SIAM J. COMPUT.
Vol. 39, No. 1, pp. 59–121

NARROW PROOFS MAY BE SPACIOUS: SEPARATING SPACE AND WIDTH IN RESOLUTION*

JAKOB NORDSTRÖM†

Abstract. The width of a resolution proof is the maximal number of literals in any clause of the proof. The space of a proof is the maximal number of clauses kept in memory simultaneously if the proof is only allowed to infer new clauses from clauses currently in memory. Both of these measures have previously been studied and related to the resolution refutation size of unsatisfiable conjunctive normal form (CNF) formulas. Also, the minimum refutation space of a formula has been proven to be at least as large as the minimum refutation width, but it has been open whether space can be separated from width or the two measures coincide asymptotically. We prove that there is a family of k-CNF formulas for which the refutation width in resolution is constant but the refutation space is nonconstant, thus solving a problem mentioned in several previous papers.

Key words. proof complexity, resolution, width, space, separation, lower bound, pebble game, pebbling contradiction

AMS subject classifications. 03B35, 03B70, 03F20, 68Q17, 68T15

DOI. 10.1137/060668250

1. Introduction.

A proof system for a language L is an algorithm $P(x, \pi)$ which runs in time polynomial in $|x|$ and $|\pi|$ such that for all $x \in L$ there is a string π (a *proof*) for which $P(x, \pi) = 1$. For $x \notin L$, it should hold for all strings π that $P(x, \pi) = 0$. The complexity of a proof system P is the smallest bounding function $g : \mathbb{N} \mapsto \mathbb{N}$ such that $x \in L$ if and only if there is a proof π of size $|\pi| \leq g(|x|)$ for which $P(x, \pi) = 1$. If a proof system is of polynomial complexity, it is said to be polynomially bounded. A *propositional proof system* is a proof system for tautologies in propositional logic.

The central task of proof complexity is to construct and investigate the power of different propositional proof systems. The purpose of this endeavor is at least twofold.

First, propositional proof complexity is closely related to the question of P versus NP, which is recognized as a major open problem in theoretical computer science and mathematics. Since NP is exactly the set of languages with polynomially bounded proof systems, and since TAUTOLOGY can be seen to be the dual problem of SATISFIABILITY, we have the famous theorem of Cook and Reckhow [24] that $\mathsf{NP} = \mathsf{co\text{-}NP}$ if and only if there exists a polynomially bounded propositional proof system. Thus, if it could be shown that there are no polynomially bounded proof systems for propositional tautologies, $\mathsf{P} \neq \mathsf{NP}$ would follow as a corollary since P is closed under complement. One way of approaching this distant goal is to study stronger and stronger proof systems and try to prove superpolynomial lower bounds on proof size. However, although great progress has been made in the last couple of decades for a variety of propositional proof systems, it seems that we are still very far from fully understanding the reasoning power of even quite simple ones.

Second, designing efficient algorithms for proving tautologies (or, equivalently, testing satisfiability) is a very important problem not only in theoretical computer science but also in applied research and industry, for instance, in the context of

*Received by the editors August 24, 2006; accepted for publication (in revised form) July 2, 2007; published electronically May 28, 2009.
http://www.siam.org/journals/sicomp/39-1/66825.html

†Royal Institute of Technology (KTH), SE-100 44 Stockholm, Sweden (jakobn@kth.se).

formal methods. All automated theorem provers, regardless of whether they actually produce a written proof, explicitly or implicitly define a system in which proofs are searched for and rules which determine what proofs in this system look like. Lower bounds for proofs in such proof systems imply lower bounds on the running time of corresponding automated theorem provers. In the other direction, theoretical upper bounds on proof size in a system can give upper bounds on the running time of a proof search algorithm, provided that the algorithm can be shown to search for proofs in the system in an efficient manner.

Also, the field of proof complexity has rich connections to cryptography, artificial intelligence, and mathematical logic. Some good surveys of proof complexity are [9, 12, 21, 55].

1.1. Previous work. Any propositional logic formula can be converted to a formula in conjunctive normal form (CNF) that is only linearly larger and is unsatisfiable if and only if the original formula is a tautology. Therefore, any sound and complete system which produces refutations of unsatisfiable CNF formulas can be considered as a general propositional proof system.

One such proof system, arguably the most studied propositional proof system, is *resolution*. The resolution proof system appeared in [17] and began to be investigated in connection with automated theorem proving in the 1960s [26, 27, 49]. Because of its simplicity—there is only one derivation rule—and because all lines in a proof are clauses, this proof system is well adapted to proof search algorithms. Many real-world automated theorem provers are based on resolution.

Being so simple and fundamental, resolution was also a natural target to attack when developing methods for proving lower bounds in proof complexity. In this context, it is most straightforward to prove bounds on the *length* of proofs, i.e., the number of clauses, which is easily seen to be polynomially related to the proof size. In 1968, Tseitin [53] presented a superpolynomial lower bound on proof length for a restricted form of resolution, called *regular* resolution, but it was not until almost 20 years later that Haken [33] proved the first superpolynomial lower bound for general resolution. This exponential lower bound of Haken has later been followed by many other strong results on resolution proof length, for instance, in [10, 11, 16, 22, 23, 44, 47, 48, 54].

A second complexity measure for resolution, first made explicit by Galil [31], is the *width*, measured as the maximal size of a clause in the proof. Ben-Sasson and Wigderson [16] showed a strong upper bound on width in terms of length, thus providing a new method for proving lower bounds on proof length by proving lower bounds on width.

The tight connection between proof length and proof width raised the question of whether other complexity measures could yield interesting insights as well. In [29, 51], Esteban and Torán introduced the concept of *space* in resolution, transforming a previous definition from [36]. Intuitively, the space of a resolution proof is the maximal number of clauses one needs to keep in memory while verifying the proof. This measure is related to the memory required by the family of so-called DPLL algorithms for propositional satisfiability based on [26, 27].

A number of upper and lower bounds for proof space in resolution and other proof systems were subsequently presented in, for instance, [3, 14, 28, 30]. In several of these papers, it was noted that, somewhat unexpectedly, the lower bounds on resolution proof space for different formula families exactly matched previously known lower bounds on proof width. Atserias and Dalmau [7] showed that this was not a coincidence but that the minimum space of refuting any unsatisfiable k-CNF formula$F$ is at least as large as the minimum width of refuting it minus a small constant depending on k.

1.2. Questions left open by previous research. Some natural remaining open questions in this line of research relating the proof complexity measures of length, width, and space in resolution are as follows:

1. The main theorem in [7] says that (essentially) space $\geq$ width, but it leaves open whether this relationship is tight up to additive or multiplicative constants. Do refutation space and width always coincide, or is there a formula family that separates the two measures asymptotically?
2. What is the relation between space and length? It is not too hard to see that upper bounds on width imply upper bounds on length, and as a consequence of [7] this must be true for space with respect to length as well. In the other direction, we have the result from [16] discussed above that upper bounds on length imply upper bounds on width. Is there a similar Ben-Sasson–Wigderson-style upper bound on space in terms of length, or can short resolution proofs be arbitrarily complex with respect to space?
3. A third, intimately connected question is to determine the refutation space of the formula family of *pebbling contradictions* defined in terms of pebble games on directed acyclic graphs (DAGs). Nonconstant lower bounds on the space of refuting pebbling contradictions would separate space and width and possibly also clarify the relation between space and length if the bounds were good enough. On the other hand, a constant upper bound on the refutation space would improve the trade-off results for different proof complexity measures for resolution in [13].

The above three questions have been mentioned as interesting open problems in [13, 28, 30, 52].

1.3. Our contribution. In this paper, we answer the first question above by separating space and width. This is done by proving an asymptotically tight bound on space for pebbling contradictions over binary trees, thus at least partially solving the open problem about the space complexity of pebbling contradictions as well.

More precisely, our results are as follows (formal definitions are given in sections 3 and 4).

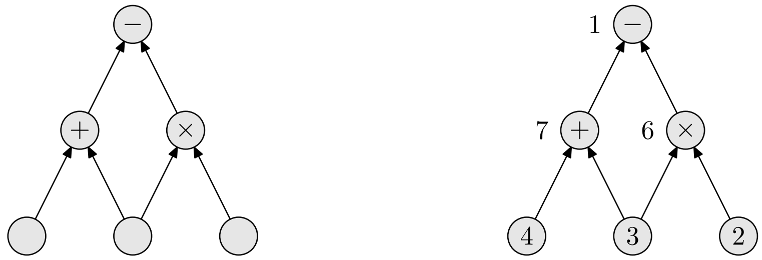
THEOREM 1.1. *The space of refuting pebbling contradictions over complete binary trees of height h in resolution grows as $\Theta(h)$, provided that the number of variables per vertex in the pebbling contradictions is at least 2.*

COROLLARY 1.2. *For all $k \geq 4$, there is a family $\{F_n\}_{n=1}^{\infty}$ of k-CNF formulas of size $\mathrm{O}(n)$ that can be refuted in width $W(F_n \vdash 0) = \mathrm{O}(1)$ but require space $Sp(F_n \vdash 0) = \Theta(\log n)$.*

2. Proof overview and paper organization. We now outline how this paper is organized and in the process try to give an intuitive, high-level description of the lower bound proof that is the main component in Theorem 1.1.

2.1. Preliminaries. We start in section 3 by defining the resolution proof system (Definition 3.2) and the measures length, width, and space. This then allows us to give more precise statements of the results referred to in section 1.1.

A quick, informal summary of section 3 is that a resolution refutation of a CNF formula F is a sequence of derivation steps. In each step we can write a clause from F on the blackboard, erase a clause from the blackboard, or derive some new disjunctive



(a) *DAG encoding calculation.* (b) *After pebbling with results filled in.*

FIG. 1. *Example of modeling calculation as pebbling of DAG.*

clause implied by the current content of the blackboard.[1] The refutation ends when we reach the contradictory empty clause. The *width* of a resolution refutation is the size of the largest clause in the refutation. The *space* is the maximum number of clauses on the blackboard simultaneously. We write $W(F \vdash 0)$ to denote the minimum width of any refutation of F, and $Sp(F \vdash 0)$ to denote the minimum space of any refutation.

Section 4 introduces pebble games and CNF formulas defined in terms of these games.

Pebble games on DAGs model the calculations described by these DAGs, where the source vertices contain the input and nonsource vertices specify operations on the values of the predecessors (see Figure 1). Placing a pebble on a vertex v corresponds to storing in memory the partial result of the calculation described by the subgraph rooted at v. Removing a pebble from v corresponds to deleting the partial result of v from memory. A *pebbling* of a DAG G is a sequence of moves starting with the empty graph G and ending with all vertices in G empty except for a pebble on the (unique) sink vertex. The *cost* of a pebbling is the maximal number of pebbles used simultaneously at any point in time during the pebbling. The *pebbling price* of a DAG G is the minimum cost of any pebbling, i.e., the minimum number of memory registers required to perform the complete calculation described by G.

The pebble game on a DAG G can be encoded as an unsatisfiable CNF formula Peb^d_G, a so-called *pebbling contradiction* (Definition 4.4). See Figure 2 for a small example. Very briefly, pebbling contradictions are constructed as follows:

- Associate d variables $x(v)_1, \dots, x(v)_d$ with each vertex v for some fixed d (in Figure 2 we have $d = 2$).
- Specify that all sources have at least one true variable (for example, the clause $x(r)_1 \vee x(r)_2$ for the vertex r).
- Add clauses saying that truth propagates from predecessors to successors (for instance, for u with predecessors r and s, clauses 4–7 in Figure 2 are the CNF encoding of $(x(r)_1 \vee x(r)_2) \wedge (x(s)_1 \vee x(s)_2) \rightarrow (x(u)_1 \vee x(u)_2)$).
- To get a contradiction, conclude the formula with $\overline{x(z)}_1 \wedge \cdots \wedge \overline{x(z)}_d$ for z the sink of the DAG.

In section 4.2, we define these formulas formally and review what is known about them. In particular, we recall that pebbling contradictions can be refuted in resolution in constant width (Theorem 4.5).

[1]The resolution derivation rule is as in (3.2) on page 67, but for our purposes, it turns out that the exact definition of the rule is not essential—our lower bound holds for any sound derivation rule. What is important is that we are only allowed to derive new clauses that follow logically from the set of clauses currently on the blackboard.

$$
\begin{aligned}
& (x(r)_1 \lor x(r)_2) \\
\land\ & (x(s)_1 \lor x(s)_2) \\
\land\ & (x(t)_1 \lor x(t)_2) \\
\land\ & (\overline{x(r)}_1 \lor \overline{x(s)}_1 \lor x(u)_1 \lor x(u)_2) \\
\land\ & (\overline{x(r)}_1 \lor \overline{x(s)}_2 \lor x(u)_1 \lor x(u)_2) \\
\land\ & (\overline{x(r)}_2 \lor \overline{x(s)}_1 \lor x(u)_1 \lor x(u)_2) \\
\land\ & (\overline{x(r)}_2 \lor \overline{x(s)}_2 \lor x(u)_1 \lor x(u)_2) \\
\land\ & (\overline{x(s)}_1 \lor \overline{x(t)}_1 \lor x(v)_1 \lor x(v)_2) \\
\land\ & (\overline{x(s)}_1 \lor \overline{x(t)}_2 \lor x(v)_1 \lor x(v)_2) \\
\land\ & (\overline{x(s)}_2 \lor \overline{x(t)}_1 \lor x(v)_1 \lor x(v)_2) \\
\land\ & (\overline{x(s)}_2 \lor \overline{x(t)}_2 \lor x(v)_1 \lor x(v)_2) \\
\land\ & (\overline{x(u)}_1 \lor \overline{x(v)}_1 \lor x(z)_1 \lor x(z)_2) \\
\land\ & (\overline{x(u)}_1 \lor \overline{x(v)}_2 \lor x(z)_1 \lor x(z)_2) \\
\land\ & (\overline{x(u)}_2 \lor \overline{x(v)}_1 \lor x(z)_1 \lor x(z)_2) \\
\land\ & (\overline{x(u)}_2 \lor \overline{x(v)}_2 \lor x(z)_1 \lor x(z)_2) \\
\land\ & \overline{x(z)}_1 \\
\land\ & \overline{x(z)}_2
\end{aligned}
$$

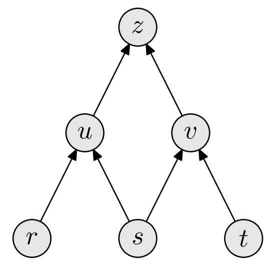


FIG. 2. *The pebbling contradiction $Peb^2_{\Pi_2}$ for the pyramid graph Π_2 of height 2.*

2.2. Tentative proof idea. Now one could try to argue that if we pick DAGs G with high pebbling price, since the corresponding pebbling contradictions encode calculations which require large memory, any resolution proofs refuting these formulas should require large space.

More specifically, what we would like to do is to establish a connection between resolution refutations of pebbling contradictions on the one hand, and the so-called *black-white pebble game* modeling the nondeterministic computations described by the underlying graphs on the other. Our intuition is that resolution should have to conform to the combinatorics of the pebble game in the sense that from any resolution refutation of a pebbling contradiction Peb^d_G we should be able to extract a pebbling of the DAG G in terms of which the pebbling contradiction is defined.

Ideally, we would like to give a proof of a lower bound for the refutation space of pebbling contradictions along the following lines:

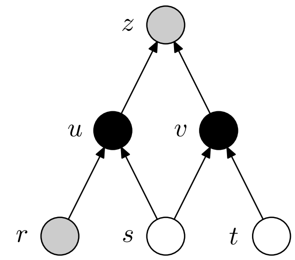
1. First, find a natural interpretation of sets of clauses currently "on the blackboard" in a resolution refutation of Peb^d_G in terms of black and white pebbles on the vertices of G.
2. Then, prove that this interpretation of clauses in terms of pebbles captures the pebble game in the following sense: for any resolution refutation of Peb^d_G, looking at consecutive sets of clauses on the blackboard and considering the corresponding sets of pebbles in the graph yields a black-white pebbling of G in accordance with the rules of the pebble game.
3. Finally, show that the interpretation captures resolution space in the sense that if some blackboard clause configuration induces a lot of pebbles on the graph, then there must be many clauses on the blackboard.

Combining this with known lower bounds on the pebbling price of G, this would imply a lower bound on the refutation space of pebbling contradictions. As a corollary, we would get a separation of space and width in resolution.

Let us sketch what the formal argument would look like: Consider an arbitrary resolution refutation of Peb^d_G. From this refutation we extract a pebbling of G. At some point in the obtained pebbling, there must be a lot of pebbles on the vertices

$$\begin{bmatrix} x(u)_1 \lor x(u)_2 \\ \overline{x(s)}_1 \lor \overline{x(t)}_1 \lor x(v)_1 \lor x(v)_2 \\ \overline{x(s)}_1 \lor \overline{x(t)}_2 \lor x(v)_1 \lor x(v)_2 \\ \overline{x(s)}_2 \lor \overline{x(t)}_1 \lor x(v)_1 \lor x(v)_2 \\ \overline{x(s)}_2 \lor \overline{x(t)}_2 \lor x(v)_1 \lor x(v)_2 \end{bmatrix}$$

(a) *Clauses on blackboard.*



(b) *Corresponding pebbles in the graph.*

FIG. 3. *Example of intuitive correspondence between sets of clauses and pebbles.*

of G, since G was chosen with high pebbling price. But this means that at some point in the proof, there are a lot of clauses on the blackboard. Since this holds for any resolution refutation, the refutation space of Peb_G^d must be large. The separation result follows from the fact that we know that pebbling contradictions can be refuted in constant width.

Unfortunately, we will not quite be able to make this proof idea work. In the next subsection, we describe the modifications that we are forced to make. We then state precisely the results that we need and show how the bits and pieces fit together to yield Theorem 1.1 and Corollary 1.2.

2.3. Detailed overview of formal proof. The black-white pebble game played on a DAG G can be viewed as a way of proving the end result of the calculation described by G. Black pebbles correspond to proven partial results of the computation. White pebbles correspond to assumptions about partial results which have been used to derive other partial results (i.e., black pebbles), but these assumptions will have to be verified for the calculation to be complete. The final goal is a black pebble on the sink z and no other pebbles in the graph, which corresponds to us having unconditionally proven the end result of the calculation, having eliminated any assumptions made along the way.

Translating this to pebbling contradictions, it turns out that a fruitful way to think of a black pebble on v is that it should correspond to truth of the disjunction $\bigvee_{i=1}^{d} x(v)_i$ of all positive literals over v, or to "truth of v." A white pebble on a vertex w can be understood to mean that we need to *assume* the partial result on w to derive the black pebbles above w in the graph. Needing to assume the truth of w is the opposite of knowing the truth of w, so extending the reasoning above we get that a white-pebbled vertex should correspond to "falsity of w," i.e., to all negative literals $\overline{x(w)}_i$, $i \in [d]$, over w. Using this intuitive correspondence, we can translate clauses of a resolution refutation of Peb_G^d into black and white pebbles in G as in Figure 3.

The translation from sets of clauses to sets of black and white pebbles sketched in the previous paragraph is rather straightforward and seems to yield well-behaved black-white pebblings for all "sensible" resolution refutations of Peb_G^d. The problem is that we have no guarantee that the refutations will be "sensible." Although it might seem more or less clear how an optimal resolution refutation of a pebbling contradiction should proceed, a particular refutation might contain unintuitive and seemingly nonoptimal derivation steps that do not make much sense from a pebble game perspective.

A first idea would be to try to *prove* that such derivation steps are indeed nonoptimal and can be eliminated, but it appears tricky to nail down formally wherein the supposed "nonoptimality" lies. Also, it seems hard to interpret clauses in terms of pebbles in such a way that such apparently arbitrary derivation steps comply with the rather restrictive set of rules of the black-white pebble game. Instead, what we do is to modify the pebbling rules.

The most important change is that we have to allow "sliding moves" of black pebbles downward and of white pebbles upward in the graph. As we will see, this leads to serious technical difficulties. Another, less far-reaching, modification of the game can be motivated as follows. Looking at the clauses and pebbles in Figure 3, it somehow seems that the white pebbles on s and t are relevant only for the black pebble on v. The black pebble on u corresponding to $\bigvee_{i=1}^{d} x(u)_i$ is wholly independent of these white pebbles, although strictly from a pebbling perspective it is not, since the white pebble on s is below u. It turns out that it is important to formalize and keep track of this "dependence" relation between black and white pebbles, so we will have to label each black pebble in the graph with the white-pebbled vertices it depends on. Because of this property, we name the new game the *labeled pebble game* (Definition 5.2).

Once we have defined this labeled pebble game in section 5, we continue according to the proof outline of section 2.2. However, for reasons that will be clear below, we restrict our attention to binary trees.

In section 6, we show that a resolution refutation of a pebbling contradiction defined over a binary tree induces a pebbling of this tree in our modified pebble game.

THEOREM 2.1. *Let $Peb^{d}_{T_h}$ denote the pebbling contradiction of degree $d \geq 1$ over the complete binary tree T_h of height h. Then there is a translation function from sets of clauses derived from $Peb^{d}_{T_h}$ into sets of pebbles in T_h such that any resolution refutation π of $Peb^{d}_{T_h}$ corresponds to a labeled pebbling $\mathcal{L}_\pi$ of T_h under this translation.*

In section 7, we prove that if the number of variables d associated to each vertex is at least 2, then the cost of the labeled pebbling $\mathcal{L}_\pi$ in Theorem 2.1 is related to the space of the resolution refutation π.

THEOREM 2.2. *If π is a resolution refutation of a pebbling contradiction $Peb^{d}_{T_h}$ of degree $d > 1$, then the cost of the associated labeled pebbling $\mathcal{L}_\pi$ is asymptotically bounded by the space of π, or in formal notation, $\mathsf{cost}(\mathcal{L}_\pi) = \mathrm{O}(Sp(\pi))$.*

Finally, we need a lower bound for the pebbling price of binary trees in the labeled pebble game.

THEOREM 2.3. *Any complete labeled pebbling $\mathcal{L}$ of T_h must have cost at least linear in the tree height h. That is, the labeled pebbling price of T_h is $\mathsf{L\text{-}Peb}(T_h) = \Omega(h)$.*

We establish this result by transforming labeled pebblings to pebblings in the standard black-white pebble game and then using known bounds on the black-white pebbling price of binary trees (Theorem 4.3). The technically quite complicated proof is given in the appendix. The reason that we consider only binary trees is that the analogue of Theorem 2.3 does not hold for more general DAGs. For instance, it is false for the pyramid graph in Figure 3 (Lemma 5.3).

Putting all of this together, we can now prove our main theorem.

THEOREM 1.1 (restated). *Let T_h denote the complete binary tree of height h and $Peb^{d}_{T_h}$ the pebbling contradiction of degree $d > 1$ defined over T_h. Then the space of refuting $Peb^{d}_{T_h}$ by resolution is $Sp\big(Peb^{d}_{T_h} \vdash 0\big) = \Theta(h)$.*

Proof. The upper bound $Sp\big(Peb^{d}_{T_h} \vdash 0\big) = \mathrm{O}(h)$ is the easy part. It follows from Theorems 4.3 and 4.8, since the refutation space of a pebbling contradiction is upper-bounded by the black pebbling price of its underlying graph, and binary trees of height h have black pebbling price $\mathrm{O}(h)$.

For the lower bound, let π be any resolution refutation of $Peb^d_{T_h}$. Consider the associated labeled pebbling $\mathcal{L}_\pi$ provided by Theorem 2.1.

On the one hand, we know that $\mathsf{cost}(\mathcal{L}_\pi) = \mathrm{O}(Sp(\pi))$ by Theorem 2.2, provided that $d > 1$. On the other hand, Theorem 2.3 tells us that the cost of any pebbling of T_h is $\Omega(h)$, so in particular we must have $\mathsf{cost}(\mathcal{L}_\pi) = \Omega(h)$. Combining these two bounds on $\mathsf{cost}(\mathcal{L}_\pi)$, we see that $Sp(\pi) = \Omega(h)$. Since this bound holds for any resolution refutation π, it follows that the minimum clause space of refuting $Peb^d_{T_h}$ is $Sp\bigl(Peb^d_{T_h} \vdash 0\bigr) = \Omega(h)$ for $d > 1$.

The theorem follows. $\square$

The pebbling contradiction Peb^d_G is a $(2{+}d)$-CNF formula, and for fixed d the size of the formula is linear in the number of vertices of G (compare Figure 2). Thus, for binary trees, $Peb^d_{T_h}$ has size exponential in the tree height h. Also, Theorem 4.5 tells us that Peb^d_G can be refuted in width $W\bigl(Peb^d_G \vdash 0\bigr) = \mathrm{O}(d)$ for any graph G.

The separation of space and width in Corollary 1.2 follows from this if we fix $d > 1$ and set $F_n = Peb^d_{T_h}$ for $h = \lfloor \log(n+1) \rfloor$ in Theorem 1.1.

Corollary 1.2 (restated). *For all $k \geq 4$, there is a family of k-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\mathrm{O}(n)$ such that $W(F_n \vdash 0) = \mathrm{O}(1)$ but $Sp(F_n \vdash 0) = \Theta(\log n)$.*

2.4. Paper organization. The preliminaries about resolution and pebbling are presented in sections 3 and 4, respectively. In section 5, we introduce the modified pebble game that we will work with. Theorem 2.1 is proven in section 6 and Theorem 2.2 in section 7, while the quite lengthy proof of Theorem 2.3 is postponed to the appendix. We conclude the paper proper in section 8 by giving suggestions for further research.

3. The resolution proof system. A *literal* is either a propositional logic variable or its negation, denoted x and $\overline{x}$, respectively. We define $\overline{\overline{x}} = x$. Two literals a and b are *strictly distinct* if $a \neq b$ and $a \neq \overline{b}$, i.e., if they refer to distinct variables.

A *clause* $C = a_1 \vee \cdots \vee a_k$ is a set of literals. Throughout this paper, all clauses C are assumed to be nontrivial in the sense that all literals in C are pairwise strictly distinct (otherwise, C is trivially true). We say that C is a *subclause* of D if $C \subseteq D$. A clause containing at most k literals is called a *k-clause*.

A *CNF formula* $F = C_1 \wedge \cdots \wedge C_m$ is a set of clauses. A *k-CNF formula* is a CNF formula consisting of k-clauses.

In this paper, when nothing else is stated we let A, B, C, D denote clauses; $\mathbb{C}, \mathbb{D}$ denote sets of clauses; x, y denote propositional variables; a, b, c denote literals; α, β denote truth value assignments; and ν denote a truth value 0 or 1. We define

$$\alpha^{x=\nu}(y) = \begin{cases} \alpha(y) & \text{if } y \neq x, \\ \nu & \text{if } y = x. \end{cases} \tag{3.1}$$

We let $Vars(C)$ denote the set of variables and $Lit(C)$ the set of literals in a clause C.[2] This notation is extended to sets of clauses by taking unions. Also, we employ the standard notation $[n] = \{1, 2, \ldots, n\}$.

[2]Although the notation $Lit(C)$ is slightly redundant given the definition of a clause as a set of literals, we include it for clarity.

The size of a formula F is the total number of literals in F counted with repetitions. More often, we will be interested in the number of clauses $|F|$ of F.

A *resolution derivation* $\pi : F \vdash A$ of a clause A from a CNF formula F is a sequence of clauses $\pi = \{D_1, \ldots, D_\tau\}$ such that $D_\tau = A$ and each line D_i, $i \in [\tau]$, is either one of the clauses in F (*axioms*) or is derived from clauses D_j, D_k in π with $j, k < i$ by the *resolution rule*

$$\frac{B \vee x \quad C \vee \overline{x}}{B \vee C}. \tag{3.2}$$

We refer to (3.2) as *resolution on the variable* x and to $B \vee C$ as the *resolvent* of $B \vee x$ and $C \vee \overline{x}$ on x. A *resolution refutation* of a CNF formula F is a resolution derivation of the empty clause 0 (the clause with no literals) from F.

For a formula F and a set of formulas $\mathcal{G} = \{G_1, \ldots, G_n\}$, we say that $\mathcal{G}$ *implies* F, denoted $\mathcal{G} \vDash F$, if every truth value assignment satisfying all formulas $G \in \mathcal{G}$ satisfies F as well. It is well known that resolution is sound and implicationally complete. That is, if there is a resolution derivation $\pi : F \vdash A$, then $F \vDash A$, and if $F \vDash A$, then there is a resolution derivation $\pi : F \vdash A'$ for some $A' \subseteq A$. In particular, F is unsatisfiable if and only if there is a resolution refutation of F.

With every resolution derivation $\pi : F \vdash A$ we can associate a DAG G_π, with the clauses in π labeling the vertices and with edges from the assumption clauses to the resolvent for each application of the resolution rule. There might be several different derivations of a clause C in π, but if so, we can label each occurrence of C with a timestamp when it was derived and keep track of which copy of C is used where. A resolution derivation π is *tree-like* if any clause in the derivation is used at most once as a premise in an application of the resolution rule, i.e., if G_π is a tree. (We may make different "time-stamped" vertex copies of the axiom clauses in order to make G_π into a tree.)

The *length* $L(\pi)$ of a resolution derivation π is the number of clauses in it. The length of deriving a clause A from a formula F is $L(F \vdash A) = \min_{\pi : F \vdash A} \{L(\pi)\}$, where the minimum is taken over all resolution derivations of A. In particular, the length of refuting F by resolution is $L(F \vdash 0)$. The length of refuting F by tree-like resolution $L_{\mathfrak{T}}(F \vdash 0)$ is defined by taking the minimum over all tree-like resolution refutations π_T of F.

The *width* $W(C)$ of a clause C is $|C|$. The width of a set of clauses $\mathbb{C}$ is $W(\mathbb{C}) = \max_{C \in \mathbb{C}}\{W(C)\}$. The width of deriving A from F by resolution is $W(F \vdash A) = \min_{\pi : F \vdash A}\{W(\pi)\}$, and the width of refuting F is denoted $W(F \vdash 0)$. (Note that the minimum width measures in general and tree-like resolution coincide, so it makes no sense to define $W_{\mathfrak{T}}(F \vdash 0)$.)

If a resolution refutation has constant width, it is easy to see that it must be of size polynomial in the number of variables (just count the maximum possible number of distinct clauses). Conversely, if all refutations of a formula are very wide, it seems reasonable that any refutation of this formula must be very long as well. This intuition was made precise by Ben-Sasson and Wigderson.

THEOREM 3.1 (see [16]). *The width of refuting a CNF formula F is bounded from above by*

$$W(F \vdash 0) \leq W(F) + c \cdot \sqrt{n \log L(F \vdash 0)}\,,$$

where n is the number of variables in F and c is a universal constant independent of F.

Bonet and Galesi [19] showed that this bound on width in terms of length is essentially optimal.

We next define the measure of *space*. Following the exposition in [29], a proof can be seen as a Turing machine computation, with a special read-only input tape from which the axioms can be downloaded and a working memory where all derivation steps are made. The *clause space* of a resolution proof is the maximum number of clauses that need to be kept in memory simultaneously during a verification of the proof. The *variable space* is the maximum total space needed, where also the width of the clauses is taken into account.

For the formal definitions, it is convenient to use an alternative definition of resolution introduced in [3].

Definition 3.2 (resolution). *A* clause configuration $\mathbb{C}$ *is a set of clauses. A sequence of clause configurations* $\{\mathbb{C}_0, \ldots, \mathbb{C}_\tau\}$ *is a* resolution derivation *from a CNF formula* F *if* $\mathbb{C}_0 = \emptyset$ *and for all* $t \in [\tau]$, $\mathbb{C}_t$ *is obtained from* $\mathbb{C}_{t-1}$ *by one*[3] *of the following rules:*
Axiom download. $\mathbb{C}_t = \mathbb{C}_{t-1} \cup \{C\}$ *for some* $C \in F$.
Erasure. $\mathbb{C}_t = \mathbb{C}_{t-1} \setminus \{C\}$ *for some* $C \in \mathbb{C}_{t-1}$.
Inference. $\mathbb{C}_t = \mathbb{C}_{t-1} \cup \{D\}$ *for some* D *inferred by resolution from* $C_1, C_2 \in \mathbb{C}_{t-1}$.
A resolution derivation $\pi : F \vdash A$ *of a clause* A *from a formula* F *is a derivation* $\{\mathbb{C}_0, \ldots, \mathbb{C}_\tau\}$ *such that* $\mathbb{C}_\tau = \{A\}$. *A* resolution refutation *of* F *is a derivation of* 0 *from* F.

Definition 3.3 (clause space [3, 13]). *The* clause space *of a resolution derivation* $\{\mathbb{C}_0, \ldots, \mathbb{C}_\tau\}$ *is* $\max_{t\in[\tau]}\{|\mathbb{C}_t|\}$. *The clause space of deriving the clause* A *from the formula* F *is* $Sp(F \vdash A) = \min_{\pi: F\vdash A}\{Sp(\pi)\}$. $Sp(F \vdash 0)$ *is the minimum clause space of any resolution refutation of* F.

Definition 3.4 (variable space [3]). *The* variable space *of a configuration* $\mathbb{C}$ *is* $VarSp(\mathbb{C}) = \sum_{C\in\mathbb{C}} W(C)$. *The variable space of a resolution derivation* $\{\mathbb{C}_0, \ldots, \mathbb{C}_\tau\}$ *is* $\max_{t\in[\tau]}\{VarSp(\mathbb{C}_t)\}$, *and* $VarSp(F \vdash 0)$ *is the minimum variable space of any resolution refutation of* F.

Restricting the resolution derivations to tree-like resolution, we get the measures $Sp_{\mathfrak{T}}(F \vdash 0)$ and $VarSp_{\mathfrak{T}}(F \vdash 0)$ in analogy with $L_{\mathfrak{T}}(F \vdash 0)$ defined above.

In this paper, we will be almost exclusively interested in the clause space of general resolution refutations. When we write simply "space" (for brevity), we mean clause space.

All contradictory CNF formulas can be refuted in clause space linear in the formula size. This is stated more precisely in the following theorem.

Theorem 3.5 (see [29]). *Any unsatisfiable CNF formula* F *can be refuted in clause space* $Sp(F \vdash 0) \leq \max\bigl\{|Vars(F)| + 2, |F| + 1\bigr\}$.

Hence, the interesting question is which formulas demand this much space, and which formulas can be refuted in, for instance, logarithmic or even constant space. It has been shown that there are polynomial-size formulas that meet the upper bounds of Theorem 3.5 up to a multiplicative constant.

Theorem 3.6 (see [3, 51]). *There is a polynomial-size family* $\{F_n\}_{n=1}^{\infty}$ *of unsatisfiable* 3*-CNF formulas such that* $Sp(F \vdash 0) = \Omega(|F|) = \Omega(|Vars(F)|)$.

Lower bounds on clause space have been presented for a number of different CNF formula families [3, 14, 51]. As was mentioned above, in these papers it was

[3] In some previous papers, resolution is defined so as to allow every derivation step to *combine* one or zero applications of each of the three derivation rules. Therefore, some of the bounds stated in this paper for space as defined next are off by a constant as compared to the cited sources.

observed that the lower bounds on refutation space coincided with the lower bounds on refutation width. This lead to the conjecture that the width measure is a lower bound for the clause space measure, a conjecture that was proven true by Atserias and Dalmau.

THEOREM 3.7 (see [7]). *Let F be an arbitrary unsatisfiable CNF formula. Then it holds that $Sp(F \vdash 0) - 3 \geq W(F \vdash 0) - W(F)$.*[4]

In other words, the extra clause space exceeding the minimum 3 needed for any resolution derivation is bounded from below by the extra width exceeding the width of the formula. An immediate corollary of this theorem is that for polynomial-size k-CNF formulas, constant clause space implies polynomial proof length.

Thus, upper bounds on clause space imply upper bounds on length and width. As was discussed in the introduction, it has remained open what holds in the other direction. In particular, a natural follow-up question to [7] is whether the space and width measures coincide asymptotically or whether there is a formula family separating them. We remark that in order for this question to be interesting, we should restrict our attention to families of k-CNF formulas. Any resolution refutation of an unsatisfiable CNF formula F with *minimum* clause width k can be shown to require clause space at least $k+2$ (see [29]), so it is easy to find CNF formulas $\{F_n\}_{n=1}^{\infty}$ of growing width such that $W(F_n \vdash 0) - W(F_n) = \mathrm{O}(1)$ but $Sp(F_n \vdash 0) = \Omega(n)$.

The main contribution of this paper is that we settle the open question of the relationship between space and width by proving a separation of the two measures. More precisely, we show that there is a family of k-CNF formulas $\{F_n\}_{n=1}^{\infty}$ such that $W(F_n \vdash 0) = \mathrm{O}(1)$ but $Sp(F_n \vdash 0) = \Theta(\log |F_n|) = \omega(1)$.

4. Pebble games and pebbling contradictions.

Pebble games were devised for studying programming languages and compiler construction but have found a variety of applications in computational complexity theory. In connection with resolution, pebble games have been employed both to analyze resolution derivations with respect to how much memory they consume (using the original definition of space in [29]) and to construct CNF formulas which are hard for different variants of resolution in various respects (see, for example, [4, 15, 18, 20]). An excellent survey of pebbling up to 1980 is [43].

4.1. Pebble games.

The black pebbling price of a DAG G captures the memory space, i.e., the number of registers, required to perform the deterministic computation described by G. The space of a nondeterministic computation is measured by the black-white pebbling price of G. We say that vertices of G with indegree 0 are *sources* and that vertices with outdegree 0 are *sinks* or *targets*. The next definition is adapted from [25], though we use the established pebbling terminology introduced by [34].

DEFINITION 4.1 (pebble game). *Suppose that G is a DAG with sources S and a unique target z. The* black-white pebble game *on G is the following one-player game. At any point in the game, there are black and white pebbles placed on some vertices of G, at most one pebble per vertex. A* pebble configuration *is a pair of subsets $\mathbb{P} = (B, W)$ of $V(G)$, comprising the black-pebbled vertices B and white-pebbled vertices W. The rules of the game are as follows:*

1. *If all immediate predecessors of an empty vertex v have pebbles on them, a black pebble may be placed on v. In particular, a black pebble can always be placed on any vertex in S.*

[4] The statement of the theorem in [7] is $Sp(F \vdash 0) \geq W(F \vdash 0) - W(F)$, but this can be sharpened by a constant if one does the calculations carefully.

