optimal for formulas refutable in linear length.

The next two theorems give examples of trade-offs between proof length and proof space. These are just two particular instances of theorems that can be proven with this machinery—we refer to [BN11] for more examples.

Theorem 8.2 ([BN11, Nor12]). *Let $g : \mathbb{N}^+ \to \mathbb{N}^+$ be any non-constant monotone function with $\omega(1) = g(n) = \mathrm{O}\big(n^{1/7}\big)$ and fix any $\epsilon > 0$. Then there is a family of explicitly constructible 6-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ such that:*

- *The total space of refuting F_n is $TotSp_{\mathcal{R}}(F_n \vdash \bot) = \mathrm{O}\big(g(n)\big)$ (i.e., very small space).*
- *There are resolution refutations $\pi_n : F_n \vdash \bot$ with length $L(\pi_n) = \mathrm{O}(n)$ and total space $TotSp(\pi_n) = \mathrm{O}\left(\left(n/(g(n))^2\right)^{1/3}\right)$ (i.e., very short refutations, but in substantially larger space).*
- *Any refutation $\pi_n : F_n \vdash \bot$ in clause space $Sp(\pi_n) = \mathrm{O}\left(\left(n/(g(n))^2\right)^{1/3-\epsilon}\right)$ has superpolynomial length $L(\pi_n) = n^{\omega(1)}$ (i.e., decreasing the clause space significantly compared to the short refutations above leads to a superpolynomial blow-up in length).*

Theorem 8.3 ([BN11]). *Let κ be a sufficiently large constant. Then there is a family of explicitly constructible 6-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ and a constant $\kappa' \ll \kappa$ such that:*

- *The total space of refuting F_n is bounded by $TotSp_{\mathcal{R}}(F_n \vdash \bot) \leq \kappa' \frac{n}{\log n}$.*
- *There are resolution refutations $\pi_n : F_n \vdash \bot$ with length $L(\pi_n) = \mathrm{O}(n)$ and linear total space $TotSp(\pi_n) = \mathrm{O}(n)$.*
- *Any refutation $\pi_n : F_n \vdash \bot$ in clause space $Sp(\pi_n) \leq \kappa \frac{n}{\log n}$ must have exponential length $L(\pi_n) = \exp\left(n^{\epsilon}\right)$ for some $\epsilon > 0$.*

A very interesting question is whether results similar to the ones above could be proven for other strictly stronger proof systems such as polynomial calculus, polynomial calculus resolution, or cutting planes. It can be shown that the projection in Definition 7.2 will not work for these proof systems (i.e., it is not space-faithful) and although we did not have time to cover this in class we discuss this briefly in Appendix B.

For polynomial calculus, it would still seem to make sense to ask if something along the lines of what we have covered in this lecture could be made to work.

Open Problem 5. *Is it possible to prove space lower bounds and/or tight trade-offs between proof length/size and space for polynomial calculus or polynomial calculus resolution by designing smarter projections than in Definition 7.2 that are space-faithful for these proof systems?*

Disregarding the general framework presented in this lecture, it would be extremely interesting (at least to the lecturer) if we could prove some kind of monomial space lower bounds for polynomial calculus refutations (or PCR refutations) of pebbling contradictions by any means whatsoever.

Open Problem 6. *Is it true for generalized pebbling formulas $Peb_G[f]$ for some well-chosen f (such as binary exclusive or, ternary majority, or whatever) that the monomial space of refuting $Peb_G[f]$ in polynomial calculus (or even polynomial calculus resolution) is lower-bounded by $MSp(Peb_G[f] \vdash \bot) = \Omega(\mathsf{BW\text{-}Peb}(G))$?*

For cutting planes, the situation is more problematic, since for the line space measure that is the analogue of clause space it was shown in [GPT15] that cutting planes with unbounded coefficients can refute any formula in constant line space. There are other ways of getting trade-off results for line space, notably in [HN12, dRNV16], but these results are built on other ideas and are technically much more challenging (though they still make use of pebbling formulas).

9 Trade-offs for Polynomial Calculus

INSTRUCTOR'S COMMENT 2: *The notes below are **extremely** rough.*

Let us recap what we formally mean by projections.

