

Hardness of Approximation in PSPACE and Separation Results for Pebble Games*

Siu Man Chan, Massimo Lauria¹, Jakob Nordström², and Marc Vinyals³

¹Sapienza — Università di Roma, Italy

²University of Copenhagen, Denmark, and Lund University, Sweden

³University of Auckland, New Zealand

May 31, 2023

Abstract

We consider the pebble game on DAGs with bounded fan-in introduced in [Paterson and Hewitt '70] and the reversible version of this game in [Bennett '89], and study the question of how hard it is to decide exactly or approximately the number of pebbles needed for a given DAG in these games.

We prove that the problem of deciding whether s pebbles suffice to reversibly pebble a DAG G is PSPACE-complete, as was previously shown for the standard pebble game in [Gilbert, Lengauer and Tarjan '80]. Via two different graph product constructions we then strengthen these results to establish that both standard and reversible pebbling space are PSPACE-hard to approximate to within any additive constant. To the best of our knowledge, these are the first hardness of approximation results for pebble games in an unrestricted setting (even for polynomial time). Also, since [Chan '13] proved that reversible pebbling is equivalent to the games in [Dymond and Tompa '85] and [Raz and McKenzie '99], our results apply to the Dymond–Tompa and Raz–McKenzie games as well, and from the same paper it follows that resolution depth is PSPACE-hard to determine up to any additive constant.

We also obtain a multiplicative logarithmic separation between reversible and standard pebbling space. This improves on the additive logarithmic separation previously known and could plausibly be tight, although we are not able to prove this.

We leave as an interesting open problem whether our additive hardness of approximation result could be strengthened to a multiplicative bound if the computational resources are decreased from polynomial space to the more common setting of polynomial time.

1 Introduction

In the *pebble game* first studied by Paterson and Hewitt [PH70], one starts with an empty directed acyclic graph (DAG) G with bounded fan-in (and which in this paper in addition will always have a single sink) and places pebbles on the vertices according to the following rules:

- If all (immediate) predecessors of an empty vertex v contain pebbles, a pebble may be placed on v .
- A pebble may be removed from any vertex at any time.

*This is the full-length version of the paper with the same title that appeared in *Proceedings of the 56th IEEE Symposium on Foundations of Computer Science (FOCS '15)*.

The goal is to get a pebble on the sink vertex of G with all other vertices being empty, and to do so while minimizing the total number of pebbles on G at any given time (the *pebbling price* of G). This game models computations with execution independent of the actual input. A pebble on a vertex indicates that the corresponding value is currently kept in memory and the objective is to perform the computation with the minimum amount of memory.

The pebble game has been used to study flowcharts and recursive schemata [PH70], register allocation [Set75], time and space as Turing-machine resources [Coo74, HPV77], and algorithmic time-space trade-offs [Cha73, SS77, SS79, SS83, Tom78]. In the last 10–15 years, there has been a renewed interest in pebbling in the context of proof complexity as discussed in the survey [Nor13] (although in this context one is often interested also in the slightly more general *black-white pebble game* introduced in [CS76]), and pebbling has also turned out to be useful for applications in cryptography [DNW05, AS15]. An excellent overview of pebbling up to ca. 1980 is given in [Pip80] and some more recent developments are covered in the upcoming survey [Nor20].

Bennett [Ben89] introduced the *reversible pebble game* as part of a broader program [Ben73] to investigate possibilities to eliminate (or significantly reduce) energy dissipation in logical computation. Another reason reversible computation is of interest is that observation-free quantum computation is inherently reversible. In the reversible pebble game, the moves performed in reverse order should also constitute a legal pebbling, which means that the rules for pebble placement and removal become symmetric as follows:

- If all predecessors of an empty vertex v contain pebbles, a pebble may be placed on v .
- If all predecessors of a pebbled vertex v contain pebbles, the pebble on v may be removed.

Reversible pebblings of DAGs have been studied in [LV96, Krá04] and have been employed to shed light on time-space trade-offs in reversible simulation of irreversible computation in [LTV98, LMT00, Wil00, BTV01]. In a different line of work Potechin [Pot10] implicitly used the reversible pebble game for proving lower bounds on monotone space complexity, with the connection made explicit in the follow-up works [CP14, FPRC13].

Another pebble game on DAGs that will be of interest in this paper is the *Dymond–Tompa game* [DT85] played on a DAG G by a *Pebbler* and a *Challenger*. This game is played in rounds, with both players starting at the sink in the first round. In the following rounds, Pebbler places a pebble on some vertex of G after which Challenger either stays at the current vertex or moves to the newly pebbled vertex. This repeats until at the end of a round Challenger is standing on a vertex with all (immediate) predecessors pebbled (or on a source, in which case the condition vacuously holds), at which point the game ends. Intuitively, Challenger is challenging Pebbler to “catch me if you can” and wants to play for as many rounds as possible, whereas Pebbler wants to “surround” Challenger as quickly as possible. The *Dymond–Tompa price* of G is the smallest number r such that Pebbler can always finish the game in at most r rounds. The Dymond–Tompa game has been used to establish that for parallel time a speed-up by a logarithmic factor is always possible [DT85], and in [VT89] it was shown that a slightly modified variant of this game exactly characterizes parallelism in complexity classes like AC^i , NC, and P, and can be used to re-derive, for instance, Savitch’s theorem. Furthermore, collapses or separations of these classes can in principle be recast (or discovered) as bounds on Dymond–Tompa price. Interestingly, this characterization of parallelism extends to proof complexity as well as discussed in [Cha13a].

A final game with pebbles that we want to just mention without going into any details is the *Raz–McKenzie game* introduced in [RM99] to study the depth complexity of decision trees solving search problems. The reason for bringing up the Dymond–Tompa and Raz–McKenzie games is that it was shown in [Cha13a] that both games are actually equivalent to the reversible pebble game. Hence, any bounds derived for the reversible pebble game also hold for Dymond–Tompa price and Raz–McKenzie price.

The main focus of this paper is to study how hard it is to decide exactly or approximately the pebbling price of a DAG. For the standard pebble game Gilbert et al. [GLT80] showed that given a DAG G and a

positive integer s it is PSPACE-complete to determine whether space s is sufficient to pebble G or not. It would seem natural to suspect that reversible pebbling price should be PSPACE-complete as well, but the construction in [GLT80] cannot be used to show this.

Given that pebbling price is hard to determine exactly, an even more interesting question is if anything can be said regarding the hardness of approximating pebbling price. Although this seems like a very natural and appealing question, apparently next to nothing has been known about this.

Wu et al. [WAPL14] showed that “one-shot” standard pebbling price is hard to approximate to within any multiplicative constant assuming the so-called Small Set Expansion (SSE) hypothesis. In a one-shot pebbling one is only allowed to pebble each vertex once, however, and this is a major restriction since the complexity now drops from PSPACE-complete to NP-complete [Set75]. Note that containment in NP is easy to see since any one-shot pebbling can be described concisely just by listing the order in which the vertices should be pebbled (and it is easy to compute when a pebble is no longer needed and can be removed). In contrast, in the general case pebbling strategies that are optimal with respect to space can sometimes provably require exponential time.

One can also go in the other direction and study more general pebble games, such as the AND/OR pebble game introduced by Lingas [Lin78] in one of the works leading up to [GLT80]. Here every vertex is labelled AND or OR. For AND-vertices we have the usual pebbling rule, but for OR-vertices it is sufficient to just have one pebble on some predecessor in order to be allowed to pebble the vertex. This game has a relatively straightforward reduction from hitting set [FNPW10], which shows that it is hard to approximate to within a logarithmic factor, but the reduction crucially depends on the OR-nodes.

We remark that hardness of approximation in PSPACE for other problems has been studied in [CFLS95], but those techniques seem hard to adapt to pebble games since the reduction from QBF to pebbling is inherently unable to preserve gaps.

Another problem that we study in the current paper is the relation between standard pebbling price and reversible pebbling price. Clearly, the space needed to reversibly pebble a graph is at least the space required in the standard pebble game. It is also not hard to see that there are graphs that require strictly more pebbles in a reversible setting: for a directed path on n vertices only 2 pebbles are needed in the standard game, while it is relatively straightforward to show that the reversible pebbling space is $\Theta(\log n)$ [Ben89, LV96]. However, for “classic” graphs studied in the pebbling literature, such as binary trees, pyramids, certain superconcentrators, and the worst-case graphs in [PTC77], the reversible and standard pebbling prices coincide asymptotically, and are sometimes markedly closer than an additive logarithm apart.

This raises the question whether reversible and standard pebbling can be asymptotically separated with respect to space. It might be worth pointing out in this context that for Turing machines it was proven in [LMT00] that any computation can be simulated reversibly in exactly the same space. In the more restricted pebbling model, it was shown in [Kra04] that if the standard pebbling price of a DAG G on n vertices is s , then G can be reversibly pebbled with at most $s^2 \log n$ pebbles. Thus, if there is not only an additive but also a multiplicative separation between standard and reversible pebbling price, such a separation cannot be too large.

1.1 Our Results

We obtain the following results:

1. We establish an asymptotic separation between standard and reversible pebbling by exhibiting families of graphs $\{G_n\}_{n=1}^\infty$ of size $\Theta(n)$ with a single sink and fan-in 2 which have standard pebbling price $s(n)$ and reversible pebbling price $\Omega(s(n) \log n)$. This construction works for any $s(n) = O(n^{1/2-\epsilon})$ with $\epsilon > 0$ constant, where the constant hidden in the asymptotic notation in the lower bound has a (mild) dependence on ϵ .

2. We prove that determining reversible pebbling price is PSPACE-complete. That is, given a single-sink DAG G of fan-in 2 and a parameter s , it is PSPACE-complete to decide whether G can be reversibly pebbled in space s or not.
3. Finally, we present two different graph products (for standard and reversible pebbling, respectively) that take DAGs G_i of size n_i with pebbling price s_i for $i = 1, 2$ and yield a DAG of size $O((n_1 + n_2)^2)$ with pebbling price $s_1 + s_2 + K_p$ (for $K_p = \pm 1$ depending on the flavour of the pebble game). Combining these graph products with the PSPACE-completeness results for standard pebbling in [GLT80] and reversible pebbling in item 2, we deduce that for any fixed K the promise problem of deciding for a DAG G (with a single sink and fan-in 2) whether it can be pebbled in space s or requires space $s + K$ is PSPACE-hard in both the standard and the reversible pebble game.

We need to provide more formal definitions before going into a detailed discussion of techniques, but want to stress right away that a key feature of the above results is the bounded fan-in condition. This is the standard setting for pebble games in the literature and is also crucial in most of the applications mentioned above. Without this constraint it would be much easier, but also much less interesting, to prove our results.¹

Another aspect worth pointing out is that although the reversible pebble game is weaker than the standard pebble game, it is technically much more challenging to analyze. The reason for this is that a standard pebbling will always progress in a “forward sweep” through the graph in topological order, and so one can often assume without loss of generality that once one has pebbled through some subgraph the pebbling will never touch this subgraph again. For a reversible pebbling this is not so, since any pebble placed on any descendant of vertices in the subgraph will also have to be removed at some later time, and this has to be done in reverse topological order. Therefore, in any reversible pebbling there will be alternating phases of “forward sweeps” and “reverse sweeps,” and these phases can also be interleaved at various levels. For this reason, controlling the progress of a reversible pebbling is substantially more complicated. Despite the additional technical difficulties, however, we consider the reversible pebble game to be at least as interesting to study as the standard and black-white pebble games in view of its tight connection with parallelism in circuit and proof complexity as described in [Cha13a].

1.2 Follow-up Work

Our hardness of approximation result for the standard pebble game was improved by Demaine and Liu [DL17], who proved that it is PSPACE-hard to approximate the standard space of a graph of size n within an additive $n^{1/3-\epsilon}$ term.

1.3 Organization of This Paper

We present the necessary preliminaries in Section 2 and then give a detailed overview of our results in Section 3. We prove an asymptotic separation between standard and reversible pebbling in Section 4. In Section 5 we compute the exact price of some classic graphs, trees and pyramids, that we use in Section 6 to construct technical gadgets. These play a key role in Section 7, where we show that reversible pebbling is PSPACE-complete. We detail the graph product for reversible pebbling in Section 8 and its counterpart for standard pebbling in Section 9. Some concluding remarks are presented in Section 10.

¹The reason to emphasize this is that for unbounded fan-in the first author proved a PSPACE-completeness result for reversible pebbling in [Cha13b], but this result uses simpler constructions and techniques that do not transfer to the bounded fan-in setting. Another, somewhat related, example is that deciding space in the black-white pebble game has also been shown to be PSPACE-complete for unbounded indegree in [HP10], but there the unbounded fan-in can be used to eliminate the white pebbles completely, and again the techniques fail to transfer to the bounded indegree case.

2 Preliminaries

All logarithms in this paper are base 2 unless otherwise specified. For a positive integer n we write $[n]$ to denote the set of integers $\{1, 2, \dots, n\}$. We use Iverson bracket notation

$$\llbracket B \rrbracket = \begin{cases} 1 & \text{if the Boolean expression } B \text{ is true;} \\ 0 & \text{otherwise;} \end{cases} \quad (2.1)$$

to convert Boolean values to integer values.

2.1 Boolean Formula Notation and Terminology

A *literal* a over a Boolean variable x is either the variable x itself or its negation \bar{x} (a *positive* or *negative* literal, respectively). A *clause* $C = a_1 \vee \dots \vee a_k$ is a disjunction of literals. A *k-clause* is a clause that contains at most k literals. A formula F in *conjunctive normal form (CNF)* is a conjunction of clauses $F = C_1 \wedge \dots \wedge C_m$. A *k-CNF formula* is a CNF formula consisting of k -clauses. We think of clauses and CNF formulas as sets, so that the order of elements is irrelevant and there are no repetitions.

A *quantified Boolean formula (QBF)* is a formula $\phi = Q_1 x_1 Q_2 x_2 \dots Q_n x_n F$, where F is a CNF formula over variables x_1, \dots, x_n and $Q_i \in \{\forall, \exists\}$ are universal or existential quantifiers (i.e., the formula is in prenex normal form with all variables bound by quantifiers). It was shown in [SM73] that it is PSPACE-complete to decide whether a QBF is true or not (where we can assume without loss of generality that F is a 3-CNF formula).

2.2 Graph Notation and Terminology

We write $G = (V, E)$ to denote a graph with vertices $V(G) = V$ and edges $E(G) = E$. All graphs in this paper are directed acyclic graphs (DAGs). An edge $(u, v) \in E(G)$ is an *outgoing edge* of u and an *incoming edge* of v , and we say that u is a *predecessor* of v and that v is a *successor* of u . We write $\text{pred}_G(v)$ to denote the set of all predecessors of v in G and $\text{succ}_G(v)$ to denote all its successors. Vertices with no incoming edges are called *sources* and vertices with no outgoing edges are called *sinks*. For brevity, we will sometimes refer to a DAG with a unique sink as a *single-sink DAG*, and this sink will usually be denoted z .

Taking the transitive closures of the predecessor and successor relations, we define the *ancestors* $\text{anc}_G(v)$ of v to be the set of vertices that have a path to v and the *descendants* $\text{desc}_G(v)$ to be the set of vertices on some path from v . By convention, v is an ancestor and descendant of itself. We write $\text{anc}_G^*(v) = \text{anc}_G(v) \setminus \{v\}$ and $\text{desc}_G^*(v) = \text{desc}_G(v) \setminus \{v\}$ to denote the *proper ancestors* and *proper descendants* of v , respectively. These concepts are extended to sets of pairwise incomparable vertices by taking unions so that $\text{anc}_G(U) = \bigcup_{u \in U} \text{anc}_G(u)$, $\text{anc}_G^*(U) = \bigcup_{u \in U} \text{anc}_G^*(u)$, et cetera, where we say that the vertices in U are pairwise incomparable when no vertex in the set is an ancestor of any other vertex in the set. When the graph G is clear from context we will sometimes drop it from the notation.