2. *A black pebble may be removed from any vertex at any time.*
3. *A white pebble may be placed on any empty vertex at any time.*
4. *If all immediate predecessors of a white-pebbled vertex v have pebbles on them, the white pebble on v may be removed. In particular, a white pebble can always be removed from a source vertex.*

A complete black-white pebbling *of G is a sequence of pebble configurations $\mathcal{P} = \{\mathbb{P}_0, \ldots, \mathbb{P}_\tau\}$ such that $\mathbb{P}_0 = (\emptyset, \emptyset)$, $\mathbb{P}_\tau = (\{z\}, \emptyset)$, and for all $t \in [\tau]$, $\mathbb{P}_t$ follows from $\mathbb{P}_{t-1}$ by one of the rules above. The* cost *of a pebbling $\mathcal{P}$, denoted $\mathsf{cost}(\mathcal{P})$, is the maximal number of pebbles $|B_t \cup W_t|$ used in any configuration $\mathbb{P}_t = (B_t, W_t)$ of $\mathcal{P}$. The* black-white pebbling price *$\mathsf{BW\text{-}Peb}(G)$ of G is the minimum cost of any complete pebbling of G.*

A complete black pebbling of G *is a complete pebbling using black pebbles only, i.e., $W_t = \emptyset$ for all t, and the* (black) pebbling price *of G, denoted $\mathsf{Peb}(G)$, is the minimum cost of any complete black pebbling of G.*

We think of the moves in a pebbling as occurring at integral time intervals $t = 1, 2, \ldots$ and talk about the pebbling move "at time t" (which is the move resulting in configuration $\mathbb{P}_t$) or the moves "during the time interval $[t_1, t_2]$."

We make an observation that the black-white pebble game has the following "antisymmetry" property, which will be helpful for our intuition in what comes later.

Proposition 4.2 (see [25]). *Suppose that $\mathcal{P}$ is a black-white pebbling of a DAG G starting with no pebbles and ending with a black pebble on the target z. Then by reversing the sequence of moves and switching the colors of the pebbles, one gets a dual pebbling $\overline{\mathcal{P}}$ of G starting with a white pebble on z and ending with no pebbles in the DAG.*

The proof immediately follows from Definition 4.1, observing that the rules for placing and removing a black pebble are the duals of the rules for removing and placing a white pebble, respectively.

In this paper we will consider pebblings of complete binary trees. We let T denote a complete binary tree considered as a DAG with edges directed toward the root. We write T_h when we want to specify that the height of the tree is h. We use z to denote the unique target vertex of T, i.e., the root.

The black pebbling price of T_h can be established by an easy induction over the tree height. For black-white pebbling, general bounds for the pebbling price of trees of any arity were presented in [40], and for the case of binary trees, we can simplify this result to an exact equality. We collect the results on black and black-white pebbling prices of complete binary trees in a theorem for reference.

Theorem 4.3. *$\mathsf{Peb}(T_h) = h + 2$ and $\mathsf{BW\text{-}Peb}(T_h) = \left\lfloor \frac{h}{2} \right\rfloor + 3$.*

A proof of the black-white pebbling price bound can be found in section 4 of [41].

4.2. Pebbling contradictions. A *pebbling contradiction* defined on a DAG G encodes the pebble game on G by defining the sources to be true and the target to be false and specifying that truth propagates through the graph according to the pebbling rules. The definition below is a generalization of formulas previously studied in [18, 45].

Definition 4.4 (pebbling contradiction [16]). *Let G be a DAG with sources S and a unique target z and with all vertices having indegree 0 or 2, and let $d \in \mathbb{N}^+$. Associate d distinct variables $x(v)_1, \ldots, x(v)_d$ with every vertex $v \in V(G)$. The dth degree* pebbling contradiction *on G, denoted Peb_G^d, is the conjunction of the following clauses:*

- $\bigvee_{i=1}^{d} x(s)_i$ *for all $s \in S$ (*source axioms*),*

- $\overline{x(z)}_i$ *for all* $i \in [d]$ *(*target axioms*),*
- $\overline{x(u)}_i \vee \overline{x(v)}_j \vee \bigvee_{l=1}^{d} x(w)_l$ *for all* $i, j \in [d]$ *and all* $w \in V(G) \setminus S$*, where* u, v *are the two predecessors of* w *(*pebbling axioms*).*

The formula Peb_G^d is a $(2{+}d)$-CNF formula with $\mathrm{O}\big(d^2 \cdot |V(G)|\big)$ clauses over $d \cdot |V(G)|$ variables. A small example of a pebbling contradiction is presented in Figure 2 on page 63.

We first observe that pebbling contradictions are indeed unsatisfiable. As shown in [15], Peb_G^d can be refuted in resolution by deriving $\bigvee_{i=1}^{d} x(v)_i$ for all $v \in V(G)$ inductively in topological order and then resolving with the target axioms $\overline{x(z)}_i$, $i \in [d]$. Writing this resolution proof, one gets the following theorem (which is proven together with Theorem 4.8 below).

THEOREM 4.5 (see [15]). *For any DAG G with all vertices having indegree* 0 *or* 2*, there is a resolution refutation* $\pi : Peb_G^d \vdash 0$ *in length* $L(\pi) = \mathrm{O}\big(d^2 \cdot |V(G)|\big)$ *and width* $W(\pi) = \mathrm{O}(d)$*.*

Tree-like resolution is good at refuting pebbling contradictions Peb_G^1 but is bad at refuting Peb_G^d for $d \geq 2$.

THEOREM 4.6 (see [13]). *For any DAG G with all vertices having indegree* 0 *or* 2*, there is a tree-like resolution refutation π of Peb_G^1 such that* $L(\pi) = \mathrm{O}(|V(G)|)$ *and* $Sp(\pi) = \mathrm{O}(1)$*.*

THEOREM 4.7 (see [15]). *For any DAG G with all vertices having indegree* 0 *or* 2*,* $L_{\mathfrak{T}}\big(Peb_G^2 \vdash 0\big) = 2^{\Omega(\mathsf{Peb}(G))}$*.*

As to refutation space, it is not too difficult to see that the space of refuting Peb_G^d is upper-bounded by the black pebbling price of G, using an optimal black pebbling of G together with the resolution refutation from [15] sketched above.

THEOREM 4.8. *For any DAG G with vertex indegrees* 0 *or* 2*,* $Sp\big(Peb_G^d \vdash 0\big) \leq \mathsf{Peb}(G) + \mathrm{O}(1)$*.*

Since we need the upper bounds on width and space in Theorems 4.5 and 4.8 in our main theorem, we present the proofs for completeness.

Proof of Theorems 4.5 *and* 4.8. Consider first the bound on space.

Given a black pebbling of G, we construct a resolution refutation of Peb_G^d such that if at some point in time there are black pebbles on a set of vertices V, then we have the clauses $\big\{\bigvee_{i=1}^{d} x(v)_i \mid v \in V\big\}$ in memory. When some new vertex v is pebbled, we derive $\bigvee_{i=1}^{d} x(v)_i$ from the clauses already in memory. We claim that with a little care, this can be done in constant extra space independent of d. When a black pebble is removed from v, we erase the clause $\bigvee_{i=1}^{d} x(v)_i$. We conclude the resolution proof by resolving $\bigvee_{i=1}^{d} x(z)_i$ for the target z with all target axioms $\overline{x(z)}_i$, $i \in [d]$, in space 3.

It is clear that given our claim about the constant extra space needed when a vertex is black-pebbled, this yields a resolution refutation in space equal to the pebbling cost plus some constant. In particular, given an optimal black pebbling of G, we get a refutation in space $\mathsf{Peb}(G) + \mathrm{O}(1)$.

To prove the claim, note first that it trivially holds for source vertices v, since $\bigvee_{i=1}^{d} x(v)_i$ is an axiom of the formula. Suppose for a nonsource vertex r with predecessors p and q that at some point in time a black pebble is placed on r. Then p and q must be black-pebbled, so by induction we have the clauses $\bigvee_{i=1}^{d} x(p)_i$ and $\bigvee_{j=1}^{d} x(q)_j$ in memory. It is not hard to verify that $\overline{x(p)}_i \vee \bigvee_{l=1}^{d} x(r)_l$ can be derived in additional space 3 by resolving $\bigvee_{j=1}^{d} x(q)_j$ with $\overline{x(p)}_i \vee \overline{x(q)}_j \vee \bigvee_{l=1}^{d} x(r)_l$ for $j \in [d]$. Resolve $\bigvee_{i=1}^{d} x(p)_i$ with $\overline{x(p)}_1 \vee \bigvee_{l=1}^{d} x(r)_l$ to get $\bigvee_{i=2}^{d} x(p)_i \vee \bigvee_{l=1}^{d} x(r)_l$, and then resolve this clause with $\overline{x(p)}_i \vee \bigvee_{l=1}^{d} x(r)_l$ for $i = 2, \ldots, d$ to get $\bigvee_{l=1}^{d} x(r)_l$ in total extra space 4.

It is easy to see that this proof has width $\mathrm{O}(d)$, which proves the claim about width in Theorem 4.5. To get the claim about length, we observe that the subderivation needed when a vertex is black-pebbled has length $\mathrm{O}(d^2)$. If we use a pebbling that black-pebbles all vertices once in topological order without ever removing a pebble, we thus get a refutation in length $L(\pi) = \mathrm{O}(d^2 \cdot |V(G)|)$. ∎

Theorem 4.8 is not quite an optimal strategy with respect to clause space, though. For binary trees, [30] demonstrated that we can do at least a little bit better.

THEOREM 4.9 (see [30]). $Sp(Peb^2_{T_h} \vdash 0) \leq \lceil \frac{2h+1}{3} \rceil + 3 = \frac{2}{3}\mathsf{Peb}(G) + \mathrm{O}(1)$.

It is not known if the bounds in Theorems 4.8 and 4.9 are tight, since no corresponding lower bound on $Sp(Peb^d_G \vdash 0)$ has been shown for pebbling degree $d \geq 2$ in general resolution (in terms of pebbling price or otherwise). The only previously known lower bound on the refutation clause space of pebbling contradictions is a bound $Sp_{\mathfrak{T}}(Peb^d_{T_h} \vdash 0) = h + \mathrm{O}(1)$ for the special case of tree-like resolution [30]. Unfortunately, this does not tell us anything about general resolution. For tree-like resolution, if the only way of deriving D is from clauses C_1, C_2 such that $Sp_{\mathfrak{T}}(F \vdash C_i) \geq s$, then $Sp_{\mathfrak{T}}(F \vdash D) \geq s + 1$ since one of the clauses C_i must be kept in memory while deriving the other clause. This seems to be very different from how general resolution works with respect to space.

However, the resolution refutation of $Peb^2_{T_h}$ in [30] used to prove Theorem 4.9 is structurally quite similar to the optimal black-white pebbling of T_h presented in [40], and it is hard to see how any resolution refutation could do better. This raises the suspicion that the black-white pebbling price $\mathsf{BW\text{-}Peb}(T_h) = h/2 + \mathrm{O}(1)$ might be a lower bound for $Sp(Peb^d_{T_h} \vdash 0)$, and in general that $Sp(Peb^d_G \vdash 0) \geq \mathsf{BW\text{-}Peb}(G)$ for any DAG G and $d \geq 2$.

This suspicion is somewhat strengthened by the fact that for variable space, we do have a lower bound for general resolution.[5]

THEOREM 4.10 (see [13]). *For any $d \in \mathbb{N}^+$, $VarSp(Peb^d_G \vdash 0) \geq \mathsf{BW\text{-}Peb}(G)$.*

If the refutation clause space of pebbling contradictions would be constant, Theorem 4.10 would imply that as $\mathsf{BW\text{-}Peb}(G)$ grows larger, the clauses in memory get wider, and thus weaker. Still it would somehow be possible to derive a contradiction from a constant number of these clauses of unbounded width. This appears counterintuitive.

On the other hand, for one variable per vertex, i.e., $d = 1$, refutations of Peb^1_G in constant space have exactly these "counterintuitive" properties. The resolution refutation of Peb^1_G in [13] is constructed by first downloading the pebbling axiom for the target z and then moving the false literals downward by resolving with pebbling axioms for vertices $v \in V(G) \setminus S$ in reverse topological order. This finally yields a clause $\bigvee_{v \in S} \overline{x(v)}_1 \vee x(z)_1$ of width $|S| + 1$, which can be eliminated by resolving with the source axioms $x(v)_1$ one by one for all $v \in S$ and then with the target axiom $\overline{x(z)}_1$ to yield the empty clause 0.

If we want to establish a nonconstant lower bound on $Sp(Peb^d_G \vdash 0)$ for $d \geq 2$, we have to pin down why this case is different. Intuitively, the difference is that with only one variable per vertex, a single clause $\overline{x(v_1)}_1 \vee \cdots \vee \overline{x(v_m)}_1$ can express the disjunction of the falsity of an arbitrary number of vertices $v_1, \ldots, v_m$, but for $d = 2$, the straightforward way of expressing that both variables $x(v_i)_1$ and $x(v_i)_2$ are false for at least one out of m vertices requires $\Omega(m)$ clauses.

[5] To be precise, the result in [13] is for $d = 1$, but the proof generalizes easily to any $d \in \mathbb{N}^+$.

As was argued in section 2, to prove a lower bound on the refutation clause space of pebbling contradictions, it seems natural to try to interpret resolution refutations of Peb^d_G in terms of pebblings of the underlying graph G. Let us say that a vertex v is "true" if $\bigvee_{i=1}^{d} x(v)_i$ has been derived and "false" if $\overline{x(v)}_i$ has been derived for all $i \in [d]$. Any resolution proof refutes a pebbling contradiction by deriving that some vertex v is both true and false, and then resolves to get 0. If we let w be any vertex with predecessors u, v, we see that if we have derived that u and v are true, by downloading $\overline{x(u)}_i \vee \overline{x(v)}_j \vee \bigvee_{l=1}^{d} x(w)_l$ for all $i, j \in [d]$ we can derive $\bigvee_{l=1}^{d} x(w)_l$. This appears analogous to the rule that if u and v are black-pebbled we can place a black pebble on w. In the opposite direction, if we know $\overline{x(w)}_l$ for all $l \in [d]$, using the axioms $\overline{x(u)}_i \vee \overline{x(v)}_j \vee \bigvee_{l=1}^{d} x(w)_l$ we can derive that either u or v is false. This looks similar to eliminating a white pebble on w by placing white pebbles on the predecessors u and v, and then removing the pebble from w. Generalizing this loose, intuitive reasoning, we argue that a set of black-pebbled vertices V should correspond to the derived conjunction of truth of all $v \in V$, and that a set of white-pebbled vertices W should correspond to the derived disjunction of falsity of some $w \in W$.

Suppose that we could show that as the resolution derivation proceeds, the black and white pebbles corresponding to different clause configurations as outlined above move about on the vertices of G in accordance with the rules of the pebble game. If so, we would get that there is some clause configuration $\mathbb{C}$ corresponding to a lot of pebbles. This could in turn hopefully yield a nonconstant lower bound for the refutation clause space. For if $\mathbb{C}$ induces N black pebbles, i.e., implies N disjoint clauses, it seems likely that $|\mathbb{C}|$ should be linear in N. And if $\mathbb{C}$ induces N white pebbles, $|\mathbb{C}|$ should grow with N if $d \geq 2$, since $\mathbb{C}$ has to force d literals false simultaneously for one out of N vertices. This is the guiding intuition behind the result proven in this paper.

5. Modifying the black-white pebble game.

To prove a lower bound on the refutation space of pebbling contradictions, we want to interpret resolution derivation steps in terms of pebble placements and removals in the corresponding graph. At the end of the previous section, we outlined an intuitive correspondence between clauses and pebbles. The formal translation from sets of clauses to sets of black and white pebbles, which is presented in section 6, reflects this intuition (and the example in Figure 3 on page 64) quite faithfully. However, the pebble configurations that result when we apply this translation on a resolution derivation do not obey the rules of the black-white pebble game. Therefore, we are forced to change the pebbling rules.

In this section, we present the modified pebble game used for analyzing resolution derivations. We then argue that for binary trees, we get essentially the same bound on pebbling price in this new pebble game as in the black-white pebble game of Definition 4.1.

Our first modification of the pebble game is to change the rule for white pebble removal so that a white pebble can be removed from a vertex when a black pebble is placed on that same vertex. This will make the correspondence between pebblings and resolution derivations much more natural. Clearly, this is only a minor adjustment, and it is easy to prove formally that it does not really change anything.

Our second, and far more substantial, modification of the pebble game is motivated by the fact that in general, a resolution refutation has no obvious reason to follow our pebble game intuition sketched at the end of section 4.2. Since pebbles are induced by clauses, if at some derivation step the refutation chooses to erase "the wrong clause" from the point of view of the induced pebble configuration, this can lead to pebbles just disappearing. This is all in order for black pebbles, but if we allow uncontrolled removal of white pebbles, we cannot hope for any nontrivial lower bounds on pebbling price (just white-pebble the two predecessors of the target, then black-pebble the target itself, and finally remove the white pebbles).

Our solution to this problem is to keep track of exactly which white pebbles have been used to get a black pebble on a vertex. Loosely put, removing a white pebble from a vertex v without placing a black pebble on the same vertex should be in order, provided that all black pebbles placed on vertices above v in the DAG with the help of the white pebble on v are removed as well. We define a pebble *subconfiguration* to consist of a black pebble together with all the white pebbles this black pebble depends on, and require that if a white pebble in a subconfiguration is removed, then all pebbles in this subconfiguration must be removed.

Another problem is that some resolution derivation steps can lead to what looks like "backward" pebbling moves, with white pebbles moving upward and black pebbles downward in the DAG. This problem turns out to be even more serious. We try to get around it by introducing an order relation on pebble subconfigurations, where the intuition is that "stronger" pebble subconfigurations are "closer" to the final goal of getting the target black-pebbled. Using this order relation, the backward pebbling moves can be characterized as moves from stronger to weaker pebble subconfigurations, so we add a pebbling rule allowing such moves.

To define this modified pebble game formally, we need some notation and terminology. We use z to denote the unique target vertex of the DAG G, $p, q, r, s, u, v, w, x, y$ to denote arbitrary vertices, and U, V, W to denote arbitrary subsets of vertices. We let $succ(v)$ denote the immediate successor of v and $pred(v)$ the immediate predecessors. For a leaf v we have $pred(v) = \emptyset$, and for the target z we have $succ(z) = \emptyset$. We say that w is *below* v if there is a path from w to v and *above* v if there is a path from v to w. If in addition $v \neq w$, the vertex w is said to be *strictly* below/above v. We say that v and w are *unrelated* if v is neither above nor below w. The vertex set W is *(strictly) below* v if all $w \in W$ are (strictly) below v.

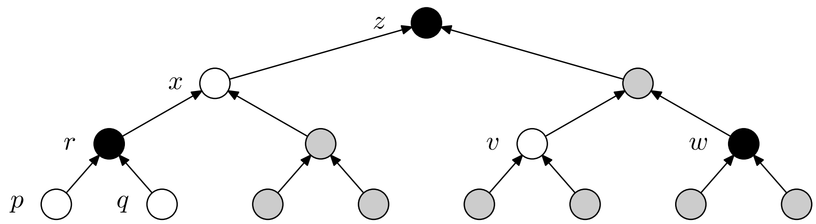
We now present the concept used to "label" each black pebble with the set of white pebbles (if any) that this black pebble is dependent on. The intuition behind the next definition is that $v\langle W\rangle$ should denote a black pebble on v together with the white pebbles W below v with the help of which we have been able to place the black pebble on v.

DEFINITION 5.1 (pebble subconfiguration). *For a vertex v and a set of vertices W strictly below v, we say that $v\langle W\rangle$ is a* pebble subconfiguration *with a black pebble on v supported by white pebbles on $w \in W$. The black pebble on v in $v\langle W\rangle$ is said to be* dependent *on the white pebbles in W. We refer to $v\langle\emptyset\rangle$ as an* independent black pebble.

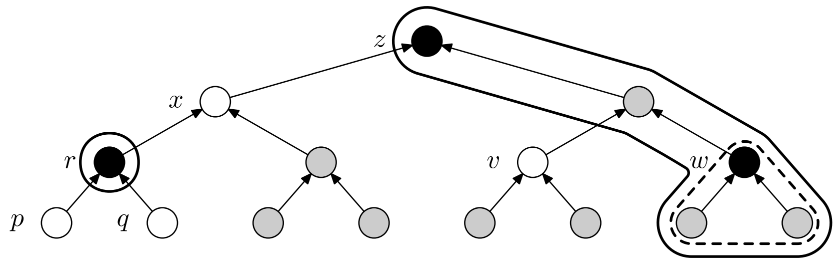
The cover *of $v\langle W\rangle$, denoted $cover(v\langle W\rangle)$, consists of all vertices U such that there is a path $P : u \rightsquigarrow v$ from $u \in U$ to v that does not intersect W, i.e., $P \cap W = \emptyset$. If $cover(v_1\langle W_1\rangle) \subseteq cover(v_2\langle W_2\rangle)$, we say that $v_1\langle W_1\rangle$ is* covered by *$v_2\langle W_2\rangle$ and write $v_1\langle W_1\rangle \preceq v_2\langle W_2\rangle$. If $cover(v_1\langle W_1\rangle) \subsetneqq cover(v_2\langle W_2\rangle)$, we write $v_1\langle W_1\rangle \prec v_2\langle W_2\rangle$.*

We use $\mathbb{L}$ to denote a set of pebble subconfigurations and refer to such a set as a labeled pebble configuration *or an* L-configuration. *The cover of an L-configuration $\mathbb{L}$ is defined as $cover(\mathbb{L}) = \bigcup_{v\langle W\rangle\in\mathbb{L}} cover(v\langle W\rangle)$, and we write $\mathbb{L}_1 \preceq \mathbb{L}_2$ if $cover(\mathbb{L}_1) \subseteq cover(\mathbb{L}_2)$.*

In the following, when we specify the set W of white-pebbled vertices in $v\langle W\rangle$ by enumerating the members of W, we will abuse notation somewhat by omitting the curly brackets inside $\langle$ and $\rangle$ around this set.



(a) *The pebble subconfigurations $z\langle x, v\rangle$, $r\langle p, q\rangle$, and $w\langle\emptyset\rangle$.*



(b) *The covered vertices $cover(z\langle x, v\rangle)$, $cover(r\langle p, q\rangle)$, and $cover(w\langle\emptyset\rangle)$ (dashed).*

FIG. 4. *Three pebble subconfigurations and their covered vertices.*

For an illustration of Definition 5.1, see Figure 4. Note that $w\langle\emptyset\rangle \prec z\langle x, v\rangle$ since $cover(w\langle\emptyset\rangle) \subsetneq cover(z\langle x, v\rangle)$ (see Figure 4(b)). We remark that $\preceq$ is an order relation on pebble subconfigurations, as the notation suggests, and that the minimal elements are subconfigurations $v\langle pred(v)\rangle$.

Our modified pebble game is defined in terms of moves not of individual pebbles, but of entire pebble subconfigurations. In this pebble game, a black pebble on v is always placed together with white pebbles on $pred(v)$ below (except for at the leaves where $pred(v) = \emptyset$). Removals of white pebbles are always allowed, but since we can remove only a whole subconfiguration, the removal rule ensures that any black pebble dependent on the removed white pebbles is removed as well. A "traditional" removal of a white pebble from w corresponds to merging two subconfigurations $v\langle V\rangle$ and $w\langle W\rangle$ into $v\langle (V \cup W) \setminus \{w\}\rangle$ and then erasing $v\langle V\rangle$ and $w\langle W\rangle$ (see Figure 5 for an example). Finally, we allow *reversal* moves to weaker subconfigurations. The formal definition is as follows.

DEFINITION 5.2 (labeled pebble game). *For G any DAG with unique target z, a* labeled pebbling*, or* L-pebbling*, on G is a sequence $\mathcal{L} = \{\mathbb{L}_0, \ldots, \mathbb{L}_\tau\}$ of labeled pebble configurations such that for all t it holds that $\mathbb{L}_t \neq \mathbb{L}_{t+1}$ and $\mathbb{L}_{t+1}$ is obtained from $\mathbb{L}_t$ by one of the following rules:*

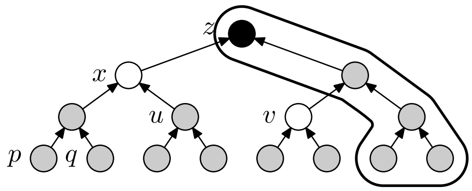
Introduction. $\mathbb{L}_{t+1} = \mathbb{L}_t \cup \{v\langle pred(v)\rangle\}$.

Erasure. $\mathbb{L}_{t+1} = \mathbb{L}_t \setminus \{v\langle V\rangle\}$ *for* $v\langle V\rangle \in \mathbb{L}_t$.

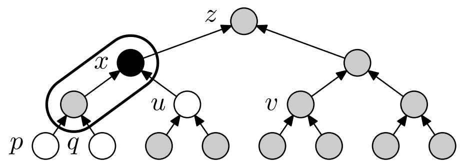
Merger. $\mathbb{L}_{t+1} = \mathbb{L}_t \cup \big\{v\langle (V \cup W) \setminus \{w\}\rangle\big\}$ *for* $v\langle V\rangle, w\langle W\rangle \in \mathbb{L}_t$ *with* $w \in V$*. We denote this subconfiguration* $\mathsf{merge}(v\langle V\rangle, w\langle W\rangle)$*, where the pair of subconfigurations* $v\langle V\rangle, w\langle W\rangle$ *is always ordered so that* $w \in V$*, and refer to it as a* merger *on* w.

Reversal. $\mathbb{L}_{t+1} = \mathbb{L}_t \cup \{v\langle V\rangle\}$ *if* $v\langle V\rangle \preceq u\langle U\rangle$ *for some* $u\langle U\rangle \in \mathbb{L}_t$.

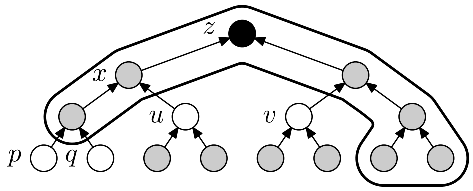
Let $Bl(\mathbb{L}_t) = \bigcup\{v \mid v\langle W\rangle \in \mathbb{L}_t\}$ denote the set of all black pebbles in $\mathbb{L}_t$ and $Wh(\mathbb{L}_t) = \bigcup\{W \mid v\langle W\rangle \in \mathbb{L}_t\}$ the set of all white pebbles. Then the cost *of an L-configuration $\mathbb{L}$ is $\mathsf{cost}(\mathbb{L}) = |Bl(\mathbb{L}) \cup Wh(\mathbb{L})|$, and the cost of an L-pebbling $\mathcal{L} = \{\mathbb{L}_0, \ldots, \mathbb{L}_\tau\}$ is $\max_{t\in[\tau]}\{\mathsf{cost}(\mathbb{L}_t)\}$.*



(a) $z\langle x, v\rangle$ *and cover*$(z\langle x, v\rangle)$.



(b) $x\langle p, q, u\rangle$ *and cover*$(x\langle p, q, u\rangle)$.



(c) *The merger* $z\langle p, q, u, v\rangle$ *with cover.*

FIG. 5. *Two pebble subconfigurations and their merger with covered vertices.*

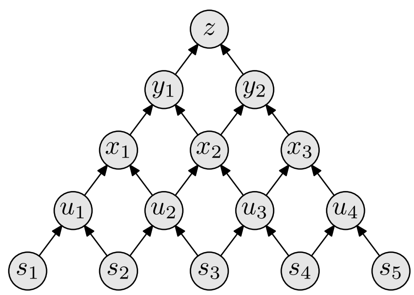


FIG. 6. *Example pyramid* Π_4 *of height* 4 *in proof of Lemma* 5.3.

A complete labeled pebbling *of G is an L-pebbling $\mathcal{L}$ such that $\mathbb{L}_0 = \emptyset$ and $\mathbb{L}_\tau = \{z\langle\emptyset\rangle\}$. The* labeled pebbling price *of G, denoted $\textsf{L-Peb}(G)$, is the minimum cost of any complete L-pebbling of G.*

The "backward" pebbling moves mentioned at the beginning of this section are moves according to the reversal rule. It can be shown that the L-pebble game without reversal moves is essentially just a disguised version of the ordinary black-white pebble game. Arguing very informally, it seems plausible that making reversals in an L-pebbling should only "weaken" the pebble configurations (for example, reversing from $z\langle x, v\rangle$ to $w\langle\emptyset\rangle$ in Figure 4), and that it should therefore be possible to eliminate all reversal moves from a pebbling without affecting the pebbling cost.

Unfortunately, this intuition does not hold in general.

LEMMA 5.3. *There are families of DAGs $\{G_n\}_{n=1}^{\infty}$ such that $\textsf{BW-Peb}(G_n)$ goes to infinity with n but $\textsf{L-Peb}(G_n)$ is constant.*

Proof. Consider the pyramid graphs Π_h (with Π_4 shown in Figure 6). Klawe [35] showed that $\textsf{BW-Peb}(\Pi_h) = h/2 + \mathrm{O}(1)$. We prove by induction that Π_h can be L-pebbled with 4 pebbles if we allow reversal moves of black pebbles downward.

The base case for a pyramid of height 1 is clear.

For the induction step, suppose that we have been able to get to the pebble subconfiguration $y_2\langle\emptyset\rangle$ in Figure 6 in L-pebbling cost at most 4. We show how to get a black pebble on the target z by an introduction move and then move the white pebbles downward one level at a time until we reach the sources.

Introducing $z\langle y_1, y_2\rangle$ and merging $z\langle y_1, y_2\rangle$ with $y_2\langle\emptyset\rangle$ on y_2, we get $z\langle y_1\rangle$. Next, reverse $y_2\langle\emptyset\rangle$ to $x_2\langle\emptyset\rangle$ (this is a legal reversal move, since $\mathit{cover}(x_2\langle\emptyset\rangle) \subseteq \mathit{cover}(y_2\langle\emptyset\rangle)$). Conclude this first subsequence of L-pebbling moves by erasing $z\langle y_1, y_2\rangle$ and $y_2\langle\emptyset\rangle$.

Now we have the L-configuration $\{z\langle y_1\rangle, x_2\langle\emptyset\rangle\}$. Introduce $y_1\langle x_1, x_2\rangle$, merge $z\langle y_1\rangle$ and $y_1\langle x_1, x_2\rangle$ on y_1 resulting in $z\langle x_1, x_2\rangle$, and erase $z\langle y_1\rangle$ and $y_1\langle x_1, x_2\rangle$. Then merge $z\langle x_1, x_2\rangle$ and $x_2\langle\emptyset\rangle$ on x_2 to get $z\langle x_1\rangle$. As above, conclude the subsequence of moves by reversing $x_2\langle\emptyset\rangle$ to $u_2\langle\emptyset\rangle$ and then erasing $z\langle x_1, x_2\rangle$ and $x_2\langle\emptyset\rangle$.

The next round of moves is entirely analogous: Introduce $x_1\langle u_1, u_2\rangle$, merge with $z\langle x_1\rangle$ to get $z\langle u_1, u_2\rangle$, and then erase the merged subconfigurations. Then merge $z\langle u_1, u_2\rangle$ with $u_2\langle\emptyset\rangle$ resulting in $z\langle u_1\rangle$, reverse $u_2\langle\emptyset\rangle$ to $s_2\langle\emptyset\rangle$, and erase $z\langle u_1, u_2\rangle$ and $u_2\langle\emptyset\rangle$.

At the start of the final subsequence of moves, we have the L-configuration $\{z\langle u_1\rangle, s_2\langle\emptyset\rangle\}$. Introducing $u_1\langle s_1, s_2\rangle$ and merging this subconfiguration with $z\langle u_1\rangle$ on u_1 result in $z\langle s_1, s_2\rangle$. Introducing $s_1\langle\emptyset\rangle$ and merging $z\langle s_1, s_2\rangle$ with $s_1\langle\emptyset\rangle$ and then $s_2\langle\emptyset\rangle$, we get $z\langle\emptyset\rangle$.

The cost of this pebbling is 4, and it is easy to see that it generalizes to pyramids of arbitrary height. ∎

For binary trees, however, we can prove that the L-pebbling price and the black-white pebbling price coincide asymptotically.

THEOREM 5.4. *For a complete binary tree T, $\textsf{L-Peb}(T) = \Theta\big(\textsf{BW-Peb}(T)\big)$.*

The technically quite complicated proof of this fact, which is a cornerstone of our result, is presented in the appendix.

Given Theorem 5.4, the lower bound on the L-pebbling price in Theorem 2.3 on page 65 follows.

THEOREM 2.3 (restated). $\textsf{L-Peb}(T_h) = \Omega(h)$.

Proof. Theorem 5.4 says that $\textsf{L-Peb}(T_h) = \Theta\big(\textsf{BW-Peb}(T_h)\big)$, and Theorem 4.3 on page 70 says that $\textsf{BW-Peb}(T_h) = \Theta(h)$. ∎

6. Resolution derivations induce labeled pebblings. The next step in our proof is to show that sets of clauses can be interpreted in terms of pebble configurations in such a way that resolution derivations induce legal labeled pebblings.

For simplicity, from now on let us write $v_1, \ldots, v_d$ instead of $x(v)_1, \ldots, x(v)_d$ for the d variables associated with the vertex v in a dth degree pebbling contradiction.

DEFINITION 6.1. *Assume that G is a DAG with a unique target z and all vertices having indegree 0 or 2. Then we define $^*Peb^d_G = Peb^d_G \setminus \{\overline{z}_1, \ldots, \overline{z}_d\}$ to be the pebbling contradiction with target axioms removed. If $pred(r) = \{p, q\}$, the* axioms *for r are the set $Ax^d(r) = \{\overline{p}_i \lor \overline{q}_j \lor \bigvee_{l=1}^{d} r_l \mid i, j \in [d]\}$, and for r a source we let $Ax^d(r) = \{\bigvee_{i=1}^{d} r_i\}$. For a set of vertices V, we define $Ax^d(V) = \{Ax^d(v) \mid v \in V\}$.*

Let us first observe that instead of refutations of Peb^d_G, we can just as well study derivations of $\bigvee_{i=1}^{d} z_i$ from $^*Peb^d_G$. This will help us to avoid some artificial technicalities when defining the correspondence between resolution derivations and L-pebblings.

LEMMA 6.2. *For any DAG G with a unique target z and all vertices having indegree 0 or 2, it holds that $Sp\big(Peb^d_G \vdash 0\big) = Sp\big({}^*Peb^d_G \vdash \bigvee_{l=1}^{d} z_l\big)$. In particular, for every resolution refutation $\pi : Peb^d_G \vdash 0$ we can find a resolution derivation $\pi^* : {}^*Peb^d_G \vdash \bigvee_{l=1}^{d} z_l$ in the same space.*

Proof. For any resolution derivation $\pi^* : {}^*Peb^d_G \vdash \bigvee_{l=1}^{d} z_l$, we can get a resolution refutation of Peb^d_G from π^* in the same space by resolving $\bigvee_{l=1}^{d} z_l$ with all $\overline{z}_l$, $l = 1, \ldots, d$, in space 3.

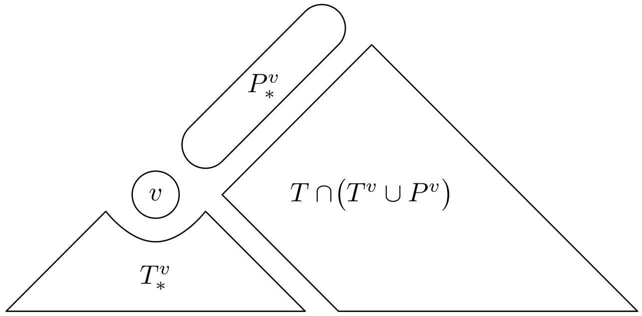


FIG. 7. *Referencing sets of vertices of a tree T relative to a vertex $v \in V(T)$.*

In the other direction, for $\pi : Peb_G^d \vdash 0$ we can extract a derivation of $\bigvee_{l=1}^{d} z_l$ in at most the same space by simply omitting all downloads of and resolution steps on $\overline{z}_l$ in π, leaving the literals z_l in the clauses. Instead of the final empty clause 0 we get some clause $D \subseteq \bigvee_{l=1}^{d} z_l$, and since $^*Peb_T^d \nvDash D \subsetneqq \bigvee_{l=1}^{d} z_l$ and resolution is sound, we have $D = \bigvee_{l=1}^{d} z_l$. ∎

Now we try to develop some intuition for how clause configurations in a resolution derivation of $\bigvee_{i=1}^{d} z_i$ from $^*Peb_G^d$ should be translated into pebble configurations in the L-pebble game. Since we know from Lemma 5.3 that we cannot hope to get lower bounds for refutation space of pebbling contradictions over general DAGs by using the L-pebble game, from now on we concentrate exclusively on binary trees. To do this, we need some more notation and terminology

DEFINITION 6.3. *For a vertex v in a binary tree T, we let T^v denote the vertices in the complete binary subtree of T rooted at v, and $T_*^v = T^v \setminus \{v\}$ the vertices in T^v without its root v. We let P^v denote the vertices in the unique path from v to the root z of T and $P_*^v = P^v \setminus \{v\}$ the path without v.*

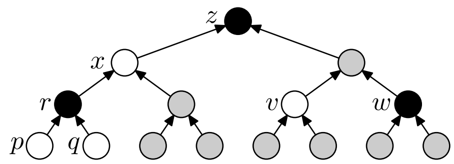
Definition 6.3 is illustrated in Figure 7. We blur the distinction somewhat between a tree T and the vertices in $V(T)$ and write, for instance, $T \setminus (T^v \cup P^v)$ instead of $V(T) \setminus (T^v \cup P^v)$ to denote all vertices in the tree unrelated to v.

In the standard black-white pebble game, if at some time t there is an independent black pebble on v, a pebbling need not place any pebbles on T^v after time t. As an analogy, if $\mathbb{C}_t \vDash \bigvee_{i=1}^{d} v_i$, it is not difficult to see that no axioms from $Ax^d(T^v)$ need be used in the resolution derivation after time t to derive $\bigvee_{i=1}^{d} z_i$. Therefore, it seems natural to think of a black pebble on v as derived truth $\bigvee_{i=1}^{d} v_i$ of v, and we want $\mathbb{C}_t$ to induce a subconfiguration $v\langle\emptyset\rangle$ if $\mathbb{C}_t \vDash \bigvee_{i=1}^{d} v_i$.

What kind of clause configuration should correspond to a dependent black pebble on v supported by white pebbles on W, i.e., a subconfiguration $v\langle W\rangle$? Well, one way of looking at $v\langle W\rangle$ is that this is the subconfiguration such that we would obtain an independent black pebble on v from it if the white pebbles on W were removed. But getting white pebbles *off* vertices is exactly as hard as getting black pebbles *on* vertices (compare with Proposition 4.2 on page 70). In view of this, we can describe $v\langle W\rangle$ as the subconfiguration from which we can immediately derive $v\langle\emptyset\rangle$ by assuming black pebbles on W. And as to black pebbles, we just argued that they should correspond to clauses $\bigvee_{i=1}^{d} v_i$. Our conclusion is that $\mathbb{C}_t$ should induce $v\langle W\rangle$ if this clause configuration *together with assumed independent black pebbles on all $w \in W$* implies an independent black pebble on v, i.e., if $\mathbb{C}_t \cup \left\{\bigvee_{i=1}^{d} w_i \mid w \in W\right\} \vDash \bigvee_{i=1}^{d} v_i$. Continuing our example from Figure 4, in Figure 8 we present a clause configuration

$$\left[\begin{array}{l|l} \overline{x}_i \vee \overline{v}_j \vee \bigvee_{l=1}^{d} z_l & \\ \overline{p}_i \vee \overline{q}_j \vee \bigvee_{l=1}^{d} r_l & i,j \in [d] \\ \bigvee_{l=1}^{d} w_l & \end{array}\right]$$

(a) *Clause configuration* $\mathbb{C}$.



