comput. complex. (2021) 30:7

1016-3328/21/010001-37
published online May 1, 2021
https://doi.org/10.1007/s00037-021-00206-1

computational complexity

REVERSIBLE PEBBLE GAMES AND THE RELATION BETWEEN TREE-LIKE AND GENERAL RESOLUTION SPACE

Jacobo Torán and Florian Wörz

Abstract. We show a new connection between the clause space measure in tree-like resolution and the reversible pebble game on graphs. Using this connection, we provide several formula classes for which there is a logarithmic factor separation between the clause space complexity measure in tree-like and general resolution. We also provide upper bounds for tree-like resolution clause space in terms of general resolution clause and variable space. In particular, we show that for any formula F, its tree-like resolution clause space is upper bounded by $\text{space}(\pi)\log\big(\text{time}(\pi)\big)$, where π is any general resolution refutation of F. This holds considering as $\text{space}(\pi)$ the clause space of the refutation as well as considering its variable space. For the concrete case of Tseitin formulas, we are able to improve this bound to the optimal bound $\text{space}(\pi)\log n$, where n is the number of vertices of the corresponding graph.

Keywords. Proof complexity, Resolution, Tree-like resolution, Pebble game, Reversible pebbling, Prover–Delayer game, Raz–McKenzie game, Clause space, Variable space.

Subject classification. Primary 03F20; Secondary 68Q17.

1. Introduction

Resolution is one of the best-studied systems for refuting unsatisfiable propositional formulas. This is due to its theoretical sim-

plicity, as well as its practical importance since it is the proof system at the root of many modern SAT solvers. Several complexity measures for the analysis of resolution refutations have been used in the last decades. In this paper, we will mainly concentrate on space bounds, which measure the amount of memory that is needed in a resolution refutation. Intuitively, the clause space (CS) measures the number of clauses required simultaneously in a refutation, while the variable space (VS) measures the maximum number of distinct variables kept simultaneously in memory during this process. Experimental results in Ansótegui *et al.* (2008) and Järvisalo *et al.* (2012) have shown that space measures for resolution correlate well with the hardness of refuting unsatisfiable formulas with SAT solvers in practice.

Tree-like resolution is a restricted kind of resolution that is especially important since the original DPLL algorithm of Davis & Putnam (1960) and Davis *et al.* (1962) on which many SAT solvers are based, is equivalent to this restriction of the resolution system. Contrary to general resolution, in the tree-like case, if a clause is needed more than once in a refutation, it has to be rederived each time. It is known from Bonet *et al.* (1998) and Ben-Sasson *et al.* (2004) that general resolution can be exponentially more efficient than tree-like resolution in terms of the number of clauses in a refutation. In Ben-Sasson *et al.* (2004), the authors give an almost optimal separation between general and tree-like resolution. They show that for each positive integer n, there are unsatisfiable formulas in $\mathrm{O}(n)$ variables that have resolution refutations of length L, linear in n, but for which any tree-like resolution refutation of the formula requires length $\exp\left(\Omega(\frac{L}{\log L})\right)$. They also give an almost matching upper bound of $\exp\left(\mathrm{O}\left(\frac{L \log\log L}{\log L}\right)\right)$ for the tree-like resolution length of any formula that can be refuted in length L by general resolution.

In this paper, we study clause space separations between general and tree-like resolution. Space separations are much more modest than the ones for length. It is known from Esteban & Torán (2001) that all space measures considered in this paper for a formula with n variables are between constant and $n + 2$. Also, it is not hard to see that variable space coincides in general and

tree-like resolution. Therefore, we only consider the clause space measure for the case of tree-like resolution.

The first clause space separation between general and tree-like resolution was given in Esteban & Torán (2003). A family of formulas $(F_n)_{n=1}^{\infty}$ was presented which requires tree-resolution clause space s_n but has a general resolution refutation in clause space $c \cdot s_n$, for some constant $c < 1$, where s_n is logarithmic in the number of variables of the formulas. More recently, in Järvisalo *et al.* (2012), a family of formulas $(F_n)_{n=1}^{\infty}$ is given with $\mathrm{O}(n)$ variables that can be refuted by general resolution in constant clause space but requires $\Theta(\log n)$ tree-like resolution clause space, thus showing that both measures are fundamentally different.

In this paper, we present a systematic study of tree-like resolution clause space providing upper bounds for this measure, which show that the logarithmic factor in the separation of Järvisalo *et al.* (2012) as well as in other separations provided here are basically optimal. Our main tools are several versions of pebbling games played on graphs, which have been extensively used in the past for analyzing different computation models and in particular for analyzing proof systems (see Nordström (2015) for an excellent survey). Intuitively, the idea of the pebble games is to measure the number of pebbles needed by a single player in order to place a pebble on the sink of a directed acyclic graph following certain rules. Black pebbles can only be placed on a vertex if it is a source or if all its direct predecessors already have a pebble, but these pebbles can be removed at any time. White pebbles (modeling non-determinism) can be placed on any vertex at any time but can only be removed if all its direct predecessors contain a pebble. In the reversible pebble game, pebbles can only be placed or removed from a vertex if all the direct predecessors of the vertex contain a pebble. Based on the pebble game, a class of contradictory formulas, called pebbling formulas, was introduced by Ben-Sasson & Wigderson (2001). These formulas have been extremely useful for analyzing several proof systems. The reason for this is that some of the pebbling properties of the underlying graphs can be translated into parameters for the complexity of their corresponding pebbling contradictions. Known results of pebbling can therefore be trans-

lated into proof complexity results. Chan (2013) proved the first relationship between reversible pebbling and proof complexity. He showed that the resolution depth of a pebbling formula corresponds to the reversible pebbling number of its underlying graph.

The formulas used for the separation between general and tree-like resolution clause space in Esteban & Torán (2003) are pebbling formulas. An examination of this result shows that it relies on the fact that the graphs on which the formulas are based have a black-white pebbling price that is smaller than their black pebbling number. With this observation and using existing separation results for pebble games, the separation in Esteban & Torán (2003) can be significantly improved. On the one hand, in Ben-Sasson *et al.* (2004) the authors implicitly show that for any graph G the tree-like clause space of the pebbling contradiction associated with G is at least as large as the black pebbling number of the graph. On the other hand, Nordström (2012) shows that for most of the graph examples existing in the literature with a difference between their black and black-white pebbling numbers, the resolution clause space of a version of the pebbling contradictions based on the graphs (more precisely, the second degree XORification of the pebbling contradiction) is upper bounded by the black-white pebbling number of the graphs. Putting these two facts together, it follows that there are unsatisfiable formulas that have resolution clause space $O(s)$ (logarithmic in the number of variables of the formulas) while their tree-like resolution clause space is lower bounded by $\Omega(s^2)$. This is the largest separation that can be obtained using this method since it is known from Meyer auf der Heide (1981) that the difference between the black and black-white pebbling number of any graph is at most quadratic and can therefore not explain the logarithmic factor in the separation of Järvisalo *et al.* (2012) where the (pebbling) formulas have constant general resolution clause space.

1.1. Our contributions.

New connection of Tree-CS and reversible pebbling. We prove new connections between tree-like resolution clause space and the reversible pebble game. We show that for any graph G, the tree-like resolution clause space of a certain kind of pebbling

contradiction (the second degree XORification of the pebbling contradiction) coincides asymptotically with the reversible pebbling number of the graph:

$$\mathsf{Rev}(G) \leq \text{Tree-CS}\big(\text{Peb}_G[\oplus_2] \vdash \square\big) \leq 2 \cdot \mathsf{Rev}(G) + 4.$$

More interestingly, we show a close relation between the tree-like space of any unsatisfiable CNF formula F with n variables, and the minimum reversible pebbling price of a resolution refutation G_π for F, not necessarily a tree-like refutation:

$$\text{Tree-CS}(F \vdash \square) \lesssim \min_{\pi : F \vdash \square} \mathsf{Rev}(G_\pi) \lesssim \text{Tree-CS}(F \vdash \square) \cdot \log n.$$

This result can be seen as a loose analogue of the relation between general clause space and black pebbling and adds one more connection to the rich set of interrelations between pebbling and resolution surveyed in Nordström (2015).

Separations between Tree-CS and CS. Using these connections to reversible pebbling and known pebbling results by Chan *et al.* (2015), we show that there are families of pebbling formulas $(F_n)_{n=1}^{\infty}$ with $\mathrm{O}(n)$ variables that have general clause space $\mathrm{O}(s)$ but tree-like resolution clause space $\Omega(s \log n)$ for any function s smaller than $n^{1/2-\varepsilon}$. This separation—as well as the one in Järvisalo *et al.* (2012)—is not far from optimal for this kind of formulas since we also show that for any pebbling formula

$$\text{Tree-CS}\big(\text{Peb}_G[\oplus_2] \vdash \square\big) = \mathrm{O}\left(\min_{\mathcal{P}} \big(\mathsf{space}(\mathcal{P}) \cdot \log \mathsf{time}(\mathcal{P})\big)\right),$$

where $\mathcal{P}$ is a black pebbling of the underlying graph G of the pebbling formula. This means that for graphs of size n where the smallest black pebbling space is achieved in a one-shot pebbling strategy, that is, a strategy in which every vertex in the graph is pebbled at most once, the $\log n$ factor in the separation is optimal and the only room for improvement is obtainable with graph families in which the space-optimal black pebbling is not one-shot. It could be possible that there exists such a family, for which the $\log n$ separation factor can be improved to a $\log \mathsf{time}(\mathcal{P})$ factor. We provide, however, a family of graphs (based on the Carlson–Savage graphs)

for which the minimum pebbling space is obtained in a strategy that is not one-shot, but for which the clause space separation between general and tree-like resolution is also only a $\log n$ factor. We thus conjecture that this is indeed optimal and that this separation cannot be improved for other graph classes. We remark that this question is closely related to proving optimal upper bounds for reversible pebbling in terms of black pebbling. Another motivation for analysing this graph family based on the Carlson–Savage graphs is to increase the set of examples of formulas with concrete tree-like resolution clause space bounds that can be used for the testing of SAT solvers, as done for example in Järvisalo *et al.* (2012).