2.3 Standard and Reversible Pebble Games

A *pebble configuration* on a DAG $G = (V, E)$ is a subset of vertices $\mathbb{P} \subseteq V$. We consider the following three rules for manipulating pebble configurations:

1. $\mathbb{P}' = \mathbb{P} \cup \{v\}$ for $v \notin \mathbb{P}$ such that $\text{pred}_G(v) \subseteq \mathbb{P}$ (a *pebble placement* on v).
2. $\mathbb{P}' = \mathbb{P} \setminus \{v\}$ for $v \in \mathbb{P}$ (a *pebble removal* from v).
3. $\mathbb{P}' = \mathbb{P} \setminus \{v\}$ for $v \in \mathbb{P}$ such that $\text{pred}_G(v) \subseteq \mathbb{P}$ (a *reversible pebble removal* from v).

A *standard pebbling* from \mathbb{P}_0 to \mathbb{P}_τ is a sequence of pebble configurations $\mathcal{P} = (\mathbb{P}_0, \mathbb{P}_1, \dots, \mathbb{P}_\tau)$ where each configuration is obtained from the preceding one by the rules 1 and 2 while in a *reversible pebbling* rules 1 and 3 should be used. The *time* of a pebbling $\mathcal{P} = (\mathbb{P}_0, \dots, \mathbb{P}_\tau)$ is $\text{time}(\mathcal{P}) = \tau$, and the *space* is $\text{space}(\mathcal{P}) = \max_{0 \leq t \leq \tau} \{|\mathbb{P}_t|\}$.

We say that a pebbling is *unconditional* if $\mathbb{P}_0 = \emptyset$ and *conditional* otherwise. The *pebbling price* $\text{Peb}_G(\mathbb{P})$ of a pebble configuration \mathbb{P} is the minimum space of any unconditional standard pebbling on G ending in $\mathbb{P}_\tau = \mathbb{P}$, and we define the *reversible pebbling price* $\text{RPeb}_G(\mathbb{P})$ by taking the minimum over all unconditional reversible peblings reaching \mathbb{P} . The pebbling price of a single-sink DAG G with sink z is $\text{Peb}(G) = \text{Peb}_G(\{z\})$, and the reversible pebbling price of G is $\text{RPeb}(G) = \text{RPeb}_G(\{z\})$. We refer to such peblings as (*complete*) *peblings of G* or *pebbling strategies for G* . Again, when G is clear from context we can drop it from the notation, and from now on we will usually abuse notation by omitting the curly brackets around singleton vertex sets.

For technical reasons, we will often be interested in distinguishing particular flavours of reversible peblings. Suppose that v is a vertex in G and that $\mathcal{P} = (\mathbb{P}_0 = \emptyset, \mathbb{P}_1, \dots, \mathbb{P}_\tau)$ is a reversible pebbling. We will use the following terminology and notation:

- \mathcal{P} is a *visiting pebbling* of v if $v \in \mathbb{P}_\tau$. The *visiting price* $\text{RPeb}^V(v)$ of v is the minimal space of any such pebbling.
- \mathcal{P} is a *surrounding pebbling* of v if $\text{pred}(v) \subseteq \mathbb{P}_\tau$ and the *surrounding price* $\text{RPeb}^S(v)$ is the minimal space of any such pebbling.
- \mathcal{P} is a *persistent pebbling* of v if it is a reversible pebbling of v in the sense defined before, i.e., such that $\mathbb{P}_\tau = \{v\}$. We will sometimes refer to $\text{RPeb}(v)$ as the *persistent price* of v to distinguish it from the visiting and surrounding prices.

We also define the visiting price for a single-sink DAG G with sink z as $\text{RPeb}^V(G) = \text{RPeb}_G^V(z)$ and the surrounding price as $\text{RPeb}^S(G) = \text{RPeb}_G^S(z)$.

Note that because of reversibility we could obtain exactly the same visiting space measure by defining a visiting pebbling of v to be a pebbling $\mathcal{P} = (\mathbb{P}_0, \mathbb{P}_1, \dots, \mathbb{P}_\tau)$ such that $\mathbb{P}_0 = \mathbb{P}_\tau = \emptyset$ and $v \in \bigcup_{0 \leq t \leq \tau} \mathbb{P}_t$, and let the visiting price be the minimal space of any such pebbling. This is because once we have reached a configuration containing v we can simply run the pebbling backwards (because of reversibility) until we reach the empty configuration again. We can therefore think of a pebbling as *visiting v* if there is a pebble on v at some point but this pebble does not stay on v until the end of the pebbling. In a *persistent* pebbling the pebble remains on v until all other pebbles have been removed. A *surrounding* pebbling, finally, is a pebbling that reaches exactly the point where a pebble could be placed on v , since all its predecessors are covered by pebbles (i.e., v is “surrounded” by pebbles), but where v is not necessarily pebbled.

It is not hard to see that for a single-sink DAG G we have the inequalities

$$\text{Peb}(G) \leq \text{RPeb}^V(G) \tag{2.2}$$

and

$$\text{RPeb}^S(G) \leq \text{RPeb}^V(G) \leq \text{RPeb}(G) . \tag{2.3}$$

Perhaps slightly less obviously, we also have the following useful equality.

Proposition 2.1. *For any vertex v in a DAG G it holds that $\text{RPeb}^S(v) = \text{RPeb}(v) - 1$.*

Proof. To see that $\text{RPeb}(v) \leq \text{RPeb}^S(v) + 1$ consider a surrounding pebbling \mathcal{P}^S of space $\text{RPeb}^S(v)$. Let \mathcal{P}^* be the pebbling which first runs \mathcal{P}^S to surround v , then puts a pebble on v , and finally runs the reverse of \mathcal{P}^S to “un-surround” v (while keeping the pebble on v). Since \mathcal{P}^* is a persistent pebbling of space $\text{RPeb}^S(v) + 1$, the inequality follows.

We now prove that $RPeb^S(v) \leq RPeb(v) - 1$. Consider a persistent pebbling \mathcal{P} for v of space $RPeb(v)$. Let t be the last time that a pebble is put on v . Then vertex v is surrounded at time t , and there is a pebble on v since time t . Let $\mathcal{P}_{\geq t}$ be the conditional pebbling obtained from \mathcal{P} after time t , with the modification that vertex v has no pebble throughout $\mathcal{P}_{\geq t}$, and let $\mathcal{P}_{\geq t}^R$ be this pebbling run in reverse. Then $\mathcal{P}_{\geq t}^R$ is a surrounding pebbling in space at most $RPeb(v) - 1$, and the inequality follows. \square

2.4 The Dymond–Tompa and Raz–McKenzie Games

As described above, the *Dymond–Tompa game* on a single-sink DAG G is played in rounds by two players *Pebbler* and *Challenger*. In the first round Pebbler places a pebble on the sink z and Challenger challenges this vertex. In all subsequent rounds, Pebbler places a pebble on an arbitrary empty vertex and Challenger chooses to either challenge this new vertex (which we refer to as *jumping*) or to re-challenge the previously challenged vertex (referred to as *staying*). The game ends when at the end of a round all the (immediate) predecessors of the currently challenged vertex are covered by pebbles.² The *Dymond–Tompa price* $DT(G)$ of G is the maximal number of pebbles r needed for Pebbler to finish the game, or expressed differently the smallest number r such that Pebbler has a strategy to make the game end in at most r rounds regardless of how Challenger plays.

Let us also for completeness describe the *Raz–McKenzie game*, which is also played on a single-sink DAG G by two players *Pebbler* and *Colourer*. In the first round Pebbler places a pebble on the sink z and Colourer colours it red. In all subsequent rounds, Pebbler places a pebble on an arbitrary empty vertex and Colourer then colours this new pebble either red or blue. The game ends when there is a vertex with a red pebble, while all its predecessors in the graph have blue pebbles. The *Raz–McKenzie price* $RM(G)$ of G is the smallest number r such that Pebbler has a strategy to make the game end in at most r rounds regardless of how Colourer plays.

The intuition for this game is that the vertices on the graphs have assigned values true (blue) or false (red), with the condition that each vertex has value equal to the conjunction of the values of its predecessors. Colourer claims that the sink is false, but the above condition vacuously implies that all source vertices must be true. Colourer loses when Pebbler discovers a violation of the condition. Pebbler wants to find the violation as soon as possible, while Colourer wants to fool Pebbler for as long as possible.

In [Chal13a] the first author proved that the equalities

$$DT(G) = RM(G) = RPeb(G) \tag{2.4}$$

hold for any single-sink DAG G , i.e., that the reversible pebbling price, the Dymond–Tompa price and the Raz–McKenzie price all coincide. Thus, any result we prove for one of these games is also guaranteed to hold for the other games. The above equalities are very convenient in that they allow us to switch back and forth between the reversible pebble game and the Dymond–Tompa game (or Raz–McKenzie game) when proving upper and lower bounds, depending on which perspective is more suitable at any given time. In particular, when proving lower bounds for reversible pebbings it is often helpful to do so by devising good Challenger strategies in the Dymond–Tompa game. One final technical remark in this context is that in all such strategies that we construct it holds that Challenger will either stay or jump to an ancestor of the currently challenged vertex. Because of this we can assume without loss of generality that Pebbler only pebbles vertices in the subgraph consisting of ancestors of the currently challenged vertex. If Pebbler pebbles some vertex outside of this subgraph Challenger will just stay put on the current vertex, and so Pebbler just wastes a round.

²We remark that our description follows [Chal13a] and thus differs slightly from the original definition in [DT85], but the two versions are equivalent for all practical purposes.

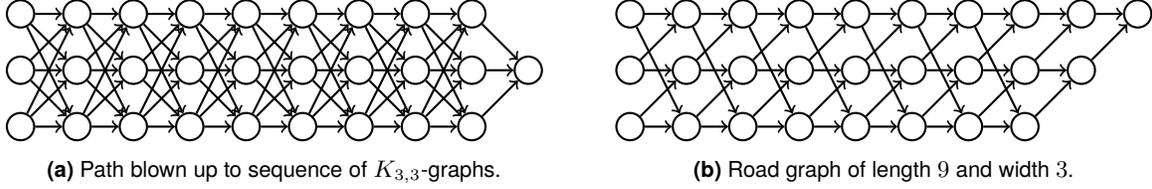


Figure 1: Modifications of path graphs to amplify difference between reversible and standard pebbling price.

3 Overview of Results and Sketches of Proofs

In this section we give a detailed overview of our results and also sketch some of the main ideas in the proofs. In the rest of the paper, we then provide all the missing technical definitions and present the actual formal proofs.

3.1 Separation Between Standard and Reversible Pebbling

As mentioned in Section 1, the strongest separation hitherto known between standard and reversible pebbling is for the length- ℓ path on vertices $\{v_1, v_2, \dots, v_{\ell+1}\}$ with edges (v_i, v_{i+1}) for all $i \in [\ell]$, which has a standard pebbling with 2 pebbles whereas reversible pebbings require space $\Theta(\log \ell)$ [Ben89, LV96]. We give a simple construction improving this to a multiplicative logarithmic separation.

Theorem 3.1. *For any function $s(n) = O(n^{1/2-\epsilon})$ for $\epsilon > 0$ constant there are DAGs $\{G_n\}_{n=1}^\infty$ of size $\Theta(n)$ with a single sink and fan-in 2 such that $\text{Peb}(G) = O(s(n))$ and $\text{RPeb}(G) = \Omega(s(n) \log n)$ (where the hidden constant depends linearly on ϵ).*

A first observation is that if we did not have the bounded fan-in restriction, Theorem 3.1 would be very easy. In such a case we could just take the path of length ℓ , blow up every vertex v_i to s vertices v_i^1, \dots, v_i^s , and add edges $(v_i^j, v_{i+1}^{j'})$ for all $j, j' \in [s]$, so that we get a sequence of complete bipartite graphs $K_{s,s}$ glued together as shown in Figure 1a. It is not hard to show that any reversible pebbling of this DAG would have to do s parallel, synchronized pebblings of the paths $\{v_1^j, v_2^j, \dots, v_{\ell+1}^j\}$ for $j \in [s]$, which would require space $\Omega(s \log \ell)$, whereas a standard pebbling would clearly only need space $O(s)$.

For bounded indegree it is not a priori clear what to do, however, or indeed whether there should even be a multiplicative separation. But it turns out that one can actually simulate a lower bound proof along the same lines as above by considering a layered graph as in Figure 1b, with s parallel paths of length up to ℓ and with every path having an extra edge fanning out to its “neighbour path” above (or at the bottom for the top row) at each level. We will refer to this construction as a *road graph* of length ℓ and width s (where a path is a maximally narrow road of width 1). It is easy to verify that the standard pebbling price of a road of width $s \geq 2$ is $s + 2$. We claim that the reversible pebbling price is $\Omega(s \log(\ell/s))$, from which Theorem 3.1 follows.

To prove the reversible pebbling lower bound it is convenient to think instead in terms of Challenger strategies in the Dymond–Tomba game. The idea is that Challenger will stay put on the sink until Pebbler has pebbled enough vertices so that there are no pebble-free paths from any source vertex to the sink. Intuitively, the cheapest way for Pebbler to disconnect the graph is with a straight cut over some layer. When this happens, Challenger looks at the latest pebbled vertex and compares the subgraph between the sources and the cut with the subgraph between the cut and the sink. If more than half of the graph is before the cut, Challenger jumps to the latest pebbled vertex. If not, Challenger stays on the sink. This strategy is then repeated on a graph of at least half the length. Since every cut by Pebbler requires s pebbles, Challenger can survive for roughly $s \log \ell$

rounds (except that the rigorous argument is not quite this simple, and the slightly smaller factor $\log(\ell/s)$ in the formal statement of the theorem is in fact inherent).

3.2 PSPACE-Completeness of Reversible Pebbling

Moving on to technically more challenging material, let us next discuss our PSPACE-completeness result for reversible pebbling, which we restate here more formally for the record.

Theorem 3.2. *Given a single-sink DAG G of fan-in 2 and a parameter s , it is PSPACE-complete to decide whether G can be reversibly pebbled in space s or not. In more detail, given a QBF $\phi = Q_1x_1 Q_2x_2 \dots Q_nx_n F$, where F is a 3-CNF formula over variables x_1, \dots, x_n , there is a polynomial-time constructible single-sink graph $G(\phi)$ of fan-in 2 and a polynomial-time computable number $\gamma(\phi)$ such that $\text{RPeb}(G(\phi)) = \gamma(\phi) + \llbracket \phi \text{ is FALSE} \rrbracket$.*

At a high level, our proof is similar to that in [GLT80] for standard pebbling: we build gadgets for variables, clauses, and universal and existential quantifiers, and then glue them together in the right way so that pebbling through the gadgets corresponds to verifying satisfying assignments for universally and existentially quantified subformulas of the QBF ϕ . However, the execution of this simple idea is highly nontrivial even in [GLT80], and we run into several additional technical difficulties when we want to do an analogous reduction for reversible pebbling.

For starters, since the difference in pebbling price for graphs $G(\phi)$ obtained from true and false QBFs ϕ is just an additive 1, we need exact control over the pebbling price of all components used in the reduction. For standard pebbling there is no problem here—exact bounds on pebbling price are known for quite a wide selection of graphs—but in the reversible setting this becomes an issue already for almost the simplest possible graph: the complete binary tree of height h . An easy inductive argument shows that the standard pebbling price of such a tree is exactly $h + 2$. Since reversible pebbles find paths more challenging than do standard pebbles, one could perhaps expect an extra additive $\log h$ or so in the reversible pebbling bound. However, the asymptotically correct bound turns out to be $h + \Theta(\log^* h)$ as shown in [Krá04], and the upper and lower bounds on the multiplicative constant obtained in that paper are far from tight.

The story is even worse for the workhorse of the construction in [GLT80] (and many other pebbling results), namely *pyramids* of height h , which have i vertices at level i for $i = 1, \dots, h + 1$, and where the j th vertex at level i has incoming edges from the j th and $(j + 1)$ st vertices at level $i + 1$. There is a very neat proof in [Coo74] that the standard pebbling price is again exactly $h + 2$, but for reversible pebbling price nothing has been known except that it has to be somewhere between $h + 2$ and $h + O(\log^* h)$ (where the latter bound follows since any strategy for a complete binary tree of height h works for any DAG of height h). As a crucial first step towards establishing Theorem 3.2, we exactly determine the reversible pebbling price of pyramids (and also binary trees).