Definition 9.1 (Projection). Let $\mathcal{P}$ be any proof system and $\mathbb{D}$ a $\mathcal{P}$-configuration. Then $\mathbb{D} \rhd C[f]$ if $\mathbb{D} \vDash C[f]$ and $\forall C' \neq C, \mathbb{D} \nvDash C'[f]$. Then we say that

$$\mathsf{Rproj}^*(\mathbb{D}) = \{C \mid \exists\, \mathbb{D}' \subseteq \mathbb{D}, \mathbb{D} \rhd C[f]\} \tag{9.1}$$

Let us give an example of a projection

Example 9.2. Let $f = \oplus_2$ and $\mathbb{D} = \{x_1 \oplus x_2, (x_1 \wedge \overline{x}_2) \vee \neg(y_1 \oplus y_2)\}$. Then $\mathsf{Rproj}^*(\mathbb{D}) = \{x, x \vee \overline{y}\}$.

What we needed for the size-space trade-off to go through for a proof system $\mathcal{P}$ was a projection where the variable space in resolution was related to the space in $\mathcal{P}$ - however it is measured. This we called that the projection was *space faithful*.

A trivial observation is that for any "reasonable" proof system (including PC, PCR, CP) Rproj^* is exactly space faithful for $f = id$ provided that the measure of space in $\mathcal{P}$ is the variable space measure. The reason for this is that if some variable y is projected then it simply has to be mentioned in the configuration it is projected from. That is, it appears in the configuration on the "$\mathcal{P}$-side". And if $\mathcal{P}$ can simulate resolution efficiently then we also get (tight) upper bounds. When a refutation in $\mathcal{P}$ was projected to resolution we only did an axiom download on the resolution side if an axiom download was made on the $\mathcal{P}$ side. This implies that we get similar trade-offs for any "reasonable" $\mathcal{P}$-system that can simulate resolution.

Why is this useful? Recall what we know about width and (clause) space in resolution:

1. If a formula can be refuted in small space then it implies that it can be refuted in small width [AD08]
2. If a formula can be refuted in small width, the lower bound on space required to refute it can still be very large [BN08]
3. There are strong trade-offs between width and space [Ben09]

If we look at PCR we have the analogies that clause space in resolution is the analogue of monomial space in PCR and that clause width in resolution is the analogue of the degree in PCR.

Then you can ask: Are the analogous width and space relations in resolution true in PCR as well? The answers as of today are:

Question 7 (Q1 for PCR). *This is a wide open question.*

Question 8 (Q2 for PCR). *This is formally an open question. But it is strongly believed that the answer is "yes". The reason for this is that pebbling formulas should behave the same in PCR.*

Question 9 (Q3 for PCR). *Here we know that the answer is as in resolution. And we are going to spend the rest of the lecture proving this.*

We claim, regarding the third question, that we can prove strong width- space trade-offs for PCR. We have that PCR can efficiently simulate resolution. So we get that, for any graph G:

$$Deg_{\mathcal{PCR}}(Peb_G \vdash \bot) \leq W_{\mathcal{R}}(Peb_G \vdash \bot) = \mathrm{O}(1) \tag{9.2}$$

$$Sp_{\mathcal{PCR}}(Peb_G \vdash \bot) \leq Sp_{\mathcal{R}}(Peb_G \vdash \bot) = \mathrm{O}(1) \tag{9.3}$$

Now what is remaining is that we have to prove that the two measures cannot be optimized simultaneously. Let G_n be the graphs of size n in [GT78] such that

$$\mathsf{BW\text{-}Peb}(G_n) = \Omega(n/\log n) \tag{9.4}$$

Claim 9.3. For Peb_{G_n} we get that any PCR refutation $\pi : Peb_{G_n}$ that

$$Sp(\pi) \cdot Deg(\pi) \geq \Omega(n/\log n) \tag{9.5}$$

So we get that the space of any such refutation gives the degree upper bounds the variable space. Any by what we previously argued for we have that the variable space is lower bounded by the pebbling cost. Q.E.D.

10 Sublinear Space Trade-Offs for PCR

Now we are going to talk about sublinear space trade-offs for PCR presented in [BNT13].

It is not hard to see that given any graph G and any fixed function f, $Peb_G[f]$ can be refuted in both resolution and polynomial calculus in linear length/size and constant width/degree simultaneously. We have that both resolution and PC can simulate black pebbling of G in roughly the same length and space. We just have to make sure that resolution can do this and every formula we will consider will have constant width and then the "naive" simulation of the resolution refutation by PC will not get any non-constant blow-ups. So this gives us that if there is a good pebbling strategy, then we will have a good upper bound on length and space.