(b) $\mathbb{L}(\mathbb{C}) = \{z\langle x, v\rangle, r\langle p, q\rangle, w\langle\emptyset\rangle\}$.

FIG. 8. *An example clause configuration* $\mathbb{C}$ *and its induced L-configuration* $\mathbb{L}(\mathbb{C})$.

corresponding to the given set of pebbles according to this intuitive understanding of induced pebble configurations.

Our formal definitions follow this intuition fairly closely, but since resolution derivations have no reason to be as well behaved as to fit the description above, we need to add a number of technical details.

For white pebbles, it will simplify matters if we can ensure that they have the following property.

DEFINITION 6.4. *For a vertex v and a vertex set W strictly below v, if for every $w \in W$ there is a path $P : w \rightsquigarrow v$ not intersecting $W \setminus \{w\}$, we say that W is a* simple set below v *and that $v\langle W\rangle$ is a* simple subconfiguration. *$\mathbb{L}$ is a* simple L-configuration *if all subconfigurations $v\langle W\rangle \in \mathbb{L}$ are simple.*

In the following, $\mathbb{B}(V)$ can be thought of as "truth of all vertices in V" and A_V as "truth of some vertex in V." We will be particularly interested in clauses A_{P^v}, i.e., clauses stating that some variable on the path from v to the root z is true.

DEFINITION 6.5. *Let* $\mathbb{B}(V) = \left\{\bigvee_{i=1}^{d} v_i \mid v \in V\right\}$ *and* $A_V = \bigvee_{v \in V} \bigvee_{i=1}^{d} v_i$.

Given a set of clauses $\mathbb{C}$ and a vertex v, if a vertex set $V \subseteq T \setminus P^v$ is such that $\mathbb{C} \cup \mathbb{B}(V) \vDash A_{P^v}$, we say that V is a support *for v with respect to $\mathbb{C}$. If there is no $V' \subsetneqq V$ such that $\mathbb{C} \cup \mathbb{B}(V') \vDash A_{P^v}$, the support is* minimal. *If V is a support for v with respect to $\mathbb{C}$ such that $\mathbb{C} \cup \mathbb{B}(V) \nvDash A_{P^v_*} = A_{P^v} \setminus \bigvee_{i=1}^{d} v_i$, we say that v is* maximal *with respect to $\mathbb{C}$ and V.*

We define the supporting white pebbles *in the set V of the vertex v as $swp(v, V) = \left\{w \in V \cap T^v_* \mid P^w_* \cap V = \emptyset\right\}$.*

When it is clear from context, we sometimes omit which support or vertex is minimal or maximal with respect to what. Note that $swp(v, V)$ is a simple set below v in the sense of Definition 6.4.

DEFINITION 6.6 (induced L-configuration). *For a set of clauses $\mathbb{C}$ derived from* $^*Peb^d_T$, *the* induced L-configuration $\mathbb{L}(\mathbb{C})$ *consists of all subconfigurations $v\langle V\rangle$ such that*

1. *there is a minimal support $V' \subseteq T \setminus P^v$ for $v$with respect to $\mathbb{C}$,*
2. *v is maximal with respect to $\mathbb{C}$ and V', and*
3. *$V = swp(v, V')$.*

That is, it holds that $\mathbb{C} \cup \mathbb{B}(V') \vDash A_{P^v}$ but $\mathbb{C} \cup \mathbb{B}(V') \nvDash A_{P^v_*}$, the set V' is minimal with this property, and if V' is not simple below v, we remove vertices in a bottom-up fashion until we get such a set $V \subseteq V'$. The reader can verify that this definition matches the example in Figure 8.

Remark 6.7. Note that a black pebble on v is defined in terms of $A_{P^v} = \bigvee_{u \in P^v} \bigvee_{i=1}^{d} u_i$ instead of just $\bigvee_{i=1}^{d} v_i$. Otherwise, we will not be able to prove the correspondence between L-pebblings and resolution derivation that we need. This means that if we let, say,

$$\mathbb{C}' = \left[\begin{array}{l|l} \overline{x}_i \vee \overline{v}_j \vee \bigvee_{n=1}^{d} z_n & \\ \overline{p}_i \vee \overline{q}_j \vee \bigvee_{n=1}^{d} r_n \vee \bigvee_{n=1}^{d} x_n & i,j \in [d] \\ \bigvee_{n=1}^{d} w_n \vee \bigvee_{n=1}^{d} z_n & \end{array} \right] \tag{6.1}$$

in Figure 8, then $\mathbb{C}'$ induces the same pebble subconfigurations as does $\mathbb{C}$, so $\mathbb{L}(\mathbb{C}') = \mathbb{L}(\mathbb{C}) = \{z\langle x, v\rangle, r\langle p, q\rangle, w\langle\emptyset\rangle\}$.

The reason we use $V = \mathit{swp}(v, V')$ instead of $V' \cap T^v_*$ (or even $V' \setminus P^v$) to define the white pebbles is that for technical purposes, we would like to have simple sets V below v in our induced subconfigurations $v\langle V\rangle$, but the minimal supporting sets V' do not necessarily have this property. For instance, in the clause configuration

$$\mathbb{C}'' = \left[\begin{array}{l|l} \overline{r}_i \vee \overline{x}_j \vee \overline{v}_l \vee \bigvee_{n=1}^{d} z_n & \\ \overline{p}_i \vee \overline{q}_j \vee \bigvee_{n=1}^{d} r_n \vee \bigvee_{n=1}^{d} x_n & i,j,l \in [d] \\ \overline{v}_l \vee \bigvee_{n=1}^{d} w_n \vee \bigvee_{n=1}^{d} z_n & \end{array} \right] \tag{6.2}$$

the vertices z and w have minimal supports $\{r, x, v\}$ and $\{v\}$, respectively, which are not simple sets below z and w, but since Definition 6.6 ignores all but the topmost vertices below the supported vertex, we get $\mathbb{L}(\mathbb{C}'') = \mathbb{L}(\mathbb{C}) = \{z\langle x, v\rangle, r\langle p, q\rangle, w\langle\emptyset\rangle\}$.

Thanks to this we get cleaner pebblings to work with (this will be used in the appendix), and it seems very plausible anyway that optimal resolution derivations should never result in clause configurations like $\mathbb{C}''$. Indeed, since the bound we will prove is asymptotically tight, we see that we do not really lose anything by restricting the white pebbles to $V = \mathit{swp}(v, V')$ instead of $V' \cap T^v_*$ or $V' \setminus P^v$.

Recall that the goal of this section is to demonstrate that resolution derivations induce L-pebblings. Suppose that $\pi = \{\mathbb{C}_0, \ldots, \mathbb{C}_\tau\}$ is a resolution derivation of $\bigvee_{i=1}^{d} z_i$ from $^*Peb^d_T$. For $\mathbb{C}_0 = \emptyset$ we have $\mathbb{L}(\mathbb{C}_0) = \emptyset$, and $\mathbb{C}_\tau = \{\bigvee_{i=1}^{d} z_i\}$ induces a single independent black pebble $\mathbb{L}(\mathbb{C}_\tau) = \{z\langle\emptyset\rangle\}$ on the root of T. Hence, we are done if we can show that $\{\mathbb{L}(\mathbb{C}_0), \ldots \mathbb{L}(\mathbb{C}_\tau)\}$ is a legal L-pebbling.

The rest of this section is devoted to proving that this is (almost) the case. We start by stating three technical lemmas. The first lemma relates subset containment of supporting sets and the order relation between corresponding subconfigurations.

LEMMA 6.8. *For a vertex $v \in V(T)$, if $u \in P^v$ is a vertex and $U', V' \subseteq T \setminus P^v$ are vertex sets such that $U' \cap T^v_* \subseteq V' \cap T^v_*$, then $u\langle \mathit{swp}(u, U')\rangle \succeq v\langle \mathit{swp}(v, V')\rangle$.*

Proof. Let $U = \mathit{swp}(u, U')$ and $V = \mathit{swp}(v, V')$. According to Definition 5.1, we need to show that $\mathit{cover}(v\langle V\rangle) \subseteq \mathit{cover}(u\langle U\rangle)$.

Suppose $w \in \mathit{cover}(v\langle V\rangle)$. This means that there is a path $P_1 : w \rightsquigarrow v$ from w to v such that $P_1 \cap V = \emptyset$. Also, since $u \in P^v$ there is a path $P_2 : v \rightsquigarrow u$. Concatenating these paths, we get a path $P = P_1 \cup P_2$ from w to u. We claim that $P \cap U = \emptyset$. If this is true, we have $w \in \mathit{cover}(u\langle U\rangle)$ and thus $\mathit{cover}(v\langle V\rangle) \subseteq \mathit{cover}(u\langle U\rangle)$, and the lemma follows.

To prove the claim, note first that since $U \subseteq U' \subseteq T \setminus P^v$, it holds that $P_2 \cap U = \emptyset$. Suppose P_1 intersects U, and let $x \in P_1 \cap U$. By assumption, $x \notin V$ since $P_1 \cap V = \emptyset$. But $x \in U \subseteq U' \cap T^v_* \subseteq V' \cap T^v_*$, so Definition 6.5 tells us that the reason $x \notin V$ must be that $P^x_* \cap V' \cap T^v \neq \emptyset$. Let $y \in P^x_* \cap V' \cap T^v$ be the vertex closest to v. Looking at Definition 6.5 again, since $P^y_* \cap V' = \emptyset$ by construction, we have $y \in V$. But if so, $P_1 \cap V \neq \emptyset$, which is a contradiction. ∎

A second handy lemma is that if V' is not minimal or v is not maximal with respect to $\mathbb{C}$, this just means that $\mathbb{C}$ induces something stronger than the subconfiguration $v\langle \mathit{swp}(v, V')\rangle$.

LEMMA 6.9. *If* $\mathbb{C} \cup \mathbb{B}(V') \vDash A_{P^v}$ *for* $V' \subseteq T \setminus P^v$*, then there is an induced subconfiguration* $u\langle U\rangle \in \mathbb{L}(\mathbb{C})$ *such that* $v\langle swp(v, V')\rangle \preceq u\langle U\rangle$*.*

Proof. Minimize $U' \subseteq V'$ and then pick $u \in P^v$ as close to the root as possible so that $\mathbb{C} \cup \mathbb{B}(U') \vDash A_{P^u}$. Set $U = swp(u, U')$ and use Lemma 6.8. ☐

The following easy lemma will be used repeatedly.

LEMMA 6.10. *Suppose that* C, D *are clauses and* $\mathbb{C}$ *is a set of clauses. Then* $\mathbb{C} \cup \{C\} \vDash D$ *if and only if* $\mathbb{C} \vDash \overline{a} \vee D$ *for all* $a \in Lit(C)$*.*

Proof. Assume that $\mathbb{C} \cup \{C\} \vDash D$ and consider an assignment α such that $\alpha(\mathbb{C}) = 1$ and $\alpha(D) = 0$ (if there is no such α, then $\mathbb{C} \vDash D \subseteq \overline{a} \vee D$). Such an α must set all $\overline{a}$ to true. Conversely, if $\mathbb{C} \vDash \overline{a} \vee D$ for all $a \in Lit(C)$ and α is such that $\alpha(\mathbb{C}) = \alpha(C) = 1$, it must hold that $\alpha(D) = 1$, since otherwise $\alpha(\overline{a} \vee D) = 0$ for some literal $a \in Lit(C)$ satisfied by α. ☐

Using these lemmas, we can prove that resolution derivations induce L-pebblings. By the L-pebbling rules in Definition 5.2, any subconfiguration $v\langle V\rangle$ may be erased freely at any time. Consequently, we need not worry about subconfigurations disappearing during the transition from $\mathbb{C}_t$ to $\mathbb{C}_{t+1}$. What we do need to check, though, is that no $v\langle V\rangle$ appears inexplicably in $\mathbb{L}(\mathbb{C}_{t+1})$ as a result of a derivation step $\mathbb{C}_t \rightsquigarrow \mathbb{C}_{t+1}$, but that we can always derive any subconfiguration $v\langle V\rangle \in \mathbb{L}(\mathbb{C}_{t+1}) \setminus \mathbb{L}(\mathbb{C}_t)$ from $\mathbb{L}(\mathbb{C}_t)$ by the L-pebbling rules.

Let us consider the resolution derivation rules one by one.

OBSERVATION 6.11 (inference). *If* $\mathbb{C}_{t+1}$ *is derived from* $\mathbb{C}_t$ *by inference, then* $\mathbb{L}(\mathbb{C}_{t+1}) = \mathbb{L}(\mathbb{C}_t)$*.*

Proof. This is immediate, since $\mathbb{C}_t$ and $\mathbb{C}_{t+1}$ imply exactly the same clauses. ☐

We remark that, as was stated in section 2.1, this means that the exact definition of the resolution derivation rule is not important. The lower bound on space will hold for any sound derivation rule as long as the lines in the proof are disjunctive clauses.

LEMMA 6.12 (erasure). *Suppose that* $\mathbb{C}_{t+1}$ *is derived from* $\mathbb{C}_t$ *by erasure. Then for each* $v\langle V\rangle \in \mathbb{L}(\mathbb{C}_{t+1})$ *there is a* $u\langle U\rangle \in \mathbb{L}(\mathbb{C}_t)$ *such that* $v\langle V\rangle \preceq u\langle U\rangle$*.*

Proof. By assumption there is a $V' \subseteq T \setminus P^v$ such that $V = swp(v, V')$ and $\mathbb{C}_{t+1} \cup \mathbb{B}(V') \vDash A_{P^v}$. Certainly, the same implication holds for $\mathbb{C}_t \supseteq \mathbb{C}_{t+1}$. The lemma follows from Lemma 6.9. ☐

In particular, all new subconfigurations resulting from an erasure $\mathbb{C}_t \rightsquigarrow \mathbb{C}_{t+1}$ can be obtained from $\mathbb{L}(\mathbb{C}_t)$ by reversal moves.

LEMMA 6.13 (axiom download). *If* $\mathbb{C}_{t+1} = \mathbb{C}_t \cup \{C\}$ *for an axiom clause* $C \in Ax^d(r)$*, then all subconfigurations* $v\langle V\rangle \in \mathbb{L}(\mathbb{C}_{t+1}) \setminus \mathbb{L}(\mathbb{C}_t)$ *can be obtained from* $\mathbb{L}(\mathbb{C}_t) \cup r\langle pred(r)\rangle$ *by reversals from subconfigurations in* $\mathbb{L}(\mathbb{C}_t)$ *followed by mergers on the vertices* $\{r\} \cup pred(r)$*.*

Proof. Let us fix a vertex $v \in V(T)$ and an axiom $C \in Ax^d(r)$. If $v\langle V\rangle$ is a pebble subconfiguration induced at time $t+1$, by assumption there is a minimal $V' \subseteq T \setminus P^v$ with $V = swp(v, V') \subseteq V'$ such that $\mathbb{C}_t \cup \{C\} \cup \mathbb{B}(V') \vDash A_{P^v}$.

Our intuition is that downloading $C \in Ax^d(r)$ should not yield any interesting new subconfigurations $v\langle V\rangle$ if $r \in T \setminus T^v$, and for $r \in T^v$ we should be able to explain new subconfigurations with the help of an introduction of $r\langle pred(r)\rangle$ in our L-pebbling. We prove this by a case analysis over r.

$r \in T \setminus (T^v \cup P^v)$: Observing that $\mathbb{B}(r) \vDash C$ (this will be used repeatedly), we get that $\mathbb{C}_t \cup \mathbb{B}(V' \cup \{r\}) \vDash A_{P^v}$ for $V' \cup \{r\} \subseteq T \setminus P^v$. Lemma 6.9 tells us that there is a $u\langle U\rangle \in \mathbb{L}(\mathbb{C}_t)$ such that $u\langle U\rangle \succeq v\langle swp(v, V' \cup \{r\})\rangle = v\langle swp(v, V')\rangle = v\langle V\rangle$, where the first equality follows since $r \notin T^v_*$. Hence, we can get $v\langle V\rangle$ from $\mathbb{L}(\mathbb{C}_t)$ by a reversal move.

$r \in P^v_*$: Write $C = \overline{p}_i \vee \overline{q}_j \vee \bigvee_{l=1}^{d} r_l$ for $\{p, q\} = \mathit{pred}(r) \neq \emptyset$ and assume without loss of generality that p is the vertex in $P^v \cap \mathit{pred}(r)$. Using Lemma 6.10 to move p_i to the right of the implication sign yields $\mathbb{C}_t \cup \mathbb{B}(V') \vDash A_{P^v} \vee p_i = A_{P^v}$, and since V' is minimal it follows that $v\langle V\rangle \in \mathbb{L}(\mathbb{C}_t)$.

$r = v$: Note first that the introduction of $r\langle \mathit{pred}(r)\rangle$ is a legal pebbling move, so if $\mathbb{C}_t \cup \{C\} \cup \mathbb{B}(V') \vDash A_{P^r}$ for $\mathit{pred}(r) \subseteq V'$, no further analysis is needed for $r\langle \mathit{swp}(r, V')\rangle = r\langle \mathit{pred}(r)\rangle$. In particular, this is always the case if $\mathit{pred}(r) = \emptyset$, i.e., if r is a source.

Suppose that $v\langle V\rangle = r\langle \mathit{swp}(r, V')\rangle \in \mathbb{L}(\mathbb{C}_{t+1})$ for $V \neq \mathit{pred}(r) = \{p, q\}$, and write $C = \overline{p}_i \vee \overline{q}_j \vee \bigvee_{l=1}^{d} r_l$. We want to derive $r\langle V\rangle$ by the pebbling rules from $\mathbb{L}(\mathbb{C}_t) \cup \bigl\{r\langle \mathit{pred}(r)\rangle\bigr\}$. By symmetry, we get two subcases.

1. $p \in V$, $q \notin V$: By Definition 6.5, we have $p \in V' \supseteq V$. Also, it must hold that $q \notin V'$, since otherwise $P^q_* \cap V' \subseteq P^q_* \cap (T \setminus P^r) = P^q_* \cap (T \setminus P^q_*) = \emptyset$ would imply that $q \in V = \mathit{swp}(v, V')$, contrary to assumption. It follows that $V' \subseteq T \setminus P^q$. Also, we can use Lemma 6.10 to move q_j to the right-hand side of the implication sign and get $\mathbb{C}_t \cup \mathbb{B}(V') \vDash A_{P^r} \vee q_j \subseteq A_{P^r} \vee \bigvee_{l=1}^{d} q_l = A_{P^q}$. Plugging this into Lemma 6.9 shows that there is a $w\langle W\rangle \in \mathbb{L}(\mathbb{C}_t)$ such that $q\bigl\langle V \setminus \{p\}\bigr\rangle = q\bigl\langle \mathit{swp}(q, V')\bigr\rangle \preceq w\langle W\rangle$. Thus we can derive $q\bigl\langle V \setminus \{p\}\bigr\rangle$ from $\mathbb{L}(\mathbb{C}_t)$ by reversal and then merge $r\langle \mathit{pred}(r)\rangle = r\langle p, q\rangle$ with $q\bigl\langle V \setminus \{p\}\bigr\rangle$ to obtain $r\Bigl\langle \bigl(\{p, q\} \cup (V \setminus \{p\})\bigr) \setminus \{q\}\Bigr\rangle = r\langle V\rangle$.
2. $p, q \notin V$: Again, by Definition 6.5 we have $p, q \notin V'$. If we use Lemma 6.10 twice, we get $\mathbb{C}_t \cup \mathbb{B}(V') \vDash A_{P^p} \wedge A_{P^q}$, and noting that $V' \subseteq T \setminus \bigl(P^p \cup P^q\bigr)$ we can apply Lemma 6.9 to derive $p\bigl\langle V \cap T^p_*\bigr\rangle$ and $q\bigl\langle V \cap T^q_*\bigr\rangle$ from $\mathbb{L}(\mathbb{C}_t)$ by reversal moves. Merging these pebble subconfigurations with $r\langle p, q\rangle$, we get the desired pebble subconfiguration $r\Bigl\langle \bigl(V \cap T^p_*\bigr) \cup \bigl(V \cap T^q_*\bigr)\Bigr\rangle = r\langle V\rangle$.

We note in passing that this is the place in the proof where we critically need black pebbles to be defined in terms of $A_{P^v} = \bigvee_{u \in P^v} \bigvee_{i=1}^{d} u_i$ instead of just $\bigvee_{i=1}^{d} v_i$. (Although it also simplifies the proof of the case $r \in P^v_*$, there it is not strictly necessary.)

$r \in T^v_*$: By assumption, $\mathbb{C}_t \cup \{C\} \cup \mathbb{B}(V') \vDash A_{P^v}$, and since $r \in T^v_*$ and $\mathbb{B}(r) \vDash C$ we have $\mathbb{C}_t \cup \mathbb{B}(V' \cup \{r\}) \vDash A_{P^v}$ for $V' \cup \{r\} \subseteq T \setminus P^v$. If $P^r \cap V' \neq \emptyset$, it holds that $\mathit{swp}(v, V' \cup \{r\}) = \mathit{swp}(v, V')$, and we can obtain $v\langle V\rangle$ from $\mathbb{L}(\mathbb{C}_t)$ by reversal according to Lemma 6.9. Suppose therefore that $P^r \cap V' = \emptyset$. Also, we assume that $\mathbb{C}_t \cup \mathbb{B}(V') \nvDash A_{P^v}$ since otherwise $v\langle V\rangle \in \mathbb{L}(\mathbb{C}_t)$ and there is nothing to prove.

Pick $U' \subseteq V' \cup \{r\}$ minimal and then $u \in P^v$ maximal with respect to U' such that $\mathbb{C}_t \cup \mathbb{B}(U') \vDash A_{P^u}$. Since $\mathbb{C}_t \cup \mathbb{B}(V') \nvDash A_{P^v}$ we must have $r \in U'$. Set $U = \mathit{swp}(u, U')$. Using that $P^r_* \cap U' \subseteq P^r_* \cap V' = \emptyset$, we see that $r \in U$. Consequently, we cannot use $u\langle U\rangle \in \mathbb{L}(\mathbb{C}_t)$ to derive $v\langle V\rangle \not\preceq u\langle U\rangle$ by reversal. However, since $U' \subseteq V' \cup \{r\}$, Lemma 6.8 says that $v\bigl\langle (V \cup \{r\}) \setminus T^r_*\bigr\rangle = v\bigl\langle \mathit{swp}(v, V' \cup \{r\})\bigr\rangle \preceq u\langle U\rangle$ can be derived by reversal from $\mathbb{L}(\mathbb{C}_t)$. If we could also derive $r\bigl\langle V \cap T^r_*\bigr\rangle$ from $\mathbb{L}(\mathbb{C}_t) \cup \bigl\{r\langle \mathit{pred}(r)\rangle\bigr\}$, we could do a merger to get $v\Bigl\langle \bigl(((V \cup \{r\}) \setminus T^r_*) \cup (V \cap T^r_*)\bigr) \setminus \{r\}\Bigr\rangle = v\langle V\rangle$.

Hence, we are done if we can derive the pebble subconfiguration $r\bigl\langle V \cap T^r_*\bigr\rangle = r\bigl\langle \mathit{swp}(v, V') \cap T^r_*\bigr\rangle = r\bigl\langle \mathit{swp}(r, V')\bigr\rangle$ from $\mathbb{L}(\mathbb{C}_t) \cup \bigl\{r\langle \mathit{pred}(r)\rangle\bigr\}$. But $A_{P^r} \supseteq A_{P^v}$, so by assumption we have $\mathbb{C}_t \cup \{C\} \cup \mathbb{B}(V') \vDash A_{P^r}$ for $V' \subseteq T \setminus P^r$. This is almost exactly the case $r = v$ above, where we proved that $r\bigl\langle \mathit{swp}(r, V')\bigr\rangle$

is derivable from $\mathbb{L}(\mathbb{C}_t) \cup \{r\langle \mathit{pred}(r)\rangle\}$. The only difference is that now it is not necessarily true that V' is a minimal support and that r is maximal with respect to V'. But these assumptions were not used in the derivation of $r\langle \mathit{swp}(r, V')\rangle$ from $\mathbb{L}(\mathbb{C}_t) \cup \{r\langle \mathit{pred}(r)\rangle\}$ anyway, so we can reuse exactly the same proof to get $r\langle \mathit{swp}(r, V')\rangle$. This concludes the analysis for $r \in T^v_*$.

Studying the pebbling moves performed in the case analysis above, we see that all subconfigurations $v\langle V\rangle \in \mathbb{L}(\mathbb{C}_{t+1}) \setminus \mathbb{L}(\mathbb{C}_t)$ resulting from an axiom download can be obtained from $\mathbb{L}(\mathbb{C}_t) \cup r\langle \mathit{pred}(r)\rangle$ by a (possibly empty) sequence of reversals from $\mathbb{L}(\mathbb{C}_t)$, followed by a (possibly empty) sequence of mergers on $\{r\} \cup \mathit{pred}(r)$. ∎

Combining the results proven for axiom download, inference, and erasure, we can show that a resolution derivation induces a legal L-pebbling. We need a pair of easy technical observations about L-pebbling cost, which we state as a separate proposition for clarity.

Proposition 6.14. *For $\mathbb{L}_1$ and $\mathbb{L}_2$ arbitrary L-configurations, it holds that*

1. *if $\mathbb{L}_1 \subseteq \mathbb{L}_2$ then $\mathsf{cost}(\mathbb{L}_1) \leq \mathsf{cost}(\mathbb{L}_2)$, and*
2. *$\mathsf{cost}(\mathbb{L}_1 \cup \mathbb{L}_2) \leq \mathsf{cost}(\mathbb{L}_1) + \mathsf{cost}(\mathbb{L}_2)$.*

Proof. This is fairly obvious, but we give a short formal proof for completeness. According to Definition 5.2, if $\mathit{Bl}(\mathbb{L}_1) \cup \mathit{Wh}(\mathbb{L}_1) \subseteq \mathit{Bl}(\mathbb{L}_2) \cup \mathit{Wh}(\mathbb{L}_2)$, then $\mathsf{cost}(\mathbb{L}_1) = |\mathit{Bl}(\mathbb{L}_1) \cup \mathit{Wh}(\mathbb{L}_1)| \leq |\mathit{Bl}(\mathbb{L}_2) \cup \mathit{Wh}(\mathbb{L}_2)| = \mathsf{cost}(\mathbb{L}_2)$. Part 1 follows immediately from this observation. Part 2 also follows easily, since each pebbled vertex on the left-hand side is counted at least once on the right-hand side. ∎

Theorem 6.15. *Let $\pi = \{\mathbb{C}_0, \ldots, \mathbb{C}_\tau\}$ be a resolution derivation of $\bigvee_{l=1}^{d} z_l$ from $^*\mathit{Peb}^d_T$. Then the L-configurations $\mathbb{L}(\mathbb{C}_0), \ldots, \mathbb{L}(\mathbb{C}_\tau)$ are contained in a legal, complete L-pebbling $\mathcal{L}$ of T such that $\max_{t\in[\tau]}\{\mathsf{cost}(\mathbb{L}(\mathbb{C}_t))\} = \Omega(\mathsf{cost}(\mathcal{L}))$.*

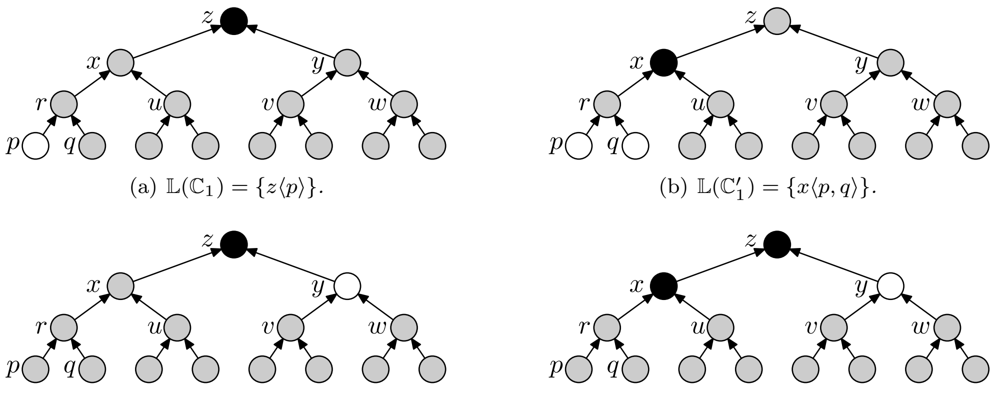
Proof. The fact that $\{\mathbb{L}(\mathbb{C}_0), \ldots, \mathbb{L}(\mathbb{C}_\tau)\}$ is the "backbone" of a legal L-pebbling was proven in Observation 6.11, Lemma 6.12, and Lemma 6.13, where it was explicitly indicated how the "holes" in $\mathbb{L}(\mathbb{C}_t) \rightsquigarrow \mathbb{L}(\mathbb{C}_{t+1})$ could be filled in by L-pebbling moves to get a legal pebbling $\mathcal{L}$. It was also noted above that $\mathbb{L}(\mathbb{C}_0) = \emptyset$ and $\mathbb{L}(\mathbb{C}_\tau) = \{z\langle\emptyset\rangle\}$, so filling in the holes results in a complete pebbling of T.

The bound $\max_{t\in[\tau]}\{\mathsf{cost}(\mathbb{L}(\mathbb{C}_t))\} = \Omega(\mathsf{cost}(\mathcal{L}))$ does not follow immediately from this, however. The problem is that a single resolution derivation step $\mathbb{C}_t \rightsquigarrow \mathbb{C}_{t+1}$ may induce several L-pebbling moves to get from $\mathbb{L}(\mathbb{C}_t)$ to $\mathbb{L}(\mathbb{C}_{t+1})$ in $\mathcal{L}$. Therefore, we have to consider the possibility[6] that the maximal pebbling cost in $\mathcal{L}$ is reached in some intermediate L-configuration $\mathbb{L}'$ between $\mathbb{L}(\mathbb{C}_t)$ and $\mathbb{L}(\mathbb{C}_{t+1})$.

Since inference steps in π do not change the set of induced L-configurations, we get two cases.

1. $\mathbb{C}_t \rightsquigarrow \mathbb{C}_{t+1}$ is an erasure. The moves to get from $\mathbb{L}(\mathbb{C}_t)$ to $\mathbb{L}(\mathbb{C}_{t+1})$ are a series of reversals from $\mathbb{L}(\mathbb{C}_t)$ followed by a series of erasures from $\mathbb{L}(\mathbb{C}_t)$. In view of part 1 of Proposition 6.14, the maximal cost is incurred in the intermediate L-configuration $\mathbb{L}'$ after all reversals but before all erasures. We have $\mathbb{L}' = \mathbb{L}(\mathbb{C}_t) \cup \mathbb{L}(\mathbb{C}_{t+1})$, and by part 2 of Proposition 6.14 it follows that $\mathsf{cost}(\mathbb{L}') \leq \mathsf{cost}(\mathbb{L}(\mathbb{C}_t)) + \mathsf{cost}(\mathbb{L}(\mathbb{C}_{t+1})) \leq 2 \cdot \max_{i\in[t,t+1]}\{\mathsf{cost}(\mathbb{L}(\mathbb{C}_i))\}$.
2. $\mathbb{C}_t \rightsquigarrow \mathbb{C}_{t+1}$ is a download of $C \in \mathit{Ax}^d(v)$. In this case the sequence of moves to get from $\mathbb{L}(\mathbb{C}_t)$ to $\mathbb{L}(\mathbb{C}_{t+1})$ is a possible introduction of $v\langle \mathit{pred}(v)\rangle$ followed by a series of reversals from $\mathbb{L}(\mathbb{C}_t)$, then a series of mergers on $\{v\} \cup \mathit{pred}(v)$, and finally a series of erasures of subconfigurations not derived in the merger

[6]In fact, this does not happen, but instead of proving this we happily sacrifice a constant 2 here in order to get a simpler (or at least slightly less involved) proof.



(a) $\mathbb{L}(\mathbb{C}_1) = \{z\langle p\rangle\}$.

(b) $\mathbb{L}(\mathbb{C}'_1) = \{x\langle p, q\rangle\}$.

(c) $\mathbb{L}(\mathbb{C}_2) = \{z\langle y\rangle\}$.

(d) $\mathbb{L}(\mathbb{C}'_2) = \{z\langle y\rangle, x\langle\emptyset\rangle\}$.

FIG. 9. *Illustration of reversal moves in Remark* 6.16.

moves. Again by part 1 of Proposition 6.14, we may concentrate on the L-configuration $\mathbb{L}'$ after all reversals and mergers but before the erasures.

All pebbles in $Bl(\mathbb{L}') \cup Wh(\mathbb{L}')$ are present in either $\mathbb{L}(\mathbb{C}_t)$ or $\mathbb{L}(\mathbb{C}_{t+1})$, except possibly for the pebbles on $\{v\} \cup pred(v)$ which may have been introduced and then merged away. Since by construction all subconfigurations resulting from these mergers must be contained in $\mathbb{L}(\mathbb{C}_{t+1})$, the pebbles on $\{v\} \cup pred(v)$ are the only ones that can appear and then disappear during the intermediate pebbling steps. If we remove $\{v\} \cup pred(v)$ from $Bl(\mathbb{L}') \cup Wh(\mathbb{L}')$, the pebbling cost cannot decrease by more than 3.

Since all pebbles $Bl(\mathbb{L}') \setminus \big(\{v\} \cup pred(v)\big)$ and $Wh(\mathbb{L}') \setminus \big(\{v\} \cup pred(v)\big)$ are contained in $Bl(\mathbb{L}(\mathbb{C}_t)) \cup Bl(\mathbb{L}(\mathbb{C}_{t+1}))$ and $Wh(\mathbb{L}(\mathbb{C}_t)) \cup Wh(\mathbb{L}(\mathbb{C}_{t+1}))$, respectively, appealing to part 2 of Proposition 6.14 again we get the inequality $\max_{i\in[t,t+1]}\big\{\mathit{cost}(\mathbb{L}(\mathbb{C}_i))\big\} \geq \frac{1}{2}\big(\mathit{cost}(\mathbb{L}') - 3\big)$.

This establishes that even if the maximal cost in the L-pebbling $\mathcal{L}$ induced by derivation $\pi = \{\mathbb{C}_0, \ldots, \mathbb{C}_\tau\}$ is attained in some intermediate L-configuration $\mathbb{L}' \notin \big\{\mathbb{L}(\mathbb{C}_t) \mid t \in [\tau]\big\}$, it still holds that $\max_{t\in[\tau]}\big\{\mathit{cost}(\mathbb{L}(\mathbb{C}_t))\big\} \geq \frac{1}{2}\mathit{cost}(\mathcal{L}) + \mathrm{O}(1)$. The theorem follows. $\square$

Remark 6.16. At this point, the reader might ask whether we really need the reversal rule in the L-pebble game in order to get Theorem 6.15 or whether it is just a convenience to simplify the proofs. The answer is that unfortunately, the reversal rule is really needed. We provide two examples of this below, using the binary tree of height 3 with vertex labels as in Figure 9.

Suppose that we have

$$\mathbb{C}_1 = \left[\begin{array}{l|l} \overline{p}_i \vee \overline{q}_j \vee \bigvee_{l=1}^{d} x_l & \\ \overline{p}_i \vee \bigvee_{l=1}^{d} z_l & i,j \in [d] \end{array}\right] \quad \text{with} \quad \mathbb{L}(\mathbb{C}_1) = \big\{z\langle p\rangle\big\} \tag{6.3}$$

(see Figure 9(a)). Note that only the subset of clauses on the second line in $\mathbb{C}_1$ contributes to $\mathbb{L}(\mathbb{C}_1)$. It is true that, because of the clauses on the first line, we have

$$\mathbb{C}_1 \cup \mathbb{B}(p,q) \vDash A_{P^x} = \bigvee_{l=1}^{d} x_l \vee \bigvee_{l=1}^{d} z_l, \tag{6.4}$$

but the support $\{p,q\}$ is not minimal and x is not maximal with respect to $\mathbb{C}_1$ and

$\{p, q\}$ (Definition 6.5) since it also holds that

$$\mathbb{C}_1 \cup \mathbb{B}(p) \vDash A_{P^x_*} = \bigvee_{l=1}^{d} z_l. \tag{6.5}$$

However, if we erase the second line of clauses from (6.3), the implication in (6.4) comes into play, and we get

$$\mathbb{C}'_1 = \left[\overline{p}_i \vee \overline{q}_j \vee \bigvee_{l=1}^{d} x_l \mid i \in [d]\right] \quad \text{with} \quad \mathbb{L}(\mathbb{C}'_1) = \{x\langle p, q\rangle\} \tag{6.6}$$

as in Figure 9(b). It is necessary to have the reversal rule to go from Figure 9(a) to Figure 9(b), which shows why reversals are needed in Lemma 6.12.

This might perhaps look like a somewhat silly example, but it nevertheless pinpoints the problem: although the erasures going from $\mathbb{C}_1$ in (6.3) to $\mathbb{C}'_1$ in (6.6) might seem clearly nonoptimal, we cannot exclude the possibility that such derivation steps are made, and so we have to be able to match such steps by pebbling moves.

As a second example, consider

$$\mathbb{C}_2 = \left[\begin{array}{l|c} x_1 \vee v_1 \vee \bigvee_{j=1}^{d} z_j & \\ x_1 \vee w_1 \vee \bigvee_{j=1}^{d} z_j & i \in [d] \\ \overline{y}_i \vee \bigvee_{j=1}^{d} z_j & \end{array}\right] \quad \text{with} \quad \mathbb{L}(\mathbb{C}_2) = \{z\langle y\rangle\} \tag{6.7}$$

(see Figure 9(c)). Here the first two clauses do not contribute to $\mathbb{L}(\mathbb{C}_2)$, but if we download the axiom $\overline{v}_1 \vee \overline{w}_1 \vee \bigvee_{j=1}^{d} y_j$, we get

$$\mathbb{C}'_2 = \left[\begin{array}{l|c} x_1 \vee v_1 \vee \bigvee_{j=1}^{d} z_j & \\ x_1 \vee w_1 \vee \bigvee_{j=1}^{d} z_j & \\ \overline{y}_i \vee \bigvee_{j=1}^{d} z_j & i \in [d] \\ \overline{v}_1 \vee \overline{w}_1 \vee \bigvee_{j=1}^{d} y_j & \end{array}\right] \quad \text{with} \quad \mathbb{L}(\mathbb{C}'_2) = \{z\langle y\rangle, x\langle\emptyset\rangle\} \tag{6.8}$$

as in Figure 9(d), since it is easy to check that $\mathbb{C}'_2 \vDash A_{P^x} = \bigvee_{j=1}^{d} x_j \vee \bigvee_{j=1}^{d} z_j$ but $\mathbb{C}'_2 \nvDash A_{P^x_*} = \bigvee_{j=1}^{d} z_j$. We cannot get $x\langle\emptyset\rangle$ from $\mathbb{L}(\mathbb{C}_2)$ unless we have reversal moves, so the reversal rule is needed also in Lemma 6.13.