Theorem 3.3. *For Δ denoting a positive integer, let g be the function defined recursively as*

$$g(\Delta) = \begin{cases} 0 & \text{if } \Delta = 1; \\ 2^{g(\Delta-1)+\Delta-2} + g(\Delta - 1) & \text{otherwise;} \end{cases}$$

and let the inverse g^{-1} of this function be defined as

$$g^{-1}(h) = \min\{\Delta \mid g(\Delta) \geq h\} .$$

Then the persistent pebbling price of a pyramid of height h , as well as of a complete binary tree of height h , is $h + g^{-1}(h)$, where g^{-1} is efficiently computable.

Even though Theorem 3.3 is an important step, we immediately run into new problems when trying to use it as a building block in our reduction for reversible pebbling. In the standard pebble game a complete pebbling is any pebbling that reaches the sink. For the reversible game there is a subtle distinction in that we can ask whether it is sufficient to just reach the sink or whether the rest of the graph must also be cleared of pebbles. As discussed in Section 2, this leads to two different flavours of reversible pebbings, namely *persistent pebbings*, which leave a pebble on the sink with the rest of the graph being empty, and *visiting pebbings*, which just reach the sink (and can then be thought to run in reverse after having visited the sink to clear the whole graph including the sink from pebbles). The pebbings we actually care about are the persistent ones, but we cannot rule out the possibility that subpebbings of gadgets are visiting pebbings. Clearly, the difference in pebbling space is at most 1, but this is exactly the additive 1 of which we cannot afford to lose control! To make things worse, for pyramids it turns out that persistent and visiting pebbling prices actually do differ except in very rare cases.

Because of this, we have to build more involved graph gadgets for which we can guarantee that visiting and persistent prices coincide. These gadgets are constructed in two steps. First, we take a pyramid and append a path of suitable length, depending on the height of the pyramid, to the pyramid sink, resulting in a graph that we call a *teabag*. Second, we take such teabags of smaller and smaller size and stack them on top of one another, which yields a graph that looks a bit like a *Christmas tree*. These Christmas tree graphs are guaranteed to have the same pebbling price regardless of whether a reversible pebbling is visiting or persistent.

With this in hand we are almost ready to follow the approach in the PSPACE-completeness reduction for standard pebbling in [GLT80]. The idea is that we want to build gadgets for the quantifiers in a formula $\phi = \forall x \exists y \dots Qz F$ of specified pebbling price so that the only way to pebble the graph $G(\phi)$ without using too much space is to first pebble the gadget for $\forall x$, then $\exists y$, et cetera, in the correct order until all quantifier gadgets have been pebbled. Once we get to the clause gadgets, we would like that the pebbles in the quantifier gadgets are locked in place encoding a truth value assignment to the variables, and that the only way to pebble through the clause gadgets without exceeding the space budget is if every clause contains at least one literal satisfied by this truth value assignment.

In order to realize this plan, there remains one more significant technical obstacle to overcome, however. To try to explain what the issue is, we need to discuss the PSPACE-completeness reduction in [GLT80] in slightly more detail. The way this reduction imposes an order in which the quantifier gadgets have to be pebbled is that pyramid graphs are included “at the bottom” of the gadgets (i.e., topologically first in order). The source vertices of the quantifier gadgets all appear in such pyramids, and one has to pebble through these pyramids to reach the rest of a gadget (where pebble placements encode variable assignments as mentioned above).

In the first, outermost quantifier gadget the pyramids have large height. In the second gadget the pyramid heights are slightly smaller, et cetera, down to the last, innermost quantifier gadget where the pyramids have smallest height. In this way, the pyramids are used to “lock up” pebbles and force a strict order of pebbling of the gadgets. It can be shown that in order not to exceed the pebbling space budget, any pebbling strategy has to start by pebbling the highest pyramids in the first gadget. If the pebbling starts anywhere else in the graph, this will mean that there are already pebbles elsewhere in the graph when the pebbling strategy reaches the first, highest pyramids in the outermost quantifier, but if so the overall pebbling has to use up too much space to pebble through this pyramid. One can also show that once the pyramids in the outermost quantifier gadget have been pebbled, the pebbling cannot proceed until the next quantifier gadget is pebbled. The pyramids in this gadget have smaller height, but there are also pebbles stuck in place in the outermost gadget, meaning that pyramids must again be pebbled in exactly the right order to stay within the space budget.

These properties can be used to *normalize* pebbling strategies in the standard pebble game. Without loss of generality, one can assume that any strategy that starts pebbling a pyramid in a gadget will complete this local pebbling in one go, leaving a pebble at the sink of the pyramid, and will not place pebbles anywhere else



Figure 2: Legend for technical gadget building blocks.

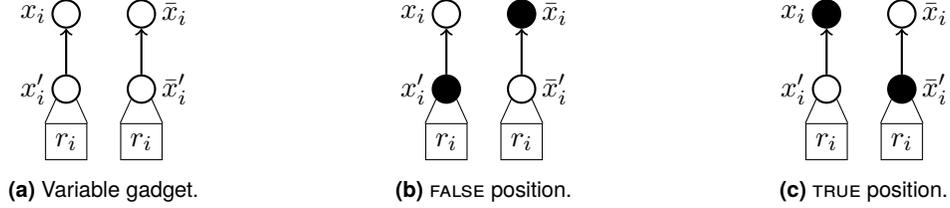


Figure 3: Gadget for variable x_i and pebble positions corresponding to truth value assignments.

until the pebbling of the pyramid has been completed. Also, once a pyramid in a quantifier gadget has been pebbled in this way, one can prove that it will never be pebbled again since there is now at least one additional pebble at some vertex later in the topological order in the graph, and a repeated pebbling of the pyramid in question would therefore exceed the space budget. Thus, not only do the pyramids enforce that the gadgets are pebbled in the right order—they also serve as single-entry access points to the gadgets, making sure that each gadget is pebbled exactly once.

There is no hope of building gadgets with such properties in a reversible pebbling setting. It is simply not true that a reversible pebbling will pebble through a subgraph and then never return. Instead, as already discussed reversible pebblings will proceed in alternating phases of interleaved “forward sweeps” and “reverse sweeps,” and subgraphs will be entered also in reverse topological order. Therefore, it is not sufficient to add “space-locking” subgraphs at the source vertices of the gadgets. Rather, we have to insert “single-passage points” inside and in between the gadgets for quantifiers and clauses. We obtain such subgadgets by further tweaking our Christmas tree construction so that it can also connect two vertices in such a way that any pebbling has to “pay a toll” to go through this subgraph. We cannot describe these gadgets, which we call *turnpikes*, in detail here, but mention that the “space-locking” property that they have is that when the entrance vertex is eliminated by having a pebble placed on that vertex, then the cost of pebbling through the rest of the turnpike drops by 1. This is critically used in the subgraph compositions described next.

Assuming the existence of the necessary technical subgraph constructions sketched above, we can now describe the overall structure of our reduction from quantified Boolean formulas to reversible pebbling (where all parameters shown in the figures are fixed appropriately in the formal proofs). In the following figures we denote a Christmas tree of (visiting and persistent) pebbling price r by the symbol in Figure 2a, where we only display the sink vertex. We denote the turnpike gadget just discussed by the symbol in Figure 2b. We write r to denote the *toll* parameter of the turnpike, where a turnpike with toll r has persistent price $r + 2$, but only $r + 1$ if we do not count the source a as part of the turnpike.

For every variable x_i we have a *variable gadget* as shown in Figure 3a, where we think of a truth value assignment ρ as represented by pebbles on vertices $\{\bar{x}_i, x'_i\}$ when $\rho(x_i) = \text{FALSE}$ and on $\{x_i, \bar{x}'_i\}$ when $\rho(x_i) = \text{TRUE}$, as shown in Figures 3b and 3c, respectively.

For every clause C_j we have a *clause gadget* as depicted in Figure 4a. The vertices labelled $\ell'_{j,k}$ and $\ell_{j,k}$ in Figure 4a are identified with the corresponding vertices for the positive or negative literal $\ell_{j,k}$ in the variable

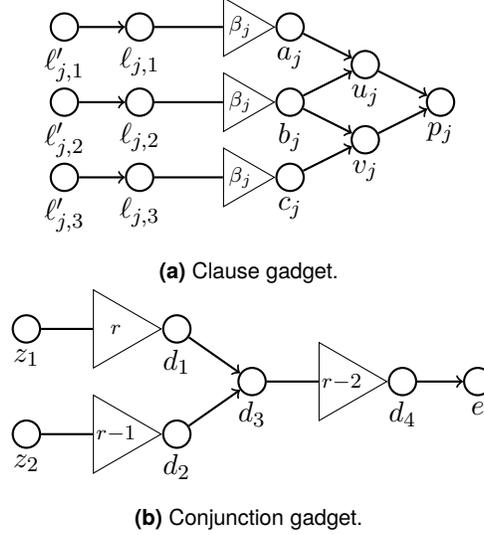


Figure 4: Gadgets for clauses and CNF formulas.

gadget in Figure 3a. If ρ satisfies a literal, then there is a pebble on the entrance vertex of the corresponding turnpike, meaning that we can pebble through the gadget for a clause containing that literal with one less pebble than if ρ does not satisfy the clause.

To build the subgraph corresponding to a 3-CNF formula $F = \bigwedge_{j=1}^m C_j$ we join clause gadgets sequentially using the *conjunction gadget* in Figure 4b. For technical reasons we start by joining a dummy graph with the first clause gadget, then we join the result to the second clause gadget, and so on up to the m th clause of F . The resulting graph has the property that if pebbles are placed on the variable gadgets according to an assignment ρ that satisfies F , then the number of additional pebbles needed to pebble the graph is one less than if the assignment is falsifying.

Finally we have one *quantifier gadget* for each variable. To describe this part of the construction, we sort the variables indices in reverse order from the innermost to the outermost quantifier and denote by ϕ_i the subformula with just the i innermost quantifiers, so that $\phi_0 = F = \bigwedge_{j=1}^m C_j$, $\phi_i = Q_i x_i \phi_{i-1}$ for $Q_i \in \{\forall, \exists\}$, and $\phi = \phi_n$. We construct graphs $G^{(i)} := G(\phi_i)$, starting with $G^{(0)}$ which is just the subgraph corresponding to the CNF formula F . To construct $G^{(i+1)}$ from $G^{(i)}$ we add an existential gadget as in Figure 5a if x_i is existentially quantified and a universal gadget as in Figure 5b if x_i is universally quantified. An example of the full construction can be found in Figure 6.

Given this construction we argue along the same lines as in in [GLT80], although as mentioned above there are numerous additional technical complications that we cannot elaborate on in this brief overview of the proof. We show that given an assignment ρ_i to $\{x_n, \dots, x_{i+1}\}$, the number of additional pebbles needed to pebble $G^{(i)}$ differs by 1 depending on whether ϕ_i is true under the assignment ρ_i or not. An existential gadget can be optimally pebbled by setting x_i to any value that satisfies ϕ_{i-1} . To pebble a universal gadget one needs to assign x_i to some value, pebble through the gadget, unset x_i and assign it to the opposite value, and finally pebble through the gadget again, and both assignments to x_i must yield satisfying assignments to ϕ_{i-1} in order for the pebbling not to go over budget. Proceeding by induction, we establish that the complete graph $G^{(n)}$ can be pebbled within the specified space budget only if $\phi = \phi_n$ is true, which yields Theorem 3.2.

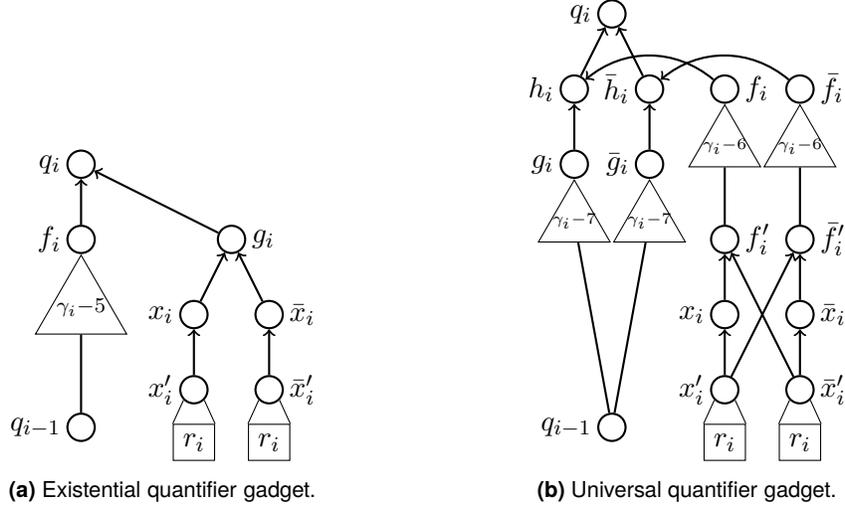


Figure 5: Quantifier gadgets for variable x_i .

3.3 PSPACE-Inapproximability up to Additive Constants

Let us conclude the detailed overview of our contributions by describing what is arguably the strongest result in this paper, namely a strengthening of the PSPACE-completeness of standard pebbling in [GLT80] and of reversible pebbling in Theorem 3.2 to PSPACE-hardness results for approximating standard and reversible pebbling price to within any additive constant K .

Theorem 3.4. *For any fixed positive integer K it is PSPACE-complete to decide whether a single-sink DAG G with fan-in 2 has (standard or reversible) pebbling price at most s or at least $s + K$.*

We remark that it would of course have been even nicer to prove multiplicative hardness results. We want to stress again, though, that to the best of our knowledge these are the first results ever for hardness of approximation of pebble games in a general setting. The fact that these results hold even for PSPACE could perhaps be taken both as an indication that it should be possible to prove much stronger hardness results for algorithms limited to polynomial time, and as a challenge to do so.

We obtain Theorem 3.4 by defining and analyzing two graph product constructions, one for standard and one for reversible pebbling, which take two graphs and output product graphs with pebbling price equal to the sum of the pebbling prices of the two input graphs (except for an additive adjustment). These graph products can then be applied iteratively $K - 1$ times to the graphs obtained by the reductions from QBFs. In the next theorem we state the formal properties of these graph products.