Upper bounds for Tree-CS for general formulas. Additionally, we prove upper bounds on the tree-like clause space for any unsatisfiable CNF formula F in terms of the variable space and clause space for general resolution of the formula. We use the *amortized space measures* for resolution introduced in Razborov (2018) that penalize configurational proofs for being unreasonably long. We show that tree-like resolution clause space is upper bounded by amortized variable space as well as by amortized clause space:

$$\begin{aligned} &\text{Tree-CS}(F \vdash \square) \lesssim \text{VS}^*(F \vdash \square), \quad \text{and} \\ &\text{Tree-CS}(F \vdash \square) \lesssim \text{CS}^*(F \vdash \square). \end{aligned}$$

The first inequality follows from known results and provides a new bound for clause space in terms of variable space.

Optimal separations for Tseitin formulas. Finally, we give optimal separations for the clause space in tree-like resolution for the class of Tseitin formulas. We show that for any graph G with n vertices and odd marking χ, the inequalities

$$\begin{aligned} &\text{Tree-CS}\big(\text{Ts}(G,\chi) \vdash \square\big) \lesssim \text{CS}\big(\text{Ts}(G,\chi) \vdash \square\big) \cdot \log n, \quad \text{and} \\ &\text{Tree-CS}\big(\text{Ts}(G,\chi) \vdash \square\big) \lesssim \text{VS}\big(\text{Ts}(G,\chi) \vdash \square\big) \cdot \log n \end{aligned}$$

hold, thus improving the previously discussed upper bounds from logarithmic in the resolution length down to a $\log n$ factor. We also provide a class of formulas with a matching clause space separation showing that these upper bounds are optimal.

1.2. Outline of this paper. The rest of this paper is organized as follows. In the Preliminaries, pebble games, resolution complexity measures, and all formula families needed are introduced. We recall the combinatorial characterization for tree-like clause space in resolution through the Prover–Delayer game and introduce the Raz–McKenzie game. In Section 3, we prove separations between tree-like and general resolution clause space for pebbling formulas. Further upper bounds for tree-like resolution clause space, now for general formulas, are proven in Section 4. Finally, in Section 5, optimal separations between general and tree-like resolution clause space are provided for Tseitin formulas. We conclude in Section 6 with some conclusions and open problems.

2. Preliminaries

For a positive integer n, we write $[n]$ to denote the set of integers $\{1, 2, \ldots, n\}$. The base of all logarithms in this paper is 2. The *size of a graph* is the number of vertices of the graph. Given a directed acyclic graph (*DAG*) $G = (V, E)$, we say that a vertex u is a *direct predecessor* of a vertex v, if there exists a directed edge from u to v. We denote by $\mathrm{pred}_G(v)$ the *set of all direct predecessors* of v in G. The *maximal in-degree* of a graph G is defined to be $\max_{v \in V} |\mathrm{pred}_G(v)|$. A vertex in a DAG with no incoming edges is called a *source* and one with no outgoing edges is called a *sink*.

2.1. Pebble games. Black pebbling was first mentioned implicitly in Paterson & Hewitt (1970) and has been studied extensively during the 1980s. Note, that there exist several variants of the pebble game in the literature. In this paper, we focus on the variant without *sliding* and requiring the sink of the graph to be pebbled at the end. For differences between these variants, we refer to the survey Nordström (2015), from which we borrowed most of our notation. For the following definitions, let $G = (V, E)$ be a DAG with a unique sink vertex z.

Definition 2.1 (Black pebble game). *The* black pebble game *on G is the following one-player game: At any time i of the game, we have a* pebble configuration $\mathbb{P}_i$, *where $\mathbb{P}_i \subseteq V$ is the set of*

black pebbles. A pebble configuration $\mathbb{P}_{i-1}$ *can be changed to* $\mathbb{P}_i$ *by applying exactly one of the following rules:*

Black pebble placement on v: *If all direct predecessors of an empty vertex* v *have pebbles on them, a black pebble may be placed on* v*. More formally, letting* $\mathbb{P}_i = \mathbb{P}_{i-1} \cup \{v\}$ *is allowed if* $v \notin \mathbb{P}_{i-1}$ *and* $\mathrm{pred}_G(v) \subseteq \mathbb{P}_{i-1}$*. In particular, a black pebble can always be placed on an empty source vertex.*

Black pebble removal from v: *A black pebble may be removed from any vertex at any time. Formally, if* $v \in \mathbb{P}_{i-1}$*, then we can set* $\mathbb{P}_i = \mathbb{P}_{i-1} \backslash \{v\}$*.*

A black pebbling *of* G *is a sequence of pebble configurations* $\mathcal{P} = (\mathbb{P}_0, \mathbb{P}_1, \ldots, \mathbb{P}_t)$ *such that* $\mathbb{P}_0 = \varnothing$*,* $\mathbb{P}_t = \{z\}$*, and for all* $i \in [t]$ *it holds that* $\mathbb{P}_i$ *can be obtained from* $\mathbb{P}_{i-1}$ *by applying exactly one of the above-stated rules. A pebbling is called* one-shot *if each* $v \in V$ *is pebbled at most once.*

Finally, we mention the reversible pebble game introduced by Bennett (1989). In the reversible pebble game, the moves performed in reverse order should also constitute a legal black pebbling, which means that the rules for pebble placements and removals have to become symmetric.

Definition 2.2 (Reversible pebble game). *The* reversible pebble game *on* G *is the following one-player game: At any time* i *of the game, we have a pebble configuration* $\mathbb{P}_i \subseteq V$*. A pebble configuration* $\mathbb{P}_{i-1}$ *can be changed to* $\mathbb{P}_i$ *by applying exactly one of the following rules:*

Pebble placement on v: *If all direct predecessors of an empty vertex* v *have pebbles on them, a pebble may be placed on* v*. More formally, letting* $\mathbb{P}_i = \mathbb{P}_{i-1} \cup \{v\}$ *is allowed if* $v \notin \mathbb{P}_{i-1}$ *and* $\mathrm{pred}_G(v) \subseteq \mathbb{P}_{i-1}$*. In particular, a pebble can always be placed on an empty source vertex.*

Reversible pebble removal from v: *If all direct predecessors of a pebbled vertex* v *have pebbles on them, the pebble on* v *may be removed. Formally, letting* $\mathbb{P}_i = \mathbb{P}_{i-1} \backslash \{v\}$ *is allowed*

if $v \in \mathbb{P}_{i-1}$ and $\mathrm{pred}_G(v) \subseteq \mathbb{P}_{i-1}$. In particular, a pebble can always be removed from a source vertex.

A reversible pebbling *of G is a sequence of pebble configurations $\mathcal{P} = (\mathbb{P}_0, \mathbb{P}_1, \ldots, \mathbb{P}_t)$ such that $\mathbb{P}_0 = \varnothing$, $\mathbb{P}_t = \{z\}$, and for all $i \in [t]$ it holds that $\mathbb{P}_i$ can be obtained from $\mathbb{P}_{i-1}$ by applying exactly one of the above-stated rules.*

DEFINITION 2.3 (Pebbling time, space, and price). *The* time *of a pebbling $\mathcal{P} = (\mathbb{P}_0, \mathbb{P}_1, \ldots, \mathbb{P}_t)$ is $\mathsf{time}(\mathcal{P}) := t$ and the* space *of it is $\mathsf{space}(\mathcal{P}) := \max_{i \in [t]} |\mathbb{P}_i|$. The* (black) pebbling price *(or* number*) of G, denoted by $\mathsf{Black}(G)$, is the minimum space of any black pebbling of G, whereas the* reversible pebbling price *of G, which we will denote by $\mathsf{Rev}(G)$, is the minimum space of any reversible pebbling of G.*

2.2. Resolution. A *literal* over a Boolean variable x is either x itself (also denoted as x^1) or its negation $\overline{x}$ (also denoted as x^0). A *clause* $C = a_1 \vee \cdots \vee a_\ell$ is a (possibly empty) disjunction of literals a_i over pairwise disjoint variables. The *set of variables occurring in a clause* C will be denoted by $\mathrm{Vars}(C)$. A clause C is called *unit* or *unitary* if $|\mathrm{Vars}(C)| = 1$. We let $\square$ denote the contradictory *empty clause* (the clause without any literals). A *CNF formula* $F = C_1 \wedge \cdots \wedge C_m$ is a conjunction of clauses. It is often advantageous to think of clauses and CNF formulas as sets. The notion of the set of variables in a clause is extended to CNF formulas by taking unions. A CNF formula is a *k-CNF*, if all clauses in it have at most k literals. An *assignment/restriction* α for a CNF formula F is a function that maps some subset of $\mathrm{Vars}(F)$ to $\{0,1\}$. It is applied to F, which we denote by $F{\restriction_\alpha}$, by replacing variables by the truth values specified for them in α and simplifying the resulting expression.

The standard definition of a *resolution derivation* of a clause D from a CNF formula F (denoted by $\pi : F \vdash D$) is an ordered sequence of clauses $\pi = (C_1, \ldots, C_t)$ such that $C_t = D$, and each clause C_i, for $i \in [t]$, is either an *axiom clause* $C_i \in F$ or is derived from clauses C_j and C_k with $j, k < i$ by the *resolution rule*

$$\frac{B \vee x \qquad C \vee \overline{x}}{B \vee C}. \tag{2.4}$$

In the resolution rule (2.4), we call $B \vee x$ and $C \vee \overline{x}$ the *parents* and $B \vee C$ the *resolvent*. A derivation $\pi : F \vdash \square$ of the empty clause from an unsatisfiable CNF formula F is called *refutation*. Note that resolution is a sound and complete proof system for unsatisfiable formulas in CNF.

To study space in resolution, we consider the following definitions of the resolution proof system from Alekhnovich *et al.* (2002); Esteban & Torán (2001).

Definition 2.5 (Configuration-style resolution). *A* resolution refutation $\pi : F \vdash \square$ *of an unsatisfiable CNF formula* F *is an ordered sequence of* memory configurations *(sets of clauses)* $\pi = (\mathbb{M}_0, \ldots, \mathbb{M}_t)$ *such that* $\mathbb{M}_0 = \varnothing$, $\square \in \mathbb{M}_t$ *and for each* $i \in [t]$, *the configuration* $\mathbb{M}_i$ *is obtained from* $\mathbb{M}_{i-1}$ *by applying exactly one of the following rules:*

Axiom Download: $\mathbb{M}_i = \mathbb{M}_{i-1} \cup \{C\}$ *for some* axiom $C \in F$.