Now we want to prove a lower bound for PCR for these formulas. That is, we get upper bounds on the weaker systems resolution and PC. And we will now get a upper bound on the stronger system PCR. This will be done using *random restrictions*.

When proving sublinear trade-offs for resolution we got that the trade-offs were essentially tight. We will not quite get that in the sense that the lower bound and upper bound wont match - but they will be close. When we extended the variable space lower bound to clause space for resolution we had that we could use any non-authoritarian substitution function. This time our proof will be more or less dependent on that we will use exclusive OR in particular.

Remark 10.1. From here and onwards: Let m be a monomial derived from $Peb_G[\oplus_2]$.

Now pick a random restriction ρ. Choose variables x_1 and x_2 and set them each to be *true* or *false* independently and uniformly at random.

Consider what will happen here if we apply such a restriction to a formula. To a monomial in the formula one out of three things can happen:

1. The monomial did not contain any of the restricted variables so nothing happens.
2. The monomial contained one or more variables that we all set to *false* (which is 1) so the monomial simply shrinks.
3. The monomial contained a variable that was set to *true* (which is 0), so it disappears.

The higher the degree of the monomial, the more likely it is to vanish after such a restriction. Formally we have that:

1. Suppose that $x_1 \in Vars(m), x_2 \notin Vars(m)$. Then with probability $3/4$, m does not vanish after the restriction.
2. Suppose $x_1, x_2 \in Vars(m)$. Then m does not vanish with probability $1/2 \leq (3/4)^2$.

From this we claim that

$$\Pr[m\restriction_\rho \neq 0] \leq (3/4)^{Deg(m)} \tag{10.1}$$

So the chance that a monomial survives a restriction shrinks exponentially with the degree of the monomial. Now if we fix the degree threshold d then we get that

$$\Pr[Deg(m\restriction_\rho) \geq d] < (3/4)^d \tag{10.2}$$

The reason for this is that ether it had a degree less than d in the first place or it had a higher degree and the it vanishes after the restriction with this probability.

Now suppose we have a PCR refutation $\pi : Peb_G[\oplus_2] \vdash \bot$ of size T and space s. Then by union bound we get that

$$\Pr[\exists m \in \pi\restriction_\rho \; Deg(m\restriction_\rho) > d] < T \cdot (3/4)^d \tag{10.3}$$

To see this, simply look at each monomial of the refutation and see if it survives the restriction. Now let us set $d = \log_{4/3} T$. Then we get that

$$(10.3) \leq T \cdot 1/T = 1 \tag{10.4}$$

which in particular means that there is a non-zero probability that all monomials will have its degree upper bounded by d. And of course the only way this can happen is if there is such a restriction ρ. So let us fix such a restriction.

Claim 10.2. $\exists \rho$ such that $Deg(m\restriction_\rho) \leq d = \mathrm{O}(\log T)$

Since the variables in the formula is substituted by $\oplus_2$ we get that what this restriction does is only some possible flipping of signs. But the refutation will still mimic a refutation of the original non-substituted formula. That is $\pi\restriction_\rho$ refutes Peb_G (modulo polarity flips) and we got that the number of equations are upper bounded by T and the variable space is upper bounded by $s \log T$ simultaneously.

Now let G_n be a graph family with constant fan-in such that

1. It is possible to do black-only pebbling, or "moderately white" pebbling in time $T = T(n)$ and space $s = s(n)$ simultaneously.

2. It is impossible to do black-white pebbling in time $\mathrm{o}(T)$ and space $\mathrm{o}(s)$ simultaneously.

Then for $Peb_G[\oplus_2]$, which are k-CNF formulas ($k = \mathrm{O}(1)$) we have that

1. $\exists$ resolution and PC refutations in length/size $\mathrm{O}(T)$ and space $\mathrm{O}(s)$ simultaneously.

2. There are no PCR refutations in size $\mathrm{o}(T)$ and space $\mathrm{o}(s/\log T)$ simultaneously.

This gives us *almost* the same set of trade-off results in PCR as for resolution (with upper bounds in resolution and PC) except they are not quite tight – we loose a log factor as a result of the restriction argument and we do not get unconditional space lower bounds.

