DD3501 Current Research in Proof Complexity	Jan 17, 2012
Lecture 12	
Lecturer: Jakob Nordström	*Scribe: Oliver Schwarz*

1 Recap of Pebble Games and Pebbling Contradictions

Towards the end of lecture 4, and in the additional notes which became famous as "lecture 4½," we introduced black-white pebble games and related CNF formulas called pebbling contradictions. Recall that the latter are defined as follows.

Definition 1.1 (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* over G, denoted Peb_G, is the conjunction of the following clauses:

- for each source vertex s, a unit clause[2] s (*source axioms*),
- for each non-source vertex w with immediate predecessors u and v, a clause $\overline{u} \lor \overline{v} \lor w$ (*pebbling axioms*),
- for the sink z, the unit clause $\overline{z}$ (*sink axiom*).

Note that the pebbling axioms are just encoding implications $(u \land v) \rightarrow w$ 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.

In lectures 4 and 4½, we used pebbling contradictions to show that there are trade-offs between width and clause space in resolution. In today's lecture, we will use these formulas to investigate trade-offs between length and clause space.

Recall also the correspondence between pebble games and pebbling contradictions as discussed in lectures 4 and 4½. We showed then 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 small length and total space. Note that this upper bound is on *total* space and thus is stronger than claiming the same bound for *clause* space. In the other direction, if there is a resolution refutation of Peb_G in small length and *variable* space, then there exists a black-white pebbling of G in small time and space. Let us recall the definition of variable space.

Definition 1.2 (Variable space). The *variable space* of a clause configuration $\mathbb{C}$ is defined as $VarSp(\mathbb{C}) = |\,Vars(\mathbb{C})|$, i.e., counting all variables of the configuration without repetitions. The variable space of a resolution derivation $\pi = \{\mathbb{C}_0, \mathbb{C}_1, \ldots, \mathbb{C}_\tau\}$ is $VarSp(\pi) = \max_{\mathbb{C} \in \pi}\{VarSp(\mathbb{C})\}$ and the variable space of refuting a CNF formula F in resolution is $VarSp_{\mathcal{R}}(F \vdash \bot) = \min_{\pi : F \vdash \bot}\{VarSp(\pi)\}$.

As before, we omit the subscript specifying the proof system ($\mathcal{R}$ for resolution) when this is clear from context. Clearly, the refutation width of a formula is a lower bound on the refutation variable space, and a trivial upper bound is the number of variables in the formula.

[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.

[2] I.e., a clause of size 1.

2 Three Questions Regarding Length and Space in Resolution

We left three questions about resolution length and space open after the lectures of the autumn term, promising to get back to them after the Christmas break. We will discuss these questions in quite some detail in the next few lectures, so let us start by recalling what the questions were. Then we will (spoiler alert!) right away reveal short versions of the answers to these questions that we will find out in the coming lectures.

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. *If a k-CNF formula 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?*

The second question we asked is whether, given that we know that a formula can be refuted in resolution in both small length and small 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 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 here. 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 $\mathrm{O}(L)$ and in (linear) clause space $\mathrm{O}(S(F))$ simultaneously?*

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., having linear-length refutations) which exhibit almost worst-case hardness with respect to space; namely they require clause space $\Omega(S(F)/\log S(F))$. In fact, this is worst case, period, since it can be proven that any formula refutable in length n can also be refuted in clause space $\mathrm{O}(n/\log n)$. Intuitively speaking, what this says is that "space complexity and length complexity of formulas are (almost) completely unrelated."

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 and 2 can be established. All of the formulas we will study will have refutations in linear length, however, and hence also in linear clause space, so we will not be able to shed any light on Question 3 in this way.

However, very recently, it was shown in [BBI12] that also for Question 3 the answer is a very strong "no." There are 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. After our two lectures on [BN08, BN11], we will have two guest lectures by Chris Beck, one of the authors of [BBI12], covering this exciting new result.

3 Some Interesting 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 lecture 4½ based on [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 refutation length $L(F_n \vdash \bot) = \mathrm{O}(n)$ but require variable space $\mathit{VarSp}(F_n \vdash \bot) = \Omega\bigl(\frac{n}{\log n}\bigr)$.*

We note in passing that for this family of pebbling formulas, as for the other formulas mentioned below, the linear-length refutation can be carried out in constant width as well.

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

- *The total space of refuting F_n is $\mathit{TotSp}(F_n \vdash \bot) = \mathrm{O}\bigl(g(n)\bigr)$ (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 $\mathit{TotSp}(\pi_n) = \mathrm{O}\left(\left(n/(g(n))^2\right)^{1/3}\right)$ (i.e., very short refutations, but in substantially larger space).*
- *Any refutation $\pi_n : F_n \vdash \bot$ in variable space $\mathit{VarSp}(\pi_n) = \mathrm{O}\left(\left(n/(g(n))^2\right)^{1/3-\epsilon}\right)$ 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:*

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

These theorems look promising in that they are they 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 lecture 4½ 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 Generalizing 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.

As explained in lecture 4½, one way to make pebbling formulas harder is to use some suitable non-constant Boolean function $f : \{0,1\}^d \mapsto \{0,1\}$ 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 $f(\vec{x}) = f(x_1,\ldots,x_d)$. 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 $F[f]$.

A more formal description of how to do this can be found in the notes for lecture 4½. For simplicity, we give an example here instead.