Erasure: $\mathbb{M}_i = \mathbb{M}_{i-1} \backslash \{C\}$ *for some* $C \in \mathbb{M}_{i-1}$.

Inference: $\mathbb{M}_i = \mathbb{M}_{i-1} \cup \{D\}$ *for some resolvent* D *inferred from clauses* $C_1, C_2 \in \mathbb{M}_i$ *by the resolution rule* (2.4).

In Esteban & Torán (2001), a proof π *was defined to be* tree-like, *if we replace the inference rule with the following rule:*

Tree-like Inference: $\mathbb{M}_i = \big(\mathbb{M}_{i-1} \cup \{D\}\big) \backslash \{C_1, C_2\}$ *for some resolvent* D *inferred from* $C_1, C_2 \in \mathbb{M}_i$ *by the resolution rule* (2.4), *i. e., we immediately delete both parent clauses.*

To every configurational refutation π, we can associate a *refutation-DAG* G_π, with the downloaded or inferred clauses of the refutation labeling the vertices of the DAG and with edges from the parents to the resolvent for each application of the resolution rule (2.4). There might be several different derivations of a clause C during the course of the refutation, 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 (Nordström 2015). Using this representation, if π is tree-like, then G_π is a tree.

DEFINITION 2.6 (Complexity measures for resolution). *The* length *of a refutation* $\pi = (\mathbb{M}_0, \dots, \mathbb{M}_t)$ *is defined to be* $\mathrm{L}(\pi) := t$.

The depth $\mathrm{Depth}(\pi)$ *of a refutation* π *is the length of the longest path in the underlying refutation DAG* G_π.

The clause space *of a memory configuration* $\mathbb{M}$ *is defined as* $\mathrm{CS}(\mathbb{M}) := |\mathbb{M}|$, *i. e., the number of clauses in* $\mathbb{M}$. *The* variable space *of a memory configuration* $\mathbb{M}$ *is defined as*

$$\mathrm{VS}(\mathbb{M}) := \left| \bigcup_{C \in \mathbb{M}} \mathrm{Vars}(C) \right|,$$

i. e., the number of distinct variables mentioned in $\mathbb{M}$.

The clause space of a refutation $\pi = (\mathbb{M}_0, \dots, \mathbb{M}_t)$ *is defined by* $\mathrm{CS}(\pi) := \max_{i \in [t]} \mathrm{CS}(\mathbb{M}_i)$ *and its variable space by* $\mathrm{VS}(\pi) := \max_{i \in [t]} \mathrm{VS}(\mathbb{M}_i)$.

For a complexity measure $\mathscr{C} \in \{\mathrm{L}, \mathrm{Depth}, \mathrm{CS}, \mathrm{VS}\}$, *by taking the minimum over all refutations of a formula* F, *we define*

$$\mathscr{C}(F \vdash \square) := \min_{\pi : F \vdash \square} \mathscr{C}(\pi)$$

as the length, depth, clause space and variable space of refuting F *in resolution, respectively. We further define* $\text{Tree-CS}(F \vdash \square) := \min_{\pi' : F \vdash \square} \mathrm{CS}(\pi')$, *where the minimum is taken over all tree-like refutations* π' *of the formula* F.

REMARK 2.7. *In some publications, the authors allow for subsets of the previous memory configuration to be erased. We will not allow this, since our version is more suitable when working with pebbling. Note that not allowing subset-erasures can at most double the number of configurations in a refutation.*

Also note, that in the literature, the length of a proof π *is sometimes defined to be the total number of axiom downloads and inferences made in* π, *i. e., the total number of clauses counted with repetitions. We, however, also consider the amount of erasure steps, since this is more natural when working with pebbling. Counting the erasure steps can, however, again only increase the length measure by a factor of 2, since every clause being deleted has to be downloaded or inferred prior to its deletion and thus was already counted once in the length measure.*

The following proposition is immediately clear from the definition of the clause space measure and was first mentioned in Esteban & Torán (2001).

Proposition 2.8 (Esteban & Torán 2001). *Let F be an unsatisfiable formula. Then it holds* $\mathrm{CS}(F \vdash \square) = \min_{\pi:F\vdash\square} \mathsf{Black}(G_\pi)$.

As shown by Razborov (2018), most resolution complexity measures can be related to resolution depth. It is not hard to see that resolution depth is also an upper bound for tree-like clause space:

Proposition 2.9. *For any unsatisfiable formula F, we have*

$$\text{Tree-CS}(F \vdash \square) \leq \mathrm{Depth}(F \vdash \square) + 2.$$

Proof. Any refutation graph for F can be "unfolded" into a tree of the same depth, and the black pebbling number of a tree of depth d is at most $d + 2$. □

Amortized space measures for resolution were also introduced by Razborov (2018).

Definition 2.10 (Amortized space measures for resolution). *The* amortized clause space *(*amortized variable space*) of a resolution refutation π is defined by* $\mathrm{CS}^*(\pi) := \mathrm{CS}(\pi)\cdot\log \mathrm{L}(\pi)$ *and* $\mathrm{VS}^*(\pi) := \mathrm{VS}(\pi)\cdot\log \mathrm{L}(\pi)$, *respectively.*

Taking the minimum over all refutations of a formula F, we let

$$\mathrm{CS}^*(F \vdash \square) := \min_{\pi:F\vdash\square} \mathrm{CS}^*(\pi), \text{ and}$$
$$\mathrm{VS}^*(F \vdash \square) := \min_{\pi:F\vdash\square} \mathrm{VS}^*(\pi).$$

2.3. Formula families. In this section, we will introduce two families of formulas used throughout this paper.

Pebbling formulas and their XORification. In the last years, there has been renewed interest in pebbling in the context of proof complexity. This is so because pebbling results can be partially translated into proof complexity results by studying so-called pebbling formulas (Ben-Sasson & Nordström 2011; Ben-Sasson & Wigderson 2001). These are unsatisfiable CNF formulas encoding the pebble game played on a DAG G.

Definition 2.11 (Pebbling formulas). *Let $G = (V, E)$ be a DAG with a set of sources $S \subseteq V$ and a unique sink z. We identify every vertex $v \in V$ with a Boolean variable v. The* pebbling contradiction *over G, denoted Peb_G, is the conjunction of the clauses:*

- *for all sources $s \in S$, a unit clause s,* (source axioms)
- *for all $v \notin S$, the clause $\bigvee_{u \in \mathrm{pred}_G(v)} \overline{u} \vee v$,* (pebbling axioms)
- *for the unique sink z, the unit clause $\overline{z}$.* (sink axiom)

Often, it turns out, that the formulas in Definition 2.11 are a bit too easy to refute. A good way to make them slightly harder is to substitute some suitable Boolean function $f(x_1, \ldots, x_d)$ of arity d for each variable x and expand the result into CNF. This general case is discussed in Nordström (2015). We restrict ourselves to the special case of the second degree XORification.

For notational convenience, we assume that the formula F we are trying to make harder only has variables x, y, z, et cetera, without subscripts, so that $x_1, x_2, y_1, y_2, z_1, z_2$, et cetera, are new variables not occurring in F.

Definition 2.12. (Substitution formulas, Ben-Sasson & Nordström (2008)). *For a positive literal x define the* XORification *of x to be $x[\oplus_2] := \{x_1 \vee x_2,\ \overline{x_1} \vee \overline{x_2}\}$. For a negative literal $\overline{y}$, the XORification is $\overline{y}[\oplus_2] := \{y_1 \vee \overline{y_2},\ \overline{y_1} \vee y_2\}$. The XORification of a clause $C = a_1 \vee \cdots \vee a_k$ is the CNF formula*

$$C[\oplus_2] := \bigwedge_{C_1 \in a_1[\oplus_2]} \cdots \bigwedge_{C_k \in a_k[\oplus_2]} (C_1 \vee \cdots \vee C_k)$$

and the XORification of a CNF formula F is $F[\oplus_2] := \bigwedge_{C \in F} C[\oplus_2]$.

Remark 2.13 (Ben-Sasson & Nordström 2008). *If the graph G has n vertices and maximal in-degree ℓ, then $\mathrm{Peb}_G[\oplus_2]$ is an unsatisfiable $2(\ell+1)$-CNF formula with at most $2^{\ell+1} \cdot n$ clauses over $2n$ variables.*

Tseitin formulas. Tseitin formulas encode the combinatorial principle that for all graphs the sum of the degrees of the vertices is even. This class of formulas was introduced in Tseitin (1968) and has been extremely useful for the analysis of proof systems.

Definition 2.14 (Tseitin formulas). *Let $G = (V, E)$ be a connected undirected graph and let $\chi\colon V \to \{0,1\}$ be a* marking *of the vertices of G. A marking χ is called* odd *if it satisfies the property $\sum_{v\in V} \chi(v) \equiv 1 \pmod 2$ otherwise it is called* even. *Associate to every edge $e \in E$ a propositional variable e. The CNF formula $\mathrm{PARITY}_{v,\chi(v)}$ states that the parity of the values of the edges that have vertex v as endpoint coincides with $\chi(v)$, i. e.,*

$$\mathrm{PARITY}_{v,\chi(v)} := \bigwedge \left\{ \bigvee_{e\ni v} e^{a(e)} \;\middle|\; \begin{array}{l} a(e) \in \{0,1\}, \text{ such that} \\ \bigoplus_{e\ni v} \big(a(e) \oplus 1\big) \not\equiv \chi(v) \end{array} \right\}.$$

Then, the Tseitin formula *associated to the graph G and the marking χ is defined by* $\mathrm{Ts}(G,\chi) := \bigwedge_{v\in V} \mathrm{PARITY}_{v,\chi(v)}$.

For a partial truth assignment α, applying α to $\mathrm{Ts}(G,\chi)$ corresponds to the following simplification of the underlying graph: Setting a variable $e = \{u, v\}$ to 0 corresponds to deleting the edge e in the graph, and setting it to 1 corresponds to deleting the edge from the graph and toggling the values of $\chi(u)$ and $\chi(v)$ in G. We denote by $G{\restriction_\alpha}$ and by $\chi{\restriction_\alpha}$ the remaining graph and marking after applying α according to this process.