We leave it to the reader to verify that $\mathbb{C}_1$ and $\mathbb{C}_2$ can indeed be derived from $*Peb^d_{T_3}$. We note, though, that it appears that in order to derive $\mathbb{C}_2$ one needs to pass stronger clause configurations along the way, and it seems very unclear why anyone would like to go from these clause configurations to the weaker configuration $\mathbb{C}_2$.

We conclude this section by proving Theorem 2.1 on page 65. Since we wanted to avoid unnecessary technicalities in section 2, Theorem 2.1 talks about refutations $\pi : Peb^d_{T_h} \vdash 0$ rather than derivations $\pi^* : *Peb^d_{T_h} \vdash \bigvee_{i=1}^{d} z_i$, but this is easily taken care of.

Theorem 2.1 (restated). *There is a translation function from clause configurations derived from $Peb^d_{T_h}$ into L-configurations in T_h such that any resolution refutation π of $Peb^d_{T_h}$ corresponds to a complete labeled pebbling $\mathcal{L}_\pi$ of T_h under this translation.*

Proof. Given a resolution refutation $\pi : Peb^d_{T_h} \vdash 0$, use (the proof of) Lemma 6.2 to transform the refutation π clause configuration by clause configuration into a derivation $\pi^* : {}^*Peb^d_{T_h} \vdash \bigvee_{i=1}^d z_i$ in the same space. Then use Definition 6.6 as the translation function, and let $\mathcal{L}_\pi$ be the labeled pebbling constructed from π^* in Theorem 6.15. $\square$

We comment that as another attempt to simplify the exposition in section 2, Theorem 2.1 leaves out the crucial information in Theorem 6.15 that the cost of $\mathcal{L}_\pi$ is upper-bounded by the maximal cost of the induced L-configurations $\mathbb{L}(\mathbb{C}_t)$. We will return to Theorem 6.15 and use this information in the proof of Theorem 2.2 at the end of the next section.

7. Induced L-pebble configurations measure clause set size. In the last section, we proved that $Sp\big(Peb^d_{T_h} \vdash 0\big) = Sp\big({}^*Peb^d_{T_h} \vdash \bigvee_{i=1}^d z_i\big)$ and that each resolution derivation $\pi : {}^*Peb^d_{T_h} \vdash \bigvee_{i=1}^d z_i$ induces a complete L-pebbling $\mathcal{L}$ of T_h such that $\max_{\mathbb{C}\in\pi}\big\{\mathit{cost}(\mathbb{L}(\mathbb{C}))\big\} = \Omega\big(\mathit{cost}(\mathcal{L})\big)$. In section 5 we stated (promising a proof in the appendix) that $\mathit{cost}(\mathcal{L}) = \Omega\big(\mathit{BW\text{-}Peb}(T)\big)$. The final component needed to piece together the proof of our lower bound on the refutation space of pebbling contradictions is that the number of pebbles in an induced L-configuration $\mathbb{L}(\mathbb{C})$ and the number of clauses in $\mathbb{C}$ are somehow connected.

Note that we cannot expect a proof of this fact to work regardless of the pebbling degree d. The induced L-pebbling in section 6 makes no assumptions about d, but we know that $Sp\big({}^*Peb^1_G \vdash z_1\big) = Sp\big(Peb^1_G \vdash 0\big) = \mathrm{O}(1)$. If we look at the resolution refutation π of Peb^1_G in constant space sketched at the end of section 4.2, we see that the induced L-pebbling starts by placing white pebbles on $pred(z)$ and a black pebble on z, i.e., introducing $z\langle pred(z)\rangle$, and then pushes the white pebbles downward by introducing $v\langle pred(v)\rangle$ for all v in reverse topological order and merging until it reaches $z\langle S\rangle$ for S the source vertices of G. Finally, the white pebbles $s \in S$ are eliminated one by one by introducing $s\langle\emptyset\rangle$ and merging. The reason that Peb^1_G can be refuted in constant space is that one single clause $z_1 \vee \bigvee_{v\in V} \overline{v}_1$ can induce an arbitrary number $|V|$ of white pebbles, or, phrasing it differently, that white pebbles are free for $d = 1$.

In Theorem 7.6 below we show that provided $d > 1$ one has to pay at least $|\mathbb{C}| \geq N$ clauses to get N induced pebbles. This completes the proof of our main theorem which was outlined in section 2.3. We first show some technical results about CNF formulas that will be needed in the proof.

Lemma 7.1. *Suppose that it holds for a set of clauses $\mathbb{C}$ and clauses D_1 and D_2 with $Vars(D_1) \cap Vars(D_2) = \emptyset$ that $\mathbb{C} \vDash D_1 \vee D_2$ but $\mathbb{C} \nvDash D_2$. Then there is a literal $a \in Lit(\mathbb{C}) \cap Lit(D_1)$.*

Proof. Pick a truth value assignment α such that $\alpha(\mathbb{C}) = 1$ but $\alpha(D_2) = 0$. By assumption $\alpha(D_1) = 1$. Let α' be the same assignment except that all satisfied literals in D_1 are flipped to false so that $\alpha'(D_1) = 0$ (which is possible since all literals are pairwise strictly distinct). Then $\alpha'(D_1 \vee D_2) = 0$ forces $\alpha'(\mathbb{C}) = 0$, so the flip must have falsified some previously satisfied clause in $\mathbb{C}$. $\square$

Definition 7.2. *A set of clauses $\mathbb{C}$ implies a clause D* minimally *if $\mathbb{C} \vDash D$ but for all $\mathbb{C}' \subsetneqq \mathbb{C}$ it holds that $\mathbb{C}' \nvDash D$. If $\mathbb{C} \vDash 0$ minimally, $\mathbb{C}$ is said to be* minimally unsatisfiable*.*

Lemma 7.3. *Let $\mathbb{C}$ be a set of clauses and D a clause such that $\mathbb{C} \vDash D$ minimally and $a \in Lit(\mathbb{C})$ but $\overline{a} \notin Lit(\mathbb{C})$. Then $a \in Lit(D)$.*

Proof. Suppose not. Let $\mathbb{C}_1 = \{C \in \mathbb{C} \mid a \in Lit(C)\}$ and $\mathbb{C}_2 = \mathbb{C} \setminus \mathbb{C}_1$. Since $\mathbb{C}_2 \nvDash D$ there is an α such that $\alpha(\mathbb{C}_2) = 1$ and $\alpha(D) = 0$. Note that $\alpha(a) = 0$, since otherwise $\alpha(\mathbb{C}_1) = 1$. It follows that $\overline{a} \notin Lit(D)$. Flip a to true. By construction

$\alpha^{a=1}(\mathbb{C}_1) = 1$, and $\mathbb{C}_2$ and D are not affected since $\{a, \overline{a}\} \cap \big(Lit(\mathbb{C}_2) \cup Lit(D)\big) = \emptyset$, so $\alpha^{a=1}(\mathbb{C}) = 1$ and $\alpha^{a=1}(D) = 0$, which is a contradiction. ∎

The fact that a minimally unsatisfiable CNF formula must have more clauses than variables seems to have been proven independently a number of times (see, for instance, [1, 8, 23, 38]). We will need the following formulation of this result, relating subsets of variables in a minimally implicating CNF formula and the clauses containing variables from these subsets.

THEOREM 7.4. *Suppose that F is a CNF formula that implies a clause D minimally. For any subset of variables V, let $F_V = \{C \in F \mid Vars(C) \cap V \neq \emptyset\}$. Then if $V \subseteq Vars(F) \setminus Vars(D)$, it holds that $|F_V| > |V|$. In particular, if F is minimally unsatisfiable, we have $|F_V| > |V|$ for all $V \subseteq Vars(F)$.*

Proof. The proof is by induction over $V \subseteq Vars(F) \setminus Vars(D)$.

If $|V| = 1$, then $|F_V| \geq 2$, since any $x \in V$ must occur both positively and negatively in F by Lemma 7.3.

The inductive step just generalizes the proof of Lemma 7.3. Suppose that $|F_{V'}| > |V'|$ for all strict subsets $V' \subsetneq V \subseteq Vars(F) \setminus Vars(D)$ and consider V. Since $F_{V'} \subseteq F_V$ if $V' \subseteq V$, choosing any V' of size $|V| - 1$ we see that $|F_V| \geq |F_{V'}| \geq |V'| + 1 = |V|$.

If $|F_V| > |V|$ there is nothing to prove, so assume that $|F_V| = |V|$. Consider the bipartite graph with the variables V and the clauses in F_V as vertices, and edges between variables and clauses for all variable occurrences. Since for all $V' \subseteq V$ the set of neighbors $N(V') = F_{V'} \subseteq F_V$ satisfies $|N(V')| \geq |V'|$, by Hall's marriage theorem there is a perfect matching between V and F_V. Use this matching to satisfy F_V assigning values to variables in V only.

The clauses in $F' = F \setminus F_V$ are not affected by this partial truth value assignment, since they do not contain any occurrences of variables in V. Furthermore, by the minimality of F it must hold that F' can be satisfied and D falsified simultaneously by assigning values to variables in $Vars(F') \setminus V$.

The two partial truth value assignments above can be combined to an assignment that satisfies all of F but falsifies D, which is a contradiction. Thus $|F_V| > |V|$. The theorem follows by induction. ∎

We need one final definition relating vertices of T and literal occurrences in clauses for the variables associated with these vertices.

DEFINITION 7.5. *We say that a vertex v is* represented positively *in a clause C if $\{v_1, \ldots, v_d\} \cap Lit(C) \neq \emptyset$ and* negatively *if $\{\overline{v}_1, \ldots, \overline{v}_d\} \cap Lit(C) \neq \emptyset$, and that C* mentions *v* positively *or* negatively*, respectively. This definition is extended to sets of vertices and clauses by taking unions.*

For a set of vertices U, we let $Vars^d(U) = \{u_1, \ldots, u_d \mid u \in U\}$ denote the set of all variables representing vertices in U. For a set of clauses $\mathbb{C}$, we use $V(\mathbb{C}) = \{u \in U \mid Vars^d(u) \cap Vars(\mathbb{C}) \neq \emptyset\}$ to denote all vertices represented (positively or negatively) in $\mathbb{C}$, and we write $\mathbb{C}[U] = \{C \in \mathbb{C} \mid V(C) \cap U \neq \emptyset\}$ to denote the subset of all clauses in $\mathbb{C}$ mentioning vertices in U.

We now prove by induction over the (sub)sets of induced pebbles that a clause configuration is at least as large as the number of pebbles it induces.

THEOREM 7.6. *Suppose that $\mathbb{C}$ is a set of clauses derived from $*Peb^d_T$ for $d \geq 2$ that induces the labeled pebble configuration $\mathbb{L}(\mathbb{C})$. Then $\mathsf{cost}(\mathbb{L}(\mathbb{C})) \leq |\mathbb{C}|$.*

Proof. Suppose that $\mathbb{C}$ induces a subconfiguration $v\langle W \rangle$. By Definition 6.6, there is a minimal support $V_v \subseteq T \setminus P^v$ with $W = swp(v, V_v) \subseteq V_v$ such that $\mathbb{C} \cup \mathbb{B}(V_v) \vDash A_{P^v}$ but $\mathbb{C} \cup \mathbb{B}(V_v) \nvDash A_{P^v_*}$ and $\mathbb{C} \cup \mathbb{B}(V'_v) \nvDash A_{P^v}$ for all $V'_v \subsetneq V_v$.

Fix for each induced subconfiguration $v\langle W \rangle$ with $W = swp(v, V_v)$ a subset $\mathbb{C}_v \subseteq \mathbb{C}$ such that $\mathbb{C}_v \cup \mathbb{B}(V_v) \vDash A_{P^v}$ minimally. Since $Vars(\mathbb{B}(V_v)) \cap Vars(A_{P^v}) = \emptyset$ by definition, using Lemma 7.1 with $D_1 = \bigvee_{i=1}^{d} v_i$ and $D_2 = A_{P_*^v}$, we see that the vertex v must be represented in $\mathbb{C}_v$ by some positive literal v_i. For the white pebbles in $W \subseteq V_v$, it follows for the same reason from Lemma 7.3 that all literals $\overline{w}_j$, $j \in [d]$, must be present in $\mathbb{C}_v$.

We prove by induction over $U \subseteq Bl(\mathbb{L}(\mathbb{C})) \cup Wh(\mathbb{L}(\mathbb{C}))$ that $|\mathbb{C}[U]| \geq |U|$, from which the theorem clearly follows. The base case $|U| = 1$ is immediate, since we just observed that all pebbled vertices $v \in V$ are represented in $\mathbb{C}$.

For the induction step, suppose that $\left|\mathbb{C}[U']\right| \geq \left|U'\right|$ for all $U' \subsetneqq U$. Pick a "topmost" vertex $u \in U$, i.e., such that $P_*^u \cap U = \emptyset$, and look at the subconfiguration $v\langle W \rangle$ containing u (with $u = v$ if u is black and u strictly below v otherwise) and the associated subset $\mathbb{C}_v \subseteq \mathbb{C}$ fixed above. Note that $\mathit{Vars}^d(U) \cap \mathit{Vars}(A_{P^v}) \subseteq \{u_1, \ldots, u_d\}$. Let $S = U \cap V(\mathbb{C}_v)$ be the set of all vertices in U mentioned by $\mathbb{C}_v$. We claim that $|\mathbb{C}_v[S]| \geq |S|$.

To show this, note first that $u \in S$ as was argued above, and if $S = \{u\}$ we trivially have $|\mathbb{C}_v[S]| \geq 1 = |S|$. Suppose therefore that $S \supsetneqq \{u\}$. We want to apply Theorem 7.4 on the formula $F = \mathbb{C}_v \cup \mathbb{B}(V_v)$, which as we recall implies A_{P^v} minimally. To this end, let $S' = S \setminus \{u\}$, write $S' = S_1 \mathbin{\dot\cup} S_2$ for $S_1 = S' \cap V_v$ and $S_2 = S' \setminus S_1$, and consider

$$
\begin{aligned}
F_{S'} &= \bigl\{C \in \bigl(\mathbb{C}_v \cup \mathbb{B}(V_v)\bigr) \mid V(C) \cap S' \neq \emptyset\bigr\} \\
&= \mathbb{C}_v[S'] \cup \mathbb{B}(S_1).
\end{aligned}
\tag{7.1}
$$

For each $w \in S_1$, the clauses in $\mathbb{B}(S_1)$ contain d literals $w_1, \ldots, w_d$, and these literals must all occur negated in $\mathbb{C}_v$ by Lemma 7.3. For each $w \in S_2$, the clauses in $\mathbb{C}_v[S']$ contain at least one variable w_i. Appealing to Theorem 7.4 with the subset of variables $\mathit{Vars}^d(S') \cap \mathit{Vars}(\mathbb{C}_v) \subseteq \mathit{Vars}(\mathbb{C}_v \cup \mathbb{B}(V_v)) \setminus \mathit{Vars}(A_{P^v})$, we get

$$
\begin{aligned}
\left|F_{S'}\right| &= \left|\mathbb{C}_v[S'] \cup \mathbb{B}(S_1)\right| \\
&\geq \left|\mathit{Vars}^d(S') \cap \mathit{Vars}(\mathbb{C}_v)\right| + 1 \\
&\geq d\left|S_1\right| + \left|S_2\right| + 1,
\end{aligned}
\tag{7.2}
$$

and rewriting this as

$$
\begin{aligned}
\left|\mathbb{C}_v[S]\right| &\geq \left|\mathbb{C}_v[S']\right| \\
&= \left|F_{S'}\right| - \left|\mathbb{B}(S_1)\right| \\
&\geq (d-1)\left|S_1\right| + \left|S_2\right| + 1 \\
&\geq \left|S\right|
\end{aligned}
\tag{7.3}
$$

proves the claim (this is where we use that $d \geq 2$).

Note that $\mathbb{C}_v[S] \subseteq \mathbb{C}[U]$, since $\mathbb{C}_v \subseteq \mathbb{C}$ and $S \subseteq U$. Also, by construction $\mathbb{C}_v[S]$ does not mention any vertices in $U \setminus S$ since $S = U \cap V(\mathbb{C}_v)$. In other words, $\mathbb{C}[U] \supseteq \mathbb{C}_v[S] \cup \mathbb{C}[U \setminus S]$ for $\mathbb{C}_v[S] \cap \mathbb{C}[U \setminus S] = \emptyset$, and using the induction hypothesis for $U \setminus S \subsetneqq U$ we get

$$
\left|\mathbb{C}[U]\right| \geq \left|\mathbb{C}_v[S]\right| + \left|\mathbb{C}[U \setminus S]\right| \geq |S| + |U \setminus S| = |U|. \tag{7.4}
$$

The theorem follows by induction. ☐

We can now prove Theorem 2.2 on page 65.

Theorem 2.2 (restated). *If π is a resolution refutation of a pebbling contradiction $Peb_{T_h}^d$ of degree $d > 1$ and $\mathcal{L}_\pi$ is the associated labeled pebbling from Theorem 2.1, then $\mathsf{cost}(\mathcal{L}_\pi) = \mathrm{O}(Sp(\pi))$.*

Proof. As in the proof of Theorem 2.1, given a refutation $\pi : Peb^d_{T_h} \vdash 0$, we use Lemma 6.2 to get a derivation $\pi^* = \{\mathbb{C}_0, \ldots, \mathbb{C}_\tau\}$ of $\bigvee_{i=1}^d z_i$ from $^*Peb^d_{T_h}$ in the same space and consider the L-pebbling $\mathcal{L}_\pi$ constructed in Theorem 6.15. Then

$$cost(\mathcal{L}_\pi) = \mathrm{O}\big(\max_{t\in[\tau]}\{cost(\mathbb{L}(\mathbb{C}_t))\}\big) \tag{7.5}$$

by Theorem 6.15, and for all $t \in [\tau]$ it holds that

$$cost(\mathbb{L}(\mathbb{C}_t)) \leq |\mathbb{C}_t| \tag{7.6}$$

by Theorem 7.6. Thus

$$cost(\mathcal{L}_\pi) = \mathrm{O}\big(\max_{t\in[\tau]}\{|\mathbb{C}_t|\}\big) = \mathrm{O}\big(Sp(\pi^*)\big) = \mathrm{O}\big(Sp(\pi)\big) \tag{7.7}$$

and the theorem follows. ∎

The proof of the tight bound for the refutation clause space of pebbling contradictions over binary trees in Theorem 1.1 as presented in section 2.3 is thereby complete.

8. Conclusion and open problems. We have proven an asymptotically tight bound on the refutation clause space in resolution of pebbling contradictions over binary trees. Our result is the first lower bound on resolution refutation space, which is not the consequence of a lower bound on the refutation width of the same formulas, but instead separates the two measures. This answers an open question in [13, 28, 30, 52]. However, we believe that it should be possible to strengthen our answer in several interesting ways.

First, we would like to determine the refutation space complexity of pebbling contradictions over binary trees in the stronger k-DNF resolution proof systems $\mathfrak{R}(k)$ introduced by Krajíček [37], where the lines in the proofs are k-DNF formulas instead of clauses and one can "resolve" over up to k variables simultaneously.

It is easy to prove the generalization of Theorem 4.6 that pebbling contradictions of degree d can be refuted in space $Sp_{\mathfrak{R}(k)}\big(Peb^d_G \vdash 0\big) = \mathrm{O}(1)$ in k-DNF resolution if $d \leq k$. For $d > k$, one could argue that it seems plausible that k-DNF resolution should be hard pressed to do anything better with $Peb^d_{T_h}$ than ordinary resolution (i.e., 1-DNF resolution) can do with $Peb^2_{T_h}$. But although the difference between resolution and k-DNF resolution might appear small, going from disjunctive clauses to 2-DNF formulas, or more generally from k-DNFs to $(k+1)$-DNFs, increases the proof power exponentially [50]. And while many lower bounds have been proven on k-DNF resolution proof length, for instance, in [2, 5, 6, 46, 50], it seems that the tools developed in these papers cannot be used to obtain lower bounds on space.

A careful reading of our proofs reveals that the only place where we actually use that the configurations in the derivations contain disjunctive clauses is in section 7. The proof in section 6 that resolution derivations induce labeled pebblings works just as well for derivations that use any sound derivation rules and operate with configurations containing arbitrary logical formulas (compare the remark after Observation 6.11). The main difficulty if one tries to prove a lower bound on k-DNF resolution refutation space along the lines of the current paper appears to be that one needs an analogue of Theorem 7.4 for minimally unsatisfiable sets of k-DNF formulas, with a strong lower bound on the number of k-DNF formulas in terms of the number of variables. This result should then be plugged into Theorem 7.6 to yield a lower bound for k-DNF resolution refutation space. Unfortunately, to the best of our knowledge no such bounds for minimally unsatisfiable sets of k-DNF formulas have been shown, and it is not even intuitively clear to us exactly what such an analogous result should look like.

Nevertheless, we believe that pebbling contradictions $Peb^k_{T_h} + 1$ separate k-DNF resolution and $(k+1)$-DNF resolution with respect to space.

CONJECTURE 1. *For k-DNF resolution refutations of pebbling contradictions on complete binary trees, fixing k it holds that $Sp_{\mathfrak{R}(k+1)}\big(Peb^k_{T_h} + 1 \vdash 0\big) = \mathrm{O}(1)$ but $Sp_{\mathfrak{R}(k)}\big(Peb^k_{T_h} + 1 \vdash 0\big) = \Omega(h)$.*

Proving this conjecture, or any nonconstant lower bound on the k-DNF resolution space $Sp_{\mathfrak{R}(k)}\big(Peb^k_{T_h} + 1 \vdash 0\big)$, would establish that the k-DNF resolution proof systems form a strict hierarchy with respect to space, which would be an improvement of the separation result in [28] for the restricted case of tree-like k-DNF resolution. Also, in our opinion, understanding the structure of minimally unsatisfiable sets of k-DNF formulas seems like an interesting (and challenging) combinatorial problem in its own right.

A second question, which was mentioned already in the introduction, is whether formulas refutable in short length can be arbitrarily complex with respect to refutation space. Let us discuss this question a bit more carefully to make clear what we mean by this.

For width, we know that if a k-CNF formula F is refutable in short length it must also be refutable in small width. More precisely, Theorem 3.1 tells us (rewriting the bound in terms of the number of clauses instead of variables using Theorem 7.4)[7] that if the width of refuting F is $\omega\big(\sqrt{|F| \log |F|}\big)$, then the length of refuting F must be superpolynomial in $|F|$. This is known to be almost tight, since [19] exhibits a k-CNF formulafamily $\{F_n\}_{n=1}^{\infty}$ with $W(F_n \vdash 0) = \Omega\big(\sqrt[3]{|F_n|}\big)$ but $L(F_n \vdash 0) = \mathrm{poly}(|F_n|)$. Hence, formula families refutable in polynomial length can have somewhat wide minimum-width refutations, but not arbitrarily wide ones.

What does the corresponding relation between space and length look like? Since $Sp(F \vdash 0) \geq W(F \vdash 0) + \mathrm{O}(1)$ by Theorem 3.7, it follows immediately from the preceding paragraph that k-CNF formulas refutable in polynomial length may have somewhat spacious minimum-space refutations. This gives a lower bound for any trade-off that we could hope to prove. At the other end of the spectrum, given any resolution refutation $\pi : F \vdash 0$, we can write down the DAG G_π corresponding to π (with $L(\pi)$ vertices) and then construct a space-efficient refutation by deriving and erasing clauses in the order of an optimal black pebbling of G_π (this is the original definition of the space of a derivation in [29]). Since it is known from [34] that any graph on n vertices can be black-pebbled in cost $\mathrm{O}(n/\log n)$, this shows that $Sp(F \vdash 0) = \mathrm{O}\big(L(F \vdash 0)/\log L(F \vdash 0)\big)$.

Now we can rephrase the question above about space and length in the following way: Is there a Ben-Sasson–Wigderson kind of lower bound, say, $L(F \vdash 0) = \exp\big(\Omega\big(Sp(F \vdash 0)^2/|F|\big)\big)$ or so, on length in terms of space? Or do there exist k-CNF formulas F with short refutations but maximum possible refutation space $Sp(F \vdash 0) = \Omega\big(L(F \vdash 0)/\log L(F \vdash 0)\big)$ in terms of length? (Note that the refutation length $L(F \vdash 0)$ must indeed be short in this case—essentially linear, since any formula F can be refuted in space $\mathrm{O}(|F|)$ by Theorem 3.5.) Or is the relation between refutation space and refutation length somewhere in between these extremes?

[7] Or, if one wants to be precise, actually using Theorem 3.8 in [41], since in general the formulas will not be minimally unsatisfiable.

We think that the true answer should be at the latter extreme, i.e., that space and length can be separated in the strongest sense possible.

CONJECTURE 2. *There is a family of unsatisfiable k-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size* $\mathrm{O}(n)$ *such that* $L(F_n \vdash 0) = \mathrm{O}(n)$ *but* $Sp(F_n \vdash 0) = \Omega(n/\log n)$.

The reason for our belief brings us over to our third and final open question: What is the refutation space complexity of pebbling contradictions over arbitrary graphs? We have determined the space complexity of pebbling contradictions over trees, but it would be nice to generalize this bound to pebbling contradictions over other DAGs that have better size-pebbling price trade-off. We believe that the lower bound on space in terms of black-white pebbling price should hold not just for trees, but for any DAG.

CONJECTURE 3. *For $d > 1$ and for an arbitrary DAG G with a unique target and with all vertices having indegree* 0 *or* 2*,* $Sp(Peb_G^d \vdash 0) = \Omega(\mathsf{BW\text{-}Peb}(G))$.

If we could prove Conjecture 3, we would immediately get a positive answer to Conjecture 2 as well. For it was shown in [32][8] that there are DAGs G_n of fan-in 2 and size $\mathrm{O}(n)$ that have black-white pebbling price $\mathsf{BW\text{-}Peb}(G_n) = \Theta(n/\log n)$. Thus, assuming Conjecture 3 and plugging in the pebbling contradictions defined over these DAGs G_n, we would get a k-CNF formulafamily $\{F_n\}_{n=1}^{\infty}$ of size $\mathrm{O}(n)$ with $L(F_n \vdash 0) = \mathrm{O}(n)$, $W(F_n \vdash 0) = \mathrm{O}(1)$, and $Sp(F_n \vdash 0) = \Omega(n/\log n)$. Note that this would also yield an almost optimal separation of space and width. (The best conceivable result would be a linear separation.)

However, it is not possible to prove Conjecture 3 by a simple generalization of the L-pebble game with reversal moves in section 5 to general DAGs G. As was observed in Lemma 5.3, because we allow moving black pebbles downward it is not true that $\mathsf{L\text{-}Peb}(G) = \Omega(\mathsf{BW\text{-}Peb}(G))$.

The problem with reversal moves arises because we do not a priori have any restrictions on what kind of clauses a resolution derivation from a pebbling contradiction might derive. For all candidate definitions of induced pebbles that we have been able to come up with (more or less radical variations of Definition 6.6), the example resolution derivations resulting in reversal moves that we have found all seem clearly nonoptimal (see, for instance, Remark 6.16), while all intuitively "reasonable" resolution derivations appear to yield well-behaved pebblings without reversals.

One way of circumventing this problem would be if one could define formally what constitutes a "reasonable" refutation of a pebbling contradiction and then show that each "unreasonable" refutation can be replaced by a "reasonable" one in asymptotically the same space. Alternatively, one could try to find new ideas for the connection between pebble games and resolution refutations of pebbling contradictions, perhaps experimenting with even more general games than the labeled pebble game in this paper.

Appendix. The labeled pebbling price of binary trees. In this appendix we present a proof of Theorem 5.4 on page 77, i.e., that for binary trees, the L-pebbling price coincides with the black-white pebbling price up to (small) constant factors. Since the argument is quite lengthy, we begin by giving an outline of its structure.

A.1. High-level overview of proof. The proof of the theorem consists of two main components. The first component is pretty straightforward and is taken care of in section A.2. The second component is much more involved and takes up the

[8]Note that in several papers, this result is incorrectly attributed to [39], but [39] itself gives the correct reference.

rest of the appendix. In this first subsection we discuss these two parts of the proof informally, state the two corresponding formal lemmas that we will need, and show how they together yield Theorem 5.4.

For the first part, studying Definition 5.2 on page 75 carefully, one can argue that if we remove the reversal rule from the labeled pebble game, what remains looks essentially just like a disguised version of the standard black-white pebble game in Definition 4.1 on page 69. True, the rule for white pebble removal has been somewhat changed, and we are grouping pebbles together in pebble subconfigurations, but if we take any "sensible" L-pebbling $\mathcal{L} = \{\mathbb{L}_0, \ldots, \mathbb{L}_\tau\}$, ignore this pebble grouping, and just look at how the set of all black and white pebbles $\big(Bl(\mathbb{L}_t), Wh(\mathbb{L}_t)\big)$ changes over time with t, it seems plausible that we should obtain something pretty close to a standard black-white pebbling.

This is indeed the case, and we formalize the intuition above in section A.2, where we prove the following lemma.

Lemma A.1. *Suppose that G is an arbitrary DAG with unique sink, and let $\mathcal{L}$ be any complete L-pebbling of G without reversal moves. Then from $\mathcal{L}$ we can construct a complete black-white pebbling $\mathcal{P}$ of G such that $\mathsf{cost}(\mathcal{P}) \leq \mathsf{cost}(\mathcal{L})$.*

Thus, if we could somehow do away with the reversal moves without increasing pebbling price, we would be done. Recall that we know from Lemma 5.3 on page 76 that for general DAGs, this *cannot* be done. The counterexample in Lemma 5.3 does not apply to binary trees, though. Rather, on the contrary, toying around with L-pebblings of small binary trees, one cannot help getting the feeling that the removal of reversals should not affect the L-pebbling price in any way whatsoever. To show this formally, we need to make a detailed analysis of L-pebblings of trees and find out what structural properties can help us get rid of reversal moves in this special case. This is the second, much harder, component in the proof of Theorem 5.4.

In section A.3, we present some further definitions and notation that we will use when studying this problem, and we make some useful technical observations. The rest of the appendix is then spent proving that for binary trees, the rule for reversal can in fact be omitted from the L-pebble game. We do not quite get the result that the pebbling price is not affected at all by this, but we show that it cannot increase by more than a constant factor 2. Such a bound is wholly sufficient for our purposes.

Unfortunately, the proof of this fact is *very* technical, but the structure of the underlying argument is not that complicated. Below, we try to sketch what it looks like to give the reader an idea of where we are going.

1. We first take care of a minor technical issue. In the pebble configurations of the standard black-white pebble game, we have black pebbles and, below each black pebble, the white pebbles it depends on, with nothing in between. In contrast, in the L-pebble game there can be other black pebbles in between a black pebble and its white pebbles, or two black pebbles, one above the other, without any white pebbles below them (see, for example, $v_1\langle v_2, v_6\rangle$ and $v_7\langle\emptyset\rangle$ in Figure 10 on page 100, or $v\langle v_1, v_2, v_3\rangle$ and $w\langle w_4, w_5\rangle$ in Figure 11(a) on page 103).
   The first step in our elimination of reversal moves is to show that this difference is inconsequential. Namely, we establish that without loss of generality we can assume that any L-pebbling $\mathcal{L}$ is *nonoverlapping* in the sense that, roughly speaking, different pebble subconfigurations in the same labeled pebble configuration do not intersect (Definition A.21 and Lemma A.26 in section A.4).
2. Next, we study the connection between reversal moves and the set of vertices

covered by the pebble subconfigurations in the sense of Definition 5.1 on page 74. In a standard black-white pebbling of a binary tree T, the set of vertices covered (generalizing Definition 5.1 in the natural way) expands monotonically as the pebbling proceeds, but in an L-pebbling it might also shrink as a result of reversal moves.

As was discussed above, our intuition is that for trees this "shrinking" should not help to produce cheaper pebblings. As a part of our attempt to understand what happens during reversal moves, we observe that if we restrict an L-pebbling $\mathcal{L}$ to a subset of the vertices in T and let $\mathcal{L}$ act on these vertices in the natural way, we get a legal L-pebbling on this subset of vertices. We refer to this restriction operation as *projection* (Definition A.22 and Lemma A.28 in section A.5).

3. This leads to the idea of trying to get rid of reversal moves altogether in the following way: When the cover of a labeled pebble configuration shrinks as the result of a reversal move, we eliminate this reversal by projecting the L-pebbling moves made so far on what remains *after* the reversal move. We know that every such projection results in a legal L-pebbling, and if we do this by forward induction for all reversal moves in $\mathcal{L}$, we get a reversal-free complete L-pebbling $\mathcal{L}'$ of T (section A.6).
4. The problem is that these projection operations do not preserve pebbling cost—the pebbling $\mathcal{L}$ may contain reversal moves such that the projected pebbling $\mathcal{L}'$ becomes more expensive than $\mathcal{L}$. We identify which kind of reversals in $\mathcal{L}$ spoil our construction of a reversal-free and cheap pebbling $\mathcal{L}'$ by projection and note that, from a global perspective, such *wasteful* reversal moves seem clearly nonoptimal (Example A.29).

   Encouraged by this, and allowing some temporary wishful thinking, we then demonstrate that, for all L-pebblings that contain reversal moves but avoid this special class of wasteful reversals, the projection construction sketched above works (Definition A.30 and Lemma A.32 in section A.7).
5. In this way, the whole problem finally boils down to whether wasteful reversals can be eliminated. In general, we cannot assume that an L-pebbling $\mathcal{L}$ does not make wasteful reversal moves, but we show that if $\mathcal{L}$ contains such moves, we can construct another L-pebbling $\mathcal{L}'$ in which these wasteful reversals are replaced by stronger, nonwasteful moves without increasing the total pebbling cost by more than a constant factor (Lemma A.37 in section A.8).

Summing this up, we get the next lemma.

LEMMA A.2. *Suppose that $\mathcal{L}$ is a complete L-pebbling of a complete binary tree T. Then from $\mathcal{L}$ we can construct a complete L-pebbling $\mathcal{L}'$ of T without reversals such that* $\mathsf{cost}(\mathcal{L}') = \mathrm{O}(\mathsf{cost}(\mathcal{L}))$.

Assuming Lemmas A.1 and A.2, it is easy to prove that the L-pebbling price and the black-white pebbling price of a complete binary tree T_h of height h coincide asymptotically.

THEOREM 5.4 (restated). $\mathsf{L\text{-}Peb}(T_h) = \Theta\big(\mathsf{BW\text{-}Peb}(T_h)\big)$.

Proof. The black pebbling price of T_h is $\mathsf{Peb}(T_h) = \mathrm{O}(h) = \mathrm{O}(\mathsf{BW\text{-}Peb}(T_h))$ according to Theorem 4.3 on page 70. It is not hard to see that an L-pebbling $\mathcal{L}$ can imitate a black pebbling $\mathcal{P}$ in the same cost. For suppose that at some point in time t a black pebble is placed on the vertex r in $\mathcal{P}$. If r is a source, $\mathcal{L}$ can match this move by introducing $r\langle\emptyset\rangle$. Otherwise, if $\mathit{pred}(r) = \{p, q\}$, both these vertices must be black-pebbled at time t in $\mathcal{P}$, so by induction we have $p\langle\emptyset\rangle$ and $q\langle\emptyset\rangle$ in $\mathcal{L}$. Introducing $r\langle p, q\rangle$, merging with $p\langle\emptyset\rangle$ and $q\langle\emptyset\rangle$ on p and q, respectively, and then erasing $r\langle p, q\rangle$, we get $r\langle\emptyset\rangle$. Thus $\textsf{L-Peb}(T_h) \leq \textsf{Peb}(T_h) = \mathrm{O}(\textsf{BW-Peb}(T_h))$.

In the other direction, let $\mathcal{L}$ be a complete L-pebbling of T_h in minimal cost. By Lemma A.2, there exists a complete L-pebbling $\mathcal{L}'$ of T_h without reversal moves such that $\textsf{cost}(\mathcal{L}') = \mathrm{O}(\textsf{cost}(\mathcal{L}))$. By Lemma A.1 we can construct a plain old black-white pebbling $\mathcal{P}$ of T_h from $\mathcal{L}'$ for which $\textsf{cost}(\mathcal{P}) \leq \textsf{cost}(\mathcal{L}')$. Hence $\textsf{BW-Peb}(T_h) = \mathrm{O}(\textsf{L-Peb}(T_h))$, and the theorem follows. ☐

So all that needs to be done is to prove Lemmas A.1 and A.2, which we do starting in the next subsection.

We make one final remark before plunging into the proofs. We are aware that the technical machinery in this appendix can appear cumbersome. However, this might mainly be due to the fact that sometimes, one picture says more than the thousand words used to formalize it mathematically. We feel that at times in this appendix, we are forced to go to great lengths to prove statements that seem intuitively very plausible once one visualizes what they actually say. Therefore, we believe that the arguments should be possible to follow more easily if the reader tries to digest what the definitions mean and what is proven about them simply by drawing a binary tree of suitable height and working out small examples in this binary tree while reading.

A.2. Reversal-free L-pebblings are (almost) standard pebblings. We present the proof of Lemma A.1 in two steps, one easy and one harder.

The first modification of the pebble game when going from Definition 4.1 to Definition 5.2 was that in the context of resolution, it appears that a more natural rule for white pebble removal is that a white pebble can be removed from a vertex when a black pebble is placed on that same vertex. It is thanks to this that we get the close correspondence between clauses and pebbles in section 6.

It seems intuitively fairly obvious that this rule change should not really affect the pebble game, but for completeness we state and prove this fact formally.

DEFINITION A.3 (S-pebble game). *Suppose that G is a DAG with unique sink z. The* superpositioned black-white pebble game*, or* S-pebble game*, is as in Definition* 4.1*, except that a vertex may have both a black and a white pebble on itself, and the pebbling rules are* 1–3 *in Definition* 4.1 *and* 4' *below instead of rule* 4 *in Definition* 4.1*.*

4'. *A white pebble on v can be removed only if there is a black pebble on v.*

We write $\textsf{S-Peb}(G)$ to denote the minimum cost of any complete S-pebbling of G.