Theorem 3.5. *Given single-sink DAGs G_i of fan-in 2 and size n_i for $i = 1, 2$, there are polynomial-time constructible single-sink DAGs $\mathcal{S}(G_1, G_2)$ and $\mathcal{R}(G_1, G_2)$ of fan-in 2 and size $O((n_1 + n_2)^2)$ such that*

- *For standard pebbling price it holds that $\text{Peb}(\mathcal{S}(G_1, G_2)) = \text{Peb}(G_1) + \text{Peb}(G_2) - 1$.*
- *For reversible pebbling price it holds that $\text{RPeb}(\mathcal{R}(G_1, G_2)) = \text{RPeb}(G_1) + \text{RPeb}(G_2) + 1$.*

In the remainder of this section we try to convey some of the flavour of the arguments used to prove Theorem 3.5 and to give a sense of some of the technical obstacles that have to be overcome during the analysis. In what follows, we will mostly focus on the reversible pebble game, since it is the technically more challenging and therefore also the more interesting case. We will briefly discuss the product construction for

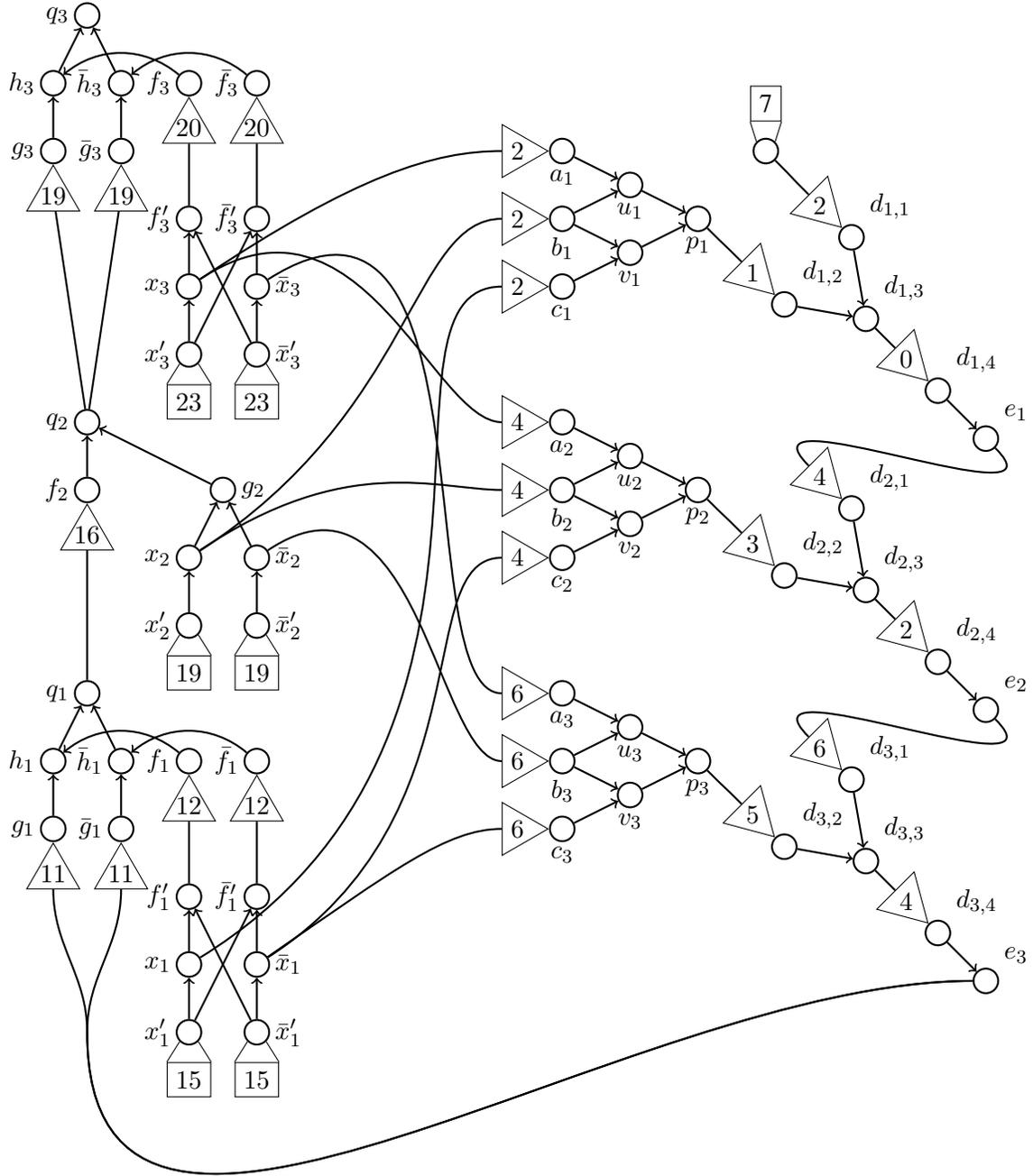


Figure 6: Example of QBF-to-DAG reduction for $\forall x_3 \exists x_2 \forall x_1 (x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3)$.

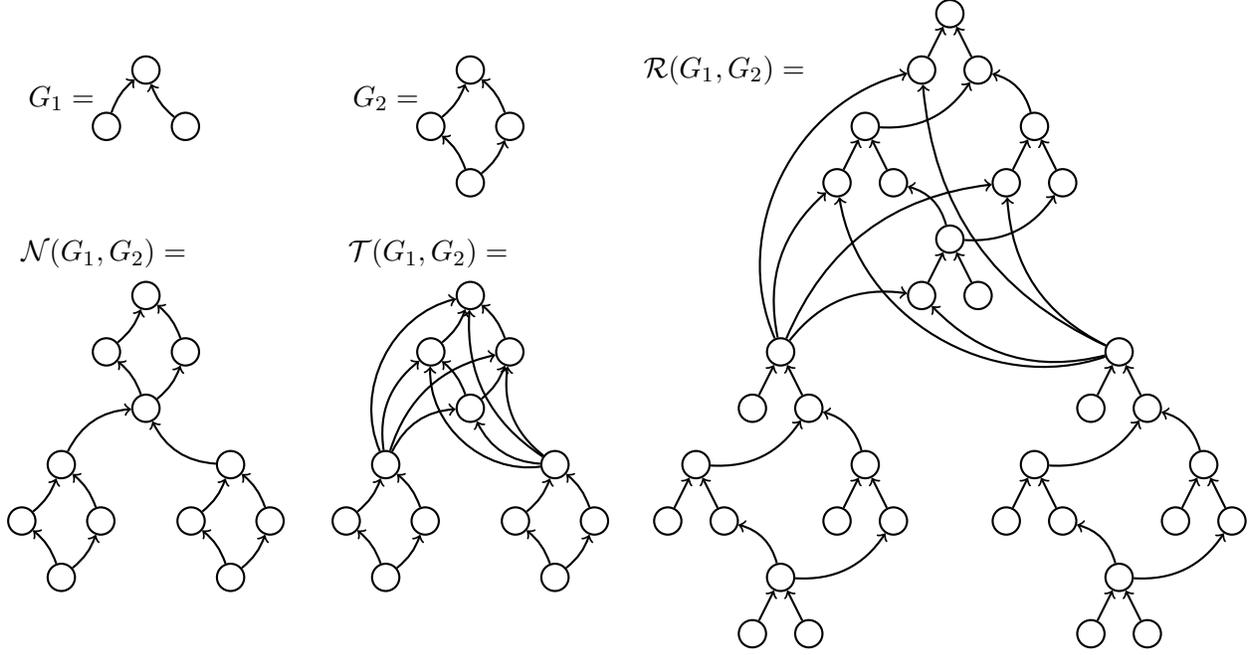


Figure 7: Examples of graph products as applied to a pyramid of height 1 (denoted G_1) and a rhombus (denoted G_2).

standard pebbling at the very end of the section. We will refer to G_1 as the *outer graph* and G_2 as the *inner graph* in the graph products $\mathcal{R}(G_1, G_2)$ and $\mathcal{S}(G_1, G_2)$.

Intuitively, when taking the graph product of G_1 and G_2 the idea is to replace every vertex v of the outer graph G_1 with a (possibly slightly modified) copy of the inner graph G_2 . We will refer to this copy as the v -block in the product graph. The edges inside blocks are specified by the inner graph. For edges $(u, v) \in E(G_1)$ in the outer graph, we will need to connect the sink of the u -block to vertices in the v -block in some way, and this is the crux of the construction.

A first naive approach would be to add an edge from the sink of the u -block to every source vertex of the v -block (as shown in the graph product $\mathcal{N}(G_1, G_2)$ in Figure 7). Sadly, this simple idea fails for both standard and reversible pebbling. It is not hard to find examples showing that the pebbling price of $\mathcal{N}(G_1, G_2)$ is not a function of the pebbling prices of G_1 and G_2 .

A slightly more refined idea is to add edges from the sink of the u -block to all vertices in the v -block (as in the graph $\mathcal{T}(G_1, G_2)$ in Figure 7). While we can observe right away that this idea is a non-starter, since it will blow up the fan-in of the product DAG (and with no bounds on fan-in the gap amplification would be trivial), it turns out that the analysis yields interesting insights for the graph product that we will actually use. We will therefore employ this toy construction to showcase some of the ideas and technical challenges that arise in the actual proof of Theorem 3.5.

Recall that we want to prove that $RPeb(\mathcal{T}(G_1, G_2)) = s_1 + s_2 - 1$, where $s_i = RPeb(G_i)$ for $i = 1, 2$. To reversibly pebble the product graph $\mathcal{T}(G_1, G_2)$ in at most this amount of space we simulate a minimal space pebbling of G_1 , where pebble placement or removal involving a vertex v of G_1 invokes a complete pebbling (or unpebbling) of the copy of G_2 corresponding to the v -block. This simulation uses at most s_2 pebbles in the relevant v -block and at most $s_1 - 1$ pebbles on sinks of other blocks, i.e., no more than $s_1 + s_2 - 1$ pebbles in total.

Proving the lower bound $RPeb(\mathcal{T}(G_1, G_2)) \geq s_1 + s_2 - 1$ is the difficult part. Here the approach is to assume that we are given a complete pebbling \mathcal{P}^T of $\mathcal{T}(G_1, G_2)$ and extract from it a pebbling strategy \mathcal{P} for G_1 with the hope that an expensive configuration in \mathcal{P} will also help us to pinpoint an expensive configuration

in \mathcal{P}^T .

The most straightforward way to obtain a pebbling strategy \mathcal{P} for G_1 from \mathcal{P}^T would be to make a vertex v in G_1 contain a pebble or not depending only on the local pebble configuration of the v -block in $\mathcal{T}(G_1, G_2)$. A natural idea is that v should get a pebble if the v -block has a pebble on its sink and that this pebble should be removed from v when the corresponding block has been emptied of pebbles. If we apply this reduction to a pebbling \mathcal{P}^T of $\mathcal{T}(G_1, G_2)$ we obtain a valid pebbling of G_1 . The problem, however, is that \mathcal{P}^T might locally be doing a visiting pebbling (as defined in Section 2.3) of the copy of G_2 corresponding to the v -block as a way of moving pebbles on or off other blocks. The consequence of this would be that a configuration of maximal space s_1 in \mathcal{P} may result from a configuration in \mathcal{P}^T that uses space only $s_1 + s_2 - 2$, which is off by one compared to what we need and hence destroys the gap in pebbling price that we are trying to create.

If the visiting price of G_2 is the same as its persistent price, then this problem does not arise, but since this does not hold for graphs in general we need to argue more carefully. It is true that a visiting pebbling of a copy of G_2 might save one pebble as compared to a persistent pebbling, but whenever the sink contains a pebble in a visiting but not persistent pebbling we know that there must also be some other vertex in G_2 that has a pebble (or else the pebbling would be persistent by definition). We need to count such pebbles also in our analysis.

To this end, we make a distinction between blocks that have paid the persistent price and the blocks that have paid the visiting price but not the persistent price. We say that the copy of G_2 corresponding to some v -block is *visiting-locked*, or just *v-locked* for brevity, at some point in time if the current pebble configuration on its vertices requires reversible pebbling space $s_2 - 1$ to be reached, and that the v -block is *persistent-locked* (or *p-locked* for short) if the configuration has reversible pebbling price s_2 .

We can now define a more refined way of projecting \mathcal{P}^T -configurations to \mathcal{P} -configurations as follows. If a v -block has paid the persistent price, we put a pebble on the corresponding vertex v in G_1 . If a block has paid just the visiting price but not the persistent price, then we might still put a pebble on v in G_1 , but we only do so if an additional (and slightly delicate) technical condition³ holds for the pebbling configurations in the blocks corresponding to predecessors of v . This technical condition is designed so that with some additional work⁴ we are still able to extract a legal pebbling strategy for G_1 by applying this projection. Furthermore, it will be the case that every pebbling move on a vertex in the outer graph G_1 is the result of the copy of G_2 corresponding to some v -block paying the persistent or visiting price.

The reversible pebbling \mathcal{P} thus extracted will be a persistent pebbling of G_1 by construction, so it must contain a configuration with s_1 pebbles. If this configuration was reached because a block paid the persistent price, then that block contains s_2 pebbles at a time when at least $s_1 - 1$ other blocks have at least 1 pebble each, which is the lower bound that we are after. If the pebble configuration on G_1 in \mathcal{P} was reached because a block paid the visiting price, however, then we are potentially still one pebble short. This is where the additional technical condition mentioned above comes into play. This condition on the predecessor blocks implies that we can find at least one other block that also paid just the visiting price and therefore must contain two pebbles. Summing up, we obtain one block that has at least $s_2 - 1$ pebbles, another block that has at least

³We do not want to get into too detailed a technical argument here, but just for the record pebble configurations on $\mathcal{T}(G_1, G_2)$ can be projected to configurations on G_1 in two stages as follows:

1. Let $\mathbb{P} \subseteq V(G_1)$ consist of all vertices u such that the configuration on the u -block in $\mathcal{T}(G_1, G_2)$ is persistent-locked.
2. Let $\mathbb{P}' \subseteq V(G_1) \setminus \mathbb{P}$ consist of all vertices v such that (a) v is not already surrounded by \mathbb{P} , and (b) the configuration on the v -block in $\mathcal{T}(G_1, G_2)$ is visiting-locked.

With this notation, the projected pebble configuration on G_1 is defined to be $\mathbb{P} \cup \mathbb{P}'$.

⁴One added technical complication that we have to take care of here is that when we apply our projection to a pebbling \mathcal{P}^T of $\mathcal{T}(G_1, G_2)$ to obtain a sequence of pebble configurations on G_1 , this sequence need not be a valid pebbling of G_1 . However, when the projected pebble configuration on G_1 changes after a pebbling move we can insert a legal pebbling sequence between the two projected configurations that passes through all vertices of G_1 corresponding to v-locked blocks, where pebbles are added in topological order and removed in inverse topological order, and this local pebbling does not affect the overall argument.

2 pebbles, and at least $s_1 - 2$ additional blocks that contain at least 1 pebble each, and so the lower bound holds in this case as well. (Incidentally, this second case is the one where our first, naive, graph product $\mathcal{N}(G_1, G_2)$ fails.)

We already observed, however, that the construction $\mathcal{T}(G_1, G_2)$ does not get us very far because it blows up the indegree of the resulting product graph. Therefore, in the actual proof of Theorem 3.5 we have to consider a different construction. Briefly, the idea is to start with the graph $\mathcal{T}(G_1, G_2)$ but to bring the indegree down by splitting each vertex w in every block into three vertices $w_{\text{ext}}, w_{\text{int}}, w_{\text{out}}$. All edges to w from other blocks are routed to w_{ext} , all edges from within the block are routed to w_{int} , and finally we add edges from w_{ext} and w_{int} to w_{out} . This is the graph product $\mathcal{R}(G_1, G_2)$ that we use to amplify differences in reversible pebbling price, and that is also illustrated in Figure 7. Now we have to prove that the ideas just outlined work for this new construction where each vertex has been replaced by a small “cloud” of three vertices. The proof of this is much more technically challenging than for the toy case discussed above, and there is no room to go into details here.

At this point we want to switch gears a bit and briefly discuss an application in proof complexity of the PSPACE-hardness result for reversible pebbling. Perhaps the most well-studied proof system for proving the unsatisfiability of, or refuting, CNF formulas is *resolution* (we do not give any formal definition here, referring instead to, for instance, [Seg07] for the necessary details). Every resolution proof can be represented as a DAG, and the *depth* of this proof is the length of a longest path in this DAG. The *resolution depth* of refuting an unsatisfiable CNF formula is the smallest depth of any resolution proof for the formula. It was shown in [Cha13a] that computing the reversible pebbling price of a graph of fan-in ℓ reduces to computing the resolution depth of a $(\ell + 1)$ -CNF formula, and from this we can obtain the following corollary.

Corollary 3.6. *For any fixed positive integer K , it is PSPACE-complete to compute the resolution depth of refuting 3-CNF formulas up to an additive error K .*

Proof. Assuming that we can efficiently compute the resolution depth within an additive error at most K , we show how to efficiently compute the reversible pebbling price of any graph G within an additive error $K + 1$, contradicting Theorem 3.4.

Letting z denote the unique sink of G , we consider a new graph G' which is G augmented with a new successor z' of z (i.e., $G' = (V \cup \{z'\}, E \cup \{(z, z')\})$ in formal notation). The reversible pebbling prices of G and G' differ by at most one. For any graph G , [Cha13a] exhibits an efficiently constructible unsatisfiable CNF formula F_G that requires resolution depth equal to the reversible pebbling price of G' . The width of the formula is equal to the fan-in of G plus one, so the result holds for 3-CNFs.