Example 4.1. Let f be the exclusive or function $f(x_1,x_2) = x_1 \oplus x_2$ of arity 2. Consider the clause

$$C = \overline{x} \lor y \ . \tag{4.1}$$

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

$$\neg(x_1 \oplus x_2) \lor (y_1 \oplus y_2) \ , \tag{4.2}$$

and expanding this to CNF we get the clause set

$$C[f] = \begin{Bmatrix} x_1 \lor \overline{x}_2 \lor y_1 \lor y_2 \\ x_1 \lor \overline{x}_2 \lor \overline{y}_1 \lor \overline{y}_2 \\ \overline{x}_1 \lor x_2 \lor y_1 \lor y_2 \\ \overline{x}_1 \lor x_2 \lor \overline{y}_1 \lor \overline{y}_2 \end{Bmatrix} \ . \tag{4.3}$$

It is straightforward to verify that (4.2) holds if and only if all clauses in (4.3) 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., $\neg(x_1 \oplus x_2) \lor (y_1 \oplus y_2)$ is false, then one of the clauses in $C[f]$ must be unsatisfied.

Note that 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.

It is an easy exercise to show that if F is unsatisfiable, then 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 $Sp(\pi_f) \geq VarSp(\pi)$. We leave the verification of this fact to the reader, but note that Example 4.1 above shows how the two variables in (4.1) blow up to four clauses in (4.3). 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 4.2 ([BN11] (informal)). *For well-chosen functions f, the above-mentioned blow-up in space $Sp(\pi_f) \geq VarSp(\pi)$ 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 4.2 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 4.2. 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 $\pi = \{\mathbb{C}_0, \mathbb{C}_1, \ldots, \mathbb{C}_\tau\}$) 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 in (4.3), 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 worked, 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. We will start setting up the machinery for this today, and then give the full proof in the next lecture.

However, before we do 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 4.3 ([BN11]). *Suppose F is an unsatisfiable CNF formula and $f : \{0,1\}^d \mapsto \{0,1\}$ 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 $L(\pi_f) = L \cdot \exp(O(dw))$, total space $TotSp(\pi_f) = s \cdot \exp(O(dw))$, and width $W(\pi_f) = O(dw)$.*

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 4.3. 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 $C_1 \lor x$ and $C_2 \lor \overline{x}$ to derive $C_1 \lor C_2$, we let π_f derive $(C_1 \lor C_2)[f]$ from $(C_1 \lor x)[f]$ and $(C_2 \lor \overline{x})[f]$ (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 4.2. 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 $\neg f(\vec{x}) \lor f(\vec{y})$, then this should project the clause $\overline{x} \lor y$. 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 $\mathbb{C} \lor \mathbb{D} = \{C \lor D \mid C \in \mathbb{C},\, D \in \mathbb{D}\}$. Then for substitution in clauses it holds that $(C \lor a)[f] = C[f] \lor a[f]$ (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 $f : \{0,1\}^d \mapsto \{0,1\}$ 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 $\mathbb{D} \vDash C[f]$, then there is a $C' \subseteq C$ such that $C' \in proj_f(\mathbb{D})$ (i.e., the clause C either is in $proj_f(\mathbb{D})$ or is derivable from $proj_f(\mathbb{D})$ by weakening).

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

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

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 $C \in proj_f(\mathbb{D} \cup \{L_A\})$, it holds for all literals $a \in Lit(A) \setminus Lit(C)$ that $\overline{a} \lor C \supseteq C_a \in proj_f(\mathbb{D})$.

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 $f : \{0,1\}^d \mapsto \{0,1\}$ a Boolean function, and suppose that $proj_f$ 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 of sets of projected clauses $\{proj_f(\mathbb{D}_0), proj_f(\mathbb{D}_1), \ldots, proj_f(\mathbb{D}_\tau)\}$ forms the "backbone" of a resolution refutation π of F in the following sense:*

1. *We start with an empty blackboard: $proj_f(\mathbb{D}_0) = \emptyset$.*
2. *We end with contradiction: $\bot \in proj_f(\mathbb{D}_\tau)$.*
3. *All transitions from $proj_f(\mathbb{D}_{t-1})$ to $proj_f(\mathbb{D}_t)$ for time $t \in [\tau]$ can be accomplished in resolution in such a fashion that $VarSp(\pi) = \mathrm{O}\big(\max_{\mathbb{D} \in \pi_f}\{VarSp(proj_f(\mathbb{D}))\}\big)$.*
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 $F[f]$.*

Lemma 5.2 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 space in π_f and π in Lemma 4.2. We will return to this question in the next lecture, where we will restrict our attention to the proof system $\mathcal{P}$ being resolution in order to establish such a connection. We will also prove Lemma 5.2 next time, or at least sketch the proof, but this is all we had for today.

References

[BBI12] Paul Beame, Chris Beck, and Russell Impagliazzo. Time-space tradeoffs in resolution: Superpolynomial lower bounds for superlinear space. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC '12)*, May 2012. To appear.

[Ben09] Eli Ben-Sasson. Size space tradeoffs for resolution. *SIAM Journal on Computing*, 38(6):2511–2525, May 2009. Preliminary version appeared 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. Full-length version available at http://eccc.hpi-web.de/report/2010/125/.

[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.

[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.

[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 appeared in *STOC '79*.

[Nor12] Jakob Nordström. On the relative strength of pebbling and resolution. *ACM Transactions on Computational Logic*, 13(2), 2012. To appear. Preliminary version appeared in *CCC '10*.

[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.