Fact 2.15 (Tseitin 1968; Urquhart 1987). *(i) If χ is an even marking of a connected graph G, then $\mathrm{Ts}(G,\chi)$ is satisfiable.*

(ii) Let χ be an odd marking of a connected graph G and e an edge in G that, when deleted divides G in two connected components G_1 and G_2. Then for $i \in \{1,2\}$ there is a partial assignment α_i of the variable e such that $\chi{\restriction_{\alpha_i}}$ is an odd marking of G_i.

2.4. Combinatorial games for Tree-CS in resolution. Important tools for our results are two two-player combinatorial games.

The Prover–Delayer game is played on *formulas* and was introduced in Pudlák & Impagliazzo (2000) in order to prove lower bounds for tree-like resolution length. Later it was shown in Esteban & Torán (2003) that the game exactly characterizes tree-like resolution clause space.

The Raz–McKenzie game is played on *DAGs* and was introduced in Raz & McKenzie (1999) as a tool for studying the depth complexity of decision trees for search problems.

Definition 2.16 (Prover–Delayer game). *The* Prover–Delayer game *is played between two players, called* Prover *(he), and* Delayer *(she), played on an unsatisfiable CNF formula F. The game is played in rounds. Each round starts with Prover querying the value of a variable. Delayer can give one of three answers:* 0, 1, *or* $*$. *If* 0 *or* 1 *is chosen by Delayer, no points are scored by her and the queried variable is set to the chosen value. If Delayer answers* $*$, *then Prover gets to decide the value of that variable, and Delayer scores one point. The game finishes when any clause in F has been falsified (all its literals are set to* 0*) by the partial assignment constructed this way. If this is not the case, the next round begins. The aim of Delayer is to win as many points as possible, while Prover aims to minimize this quantity.*

Definition 2.17 (Game value of the Prover–Delayer game). *Let F be an unsatisfiable CNF formula. The game value of the Prover–Delayer game played on F, denoted by $\mathsf{PD}(F)$, is the greatest number of points Delayer can score on F against an optimal strategy of Prover.*

The Prover–Delayer game exactly characterizes the tree-like clause space of a formula. The constant term of the original result in Esteban & Torán (2003, Theorem 2.2) was slightly modified to match our definitions of clause space and the pebble game without sliding.

Theorem 2.18 (Esteban & Torán 2003). *Let F be an unsatisfiable CNF formula. Then*

$$\text{Tree-CS}(F \vdash \square) = \mathsf{PD}(F) + 2.$$

Definition 2.19 (Raz–McKenzie game). *The* Raz–McKenzie game *is played on a single-sink DAG G by two players,* Pebbler *and* Colorer. *The game is played in rounds. In the first round, Pebbler places a pebble on the sink and Colorer colors the pebble red. In all subsequent rounds, Pebbler places a pebble on an arbitrary empty vertex of G and Colorer colors this new pebble either red or blue. The game ends when there is a vertex with a red pebble that is either a source vertex or all its direct predecessors in the graph have blue pebbles.*

Definition 2.20 (Raz–McKenzie price). *The* Raz–McKenzie price $\mathsf{R\text{-}Mc}(G)$ *of a single sink DAG G is the smallest number r such that Pebbler has a strategy to make the game end in at most r rounds against an optimal strategy of Colorer.*

In Chan (2013), it was shown that the reversible pebbling price and the Raz–McKenzie price coincide for any single-sink DAG.

Theorem 2.21 (Chan 2013). *For any single-sink DAG G, we have*

$$\mathsf{R\text{-}Mc}(G) = \mathsf{Rev}(G).$$

3. Separations between tree-like and general resolution clause space for pebbling formulas

It can be observed from existing results about resolution depth that the tree-resolution clause space for the XORification of any formula F is within a constant factor of the resolution depth of F. As a corollary of this result, we will show that the tree-resolution clause space of the XORification of a pebbling formula is within a constant factor of the reversible pebbling number of the underlying graph. This fact is a useful tool for obtaining clause space separations between tree-like and general resolution.

Theorem 3.1. *For any unsatisfiable CNF formula F, it holds*

$$\mathrm{Depth}(F \vdash \square) \leq \text{Tree-CS}\big(F[\oplus_2] \vdash \square\big) \leq 2 \cdot \mathrm{Depth}(F \vdash \square) + 2.$$

Proof. The first inequality follows from the proof of Theorem 5.4 in Urquhart (2011). There it is proven that $2^{\text{Depth}(F\vdash\square)} \leq \text{Tree-Size}\big(F[\oplus_2]\vdash\square\big)$ but the same proof is easily adapted to show $\text{Depth}(F\vdash\square) \leq \text{Tree-CS}\big(F[\oplus_2]\vdash\square\big)$. The second inequality follows from Proposition 2.9 and the fact that $\text{Depth}\big(F[\oplus_2]\vdash\square\big) \leq 2\cdot\text{Depth}(F\vdash\square)$. □

We obtain the following consequence for the special case of pebbling formulas:

Corollary 3.2. *For any single-sink DAG G it holds*

$$\mathsf{Rev}(G) \leq \text{Tree-CS}\big(\text{Peb}_G[\oplus_2]\vdash\square\big) \leq 2\cdot\mathsf{Rev}(G)+4.$$

Proof. We use the equivalence $\mathsf{Rev}(G) = \mathsf{R\text{-}Mc}(G)$ between the Raz–McKenzie game and reversible pebbling. For a single-sink DAG G, let the augmented graph $\hat{G}$ be the graph obtained by introducing a new vertex v and adding a new directed edge from the sink of G to v. Then v is the unique sink of $\hat{G}$. It is not hard to see that $\mathsf{R\text{-}Mc}(G) \leq \mathsf{R\text{-}Mc}(\hat{G}) \leq \mathsf{R\text{-}Mc}(G)+1$. Chan (2013) showed that $\mathsf{R\text{-}Mc}(\hat{G}) = \text{Depth}(\text{Peb}_G\vdash\square)$ (see also de Rezende *et al.* (2020)).

Let $F = \text{Peb}_G$. Using Theorem 3.1 and the above mentioned results, we then have

$$\begin{aligned}\mathsf{Rev}(G) \leq \mathsf{R\text{-}Mc}(\hat{G}) &= \text{Depth}(\text{Peb}_G\vdash\square)\\ &\leq \text{Tree-CS}\big(\text{Peb}_G[\oplus_2]\vdash\square\big)\\ &\leq 2\cdot\text{Depth}(\text{Peb}_G\vdash\square)+2\\ &= 2\cdot\mathsf{R\text{-}Mc}(\hat{G})+2 \leq 2\cdot\mathsf{Rev}(G)+4.\end{aligned}$$ □

From Corollary 3.2 and Proposition 2.9, it follows that for any graph G with a gap between its black and reversible pebbling prices, the same separation can be obtained between the general and tree-like clause space of the corresponding pebbling formula. We mention some examples for which such a pebbling separation is known:

- The *path graphs*. Let P_n be a directed path with n vertices. Bennett (1989) noticed, that these graphs provide a separation between black and reversible pebbling, proving that

$\mathsf{Rev}(P_n) = \lceil \log n \rceil$. It was shown in Järvisalo *et al.* (2012), using a direct proof, that $\mathrm{CS}\big(\mathrm{Peb}_{P_n}[\oplus_2] \vdash \square\big) = \mathrm{O}(1)$ while $\text{Tree-CS}\big(\mathrm{Peb}_{P_n}[\oplus_2] \vdash \square\big) = \Theta(\log n)$.

- The *road graphs* from Chan *et al.* (2015) provide a class of graphs for which the black pebbling price is non-constant and the reversible pebbling number is larger by a logarithmic factor. A road graph of length ℓ and width w contains ℓ layers each having w vertices, and the i-th vertex of a layer has two incoming edges from the i-th and $(i+1)$-st vertices of the previous layer (modulo w). The graphs narrow towards the end in a pyramid-like fashion in order to have a unique sink.

Theorem 3.3 (Chan *et al.* 2015). *For any function* $s(n) = \mathrm{O}\big(n^{1/2-\varepsilon}\big)$ *with* $0 < \varepsilon < \frac{1}{2}$ *constant, there is a family of DAGs* $(G_n)_{n=1}^{\infty}$ *of size* $\Theta(n)$ *with a single sink and maximal in-degree* 2 *such that* $\mathsf{Black}(G_n) = \mathrm{O}\big(s(n)\big)$ *and* $\mathsf{Rev}(G_n) = \Omega\big(s(n) \log n\big)$.

Corollary 3.4. *For any function* $s(n) = \mathrm{O}\big(n^{1/2-\varepsilon}\big)$ *with constant parameter* $0 < \varepsilon < \frac{1}{2}$, *there is a family of pebbling formulas* $\big(\mathrm{Peb}_{G_n}[\oplus_2]\big)_{n=1}^{\infty}$ *with* $\Theta(n)$ *variables such that* $\mathrm{CS}\big(\mathrm{Peb}_{G_n}[\oplus_2] \vdash \square\big) = \mathrm{O}\big(s(n)\big)$ *and* $\text{Tree-CS}\big(\mathrm{Peb}_{G_n}[\oplus_2] \vdash \square\big) = \Omega\big(s(n) \log n\big)$.

The logarithmic factor in the number of vertices is almost the largest separation that can be obtained using this method since it is known that the reversible pebbling price can be upper bounded in terms of black pebbling space and time:

Theorem 3.5 (Královič 2004). *For any DAG* G *with simultaneous black pebbling space* s *and black pebbling time* t *it holds*

$$\mathsf{Rev}(G) \leq s\lceil \log t \rceil.$$

By virtue of this result and Corollary 3.2 we obtain:

Corollary 3.6. *For any single-sink DAG* G *it holds*

$$\text{Tree-CS}\big(\mathrm{Peb}_G[\oplus_2] \vdash \square\big) = \mathrm{O}\left(\min_{\mathcal{P}} \big(\mathsf{space}(\mathcal{P}) \cdot \log \mathsf{time}(\mathcal{P})\big)\right),$$

where the minimum is taken over all black pebblings $\mathcal{P}$ *of* G.