Hence, if we could estimate the resolution depth of refuting F_G , i.e., the reversible pebbling price of G' , within error K , this would yield an estimate of the reversible pebbling price of G to within error $K + 1$. \square

We wrap up this section by switching back to pebbling and describing the product construction $\mathcal{S}(G_1, G_2)$ used to amplify standard pebbling price. In this construction we also replace every vertex of G_1 with a copy of G_2 , but this time we append what we refer to as a *centipede* graph to the sink of every copy. A centipede is a path where each vertex but the source has an extra, unique predecessor. To connect the blocks, for every edge $(u, v) \in E(G_1)$ we add edges from the sink of the u -centipede to every source of the v -centipede. See Figure 8 for an illustration.

Setting $s_i = \text{Peb}(G_i)$ for $i = 1, 2$, we can pebble the graph product $\mathcal{S}(G_1, G_2)$ in space $s_1 + s_2 - 1$ by simulating an optimal pebbling of G_1 : placing a pebble on a vertex v of G_1 is simulated by optimally pebbling the sink of the corresponding v -block, and removing a pebble is simulated by removing the pebble on the sink.

This pebbling strategy is in fact optimal, and we can show this by projecting any standard pebbling $\mathcal{P}^{\mathcal{S}}$ of $\mathcal{S}(G_1, G_2)$ to a strategy \mathcal{P} for G_1 . Each time any block in $\mathcal{S}(G_1, G_2)$ contains s_2 pebbles, we pebble all vertices in G_1 whose predecessors have pebbles and whose corresponding block in \mathcal{S} has a pebble. When

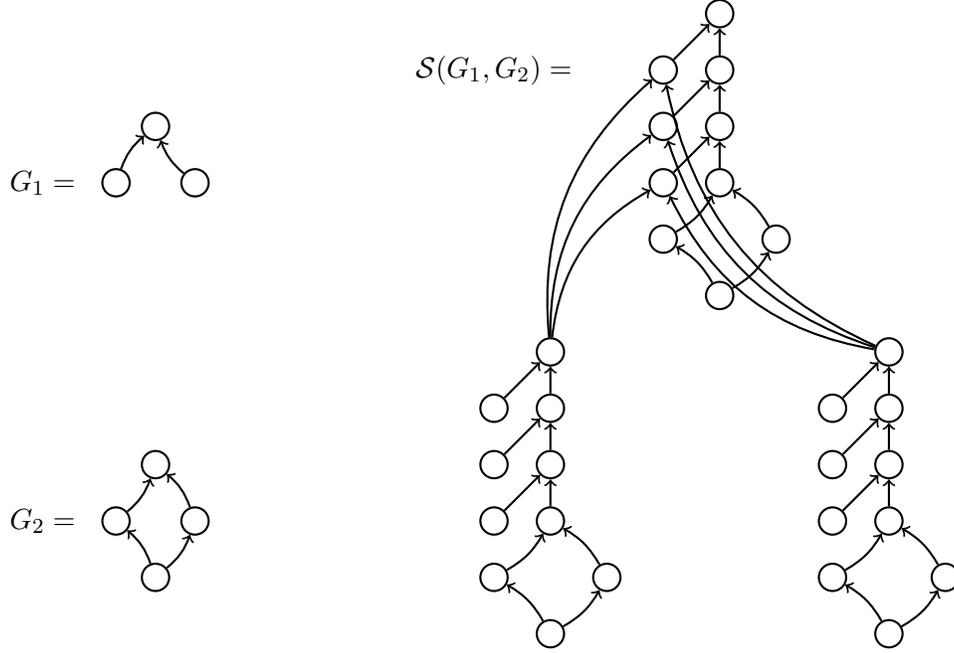


Figure 8: Illustration of standard pebbling graph product $\mathcal{S}(G_1, G_2)$.

a block in $\mathcal{S}(G_1, G_2)$ becomes empty, we remove the pebble from the corresponding vertex in G_1 . This projection has the property that when the sink of a block is pebbled, the corresponding vertex in G_1 is also pebbled. Arguing similarly to in the reversible case, we show that a strategy \mathcal{P}^S for \mathcal{S} using s pebbles yields a strategy for G_1 using $s - s_2 + 1$ pebbles. Therefore, \mathcal{P}^S must use space at least $s_1 + s_2 - 1$, and hence the graph product $\mathcal{S}(G_1, G_2)$ has the property claimed in Theorem 3.5.

4 Separation between Standard and Reversible Pebbling

In this section we discuss how the reversible pebbling price compares with the standard one. A reversible pebbling is also a legal standard pebbling, but the opposite is not always true. However it is possible to construct a reversible pebbling from a standard pebbling of time τ that costs at most $\log \tau$ times the price of the standard pebbling.

Theorem 4.1 ([Krá04]). *If graph G has a standard pebbling of time τ and space p , then G has reversible pebbling price at most $p \lceil \log \tau \rceil$.*

Proof sketch. Let $\mathcal{P} = (\mathbb{P}_0, \dots, \mathbb{P}_\tau)$ be a standard pebbling of G in space p . We show a Pebbler strategy for the Dymond–Tompa game on G that allows Pebbler to win in at most $p \lceil \log \tau \rceil$ rounds. Since $DT(G) = RPeb(G)$ this is sufficient. Pebbler keeps as an invariant an interval $[a, b]$ such that the challenged pebble is in \mathbb{P}_b and all vertices in \mathbb{P}_a are pebbled but not challenged. Initially the interval is $[0, \tau]$, and the strategy proceeds by bisection. At each bisection step Pebbler starts pebbling the vertices in the configuration \mathbb{P}_m , with $m = (a + b)/2$, in any order. If Challenger jumps to a vertex v , then let t be the smallest number such that $a \leq t$ and $v \in \mathbb{P}_t$. Pebbler now plays in $[a, t]$. The interval halves because $t \leq m$. If Challenger stays in all moves, then Pebbler plays in $[m, b]$ and the interval also halves. When the interval becomes unit, the Pebbler invariant implies that the move from \mathbb{P}_a to \mathbb{P}_b is precisely a placement on the challenged vertex. Therefore, the predecessors of the challenged vertex are pebbled and the game ends. The game considers at most $\lceil \log \tau \rceil$ configurations of \mathcal{P} and spends at most p rounds on each. \square

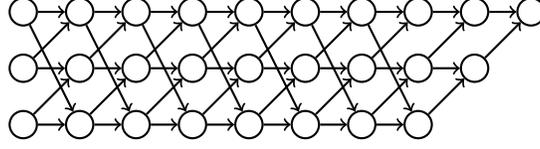


Figure 9: Road graph of length 9 and width 3.

We already know that the difference between the standard and reversible pebbling price is unbounded. For example the standard pebbling price for a path of length n is 2, while its reversible pebbling price is $\Theta(\log n)$. It follows that if a DAG G has depth d and a standard pebbling of time τ and space p , then

$$\max\{p, \log d\} \leq RPeb(G) \leq p \log \tau$$

We rule out the possibility of a simulation with only an additive loss. Indeed we show a separation which is multiplicative in terms of the logarithm of the size of the graph.

Theorem 4.2. *For any function $s(n) = O(n^{1/2-\epsilon})$ where $\epsilon > 0$ is constant there are DAGs $\{G_n\}_{n=1}^\infty$ of size $\Theta(n)$ with a single sink and fan-in 2 such that $Peb(G) = O(s(n))$ and $RPeb(G) = \Omega(s(n) \log n)$ (where the hidden constant depends linearly on ϵ).*

The graphs that we use to witness the separation are the chains or “wide paths”. A *chain* of width w and length ℓ is a graph with $\ell + 1$ layers, each having w vertices, where the i -th vertex of a layer has two incoming edges from the i and $i + 1$ -th vertices of the previous layer (modulo w). The layers are indexed from 0 (the layer of the sources) to ℓ (the layer of the sinks).

Since we want single sink graphs, we define a *road* of width w and length ℓ to be a chain of width w and length $\ell - w + 1$ plus a pyramid of height $w - 1$, where we identify the sinks of the chain with the sources of the pyramid. The layers are indexed in the same way as in the chain.⁵

By pebbling each layer in order, we get a standard pebbling of a road of width w which uses $w + 2$ pebbles. The reversible pebbling of a road depends on its length: a road of width w and length ℓ has a reversible pebbling price $O(w \log \ell)$. The idea is to simulate in parallel w copies of the reversible pebbling of the path of length ℓ , which has price $O(\log \ell)$. We prove Theorem 4.2 by choosing for each n a road of width $w = s(n)$ and length $\ell = n/w$, and showing that this pebbling is essentially optimal.

A *blocking set* for a vertex set $T \subseteq V(G)$ is a subset of vertices $B \subseteq V(G)$ such that every path from any source to any vertex in T must contain a vertex in B . We also say that B *blocks* T . A blocking set for all sinks of a directed acyclic graph G is also called a blocking set of G . We say that B is a minimal blocking set if no subset of B is a blocking set.

A chain of width w has blocking sets with w vertices, all in the same layer. It turns out that if a minimal blocking set has vertices in multiple layers then it must be larger than that. We say that a blocking set *spreads* over d layers when a and b are the lowest and the highest layers that the blocking set intersects, respectively, and $d = b - a + 1$.

Lemma 4.3. *A minimal blocking set of a chain that spreads over d layers has size at least $d + w - 1$.*

Proof. Consider such a minimal blocking set B . Sort the layers of the chain from 0 (sources) to ℓ (sinks) and let a and b be the first and last layers with vertices in B , so that $d = b - a + 1$. If $a = b$ then B must contain w vertices and the Lemma holds. For the rest of the proof we assume $a < b$.

⁵Equivalently, a road of length ℓ and width $w \leq \ell$ is an induced subgraph of a chain of length ℓ and width w . Fix one arbitrarily sink s in the chain: the subgraph induced by vertices in $anc(s)$ is indeed a road of length ℓ and width w .

Define $f(i)$ to be the number of vertices at layer i that can reach a sink without passing through a vertex in B (and are not in B themselves). By definition and minimality we get that $f(i) = 0$ for all layers $i \leq a$; $0 < f(i) < w$ for all layers $a < i \leq b$ and $f(i) = w$ for $i > b$.

Now we compute the intersection between B and each layer. All vertices at layer b can reach the sink unless they are in B , therefore B must have $w - f(b)$ elements at the last layer. We claim that that B contains at least $f(i + 1) - f(i) + 1$ vertices from layer i , for $a \leq i < b$.

Indeed, consider the set of the $f(i + 1)$ vertices at layer $i + 1$ that can reach a sink, and define N_i to be the set of their predecessors in layer i . Since $0 < f(i + 1) < w$, there are at least $f(i + 1) + 1$ vertices in N_i .

A vertex in the i -th layer can reach a sink if and only if it is in N_i and not blocked by B . Since we assumed that exactly $f(i)$ of them can reach a sink, it must be that $|N_i| - f(i)$ vertices are blocked by B right on layer i , i.e., they are contained in B . Thus the intersection between B and layer i is $|B_i| \geq |N_i| - f(i) = f(i + 1) - f(i) + 1$.

Using these facts we get that

$$|B| = \sum_{i=a}^b |B_i| \geq w - f(b) + \sum_{i=a}^{b-1} f(i + 1) - f(i) + 1 = w + f(a) + (d - 1) = d + w - 1 .$$

□

We need to generalize Lemma 4.3 to a road in order to handle blocking sets within the pyramid part.

Lemma 4.4. *A minimal blocking set of a road that spreads over d layers has size at least $d + q - 1$, where q is the width of the topmost layer.*

Proof. If the blocking set is located on a single layer the lemma follows immediately. Otherwise the proof is very similar to the one of Lemma 4.3, except that the intersection between B and its last layer b has size at least $q - f(b)$. □

Now we prove the lower bound in Theorem 4.2

Lemma 4.5. *The reversible pebbling price of a road of width w and length ℓ is at least $w \log(\ell/w)/2$.*

Proof. We give a Challenger strategy by induction over ℓ that lasts for $w \log(\ell/w)/2$ moves. Furthermore, the strategy stays as long as the sink is connected to the sources. The base case is a road of length $\ell \leq w - 1$, i.e., a pyramid of height ℓ , in which case the lemma holds vacuously.

We say that a directed path in the graph is *semiopen* when there are no pebbles on it except for its last vertex. A semiopen path from a vertex to itself is a single vertex with a pebble on it.

During the game Challenger *focuses* on a subgraph of the road, and keeps the following invariant at every round: there is a semiopen path from the sink of this subgraph to the currently challenged pebble. This concretely means that if Pebbler places a pebble inside the subgraph then Challenger plays according to its strategy for that subgraph. Instead if the new pebble blocks the semiopen path between the currently challenged pebble and the sink of the subgraph, Challenger jumps to the new pebble—essentially making the path shorter.

If at some round Challenger focuses on a subgraph, in later rounds Challenger will never challenge a vertex which is neither in the subgraph nor in the semiopen path between its sink and the current challenge.

Let us now give the strategy for playing inside the subgraph. As long as the sink of the subgraph is connected to the sources, Challenger stays. If the sink is disconnected from the sources by a blocking set, Challenger decides to jump or to stay depending on the position of the blocking set. Before describing how Challenger decides, we describe how the strategy continues in both cases.

We consider a minimal blocking set B and note that the last pebbled vertex u is in any blocking set. Indeed, there is a path from the sources to the sink that only has pebbles at u and at the sink, otherwise the sink would have already been blocked.

We first consider the strategy after Challenger decides to jump. Let v be the vertex in the semiopen path from the sources to u at the layer immediately before all the vertices in B . From now on Challenger focuses on the subgraph induced by the ancestors of v , which is a road. This road is not blocked, has no vertex in the blocking set B , and there is a semiopen path from v to the challenged pebble u .

If Challenger decides to stay, it focuses on the road with sources at the layer immediately after all vertices in B . Again, this road is not blocked, it is disjoint from the blocking set B , and there is a semiopen path from its sink to the currently challenged pebble.

It remains to describe how Challenger decides to jump or to stay. If the last layer of the blocking set has width $q < w$, then by Lemma 4.4 it has size at least $q + d - 1$. In this case Challenger jumps and focuses on a road of length $\ell - (q + d - 2)$. Let $DT(w, \ell)$ be the Dymond–Tompa price of a road of width w and length ℓ . Overall the Challenger strategy lasts for

$$|B| + DT(w, \ell - (q + d - 2)) \geq q + d - 1 + w \log((\ell - (q + d - 2))/w)/2 \geq w \log(\ell/w)/2 \quad (4.1)$$

steps. Otherwise the blocking set B has size at least $w + d - 1 \geq w$ for some $d \geq 1$ and by Lemma 4.3 it spreads over at most d layers, where a is the lowest and b is the highest, with $d = b - a + 1$ and $m = (a + b)/2$. If $m \leq \ell/2$, Challenger stays and focuses on the road of length $\ell - b - 1$ obtained considering only the vertices at the layers from $b + 1$ to ℓ . Overall the Challenger strategy lasts for

$$\begin{aligned} |B| + DT(w, b - 1) &\geq w + d - 1 + w \log((b - 1)/w)/2 \geq w + d - 1 + w \log((\ell - d - 1)/2w)/2 \\ &= w/2(\log((\ell - d - 1)/2w) + 2 + 2(d - 1)/w) \geq w/2(\log(\ell/2w) + 1) = w \log(\ell/w)/2 \end{aligned} \quad (4.2)$$

steps. If $m > \ell/2$, Challenger jumps and focuses on a road of length $a - 1$ obtained considering one vertex at layer $a - 1$ which is connected to the sources, and taking all vertices which have a path toward such vertex. Overall the Challenger strategy lasts for

$$\begin{aligned} |B| + DT(w, a - 1) &\geq w + d - 1 + w \log((a - 1)/w)/2 \\ &\geq w + d - 1 + w \log((\ell - d - 1)/2w)/2 \geq w \log(\ell/w)/2 \end{aligned} \quad (4.3)$$

steps. □

Note that Theorem 4.2 follows if the road width is $w = s(n)$ and the length is $\ell = n/w$ because $w \log(\ell/w) = w \log(n/w^2) = \Theta(w \log n)$ if $w = O(n^{1/2-\epsilon})$.