The way that the machinery of this proof is set up implies that we can actually never get unconditional lower bounds. This is due to the fact that we find a good restriction by counting if the proof size is small enough. But in exponential size we do not know what can happen.

However it is natural to ask if we actually need such a component. Random restrictions are more or less a tool and it often is the case that if there is a proof using random restrictions, then is is possible to make another proof that do not use random restrictions.

It is sort of morally clear that it should hold for any pebbling formula with xor substitution that the monomial space is lower bounded by the pebbling space of the graph. The contrary would imply that: "It is true that any reasonable short refutation of these formulas indeed must have large space. But if you can go on for an exponential number of steps and keep a small number of very high degree monomials" – which seems unlikely.

A A Full Proof of Lemma 5.2

In this appendix, we present a complete proof of Lemma 5.2 including the details that we glossed over in class.

Fix any sequential proof system $\mathcal{P}$, any f-projection proj (for some Boolean function f), and any CNF formula F. Recall that we want to show that if $\pi_f = \{\mathbb{D}_0, \mathbb{D}_1, \dots, \mathbb{D}_\tau\}$ is a $\mathcal{P}$-refutation of the substitution formula $F[f]$, then the sequence of projected clause sets $\{\mathsf{proj}(\mathbb{D}_0), \mathsf{proj}(\mathbb{D}_1), \dots, \mathsf{proj}(\mathbb{D}_\tau)\}$ is essentially a resolution refutation π except for some details that we might have to fill in when going from $\mathsf{proj}(\mathbb{D}_{t-1})$ to $\mathsf{proj}(\mathbb{D}_t)$ in the derivation.

As we already mentioned above, parts 1 and 2 of Lemma 5.2 are immediate from Definition 5.1, since we have $\mathsf{proj}(\mathbb{D}_0) = \mathsf{proj}(\emptyset) = \emptyset$ by nontriviality and $\bot \in \mathsf{proj}(\mathbb{D}_\tau)$ by completeness (note that $\mathbb{D}_\tau \vDash \bot = \bot[f]$ and the empty clause clearly cannot be derived by weakening).

We want to show that a resolution refutation of F can get from $\mathsf{proj}(\mathbb{D}_{t-1})$ to $\mathsf{proj}(\mathbb{D}_t)$ as claimed in part 3 of the lemma. For brevity, let us write $\mathbb{C}_i = \mathsf{proj}(\mathbb{D}_i)$ for all i, and consider the possible derivation steps at time t.

Inference Suppose $\mathbb{D}_t = \mathbb{D}_{t-1} \cup \{L_t\}$ for some L_t inferred from $\mathbb{D}_{t-1}$. Since $\mathcal{P}$ is sound we have $\mathbb{D}_{t-1} \vDash \mathbb{D}_t$, and since the projection is monotone by definition we can conclude that all clauses in $\mathbb{C}_t \setminus \mathbb{C}_{t-1}$ are derivable from $\mathbb{C}_{t-1}$ by weakening. We go from $\mathbb{C}_{t-1}$ to $\mathbb{C}_t$ in three steps. First, we erase all clauses $C \in \mathbb{C}_{t-1}$ for which there are no clauses $C' \in \mathbb{C}_t$ such that $C \subseteq C'$. Then, we derive all clauses in $\mathbb{C}_t \setminus \mathbb{C}_{t-1}$ by weakening, noting that all clauses needed for weakening steps are still in the configuration. Finally, we erase the rest of $\mathbb{C}_t \setminus \mathbb{C}_{t-1}$. At all times during this transition from $\mathbb{C}_{t-1}$ to $\mathbb{C}_{t-1}$, the variable space of the intermediate clause configurations is upper-bounded by $\max\{\mathit{VarSp}(\mathbb{C}_{t-1}), \mathit{VarSp}(\mathbb{C}_t)\}$.

Erasure Suppose $\mathbb{D}_t = \mathbb{D}_{t-1} \setminus \{L_{t-1}\}$ for some $L_{t-1} \in \mathbb{D}_{t-1}$. Again we have that $\mathbb{D}_{t-1} \vDash \mathbb{D}_t$, and we can appeal to the monotonicity of the projection and proceed exactly as in the case of an inference above.