This shows that the given separations cannot be improved for graphs for which the minimum black pebbling space is obtained with a one-shot strategy as it is the case for the path and road graphs, since the pebbling time for such a strategy is n. We present the first graph class for which the best black pebbling strategy is not one-shot with a separation between black and reversible pebbling space. We do not obtain, however, any better separation than the $\log n$ factor obtained in the previous examples. We conjecture that this is in fact optimal. Our graphs $\hat{G}(c,k)$ are simplified versions of the original graphs in Carlson & Savage (1982). Another adaptation of the original graphs is the family $\Gamma(c,r)$ studied in Nordström (2015), for which an upper bound on the reversible pebble price was recently shown by de Rezende *et al.* (2021). We have simplified the graphs, eliminating the original pyramids since we are not analyzing the black-white pebbling price, but our lower bound on reversible pebbling can be adapted to the original graphs or those in the family $\big(\Gamma(c,r)\big)_{c,r=1}^{\infty}$.

The graphs of the following definition are depicted in Figure 3.1.

Definition 3.7 (Simplified Carlson–Savage graphs). *The class of DAGs* $\big(G(c,k)\big)_{c,k=1}^{\infty}$ *with parameters* $c,k \geq 1$ *is inductively defined in* k. *The base case* $G(c,1)$ *is the graph with one source node* s *and* c *sink nodes* $z_1,\ldots,z_c$, *as well as directed edges* (s,z_i) *for* $i=1,\ldots,c$ *from this source to all sink nodes. The graph* $G(c,k+1)$ *is composed of*

- *the graph* $G(c,k)$, *where we let* $z_1,\ldots,z_c$ *denote the sinks of this graph,*
- *and* c *disjoint so-called* spines, *where a spine is just a path of length* $2c^2k$. *Each spine is divided into* $2ck$ sections *of* c *consecutive vertices each. The last node of each of the spines is a sink for* $G(c,k+1)$.

Apart from all edges of $G(c,k)$ *and the path-edges in the spines, the following edges are added: For each section in each spine and for each* i *with* $1 \leq i \leq c$, *there is an edge from the* i*-th sink* z_i *of* $G(c,k)$ *to the* i*-th vertex in the section.*

For single-sink graphs, the following definition is needed:

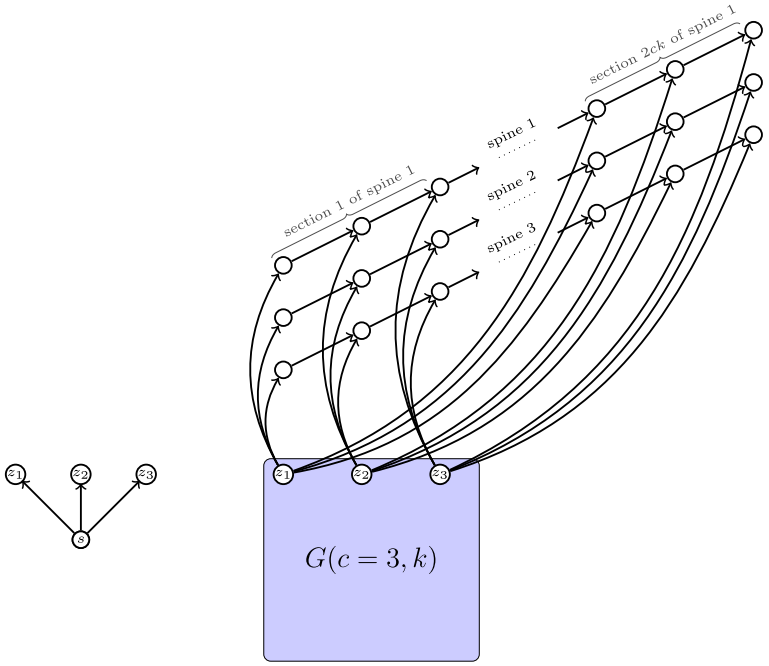

Figure 3.1: The base case $G(3,1)$ for the simplified Carlson–Savage graph with 3 spines and sinks is depicted on the left. The inductive definition of the simplified Carlson–Savage graph $G(3,k+1)$ with 3 spines and sinks is depicted on the right.

Definition 3.8 (One sink simplified Carlson–Savage graphs). *For all $c \geq 1$, we let the graph $\hat{G}(c,1)$ consists of just one edge (s,z) between the source node s and the sink node z. For $k \geq 2$ we define $\hat{G}(c,k)$ exactly as $G(c,k)$ but with just* one *spine at the k-th level (all other levels have c spines). The last vertex of this spine is the only sink of $\hat{G}(c,k)$.*

Lemma 3.9. *The following claims hold:*

(i) $\hat{G}(c,k)$ *has* $\Theta(c^3k^2)$ *vertices,*

(ii) $\mathsf{Black}\big(\hat{G}(c,k)\big) \leq k+1$ *for any* $c,k \geq 1$,

(iii) for $k > 1$ *and* $c > k+1$, *the best black pebbling strategy of the graph* $\hat{G}(c,k)$ *is not one-shot, and*

(iv) $\mathsf{Rev}\big(\hat{G}(c,k)\big) \geq \min\big\{c, (k-1)\log c + \log(k!)\big\}$ *for any* $c, k \geq 1$.

Proof. The first part follows easily by inductive counting.

For part (ii) of the lemma, we show inductively over k that any sink of $G(c,k)$ can be pebbled using $k+1$ pebbles. The result follows since $\hat{G}(c,k)$ is a subgraph of $G(c,k)$. The claim is trivial for $k=1$. For $k>1$ the first vertex in any of the spines in $G(c,k)$ can be pebbled by placing a pebble on the corresponding sink of $G(c,k-1)$, removing all the pebbles except this one, and then pebbling the first vertex in the spine. The following strategy can be used for any other vertex v in the spine once its direct predecessor in the spine is pebbled: remove all the pebbles in the graph except the one on the direct spine predecessor of v, pebble the sink connected to v in $G(c,k-1)$, remove all the pebbles except the 2 on the direct predecessors of v, and then place a pebble on v. For this, by the induction hypothesis, at most $k+1$ pebbles are needed.

For (iii), suppose $\mathcal{P}$ is a one-shot strategy for pebbling $\hat{G}(c,k)$. Let $z_1, \ldots, z_c$ be the sinks of $\hat{G}(c,k-1)$. Since each of these sinks is a direct predecessor of a vertex in each of the sections of the unique spine of $\hat{G}(c,k)$, once each of these vertices has a pebble on it, the pebble cannot be removed until the last section of the spine is pebbled. This is because we are not allowed to re-pebble the sinks since the pebbling is one-shot. Therefore $\mathsf{space}(\mathcal{P})$ is at least c. But part (ii) tells us that $\mathsf{Black}\big(\hat{G}(c,k)\big) \leq k+1$. Thus, when $c > k+1$, the best black pebbling strategy for the graphs $\hat{G}(c,k)$ with respect to space cannot be one-shot.

Part (iv) is more involved. We use the equivalence between reversible pebbling and the Raz–McKenzie game and show, also by induction over k, that the number of rounds to finish a game on $\hat{G}(c,k)$ starting from a configuration in which less than c vertices have been colored blue, and no vertex (except the sink) in the unique spine of $\hat{G}(c,k)$ is colored, is at least $\min\big\{c, (k-1)\log c + \log(k!)\big\}$. We give a strategy for Colorer obtaining this bound on the number of rounds. The base case is trivial. For $k \geq 2$, initially the only vertex colored red is the unique sink of $\hat{G}(c,k)$. Let us denote the unique spine from $\hat{G}(c,k)$ as the *k-spine*. The game is divided in k stages (starting at stage k and finishing at stage 1).

Stage k finishes when there is a blue vertex in the k-spine at a distance less than $2c$ from a red vertex. In stage k, Colorer only gives the color red to vertices in the k-spine; if some vertex in $G(c, k-1)$ is queried by Pebbler, Colorer always answers with the blue color. Because of this, the game cannot finish before the end of stage k. For simplicity, we may assume that the first vertex of the k-spine has been colored blue (for free, this can only make the strategy of Colorer harder), also for the clarity of exposition let us say that the k-spine is directed from left to right. The strategy of Colorer on the k-spine is to keep the gap between the rightmost blue vertex a (initially the initial node of the spine) and the leftmost red vertex b (initially the sink) as large as possible. That is, for any queried vertex v in the k-spine, if v lies at the left of a, it is colored blue, if it is at the right of b it is colored red; and otherwise (i. e., if v is between a and b) if the distance from a to v is smaller that or equal to the distance from v to b, then v is colored blue, otherwise it is colored red. This strategy is followed by Colorer as long as the gap between a and b is at least $2c$. Once it is smaller than $2c$, stage k ends. If at this moment at least c vertices have been queried, there have been at least c rounds and the result follows. Otherwise, there has to be a spine in $G(c, k-1)$ without any colored vertex on it (there are c spines). Let us call t the sink of this spine and t' its rightmost uncolored successor in the k-spine. We can suppose that at this moment Colorer colors (for free) t, t', as well as all uncolored vertices to the right of t' in the k-spine with color red, and all the uncolored vertices to the left of t' in the k-spine with blue. Again this only makes the strategy of Colorer harder since we are not counting these rounds. But now the game has been reduced to the instance of the graph $\hat{G}(c, k-1)$ containing the sink t. The number of rounds in stage k is at least $\log(\frac{2c^2k}{2c}) = \log c + \log k$ (this would happen with a binary search strategy of Pebbler on the k-spine). If in all the stages less than c vertices are queried, by induction, the rounds to finish the game on $\hat{G}(c, k-1)$ are at least $(k-2)\log c + \log\big((k-1)!\big)$. Adding these rounds to those from stage k we get the result. □

THEOREM 3.10. *For any function* $s(n) = \Theta\big(n^{1/5-\varepsilon}\big)$ *with* $0 < \varepsilon < \frac{1}{5}$ *constant there is a family of pebbling formulas* $\big(\mathrm{Peb}_{G_n}[\oplus_2]\big)_{n=1}^{\infty}$ *with* $\mathrm{O}(n)$ *variables such that* $\mathrm{CS}\big(\mathrm{Peb}_{G_n}[\oplus_2] \vdash \square\big) = \mathrm{O}\big(s(n)\big)$ *and* $\text{Tree-CS}\big(\mathrm{Peb}_{G_n}[\oplus_2] \vdash \square\big) = \Omega\big(s(n)\log n\big)$*, and the best black pebbling strategy for the graphs* G_n *is not one-shot.*

PROOF. We show that for any such function s, there is a graph family $\big(\hat{G}\big(c(n), \lceil s(n)\rceil\big)\big)_{n=1}^{\infty}$ with the corresponding gap between its black and reversible pebbling prices. The result follows from Corollary 3.2.