5 Tight Bounds for Trees and Pyramids

In this section we show matching upper and lower bounds for the persistent pebbling price of complete binary trees and pyramids. Asymptotically tight results for trees were given in [Krá04]. The pyramid graph of height h has a vertex for every pair (i, j) with $0 \leq i \leq j \leq h$. The sources are the vertices $(0, j)$ for $j \geq 0$, the sink is the vertex (h, h) , and every vertex (i, j) for $i < h$ has one outgoing edge going left to vertex $(i + 1, j)$ if $j > i$ and one outgoing edge going right to vertex $(i + 1, j + 1)$ if $j < h$. A pyramid can be obtained from a complete binary tree of height h by identifying together some of its vertices, in such a way that the left and right predecessors of two vertices that get identified, get pairwise identified as well. For this reason an upper bound for binary trees also holds for the pyramids, and a lower bound for pyramids also holds for binary trees.

In the following, let $p = h + \Delta$ be the persistent price of a pyramid. A pyramid of height h has standard pebbling price $h + 2$ [Coo74], which means that in the standard pebbling only two extra pebbles are needed

compared to the height of the pyramid. In a similar fashion Δ can be interpreted as the extra space needed by persistent pebbling, with respect to pyramid height. We want to estimate the height of the pyramid that has persistent pebbling with at most Δ extra pebbles.

Definition 5.1. Consider $\Delta \in \mathbb{N}^+$ and let $g(\Delta)$ be the function defined by the following recursion,

$$g(\Delta) = \begin{cases} 0 & \text{if } \Delta = 1 \\ 2^{g(\Delta-1)+\Delta-2} + g(\Delta-1) & \text{otherwise.} \end{cases}$$

We define its inverse as

$$g^{-1}(h) = \min\{\Delta \mid g(\Delta) \geq h\}.$$

We show that $g(\Delta)$ is the maximum height of a pyramid that can be persistently pebbled using Δ pebbles on top of h . Observe that $g(\Delta) = \Omega(\underbrace{2^{2^{\dots^2}}}_{\Delta})$, therefore $g^{-1}(h) = O(\log^* h)$.

Proposition 5.2. We can compute $g^{-1}(h)$ in time $(\log h)^{O(1)}$ and space $O(\log h)$.

Proof. If $h = 0$ then $g^{-1}(0)$ is 1 by definition. If $h > 0$ then we need to find the smallest $\Delta > 1$ such that $h \geq g(\Delta)$. We start from $\Delta := 2$ and go upward. At each step we keep in memory the value $g(\Delta - 1)$, which is smaller than h , and we test the condition

$$h < g(\Delta) \quad \text{equivalent to} \quad \lceil \log(h - g(\Delta - 1)) \rceil < g(\Delta - 1) + \Delta - 2 .$$

The latter test can be achieved in time in $\log h$ by checking the length of the bit representation of $h - g(\Delta - 1)$. If the test fails we output Δ otherwise we store the value $g(\Delta)$, we fix $\Delta := \Delta + 1$ and we continue. We can do the whole computation by storing at most 4 numbers less than h , each step is polynomial in the length of the binary representation of the numbers involved, and we need to do at most $O(\log^* h)$ steps. \square

Theorem 5.3. The persistent pebbling price of a binary tree of height h and a pyramid of height h is $h + g^{-1}(h)$.

To prove the theorem we need the exact value of the persistent pebbling price of paths.

Lemma 5.4 (Path graphs [LV96]). The persistent pebbling price of a path of length h (i.e., with $h + 1$ vertices) is $\lfloor \log(h) \rfloor + 2$.

Lemma 5.5 (Upper bound for binary trees). The persistent pebbling price of a complete binary tree of height $h \leq g(\Delta)$ is at most $h + \Delta$.

Proof. We are going to prove the lemma by induction over h . For the base case we observe that a binary tree of height 0 can be pebbled with 1 pebbles. For the general case we assume the statement of the lemma for height $i < h$, and we show that the surrounding pebbling price of the complete binary tree of height h is at most $h + \Delta - 1$. Proposition 2.1 immediately implies the lemma for height h .

Let us denote the root by v_h and the right child of v_i by v_{i-1} . The strategy is as follows. First we persistently pebble the left child of v_i for i from h down to $k := g(\Delta - 1) + 1$, in this order. By the induction hypothesis $(i - 1) + \Delta$ pebbles are enough to persistently pebble the left child of v_i , and there are $h - i$ pebbles left on the rest of the graph from previous steps. So we are within the bound $h - 1 + \Delta$, and after the last step we have $h - (k - 1)$ pebbles on the tree.

Then we persistently pebble v_{k-1} , the right child of v_k . Since $k - 1 = g(\Delta - 1)$, by induction hypothesis $(k - 1) + (\Delta - 1)$ pebbles are enough and we are within the bound. Let $j := h - k + 1$. So far we used $j + 1 = h - k + 2$ pebbles.

Finally we surround the sink of path $(v_k, v_{k+1}, \dots, v_h)$, which has j vertices, using $\lfloor \log(j-1) \rfloor + 1$ pebbles. Observe that by construction $j-1 = h-k \leq g(\Delta) - g(\Delta-1) - 1 < 2^{g(\Delta-1)+\Delta-2}$, hence we have the bound $\lfloor \log(j-1) \rfloor < g(\Delta-1) + \Delta - 2$. Counting the total number of pebbles in the graph gives $(h-k+2) + \lfloor \log(j-1) \rfloor + 1 \leq (h-g(\Delta-1)+1) + (g(\Delta-1) + \Delta - 3) + 1 = h + \Delta - 1$ pebbles. \square

We prove the lower bound for a slight generalization of pyramids in order to obtain a lower bound on the visiting price in addition to the persistent price.

Definition 5.6. An (h, ℓ) -teabag is the union of a pyramid of height h and a path of length ℓ , where we identify the sink of the pyramid and the source of the path.

Observe that an $(h, 0)$ -teabag is a pyramid.

For the lower bound we will also need the following basic fact about pyramids. Recall that a blocking set is a subset of vertices $B \subseteq V(G)$ such that every path from any source to the sink must contain a vertex in B . Also recall that a directed path in the graph is semiopen when there are no pebbles on it except for its last vertex.

Proposition 5.7 ([Coo74]). Consider a blocking set B on a pyramid of height h ; consider a vertex v at level k such that there is a path between v and the sink whose intersection with B is at most $\{v\}$. Let U be the set of vertices in the sub-pyramid rooted at v . Then $|B \setminus U| \geq h - k$.

Proof. Pick an arbitrary path which starts at vertex v , reaches the pyramid sink and does not intersect B anywhere other than in v . Denote such path as $(v_k, v_{k+1}, \dots, v_h)$ where v_k is another name for v and v_h is the sink. Each v_i is at height i in the pyramid. On pyramids there is a natural notion for edges to go either left or right. For each $i > k$ we define the path P_i as follows: if edge (v_{i-1}, v_i) goes right then P_i is the unique path that starts at a source vertex, always goes left, and ends at v_i ; if edge (v_i, v_{i-1}) goes left then P_i is the unique path that starts at a source vertex, always goes right, and ends at v_i . It is easy to verify that none of P_h, \dots, P_{k+1} intersects any of the vertices in U , and that these paths are all pairwise vertex disjoint. Since B is a blocking set it must contain one vertex for each P_i and the proposition follows. \square

Lemma 5.8. The persistent pebbling price of the (h, ℓ) -teabag is at least $h + \Delta + 1$ if either of the following holds:

- $h > g(\Delta)$,
- $h > g(\Delta - 1)$ and $\ell > g(\Delta) - h$.

Proof. We define a Challenger strategy for the Dymond–Tompa game by induction over h and ℓ in this order. Furthermore this strategy stays on the sink until Pebbler blocks the graph. For the base case $h = 0$, the statement is trivial.

Assume that the last Pebbler move blocks the graph, meaning that the currently pebbled vertices form a blocking set, and fix B to be a minimal one. The vertex v pebbled at that round must be in B . We have two cases depending on k the layer of vertex v .

- Case $k > g(\Delta - 1)$: Challenger jumps to v . The pebble on v blocks the sources from the sink, so there must be a semiopen path between v and a source. Let U be the set of pebbles contained in the vertices of P_v , the subgraph of predecessors of v (notice that $v \in U$). Consider a new game on P_v , in which the first actions of Pebbler are to pebble $U \setminus \{v\}$ in any order, while Challenger stays on v . The set $U \setminus \{v\}$ does not block the subgraph. If $k \geq h$ then the new sub-game ends in at least $h + \Delta$ steps, and the total number of rounds is at least $1 + h + \Delta$. Otherwise v is inside the pyramid, and the new

sub-game ends in $k + \Delta$ steps. We use Proposition 5.7 to claim that $|B \setminus U| \geq h - k$. So in total the rounds in the game are at least

$$\underbrace{1}_{\text{challenge to sink}} + \underbrace{|B \setminus U|}_{\text{outside } P_v} + \underbrace{k + \Delta}_{\text{subgame on } P_v} \geq h + \Delta + 1 .$$

- Case $k \leq g(\Delta - 1)$: there is a path of length $h - k + \ell$ from v to the sink having only a pebble on each end. So any optimal Pebbler strategy must contain a strategy for playing on the semiopen path of length $h - k + \ell - 1$ from one unpebbled successor of v to the sink. Fix $q = \lfloor \log(h - k + \ell - 1) \rfloor$, the sub-game on the path of length $h - k + \ell - 1$ lasts at least $q + 2$ rounds (see Lemma 5.4). The initial challenge on the sink of the graph is part of this sub-game, but all moves on B are not, so the total number of rounds is

$$\underbrace{|B|}_{\text{blocking set}} + \underbrace{q + 2}_{\text{subgame on path}} \geq \underbrace{h - k + 1}_{\text{blocking set}} + \underbrace{g(\Delta - 1) + \Delta}_{\text{subgame on path}} \geq h + \Delta + 1 .$$

The bound on $|B|$ holds because Proposition 5.7 on vertex v implies $|B \setminus \{v\}| \geq h - k$. The bound on $q + 2$ holds because by hypothesis $h - k + \ell - 1 \geq g(\Delta) - g(\Delta - 1)$, which implies that $q \geq \lfloor \log(g(\Delta) - g(\Delta - 1)) \rfloor \geq g(\Delta - 1) + \Delta - 2$. \square

Corollary 5.9 (Lower bound for pyramids). *The persistent pebbling price of a pyramid of height $h > g(\Delta - 1)$ is at least $h + \Delta$. The visiting price of a pyramid of height $h = g(\Delta)$ is at least $h + \Delta$.*

Proof. Lemma 5.8 claims the first statement. The second one holds because if the pyramid of height $g(\Delta)$ had visiting price $h + \Delta - 1$, then the $(g(\Delta), 1)$ -teabag would have persistent pebbling price $h + \Delta$, which contradicts Lemma 5.8. \square

6 Technical Constructions

In order to discuss lower bounds on pebbling price we need to identify expensive pebbling configurations, namely the configurations that are expensive to reach from the empty configuration. We will often use the reverse direction, i.e., that the empty configuration cannot be reached without passing through an expensive configuration.

Definition 6.1. A configuration \mathbb{P} is *v-locked* if $RPeb_G(\mathbb{P}) = RPeb^V(G)$. A configuration \mathbb{P} is *p-locked* if $RPeb_G(\mathbb{P}) = RPeb(G)$.

6.1 Christmas Tree Construction

This section builds on the pyramid graphs to provide a graph T_r with equal visiting and persistent prices r for every $r \in \mathbb{N}^+$. As a preliminary step we show a graph G_p with persistent price p for every $p \in \mathbb{N}^+$.

Lemma 6.2 (Modified Pyramids). *There is a family of graphs $\{G_p\}_{p \in \mathbb{N}^+}$ such that*

1. $RPeb(G_p) = p$;
2. G_p has in-degree at most two and a unique sink; and
3. G_p is polynomial-time computable given p , and G_p has at most p^2 nodes.

Proof. The value of $g^{-1}(h)$, which is the extra pebbling price of pyramids with respect to the height, increases only when $h = g(\Delta) + 1$. Therefore the persistent pebbling price of a pyramid increases by 1 unless $h = g(\Delta) + 1$, in which case it increases by 2. If $p = h + g^{-1}(h)$ for some $h \in \mathbb{N}$ we let G_p be the pyramid graph of height h . In this way G_p is defined for every $p > 0$, unless $p = h + g^{-1}(h) + 1$ for some $h = g(\Delta)$. In this case we let G_p be the $(h, 1)$ -teabag which, by Lemmas 5.5 and 5.8, has persistent pebbling price $p = h + g^{-1}(h) + 1$. \square

We want a polynomial-time computable family of graphs $\{T_r\}_{r \in \mathbb{N}^+}$ with matching visiting price and persistent price, i.e., $RPeb^V(T_r) = RPeb(T_r) = r$. The idea is to stack up r appropriately chosen graphs, so that any visiting or persistent pebbling strategy has to spend one pebble per graph.

We will use r graphs from the family $\{G_p\}_{p \in \mathbb{N}^+}$ where each G_p has persistent pebbling price p , as constructed in Lemma 6.2. The resulting graph is a stack of modified pyramids of increasing sizes. If there is justice in this world, the resulting graph should be called a Christmas Tree; though a graph theorist may have a hard time calling this a “tree”.

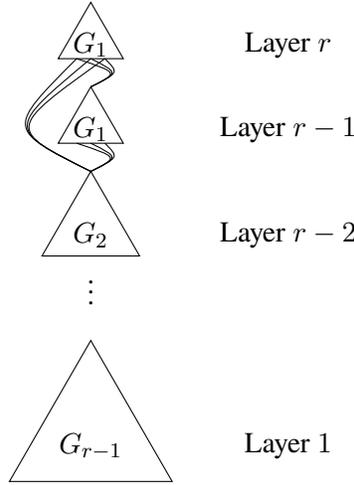


Figure 10: Illustration of a Christmas Tree in Construction 6.3.

Construction 6.3 (Christmas Tree). Let $\{G_p\}_{p \in \mathbb{N}^+}$ be given as in Lemma 6.2. Given $r \in \mathbb{N}^+$, construct a graph $T_r := (V, E)$ as follows. Its vertex set $V := V^1 \dot{\cup} V^2 \dot{\cup} \dots \dot{\cup} V^r$ is a disjoint union of r layers, where for $1 \leq t < r$ the t^{th} layer is a copy of G_{r-t} with vertices $V^t := V(G_{r-t})$, and the top-most layer is another copy of G_1 with vertices $V^r := V(G_1)$. Its edge set $E := E_{\text{intra}} \dot{\cup} E_{\text{inter}}$ consists of intra- and inter-layer edges. The intra-layer edges $E_{\text{intra}} := E^1 \dot{\cup} E^2 \dot{\cup} \dots \dot{\cup} E^r$ come from the corresponding copies of G_p , i.e., for $1 \leq t < r$ the edges on the t^{th} layer E^t are copies of $E(G_{r-t})$ and the edges on the top-most layer E^r are copies of $E(G_1)$. The inter-layer edges E_{inter} connect, for each $1 \leq t \leq r - 2$, the sink of the subgraph at layer t with all sources of the subgraphs at layer $t + 1$ and $t + 2$, and also connect the sink of the copy of G_1 at layer $r - 1$ with the sources of the copy of G_1 at layer r .

Lemma 6.4 (Christmas Tree). *The family of graphs $\{T_r\}_{r \in \mathbb{N}^+}$ satisfies*

1. $RPeb^V(T_r) = RPeb(T_r) = r$;
2. T_r has in-degree at most two and a unique sink; and
3. T_r is polynomial-time computable given r , and T_r has at most r^3 .

Proof. For Item 3, note that each of the r layers has at most r^2 nodes.

For Item 2, if v is a node in T_r we have two cases depending on whether v is some layer's source node or not. If it is, then at most two inter-layer edges from lower layers point to v , and no intra-layer edge does. If v is not a source on any layer then only two intra-layer edges point to it, since all G_p have fan-in at most 2. The only sink of T_r is the sink of layer r .