Axiom download So far, the only derivation rules used in the resolution refutation π that we are constructing are weakening and erasure, which clearly does not help π to make much progress towards proving a contradiction. Also, the only properties of the f-projection that we have used are completeness, nontriviality, and monotonicity. Note, however, that a "projection" that sends $\emptyset$ to $\emptyset$ and all other configurations to $\{\bot\}$ also satisfies these conditions. Hence, the axiom downloads are where we must expect the action to take place, and we can also expect that we will have to make crucial use of the incremental soundness of the projection.

Assume that $\mathbb{D}_t = \mathbb{D}_{t-1} \cup \{L_A\}$ for a function L_A encoding some clause from the substitution clause set $A[f]$ corresponding to an axiom clause $A \in F$. We want to show that all clauses in $\mathbb{C}_t \setminus \mathbb{C}_{t-1}$ can be derived in π by downloading A, resolving (and possibly weakening) clauses, and then perhaps erasing A, and that all this can be done without the variable space exceeding $\mathit{VarSp}(\mathbb{C}_{t-1} \cup \mathbb{C}_t) \leq \mathit{VarSp}(\mathbb{C}_{t-1}) + \mathit{VarSp}(\mathbb{C}_t)$.

We already know how to derive clauses by weakening, so consider a clause $C \in \mathbb{C}_t \setminus \mathbb{C}_{t-1}$ that cannot be derived by weakening from $\mathbb{C}_{t-1}$. By the incremental soundness of the projection, it holds for all literals $a \in \mathit{Lit}(A) \setminus \mathit{Lit}(C)$ that the clauses $\overline{a} \vee C$ can be derived from $\mathbb{C}_{t-1}$ by weakening. Once we have these clauses, we can resolve them one by one with A to derive C.

Some care is needed, though, to argue that we can stay within the variable space bound $\mathit{VarSp}(\mathbb{C}_{t-1}) + \mathit{VarSp}(\mathbb{C}_t)$ while performing these derivation steps. Observe that what was just said implies that for all $a \in \mathit{Lit}(A) \setminus \mathit{Lit}(C)$ there are clauses $\overline{a} \vee C_a \in \mathbb{C}_{t-1}$ with $C_a \subseteq C$. In particular, we have $\overline{a} \in \mathit{Lit}(\mathbb{C}_{t-1})$ for all $a \in \mathit{Lit}(A) \setminus \mathit{Lit}(C)$. This is so since by the incremental soundness there must exist some clause $C' \in \mathbb{C}_{t-1}$ such that $\overline{a} \vee C$ is derivable by weakening from C', and if $\overline{a} \notin \mathit{Lit}(C')$ we would have that C is derivable by weakening from C' as well, contrary to our assumption above.

If it happens that all clauses in $\mathbb{C}_t \setminus \mathbb{C}_{t-1}$ can be derived by weakening, we act as in the cases of inference and erasure above. Otherwise, to make the transition from $\mathbb{C}_{t-1}$ to $\mathbb{C}_t$ in a space-efficient fashion

we proceed as follows.

1. Erase all clauses in $\mathbb{C}_{t-1} \setminus \mathbb{C}_t$ not used in any of the steps below.
2. Infer all clauses in $\mathbb{C}_t \setminus \mathbb{C}_{t-1}$ that can be derived by weakening from $\mathbb{C}_{t-1}$.
3. Erase all clauses in $\mathbb{C}_{t-1} \setminus \mathbb{C}_t$ used in these weakening moves but not used in any further steps below.
4. Download the axiom clause A, and derive any clauses $C \in \mathbb{C}_t \setminus \mathbb{C}_{t-1}$ such that $A \subseteq C$ by weakening.
5. For all remaining clauses $C \in \mathbb{C}_t \setminus \mathbb{C}_{t-1}$ that have not yet been derived, derive $\overline{a} \vee C$ for all literals $a \in Lit(A) \setminus Lit(C)$ and resolve these clauses with A to obtain C.
6. Erase all remaining clauses in the current configuration that are not present in $\mathbb{C}_t$, possibly including A.

Clearly, step 1 can only decrease the variable space, and steps 2 and 3 do not increase it. Step 4 can increase the space, but as was argued above we have $Vars(A) \subseteq Vars(\mathbb{C}_{t-1}) \cup Vars(C) \subseteq Vars(\mathbb{C}t-1) \cup Vars(\mathbb{C}_t)$ for every new clause C derived with the help of A. Step 5 does not change the variable space, and step 6 can only decrease it. It follows that the set of variables mentioned during these intermediate steps is contained in $Vars(\mathbb{C}_{t-1} \cup \mathbb{C}_t)$.