Given any such function $s(n) = \Theta\big(n^{1/5-\varepsilon}\big)$ with $0 < \varepsilon < \frac{1}{5}$ constant, we define $c(n) := \lceil s(n) \cdot \log n\rceil$. This allows us to consider the graphs $\hat{G}\big(c(n), \lceil s(n)\rceil\big)$. By Lemma 3.9 (i), this graph has $\mathrm{O}\big(c(n)^3 \cdot \lceil s(n)\rceil^2\big) = \mathrm{O}\big(s(n)^5 \cdot \log^3 n\big) = \mathrm{O}\big(n^{1-5\varepsilon} \cdot \log^3 n\big) = \mathrm{O}(n)$ vertices. By Lemma 3.9 (ii), the graph has a black pebbling number upper bounded by $\lceil s(n)\rceil + 1 = \mathrm{O}\big(s(n)\big)$. It only remains to show that the reversible pebbling number of the graph is asymptotically lower bounded by $s(n)\log n$. We consider two cases:

Case 1: $\min\Big\{c(n), \big(\lceil s(n)\rceil - 1\big)\log c(n) + \log\big(\lceil s(n)\rceil!\big)\Big\} = c(n)$.
In this case, Lemma 3.9 (iv) implies that the reversible pebbling number of the graph is lower bounded by $c(n)$, which, by definition, is greater than or equal to $s(n)\log n$.

Case 2: $\min\Big\{c(n), \big(\lceil s(n)\rceil - 1\big)\log c(n) + \log\big(\lceil s(n)\rceil!\big)\Big\} \neq c(n)$.
In this case, already the first term, i. e., $\big(\lceil s(n)\rceil - 1\big)\log c(n)$ is in

$$\begin{aligned}
&\Omega\Big(\big(\lceil s(n)\rceil - 1\big)\log\big(\lceil s(n)\rceil \log n\big)\Big)\\
&\quad= \Omega\Big(\big(\lceil s(n)\rceil - 1\big)\log\big(\lceil s(n)\rceil\big) + \big(\lceil s(n)\rceil - 1\big)\log\log n\Big)\\
&\quad= \Omega\Big(\big(\lceil s(n)\rceil - 1\big)\big(1/5 - \varepsilon\big)\log n + \big(\lceil s(n)\rceil - 1\big)\log\log n\Big)\\
&\quad= \Omega\big(s(n)\log n\big).
\end{aligned}$$

□

4. Upper bounds for Tree-CS for general formulas

We provide generalizations of Corollary 3.6 for general formulas.

Observation 4.1. *For any unsatisfiable formula F, it holds*

$$\text{Tree-CS}(F \vdash \square) \leq \text{VS}^*(F \vdash \square) + 2.$$

Proof. This follows from the fact that

$$\text{Tree-CS}(F \vdash \square) \leq \text{Depth}(F \vdash \square) + 2 \leq \text{VS}^*(F \vdash \square) + 2,$$

where the first inequality is Proposition 2.9 and the second inequality is due to Razborov (2018, Theorem 3.1). □

We now want to prove that the upper bound in Observation 4.1 also works for amortized clause space, i. e., for any unsatisfiable formula F we also have $\text{Tree-CS}(F \vdash \square) \leq \text{CS}^*(F \vdash \square) + 2$ (see Corollary 4.3). For this, we first show that the tree-like clause space of a formula F is always upper bounded by the reversible pebble game played on a refutation of F. Note that the minimum in Theorem 4.2 is taken over all possible refutations of F, not only over the tree-like ones.

Theorem 4.2. *For any unsatisfiable formula F with n variables it holds*

$$\text{Tree-CS}(F \vdash \square) \leq \min_{\pi : F \vdash \square} \mathsf{Rev}(G_\pi) + 2, \quad \textit{and}$$
$$\min_{\pi : F \vdash \square} \mathsf{Rev}(G_\pi) \leq \text{Tree-CS}(F \vdash \square)\big(\lceil \log n \rceil + 1\big).$$

Proof. Let F be an unsatisfiable formula with n variables. We simplify the proof by following the intuition behind the Raz–McKenzie game and identify the color blue with 1 and the color red with 0.

For proving the first inequality, let π be a resolution refutation of F with a refutation-graph G_π and $\mathsf{Rev}(G_\pi) =: k$. We will use Theorem 2.18, as well as Theorem 2.21 applied to G_π: It suffices to give a strategy for Prover in the Prover–Delayer game played on F under which he has to pay at most k points. Prover basically simulates the strategy of Pebbler in the Raz–McKenzie game played on G_π, which coincides with reversible pebbling. By doing so, a partial assignment α falsifying an initial clause of F will

be produced. The game is divided in stages. Initially, the partial assignment is the empty assignment. In each stage, if Pebbler chooses a clause $C \in V(G_\pi)$, Prover queries the variables in C not yet assigned by α, one by one, extending the partial assignment α with the answers of Delayer, until either:

1. the clause C is satisfied or falsified by α, or
2. a variable x in C is given value $*$ by Delayer.

In case 1, Prover moves to the next stage, simulating the strategy of Pebbler assuming Colorer has given clause C the color $C\restriction_\alpha$. In case 2, Prover extends α by assigning x with the value that satisfies C and moves to the next stage, simulating the strategy of Pebbler, assuming Colorer has given clause C the color 1. The game is played until α falsifies a clause in F. After at most k stages, the Raz–McKenzie games finishes and therefore Delayer can score at most k points. It is only left to show that at the end of the game a clause in F is falsified by α. When the Raz–McKenzie game finishes, either a source in G_π is assigned color 0 by Coloreror a vertex with all its direct predecessors being colored 1 is colored 0. Since α encodes Delayer's answers (with $*$ replaced by the uniquely defined concrete Boolean values given by Prover as described above in case 2), the first situation corresponds to α falsifying a clause in F. The second situation is not possible since for any partial assignment α it cannot be that α satisfies two parent clauses in a resolution proof, while falsifying their resolvent.

For the proof of the second inequality, let $k := \text{Tree-CS}(F \vdash \square)$. By Proposition 2.8 we know that there is a refutation π of F whose underlying graph G_π is a tree with black pebbling price k. We can suppose that the refutation is *regular*, that is, in every path from the empty clause to a clause in F in the refutation tree, each variable is resolved at most once (Esteban & Torán 2001, Theorem 5.1). Then the depth of the tree is at most n.

We show that for any binary tree T with depth $d \geq 1$ and $\mathsf{Black}(T) = \kappa$, $\mathsf{Rev}(T) \leq \kappa(\lceil \log d \rceil + 1)$. The result follows from this fact.

The case $d = 1$ is trivial. Let $d > 1$. For any node v in T let T_v be the subtree rooted at v. For the sake of convenience,

we refer to $\mathsf{Black}(T_v)$ as the *pebbling number of* v. We show by induction on κ that for any vertex v in T, if $\mathsf{Black}(T_v) = \kappa$ then there is a strategy for Pebbler in the Raz–McKenzie game on T_v with most $\kappa(\lceil \log d \rceil + 1)$ rounds. For the base case $\kappa = 1$, the vertex v must be a leaf node and the game needs only one round. For $\kappa > 1$, the game starts, according to the rules, by Pebbler querying the root v of the subtree and Colorer answering 0. We consider two cases, depending on whether for both direct predecessors v_1 and v_2 of v in T, $\mathsf{Black}(T_{v_1}) = \mathsf{Black}(T_{v_2}) = \kappa - 1$ or not. In the former case, Pebbler queries one of them, say v_1. If the answer is 0, he continues on T_{v_1} and otherwise he continues on T_{v_2}. By induction, the number of rounds in this case is at most $2+(\kappa-1)(\lceil \log d \rceil+1) \leq \kappa(\lceil \log n \rceil+1)$. In case, it is not true, that $\mathsf{Black}(T_{v_1}) = \mathsf{Black}(T_{v_2}) = \kappa - 1$, since $\mathsf{Black}(T_v) = \kappa$, and G_π is a tree, one of the trees T_{v_1} or T_{v_2} leading to v must have pebbling number κ and the other one must have pebbling number smaller than κ (this is a well-known fact, see e. g. Flajolet *et al.* (1979)). Pebbler considers the path of nodes starting at v and going towards the leaves, such that the nodes in the path have pebbling number κ, until a node u is reached, for which both direct predecessors have pebbling number $\kappa - 1$. Such a node u must exist because of the previously mentioned fact. Let u_1 be one of the direct predecessors of u. The length of the path from v to u_1 is at most d. Pebbler queries the vertices in the path between v and u_1 with binary search, until a vertex t is found that is colored with color 0 by Colorer, while its direct predecessor in the path $v \rightsquigarrow u_1$ has been colored 1. At this point, Pebbler continues playing the game on the tree rooted at the uncolored predecessor of t. In all situations at most $1 + \lceil \log d \rceil$ vertices have been queried and the game has been reduced to a subgraph with smaller pebbling number. □

Corollary 4.3. *For any unsatisfiable formula F it holds*

$$\text{Tree-CS}(F \vdash \square) \leq \text{CS}^*(F \vdash \square) + 2.$$

Proof. By Theorem 3.5 we have

$$\min_{\pi : F \vdash \square} \mathsf{Rev}(G_\pi) + 2 \leq \min_{\mathcal{P}} \big(\mathsf{space}(\mathcal{P}) \cdot \log \mathsf{time}(\mathcal{P})\big) + 2,$$

where the minimum is taken over all black pebblings $\mathcal{P}$ of G_π. The result follows with (a slight adaption of) Proposition 2.8 since every black pebbling $\mathcal{P}$ of G_π defines a configurational refutation of F with clause space equal to $\mathsf{space}(\mathcal{P})$ and length $\mathsf{time}(\mathcal{P})$. □

5. Optimal separations for Tseitin formulas

In this section, we prove optimal separations between tree-like clause space and variable space as well as clause space in the context of Tseitin formulas. This complements the relations between clause space and variable space of Tseitin formulas recently given in Galesi *et al.* (2020).