LEMMA A.4. *For any DAG G it holds that $\textsf{S-Peb}(G) = \textsf{BW-Peb}(G)$.*

Proof. It is easy to see that for any standard black-white pebbling $\mathcal{P}$ of G we can make an S-pebbling $\mathcal{S}$ of G in exactly the same cost. Every white pebble removal from a vertex v in $\mathcal{P}$ according to rule 4 corresponds to first placing a black pebble on v in $\mathcal{S}$ in no extra cost and then removing first the white pebble according to rule 4' and then the black pebble according to rule 2.

In the other direction, suppose that we are given a superpositioned pebbling $\mathcal{S} = \{\mathbb{S}_0, \ldots, \mathbb{S}_\tau\}$ of G. We construct a standard black-white pebbling $\mathcal{P} = \{\mathbb{P}_0, \ldots, \mathbb{P}_\tau\}$ such that for $\mathbb{P}_t = (B_t, W_t)$ and $\mathbb{S}_t = (B'_t, W'_t)$ it holds that $B_t = B'_t$, $B_t \cup W_t = B'_t \cup W'_t$, and (as required by Definition 4.1) $B_t \cap W_t = \emptyset$. In particular, this means that $\textsf{cost}(\mathcal{P}) = \textsf{cost}(\mathcal{S})$ and that if $\mathcal{S}$ is a complete pebbling, then so is $\mathcal{P}$.

The construction is by forward induction over $\mathcal{S}$. We set $\mathbb{P}_0 = \mathbb{S}_0 = (\emptyset, \emptyset)$ and then make the inductive step by a case analysis over the pebbling moves.

1. If $\mathcal{S}$ places a black pebble on v at time $t+1$, the vertices in $\mathit{pred}(v)$ must be pebbled in $\mathbb{S}_t$ and thus in $\mathbb{P}_t$. If $v \in W_t$, we remove the white pebble from v in $\mathcal{P}$. Then we place a black pebble on v.

2. If $\mathcal{S}$ removes a black pebble from v at time $t+1$, by induction v is black-pebbled and the vertices in $pred(v)$ are pebbled in $\mathcal{P}$. Thus we can remove the black pebble from v in $\mathcal{P}$, and in case $v \in W'_t$ we then place a white pebble on v.
3. If $\mathcal{S}$ places a white pebble on v at time $t+1$, we place a white pebble there in $\mathcal{P}$ if $v \notin B_t$ and otherwise do nothing.
4. When a white pebble is removed from v in $\mathcal{S}$ it holds that $v \in B'_t$. Then by induction $v \in B_t$, so the white pebble has already been removed from v in $\mathcal{P}$ or was never placed there.

Note that to avoid being overly formalistic, we ignore the fact there there might be "idle moves" $\mathbb{P}_t = \mathbb{P}_{t+1}$ and moves simultaneously removing and placing a pebble on the same vertex in $\mathcal{P}$ and $\mathcal{S}$. It should be clear that this is not a problem. ∎

The second step in the proof of Lemma A.1 is to show that if we take a complete L-pebbling $\mathcal{L} = \{\mathbb{L}_0, \ldots, \mathbb{L}_\tau\}$ of a DAG G without reversal moves and look at $\big(Bl(\mathbb{L}_t), Wh(\mathbb{L}_t)\big)$ for $t \in [\tau]$, we can extract a legal complete S-pebbling of G in at most the same cost. We prove this in the next two lemmas.

The first lemma says that without loss of generality we can assume that all L-pebblings are *nonredundant* in the sense that if a subconfiguration $v\langle V\rangle$ is derived at time t, then this subconfiguration is not just thrown away but is used at some time $t' > t$ further on in the pebbling before being erased.

From now on, in order not to clutter the notation we allow a mild abuse of notation by omitting curly brackets around singleton L-configurations, quite often writing, for instance, $v\langle V\rangle \preceq \mathbb{L}$, $u\langle U\rangle = \mathbb{L}$, and $\mathbb{L} \cup w\langle W\rangle$ instead of $\{v\langle V\rangle\} \preceq \mathbb{L}$, $\{u\langle U\rangle\} = \mathbb{L}$, and $\mathbb{L} \cup \{w\langle W\rangle\}$. Also, we sometimes drop the curly brackets around singleton sets within subconfigurations, writing, for instance, $v\langle (V \cup W) \setminus w\rangle$ instead of $v\langle (V \cup W) \setminus \{w\}\rangle$ for the merger of $v\langle V\rangle$ and $w\langle W\rangle$.

LEMMA A.5. *Let $\mathcal{L} = \{\mathbb{L}_0, \ldots, \mathbb{L}_\tau\}$ be an arbitrary complete L-pebbling of a DAG G. Then we can construct a complete L-pebbling $\mathcal{L}' = \{\mathbb{L}'_0, \ldots, \mathbb{L}'_{\tau'}\}$ of G with $\mathsf{cost}(\mathcal{L}') \leq \mathsf{cost}(\mathcal{L})$ that has the following property: If $v\langle V\rangle$ is erased at time t in $\mathcal{L}'$, i.e., $v\langle V\rangle \in \mathbb{L}'_t \setminus \mathbb{L}'_{t+1}$, then this subconfiguration has been used in a merger or reversal move immediately before being erased, and the subconfiguration resulting from this move is present in $\mathbb{L}'_{t+1}$. Also, if $\mathcal{L}$ makes no reversal moves, then neither does $\mathcal{L}'$.*

Proof. Let us first try to visualize the proof. For any L-pebbling $\mathcal{L}$, we can construct a DAG $G_{\mathcal{L}}$ encoding the pebbling as follows. For every subconfiguration $v\langle V\rangle$ appearing at time t_1 and staying in the graph until time t_2 when it is erased, we create a vertex $(v\langle V\rangle, [t_1, t_2])$. For each reversal $u\langle U\rangle \rightsquigarrow v\langle V\rangle$, we draw an edge from the vertex representing this occurrence of $u\langle U\rangle$ to the vertex representing this occurrence of $v\langle V\rangle$. For each merger $u\langle U\rangle = \mathsf{merge}(v\langle V\rangle, w\langle W\rangle)$, we draw edges from $v\langle V\rangle$ and $w\langle W\rangle$ to $u\langle U\rangle$. The sources in $G_{\mathcal{L}}$ are vertices $(v\langle pred(v)\rangle, [t_1, t_2])$, and by assumption there is a sink $(z\langle\emptyset\rangle, [t_1, \tau])$. Note that by the definition of the L-pebble game we never derive a subconfiguration that is already present in the graph, so all vertices in $G_{\mathcal{L}}$ have indegree 0, 1, or 2 corresponding to introductions, reversals, and mergers.

Consider the subgraph of $G_{\mathcal{L}}$ consisting of all vertices from which the sink vertex $(z\langle\emptyset\rangle, [t_1, \tau])$ is reachable. We construct $\mathcal{L}'$ to be the subpebbling corresponding exactly to the moves in this subgraph, except that erasures are always performed as soon as possible. Since the moves in $\mathcal{L}'$ are a subset of the moves in $\mathcal{L}$, clearly $\mathcal{L}'$ is reversal-free if $\mathcal{L}$ is.

Formally, this amounts to the following. We construct the modified pebbling $\mathcal{L}'$ by backward induction over $\mathcal{L} = \{\mathbb{L}_0, \ldots, \mathbb{L}_\tau\}$. Let $\mathbb{L}'_\tau = \mathbb{L}_\tau = \{z\langle\emptyset\rangle\}$. Our induction hypothesis is that $\mathbb{L}'_{t^*} \subseteq \mathbb{L}_{t^*}$ for $t^* > t$. The backward induction step from $t+1$ to t is a case analysis over the moves $\mathbb{L}_t \rightsquigarrow \mathbb{L}_{t+1}$ in $\mathcal{L}$. For simplicity, we allow using fractional time steps in the interval $[t, t+1]$ in the inductive constructions below.

Introduction. $\mathbb{L}_{t+1} = \mathbb{L}_t \cup v\langle pred(v)\rangle$: Set $\mathbb{L}'_t = \mathbb{L}'_{t+1} \setminus v\langle pred(v)\rangle$. Note that we might have $\mathbb{L}'_t = \mathbb{L}'_{t+1}$ if $v\langle pred(v)\rangle \notin \mathbb{L}'_{t+1}$. In any case, the induction hypothesis holds for $\mathbb{L}'_t$.

Merger. $\mathbb{L}_{t+1} = \mathbb{L}_t \cup v\langle (V \cup W) \setminus w\rangle$: If $v\langle (V \cup W) \setminus w\rangle \notin \mathbb{L}'_{t+1}$, set $\mathbb{L}'_t = \mathbb{L}'_{t+1}$. The induction hypothesis trivially remains true. Otherwise, if the merged subconfiguration is present in $\mathbb{L}'_{t+1}$, set $\mathbb{L}'_t = \big(\mathbb{L}'_{t+1} \cup \{v\langle V\rangle, w\langle W\rangle\}\big) \setminus v\langle (V \cup W) \setminus w\rangle$. We can go from $\mathbb{L}'_t$ to $\mathbb{L}'_{t+1}$ in at most three steps via intermediate L-configurations $\mathbb{L}'_{t+1/3} = \mathbb{L}'_t \cup v\langle (V \cup W) \setminus w\rangle$ and $\mathbb{L}'_{t+2/3} = \mathbb{L}'_{t+1} \cup w\langle W\rangle$ by first merging $v\langle V\rangle$ and $w\langle W\rangle$, then possibly erasing $v\langle V\rangle$, and finally possibly erasing $w\langle W\rangle$.

Reversal. $\mathbb{L}_{t+1} = \mathbb{L}_t \cup v\langle V\rangle$ for $v\langle V\rangle \prec u\langle U\rangle \in \mathbb{L}_t$: If $v\langle V\rangle \notin \mathbb{L}'_{t+1}$, set $\mathbb{L}'_t = \mathbb{L}'_{t+1}$. Otherwise, set $\mathbb{L}'_t = \big(\mathbb{L}'_{t+1} \cup u\langle U\rangle\big) \setminus v\langle V\rangle$. We can go from $\mathbb{L}'_t$ to $\mathbb{L}'_{t+1}$ in at most two steps via the intermediate L-configuration $\mathbb{L}'_{t+1/2} = \mathbb{L}'_{t+1} \cup u\langle U\rangle$, i.e., by first reversing $u\langle U\rangle$ to $v\langle V\rangle$ and then possibly erasing $u\langle U\rangle$.

Erasure. $\mathbb{L}_{t+1} = \mathbb{L}_t \setminus v\langle V\rangle$: All erasure moves in $\mathcal{L}'$ are taken care of in connection with mergers or reversals, so set $\mathbb{L}'_t = \mathbb{L}'_{t+1}$.

We claim that all moves in $\mathcal{L}'$ constructed in this way are legal (if we eliminate repeated L-configurations $\mathbb{L}'_t = \mathbb{L}'_{t+1}$). For if $u\langle U\rangle \in \mathbb{L}'_t$, then $u\langle U\rangle \in \mathbb{L}_t$, and we know that this subconfiguration must have been derived at some point in time $t^* \leq t$ in $\mathcal{L}$ by introduction, merger, or reversal. Thus the backward construction of $\mathcal{L}'$ will yield a correct derivation of $u\langle U\rangle$. Also note that by the construction for the merger and reversal moves, when a subconfiguration in $\mathcal{L}'$ is erased it has just been used in some merger or reversal move.

Finally, by construction $\mathbb{L}'_t \subseteq \mathbb{L}_t$, and for the intermediate fractional time step L-configurations $\mathbb{L}'_{t+a/b}$ in the merger and reversal moves in $\mathcal{L}'$ we have $\mathbb{L}'_{t+a/b} \subseteq \mathbb{L}_{t+1}$. This shows that for all $\mathbb{L}' \in \mathcal{L}'$ there is a corresponding $\mathbb{L} \in \mathcal{L}$ such that $\mathsf{cost}(\mathbb{L}') \leq \mathsf{cost}(\mathbb{L})$ (part 1 of Proposition 6.14). It follows that $\mathsf{cost}(\mathcal{L}') \leq \mathsf{cost}(\mathcal{L})$. ∎

For L-pebblings as in Lemma A.5, if we ignore all relations between black and white pebbles in the subconfigurations and consider $\big(Bl(\mathbb{L}_t), Wh(\mathbb{L}_t)\big)$ for $t \in [\tau]$, this is a legal S-pebbling.

LEMMA A.6. *Suppose that $\mathcal{L}$ is a complete L-pebbling of a DAG G without reversal moves. Then there is a complete S-pebbling $\mathcal{S}$ of G such that $\mathsf{cost}(\mathcal{S}) \leq \mathsf{cost}(\mathcal{L})$.*

Proof. By Lemma A.5, without loss of generality we can assume that each $v\langle V\rangle$ is erased from $\mathcal{L}$ precisely after it has been used in a merger, and that $v\langle V\rangle$ is erased before $w\langle W\rangle$ when both subconfigurations are eliminated after a move $v\langle (V \cup W) \setminus w\rangle = \mathsf{merge}(v\langle V\rangle, w\langle W\rangle)$, so that the white pebble on w is removed before the black pebble on w.

It is clear that we are done if we can construct an S-pebbling $\mathcal{S}$ with moves matching the moves in $\mathcal{L}$ exactly. Let $\mathbb{S}_0 = (\emptyset, \emptyset)$ and construct $\mathbb{S}_{t+1}$ inductively by looking at the moves in $\mathbb{L}_t \rightsquigarrow \mathbb{L}_{t+1}$.

Introduction. $\mathbb{L}_{t+1} = \mathbb{L}_t \cup v\langle pred(v)\rangle$: Place white pebbles on $pred(v)$ and then a black pebble on v in $\mathcal{S}$.

Merger. $\mathbb{L}_{t+1} = \mathbb{L}_t \cup v\langle (V \cup W) \setminus w\rangle$ for $v\langle V\rangle, w\langle W\rangle \in \mathbb{L}_t$: No pebbling moves in $\mathcal{S}$, but note that if $v\langle V\rangle$ is now removed, the change in pebbles on G in $\mathcal{L}$ is exactly the same as after an application of rule 4' on w.

Erasure. $\mathbb{L}_{t+1} = \mathbb{L}_t \setminus v\langle V\rangle$: This is the only nontrivial case. In general, an erasure move in an L-pebbling can remove an arbitrary number of white pebbles without any black pebbles being even close to these white pebbles, and there is no way we can match such a move in an S-pebbling. But since we can assume that $\mathcal{L}$ is an L-pebbling as described in Lemma A.5, we know that $v\langle V\rangle$ has just been used in a merger. Consequently, the only pebble that disappears when going from $\big(Bl(\mathbb{L}_t), Wh(\mathbb{L}_t)\big)$ to $\big(Bl(\mathbb{L}_{t+1}), Wh(\mathbb{L}_{t+1})\big)$ is either the black pebble on v, which is always a legal pebble removal, or some white pebble on $w \in V$ which has just been eliminated in the merger move by a black pebble, and this is a legal pebble removal according to rule 4'.

We see that $\mathcal{S}$ generated in this way is a legal S-pebbling if we modify each introduction step into three pebble placement moves. Clearly, $\mathsf{cost}(\mathcal{S}) \le \mathsf{cost}(\mathcal{L})$. The lemma follows. ∎

Combining Lemmas A.4 and A.6 immediately yields Lemma A.1.

Lemma A.1 (restated). *Suppose that G is an arbitrary DAG with unique sink, and let $\mathcal{L}$ be any complete L-pebbling of G without reversals. Then from $\mathcal{L}$ we can construct a complete black-white pebbling $\mathcal{P}$ of G such that $\mathsf{cost}(\mathcal{P}) \le \mathsf{cost}(\mathcal{L})$.*

Proof. Given any L-pebbling $\mathcal{L}$ of G without reversal moves, we use Lemma A.6 to find an S-pebbling $\mathcal{S}$ in at most the same cost as $\mathcal{L}$. Then Lemma A.4 helps us to transform $\mathcal{S}$ to a standard black-white pebbling $\mathcal{P}$ in at most the same cost as $\mathcal{S}$. ∎

A.3. Some technical preliminaries. In the rest of this appendix, we restrict our attention to binary trees and show that for such graphs the reversal rule can be omitted in the labeled pebble game. Before beginning to construct the proof of this statement, in this subsection we collect a number of technical observations that will simplify matters later on. In the process, we also introduce some more definitions and notation.

Recall the terminology and notation from the beginning of section 5 and from Definitions 6.3 and 6.4. We add that, in this appendix, P and Q will denote paths in T. Also, if $succ(u) = succ(v)$ for $u \neq v$, we will say that u and v are *siblings* and write $v = sibl(u)$. Note that siblings are unrelated vertices in the sense of section 5; i.e., there is no (directed) path between u and v.

We observe that for binary trees, the cover of a subconfiguration can be defined more explicitly than in Definition 5.1 and also has the following convexity property.

Definition A.7. *We say that a vertex set $V \subseteq V(G)$ in a DAG G is* convex *if for all $u_1, u_2 \in V$ there is a $u^* \in V$ above both u_1 and u_2 such that for all paths $P_i : u_i \rightsquigarrow u^*$, $i = 1, 2$, it holds that $P_i \subseteq V$.*

Proposition A.8. *For any pebble subconfiguration $v\langle W\rangle$ in a binary tree T it holds that $cover(v\langle W\rangle) = T^v \setminus \bigcup_{w\in W} T^w$. In particular, $cover(v\langle W\rangle)$ is a convex set.*

This is not true for general DAGs. Consider, for instance, the pyramid graph of height 4 with vertex labels as in Figure 6 on page 76. Then for $z\langle u_2, u_3\rangle$ it holds that $s_2, s_4 \in cover(z\langle u_2, u_3\rangle)$, but for any vertex above both s_2 and s_4 we can always pick paths going through $u_2, u_3 \notin cover(z\langle u_2, u_3\rangle)$ so $cover(z\langle u_2, u_3\rangle)$ is not convex.

Proof of Proposition A.8. The set inclusion $T^v \setminus \bigcup_{w\in W} T^w \subseteq cover(v\langle W\rangle)$ is straightforward. Since all vertices $u \in T^v \setminus \bigcup_{w\in W} T^w$ are below v, they have paths $P : u \rightsquigarrow v$ to v. But no u is below any $w \in W$, so the paths P cannot possibly intersect W. Thus $u \in cover(v\langle W\rangle)$ according to Definition 5.1.

To show that this is an equality, we have to make use of the fact that T is a tree. Namely, this implies that the path $P : u \rightsquigarrow v$, if it exists, must be unique. Suppose to get a contradiction that $u \in cover(v\langle W\rangle)$ but $u \notin T^v \setminus \bigcup_{w\in W} T^w$. By definition there is a path $P : u \rightsquigarrow v$, so $u \in T^v$. It follows that there must exist some $w \in W$ such that $u \in T^w$. But then the unique path $P : u \rightsquigarrow v$ must pass through w, so $P \cap W \neq \emptyset$, contradicting the assumption that $u \in \mathit{cover}(v\langle W\rangle)$.

To prove convexity, just set $u^* = v$ in Definition A.7 and use that the path between any two vertices in T is uniquely determined. ∎

A nice property of mergers in binary trees is that if we merge two simple subconfigurations (Definition 6.4), then the resulting subconfiguration is also simple. We remark that this is not true in more general DAGs. If we look at Figure 6 again, the subconfigurations $z\langle x_2, u_2, u_3\rangle$ and $x_2\langle s_3\rangle$ are both simple, but their merger $z\langle u_2, u_3, s_3\rangle$ is not.

Observation A.9. *If $v\langle V\rangle$ and $w\langle W\rangle$ with $w \in V$ are simple subconfigurations in a binary tree, then $\mathsf{merge}(v\langle V\rangle, w\langle W\rangle)$ is also simple.*

Proof. By definition, $\mathsf{merge}(v\langle V\rangle, w\langle W\rangle) = v\langle (V \cup W) \setminus \{w\}\rangle$. Since V is simple and we are in a binary tree, it holds that $T^w \cap \bigcup_{x \in V\setminus\{w\}} T^x = \emptyset$. To get the required paths from $u \in W$ to v in Definition 6.4, just concatenate the paths from $u \in W$ to w with the path from w to v. ∎

Another nice property of mergers of simple subconfigurations is that the cover of a merger is the disjoint union $\dot\cup$ of the covers of the merged subconfigurations. Figure 5 on page 76 provides an illustration of this. Again, this holds only in the binary tree case. Reusing the example subconfigurations $z\langle x_2, u_2, u_3\rangle$ and $x_2\langle s_3\rangle$ above, it is readily verified that $\mathit{cover}(\mathsf{merge}(z\langle x_2, u_2, u_3\rangle, x_2\langle s_3\rangle)) = \mathit{cover}(z\langle u_2, u_3, s_3\rangle) \neq \mathit{cover}(z\langle x_2, u_2, u_3\rangle) \cup \mathit{cover}(x_2\langle s_3\rangle)$.

Proposition A.10. *Suppose that $u\langle U\rangle$, $v\langle V\rangle$, and $w\langle W\rangle$ are simple pebble subconfigurations in a binary tree. Then it holds that $u\langle U\rangle = \mathsf{merge}(v\langle V\rangle, w\langle W\rangle)$ if and only if $\mathit{cover}(u\langle U\rangle) = \mathit{cover}(v\langle V\rangle) \,\dot\cup\, \mathit{cover}(w\langle W\rangle)$.*

Proof. ($\Rightarrow$) If $u\langle U\rangle = \mathsf{merge}(v\langle V\rangle, w\langle W\rangle)$, it holds that $w \in V$, and since we are in a tree and V is a simple set, we have $T^w \cap \bigcup_{x \in V\setminus w} T^x = \emptyset$. Combining this with the fact that W is below w by definition, we get $\bigcup_{y \in W} T^y \subseteq T^w$ and $\bigcup_{x \in V\setminus w} T^x \cap \bigcup_{y\in W} T^y = \emptyset$. The equality in the proposition follows by using Proposition A.8 and checking that

$$
\begin{aligned}
\mathit{cover}(u\langle U\rangle) &= \mathit{cover}\big(v\big\langle (V \cup W) \setminus w\big\rangle\big) \\
&= T^v \setminus \bigcup_{x \in (V\cup W)\setminus w} T^x \\
&= T^v \setminus \left(\bigcup_{x \in V\setminus w} T^x \,\dot\cup\, \bigcup_{y \in W} T^y \right) \\
&= \left(T^v \setminus \bigcup_{x \in V} T^x \right) \dot\cup \left(T^w \setminus \bigcup_{y \in W} T^y \right) \\
&= \mathit{cover}(v\langle V\rangle) \,\dot\cup\, \mathit{cover}(w\langle W\rangle).
\end{aligned}
\tag{A.1}
$$

($\Leftarrow$) Suppose that $\mathit{cover}(u\langle U\rangle) = \mathit{cover}(v\langle V\rangle) \,\dot\cup\, \mathit{cover}(w\langle W\rangle)$. Since by definition all vertices in $\mathit{cover}(v\langle V\rangle)$ and $\mathit{cover}(w\langle W\rangle)$ are below v and w, respectively, but $\mathit{cover}(v\langle V\rangle) \cup \mathit{cover}(w\langle W\rangle) = \mathit{cover}(u\langle U\rangle)$ is convex by Proposition A.8, setting $u_1 = v$ and $u_2 = w$ in Definition A.7 shows that either v is below w or w is below v. Suppose without loss of generality that the latter case holds. Then using the same reasoning again we see that $v = u$.

Since $w \in \mathit{cover}(u\langle U\rangle)$ there is a path $P : w \rightsquigarrow u$ with $P \subseteq \mathit{cover}(u\langle U\rangle)$. Clearly, $(P \setminus w) \cap \mathit{cover}(w\langle W\rangle) = \emptyset$. Because $\mathit{cover}(v\langle V\rangle) \cap \mathit{cover}(w\langle W\rangle) = \emptyset$ by assumption and $w \in \mathit{cover}(w\langle W\rangle)$ by definition, we must have $(P \setminus w) \subseteq \mathit{cover}(v\langle V\rangle)$ but $w \notin \mathit{cover}(v\langle V\rangle)$. Applying Proposition A.8 we see that $w \in V$, so $v\langle V\rangle$ and $w\langle W\rangle$ are mergeable. By assumption, $u\langle U\rangle$, $v\langle V\rangle$, and $w\langle W\rangle$ are all simple subconfigurations, and using Proposition A.8 again as well as the $\Rightarrow$-direction of this proposition it can be verified that the equality

$$\begin{aligned}(\text{A.2})\quad \mathit{cover}(u\langle U\rangle) &= T^u \setminus \bigcup_{x\in U} T^x = \mathit{cover}(v\langle V\rangle) \mathbin{\dot\cup} \mathit{cover}(w\langle W\rangle)\\ &= \mathit{cover}(\mathsf{merge}(v\langle V\rangle, w\langle W\rangle)) = T^v \setminus \bigcup_{x\in(V\cup W)\setminus w} T^x\end{aligned}$$

can hold only if $U = (V \cup W) \setminus w$, i.e., only if $u\langle U\rangle = \mathsf{merge}(v\langle V\rangle, w\langle W\rangle)$. ☐

Observe that we need the simplicity of the subconfigurations in order for Proposition A.10 to hold. If $v\langle V\rangle$ were not simple, white pebbles in $\bigcup_{x\in V} T^x_*$ would create problems. In a sense, requiring that subconfigurations be simple is a way of ensuring that mergers behave in the way one would expect them to.

Now the subconfigurations $v\langle \mathit{pred}(v)\rangle$ in introduction moves are obviously simple, and Observation A.9 says that mergers preserve simplicity. It is not hard to show that we can also assume that reversal moves result in simple subconfigurations, so that in any L-pebbling of a binary tree T it is always the case that all subconfigurations are simple. We next sketch a proof of this statement,[9] which will simplify matters in what follows.

LEMMA A.11. *Suppose that $\mathcal{L}$ is a complete L-pebbling of a binary tree T. Then from $\mathcal{L}$ we can construct a complete L-pebbling $\mathcal{L}'$ such that $\mathsf{cost}(\mathcal{L}') \le \mathsf{cost}(\mathcal{L})$ and $\mathcal{L}'$ contains only simple L-configurations.*

Proof sketch. Recalling Definition 6.5, let $\mathbb{L}'_t = \{v\langle \mathit{swp}(v,W)\rangle \mid v\langle W\rangle \in \mathbb{L}_t\}$ for $\mathcal{L} = \{\mathbb{L}_0, \ldots, \mathbb{L}_\tau\}$. This implies that $\mathit{cover}(\mathbb{L}'_t) = \mathit{cover}(\mathbb{L}_t)$ (perhaps most easily seen by using Proposition A.8) for $\mathbb{L}'_t$ consisting of simple subconfigurations. We claim that $\mathcal{L}' = \{\mathbb{L}'_0, \ldots, \mathbb{L}'_\tau\}$ is a legal L-pebbling if repeated L-configurations $\mathbb{L}'_t = \mathbb{L}'_{t+1}$ are eliminated. Let us outline the proof.

Introduction moves in $\mathcal{L}$ are always performed also in $\mathcal{L}'$, since $v\langle \mathit{pred}(v)\rangle = v\langle \mathit{swp}(v, \mathit{pred}(v))\rangle$.

Suppose that $v_1\langle W_1\rangle$ and $v_2\langle W_2\rangle$ are merged in $\mathcal{L}$, and let $W'_i = \mathit{swp}(v_i, W_i)$ for $i = 1, 2$. If $v_2 \notin W'_1$ we have $\mathit{swp}(v_1, (W_1 \cup W_2) \setminus v_2) = W'_1$, so nothing happens in $\mathcal{L}'$. Otherwise we can merge $v_1\langle W'_1\rangle$ and $v_2\langle W'_2\rangle$, and it is straightforward to verify that $\mathit{swp}(v_1, (W_1 \cup W_2) \setminus v_2) = \mathit{swp}(v_1, (W'_1 \cup W'_2) \setminus v_2)$.

Likewise, if $v_1\langle W_1\rangle$ is reversed to $v_2\langle W_2\rangle$ in $\mathcal{L}$, going from $v_1\langle W'_1\rangle$ to $v_2\langle W'_2\rangle$ in $\mathcal{L}'$ is a legal reversal move.

Finally, note that erasures are taken care of automatically by the definition of $\mathbb{L}'_t$. ☐

In the outline of the proof in section A.1, we said that we wanted to construct L-pebblings with "nonintersecting" subconfigurations. We next formally define two

[9]However, we note that the reader who so wishes can instead make Lemma A.11 an assumption and restrict Theorem 5.4 to the case of L-pebblings with simple subconfigurations. This is so since a careful reading of section 6 reveals that the L-pebblings that we get from resolution derivations satisfy this property. To see this, note that by Definition 6.6 all subconfigurations in $\mathbb{L}(\mathbb{C}_t)$ are simple, and by Observation A.9 and the construction in Lemma 6.13 all subconfigurations in the intermediate L-configurations are simple as well.

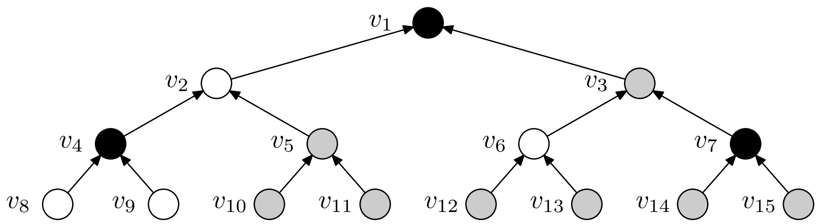


FIG. 10. *Three pebble subconfigurations $v_1\langle v_2, v_6\rangle$, $v_4\langle v_8, v_9\rangle$, and $v_7\langle\emptyset\rangle$.*

slightly different flavors of "nonintersecting" that we will use extensively below. It might be easier to parse this rather technical definition by first studying Examples A.13 and A.14.

DEFINITION A.12. *For a simple pebble subconfiguration $v\langle W\rangle$, we define the* boundary *of $v\langle W\rangle$ to be $\partial v\langle W\rangle = \{v\} \cup W$. The* interior *of $v\langle W\rangle$ is $int(v\langle W\rangle) = cover(v\langle W\rangle) \setminus \partial v\langle W\rangle$ and the* closure *is $cl(v\langle W\rangle) = cover(v\langle W\rangle) \cup \partial v\langle W\rangle$.*

If $cover(v\langle V\rangle) \cap cover(u\langle U\rangle) = \emptyset$, the subconfigurations $v\langle V\rangle$ and $u\langle U\rangle$ are said to be nonoverlapping. *If $cl(v\langle V\rangle) \cap cl(u\langle U\rangle) = \emptyset$, $v\langle V\rangle$ and $u\langle U\rangle$ are* nontouching.

Example A.13. Consider the subconfigurations in Figure 10 (which is Figure 4 but with all vertices labeled). For $v_1\langle v_2, v_6\rangle$ we have

$$\begin{aligned} cover(v_1\langle v_2, v_6\rangle) &= \{v_1, v_3, v_7, v_{14}, v_{15}\}, \\ \partial v_1\langle v_2, v_6\rangle &= \{v_1, v_2, v_6\}, \\ int(v_1\langle v_2, v_6\rangle) &= \{v_3, v_7, v_{14}, v_{15}\}, \\ cl(v_1\langle v_2, v_6\rangle) &= \{v_1, v_2, v_3, v_6, v_7, v_{14}, v_{15}\}. \end{aligned}$$

Since $cl(v_4\langle v_8, v_9\rangle) = \{v_4, v_8, v_9\}$, the subconfigurations $v_1\langle v_2, v_6\rangle$ and $v_4\langle v_8, v_9\rangle$ are nontouching. For $v_7\langle\emptyset\rangle$ we have $cover(v_7\langle\emptyset\rangle) = \{v_7, v_{14}, v_{15}\}$, so $v_7\langle\emptyset\rangle$ and $v_1\langle v_2, v_6\rangle$ are overlapping, or, more precisely, it holds that $v_7\langle\emptyset\rangle \prec v_1\langle v_2, v_6\rangle$.

Example A.14. More generally, if $v\langle V\rangle$ and $w\langle W\rangle$ are simple, mergeable subconfigurations with $w \in V$, then $v\langle V\rangle$ and $w\langle W\rangle$ are nonoverlapping (because of Proposition A.10) but touching in w. This is illustrated in Figure 5.

For the case of binary trees, it turns out that Lemma A.5 can be formulated more sharply. Remember that the cover of an L-configuration $\mathbb{L}$ is defined by taking the union of the covers of the subconfigurations in $\mathbb{L}$.

LEMMA A.15. *Suppose that $\mathcal{L} = \{\mathbb{L}_0, \ldots, \mathbb{L}_\tau\}$ is a reversal-free L-pebbling on T such that all L-configurations $\mathbb{L}_t$ are simple and $\mathbb{L}_\tau$ consists of pairwise nonoverlapping subconfigurations. Then there is a reversal-free complete L-pebbling $\mathcal{L}' = \{\mathbb{L}'_0, \ldots, \mathbb{L}'_{\tau'}\}$ with $\mathbb{L}'_0 \subseteq \mathbb{L}_0$, $\mathbb{L}'_{\tau'} = \mathbb{L}_\tau$, and $\mathsf{cost}(\mathcal{L}') \leq \mathsf{cost}(\mathcal{L})$ such that every $v\langle V\rangle$ in $\mathcal{L}'$ occurs during one contiguous time interval, and every $v\langle V\rangle$ in $\mathcal{L}'$ except those in $\mathbb{L}_\tau$ is used in exactly one merger, after which it is erased. Also, all $\mathbb{L}'_t$ are simple, and $cover(\mathbb{L}'_t)$ grows monotonically with t.*

Proof. Apply the construction in the proof of Lemma A.5, but use the stronger induction hypothesis that $\mathbb{L}'_t \subseteq \mathbb{L}_t$ for $\mathbb{L}'_t$ consisting of nonoverlapping subconfigurations.

For introduction moves, if the L-configuration $\mathbb{L}'_{t+1}$ is nonoverlapping, then so is $\mathbb{L}'_t = \mathbb{L}'_{t+1} \setminus v\langle pred(v)\rangle$.

For merger moves $u\langle U\rangle = \mathsf{merge}(v\langle V\rangle, w\langle W\rangle)$, by the induction hypothesis we have $v\langle V\rangle, w\langle W\rangle \notin \mathbb{L}'_{t+1}$, since $\mathbb{L}'_{t+1}$ is nonoverlapping and $v\langle V\rangle$ and $w\langle W\rangle$ are covered by $u\langle U\rangle$ by Proposition A.10. For the same reason $\mathbb{L}'_t$ must be nonoverlapping, since we just swap $u\langle U\rangle$ for $v\langle V\rangle$ and $w\langle W\rangle$ with $cover(v\langle V\rangle)\,\dot\cup\, cover(w\langle W\rangle) = cover(u\langle U\rangle)$. (Naturally enough, though, the intermediate L-configurations $\mathbb{L}'_{t+1/3}$ and $\mathbb{L}'_{t+2/3}$, where we merge and erase, will be overlapping.)

Also, any subconfiguration $v\langle V\rangle$ occurs only in one merger, after which it is immediately erased. For at all times $t^* > t$ after which $v\langle V\rangle$ was erased from $\mathcal{L}'$ directly after the first merger move involving $v\langle V\rangle$, there is a $u\langle U\rangle \succ v\langle V\rangle$ in $\mathbb{L}'_{t^*}$. Since all $\mathbb{L}'_{t^*}$ are nonoverlapping, the subconfiguration $v\langle V\rangle$ never appears again (this can easily be formalized by a forward induction argument).

Finally, note that in the reversal-free L-pebbling $\mathcal{L}'$, the cover increases at introduction moves, stays the same at mergers, and (by the construction for mergers) also stays the same for erasures. Hence, $cover(\mathbb{L}'_t)$ grows monotonically with t. ∎

For any L-configuration, we can find an L-configuration with the same cover but consisting only of nontouching subconfigurations. We will refer to this as a *canonical representation*.

LEMMA A.16. *Let V be any nonempty vertex set in T. Then there exists a unique simple L-configuration $\mathbb{L}'$ such that $cover(\mathbb{L}') = V$ and all subconfigurations in $\mathbb{L}'$ are simple and nontouching.*

We introduce the formal definition of canonical representation before proceeding to give a proof of Lemma A.16.

DEFINITION A.17 (canonical representation). *For an arbitrary nonempty set of vertices $V \subseteq V(T)$, we define the* canonical representation $\mathsf{canon}(V)$ *of V to be the unique $\mathbb{L}'$ in Lemma* A.16.

For $\mathbb{L}$ an arbitrary L-configuration, we define $\mathsf{canon}(\mathbb{L})$ *to be the canonical representation $\mathbb{L}' = \mathsf{canon}(cover(\mathbb{L}))$ of the vertices covered by $\mathbb{L}$.*

Once more, we note that this definition is specific for binary trees. Consider, for instance, the set $V = \{u_1, u_2, s_1, s_2, s_3\}$ in Figure 6. Both u_1 and u_2 must be black-pebbled in any $\mathbb{L}$ with $cover(\mathbb{L}) = V$, but there is no way two subconfigurations $u_1\langle U_1\rangle$ and $u_2\langle U_2\rangle$ can be nontouching.

Proof of Lemma A.16. We first show existence and then uniqueness.

We construct $\mathbb{L}'$ with $cover(\mathbb{L}') = V$ as follows: for each $v \in V$ such that $succ(v) \notin V$ or $v = z$, add the subconfiguration $v\langle W\rangle$, where $W \subseteq T^v_*$ is the maximal set such that for all $w \in W$ it holds that $P^w_* \setminus P^v_* \subseteq V$ but $w \notin V$. By construction, $v\langle W\rangle$ is simple, and applying Proposition A.8 shows that $cover(\mathbb{L}') = V$.

Clearly, every $u \in V$ is covered by exactly one subconfiguration in $\mathbb{L}'$, so all subconfigurations in $\mathbb{L}'$ must be at least nonoverlapping. Also, for all $Wh(\mathbb{L}')$ it holds that $w \notin V$ by construction, so the subconfigurations are nontouching.

To get uniqueness, suppose that $\mathbb{L}$ is any simple L-configuration with the property that $cover(\mathbb{L}) = V$. If $v \in V$ but $succ(v) \notin V$, there must be a black pebble on v in $\mathbb{L}$ by Proposition A.8. Also, if $w \notin V$ but $succ(w) \in V$, w must be white-pebbled. Thus $Bl(\mathbb{L}') \subseteq Bl(\mathbb{L})$ and $Wh(\mathbb{L}') \subseteq Wh(\mathbb{L})$.