Wrapping up the proof, we have shown that no matter what $\mathcal{P}$-derivation step is made in the transition $\mathbb{D}_{t-1} \rightsquigarrow \mathbb{D}_t$, we can perform the corresponding transition $\mathbb{C}_{t-1} \rightsquigarrow \mathbb{C}_t$ for our projected clause sets in resolution without the variable space going above $VarSp(\mathbb{C}_{t-1}) + VarSp(\mathbb{C}_t)$. Also, the only time we need to download an axiom $A \in F$ in our projected refutation π of F is when π_f downloads some axiom from $A[f_d]$. The lemma follows.

B A Look at Stronger Proof Systems

Instructor's comment 3: *We need a definition of k-DNF resolution, or at least a reference for it. Do Buss & Nordström define k-DNF resolution? What about Krajíček' book? Or Segerlind's survey?*

It can be shown that the projection in Definition 7.2 also works for the stronger so-called *k-DNF resolution* proof systems, which is, essentially, the resolution proof system but extended in the natural way to operate on k-DNF formulas rather than clauses (which are 1-DNF formulas), although we will not prove it in these notes. Unfortunately, there is quite a substantial loss in the parameters of the reduction here, as can be seen in the next theorem.

Theorem B.1 ([BN11]). *If f is a $(k+1)$-non-authoritarian Boolean function (for some fixed k), then the projection $\mathsf{Rproj}(\mathbb{D})$ is space-faithful of degree $k+1$ with respect to k-DNF resolution.*

Proof sketch for Theorem B.1. Let us restrict our attention to 2-DNF resolution, since this already captures the hardness of the general case. Also, we sweep quite a few technical details under the rug to focus on the main idea of the proof.

Suppose that we have a set of 2-DNF formulas $\mathbb{D}$ of size $|\mathbb{D}| = m$ such that the set of projected variables $V^* = Vars(\mathsf{Rproj}(\mathbb{D}))$ has size $|V^*| \geq K \cdot m^3$ for some suitably large constant K of our choice. We want to derive a contradiction.

As a first preprocessing step, let us prune all formulas $D \in \mathbb{D}$ one by one by shrinking any 2-term $a \wedge b$ in D to just a or just b, i.e., making D weaker, as long as this does not change the projection $\mathsf{Rproj}(\mathbb{D})$. This pruning step does not decrease the size (i.e., the number of formulas) of $\mathbb{D}$.

By counting, there must exist some formula $D \in \mathbb{D}$ containing literals belonging to at least $K \cdot m^2$ different variables in V^*. Consider some clause $C \in \mathsf{Rproj}(\mathbb{D})$ such that $\mathbb{D} \setminus \{D\}$ is too weak to project it. This means that there is an assignment α such that $\alpha(\mathbb{D} \setminus \{D\}) = 1$ but $\alpha\big(C[f]\big) \neq 1$, i.e., α either fixes $\alpha\big(C[f]\big)$ to false or leaves it undetermined. Let us pick such an α assigning values to the minimal amount

of variables. It is clear that the domain size of α will then be at most $2(m-1)$ since the assignment needs to fix only one 2-term for every formula in $\mathbb{D} \setminus \{D\}$. But this means that the formula D contains a huge number of unset variables. We would like to argue that somewhere in D there is a 2-term that can be set to true without satisfying $C[f]$, which would lead to a contradiction.

We note first that if D contains $2m$ 2-terms $x_i^b \wedge y_j^c$ with all literals[3] in these terms belonging to pairwise disjoint variable sets for distinct terms (but where we can have $x = y$), we immediately get a contradiction. Namely, if this is the case we can find at least one 2-term $x_i^b \wedge y_j^c$ such that α does not assign values to any variables $x_{i'}$ or $y_{j'}$. We can satisfy this 2-term, and hence all of $\mathbb{D}$, without satisfying $C[f]$ since by assumption f is 3-non-authoritarian (so any assignments to x_i and y_j can be repaired by setting other variables $x_{i'}, y_{j'}$ to appropriate values).