THEOREM 5.1. *For any connected graph G with n vertices and odd marking χ we have*

$$\text{Tree-CS}\big(\text{Ts}(G,\chi)\vdash\square\big) \le \text{CS}\big(\text{Ts}(G,\chi)\vdash\square\big)\cdot\log n + 2, \quad \textit{and}$$
$$\text{Tree-CS}\big(\text{Ts}(G,\chi)\vdash\square\big) \le \text{VS}\big(\text{Ts}(G,\chi)\vdash\square\big)\cdot\log n + 2.$$

PROOF. The proof is based on the one for the lower bound for the clause space of Tseitin formulas from Torán (1999). Let $G = (V, E)$ be a connected graph with $n := |V|$ vertices, χ an odd marking, and $\pi = (\mathbb{M}_0, \ldots, \mathbb{M}_t)$ a refutation of $\text{Ts}(G, \chi)$ with $\text{CS}(\pi) =: k$. We use the refutation π to give a strategy for Prover in the Prover–Delayer game for which he has to pay at most $k \log n$ points. The additive constant 2 in the Theorem then comes from applying Theorem 2.18.

We say that a partial assignment α of some of the variables in $\text{Ts}(G, \chi)$ is *non-splitting* if after applying α to the formula, the resulting graph still has a connected component with an odd marking (*odd component*) of size at least $\lceil \frac{|V|}{2} \rceil$, and the rest are components with even markings. Consider the last configuration $\mathbb{M}_s$ in π for which there is a partial assignment α fulfilling:

1. α simultaneously satisfies all clauses in $\mathbb{M}_s$, and
2. α is non-splitting.

This stage must exist since before the initial step the empty truth assignment is trivially a non-splitting partial assignment satisfying

the clauses in $\mathbb{M}_0 = \varnothing$. At the end, the last configuration $\mathbb{M}_t$ in the refutation contains the empty clause which cannot be satisfied by any assignment. Thus, stage s must exist in between.

The step from s to $s+1$ cannot be a deletion step or a resolution step since otherwise a partial assignment α satisfying the two conditions at step s would also satisfy them at step $s+1$. Therefore $\mathbb{M}_{s+1}$ contains one clause C more than $\mathbb{M}_s$, and it must be an axiom of $\mathrm{Ts}(G,\chi)$. For some vertex v in G, this axiom clause C belongs to the formula $\mathrm{PARITY}_{v,\chi(v)}$. Let α be a partial assignment of minimal size satisfying the conditions at stage s. Since $\mathbb{M}_s$ contains at most $k-1$ clauses, α assigns at most $k-1$ variables. It is possible to extend α to satisfy the clause $C\restriction_\alpha$ since v either belongs to an even component in $(G\restriction_\alpha, \chi\restriction_\alpha)$ or to the large odd component in this graph and therefore $C\restriction_\alpha \neq \square$. Because of this, vertex v must belong to the unique odd component since otherwise we could obtain an assignment α' satisfying the conditions at step $s+1$, by extending α in a non-splitting way to one variable in C.

Let $C\restriction_\alpha = (\ell_1 \vee \cdots \vee \ell_m)$, $m \geq 1$, where the ℓ_i's are literals corresponding to the edges e_i with endpoint v in $G\restriction_\alpha$ (see Figure 5.1). Observe that deleting any of these edges e_i in $G\restriction_\alpha$ cuts the connected component of v in two pieces because otherwise assigning any value to the corresponding edge e_i would not modify the size of the connected components in $G\restriction_\alpha$ and there would be a non-splitting way to extend α to e_i, satisfying C, and obtaining in this way an assignment α' fulfilling the conditions at $s+1$. Also, any component remaining after assigning all the literals in $C\restriction_\alpha$ must have size at most $\lfloor \frac{|V|}{2} \rfloor$ since otherwise there would be a way to extend α satisfying C and producing an odd marking for the largest such component (Fact 2.15), and this extension α' would be non-splitting.

The strategy of Prover is to query the variables assigned in α (which has size at most $k-1$) thus paying at most $k-1$ points and obtaining a partial assignment γ from Delayer. Observe that the connected components are always the same for any value of the assigned variables, only the parity of its markings can change. Therefore the values of the variables in γ are not important. If at

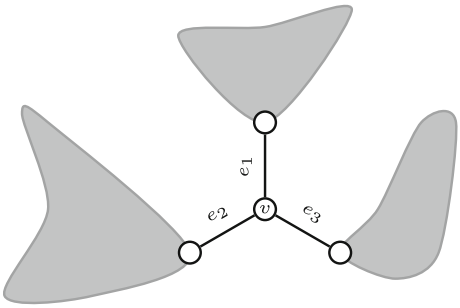

Figure 5.1: The graph $G\restriction_\alpha$ with edges $e_1, \ldots, e_m$ corresponding to the literals in $C\restriction_\alpha = (\ell_1 \vee \cdots \vee \ell_m)$. The gray areas symbolize connected components of $G\restriction_\alpha$. As argued, there are no further edges connecting two gray components.

this point one of the connected components of size at most $\lfloor \frac{|V|}{2} \rfloor$ is odd, then Prover moves to this component and starts playing the game on it. Otherwise, Prover queries the variables in $C\restriction_\gamma$ one by one. If for a variable e_i Delayer answers with $*$, Prover just has to assign e_i so that the smallest of the two components that appear in $G\restriction_\gamma$ after assigning e_i is odd (not necessarily satisfying $C\restriction_\gamma$). This is always possible because of Fact 2.15. If no $*$ is answered, Prover queries the next variable until no variable in $C\restriction_\gamma$ is left. Let γ' be the assignment obtained this way. Since all the variables in C are assigned, all the components (odd or even) remaining after applying γ' have size at most $\lfloor \frac{|V|}{2} \rfloor$. In any case, after applying γ', Prover wins the game or there is an odd connected component of size at most half as large as the initial graph. The original problem has been reduced to another one in a graph with at most $\frac{n}{2}$ many vertices. Also Prover has to pay at most k for obtaining γ'. The players continue the game on the new graph creating an extension of γ'. After repeating this process at most $\log n$ times, an initial clause is falsified.

The second part of the theorem is a little simpler and follows by considering a configurational proof π of variable space k. Everything in the above proof works in the same way, observing that the partial assignment α satisfying all clauses in memory at stage s, when extended to all the variables in the new clause at stage $s+1$

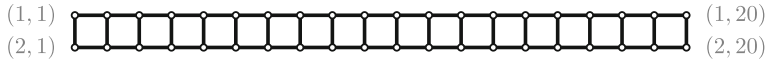

Figure 5.2: The grid graph $G_{2\times 20}$.

needs to assign at most k variables (all those included in the configuration) and is either splitting or falsifies the axiom. Observe that this implies that in every configurational proof there is a point in which every assignment to the variables in the configuration is splitting. □

Next, we show that the upper bounds in Theorem 5.1 are tight by proving that there is a family of Tseitin formulas that provide matching lower bounds. These are formulas corresponding to grid graphs with constant width (see Figure 5.2), which can be considered as the Tseitin version of path graphs.

Definition 5.2 (Grid graphs). *For a natural number $\ell \geq 1$, the* grid graph $G_{2\times\ell}$ *as depicted in Figure 5.2 is given by the vertex set* $V(G_{2\times\ell}) := [2] \times [\ell]$ *and the edge set*

$$E(G_{2\times\ell}) := \left\{ \{(i,j),(i',j')\} \;\middle|\; \begin{array}{l} i,i' \in [2],\ j,j' \in [\ell],\ \text{and} \\ |i-i'| + |j-j'| = 1 \end{array} \right\}.$$

In the following, let χ_ℓ be an odd marking of $G_{2\times\ell}$.

Theorem 5.3. *For the Tseitin formula family* $\big(\mathrm{Ts}(G_{2\times\ell},\chi_\ell)\big)_{\ell=1}^{\infty}$ *with* $3\ell-2$ *variables it holds*

(i) $\text{Tree-CS}\big(\mathrm{Ts}(G_{2\times\ell},\chi_\ell) \vdash \square\big) = \Omega(\log \ell)$,

(ii) $\mathrm{CS}\big(\mathrm{Ts}(G_{2\times\ell},\chi_\ell) \vdash \square\big) = \mathrm{O}(1)$, *and*

(iii) $\mathrm{VS}\big(\mathrm{Ts}(G_{2\times\ell},\chi_\ell) \vdash \square\big) = \mathrm{O}(1)$.

Proof. To show the lower bound on tree-like clause space with Theorem 2.18, we give a strategy for Delayer such that he scores $\Omega(\log \ell)$ points playing on $\mathrm{Ts}(G_{2\times\ell},\chi_\ell)$. In the following, for a subgraph G' of $G_{2\times\ell}$, we define

$$\mathrm{Block}(G') := \max\left\{ b \in \mathbb{N} \;\middle|\; \begin{array}{l} \text{there is a subgraph of } G' \\ \text{that is isomorphic to } G_{2\times b} \end{array} \right\}.$$

Furthermore, letting $j \in [\ell - 1]$, we say that the j-th Block of G' is *intact*, if the edges $\{(1,j),(1,j+1)\}$, $\{(2,j),(2,j+1)\}$, $\{(1,j),(2,j)\}$ and $\{(1,j+1),(2,j+1)\}$ are still present.

The strategy of Delayer is then as follows:

1. If a variable corresponding to an edge e in an even component is queried, Delayer should answer according to some assignment satisfying this component.

2. If a variable corresponding to an edge e in an odd component is queried, Delayer proceed as follows:

   (a) If the deletion of e does not increase the number of connected components in G, Delayer should answer $*$.

   (b) If the deletion of e cuts the graph and both endpoints of e are separated in different connected components, Delayer should answer in a way, that from these two components, the component G' with largest Block(G') receives the odd marking. This is always possible according to Fact 2.15 (b).

At the beginning of the Prover–Delayer game, Block($G_{2\times\ell}$) $= \ell$. After each assignment of a variable in the game, we have Block(G') $\geq \lfloor \frac{1}{2}\,\text{Block}(G) \rfloor$, where we let G denote the underlying graph before the assignment and G' the graph after the assignment. Notice that rule 2 (b) guarantees that the component with the largest Block-value always receives an odd marking. If Delayer plays according to this strategy, we must have Block(G) $= 0$ at the beginning of some round. This means that the Block-value, starting the game with $G_{2\times\ell}$, has to change at least $\Omega(\log \ell)$ times before the game can end. One can notice that if the Block-value changes in a step, the number of connected components does not increase in this step. (Suppose, to reach a contradiction, that the number of connected components changes due to the deletion of an edge e *and* the Block-value also changes. Then e must have been the only edge connecting the connected components C_1 and C_2. Furthermore, e must have been an edge of an intact block B_j. But the intactness of the block B_j implies that after the deletion of e there is still a path that is connecting the components C_1 and C_2, which

contradicts our assumption). According to rule 2 (a), Delayer has answered $*$ in this round and has scored a point.