The L-configuration $\mathbb{L}$ cannot have pebbles outside $cover(\mathbb{L}') \cup Wh(\mathbb{L}')$, for if so we would have $cover(\mathbb{L}) \supsetneqq V$ (by the convexity property in Definition A.7 of subconfigurations and since all subconfigurations are simple). And if $\mathbb{L}$ has pebbles inside $cover(\mathbb{L}') \setminus \big(Bl(\mathbb{L}') \cup Wh(\mathbb{L}')\big)$, there must exist touching subconfigurations in $\mathbb{L}$. Hence, if $\mathbb{L}'$ does not contain touching subconfigurations it holds that $\mathbb{L}' = \mathbb{L}$. ∎

Note, in particular, that if V is a convex set in T in the sense of Definition A.7, then $\mathsf{canon}(V)$ is a single subconfiguration.

We use the canonical representation to extend Definition A.12 to L-configurations.

DEFINITION A.18. *Suppose that $\mathbb{L}, \mathbb{L}_1, \mathbb{L}_2$ are simple L-configurations.*

If $\mathit{cover}(\mathbb{L}_1) = \mathit{cover}(\mathbb{L}_2)$, *we say that* $\mathbb{L}_1$ *and* $\mathbb{L}_2$ coincide *and write* $\mathbb{L}_1 \sim \mathbb{L}_2$. $\mathbb{L}$ *is* nonoverlapping *if all distinct* $v\langle V\rangle, u\langle U\rangle \in \mathbb{L}$ *are pairwise nonoverlapping and* nontouching *if all distinct* $v\langle V\rangle, u\langle U\rangle \in \mathbb{L}$ *are pairwise nontouching.* $\mathbb{L}_1$ *and* $\mathbb{L}_2$ *are* mutually nonoverlapping *or* mutually nontouching *if all* $v\langle V\rangle \in \mathbb{L}_1$ *and* $u\langle U\rangle \in \mathbb{L}_2$ *are pairwise nonoverlapping or nontouching, respectively.*

Let $\mathbb{L}' = \mathsf{canon}(\mathbb{L})$ *be the canonical representation of* $\mathbb{L}$. *Then the* boundary *of* $\mathbb{L}$ *is defined to be* $\partial\mathbb{L} = \bigcup_{v\langle V\rangle\in\mathbb{L}'} \partial v\langle V\rangle$, *the* interior *is defined to be* $\mathit{int}(\mathbb{L}) = \bigcup_{v\langle V\rangle\in\mathbb{L}'} \mathit{int}(v\langle V\rangle)$, *and the* closure *is* $\mathit{cl}(\mathbb{L}) = \bigcup_{v\langle V\rangle\in\mathbb{L}'} \mathit{cl}(v\langle V\rangle)$.

Observe that $\mathbb{L}_1 = \mathbb{L}_2$ implies $\mathbb{L}_1 \sim \mathbb{L}_2$, but not the other way round. For nontouching L-configurations, however, the two notions are identical. Also, $\mathbb{L} \sim \mathsf{canon}(\mathbb{L})$ by definition.

Example A.19. Returning to Figure 10, if we look at the L-configuration $\mathbb{L} = \bigl\{v_1\langle v_2, v_6\rangle, v_4\langle v_8, v_9\rangle, v_7\langle\emptyset\rangle\bigr\}$ we have $\mathit{cover}(\mathbb{L}) = \{v_1, v_3, v_4, v_7, v_{14}, v_{15}\}$. Since $v_7\langle\emptyset\rangle$ is covered by $v_1\langle v_2, v_6\rangle$ and the subconfigurations $v_1\langle v_2, v_6\rangle$ and $v_4\langle v_8, v_9\rangle$ are nontouching, we get the canonical representation simply by leaving out $v_7\langle\emptyset\rangle$, i.e., $\mathsf{canon}(\mathbb{L}) = \bigl\{v_1\langle v_2, v_6\rangle, v_4\langle v_8, v_9\rangle\bigr\}$. Using this canonical representation of $\mathbb{L}$, we see that

$$
\begin{aligned}
\partial\mathbb{L} &= \{v_1, v_2, v_4, v_6, v_8, v_9\},\\
\mathit{int}(\mathbb{L}) &= \{v_3, v_7, v_{14}, v_{15}\},\\
\mathit{cl}(\mathbb{L}) &= \{v_1, v_2, v_3, v_4, v_6, v_7, v_8, v_9, v_{14}, v_{15}\}.
\end{aligned}
$$

The L-configuration $\mathbb{L}$ is overlapping because of $v_7\langle\emptyset\rangle$ and $v_1\langle v_2, v_6\rangle$, but, for instance, $\mathbb{L}_1 = \bigl\{v_1\langle v_2, v_6\rangle, v_7\langle\emptyset\rangle\bigr\}$ and $\mathbb{L}_2 = \bigl\{v_4\langle v_8, v_9\rangle\bigr\}$ are mutually nontouching.

As a final preliminary before moving on to part 1 in the proof outline in section A.1, we collect some properties of the L-pebbling cost function of Definition 5.2.

Proposition A.20. *Suppose that* $\mathbb{L}, \mathbb{L}_1, \ldots, \mathbb{L}_m$ *are arbitrary simple L-configurations.*

1. *If* $\mathbb{L}_1 \subseteq \mathbb{L}_2$ *then* $\mathit{cost}(\mathbb{L}_1) \leq \mathit{cost}(\mathbb{L}_2)$.
2. $\mathit{cost}(\mathbb{L}_1 \cup \mathbb{L}_2) \leq \mathit{cost}(\mathbb{L}_1) + \mathit{cost}(\mathbb{L}_2)$.
3. *If* $\mathbb{L}$ *is nontouching,* $\mathit{cost}(\mathbb{L}) = \bigl|\mathit{Bl}(\mathbb{L})\bigr| + \bigl|\mathit{Wh}(\mathbb{L})\bigr| = \bigl|\partial\mathbb{L}\bigr|$.
4. *If* $\mathbb{L}_i$ *and* $\mathbb{L}_j$ *are mutually nontouching for* $1 \leq i < j \leq m$, *it holds that* $\mathit{cost}\bigl(\bigcup_{i=1}^m \mathbb{L}_i\bigr) = \sum_{i=1}^m \mathit{cost}(\mathbb{L}_i)$.
5. *If* $\mathbb{L}'_i = \mathsf{canon}(\mathbb{L}_i)$ *for* $i = 1, \ldots, m$, *then* $\mathit{cost}\bigl(\bigcup_{i=1}^m \mathbb{L}'_i\bigr) \leq \mathit{cost}\bigl(\bigcup_{i=1}^m \mathbb{L}_i\bigr)$.
6. *If* $\mathbb{L}' = \mathsf{canon}(\mathbb{L})$, *then* $\mathit{cost}(\mathbb{L} \cup \mathbb{L}') = \mathit{cost}(\mathbb{L})$, *and there is an L-pebbling from* $\mathbb{L}$ *to* $\mathbb{L}'$ *which does not cost more than* $\mathbb{L}$.

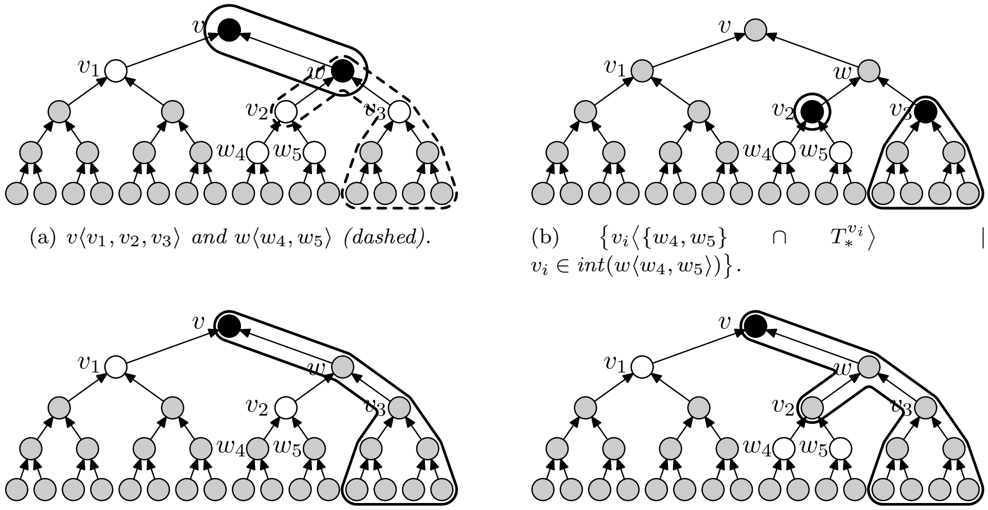
Proof. Parts 1 and 2 are from Proposition 6.14 on page 83 and were proven there.

For part 3, using Definition A.18 we see that if $\mathbb{L}$ is nontouching, it holds that $\mathit{Bl}(\mathbb{L}) \cap \mathit{Wh}(\mathbb{L}) = \emptyset$. And if the L-configurations $\mathbb{L}_i$ and $\mathbb{L}_j$ are mutually nontouching, we have $\bigl(\mathit{Bl}(\mathbb{L}_i) \cup \mathit{Wh}(\mathbb{L}_i)\bigr) \cap \bigl(\mathit{Bl}(\mathbb{L}_j) \cup \mathit{Wh}(\mathbb{L}_j)\bigr) = \emptyset$, which shows that each pebbled vertex on the left-hand side in part 4 is counted exactly once on the right-hand side.

Part 5 is again immediate since $\mathit{Bl}(\mathbb{L}'_i) \subseteq \mathit{Bl}(\mathbb{L}_i)$ and $\mathit{Wh}(\mathbb{L}'_i) \subseteq \mathit{Wh}(\mathbb{L}_i)$ for $\mathbb{L}'_i = \mathsf{canon}(\mathbb{L}_i)$ by Proposition A.8 and the proof of Lemma A.16.

For part 6, $\mathit{Bl}(\mathbb{L} \cup \mathbb{L}') = \mathit{Bl}(\mathbb{L})$ and $\mathit{Wh}(\mathbb{L} \cup \mathbb{L}') = \mathit{Wh}(\mathbb{L})$, which shows that the cost is the same. We also claim that we can do an L-pebbling from $\mathbb{L}$ to $\mathbb{L}' = \mathsf{canon}(\mathbb{L})$ at no extra cost.

To show this claim, we first note that if $v\langle V\rangle$ and $w\langle W\rangle$ are touching but nonoverlapping, we can derive a subconfiguration $u\langle U\rangle$ such that $\mathit{cover}(u\langle U\rangle) = \mathit{cover}(v\langle V\rangle) \cup \mathit{cover}(w\langle W\rangle)$ simply by merging $v\langle V\rangle$ and $w\langle W\rangle$, because either $w \in V$ or $v \in W$. Suppose therefore that $v\langle V\rangle$ and $w\langle W\rangle$ are overlapping and that $w \in T^v$



(a) $v\langle v_1, v_2, v_3\rangle$ *and* $w\langle w_4, w_5\rangle$ *(dashed).*

(b) $\{v_i\langle\{w_4, w_5\} \cap T_*^{v_i}\rangle \mid v_i \in int(w\langle w_4, w_5\rangle)\}$.

(c) $\mathsf{merge}\big(v\langle v_1, v_2, v_3\rangle, v_3\langle\{w_4, w_5\} \cap T_*^{v_3}\rangle\big)$.

(d) $\mathsf{canon}\big(\{v\langle v_1, v_2, v_3\rangle, w\langle w_4, w_5\rangle\}\big)$.

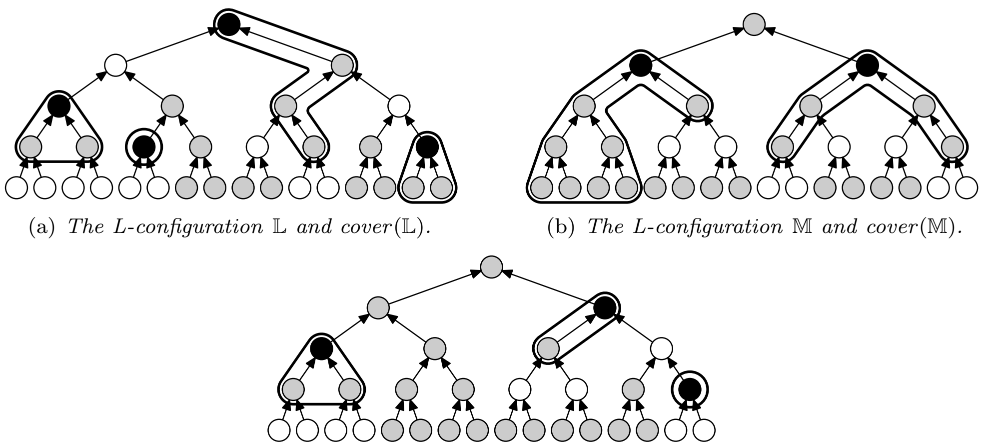
FIG. 11. *Illustration of pebbling in Proposition* A.20*, part* 6*, with covered vertices indicated.*

but $w\langle W\rangle \not\preceq v\langle V\rangle$. Then we can derive a subconfiguration $u\langle U\rangle$ with $cover(u\langle U\rangle) = cover(v\langle V\rangle) \cup cover(w\langle W\rangle)$ and substitute it for $v\langle V\rangle$ and $w\langle W\rangle$ at no extra cost by first deriving $v_i\langle W \cap T_*^{v_i}\rangle$ for all $v_i \in V \cap int(w\langle W\rangle)$ from $w\langle W\rangle$ by reversals, and then merging all $v_i\langle W \cap T_*^{v_i}\rangle$ in turn with $v\langle V\rangle$. The resulting L-configuration $\mathbb{L} \cup \big\{v_i\langle W \cap T_*^{v_i}\rangle \mid v_i \in V \cap int(w\langle W\rangle)\big\} \cup u\langle U\rangle$ costs no more than $\mathbb{L}$, since the only change is that already white-pebbled vertices are also black-pebbled. Finally, erase $v\langle V\rangle$, $w\langle W\rangle$ and all $v_i\langle W \cap T_*^{v_i}\rangle$. Repeating this for all mutually touching subconfigurations, the claim follows by induction. ∎

A "proof-by-example" pebbling move sequence for part 6 as described above is given in Figure 11, with the overlapping subconfigurations $v\langle V\rangle$ and $w\langle W\rangle$ in Figure 11(a), the two subconfigurations in $\big\{v_i\langle W \cap T_*^{v_i}\rangle \mid v_i \in V \cap int(w\langle W\rangle)\big\}$ derived by reversals from $w\langle W\rangle$ in Figure 11(b), and the two mergers of $v\langle V\rangle$ with these subconfigurations in Figures 11(c) and 11(d) leading to the canonical representation $\mathsf{canon}\big(\{v\langle V\rangle, w\langle W\rangle\}\big)$.

A.4. Nonoverlapping labeled pebblings and projections. In this subsection we turn to part 1 in the outline of the proof of Lemma A.2 in section A.1. From now on we will assume without loss of generality (in view of Lemma A.11) that all L-pebblings operate with simple subconfigurations only (Definition 6.4) and that they are nonredundant in the sense of Lemma A.5.

Parts 5 and 6 of Proposition A.20 tell us that for any given set of vertices, the cheapest way of covering these vertices is to use canonical L-configurations, and that if $\mathbb{L}$ is not canonical, it does not cost anything extra to make $\mathbb{L}$ canonical by applying reversals and mergers followed by erasures. We define *nonoverlapping pebblings* as L-pebblings which always keep the L-configurations canonical in this way. In a nonoverlapping pebbling, each introduction is immediately followed by a merger when possible, each merger is immediately followed by erasures of the merged subconfigurations, and all reversals from a subconfiguration $u\langle U\rangle$ are performed in sequence after which $u\langle U\rangle$ is erased. We refer to these merger-and-erasures and reversals-and-erasure



(a) *The L-configuration* $\mathbb{L}$ *and cover*$(\mathbb{L})$.

(b) *The L-configuration* $\mathbb{M}$ *and cover*$(\mathbb{M})$.

(c) *The projection* $\mathsf{proj}_{\mathbb{M}}(\mathbb{L})$ *with cover.*

FIG. 12. *Example L-configurations* $\mathbb{L}$ *and* $\mathbb{M}$ *and projected L-configuration* $\mathsf{proj}_{\mathbb{M}}(\mathbb{L})$.

moves as *expansions* and *implosions*, respectively.

DEFINITION A.21 (nonoverlapping pebbling). *A* nonoverlapping L-pebbling $\mathcal{L}$ *is a sequence of the following types of moves.*

Introduction. $\mathbb{L}_{t+1} = \mathbb{L}_t \cup v\langle pred(v)\rangle$ *for* $v\langle pred(v)\rangle \not\preceq \mathbb{L}_t$ *and* $\mathbb{L}_t$ *nontouching.*

Expansion. $\mathbb{L}_{t+3} = \big(\mathbb{L}_t \cup \mathsf{merge}(u\langle U\rangle, v\langle V\rangle)\big) \setminus \{u\langle U\rangle, v\langle V\rangle\}$ *for* $u\langle U\rangle, v\langle V\rangle \in \mathbb{L}_t$ *and* $\mathbb{L}_t$ *nonoverlapping.*

Implosion. $\mathbb{L}_{t+m+1} = \big(\mathbb{L}_t \setminus u\langle U\rangle\big) \cup \mathbb{M}$ *for* $\mathbb{L}_t$ *and* $\mathbb{M} = \big\{v_i\langle V_i\rangle \mid i \in [m]\big\}$ *nontouching, and* $\mathbb{M} \preceq u\langle U\rangle \in \mathbb{L}_t$.

For technical reasons, it will be convenient to allow trivial implosion moves where $\mathbb{M} = u\langle U\rangle$. We say that $u\langle U\rangle \rightsquigarrow \mathbb{M}$ is a *nontrivial implosion* if $\mathbb{M} \prec u\langle U\rangle$. Observe that after introduction and expansion the resulting L-configuration is nonoverlapping, and after implosion the L-configuration is nontouching.

We want to prove that without loss of generality we can assume L-pebblings to be nonoverlapping. The notation in the proof of this fact is simplified by introducing *projections*.

DEFINITION A.22 (projection). *Let* $u\langle U\rangle, v\langle V\rangle$ *be arbitrary subconfigurations,* $\mathbb{L}$ *an arbitrary L-configuration, and* $\mathbb{M}$ *an arbitrary nontouching L-configuration.*

If $u\langle U\rangle$ *and* $v\langle V\rangle$ *are overlapping, the* projection *of* $u\langle U\rangle$ *on* $v\langle V\rangle$ *is defined as* $\mathsf{proj}_{v\langle V\rangle}(u\langle U\rangle) = \mathsf{canon}(cover(u\langle U\rangle) \cap cover(v\langle V\rangle))$, *i.e., the unique subconfiguration* $w\langle W\rangle$ *such that* $cover(w\langle W\rangle) = cover(u\langle U\rangle) \cap cover(v\langle V\rangle)$. *If* $u\langle U\rangle$ *and* $v\langle V\rangle$ *are nonoverlapping, we define* $\mathsf{proj}_{v\langle V\rangle}(u\langle U\rangle) = \emptyset$.

The projection of $u\langle U\rangle$ *on* $\mathbb{M}$ *is* $\mathsf{proj}_{\mathbb{M}}(u\langle U\rangle) = \bigcup_{v\langle V\rangle\in\mathbb{M}} \mathsf{proj}_{v\langle V\rangle}(u\langle U\rangle)$, *and* $\mathsf{proj}_{\mathbb{M}}(\mathbb{L}) = \bigcup_{u\langle U\rangle\in\mathbb{L}} \mathsf{proj}_{\mathbb{M}}(u\langle U\rangle)$.

In order to grasp this definition, it might be helpful to study the example in Figure 12. Note, in particular, that if $u\langle U\rangle \preceq v\langle V\rangle$, then $\mathsf{proj}_{v\langle V\rangle}(u\langle U\rangle) = u\langle U\rangle$. Here and in the following, we adopt the convention that projections resulting in the undefined subconfiguration $\emptyset$ are implicitly eliminated from all L-configurations.

We will need a technical lemma relating the pebbles in an L-configuration with those in its projection. Once deciphered, the statements in the lemma are fairly obvious, and the proof is just an exercise in applying the definitions so far in this appendix. We recommend that the reader look at the projections in (the right subtree of) the tree in Figure 12 and verify what the lemma says for this example.

LEMMA A.23. *Let $\mathbb{L}$ be any L-configuration and $\mathbb{M}$ any nontouching L-configuration, and let $\mathbb{L}_p = \mathsf{proj}_{\mathbb{M}}(\mathbb{L})$ be the projection of $\mathbb{L}$ on $\mathbb{M}$. Suppose that v is a vertex that is pebbled in $\mathbb{L}_p$ but not in $\mathbb{L}$, i.e., $v \in \big(Bl(\mathbb{L}_p) \cup Wh(\mathbb{L}_p)\big) \setminus \big(Bl(\mathbb{L}) \cup Wh(\mathbb{L})\big)$. Then the following hold:*

1. *The vertex v is on the boundary of $\mathbb{M}$, i.e., $v \in \partial\mathbb{M}$.*
2. *The pebble on the vertex v has the same color in $\mathbb{L}_p$ and $\mathbb{M}$, i.e., either $v \in Bl(\mathbb{L}_p) \cap Bl(\mathbb{M})$ or $v \in Wh(\mathbb{L}_p) \cap Wh(\mathbb{M})$.*
3. *There is a subconfiguration $w_L\big\langle W_L\big\rangle \in \mathbb{L}$ such that $v \in int\big(w_L\big\langle W_L\big\rangle\big)$.*

Proof. If $v \in Bl(\mathbb{L}_p) \cup Wh(\mathbb{L}_p)$, by Definition A.22 there are $w_L\langle W_L\rangle \in \mathbb{L}$ and $w_M\langle W_M\rangle \in \mathbb{M}$ with $\mathsf{proj}_{w_M\langle W_M\rangle}(w_L\langle W_L\rangle) = u\langle U\rangle$ such that $v \in \{u\} \cup U$. We remark that since $\mathbb{M}$ is nontouching, $\mathsf{canon}(\mathbb{M}) = \mathbb{M}$ and, in particular, $\partial\mathbb{M} = Bl(\mathbb{M}) \cup Wh(\mathbb{M})$. We make a case analysis depending on the color of the pebble on v.

1. Suppose $v = u$, i.e., that v is black-pebbled in $\mathbb{L}_p$. Then

$$v \in cover(u\langle U\rangle) = cover(w_L\langle W_L\rangle) \cap cover(w_M\langle W_M\rangle) \tag{A.3}$$

and

$$succ(v) \notin cover(u\langle U\rangle) = cover(w_L\langle W_L\rangle) \cap cover(w_M\langle W_M\rangle) \tag{A.4}$$

by the proof of Lemma A.16. But $succ(v) \in cover(w_L\langle W_L\rangle)$, since otherwise $v = w_L \in Bl(\mathbb{L})$ by Proposition A.8, which is contrary to assumption. Thus for (A.4) to hold we must have $succ(v) \notin cover(w_M\langle W_M\rangle)$, so $v = w_M \in Bl(\mathbb{M}) \subseteq \partial\mathbb{M}$. Since v is not pebbled in $\mathbb{L}_p$, in particular we have $v \notin \{w_L\} \cup W_L = \partial w_L\langle W_L\rangle$, and combining this with (A.3) we see that $v \in cover(w_L\langle W_L\rangle) \setminus \partial w_L\langle W_L\rangle = int(w_L\langle W_L\rangle)$.

2. Suppose that $v \in U$, i.e., that v is white-pebbled in $\mathbb{L}_p$. Then

$$v \notin cover(w_L\langle W_L\rangle) \cap cover(w_M\langle W_M\rangle) \tag{A.5}$$

and

$$succ(v) \in cover(w_L\langle W_L\rangle) \cap cover(w_M\langle W_M\rangle) \tag{A.6}$$

by wholly analogous reasoning. We have that $v \in cover(w_L\langle W_L\rangle)$ since otherwise $v \in Wh(\mathbb{L})$ contrary to assumption, so it must hold that $v \notin cover(w_M\langle W_M\rangle)$. Hence, $v \in Wh(\mathbb{M}) \subseteq \partial\mathbb{M}$ and $v \in cover(w_L\langle W_L\rangle) \setminus \partial w_L\langle W_L\rangle = int(w_L\langle W_L\rangle)$.

This proves the lemma. ∎

The next proposition says that any L-configuration $\mathbb{L}$ can be written as a disjoint union of the sets of subconfigurations of $\mathbb{L}$ covered by each subconfiguration in $\mathsf{canon}(\mathbb{L})$, and that the cost of $\mathbb{L}$ is the sum of the costs of the sub-L-configurations in this disjoint union. This statement, too, is obvious once deciphered, and the proof is immediate from Definition A.22, (the proof of) Lemma A.16, and Proposition A.20, parts 4 and 5.

PROPOSITION A.24. *Let $\mathbb{L}' = \mathsf{canon}(\mathbb{L})$. Then it holds that $\mathbb{L}$ is a disjoint union of the sets $\mathsf{proj}_{v\langle V\rangle}(\mathbb{L}) = \big\{u\langle U\rangle \mid v\langle V\rangle \succeq u\langle U\rangle \in \mathbb{L}\big\}$ for all $v\langle V\rangle \in \mathbb{L}'$. Also, $\mathit{cost}(\mathbb{L}) = \sum_{v\langle V\rangle \in \mathbb{L}'} \mathit{cost}(\mathsf{proj}_{v\langle V\rangle}(\mathbb{L}))$, and for all $v\langle V\rangle \in \mathbb{L}'$ it holds that $\mathit{cost}(v\langle V\rangle) \leq \mathit{cost}(\mathsf{proj}_{v\langle V\rangle}(\mathbb{L}))$.*

Example A.25. As was noted in Example A.19, for the L-configuration $\mathbb{L} = \{v_1\langle v_2, v_6\rangle, v_4\langle v_8, v_9\rangle, v_7\langle\emptyset\rangle\}$ in Figure 10 we have $\mathsf{canon}(\mathbb{L}) = \{v_1\langle v_2, v_6\rangle, v_4\langle v_8, v_9\rangle\}$. Trivially, $\mathbb{L}$ can be written as the disjoint union of

$$\mathbb{L}_1 = \mathsf{proj}_{v_1\langle v_2,v_6\rangle}(\mathbb{L}) = \{v_1\langle v_2, v_6\rangle,\ v_7\langle\emptyset\rangle\} \tag{A.7}$$

and

$$\mathbb{L}_2 = \mathsf{proj}_{v_4\langle v_8,v_9\rangle}(\mathbb{L}) = \{v_4\langle v_8, v_9\rangle\}, \tag{A.8}$$

and it holds that $\mathit{cost}(\mathbb{L}) = \mathit{cost}(\mathbb{L}_1) + \mathit{cost}(\mathbb{L}_2)$.

Using Definition A.22 and Propositions A.20 and A.24, we can prove that for every overlapping L-pebbling we can find a nonoverlapping pebbling which is at least as good and at least as cheap.

LEMMA A.26. *Suppose that $\mathcal{L}$ is an arbitrary complete L-pebbling of T. Then from $\mathcal{L}$ we can construct a nonoverlapping complete L-pebbling $\mathcal{L}'$ of T such that* $\mathit{cost}(\mathcal{L}') \leq \mathit{cost}(\mathcal{L})$.

Proof. Given $\mathcal{L} = \{\mathbb{L}_0, \ldots, \mathbb{L}_\tau\}$, we create the "backbone" $\mathcal{L}' = \{\mathbb{L}'_0, \ldots, \mathbb{L}'_\tau\}$ of a nonoverlapping pebbling by setting $\mathbb{L}'_t = \mathsf{canon}(\mathbb{L}_t)$. Then we have $\mathbb{L}'_0 = \mathbb{L}_0 = \emptyset$ and $\mathbb{L}'_\tau = \mathsf{canon}(\mathbb{L}_\tau) = \mathsf{canon}(z\langle\emptyset\rangle) = z\langle\emptyset\rangle$.

By Proposition A.20, part 5, $\mathit{cost}(\mathbb{L}'_t) \leq \mathit{cost}(\mathbb{L}_t)$, so we are done if we can fill in the holes in the transitions $\mathbb{L}'_t \rightsquigarrow \mathbb{L}'_{t+1}$ using the nonoverlapping moves of Definition A.21 without paying more than $\max\{\mathit{cost}(\mathbb{L}_t), \mathit{cost}(\mathbb{L}_{t+1})\}$. This is basically just an exercise in applying Proposition A.20. Consider the moves $\mathbb{L}_t \rightsquigarrow \mathbb{L}_{t+1}$ in $\mathcal{L}$.

Introduction. $\mathbb{L}_{t+1} = \mathbb{L}_t \cup v\langle \mathit{pred}(v)\rangle$: If $v\langle\mathit{pred}(v)\rangle \preceq \mathbb{L}'_t$, set $\mathbb{L}'_{t+1} = \mathbb{L}'_t$. Otherwise, introduce $v\langle\mathit{pred}(v)\rangle$ and canonize by expanding (at most three times) to get $\mathbb{L}'_{t+1} = \mathsf{canon}(\mathbb{L}_{t+1})$. This can be done at cost at most $\mathit{cost}(\mathbb{L}_{t+1})$, since $\mathit{cost}(\mathbb{L}'_t \cup v\langle\mathit{pred}(v)\rangle) \leq \mathit{cost}(\mathbb{L}_{t+1})$ by part 5 of Proposition A.20 (note that $\mathsf{canon}(v\langle\mathit{pred}(v)\rangle) = v\langle\mathit{pred}(v)\rangle$), and since the canonization does not increase this cost by part 6 of Proposition A.20.

Merger. $\mathbb{L}_{t+1} = \mathbb{L}_t \cup \mathsf{merge}(u\langle U\rangle, v\langle V\rangle)$ for $u\langle U\rangle, v\langle V\rangle \in \mathbb{L}_t$: For merger moves it holds that $\mathbb{L}_{t+1} \sim \mathbb{L}_t$, so set $\mathbb{L}'_{t+1} = \mathbb{L}'_t = \mathsf{canon}(\mathbb{L}_{t+1})$.

Reversal. $\mathbb{L}_{t+1} = \mathbb{L}_t \cup \{v\langle V\rangle\}$ for $v\langle V\rangle \prec u\langle U\rangle \in \mathbb{L}_t$: For reversal moves it holds that $\mathbb{L}_{t+1} \sim \mathbb{L}_t$, so set $\mathbb{L}'_{t+1} = \mathbb{L}'_t = \mathsf{canon}(\mathbb{L}_{t+1})$.

Erasure. $\mathbb{L}_{t+1} = \mathbb{L}_t \setminus v\langle V\rangle$ for $v\langle V\rangle \in \mathbb{L}_t$: If $v\langle V\rangle \preceq \mathbb{L}_{t+1}$, we have $\mathbb{L}_{t+1} \sim \mathbb{L}_t$ and can set $\mathbb{L}'_{t+1} = \mathbb{L}'_t$, so assume that $v\langle V\rangle \not\preceq \mathbb{L}_{t+1}$.

Since $\mathbb{L}'_t \sim \mathbb{L}_t$ is nontouching, there is a $u\langle U\rangle \in \mathbb{L}'_t$ such that $v\langle V\rangle \preceq u\langle U\rangle$. It follows from Proposition A.24 that for $w\langle W\rangle \in \mathbb{L}'_t$, $w\langle W\rangle \neq u\langle U\rangle$, we have $\mathsf{proj}_{w\langle W\rangle}(\mathbb{L}_{t+1}) = \mathsf{proj}_{w\langle W\rangle}(\mathbb{L}_t)$. Thus, letting $\mathbb{L}^u_i = \mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_i)$ for $i = t, t+1$, by Proposition A.20, part 4, it is sufficient to show locally that we can implode $u\langle U\rangle = \mathsf{canon}(\mathbb{L}^u_t) = \mathsf{canon}(\mathbb{L}^u_{t+1} \cup v\langle V\rangle)$ into $\mathbb{M} = \mathsf{canon}(\mathbb{L}^u_{t+1})$ at cost at most $\max\{\mathit{cost}(\mathbb{L}^u_{t+1} \cup v\langle V\rangle), \mathit{cost}(\mathbb{L}^u_{t+1})\} = \mathit{cost}(\mathbb{L}^u_{t+1} \cup v\langle V\rangle)$. By part 1 of Proposition A.20, it is enough to check that the inequality $\mathit{cost}(\mathbb{M} \cup u\langle U\rangle) \leq \mathit{cost}(\mathbb{L}^u_{t+1} \cup v\langle V\rangle)$ holds. But this follows from part 5 of the same proposition by setting $\mathbb{L}_1 = \mathbb{L}^u_{t+1} \cup v\langle V\rangle$ with $\mathbb{L}'_1 = \mathsf{canon}(\mathbb{L}_1) = u\langle U\rangle$ and $\mathbb{L}_2 = \mathbb{L}^u_{t+1}$ with $\mathbb{L}'_2 = \mathsf{canon}(\mathbb{L}_2) = \mathbb{M}$.

Eliminating "idle moves" $\mathbb{L}'_{t+1} = \mathbb{L}'_t$, we see that we get a nonoverlapping pebbling in accordance with Definition A.21. ∎

Lemma A.26 tells us that as far as pebbling cost is concerned, without loss of generality we may assume that an L-pebbling $\mathcal{L}$ that reaches the subconfiguration

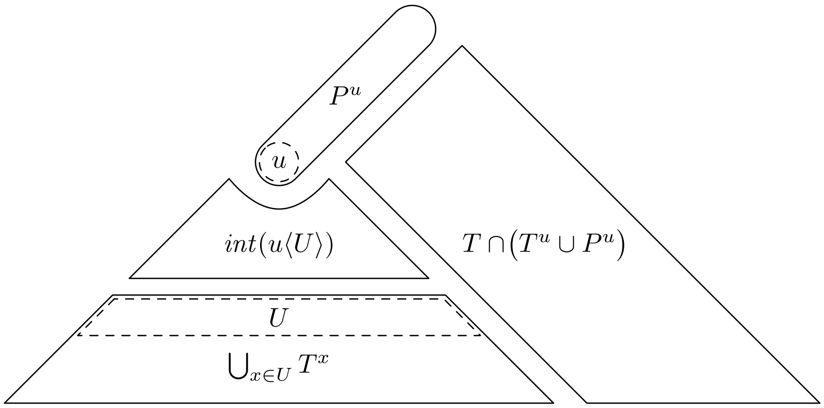


FIG. 13. *Illustration of cases for w with respect to $u\langle U\rangle$ in Proposition* A.27.

$z\langle\emptyset\rangle$ is nonoverlapping. This completes part 1 in the proof of Lemma A.2 sketched in section A.1.

In what follows, it will sometimes be convenient to consider the L-pebblings as consisting of the "aggregated" expansion and implosion moves in Definition A.21, and sometimes more convenient to consider each individual merger or reversal in these moves individually as in Definition 5.2. In view of Lemma A.26, we know that we can switch freely back and forth between these two perspectives.

A.5. Projections preserve labeled pebblings. If $\mathcal{L} = \{\mathbb{L}_0, \ldots, \mathbb{L}_\tau\}$ is a nonoverlapping pebbling ending in an implosion $u\langle U\rangle \rightsquigarrow \mathbb{M}$, it seems natural to try to replace the moves in $\mathcal{L}$ leading to $u\langle U\rangle$ by a reversal-free pebbling reaching $\mathbb{M} \preceq u\langle U\rangle$. Since $u\langle U\rangle$ and $\mathbb{L}_{\tau-1} \setminus u\langle U\rangle$ are mutually nontouching by definition, this substitution should not affect the cost of the pebbling outside $cl(u\langle U\rangle)$ by Proposition A.24.

We argue that intuitively, one natural candidate for such a substitution pebbling is what we get if we take all L-configurations in $\mathcal{L}$ and project them on $\mathbb{L}_\tau = (\mathbb{L}_{\tau-(m+1)} \cup \mathbb{M}) \setminus u\langle U\rangle$. To show that this idea makes sense, we establish as a first step that projections preserve merger moves.

PROPOSITION A.27. *Suppose that $\mathbb{M}$ is a nontouching L-configuration and that $v\langle V\rangle$ and $w\langle W\rangle$ are mergeable with $w \in V$. Then if $\mathsf{proj}_{\mathbb{M}}(\mathsf{merge}(v\langle V\rangle, w\langle W\rangle)) \neq \mathsf{proj}_{\mathbb{M}}(\{v\langle V\rangle, w\langle W\rangle\})$, it holds that $\mathsf{proj}_{\mathbb{M}}(\mathsf{merge}(v\langle V\rangle, w\langle W\rangle))$ can be derived from $\mathsf{proj}_{\mathbb{M}}(\{v\langle V\rangle, w\langle W\rangle\})$ by a single merger on w.*

Proof. Consider the merger vertex w. For each $u\langle U\rangle \in \mathbb{M}$ there are four possibilities for w:

1. $w \in \bigcup_{x\in U} T^x$,
2. $w \in P^u$,
3. $w \in T \setminus (T^u \cup P^u)$, and
4. $w \in int(u\langle U\rangle)$.

See Figure 13 for a schematic illustration.

For all $u\langle U\rangle \in \mathbb{M}$ such that $w \notin int(u\langle U\rangle)$, i.e., the first three cases, it is straightforward, if tedious, to verify that $\mathsf{merge}(v\langle V\rangle, w\langle W\rangle)$ projects the same subconfigurations on $u\langle U\rangle$ as do $v\langle V\rangle$ and $w\langle W\rangle$ together. In the fourth case, the change in projection corresponds to exactly one merger move, and since $\mathbb{M}$ is nontouching, there is at most one $u\langle U\rangle \in \mathbb{M}$ for which this case applies.

We prove these statements by analyzing the cases above one by one, using in the analysis that $\mathit{cover}(\mathsf{merge}(v\langle V\rangle, w\langle W\rangle)) = \mathit{cover}(v\langle V\rangle) \mathbin{\dot\cup} \mathit{cover}(w\langle W\rangle)$ (Proposition A.10).

1. $w \in \bigcup_{x\in U} T^x$: By Proposition A.8, $\mathit{cover}(w\langle W\rangle) \subseteq T^w$, and since by assumption it holds that $T^w \subseteq \bigcup_{x\in U} T^x$, it follows that

$$\mathit{cover}(w\langle W\rangle) \cap \mathit{cover}(u\langle U\rangle) \subseteq T^w \cap \bigl(T^u \setminus \textstyle\bigcup_{x\in U} T^x\bigr) = \emptyset \tag{A.9}$$

and hence

$$\begin{aligned}\mathit{cover}&(\mathsf{merge}(v\langle V\rangle, w\langle W\rangle)) \cap \mathit{cover}(u\langle U\rangle)\\ &= \bigl(\mathit{cover}(v\langle V\rangle) \mathbin{\dot\cup} \mathit{cover}(w\langle W\rangle)\bigr) \cap \mathit{cover}(u\langle U\rangle)\\ &= \mathit{cover}(v\langle V\rangle) \cap \mathit{cover}(u\langle U\rangle).\end{aligned} \tag{A.10}$$

Consequently, $\mathsf{proj}_{u\langle U\rangle}(\mathsf{merge}(v\langle V\rangle, w\langle W\rangle)) = \mathsf{proj}_{u\langle U\rangle}(v\langle V\rangle)$ according to Definition A.22, so the merger does not change the projection.