To see that $RPeb^S(T_r) \leq r - 1$, and thus that $RPeb(T_r) \leq r$, persistently pebble the sink node of G_{r-t} on layer t , for t from 1 to $r - 1$, keeping only the pebbles on the sinks of the previous layers. To persistently pebble layer t takes $r - t$ pebbles, assuming there is a pebble on each of the sinks of the $t - 1$ lower layers, so in total $r - 1$ pebbles suffice. Note that G_1 is a single node, hence when the sinks of the lower layers are all pebbled the sink of T_r is surrounded. The bound $RPeb(T_r) \leq r$ follows by Proposition 2.1.

To see that $RPeb^V(T_r) \geq r$, we argue that when visiting layer r there is a v -locked pebble configuration on each of the previous layers (see Definition 6.1). In particular, any layer with a pebble on the sink has a v -locked configuration and if a configuration is v -locked, then it contains a pebble. Given a pebbling configuration on T_r , for the rest of this proof we say that layer t is v -locked if the configuration, restricted to the corresponding subgraph, is v -locked for the subgraph.

Claim 6.5 (Christmas Tree Locker). *Consider any pebbling that uses less than r pebbles. In such a pebbling, whenever some layer $(t - 1)$ and layer t contain some pebbles, for $2 \leq t \leq r$, then layers $1, \dots, t - 2$ are all v -locked.*

Proof. The claim is true for $t = 2$ vacuously, establishing the base case. When $t > 2$, consider a time that layer t starts to have a pebble: a source node on layer t is pebbled, hence there are pebbles on the sink nodes of layers $t - 1$ and $t - 2$. Thus layers $t - 1$ and $t - 2$ are v -locked and each has a pebble. Induction hypothesis (on $t - 1$) further says that layer η is v -locked for any $1 \leq \eta < t - 2$. As long as there are pebbles on layers t and $t - 1$, all lower layers remain v -locked: for $1 \leq \eta \leq t - 2$, to unlock layer η takes $RPeb^V(G_{r-\eta}) \geq RPeb^S(G_{r-\eta}) = r - \eta - 1 \geq r - t + 1$ pebbles (recall Eq. (2.3)), but there are $t - 1$ pebbles on layers other than η , which cannot be done with less than r pebbles. \square

Assume for some $r \geq 2$, the sink node of layer r is pebbled using less than r pebbles. When a source node of layer r is pebbled, there is a pebble on the sink node of layer $r - 1$. Claim 6.5 shows that there is a pebble on layer η for $1 \leq \eta \leq r - 2$, for a total of r pebbles, contradicting that less than r pebbles are used. This shows $RPeb^V(T_r) \geq r$ for $r \geq 2$, and the case for $r = 1$ is obvious. In the end we get that

$$r \leq RPeb^V(T_r) \leq RPeb(T_r) = RPeb^S(T_r) + 1 \leq r \tag{6.1}$$

by equation 2.3 and Proposition 2.1, which gives Item 1. \square

6.2 Molding

Given a graph G we want to construct a graph $M(G)$ with a special source s and a single sink, such that any pebbling that visits the sink must go through a configuration with at least $RPeb^V(M(G))$ pebbles, one of which is on vertex s .

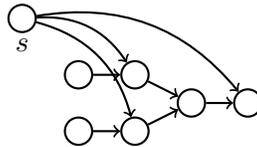


Figure 11: Example of Construction 6.6: molding of a pyramid of height 1.

Construction 6.6 (Molding). Given a graph G , we construct a graph $M(G)$ as follows. For every vertex $v \in V(G)$, we add to $M(G)$ two vertices v_{in} and v_{out} , and a directed edge $(v_{\text{in}}, v_{\text{out}})$. Also, for every edge $(u, v) \in E(G)$, we add to $M(G)$ a corresponding edge $(u_{\text{out}}, v_{\text{in}})$. Finally we add to $M(G)$ a special new vertex s that we connect to all vertices v_{out} , i.e., for every $v \in V(G)$ we add edge (s, v_{out}) to $M(G)$. Formally, $V(M(G)) := \{s\} \dot{\cup} \{v_{\text{in}}, v_{\text{out}} : v \in V(G)\}$ and $E(M(G)) := E_1 \dot{\cup} E_2 \dot{\cup} E_3$, where $E_1 := \{(v_{\text{in}}, v_{\text{out}}) : v \in V(G)\}$, $E_2 := \{(u_{\text{out}}, v_{\text{in}}) : (u, v) \in E(G)\}$ and $E_3 := \{(s, v_{\text{out}}) : v \in V(G)\}$.

By construction, if G has in-degree at most two and a unique sink then so does $M(G)$.

Lemma 6.7 (Molding). *Given a graph G , the graph $M(G)$ has the following properties.*

1. $R\text{Peb}(M(G)) \leq R\text{Peb}(G) + 2$; and
2. For any visiting pebbling $\mathcal{P}' = \langle \mathbb{P}'_0, \mathbb{P}'_1, \dots, \mathbb{P}'_\tau \rangle$ of $M(G)$, there is a configuration \mathbb{P}'_b (for some $0 \leq b \leq \tau$) using at least $R\text{Peb}^V(G) + 2$ pebbles and containing s .

Proof. For Item (1), fix any persistent pebbling \mathcal{P} of G . Simulate the pebbling \mathcal{P} as a persistent pebbling \mathcal{P}' of $M(G)$ as follows. First, pebble the special source s of $M(G)$ in \mathcal{P}' . Afterwards, whenever there is a move in \mathcal{P} to pebble a node $v \in V(G)$, make a phase of three moves in \mathcal{P}' : pebble v_{in} , pebble v_{out} , unpebble v_{in} . Similarly, whenever there is a move in \mathcal{P} to unpebble a node $v \in V(G)$, make a phase of three moves in \mathcal{P}' : pebble v_{in} , unpebble v_{out} , unpebble v_{in} . If the current configuration in \mathcal{P} is \mathbb{P} , and the configuration in \mathcal{P}' at the end of a phase is \mathbb{P}' , then \mathcal{P}' maintains the invariant that $\mathbb{P}' = \{s\} \cup \{v_{\text{out}} : v \in \mathbb{P}\}$. As a result, \mathcal{P}' is a legal pebbling on $M(G)$: whenever v_{in} is pebbled or unpebbled, all its predecessors $\text{pred}(v_{\text{in}}) = \{u_{\text{out}} : u \in \text{pred}(v)\}$ have pebbles in \mathbb{P}' , since $\text{pred}(v)$ have pebbles in \mathbb{P} ; whenever v_{out} is pebbled or unpebbled, its predecessors s and v_{in} have pebbles. If we add a final move in \mathcal{P}' to unpebble s , then \mathcal{P}' persistently pebbles $M(G)$. Note that whenever \mathcal{P} pebbles or unpebbles $v \in V(G)$ to get to configuration \mathbb{P} , the simulating pebbling \mathcal{P}' uses at most two more pebbles to get to \mathbb{P}' , namely s and v_{in} . Hence $R\text{Peb}(M(G)) \leq R\text{Peb}(G) + 2$.

For Item (2), we start with a visiting pebbling $\mathcal{P}' = \langle \mathbb{P}'_0, \mathbb{P}'_1, \dots, \mathbb{P}'_\tau \rangle$ of $M(G)$ and we construct a visiting pebbling \mathcal{P} of G . We now define two projection operators that turn pebble configurations for $M(G)$ into configurations for G . Let $\text{proj}_{\text{out}}(\mathbb{P}'_t) := \{v \in V(G) : v_{\text{out}} \in \mathbb{P}'_t\}$ be the projection of \mathbb{P}'_t to $V(G)$ via v_{out} , and $\text{proj}_{\text{any}}(\mathbb{P}'_t) := \{v \in V(G) : v_{\text{in}} \in \mathbb{P}'_t \text{ or } v_{\text{out}} \in \mathbb{P}'_t\}$ be the projection of \mathbb{P}'_t to $V(G)$ via v_{in} or v_{out} . By definition $\text{proj}_{\text{out}}(\mathbb{P}'_t) \subseteq \text{proj}_{\text{any}}(\mathbb{P}'_t)$. Whenever a vertex of $M(G)$ is pebbled or unpebbled during a pebbling step from \mathbb{P}'_t to \mathbb{P}'_{t+1} ,

- (i) if the vertex is s , then both $\text{proj}_{\text{out}}(\mathbb{P}'_t) = \text{proj}_{\text{out}}(\mathbb{P}'_{t+1})$ and $\text{proj}_{\text{any}}(\mathbb{P}'_t) = \text{proj}_{\text{any}}(\mathbb{P}'_{t+1})$;
- (ii) if the vertex is v_{in} for some $v \in V(G)$, then $\text{proj}_{\text{out}}(\mathbb{P}'_t) = \text{proj}_{\text{out}}(\mathbb{P}'_{t+1})$ but $\text{proj}_{\text{any}}(\mathbb{P}'_t)$ may differ from $\text{proj}_{\text{any}}(\mathbb{P}'_{t+1})$; and
- (iii) if the vertex is v_{out} for some $v \in V(G)$, then $\text{proj}_{\text{any}}(\mathbb{P}'_t) = \text{proj}_{\text{any}}(\mathbb{P}'_{t+1})$ but $\text{proj}_{\text{out}}(\mathbb{P}'_t) \neq \text{proj}_{\text{out}}(\mathbb{P}'_{t+1})$.

To construct \mathcal{P} , we analyze in order each configuration \mathbb{P}'_t in \mathcal{P}' . Depending on how the sequences of $\text{proj}_{\text{out}}(\mathbb{P}'_t)$ and $\text{proj}_{\text{any}}(\mathbb{P}'_t)$ evolve, we may append new configurations to \mathcal{P} . In the following η is the index of the last configuration added to \mathcal{P} and t is the configuration of \mathcal{P}' under analysis. We maintain the following invariants:

- (a) $\text{proj}_{\text{out}}(\mathbb{P}'_t) \subseteq \mathbb{P}_\eta$; and
- (b) for any v in $\text{proj}_{\text{any}}(\mathbb{P}'_t) \triangle \mathbb{P}_\eta$ it holds that $\text{pred}(v) \subseteq \text{proj}_{\text{out}}(\mathbb{P}'_t)$.

Initially at $t = 0$ and $\eta = 0$, $\mathbb{P}'_t = \text{proj}_{\text{out}}(\mathbb{P}'_t) = \text{proj}_{\text{any}}(\mathbb{P}'_t) = \mathbb{P}_\eta = \emptyset$, so the invariant holds for $t = 0$. Consider a pebbling move in \mathcal{P}' from \mathbb{P}'_t to \mathbb{P}'_{t+1} .

- (I) If $\text{proj}_{\text{any}}(\mathbb{P}'_t) = \text{proj}_{\text{any}}(\mathbb{P}'_{t+1})$ and $\text{proj}_{\text{out}}(\mathbb{P}'_t) = \text{proj}_{\text{out}}(\mathbb{P}'_{t+1})$ the construction does not append any new \mathbb{P}_η and the invariant is preserved.
- (II) If $\text{proj}_{\text{any}}(\mathbb{P}'_t) \neq \text{proj}_{\text{any}}(\mathbb{P}'_{t+1})$ then we are in case (ii) above, so $\text{proj}_{\text{out}}(\mathbb{P}'_t)$ does not change and some node v_{in} is pebbled or unpebbled. Hence the current configuration $\mathbb{P}'_t \supseteq \text{pred}(v_{\text{in}}) = \{u_{\text{out}} : u \in \text{pred}(v)\}$, thus $\text{proj}_{\text{out}}(\mathbb{P}'_t) \supseteq \text{pred}(v)$. The construction does not append a new \mathbb{P}_η and the invariant is preserved.
- (III) If $\text{proj}_{\text{out}}(\mathbb{P}'_t) \neq \text{proj}_{\text{out}}(\mathbb{P}'_{t+1})$ then we are in case (iii) above, so $\text{proj}_{\text{any}}(\mathbb{P}'_t)$ does not change and some node v_{out} is pebbled or unpebbled. The construction appends to \mathcal{P} the two sequences of moves (“Eras”) described below. After each move $\text{proj}_{\text{any}}(\mathbb{P}'_t) \triangle \mathbb{P}_\eta$ gets smaller. Note that for any $u \in \text{proj}_{\text{any}}(\mathbb{P}'_t) \triangle \mathbb{P}_\eta$, the invariant gives $\text{pred}(u) \subseteq \text{proj}_{\text{out}}(\mathbb{P}'_t) \subseteq \mathbb{P}_\eta$, so they can be pebbled or unpebbled in \mathbb{P}_η .

Unpebble Era while $\mathbb{P}_\eta \setminus \text{proj}_{\text{any}}(\mathbb{P}'_t) \neq \emptyset$, pick any $u \in \mathbb{P}_\eta \setminus \text{proj}_{\text{any}}(\mathbb{P}'_t)$, and unpebble u in \mathcal{P} (increment $\eta := \eta + 1$ and then set $\mathbb{P}_\eta := \mathbb{P}_{\eta-1} \setminus \{u\}$). Since $\text{proj}_{\text{out}}(\mathbb{P}'_t) \subseteq \text{proj}_{\text{any}}(\mathbb{P}'_t)$, $u \notin \text{proj}_{\text{out}}(\mathbb{P}'_t)$ and the invariant is preserved at time t .

Pebble Era while $\text{proj}_{\text{any}}(\mathbb{P}'_t) \setminus \mathbb{P}_\eta \neq \emptyset$, pick any $u \in \text{proj}_{\text{any}}(\mathbb{P}'_t) \setminus \mathbb{P}_\eta$, and pebble u in \mathcal{P} (increment $\eta := \eta + 1$ and then set $\mathbb{P}_\eta := \mathbb{P}_{\eta-1} \cup \{u\}$). The invariant is preserved at time t .

At the end of the two sequences, $\text{proj}_{\text{out}}(\mathbb{P}'_{t+1}) \subseteq \text{proj}_{\text{any}}(\mathbb{P}'_{t+1}) = \text{proj}_{\text{any}}(\mathbb{P}'_t) = \mathbb{P}_\eta$, so the invariant now holds also at time $t + 1$.

We complete the proof of Item (2). For any visiting pebbling \mathcal{P}' of $M(G)$, the corresponding pebbling \mathcal{P} is a visiting pebbling of G by invariant (a), thus some constructed configuration \mathbb{P}_η has at least $\text{RPeb}^V(G)$ pebbles. The configuration has been appended to \mathcal{P} in case (III) above, and without loss of generality we can assume it is either at the beginning of an “unpebble era” or at the end of a “pebble era”, since the number of pebbles in \mathbb{P}_η decreases in the former and increases in the latter. Since the beginning of an “unpebble era” other than the first is also the end of a “pebble era”, we can furthermore assume the latter. This means that for some t we have $\text{proj}_{\text{any}}(\mathbb{P}'_t) = \text{proj}_{\text{any}}(\mathbb{P}'_{t+1}) = \mathbb{P}_\eta$, so either the corresponding configuration \mathbb{P}'_t (when v_{out} is unpebbled) or \mathbb{P}'_{t+1} (when v_{out} is pebbled) has at least $\text{RPeb}^V(G) + 2$ pebbles, including s , v_{in} and v_{out} . This completes Item (2). \square

6.3 Turnpikes

As an application of molding (Construction 6.6) we show a construction that controls the visiting price of a node, and that allows us to construct gadgets for the components of a quantified boolean formula.

Construction 6.8 (Turnpike). For any non-negative integer r we define the *turnpike* of toll r from a to b (represented by Fig. 12) as follows. If $r = 0$ then the turnpike just joins the vertices a and b with an edge (a, b) . If $r > 0$ let T_r be the graph having $\text{RPeb}^V(T_r) = \text{RPeb}(T_r) = r$ given by Lemma 6.4. The turnpike of toll r from a to b is the graph $M(T_r)$, identifying a with the special source s in $M(T_r)$, and identifying b with the unique sink in $M(T_r)$.

Let G be any graph that contains a turnpike of toll r from a to b . Call the nodes $\tilde{R} := \text{anc}_G(b) \setminus \text{anc}_G(a)$ to be *properly* in the turnpike, and call the nodes $R := \tilde{R} \cup \{a\}$ to be in the turnpike. The turnpike construction is sensitive to whether the pebble on node a is counted, i.e., whether the pebbling prices are restricted to R or to \tilde{R} .