But if D does *not* contain $2m$ such 2-terms over disjoint variables, then by counting (and adjusting our constant K) there must exist some literal a that occurs in D in at least $2m$ terms $a \wedge x_i^b$ with the x_i belonging to different variables. Moreover, these 2-terms were not pruned in our preprocessing step, so they must all be necessary. Because of this, one can argue that there must exist some other assignment α' such that $\alpha'(\mathbb{D} \setminus \{D\}) = 1$, $\alpha\big(C[f]\big) \neq 1$, and $\alpha'(a) = 1$. Now at least one of the $2m$ companion variables of a is untouched by α' and can be set to true without satisfying $C[f]$. This is the contradiction we needed to finish this proof. □

Although we will not go into any details here, we want to mention that Theorem B.1 can be used to obtain the following results:

1. The k-DNF resolution proof systems form a strict hiearchy with respect to space. That is, for every $k \geq 1$ there is a family of formulas which requires non-constant formula space (the generalization of clause space) in k-DNF resolution but can be refuted in constant formula space in $(k+1)$-DNF resolution.

2. There are length space trade-offs for k-DNF resolution qualitatively similar to those in Theorems 8.1, 8.2, and 8.3, albeit with slightly worse parameters.

Comparing Theorem 7.5 to Theorem B.1, we can see that for $k = 1$ (i.e., standard resolution) the latter theorem is off by one in the exponent. A natural question is whether the exponent in Theorem B.1 can be improved from $k+1$ to k, or whether perhaps the projection in Definition 7.2 is even linearly space-faithful for k-DNF resolution for any k. The answer is that the loss in the parameters in Theorem B.1 as compared to Theorem 7.5 is necessary, except perhaps for an additive constant 1 in the degree (which does not rule out, however, that stronger trade-offs for k-DNF resolution, matching those for resolution, could be proven by other means).

Theorem B.2 ([NR11]). *Let f denote the exclusive or of $k+1$ variables. Then the projection* Rproj *cannot be space-faithful with respect to k-DNF resolution for any degree $K < k$.*

To see why we cannot hope to use Rproj to prove lower bounds for cutting planes, polynomial calculus, or PCR, consider the following examples.

Example B.3. If we have variables $x[1], x[2], x[3], \ldots$ and make substitutions using binary exclusive or $\oplus_2$ to get new variables $x[1]_1, x[1]_2, x[2]_1, x[2]_2, x[3]_1, x[3]_2, \ldots$, then the example

$$\sum_{i=1}^{k} (x[i]_1 - x[i]_2) \geq k \tag{B.1}$$

shows that just a single CP-inequality can project an arbitrarily large conjunction $x[1] \wedge x[2] \wedge \cdots \wedge x[k]$. Thus, here we have $|\mathbb{D}| = 1$ while $\mathit{VarSp}(\mathsf{Rproj}_{\oplus_2}(\mathbb{D}))$ goes to infinity.

[3] Although it is not important for this discussion, let us mention that when we use the notation x^b in the context of (k-DNF) resolution, this denotes the positive literal x if $b = 1$ and the negative literal $\overline{x}$ if $b = 0$.

Example B.4. Again using substitutions with $\oplus_2$, for polynomial calculus and PCR we have the example

$$-1 + \prod_{i=1}^{k} x[i]_1 x[i]_2 \tag{B.2}$$

showing that just two monomials can project the arbitrarily large conjunction $\overline{x[1]} \wedge \overline{x[2]} \wedge \cdots \wedge \overline{x[k]}$ if we use the projection in Definition 7.2.

As already mentioned in Open Problem 5, it would be very interesting to see whether other projections could be constructed that preserve space for polynomial calculus, or whether there are some fundamental obstacles here explaining why such an approach cannot work. For cutting planes, [GPT15] shows that we should expect major technical problems, though perhaps it could still be possible to prove line space lower bounds if we restrict the cutting planes derivations to only have coefficients on polynomial magnitude. It remains a big open problem for cutting planes to determine whether limiting coefficients to polynomial magnitude decreases the proof power of the system with respect to measures such as length or space. The strongest result in this direction is perhaps the length-space trade-offs in [dRMN+20] that hold for cutting planes with polynomially bounded coefficients but not for cutting planes with exponentially large coefficients.

References

[ABRW02] Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov, and Avi Wigderson. Space complexity in propositional calculus. *SIAM Journal on Computing*, 31(4):1184–1211, April 2002. Preliminary version in *STOC '00*.