2. $w \in P^u$: Then $\mathit{cover}(u\langle U\rangle) \subseteq T^u \subseteq T^w$, so

$$\begin{aligned}\mathit{cover}(v\langle V\rangle) \cap \mathit{cover}(u\langle U\rangle) &= \bigl(T^v \setminus \textstyle\bigcup_{x\in V} T^x\bigr) \cap \mathit{cover}(u\langle U\rangle)\\ &\subseteq (T^v \setminus T^w) \cap T^u = \emptyset\end{aligned} \tag{A.11}$$

and $\mathsf{proj}_{u\langle U\rangle}(\mathsf{merge}(v\langle V\rangle, w\langle W\rangle)) = \mathsf{proj}_{u\langle U\rangle}(w\langle W\rangle)$.

3. $w \in T \setminus \bigl(T^u \cup P^u\bigr)$: Since $\mathit{cover}(w\langle W\rangle) \subseteq T^w$ and $\mathit{cover}(u\langle U\rangle) \subseteq T^u$, in this case we have $\mathit{cover}(w\langle W\rangle) \cap \mathit{cover}(u\langle U\rangle) \subseteq T^w \cap T^u = \emptyset$, and again it holds that $\mathsf{proj}_{u\langle U\rangle}(\mathsf{merge}(v\langle V\rangle, w\langle W\rangle)) = \mathsf{proj}_{u\langle U\rangle}(v\langle V\rangle)$.

4. $w \in \mathit{int}(u\langle U\rangle)$: Note that this implies that $\mathit{cover}(v\langle V\rangle) \cap \mathit{cover}(u\langle U\rangle) \neq \emptyset$ and $\mathit{cover}(w\langle W\rangle) \cap \mathit{cover}(u\langle U\rangle) \neq \emptyset$, which means that the projected subconfigurations $\mathsf{proj}_{u\langle U\rangle}(v\langle V\rangle)$ and $\mathsf{proj}_{u\langle U\rangle}(w\langle W\rangle)$ both exist. Using simple set arithmetic we get that

$$\begin{aligned}&\mathit{cover}\bigl(\mathsf{proj}_{u\langle U\rangle}(\mathsf{merge}(v\langle V\rangle, w\langle W\rangle))\bigr)\\ &\quad= \mathit{cover}\bigl(\mathsf{merge}(v\langle V\rangle, w\langle W\rangle)\bigr) \cap \mathit{cover}(u\langle U\rangle)\\ &\quad= \bigl(\mathit{cover}(v\langle V\rangle) \mathbin{\dot\cup} \mathit{cover}(w\langle W\rangle)\bigr) \cap \mathit{cover}(u\langle U\rangle)\\ &\quad= \bigl(\mathit{cover}(v\langle V\rangle) \cap \mathit{cover}(u\langle U\rangle)\bigr)\\ &\qquad\qquad \mathbin{\dot\cup} \bigl(\mathit{cover}(w\langle W\rangle) \cap \mathit{cover}(u\langle U\rangle)\bigr)\\ &\quad= \mathit{cover}(\mathsf{proj}_{u\langle V\rangle}(v\langle V\rangle)) \mathbin{\dot\cup} \mathit{cover}(\mathsf{proj}_{u\langle V\rangle}(w\langle W\rangle))\end{aligned} \tag{A.12}$$

and applying Proposition A.10 we see that indeed

$$\begin{aligned}&\mathsf{proj}_{u\langle U\rangle}(\mathsf{merge}(v\langle V\rangle, w\langle W\rangle))\\ &\qquad= \mathsf{merge}(\mathsf{proj}_{u\langle V\rangle}(v\langle V\rangle), \mathsf{proj}_{u\langle V\rangle}(w\langle W\rangle)),\end{aligned} \tag{A.13}$$

i.e., the projection of $\mathsf{merge}(v\langle V\rangle, w\langle W\rangle)$ is derivable in one merger step from the projections of $v\langle V\rangle$ and $w\langle W\rangle$ as claimed.

It follows that either $\mathsf{proj}_{\mathbb{M}}(\mathsf{merge}(v\langle V\rangle, w\langle W\rangle)) = \mathsf{proj}_{\mathbb{M}}(\{v\langle V\rangle, w\langle W\rangle\})$, if there are no $u\langle U\rangle \in \mathbb{M}$ such that $w \in \mathit{int}(u\langle U\rangle)$, or $\mathsf{proj}_{\mathbb{M}}(\mathsf{merge}(v\langle V\rangle, w\langle W\rangle))$ can be derived from $\mathsf{proj}_{\mathbb{M}}(\{v\langle V\rangle, w\langle W\rangle\})$ by a single merger move for the unique $u\langle U\rangle \in \mathbb{M}$ such that $w \in \mathit{int}(u\langle U\rangle)$. $\square$

The other L-pebbling moves can also be taken care of easily, and we show next that, projecting any L-pebbling on any nontouching L-configuration $\mathbb{M}$, we get a legal L-pebbling inside the closure $cl(\mathbb{M})$ (modulo some technical details). In particular, this holds for the nonoverlapping pebblings of Definition A.21. This is part 2 in our proof outline.

LEMMA A.28. *For an arbitrary L-pebbling $\mathcal{L} = \{\mathbb{L}_0, \ldots, \mathbb{L}_\tau\}$ and a nontouching L-configuration $\mathbb{M}$, let $\mathsf{proj}_{\mathbb{M}}(\mathcal{L}) = \{\mathbb{L}'_0, \ldots, \mathbb{L}'_\tau\}$ for $\mathbb{L}'_t = \mathsf{proj}_{\mathbb{M}}(\mathbb{L}_t)$. Then $\mathsf{proj}_{\mathbb{M}}(\mathcal{L})$ is a legal L-pebbling if we eliminate idle moves $\mathbb{L}'_{t+1} = \mathbb{L}'_t$ and take care that one reversal or erasure $\mathbb{L}_t \rightsquigarrow \mathbb{L}_{t+1}$ in $\mathcal{L}$ may correspond to a sequence of reversals or erasures, respectively, in $\mathsf{proj}_{\mathbb{M}}(\mathcal{L})$. Legalizing $\mathsf{proj}_{\mathbb{M}}(\mathcal{L})$ by performing these moves one by one does not affect the pebbling cost, i.e., $\mathsf{cost}(\mathsf{proj}_{\mathbb{M}}(\mathcal{L})) = \max_{t \in \tau}\big\{\mathsf{cost}(\mathsf{proj}_{\mathbb{M}}(\mathbb{L}_t))\big\}$. Also, if $\mathcal{L}$ does not contain any reversals, then neither does $\mathsf{proj}_{\mathbb{M}}(\mathcal{L})$.*

Proof. The proof is by induction over the pebbling moves $\mathbb{L}_t \rightsquigarrow \mathbb{L}_{t+1}$ in $\mathcal{L}$. Case analysis:

Introduction. If $v\langle pred(v)\rangle \not\preceq \mathbb{M}$ the projection does not change, and otherwise adding $v\langle pred(v)\rangle = \mathsf{proj}_{\mathbb{M}}(v\langle pred(v)\rangle)$ is a legal introduction move.

Merger. Suppose $u\langle U\rangle = \mathsf{merge}(v\langle V\rangle, w\langle W\rangle)$. Clearly, $\mathbb{L}_t \setminus \big\{u\langle U\rangle, v\langle V\rangle, w\langle W\rangle\big\} = \mathbb{L}_{t+1} \setminus \big\{u\langle U\rangle, v\langle V\rangle, w\langle W\rangle\big\}$, so the only subconfigurations for which the projections can change are $u\langle U\rangle$, $v\langle V\rangle$, and $w\langle W\rangle$. This is Proposition A.27.

Reversal. If $v\langle V\rangle$ is derived from $u\langle U\rangle$ by reversal, we have $v\langle V\rangle \preceq u\langle U\rangle$ or, equivalently, $cover(v\langle V\rangle) \subseteq cover(u\langle U\rangle)$. Then

$$
\begin{aligned}
\text{(A.14)} \quad cover(\mathsf{proj}_{\mathbb{M}}(v\langle V\rangle)) &= cover(v\langle V\rangle) \cap cover(\mathbb{M}) \\
&\subseteq cover(u\langle U\rangle) \cap cover(\mathbb{M}) = cover(\mathsf{proj}_{\mathbb{M}}(u\langle U\rangle)),
\end{aligned}
$$

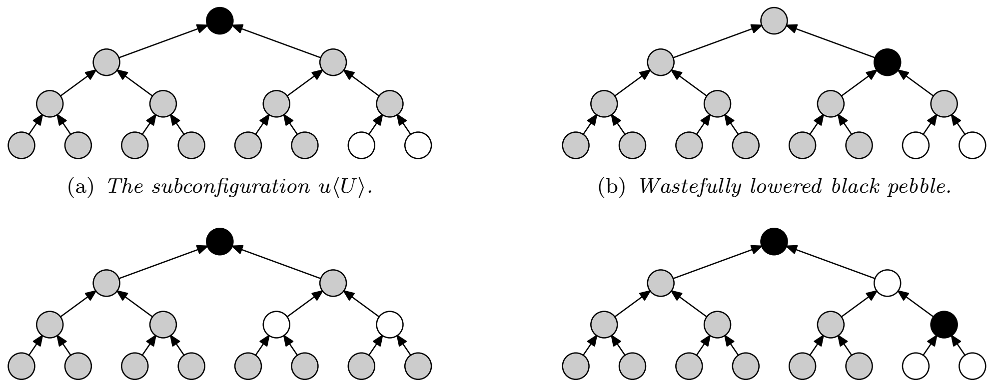
so adding $\mathsf{proj}_{\mathbb{M}}(v\langle V\rangle) \preceq \mathsf{proj}_{\mathbb{M}}(u\langle U\rangle)$ is a sequence of legal reversals. As this sequence of reversals is performed, the pebbling cost increases monotonically by part 1 of Proposition A.20.

Erasure. If $\mathbb{L}_{t+1} = \mathbb{L}_t \setminus \{v\langle V\rangle\}$ for $v\langle V\rangle \in \mathbb{L}_t$, removing $\mathsf{proj}_{\mathbb{M}}(v\langle V\rangle)$ from $\mathbb{L}'_t$ is a sequence of legal erasures. As this sequence of erasures is performed, the pebbling cost decreases monotonically by part 1 of Proposition A.20.

We see that the cost of this pebbling is $\max_{t\in[\tau]}\big\{\mathsf{proj}_{\mathbb{M}}(\mathbb{L}_t)\big\}$, and if $\mathcal{L}$ is reversal-free, then so is $\mathsf{proj}_{\mathbb{M}}(\mathcal{L})$, since every move in $\mathcal{L}$ is matched by the same kind of moves in $\mathsf{proj}_{\mathbb{M}}(\mathcal{L})$. ☐

A.6. A first (failed) attempt to eliminate reversal moves. In the light of Lemma A.28, the following transformation from a nonoverlapping pebbling $\mathcal{L}$ to a reversal-free pebbling $\mathcal{L}'$ seems very tempting: by forward induction over the moves in $\mathcal{L}$, replace each implosion $u\langle U\rangle \rightsquigarrow \mathbb{M}$ at time t by a local projection of $\{\mathbb{L}_0, \ldots, \mathbb{L}_t\}$ on $\mathbb{M}$. Since by induction there are no reversals before time t, the projection must be a reversal-free pebbling inside $cl(\mathbb{M})$. Doing this for all implosions, we get a globally reversal-free pebbling $\mathcal{L}'$ ending in $z\langle\emptyset\rangle$. This is the transformation described in part 3 of our roadmap for the proof of Lemma A.2.

There is only one problem. Although we will indeed get a complete L-pebbling of T, it is not true in general that $\mathsf{cost}(\mathsf{proj}_{\mathbb{M}}(\mathcal{L})) \le \mathsf{cost}(\mathcal{L})$. For instance, if $v\langle V\rangle \preceq u\langle\emptyset\rangle$ for $V \neq \emptyset$, then $\mathsf{proj}_{v\langle V\rangle}(u\langle\emptyset\rangle) = v\langle V\rangle$, and hence $\mathsf{cost}(\mathsf{proj}_{v\langle V\rangle}(u\langle\emptyset\rangle)) = 1 + |V| > \mathsf{cost}(u\langle\emptyset\rangle) = 1$. Looking at this counterexample, however, one might argue that having gotten as far as $u\langle\emptyset\rangle$, reversing to the weaker and more expensive configuration $v\langle V\rangle$ should be nonoptimal.



(a) *The subconfiguration $u\langle U\rangle$.* (b) *Wastefully lowered black pebble.* (c) *Superfluous white pebble.* (d) *Wasteful "split" of $u\langle U\rangle$.*

FIG. 14. *A subconfiguration $u\langle U\rangle$ and three wasteful implosions of $u\langle U\rangle$.*

What we want to do next is to define formally which reversals are *wasteful* in this sense, and to prove that for pebblings avoiding such wasteful reversals, projection does not increase the pebbling cost.

A.7. Pinpointing the problem: Wasteful reversal moves. Since the definition of wastefulness turns out to be quite technical, we first try to give some more intuition for which kind of reversals we disapprove of.

Example A.29. Consider the subconfiguration $u\langle U\rangle$ in Figure 14(a).

1. If $v \in T^u_*$, the reversal move from $u\langle U\rangle$ to $v\langle T^v_* \cap U\rangle$ seems reasonable only if $T^v_* \cap U \subsetneq U$, i.e., if we get rid of white pebbles by lowering the black pebble from u to v. The reversal in Figure 14(b) does not satisfy this, so it appears we should be better off keeping the original, stronger subconfiguration $u\langle U\rangle$ in Figure 14(a) instead.
2. Suppose that V is a simple set below u and above U in the sense that $P^u \cap V \neq \emptyset$ for all $u \in U$. Then we approve of the reversal $u\langle U\rangle \rightsquigarrow u\langle V\rangle$ only if for all $w \in V$ it holds that $T^w \cap U \neq \emptyset$. Otherwise, unnecessary white pebbles have been introduced, as in Figure 14(c).
3. If $u\langle U\rangle$ is imploded into a nontouching L-configuration $\{v_1\langle V_1\rangle, v_2\langle V_2\rangle\}$ such that, say, $v_2 \in T^{v_1}_*$, it should not be the case that $v_1\big\langle \big(V_1 \setminus P^{v_2}\big) \cup V_2\big\rangle \preceq u\langle U\rangle$, for if so we could have reversed to this stronger subconfiguration instead of $\{v_1\langle V_1\rangle, v_2\langle V_2\rangle\}$ at no extra cost. The implosion in Figure 14(d) violates this condition.

The reversals from $u\langle U\rangle$ in Figures 14(b), 14(c), and 14(d) are all examples of wasteful implosions for which our reversal-free pebbling $\mathcal{L}'$ constructed by projection may become more expensive than $\mathcal{L}$. Looking at these examples, it is easy to believe that such moves are nonoptimal and that it ought to be possible to eliminate them. The formal definition of wastefulness is as follows.

DEFINITION A.30 (wasteful implosion). *For a nontouching L-configuration $\mathbb{M} \preceq u\langle U\rangle$, the implosion $u\langle U\rangle \rightsquigarrow \mathbb{M}$ is* nonwasteful *if the following hold:*

1. *For every $v \in Bl(\mathbb{M}) \setminus \{u\}$ there is a $w \in U \cap T^{sibl(v)}$ such that for the path $Q_v = P^w \setminus P^{succ(v)}_*$ from w to $succ(v)$ it holds that $Q_v \cap \big(Bl(\mathbb{M}) \cup Wh(\mathbb{M})\big) = \emptyset$.*
2. *For every $v \in Wh(\mathbb{M})$ there is a $w \in U \cap T^v$ (possibly equal to v) such that for the path $Q_v = P^w \setminus P^v_*$ from w to v it holds that $Q_v \cap Bl(\mathbb{M}) = \emptyset$.*

3. *The paths between the vertices* $\bigl(Bl(\mathbb{M}) \cup Wh(\mathbb{M})\bigr) \setminus \{u\}$ *and (some subset of)* $Wh(u\langle U\rangle) = U$ *as described above can all be chosen pairwise disjoint, i.e., such that* $Q_v \cap Q_{v'} = \emptyset$ *if* $v \neq v'$.

If $u\langle U\rangle \rightsquigarrow \mathbb{M}$ *is not a nonwasteful implosion, it is said to be* wasteful.

Definition A.30 identifies the offending reversal moves for which our projective construction of a reversal-free but cheap pebbling fails (by not being cheap enough).

Loosely put, an implosion move $u\langle U\rangle \rightsquigarrow \mathbb{M}$ is nonwasteful if for every black and white pebble in $\mathbb{M}$ we can identify a distinct pebble in $u\langle U\rangle$ "explaining" why the implosion move could potentially be cost-saving. If there is no such correspondence (as in the implosions from Figure 14(a) to Figures 14(b), 14(c), and 14(d), which are all wasteful according to Definition A.30), the implosion intuitively seems nonoptimal, and it should be possible to do better by replacing this wasteful implosion by a stronger, nonwasteful one. And if we can assume that all wasteful implosions are changed into nonwasteful ones, it turns out that our projection idea from section A.6 does the trick!

Continuing according to part 4 in our proof plan, we show that for pebblings without wasteful moves the projective construction works. This is the next lemma. The thornier task of eliminating wasteful implosions is deferred to section A.8.

Lemma A.31. *Suppose that* $\mathcal{L} = \bigl\{\mathbb{L}_0 = \emptyset, \ldots, \mathbb{L}_{\tau-2}, \mathbb{L}_{\tau-1} = u\langle U\rangle \rightsquigarrow \mathbb{M}\bigr\}$ *is an L-pebbling ending with the nontouching L-configuration* $\mathbb{M}$ *and containing no reversal moves except for a final nonwasteful implosion* $u\langle U\rangle \rightsquigarrow \mathbb{M}$. *Then* $\mathsf{cost}(\mathsf{proj}_{\mathbb{M}}(\mathcal{L})) \leq \mathsf{cost}(\mathcal{L})$.

Proof. By Lemma A.15, without loss of generality we can assume that $cover(\mathbb{L}_t)$ grows monotonically with t until we reach $\mathbb{L}_{\tau-1} = u\langle U\rangle$. This means that, in particular, there will never be any subconfigurations covering vertices outside $cover(u\langle U\rangle)$ during the pebbling (since such subconfigurations would have to be erased in a redundant way), so it holds that $\mathbb{L}_t \preceq u\langle U\rangle$ for all t.

Let $\mathbb{L}'_t = \mathsf{proj}_{\mathbb{M}}(\mathbb{L}_t)$ for all $t < \tau$. By Lemma A.28, it suffices to show $\mathsf{cost}(\mathbb{L}'_t) \leq \mathsf{cost}(\mathbb{L}_t)$ to get $\mathsf{cost}(\mathsf{proj}_{\mathbb{M}}(\mathcal{L})) \leq \mathsf{cost}(\mathcal{L})$. This is so since we can go from $\mathbb{L}'_t$ to $\mathbb{L}'_{t+1}$ paying at most $\max\bigl\{\mathsf{cost}(\mathbb{L}'_t), \mathsf{cost}(\mathbb{L}'_{t+1})\bigr\}$, and for $\tau - 1$ we have $\mathsf{proj}_{\mathbb{M}}(\mathbb{L}_{\tau-1}) = \mathsf{proj}_{\mathbb{M}}(u\langle U\rangle) = \mathbb{M}$ since $\mathbb{M} \preceq u\langle U\rangle$.

By definition $\mathsf{cost}(\mathbb{L}_t) = \bigl|Bl(\mathbb{L}_t) \cup Wh(\mathbb{L}_t)\bigr|$, so to prove $\mathsf{cost}(\mathbb{L}'_t) \leq \mathsf{cost}(\mathbb{L}_t)$ it is enough to find for each vertex $v \in Bl(\mathbb{L}'_t) \cup Wh(\mathbb{L}'_t)$ an associated vertex $v_L \in Bl(\mathbb{L}_t) \cup Wh(\mathbb{L}_t)$ such that $v_L \neq v^*_L$ if $v \neq v^*$. These associated vertices are exactly what Definition A.30 will help us find.

If $v^* \in \bigl(Bl(\mathbb{L}'_t) \cup Wh(\mathbb{L}'_t)\bigr) \cap \bigl(Bl(\mathbb{L}_t) \cup Wh(\mathbb{L}_t)\bigr)$, an obvious choice is $v^*_L = v^*$. Suppose therefore that $v \in \bigl(Bl(\mathbb{L}'_t) \cup Wh(\mathbb{L}'_t)\bigr) \setminus \bigl(Bl(\mathbb{L}_t) \cup Wh(\mathbb{L}_t)\bigr)$. Then Lemma A.23 tells us that $v \in \partial\mathbb{M}$, that v has the same color in $\mathbb{L}'_t$ and $\mathbb{M}$, i.e., either $v \in Bl(\mathbb{L}'_t) \cap Bl(\mathbb{M})$ or $v \in Wh(\mathbb{L}'_t) \cap Wh(\mathbb{M})$, and that there is a $w_v\bigl\langle W_v\bigr\rangle \in \mathbb{L}_t$ such that $v \in int\bigl(w_v\bigl\langle W_v\bigr\rangle\bigr)$, namely, the $w_v\bigl\langle W_v\bigr\rangle$ projecting the pebble on v. We choose $v_L \in Bl(\mathbb{L}_t) \cup Wh(\mathbb{L}_t)$ for such vertices v by first associating a unique $v_u \in U = Wh(u\langle U\rangle)$ to v as follows.

1. If $v \in Bl(\mathbb{L}'_t) \cap Bl(\mathbb{M})$, pick a vertex $v_u \in U \cap T^{sibl(v)}$ and a path $Q_v = P^{v_u} \setminus P^{succ(v)}_*$ from v_u to $succ(v)$ such that $Q_v \cap \bigl(Bl(\mathbb{M}) \cup Wh(\mathbb{M})\bigr) = \emptyset$ as guaranteed by Definition A.30. For the subconfiguration $w_v\langle W_v\rangle \in \mathbb{L}_t$ projecting the black pebble on v, we must have $succ(v) \in cover\bigl(w_v\bigl\langle W_v\bigr\rangle\bigr)$ since $v \in int\bigl(w_v\bigl\langle W_v\bigr\rangle\bigr)$, and thus $succ(v) \in Q_v \cap cover\bigl(w_v\bigl\langle W_v\bigr\rangle\bigr) \neq \emptyset$.
2. If $v \in Wh(\mathbb{L}'_t) \cap Wh(\mathbb{M})$, pick $v_u \in U \cap T^v$ and $Q_v = P^{v_u} \setminus P^v_*$ such that $Q_v \cap Bl(\mathbb{M}) = \emptyset$ as provided by Definition A.30. For $w_v\langle W_v\rangle \in \mathbb{L}_t$ projecting the white pebble on v, we have $v \in int\bigl(w_v\bigl\langle W_v\bigr\rangle\bigr) \subseteq cover\bigl(w_v\bigl\langle W_v\bigr\rangle\bigr)$, so

$v \in Q_v \cap \mathit{cover}\bigl(w_v\langle W_v\rangle\bigr) \neq \emptyset$.

According to Definition A.30, all the paths Q_v above can be chosen disjoint.

We claim that $W_v \cap Q_v \neq \emptyset$ for all $v \in \bigl(\mathit{Bl}(\mathbb{L}'_t) \cup \mathit{Wh}(\mathbb{L}'_t)\bigr) \setminus \bigl(\mathit{Bl}(\mathbb{L}_t) \cup \mathit{Wh}(\mathbb{L}_t)\bigr)$. Given this claim, we can choose as our associated vertex $v_L \in \mathit{Bl}(\mathbb{L}_t) \cup \mathit{Wh}(\mathbb{L}_t)$ for v the (unique) vertex $v_L \in W_v \cap Q_v$. Since all paths Q_v are disjoint, it follows that all such vertices v_L are distinct. They must also be distinct from the vertices $v^* \in \bigl(\mathit{Bl}(\mathbb{L}'_t) \cup \mathit{Wh}(\mathbb{L}'_t)\bigr) \cap \bigl(\mathit{Bl}(\mathbb{L}_t) \cup \mathit{Wh}(\mathbb{L}_t)\bigr)$. This is so since the path Q_v does not intersect $\mathbb{M}$ except possibly in v, or in formal notation $\bigl(Q_v \setminus \{v\}\bigr) \cap \mathit{cl}(\mathbb{M}) = \emptyset$, and by construction the associated vertex $v_L \in W_v \cap Q_v$ is always distinct from v (since $v \in \mathit{int}(w_v\langle W_v\rangle)$ but by definition $\mathit{int}(w_v\langle W_v\rangle) \cap W_v = \emptyset$). Hence, for all chosen representatives $v_L \in Q_v$ it holds that $v_L \notin \mathit{cl}(\mathbb{M}) \supseteq \mathit{Bl}(\mathbb{L}'_t) \cup \mathit{Wh}(\mathbb{L}'_t)$. Summing this up, for each $v \in \mathit{Bl}(\mathbb{L}'_t) \cup \mathit{Wh}(\mathbb{L}'_t)$ we have found a distinct associated vertex $v_L \in \mathit{Bl}(\mathbb{L}_t) \cup \mathit{Wh}(\mathbb{L}_t)$, and it follows that $\mathsf{cost}(\mathbb{L}'_t) \leq \mathsf{cost}(\mathbb{L}_t)$.

It remains to prove the claim that $W_v \cap Q_v \neq \emptyset$ for the path Q_v and subconfiguration $w_v\langle W_v\rangle \in \mathbb{L}_t$ such that $v \in \mathit{int}(w_v\langle W_v\rangle)$ found for each $v \in \bigl(\mathit{Bl}(\mathbb{L}'_t) \cup \mathit{Wh}(\mathbb{L}'_t)\bigr) \setminus \bigl(\mathit{Bl}(\mathbb{L}_t) \cup \mathit{Wh}(\mathbb{L}_t)\bigr)$ above. Fix such a triple $\bigl(v, Q_v, w_v\langle W_v\rangle\bigr)$. By construction, $\mathbb{L}'_t \preceq \mathbb{L}_t \preceq u\langle U\rangle$, so, in particular, $w_v\langle W_v\rangle \preceq u\langle U\rangle$. Recall that we showed above that $Q_v \cap \mathit{cover}\bigl(w_v\langle W_v\rangle\bigr) \neq \emptyset$. Furthermore, we have $Q_v \nsubseteq \mathit{cover}(u\langle U\rangle)$ since the lowest vertex in Q_v is a white pebble of $u\langle U\rangle$. Now if $W_v \cap Q_v = \emptyset$ would hold, this would imply by Proposition A.8 that all of Q_v lies inside $w_v\langle W_v\rangle$, i.e., $Q_v \subseteq \mathit{cover}(w_v\langle W_v\rangle)$. But this yields the contradiction $w_v\langle W_v\rangle \npreceq u\langle U\rangle$. Thus $W_v \cap Q_v \neq \emptyset$, which proves the claim. The lemma follows. ∎

We can use Lemma A.31 to eliminate nonwasteful implosions one by one without increasing the cost, resulting in a reversal-free L-pebbling.

Lemma A.32. *Let $\mathcal{L} = \bigl\{\mathbb{L}_0, \ldots, \mathbb{L}_\tau\bigr\}$ be a nonoverlapping complete L-pebbling of T without wasteful implosions. Then from $\mathcal{L}$ we can construct a complete L-pebbling $\mathcal{L}'$ of T without reversal moves such that $\mathsf{cost}(\mathcal{L}') \leq \mathsf{cost}(\mathcal{L})$.*

Proof. The proof is by induction over the number of implosions. Plainly, if we can go from an L-pebbling $\mathcal{L}$ with n nonwasteful implosions to an L-pebbling $\mathcal{L}'$ with $n-1$ nonwasteful implosions and $\mathsf{cost}(\mathcal{L}') \leq \mathsf{cost}(\mathcal{L})$, the lemma follows by the induction principle.

Consider the subpebbling $\mathcal{L}^*$ of $\mathcal{L}$ consisting of the moves up to and including the first implosion. That is, $\mathcal{L}^* = \{\mathbb{L}_0, \ldots, \mathbb{L}_{\tau^*} \rightsquigarrow (\mathbb{L}_{\tau^*} \setminus u\langle U\rangle) \cup \mathbb{M}\}$ is nonoverlapping and reversal-free except for a final nonwasteful implosion $u\langle U\rangle \rightsquigarrow \mathbb{M}$.

By Definition A.22, the L-configuration $\mathbb{L}_{\tau^*}$ is nontouching, and using Proposition A.24, each $\mathbb{L}_t$, $t < \tau^*$, can be written as a union of mutually nontouching L-configurations $\mathbb{L}_t = \bigcup_{v\langle V\rangle \in \mathbb{L}_{\tau^*}} \mathbb{L}^v_t$ for $\mathbb{L}^v_t = \mathsf{proj}_{v\langle V\rangle}(\mathbb{L}_t)$ such that $\mathsf{cost}(\mathbb{L}_t) = \sum_{v\langle V\rangle \in \mathbb{L}_{\tau^*}} \mathsf{cost}(\mathbb{L}^v_t)$. Appealing to Lemma A.28, we see that for all $v\langle V\rangle \in \mathbb{L}_{\tau^*}$, it holds that $\mathcal{L}^v = \bigl\{\mathbb{L}^v_0, \ldots, \mathbb{L}^v_{\tau^*-1}, \mathbb{L}^v_{\tau^*} = v\langle V\rangle\bigr\}$ are pairwise nontouching pebblings without reversals.

Lemma A.31 now says that locally, the pebbling $\mathcal{L}^u$ corresponding to the imploded subconfiguration $u\langle U\rangle$ can be replaced by a reversal-free pebbling $\mathsf{proj}_{\mathbb{M}}(\mathcal{L}^u)$ without increasing the local pebbling cost. Then Proposition A.24 says that we can substitute $\mathsf{proj}_{\mathbb{M}}(\mathcal{L}^u)$ for $\mathcal{L}^u$ in $\mathcal{L}^*$ without increasing the *global* pebbling cost.

Doing this local substition, instead of $\mathcal{L}^*$ we get a reversal-free pebbling ending in the same L-configuration $(\mathbb{L}_{\tau^*} \setminus u\langle U\rangle) \cup \mathbb{M}$, and it is easy to check using Lemma A.15 and (the proof of) Lemma A.16 that this reversal-free pebbling can be made nonoverlapping. If we concatenate this pebbling with the rest of the pebbling moves in $\mathcal{L} \setminus \mathcal{L}^*$, we have a nonoverlapping complete L-pebbling with one less implosion move. ∎

This concludes part 4 in the proof outline in section A.1.

A.8. Wasteful reversal moves can be replaced. All that remains now is to show that in an arbitrary nonoverlapping L-pebbling we can always replace wasteful implosions by nonwasteful ones without increasing the pebbling cost by more than a constant factor. It will take a couple of technical lemmas before we get there, but the intuition from Example A.29 is clear: if $\mathbb{L}_t \rightsquigarrow \mathbb{L}_{t+m+1}$ is a wasteful implosion, we should be able to match this move with a nonwasteful implosion $\mathbb{L}'_t \rightsquigarrow \mathbb{L}'_{t+m+1}$ instead, where $\mathbb{L}'_i \succeq \mathbb{L}_i$ and $\mathsf{cost}(\mathbb{L}'_i) \leq \mathsf{cost}(\mathbb{L}_i)$ for $i = t, t+m+1$. The only thing that complicates the matter is that we may have to pay extra for the transitional L-configurations during the implosion $\mathbb{L}'_t \rightsquigarrow \mathbb{L}'_{t+m+1}$ because of overlapping subconfigurations.

The cornerstone of our proof is the fact that for every wasteful implosion move $u\langle U\rangle \rightsquigarrow \mathbb{L}$, there is a nonwasteful implosion move to $\mathbb{M} \succ \mathbb{L}$ with $\mathsf{cost}(\mathbb{M}) \leq \mathsf{cost}(\mathbb{L})$.

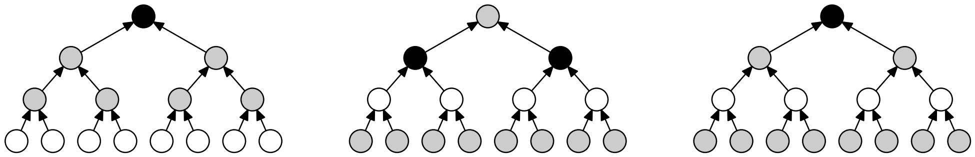
LEMMA A.33. *If $u\langle U\rangle \rightsquigarrow \mathbb{L}$ is a wasteful implosion, then there is a nontouching $\mathbb{M}$ such that $u\langle U\rangle \succeq \mathbb{M} \succ \mathbb{L}$, $\mathsf{cost}(\mathbb{M}) \leq \min\{\mathsf{cost}(u\langle U\rangle), \mathsf{cost}(\mathbb{L})\}$, and $u\langle U\rangle \rightsquigarrow \mathbb{M}$ is a nonwasteful implosion.*

Proof. If $u\langle U\rangle \rightsquigarrow \mathbb{M}$ is a nonwasteful implosion, it holds that $\mathsf{cost}(\mathbb{M}) = \left|Bl(\mathbb{M})\right| + \left|Wh(\mathbb{M})\right| \leq \mathsf{cost}(u\langle U\rangle) = 1 + |U|$, since by Definition A.30 every $v \in \big(Bl(\mathbb{M}) \cup Wh(\mathbb{M})\big) \setminus \{u\}$ can be associated with a distinct $w \in U$.

We demonstrate that if $u\langle U\rangle \rightsquigarrow \mathbb{L}$ is a wasteful implosion, we can find an $\mathbb{M}$ such that $u\langle U\rangle \succeq \mathbb{M} \succ \mathbb{L}$ and $\mathsf{cost}(\mathbb{M}) \leq \mathsf{cost}(\mathbb{L})$. If $u\langle U\rangle \rightsquigarrow \mathbb{M}$ is also a wasteful implosion, we repeat this construction to obtain L-configurations $\mathbb{M}'$ with $u\langle U\rangle \succeq \mathbb{M}' \succ \mathbb{M}$ and $\mathsf{cost}(\mathbb{M}') \leq \mathsf{cost}(\mathbb{M})$, $\mathbb{M}''$ with $u\langle U\rangle \succeq \mathbb{M}'' \succ \mathbb{M}'$ $\mathsf{cost}(\mathbb{M}'') \leq \mathsf{cost}(\mathbb{M}')$, etc. Sooner or later the process must terminate for some $\mathbb{M}^{(k)} \preceq u\langle U\rangle$ such that $u\langle U\rangle \rightsquigarrow \mathbb{M}^k$ is nonwasteful, since the set of covered vertices $cover\big(\mathbb{M}^{(i)}\big)$ grows in every step. If nothing else, we will end up with $\mathbb{M}^{(k)} = u\langle U\rangle$, and by definition the trivial implosion $u\langle U\rangle \rightsquigarrow u\langle U\rangle$ is nonwasteful.

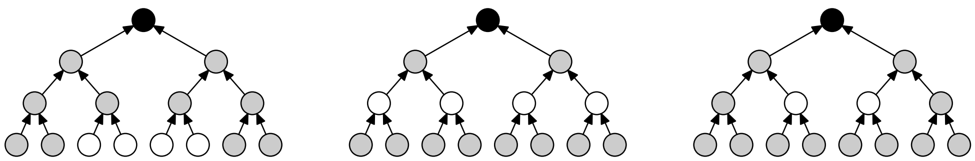
According to Definition A.30, the configuration $\mathbb{L}$ can be wasteful with respect to $u\langle U\rangle$ in three ways. For the purpose of the case analysis, it appears more natural in this lemma (but only in this lemma) to traverse the paths in T in the reverse direction, so that we move downward from above.

1. Some black pebble $v \in Bl(\mathbb{L}) \setminus \{u\}$ lacks a path. That is, all paths from $succ(v)$ downward in the sibling subtree $T^{sibl(v)}$ to white pebbles in U intersect with other pebbled vertices in $\mathbb{L}$.
   If $succ(v) \in Wh(\mathbb{L})$ we must have $succ(v) \in cover(u\langle U\rangle)$ by convexity (Definition A.7 and Proposition A.8), so we can enlarge the cover by adding the subconfiguration $\mathsf{canon}(\{succ(v)\}) = succ(v)\langle v, sibl(v)\rangle$ to $\mathbb{L}$ and canonize to get $\mathbb{M} = \mathsf{canon}(\mathbb{L} \cup succ(v)\langle v, sibl(v)\rangle) \succ \mathbb{L}$ with $\mathsf{cost}(\mathbb{M}) \leq \mathsf{cost}(\mathbb{L}) + |\{sibl(v)\}| - |\{v, succ(v)\}| < \mathsf{cost}(\mathbb{L})$. We note that this is so since a black and a white pebble on the same vertex "cancel" and can be eliminated by a merger on this vertex. Figure 14(d) is an illustration of this case.
   Otherwise, since $\mathbb{L}$ is nontouching all paths from $succ(v)$ downward in $T^{sibl(v)}$ are either blocked by $r_1, \ldots, r_m \in Bl(\mathbb{L}) \cap T^{sibl(v)}$ or reach sources in $T^{sibl(v)}$ without passing pebbled vertices (if there are no black pebbles in $T^{sibl(v)}$, we let $m = 0$). By the convexity of $cover(u\langle U\rangle)$, we conclude that $V = T^{succ(v)} \setminus \big(T^v \cup \bigcup_{i \in [m]} T^{r_i}\big) \subseteq cover(u\langle U\rangle)$, so we can add $\mathsf{canon}(V) = succ(v)\langle v, r_1, \ldots, r_m\rangle \preceq u\langle U\rangle$ to $\mathbb{L}$. This move increases the cost only by 1 for the unpebbled vertex $succ(v)$, since the vertices $v, r_1, \ldots, r_m$ are all pebbled. Setting $\mathbb{M} = \mathsf{canon}(\mathbb{L} \cup succ(v)\langle v, r_1, \ldots, r_m\rangle) \succ \mathbb{L}$ removes the pebbles from



(a) *The subconfiguration.* (b) *Wasteful implosion* $\mathbb{L}$. (c) *Nonwasteful* $\mathbb{M} \succ \mathbb{L}$.

FIG. 15. *Illustration of case* 1 *in the proof of Lemma* A.33.