For the second part, consider the variables (edges) ordered (from left to right) with

$$\big\{(1,j),(2,j)\big\} \prec \big\{(1,j),(1,j+1)\big\} \prec \big\{(2,j),(2,j+1)\big\}$$

and edges with lower j defined to be smaller (with respect to $\prec$) than those with higher j for $1 \leq j \leq \ell - 1$, and consider a regular resolution refutation completely resolving the variables in decreasing order (from right to left). That is, the clauses containing variable $\big\{(1,\ell),(2,\ell)\big\}$ will be first resolved with all clauses containing this variable in negated form (in case it is possible to resolve), and so on. Since the graph has degree at most 3, there is a small number of clauses containing this variable. Also observe that after resolving the last three variables in the ordering in this way, the set of derived clauses plus the initial clauses encode the formula $\mathrm{Ts}(G_{2\times(\ell-1)},\chi')$ for some odd marking χ'. The set of newly derived clauses needed for the resolution of this formula has constant size, and the number of clauses in all the resolution configurations until this point is also constant. Continuing in this order with the complete resolution of all the variables, we obtain a refutation of $\mathrm{Ts}(G_{2\times\ell},\chi)$ with constant clause and variable space. □

6. Conclusions and open problems

By introducing a new connection between tree-like resolution clause space and the reversible pebble game, we have studied the relation between tree-like clause space and the clause space measure for general resolution, obtaining separations between these measures. We conjecture that these separations are optimal and that in fact, the $\log\big(\mathrm{time}(\pi)\big)$ factors in the upper bounds of Theorem 3.5 and Observation 4.1 and Corollaries 3.6 and 4.3 can be improved to a $\log n$ factor (n being the number of graph vertices or formula size, depending on the setting). We have been able to prove this for the restricted case of the Tseitin contradictions.

We have seen that a source for obtaining clause space separations between tree-like and general resolution are graph classes

with a gap between their reversible and black pebbling prices and we have provided a new class of such graphs. An interesting question is whether there exists a graph class with such a separation for a space function larger than $n^{1/2}$.

Acknowledgements

This research was supported by the Deutsche Forschungsgemeinschaft (DFG) under project number 430150230, "Complexity measures for solving propositional formulas".

An extended abstract of this paper has appeared as Torán & Wörz (2020). The authors would like to thank all reviewers for many insightful comments.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov & Avi Wigderson (2002). Space Complexity in Proposi-

tional Calculus. *SIAM Journal on Computing* **31**(4), 1184–1211. Preliminary version in *STOC '00*.

Carlos Ansótegui, María Luisa Bonet, Jordi Levy & Felip Manyà (2008). Measuring the Hardness of SAT Instances. In *Proceedings of the 23rd National Conference on Artificial Intelligence (AAAI '08)*, 222–228.

Eli Ben-Sasson, Russell Impagliazzo & Avi Wigderson (2004). Near Optimal Separation of Tree-Like and General Resolution. *Combinatorica* **24**(4), 585–603.

Eli Ben-Sasson & Jakob Nordström (2008). Short Proofs May Be Spacious: An Optimal Separation of Space and Length in Resolution. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS '08)*, 709–718.

Eli Ben-Sasson & Jakob Nordström (2011). Understanding Space in Proof Complexity: Separations and Trade-offs via Substitutions. In *Proceedings of the 2nd Symposium on Innovations in Computer Science (ICS '11)*, 401–416. Full-length version available at http://eccc.hpi-web.de/report/2010/125/.

Eli Ben-Sasson & Avi Wigderson (2001). Short Proofs are Narrow—Resolution Made Simple. *Journal of the ACM* **48**(2), 149–169. Preliminary version in *STOC '99*.

Charles H. Bennett (1989). Time/Space Trade-offs for Reversible Computation. *SIAM Journal on Computing* **18**(4), 766–776.

María Luisa Bonet, Juan Luis Esteban, Nicola Galesi & Jan Johannsen (1998). Exponential Separations between Restricted Resolution and Cutting Planes Proof Systems. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science (FOCS '98)*, 638–647.

David A. Carlson & John E. Savage (1982). Extreme Time-Space Tradeoffs for Graphs with Small Space Requirements. *Information Processing Letters* **14**(5), 223–227.

Siu Man Chan (2013). Just a Pebble game. In *Proceedings of the 28th Annual IEEE Conference on Computational Complexity (CCC '13)*, 133–143.

Siu Man Chan, Massimo Lauria, Jakob Nordström & Marc Vinyals (2015). Hardness of Approximation in PSPACE and Separation Results for Pebble Games (Extended Abstract). In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS '15)*, 466–485.

Martin Davis, George Logemann & Donald Loveland (1962). A Machine Program for Theorem Proving. *Communications of the ACM* **5**(7), 394–397.

Martin Davis & Hilary Putnam (1960). A Computing Procedure for Quantification Theory. *Journal of the ACM* **7**(3), 201–215.

Susanna F. de Rezende, Or Meir, Jakob Nordström, Toniann Pitassi, Robert Robere & Marc Vinyals (2020). Lifting with Simple Gadgets and Applications to Circuit and Proof Complexity. In *Proceedings of the 61st IEEE Annual Symposium on Foundations of Computer Science (FOCS '20)*, 24–30. Full-length version available as *arXiv:2001.02144*.

Susanna F. de Rezende, Or Meir, Jakob Nordström & Robert Robere (2021). Nullstellensatz Size-Degree Trade-offs from Reversible Pebbling. *Computational Complexity* **30**(4), 1–45. Preliminary version in *CCC '19*.

Juan Luis Esteban & Jacobo Torán (2001). Space Bounds for Resolution. *Information and Computation* **171**(1), 84–97. Preliminary versions of these results appeared in *STACS '99* and *CSL '99*.

Juan Luis Esteban & Jacobo Torán (2003). A combinatorial characterization of treelike resolution space. *Information Processing Letters* **87**(6), 295–300.

Philippe Flajolet, Jean-Claude Raoult & Jean Vuillemin (1979). The Number of Registers Required for Evaluating Arithmetic Expressions. *Theoretical Computer Science* **9**, 99–125. Preliminary version in *FOCS '77*.

Nicola Galesi, Navid Talebanfard & Jacobo Torán (2020). Cops-Robber Games and the Resolution of Tseitin Formulas. *ACM Transactions on Computation Theory* **12**(2), 9:1–9:22. Preliminary version in *SAT '18*.

Matti Järvisalo, Arie Matsliah, Jakob Nordström & Stanislav Živný (2012). Relating Proof Complexity Measures and Practical Hardness of SAT. In *Proceedings of the 18th International Conference on Principles and Practice of Constraint Programming (CP '12)*, volume 7514 of *Lecture Notes in Computer Science*, 316–331. Springer.

Richard Královič (2004). Time and Space Complexity of Reversible Pebbling. *RAIRO – Theoretical Informatics and Applications* **38**(02), 137–161. Preliminary version in *SOFSEM '01*.

Friedhelm Meyer auf der Heide (1981). A Comparison of Two Variations of a Pebble Game on Graphs. *Theoretical Computer Science* **13**(3), 315–322. Preliminary version in *ICALP '79*.

Jakob Nordström (2012). On the Relative Strength of Pebbling and Resolution. *ACM Transactions on Computational Logic* **13**(2), 16:1–16:43. Preliminary version in *CCC '10*.

Jakob Nordström (2015). New Wine into Old Wineskins: A Survey of Some Pebbling Classics with Supplemental Results. Manuscript in preparation. Current draft version available at http://www.csc.kth.se/~jakobn/research/PebblingSurveyTMP.pdf.

Michael S. Paterson & Carl E. Hewitt (1970). Comparative Schematology. In *Record of the Project MAC Conference on Concurrent Systems and Parallel Computation*, 119–127.

Pavel Pudlák & Russell Impagliazzo (2000). A Lower Bound for DLL Algorithms for k-SAT (preliminary version). In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '00)*, 128–136.

Ran Raz & Pierre McKenzie (1999). Separation of the Monotone NC Hierarchy. *Combinatorica* **19**(3), 403–435. Preliminary version in *FOCS '97*.

Alexander A. Razborov (2018). On Space and Depth in Resolution. *Computational Complexity* **27**(3), 511–559.

Jacobo Torán (1999). Lower Bounds for Space in Resolution. In *Proceedings of the 13th International Workshop on Computer Science*

Logic (CSL '99), volume 1683 of *Lecture Notes in Computer Science*, 362–373. Springer.

Jacobo Torán & Florian Wörz (2019). Reversible Pebble Games and the Relation Between Tree-Like and General Resolution Space. Technical Report TR19-097, Electronic Colloquium on Computational Complexity (ECCC). URL https://eccc.weizmann.ac.il/report/2019/097.

Jacobo Torán & Florian Wörz (2020). Reversible Pebble Games and the Relation Between Tree-Like and General Resolution Space. In *Proceedings of the 37th International Symposium on Theoretical Aspects of Computer Science (STACS '20)*, volume 154 of *Leibniz International Proceedings in Informatics (LIPIcs)*, 60:1–60:18. URL https://doi.org/10.4230/LIPIcs.STACS.2020.60.

Grigori Tseitin (1968). The Complexity of a Deduction in the Propositional Predicate calculus. *Zapiski Nauchnyh Seminarov Leningradskogo Otdelenija matematicheskogo Instituta im. V. A. Steklova akademii Nauk SSSR (LOMI)* **8**, 234–259. In Russian.

Alasdair Urquhart (1987). Hard Examples for Resolution. *Journal of the ACM* **34**(1), 209–219.

Alasdair Urquhart (2011). The Depth of Resolution Proofs. *Studia Logica* **99**(1–3), 349–364.

Manuscript received 4 July 2020

Jacobo Torán
Universität Ulm
James-Franck-Ring
89081 Ulm, Germany
jacobo.toran@uni-ulm.de

Florian Wörz
Universität Ulm
James-Franck-Ring
89081 Ulm, Germany
florian.woerz@uni-ulm.de