[AD08] Albert Atserias and Víctor Dalmau. A combinatorial characterization of resolution width. *Journal of Computer and System Sciences*, 74(3):323–334, May 2008. Preliminary version in *CCC '03*.

[BBI16] Paul Beame, Chris Beck, and Russell Impagliazzo. Time-space tradeoffs in resolution: Superpolynomial lower bounds for superlinear space. *SIAM Journal on Computing*, 45(4):1612–1645, August 2016. Preliminary version in *STOC '12*.

[Ben09] Eli Ben-Sasson. Size-space tradeoffs for resolution. *SIAM Journal on Computing*, 38(6):2511–2525, May 2009. Preliminary version in *STOC '02*.

[BN08] Eli Ben-Sasson and Jakob Nordström. Short proofs may be spacious: An optimal separation of space and length in resolution. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS '08)*, pages 709–718, October 2008.

[BN11] Eli Ben-Sasson and Jakob Nordström. Understanding space in proof complexity: Separations and trade-offs via substitutions. In *Proceedings of the 2nd Symposium on Innovations in Computer Science (ICS '11)*, pages 401–416, January 2011.

[BNT13] Chris Beck, Jakob Nordström, and Bangsheng Tang. Some trade-off results for polynomial calculus. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC '13)*, pages 813–822, May 2013.

[CS80] David A. Carlson and John E. Savage. Graph pebbling with many free pebbles can be difficult. In *Proceedings of the 12th Annual ACM Symposium on Theory of Computing (STOC '80)*, pages 326–332, 1980.

[CS82] David A. Carlson and John E. Savage. Extreme time-space tradeoffs for graphs with small space requirements. *Information Processing Letters*, 14(5):223–227, 1982.

[dRMN+20] Susanna F. de Rezende, Or Meir, Jakob Nordström, Toniann Pitassi, Robert Robere, and Marc Vinyals. Lifting with simple gadgets and applications to circuit and proof complexity. In *Proceedings of the 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS '20)*, pages 24–30, November 2020.

[dRNV16] Susanna F. de Rezende, Jakob Nordström, and Marc Vinyals. How limited interaction hinders real communication (and what it means for proof and circuit complexity). In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS '16)*, pages 295–304, October 2016.

[ET01] Juan Luis Esteban and Jacobo Torán. Space bounds for resolution. *Information and Computation*, 171(1):84–97, 2001. Preliminary versions of these results appeared in *STACS '99* and *CSL '99*.

[GPT15] Nicola Galesi, Pavel Pudlák, and Neil Thapen. The space complexity of cutting planes refutations. In *Proceedings of the 30th Annual Computational Complexity Conference (CCC '15)*, volume 33 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 433–447, June 2015.

[GT78] John R. Gilbert and Robert Endre Tarjan. Variations of a pebble game on graphs. Technical Report STAN-CS-78-661, Stanford University, 1978. Available at http://infolab.stanford.edu/TR/CS-TR-78-661.html.

[HN12] Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: Amplifying communication complexity hardness to time-space trade-offs in proof complexity (Extended abstract). In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC '12)*, pages 233–248, May 2012.

[HPV77] John Hopcroft, Wolfgang Paul, and Leslie Valiant. On time versus space. *Journal of the ACM*, 24(2):332–337, April 1977. Preliminary version in *FOCS '75*.

[LT82] Thomas Lengauer and Robert Endre Tarjan. Asymptotically tight bounds on time-space trade-offs in a pebble game. *Journal of the ACM*, 29(4):1087–1130, October 1982. Preliminary version in *STOC '79*.

[Nor12] Jakob Nordström. On the relative strength of pebbling and resolution. *ACM Transactions on Computational Logic*, 13(2):16:1–16:43, April 2012. Preliminary version in *CCC '10*.

[NR11] Jakob Nordström and Alexander Razborov. On minimal unsatisfiability and time-space trade-offs for k-DNF resolution. In *Proceedings of the 38th International Colloquium on Automata, Languages and Programming (ICALP '11)*, volume 6755 of *Lecture Notes in Computer Science*, pages 642–653. Springer, July 2011.

[PTC77] Wolfgang J. Paul, Robert Endre Tarjan, and James R. Celoni. Space bounds for a game on graphs. *Mathematical Systems Theory*, 10:239–251, 1977.