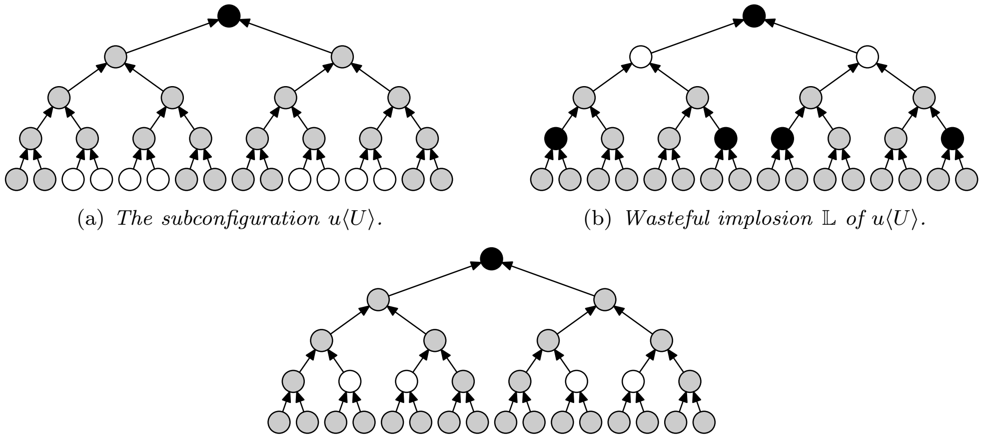
(a) *The subconfiguration.* (b) *Wasteful implosion* $\mathbb{L}$. (c) *Nonwasteful* $\mathbb{M} \succ \mathbb{L}$.

FIG. 16. *Illustration of case* 2 *in the proof of Lemma* A.33.

the black- and white-pebbled vertices $v, r_1, \ldots, r_m$ and decreases the cost by at least 1, so $\mathit{cost}(\mathbb{M}) \leq \mathit{cost}(\mathbb{L})$. See Figure 15 for a simple example.

2. There is a white pebble $w \in \mathit{Wh}(\mathbb{L})$ such that all paths downward in T^w either are blocked by $r_1, \ldots, r_m \in \mathit{Bl}(\mathbb{L}) \cap T^w_*$ or reach sources in T^w without passing pebbled vertices. If so, we have $V = T^w \setminus \bigcup_{i \in [m]} T^{r_i} \subseteq \mathit{cover}(u\langle U \rangle)$, and we can add $\mathsf{canon}(V) = w\langle r_1, \ldots, r_m \rangle \preceq u\langle U \rangle$ to $\mathbb{L}$ at no extra cost and set $\mathbb{M} = \mathsf{canon}(\mathbb{L} \cup w\langle r_1, \ldots, r_m \rangle) \succ \mathbb{L}$. Here we get a strict inequality $\mathit{cost}(\mathbb{M}) < \mathit{cost}(\mathbb{L})$ since the canonization eliminates at least the pebble on w. This case is illustrated in Figure 16.
3. There are paths for all $v \in \big(\mathit{Bl}(\mathbb{L}) \cup \mathit{Wh}(\mathbb{L})\big) \setminus \{u\}$ to vertices in U in the sense of Definition A.30, but they cannot be chosen disjoint. Start picking disjoint paths bottom-up from the leaves toward the root so that when we choose a path for a white pebble $v \in \mathit{Wh}(\mathbb{L})$ we have already determined paths for all $w \in \big(\mathit{Bl}(\mathbb{L}) \cup \mathit{Wh}(\mathbb{L})\big) \cap T^v_*$, and when we choose a path for a black pebble $v \in \mathit{Bl}(\mathbb{L})$ we have already determined paths for all $w \in \big(\mathit{Bl}(\mathbb{L}) \cup \mathit{Wh}(\mathbb{L})\big) \cap T^{\mathit{sibl}(v)}$, or in fact for all of $T^{\mathit{succ}(v)} \setminus \{v\}$. This can be done since for black pebbles, the vertex $\mathit{sibl}(v)$ itself cannot be black-pebbled in $\mathbb{L}$, for if so there would be no path for v and we would be in case 1. For the same reason, $\mathit{succ}(v)$ is not white-pebbled in $\mathbb{L}$, and then $\mathit{sibl}(v)$ cannot be white-pebbled, nor can $\mathit{succ}(v)$ be black-pebbled, since $\mathbb{L}$ is nontouching.

   At some point we reach a v such that no matter how we choose the paths below, we cannot choose a disjoint path for v. Consider the color of v.
   (a) v is black. Then there are white pebbles in $U \cap T^{\mathit{sibl}(v)}_*$ reachable from v, but they are all blocked by paths already chosen from black-pebbled vertices $r_1, \ldots, r_m \in \mathit{Bl}(\mathbb{L}) \cap T^{\mathit{sibl}(v)}_*$. (Note that all white pebbles in $T^{\mathit{sibl}(v)}_*$ are located below black pebbles since $\mathbb{L}$ is nontouching, so no paths from white-pebbled vertices in $T^{\mathit{sibl}(v)}_*$ are among the "blocking paths" for our vertex v.) This means that $\{\mathit{succ}(r_i) \mid i \in [m]\} \subseteq \mathit{cover}(u\langle U \rangle)$ by the convexity of $\mathit{cover}(u\langle U \rangle)$, so we can add all of the subconfig-



(a) *The subconfiguration* $u\langle U\rangle$.

(b) *Wasteful implosion* $\mathbb{L}$ *of* $u\langle U\rangle$.

(c) *Nonwasteful implosion* $u\langle U\rangle \rightsquigarrow \mathbb{M} \succ \mathbb{L}$.

FIG. 17. *Illustration of case* 3 *in the proof of Lemma* A.33.

urations $\mathsf{canon}(\{succ(r_i) \mid i \in [m]\}) = \{succ(r_i)\langle r_i, sibl(r_i)\rangle \mid i \in [m]\}$ to $\mathbb{L}$ at an additional cost $2m$. By similar reasoning we can also add $succ(v)\langle v, succ(r_1), \ldots, succ(r_m)\rangle$ at a further cost of 1 for the unpebbled vertex $succ(v)$. When we canonize this L-configuration, the black and white pebbles on the vertices $v, r_1, \ldots, r_m, succ(r_1), \ldots, succ(r_m)$ all cancel and disappear and the cost decreases by $2m+1$, resulting in $\mathbb{M} \succ \mathbb{L}$ with $\textit{cost}(\mathbb{M}) \leq \textit{cost}(\mathbb{L})$.

(b) v is white. The construction is analogous. Let the blocking black pebbles in T^v_* be $r_1, \ldots, r_m \in Bl(\mathbb{L}) \cap T^v_*$. Again $succ(r_i)\langle r_i, sibl(r_i)\rangle$, $i \in [m]$, can be added at an extra cost $2m$. Since $succ(r_i)$, $i \in [m]$, block all paths from v we have $T^v \setminus \bigcup_{i\in[m]} T^{succ(r_i)} \subseteq cover(u\langle U\rangle)$, so $v\langle succ(r_1), \ldots, succ(r_m)\rangle$ can be added as well at no additional cost. Canonizing decreases the cost by $2m+1$, which yields $\mathbb{M} \succ \mathbb{L}$ with $\textit{cost}(\mathbb{M}) < \textit{cost}(\mathbb{L})$. The transition from Figure 17(b) to Figure 17(c) is accomplished by applying this procedure twice.

In all cases we can find a nontouching L-configuration $\mathbb{M}$ such that $u\langle U\rangle \succeq \mathbb{M} \succ \mathbb{L}$ and $\textit{cost}(\mathbb{M}) \leq \textit{cost}(\mathbb{L})$. The lemma follows by induction. ∎

The following transitivity property of nonwasteful implosions is an immediate consequence of Definition A.30.

OBSERVATION A.34. *If* $u\langle U\rangle \rightsquigarrow \{v_i\langle V_i\rangle \mid i \in [m]\}$ *and* $v_i\langle V_i\rangle \rightsquigarrow \mathbb{M}_i$ *for* $i \in [m]$ *are all nonwasteful implosions, then* $u\langle U\rangle \rightsquigarrow \{\mathbb{M}_i \mid i \in [m]\}$ *is a nonwasteful implosion.*

Proof. For each $i \in [m]$, concatenate the paths from $\mathbb{M}_i$ to $v_i\langle V_i\rangle$ provided by Definition A.30 with those from $v_i\langle V_i\rangle$ to $u\langle U\rangle$ provided by the same definition. The result is a set of disjoint paths from $\bigcup_{i\in[m]} \mathbb{M}_i$ to $u\langle U\rangle$ as required by Definition A.30. ∎

It follows from Observation A.34 that if $u\langle U\rangle \rightsquigarrow \mathbb{L}$ is a wasteful implosion and $u\langle U\rangle \rightsquigarrow \mathbb{M} \succ \mathbb{L}$ is a corresponding nonwasteful implosion for $\mathbb{M}$ minimal, then all nontrivial "local implosions" from subconfigurations in $\mathbb{M}$ to sets of subconfigurations in $\mathbb{L}$ are wasteful. We formalize this as a lemma.

LEMMA A.35. *Suppose that* $u\langle U\rangle \rightsquigarrow \mathbb{L}$ *is a wasteful implosion and let* $\mathbb{M} \succ \mathbb{L}$ *be minimal such that* $u\langle U\rangle \rightsquigarrow \mathbb{M}$ *is nonwasteful. Then for each* $v\langle V\rangle \in \mathbb{M}$ *and each nontouching* $\mathbb{L}'$ *such that* $\mathbb{M} \succ \mathbb{L}' \succeq \mathbb{L}$*, either* $\mathsf{proj}_{v\langle V\rangle}(\mathbb{L}') = v\langle V\rangle$ *or* $v\langle V\rangle \rightsquigarrow$

$\mathsf{proj}_{v\langle V\rangle}(\mathbb{L}')$ *is a wasteful implosion. In particular, for each* $v\langle V\rangle \in \mathbb{M}$ *it holds that* $\mathit{cost}(v\langle V\rangle) \leq \mathit{cost}(\mathsf{proj}_{v\langle V\rangle}(\mathbb{L}'))$.

Proof. Given $u\langle U\rangle$ and $\mathbb{L}$ as in the statement of the lemma, we know from Lemma A.33 that we can find some $\mathbb{M}$ such that $u\langle U\rangle \rightsquigarrow \mathbb{M}$ is a nonwasteful implosion and $u\langle U\rangle \succeq \mathbb{M} \succeq \mathbb{L}$. Pick such an $\mathbb{M}$ which is minimal with respect to $\preceq$. Note that by the definition of implosion moves, $\mathbb{L}$ and $\mathbb{M}$ are nontouching.

Suppose that there is a subconfiguration $v\langle V\rangle \in \mathbb{M}$ and an L-configuration $\mathbb{L}'$ with $\mathbb{M} \succ \mathbb{L}' \succeq \mathbb{L}$ such that $\mathsf{proj}_{v\langle V\rangle}(\mathbb{L}') \prec v\langle V\rangle$ and $v\langle V\rangle \rightsquigarrow \mathsf{proj}_{v\langle V\rangle}(\mathbb{L}')$ is a nonwasteful implosion. Then by the transitivity in Observation A.34 it holds that $\mathbb{M}' = \big(\mathbb{M} \cup \mathsf{proj}_{v\langle V\rangle}(\mathbb{L}')\big) \setminus v\langle V\rangle \prec \mathbb{M}$ is a nonwasteful implosion of $u\langle U\rangle$. This contradicts the minimality of $\mathbb{M}$.

If $v\langle V\rangle \rightsquigarrow \mathsf{proj}_{v\langle V\rangle}(\mathbb{L}')$ is a wasteful implosion, Lemma A.33 says that there exists a nonwasteful implosion locally to an L-configuration $\mathbb{M}_v$ with $v\langle V\rangle \succeq \mathbb{M}_v \succ \mathsf{proj}_{v\langle V\rangle}(\mathbb{L}')$ such that $\mathit{cost}(\mathbb{M}_v) \leq \min\big\{\mathit{cost}(v\langle V\rangle), \mathit{cost}(\mathsf{proj}_{v\langle V\rangle}(\mathbb{L}'))\big\}$, and, in particular, $\mathit{cost}(\mathbb{M}_v) \leq \mathit{cost}(\mathsf{proj}_{v\langle V\rangle}(\mathbb{L}'))$. But we have just proven that this nonwasteful $\mathbb{M}_v$ must be identical with $v\langle V\rangle$, so $\mathit{cost}(v\langle V\rangle) \leq \mathit{cost}(\mathsf{proj}_{v\langle V\rangle}(\mathbb{L}'))$. ∎

Very roughly, the next lemma says that wasteful implosions are preserved under mergers.

LEMMA A.36. *Suppose for* $i = 1, 2$ *that* $u_i\langle U_i\rangle \succeq \mathbb{L}_i$ *and* $\mathit{cost}(u_i\langle U_i\rangle) \leq \mathit{cost}(\mathbb{L}_i)$ *for* $\mathbb{L}_i$ *nonoverlapping, and that* $u_1\langle U_1\rangle$ *and* $u_2\langle U_2\rangle$ *are mutually nonoverlapping with* $u_2 \in U_1$. *Then it holds that* $\mathit{cost}(\mathsf{merge}(u_1\langle U_1\rangle, u_2\langle U_2\rangle)) \leq \mathit{cost}(\mathbb{L}_1 \cup \mathbb{L}_2)$.

Proof. The L-configurations $\mathbb{L}_1$ and $\mathbb{L}_2$ must be mutually nonoverlapping since they are covered by $u_1\langle U_1\rangle$ and $u_2\langle U_2\rangle$, respectively. The only way that $\mathit{cost}(\mathbb{L}_1 \cup \mathbb{L}_2)$ could be less than $\mathit{cost}(\mathsf{merge}(u_1\langle U_1\rangle, u_2\langle U_2\rangle)) = \mathit{cost}(u_1\langle U_1\rangle) + \mathit{cost}(u_2\langle U_2\rangle) - 1 \leq \mathit{cost}(\mathbb{L}_1) + \mathit{cost}(\mathbb{L}_2) - 1$ is if there were at least two vertices in $\bigcap_{i=1,2}\big(\mathit{Bl}(\mathbb{L}_i) \cup \mathit{Wh}(\mathbb{L}_i)\big)$. But $\mathit{Bl}(\mathbb{L}_i) \cup \mathit{Wh}(\mathbb{L}_i) \subseteq \mathit{cl}(\mathbb{L}_i) \subseteq \mathit{cl}(u_i\langle U_i\rangle)$ since $\mathbb{L}_i \preceq u_i\langle U_i\rangle$ by the assumptions of the lemma, and also by assumption $\mathit{cl}(u_1\langle U_1\rangle) \cap \mathit{cl}(u_1\langle U_1\rangle) = \{u_2\}$ since $u_1\langle U_1\rangle$ and $u_2\langle U_2\rangle$ are mergeable (recall Example A.14), so this is impossible. ∎

Combining Lemmas A.35 and A.36, we can provide the fifth and final component in the proof of Lemma A.2, namely, that any nonoverlapping L-pebbling $\mathcal{L}$ can be transformed into a pebbling $\mathcal{L}'$ without wasteful implosions such that $\mathcal{L}'$ has asymptotically the same cost as $\mathcal{L}$.

LEMMA A.37. *Suppose that* $\mathcal{L}$ *is a nonoverlapping complete L-pebbling of* T. *Then we can find a nonoverlapping complete L-pebbling* $\mathcal{L}'$ *of* T *without wasteful implosions such that* $\mathit{cost}(\mathcal{L}') \leq 2 \cdot \mathit{cost}(\mathcal{L})$.

Proof. In this proof, let us assume for simplicity (and without loss of generality concerning pebbling cost, by the proof of Lemma A.26) that each introduction, expansion, or implosion move in Definition A.21 takes exactly one time step.

Given a nonoverlapping L-pebbling $\mathcal{L}$, we build a nonoverlapping L-pebbling $\mathcal{L}'$ without wasteful implosions such that if we let $\mathbb{L}_i \in \mathcal{L}$ denote the starting configuration of the ith move in $\mathcal{L}$, there is a corresponding $\mathbb{L}'_i \in \mathcal{L}'$ such that the following invariants hold:

1. $\mathbb{L}'_i$ is nontouching.
2. $\mathbb{L}'_i \succeq \mathbb{L}_i$.
3. For all $u\langle U\rangle \in \mathbb{L}'_i$, it holds that $\mathit{cost}(u\langle U\rangle) \leq \mathit{cost}(\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_i))$.
4. The cost of the L-pebbling transition from $\mathbb{L}'_{i-1}$ to $\mathbb{L}'_i$ in $\mathcal{L}'$ does not exceed $2 \cdot \max\big\{\mathit{cost}(\mathbb{L}_{i-1}), \mathit{cost}(\mathbb{L}_i)\big\}$.

To see that the lemma follows from this, note that invariants 1 and 2 imply that for every $v\langle V\rangle \in \mathbb{L}_i$ there is a $u\langle U\rangle \in \mathbb{L}'_i$ such that $v\langle V\rangle \preceq u\langle U\rangle$. In particular, for $\mathbb{L}_\tau = z\langle\emptyset\rangle$ we have $z\langle\emptyset\rangle \in \mathbb{L}'_\tau$, since $z\langle\emptyset\rangle$ is the maximal element with respect to $\preceq$. Then plugging invariant 3 into Proposition A.20, part 4, we get $\mathit{cost}(\mathbb{L}'_i) = \sum_{u\langle U\rangle\in\mathbb{L}'_i} \mathit{cost}(u\langle U\rangle) \le \sum_{u\langle U\rangle\in\mathbb{L}'_i} \mathit{cost}(\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_i)) = \sum_{u\langle U\rangle\in\mathbb{L}'_i} \mathit{cost}(\{v\langle V\rangle \in \mathbb{L}_i \mid v\langle V\rangle \preceq u\langle U\rangle\}) = \mathit{cost}(\mathbb{L}_i)$. Using invariant 4 to bound the cost of the pebbling transitions $\mathbb{L}'_{i-1} \rightsquigarrow \mathbb{L}'_i$, we get the desired result $\mathit{cost}(\mathcal{L}') \le 2 \cdot \mathit{cost}(\mathcal{L})$.

The construction is by forward induction over the moves in $\mathcal{L}$. Assume that the invariants hold for $\mathbb{L}_t$ and $\mathbb{L}'_t$.

Introduction. $\mathbb{L}_{t+1} = \mathbb{L}_t \cup v\langle pred(v)\rangle$: If $v\langle pred(v)\rangle \preceq \mathbb{L}'_t$, we set $\mathbb{L}'_{t+1} = \mathbb{L}'_t$. For the pebble subconfiguration $u\langle U\rangle \in \mathbb{L}'_t$ such that $v\langle pred(v)\rangle \preceq u\langle U\rangle$, we have $\mathit{cost}(u\langle U\rangle) \le \mathit{cost}(\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_t)) \le \mathit{cost}(\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_t \cup v\langle pred(v)\rangle))$, and for $u'\langle U'\rangle \in \mathbb{L}'_t$ distinct from $u\langle U\rangle$ nothing changes. All invariants stay true.

If $v\langle pred(v)\rangle \not\preceq \mathbb{L}'_t$, we introduce $v\langle pred(v)\rangle$ in $\mathcal{L}'$ and expand (at most three times) to get $\mathbb{L}'_{t+1} = \mathsf{canon}(\mathbb{L}'_t \cup v\langle pred(v)\rangle)$. Invariants 1 and 2 obviously hold. We claim that invariant 3 holds with respect to $\mathbb{L}_{t+1}$ instead of $\mathbb{L}_t$ for all subconfigurations in the intermediate L-configurations $\mathbb{L}'$ in the transition $\mathbb{L}'_t \rightsquigarrow \mathbb{L}'_{t+1}$ up to and including $\mathbb{L}'_{t+1} = \mathsf{canon}(\mathbb{L}'_t \cup v\langle pred(v)\rangle)$. This claim yields invariants 3 and 4 for $\mathbb{L}'_{t+1}$.

To prove the claim, observe that invariant 3 holds for $\mathbb{L}'_t \cup v\langle pred(v)\rangle$ with respect to $\mathbb{L}_{t+1} = \mathbb{L}_t \cup v\langle pred(v)\rangle$ by the induction hypothesis and the fact that $\mathsf{proj}_{v\langle pred(v)\rangle}(\mathbb{L}_t \cup v\langle pred(v)\rangle) = v\langle pred(v)\rangle$. Since $\mathbb{L}'_{t+1}$ is obtained by repeated merging of nonoverlapping subconfigurations from $\mathbb{L}'_t \cup v\langle pred(v)\rangle$, and since by induction over each such merger these subconfigurations meet the conditions in Lemma A.36, the claim follows.

Expansion. $\mathbb{L}_{t+1} = \big(\mathbb{L}_t \cup \mathsf{merge}(v_1\langle V_1\rangle, v_2\langle V_2\rangle)\big)\setminus\{v_1\langle V_1\rangle, v_2\langle V_2\rangle\}$: By induction it holds that $\mathbb{L}'_t \succeq \mathbb{L}_t \sim \mathbb{L}_{t+1}$, so there is a $u\langle U\rangle \in \mathbb{L}'_t$ such that $v_i\langle V_i\rangle \preceq u\langle U\rangle$ for $i = 1, 2$. For $u'\langle U'\rangle \in \mathbb{L}'_t$ distinct from $u\langle U\rangle$ there are no changes in the invariants, and if $\mathit{cost}(\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_{t+1})) \ge \mathit{cost}(u\langle U\rangle)$, nothing needs to be done for $u\langle U\rangle$ either, and we can set $\mathbb{L}'_{t+1} = \mathbb{L}'_t$.

It can be the case, however, that the expansion within $\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_{t+1})$ decreased the cost so that $u\langle U\rangle$ is now too expensive and invariant 3 no longer holds. If so, we implode $u\langle U\rangle$ to a minimal nonwasteful L-configuration $\mathbb{M}_u \succeq \mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_{t+1})$ and set $\mathbb{L}'_{t+1} = \big(\mathbb{L}'_t \setminus u\langle U\rangle\big) \cup \mathbb{M}_u$.

Invariants 1 and 2 are immediate. Invariant 3 follows from Lemma A.35 since $\mathbb{M}_u$ is chosen minimal. Thus, $\mathit{cost}(\mathbb{M}_u) \le \mathit{cost}(\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_{t+1}))$, and by the induction hypothesis we know that $\mathit{cost}(u\langle U\rangle) \le \mathit{cost}(\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_t))$. Using part 1 of Proposition A.20, we see that the maximal cost in the implosion sequence $\mathbb{L}'_t \rightsquigarrow \mathbb{L}'_{t+1}$ locally inside the closure $cl(u\langle U\rangle)$ is reached in the L-configuration $u\langle U\rangle \cup \mathbb{M}_u$, and using part 2 of Proposition A.20, this extra cost in the transition from $\mathbb{L}'_t$ to $\mathbb{L}'_{t+1}$ in $\mathcal{L}'$ is at most

$$\begin{aligned} \mathit{cost}(u\langle U\rangle \cup \mathbb{M}_u) &\le \mathit{cost}(u\langle U\rangle) + \mathit{cost}(\mathbb{M}_u) \\ &\le \mathit{cost}(\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_t)) + \mathit{cost}(\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_{t+1})) \\ &\le 2 \cdot \max_{i\in\{t,t+1\}} \big\{\mathit{cost}(\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_i))\big\}. \end{aligned} \tag{A.15}$$

The cost outside $cl(u\langle U\rangle)$ does not change since nothing happens there, so invariant 4 follows.

Implosion. $\mathbb{L}_{t+1} = \big(\mathbb{L}_t \setminus v\langle V\rangle\big) \cup \mathbb{M}$ for $\mathbb{M} = \big\{v_i\langle V_i\rangle \mid i \in [m]\big\}$: This case is analogous to the expansion case. By invariants 1 and 2, we know that $v\langle V\rangle$ is covered by some $u\langle U\rangle \in \mathbb{L}'_t$. Nothing happens for $u'\langle U'\rangle \in \mathbb{L}'_t$ distinct from $u\langle U\rangle$, so we can again concentrate on what is going on inside $cl(u\langle U\rangle)$.

If $u\langle U\rangle$ is too expensive with respect to $\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_{t+1})$ so that invariant 3 fails, we make a nonwasteful implosion of $u\langle U\rangle$ to an L-configuration $\mathbb{M}_u$ with $u\langle U\rangle \succeq \mathbb{M}_u \succeq \mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_{t+1})$ and set $\mathbb{L}'_{t+1} = \big(\mathbb{L}'_t \setminus u\langle U\rangle\big) \cup \mathbb{M}_u$. By part 1 of Proposition A.20, a *lower* bound for the cost locally of the pebbling sequence $\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_t) \rightsquigarrow \mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_{t+1})$ in $\mathcal{L}$ is $\max_{i\in\{t,t+1\}}\big\{\mathit{cost}(\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_{i+1}))\big\}$. Using Lemma A.35 and parts 1 and 2 of Proposition A.20, we can upper-bound the pebbling cost locally inside $cl(u\langle U\rangle)$ in $\mathcal{L}'$ in terms of this local cost in $\mathcal{L}$ by

$$\begin{aligned}\mathit{cost}(u\langle U\rangle \cup \mathbb{M}_u) &\leq \mathit{cost}(u\langle U\rangle) + \mathit{cost}(\mathbb{M}_u)\\ &\leq 2\cdot \max_{i\in\{t,t+1\}}\big\{\mathit{cost}(\mathsf{proj}_{u\langle U\rangle}(\mathbb{L}_i))\big\},\end{aligned} \tag{A.16}$$

which yields invariants 1–4.

Going through the moves in $\mathcal{L} = \{\mathbb{L}_0,\dots,\mathbb{L}_\tau\}$, this construction yields an L-pebbling $\mathcal{L}'$ without wasteful implosions such that $\mathbb{L}'_{\tau'} \succeq \mathbb{L}_\tau$ and $\mathit{cost}(\mathcal{L}') \leq 2\cdot\mathit{cost}(\mathcal{L})$. ∎

Thereby, the proof of Lemma A.2 as outlined in section A.1 is complete, and Theorem 5.4 follows. We conclude the appendix by restating the lemma and writing out the proof in full for completeness.

LEMMA A.2 (restated). *Suppose that $\mathcal{L}$ is a complete L-pebbling of a complete binary tree T. Then from $\mathcal{L}$ we can construct a complete L-pebbling $\mathcal{L}^*$ of T without reversals such that $\mathit{cost}(\mathcal{L}^*) = \mathrm{O}(\mathit{cost}(\mathcal{L}))$.*

Proof. Let $\mathcal{L}$ be an arbitrary complete L-pebbling of T. Without loss of generality, we can assume that $\mathcal{L}$ is nonredundant in the sense of Lemma A.5. By Lemma A.11, we can also assume that $\mathcal{L}$ contains only simple L-configurations. This sets the stage for applying the technical machinery developed in sections A.4–A.8.

First, using Lemma A.26, we transform $\mathcal{L}$ into a nonoverlapping L-pebbling $\mathcal{L}'$ with $\mathit{cost}(\mathcal{L}') \leq \mathit{cost}(\mathcal{L})$. If $\mathcal{L}'$ contains wasteful implosions, we then let Lemma A.37 provide us with a nonwasteful complete L-pebbling $\mathcal{L}''$ such that $\mathit{cost}(\mathcal{L}'') \leq 2\cdot\mathit{cost}(\mathcal{L}')$. But for such an L-pebbling, Lemma A.32 allows us to project away all implosion moves without increasing the pebbling cost, so we finally get a reversal-free complete L-pebbling $\mathcal{L}^*$ of T with $\mathit{cost}(\mathcal{L}^*) \leq \mathit{cost}(\mathcal{L}'') \leq 2\cdot\mathit{cost}(\mathcal{L}') \leq 2\cdot\mathit{cost}(\mathcal{L})$. This proves the lemma. ∎

Note added in proof. Very recently, we have been able to obtain an exponential improvement of the results in the current paper. This has been achieved by studying pyramid graphs and proving that the black-white pebbling price is a lower bound for the clause space of refuting pebbling contradictions over such graphs. The general structure of the proof as outlined in section 2 of the current paper is the same, although the technical details are quite different (as they have to be, given Lemma 5.3).

The formal results, to appear in [42], can be stated as follows.

IMPROVED THEOREM 1.1. *The clause space of refuting pebbling contradictions over pyramid graphs of height h in resolution grows as $\Theta(h)$, provided that the number of variables per vertex in the pebbling contradictions is at least 2.*

IMPROVED COROLLARY 1.2. *For all $k \geq 4$, there is a family $\{F_n\}_{n=1}^{\infty}$ of k-CNF formulas of size $\Theta(n)$ that can be refuted in resolution in length $L(F_n \vdash 0) = \mathrm{O}(n)$ and width $W(F_n \vdash 0) = \mathrm{O}(1)$, but require clause space $Sp(F_n \vdash 0) = \Theta(\sqrt{n})$.*

Acknowledgments. I am grateful to my advisor Johan Håstad for patiently giving feedback during this work. Also, I would like to thank Douglas Wikström for interesting discussions about theoretical computer science in general and pebble games in particular, and Joel Brynielsson for gently introducing me to the mysteries of MetaPost. Finally, a special thanks to the two anonymous referees for many insightful comments that helped to improve the presentation considerably, and to Dieter van Melkebeek for very helpful discussions during the revision process.

REFERENCES

[1] R. Aharoni and N. Linial, *Minimal non-two-colorable hypergraphs and minimal unsatisfiable formulas*, J. Combin. Theory Ser. A, 43 (1986), pp. 196–204.

[2] M. Alekhnovich, *Lower bounds for k-DNF resolution on random 3-CNFs*, in Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC '05), ACM, New York, 2005, pp. 251–256.

[3] M. Alekhnovich, E. Ben-Sasson, A. A. Razborov, and A. Wigderson, *Space complexity in propositional calculus*, SIAM J. Comput., 31 (2002), pp. 1184–1211.

[4] M. Alekhnovich, J. Johannsen, T. Pitassi, and A. Urquhart, *An exponential separation between regular and general resolution*, in Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC '02), ACM, New York, 2002, pp. 448–456.

[5] A. Atserias and M. L. Bonet, *On the automatizability of resolution and related propositional proof systems*, Inform. and Comput., 189 (2004), pp. 182–201.

[6] A. Atserias, M. L. Bonet, and J. L. Esteban, *Lower bounds for the weak pigeonhole principle and random formulas beyond resolution*, Inform. and Comput., 176 (2002), pp. 136–152.

[7] A. Atserias and V. Dalmau, *A combinatorical characterization of resolution width*, in Proceedings of the 18th IEEE Annual Conference on Computational Complexity (CCC '03), IEEE Computer Society, Los Alamitos, CA, 2003, pp. 239–247.

[8] S. Baumer, J. L. Esteban, and J. Torán, *Minimally unsatisfiable CNF formulas*, Bull. Eur. Assoc. Theor. Comput. Sci. EATCS, 74 (2001), pp. 190–192.

[9] P. Beame, *Proof complexity*, in Computational Complexity Theory, IAS/Park City Mathematics Series 10, S. Rudich and A. Wigderson, eds., AMS, Providence, RI, 2004, pp. 199–246.

[10] P. Beame, R. Karp, T. Pitassi, and M. Saks, *The efficiency of resolution and Davis–Putnam procedures*, SIAM J. Comput., 31 (2002), pp. 1048–1075.

[11] P. Beame and T. Pitassi, *Simplified and improved resolution lower bounds*, in Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science (FOCS '96), IEEE Computer Society, Los Alamitos, CA, 1996, pp. 274–282.

[12] P. Beame and T. Pitassi, *Propositional proof complexity: Past, present, and future*, Bull. Eur. Assoc. Theor. Comput. Sci. EATCS, 65 (1998), pp. 66–89.

[13] E. Ben-Sasson, *Size space tradeoffs for resolution*, in Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC '02), ACM, New York, 2002, pp. 457–464.

[14] E. Ben-Sasson and N. Galesi, *Space complexity of random formulae in resolution*, Random Structures Algorithms, 23 (2003), pp. 92–109.

[15] E. Ben-Sasson, R. Impagliazzo, and A. Wigderson, *Near optimal separation of treelike and general resolution*, Combinatorica, 24 (2004), pp. 585–603.

[16] E. Ben-Sasson and A. Wigderson, *Short proofs are narrow—resolution made simple*, J. ACM, 48 (2001), pp. 149–169.

[17] A. Blake, *Canonical Expressions in Boolean Algebra*, Ph.D. thesis, University of Chicago, Chicago, IL, 1937.

[18] M. L. Bonet, J. L. Esteban, N. Galesi, and J. Johannsen, *On the relative complexity of resolution refinements and cutting planes proof systems*, SIAM J. Comput., 30 (2000), pp. 1462–1484.

[19] M. L. Bonet and N. Galesi, *Optimality of size-width tradeoffs for resolution*, Comput. Complexity, 10 (2001), pp. 261–276.

[20] J. Buresh-Oppenheim and T. Pitassi, *The complexity of resolution refinements*, in Proceedings of the 18th IEEE Symposium on Logic in Computer Science (LICS 03), IEEE Computer Society, Los Alamitos, CA, 2003, pp. 138–147.

[21] S. R. Buss, ed., *Handbook of Proof Theory*, Elsevier Science, Amsterdam, 1998.

[22] S. R. Buss and G. Turán, *Resolution proofs of generalized pigeonhole principles*, Theoret. Comput. Sci., 62 (1988), pp. 311–317.

[23] V. Chvátal and E. Szemerédi, *Many hard examples for resolution*, J. ACM, 35 (1988), pp. 759–768.

[24] S. A. Cook and R. Reckhow, *The relative efficiency of propositional proof systems*, J. Symbolic Logic, 44 (1979), pp. 36–50.

[25] S. A. Cook and R. Sethi, *Storage requirements for deterministic polynomial time recognizable languages*, J. Comput. System Sci., 13 (1976), pp. 25–37.
[26] M. Davis, G. Logemann, and D. Loveland, *A machine program for theorem proving*, Commun. ACM, 5 (1962), pp. 394–397.
[27] M. Davis and H. Putnam, *A computing procedure for quantification theory*, J. ACM, 7 (1960), pp. 201–215.
[28] J. L. Esteban, N. Galesi, and J. Messner, *On the complexity of resolution with bounded conjunctions*, Theoret. Comput. Sci., 321 (2004), pp. 347–370.
[29] J. L. Esteban and J. Torán, *Space bounds for resolution*, Inform. and Comput., 171 (2001), pp. 84–97.
[30] J. L. Esteban and J. Torán, *A combinatorial characterization of treelike resolution space*, Inform. Process. Lett., 87 (2003), pp. 295–300.
[31] Z. Galil, *On resolution with clauses of bounded size*, SIAM J. Comput., 6 (1977), pp. 444–459.
[32] J. R. Gilbert and R. E. Tarjan, *Variations of a Pebble Game on Graphs*, Tech. report STAN-CS-78-661, Stanford University, Stanford, CA, 1978, available at http://infolab.stanford.edu/TR/CS-TR-78-661.html.
[33] A. Haken, *The intractability of resolution*, Theoret. Comput. Sci., 39 (1985), pp. 297–308.
[34] J. Hopcroft, W. Paul, and L. Valiant, *On time versus space*, J. ACM, 24 (1977), pp. 332–337.
[35] M. M. Klawe, *A tight bound for black and white pebbles on the pyramid*, J. ACM, 32 (1985), pp. 218–228.
[36] H. Kleine Büning and T. Letterman, *Propositional Logic: Deduction and Algorithms*, Cambridge University Press, Cambridge, UK, 1999.
[37] J. Krajíček, *On the weak pigeonhole principle*, Fund. Math., 170 (2001), pp. 123–140.
[38] O. Kullmann, *An application of matroid theory to the SAT problem*, in Proceedings of the 15th Annual IEEE Conference on Computational Complexity (CCC '00), IEEE Computer Society, Los Alamitos, CA, 2000, pp. 116–124.
[39] T. Lengauer and R. E. Tarjan, *Upper and lower bounds on time-space tradeoffs*, in Proceedings of the 11th Annual ACM Symposium on Theory of Computing (STOC '79), ACM, New York, 1979, pp. 262–277.
[40] T. Lengauer and R. E. Tarjan, *The space complexity of pebble games on trees*, Inform. Process. Lett., 10 (1980), pp. 184–188.
[41] J. Nordström, *Narrow Proofs May Be Spacious: Separating Space and Width in Resolution*, Tech. report TR05-066, Revision 02, Electronic Colloquium on Computational Complexity (ECCC), 2005.
[42] J. Nordström and J. Håstad, *Towards an optimal separation of space and length in resolution (extended abstract)*, in Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC '08), ACM, New York, 2008, to appear.
[43] N. Pippenger, *Pebbling*, Tech. report RC8258, IBM Watson Research Center, 1980, appeared in Proceedings of the 5th IBM Symposium on Mathematical Foundations of Computer Science, Japan, 1980, pp. 1–19.
[44] R. Raz, *Resolution lower bounds for the weak pigeonhole principle*, J. ACM, 51 (2004), pp. 115–138.
[45] R. Raz and P. McKenzie, *Separation of the monotone NC hierarchy*, Combinatorica, 19 (1999), pp. 403–435.
[46] A. A. Razborov, *Pseudorandom Generators Hard for k-DNF Resolution and Polynomial Calculus Resolution*, manuscript, 2003, available at http://www.mi.ras.ru/~razborov/.
[47] A. A. Razborov, *Resolution lower bounds for the weak functional pigeonhole principle*, Theoret. Comput. Sci., 1 (2003), pp. 233–243.
[48] A. A. Razborov, *Resolution lower bounds for perfect matching principles*, J. Comput. System Sci., 69 (2004), pp. 3–27.
[49] J. A. Robinson, *A machine-oriented logic based on the resolution principle*, J. ACM, 12 (1965), pp. 23–41.
[50] N. Segerlind, S. R. Buss, and R. Impagliazzo, *A switching lemma for small restrictions and lower bounds for k-DNF resolution*, SIAM J. Comput., 33 (2004), pp. 1171–1200.
[51] J. Torán, *Lower bounds for space in resolution*, in Proceedings of the 13th International Workshop on Computer Science Logic (CSL '99), Lecture Notes in Comput. Sci. 1683, Springer-Verlag, New York, 1999, pp. 362–373.
[52] J. Torán, *Space and width in propositional resolution*, Bull. Eur. Assoc. Theor. Comput. Sci. EATCS, 83 (2004), pp. 86–104.

[53] G. Tseitin, *On the complexity of derivation in propositional calculus*, in Structures in Constructive Mathematics and Mathematical Logic, Part II, A. O. Silenko, ed., Consultants Bureau, New York, London, 1968, pp. 115–125.

[54] A. Urquhart, *Hard examples for resolution*, J. ACM, 34 (1987), pp. 209–219.

[55] A. Urquhart, *The complexity of propositional proofs*, Bull. Symbolic Logic, 1 (1995), pp. 425–467.