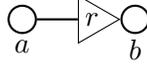


Figure 12: Turnpike of toll r from a to b .

Lemma 6.9 (Turnpike). *We have $RPeb_R(b) = RPeb_R^V(b) = r + 2$ and $RPeb_{\tilde{R}}(b) = RPeb_{\tilde{R}}^V(b) = r + 1$.*

Proof. $RPeb_R(b) \leq r + 2$ and $RPeb_{\tilde{R}}(b) \leq r + 1$ by (the proof of) Item (1) of Lemma 6.7 (since pebbles outside of R or of \tilde{R} are not counted), and $RPeb_R^V(b) \geq r + 2$ and $RPeb_{\tilde{R}}^V(b) \geq r + 1$ by Item (2) of Lemma 6.7. \square

The fact that $RPeb_R(b)$ and $RPeb_{\tilde{R}}(b)$ do not differ by more than one holds not only for turnpikes but for any graph.

Lemma 6.10 (Source Difference). *Consider regions R_1 and R_2 such that $R_1 = R_2 \setminus \{s_2\}$ for some source s_2 of R_2 (i.e., $\text{pred}(s_2) \cap R_2 = \emptyset$). We have*

$$\begin{aligned} RPeb_{R_1}(v) &\leq RPeb_{R_2}(v) \leq RPeb_{R_1}(v) + 1 \\ RPeb_{R_1}^V(v) &\leq RPeb_{R_2}^V(v) \leq RPeb_{R_1}^V(v) + 1 \\ RPeb_{R_1}^S(v) &\leq RPeb_{R_2}^S(v) \leq RPeb_{R_1}^S(v) + 1 . \end{aligned}$$

7 PSPACE-Completeness

In this section we give all details of the construction in Theorem 3.2, restated in the following theorem, and its full proof.

Theorem 7.1. *Given a quantified 3-CNF ϕ , there is a polynomial-time constructible graph $G(\phi)$ and a polynomial-time computable number $\gamma(\phi)$ such that $RPeb(G(\phi)) = \gamma(\phi) + \llbracket \phi \text{ is FALSE} \rrbracket$.*

Let $x_1 \dots x_n$ be the variables of ϕ . We sort the variables from the innermost to the outermost quantifier and denote by ϕ_i the quantified 3-CNF with just the i innermost quantifiers, namely $\phi_0 = \bigwedge_{j=1}^m C_j$, $\phi_i = Q_i x_i \phi_{i-1}$ for $Q_i \in \{\forall, \exists\}$, and $\phi = \phi_n$. For each ϕ_i we consider the gadget $G(\phi_i)$, where $G(\phi_0)$ is built as defined in Construction 7.24 and each $G(\phi_i)$ for $i \in [n]$ is built according to either Construction 7.26 or Construction 7.31, depending on the i^{th} innermost quantifier. Furthermore we fix the sequence of integers $\{\gamma_i\}_{i=0}^n$ where $\gamma_0 := 2m + 7$, $\gamma_i := 3 + \gamma_{i-1}$ if the i^{th} innermost quantifier is existential, and $\gamma_i = 5 + \gamma_{i-1}$ if the i^{th} innermost quantifier is universal. To analyze the gadgets for the subformulas we need the next definition.

Definition 7.2. A *region* is an induced subgraph of the final gadget $G(\phi)$ (or of component gadgets as we build up the final gadget). We slightly abuse notation and refer to a region by a subset of vertices; for example, given a subset of vertices \tilde{R} of a gadget G , we use $RPeb_{\tilde{R}}(G)$, $RPeb_{\tilde{R}}^V(G)$ and $RPeb_{\tilde{R}}^S(G)$ to denote the different pebbling prices over the subgraph of G induced on \tilde{R} .

With these notations and definitions in place, Theorem 7.1 follows immediately from the next lemma when i is equal to n . In this case ρ and \tilde{R} , as stated in the lemma, are respectively the empty assignment and the full graph $G(\phi)$. In the proof of the lemma we refer to definitions, lemmas and constructions that we will present in full details in the coming subsections.

Lemma 7.3 (Main Lemma). *Fix an arbitrary $0 \leq i \leq n$ and let*

- ρ be an assignment to all but the first i variables;
- S be the canonical set of ρ according to Definition 7.11;
- \check{R} be the subset of vertices of $G(\phi_i)$ defined as $V(G(\phi_i)) \setminus \text{anc}(S)$.

It holds that

$$RPeb_{\check{R}}(G(\phi_i)) = \gamma_i + \llbracket \phi_i \upharpoonright_{\rho} \text{ is FALSE} \rrbracket . \quad (7.1)$$

Proof. When $i = 0$, the gadget $G(\phi_0)$ is the CNF gadget from Construction 7.24 and the base case follows immediately by Lemma 7.25, where $\beta_m = 2m$. When $i > 0$, consider the two possible extensions of ρ that assign x_i , namely $\rho_0 := \rho \cup \{x_i = 0\}$ and $\rho_1 := \rho \cup \{x_i = 1\}$. Consider also the corresponding canonical sets S_0, S_1 and the regions $\check{R}_0 := V(G(\phi_{(i-1)})) \setminus \text{anc}(S_0)$ and $\check{R}_1 := V(G(\phi_{(i-1)})) \setminus \text{anc}(S_1)$. By induction it holds that

- $RPeb_{\check{R}_0}(G(\phi_{(i-1)})) = \gamma_{(i-1)} + \llbracket \rho_0 \text{ falsifies } \phi_{(i-1)} \rrbracket$
- $RPeb_{\check{R}_1}(G(\phi_{(i-1)})) = \gamma_{(i-1)} + \llbracket \rho_1 \text{ falsifies } \phi_{(i-1)} \rrbracket$

so the hypothesis of Lemmas 7.37 and 7.30 holds. When $\phi_i = \forall x_i \phi_{i-1}$ Lemma 7.37 gives that

$$RPeb(G(\phi_i)) = 5 + \gamma_{(i-1)} + \llbracket \rho_0 \text{ falsifies } \phi_{(i-1)} \text{ or } \rho_1 \text{ falsifies } \phi_{(i-1)} \rrbracket , \quad (7.2)$$

and $\phi_i = \exists x_i \phi_{(i-1)}$ Lemma 7.30 gives that

$$RPeb(G(\phi_i)) = 3 + \gamma_{(i-1)} + \llbracket \rho_0 \text{ falsifies } \phi_{(i-1)} \text{ and } \rho_1 \text{ falsifies } \phi_{(i-1)} \rrbracket . \quad (7.3)$$

Equations (7.2) and (7.3) are equivalent, for the respective quantifier type, to Equation (7.1). \square

7.1 Literal Gadget

Lemma 6.4 allows us to create an edge (u, v) with $RPeb(v) = RPeb(u) + 1 = r + 1$ given any $r \in \mathbb{N}^+$. This is used for constructing gadgets for literals (represented as Figure 13).

Construction 7.4 (Literal Gadget). Fix a literal ℓ and an integer $r \in \mathbb{N}^+$. Let T_r be the graph having $RPeb^V(T_r) = RPeb(T_r) = r$ given by Lemma 6.4. The literal gadget of price r for ℓ , is denoted as $L_\ell(r)$. To construct it, take a copy of T_r and call its sink ℓ' . Then add a new node, named ℓ as the corresponding literal, and add the edge (ℓ', ℓ) .



Figure 13: Literal gadget of price r for ℓ .

Lemma 7.5 (Literal Gadget). In $L_\ell(r)$, $RPeb(\ell) - 1 = RPeb(\ell') = RPeb^V(\ell') = r$.

Proof. Note that visiting node ℓ' is equivalent to surrounding node ℓ . Hence $RPeb(\ell) - 1 = RPeb^S(\ell) = RPeb^V(\ell') = RPeb(\ell') = r$ by (2.3), Proposition 2.1, and Lemma 6.4. \square

We associate pebbling configurations on $L := L_r(\ell)$ with truth value `TRUE`, `FALSE` or `*` (undefined) as follows.

Definition 7.6 (Literal Position). Fix a literal gadget $L := L_r(\ell)$. Given a pebbling configuration \mathbb{P} , say node ℓ' is *v-locked* on $\text{anc}_L(\ell')$ if $R\text{Peb}_L(\mathbb{P}) = R\text{Peb}^V(\ell') = r$ (when restricted to $\text{anc}_L(\ell')$); that is, if the empty configuration cannot be reached without entering a configuration which has r pebbles. Given a pebbling configuration on L , say literal ℓ is in

- `TRUE` position, if node ℓ has a pebble and node ℓ' is not v-locked;
- `FALSE` position, if node ℓ' is v-locked; and
- `*` position, if node ℓ has no pebble and node ℓ' is not v-locked.

A *transition* of literal ℓ is a change of position for ℓ (for instance from `TRUE` position to `FALSE` position, or from `FALSE` position to `*` position) due to a pebble move. Finally, we identify certain *canonical* positions with configurations on L as follows:

- the canonical `TRUE` position is the configuration where only node ℓ has a pebble;
- the canonical `FALSE` position is the configuration where only node ℓ' has a pebble; and
- the canonical `*` position is the empty configuration.

Clearly, the canonical `TRUE` position (resp. canonical `FALSE` position, canonical `*` position) is indeed a `TRUE` position (resp. `FALSE` position, `*` position).

Lemma 7.7 (Literal Transition). Fix a literal gadget $L := L_r(\ell)$.

1. it is impossible to transition directly from `*` position to `TRUE` position, and vice versa; and
2. at a transition, there are r pebbles on $\text{anc}_L(\ell')$.

Proof. Note that node ℓ' is v-locked on any configuration with a pebble on node ℓ' . To change from `*` position to `TRUE` position, node ℓ needs to be pebbled, and hence node ℓ' must have a pebble at some time. At that time, node ℓ' is v-locked, so the configuration is in `FALSE` position. This gives Item (1).

Hence the only valid transitions are from `*` position to `FALSE` position (or its reverse), and from `FALSE` position to `TRUE` position (or its reverse). In any of these, node ℓ' needs to switch between locked and unlocked status, which requires a configuration with r pebbles on $\text{anc}_L(\ell')$ by definition of v-locked. This gives Item (2). \square

7.2 Variable Gadget

Let $r_i \in \mathbb{N}^+$ be an integer to be specified later, which is associated with the i^{th} variable x_i .

Inspired by previous works [GLT80, HP10, Cha13b] truth values are represented using the gadget in Figure 14.

Construction 7.8 (Variable Gadget). For the variable x_i , its variable gadget $G(x_i)$ is constructed as the disjoint union two literal gadgets of price r_i , one for literal x_i and one for literal \bar{x}_i .

For the gadget $G := G(x_i)$, its nodes are $V(G) := \text{anc}_G(\{x_i, \bar{x}_i\})$.

Definition 7.9 (Variable Position). Fix a variable gadget $G := G(x_i)$ consisting of literal gadgets $L_1 := L_{r_i}(x_i)$ and $L_0 := L_{r_i}(\bar{x}_i)$. We identify certain *canonical* positions with configurations on G as follows:

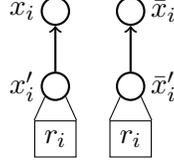


Figure 14: Variable x_i .

- the canonical **TRUE** position is the configuration where only nodes x_i and \bar{x}'_i have pebbles;
- the canonical **FALSE** position is the configuration where only nodes x'_i and \bar{x}_i have pebbles; and
- the canonical ***** position is the empty configuration.

Lemma 7.10 (Variable Assignment). *Variable x_i can be put into one among canonical **TRUE** and canonical **FALSE** positions using at most $r_i + 1$ pebbles.*

Proof. To put variable x_i in canonical **TRUE** position, persistently pebble node x_i , then persistently pebble \bar{x}'_i . It can be done with $r_i + 1$ pebbles by Lemma 7.5. A symmetric argument shows that x_i can be put in canonical **FALSE** position with $r_i + 1$ pebbles. \square

As we will see in later sections, the design of the quantifier gadgets would ensure that any pebbling strategy would effectively associate truth value via the canonical positions. This motivates the following definition.

Definition 7.11 (Canonical Nodes). Given a partial assignment $\rho: [n] \rightarrow \{\text{TRUE}, \text{FALSE}, *\}$, the canonical nodes of variable x_i under ρ are

- $\{x_i, \bar{x}'_i\}$ if $\rho(i) = \text{TRUE}$;
- $\{x'_i, \bar{x}_i\}$ if $\rho(i) = \text{FALSE}$; and
- $\{\}$ if $\rho(i) = *$.

Note that if $\rho(i) \neq *$, then there are two pebbles on the ancestors of the canonical nodes of x_i . For example, if $\rho(i) = \text{TRUE}$, then there is a pebble on $\text{anc}(x_i)$ and a pebble on $\text{anc}(\bar{x}'_i)$.

In general, we consider a partial assignment on variables $\rho: [n] \rightarrow \{\text{TRUE}, \text{FALSE}, *\}$ as a partial assignment on literals:

- if variable x_i is assigned **TRUE** under ρ (i.e., $\rho(i) = \text{TRUE}$), then literal x_i is assigned **TRUE** and literal \bar{x}_i is assigned **FALSE**;
- if variable x_i is assigned **FALSE** under ρ (i.e., $\rho(i) = \text{FALSE}$), then literal x_i is assigned **FALSE** and literal \bar{x}_i is assigned **TRUE**;
- if variable x_i undefined under ρ (i.e., $\rho(i) = *$), then literal x_i is $*$ and literal \bar{x}_i is $*$.

As we will argue later, the design of the quantifier gadgets would ensure that “invalid variable assignments” would not be a problem: for example, the two literals of the same variable cannot be put into the **TRUE** position at the same time (for instance Claim 7.29); also, it does not help to put the two literals of the same variable into the **FALSE** position at the same time, and each variable will be assigned eventually (Lemmas 7.28 and 7.34).

The canonical nodes (of a partial assignment) are useful for defining certain *regions* over different component gadgets in the overall construction.

7.3 Clause Gadget

Let $\beta_j \geq 2$ be an integer to be specified later, which is associated with the j^{th} clause C_j . The gadget for the j^{th} clause, $C_j = \ell_{j,1} \vee \ell_{j,2} \vee \ell_{j,3}$, uses as a component the turnpike gadget which is described in Construction 6.8. Its skeleton is shown in Figure 15 (the literal gadgets are simplified in Figure 15 for a cleaner diagram). Assume that the literals $\ell_{j,1}, \ell_{j,2}, \ell_{j,3}$ are over distinct variables.

Construction 7.12 (Clause Gadget). Assume that for each variable x_i , $1 \leq i \leq n$ we have the corresponding variable gadget $G(x_i)$, i.e., two literal gadgets for x_i and \bar{x}_i . For the j^{th} clause C_j , its clause gadget $G(C_j)$ is constructed as follows. Create nodes $a_j, b_j, c_j, u_j, v_j, p_j$, and edges

$$(a_j, u_j), (b_j, u_j), (b_j, v_j), (c_j, v_j), (u_j, p_j), (v_j, p_j). \quad (7.4)$$

Finally, add three turnpikes of toll β_j , from $\ell_{j,1}$ to a_j , from $\ell_{j,2}$ to b_j , and from $\ell_{j,3}$ to c_j (where the nodes $\ell_{j,1}, \ell_{j,2}, \ell_{j,3}$ are the ones from the corresponding literal gadgets).

Note that in Figure 15 the six nodes $\ell_{j,1}, \ell'_{j,1}, \ell_{j,2}, \ell'_{j,2}, \ell_{j,3}$, and $\ell'_{j,3}$ come from the variable gadgets corresponding to the variables in literals $\ell_{j,1}, \ell_{j,2}$ and $\ell_{j,3}$. Recall the definitions of canonical nodes in Definition 7.11. For example, if literal $\ell_{j,1}$ is in TRUE position, literals $\ell_{j,2}, \ell_{j,3}$ FALSE position, then their canonical nodes are $\ell_{j,1}, \ell'_{j,2}, \ell'_{j,3}$.

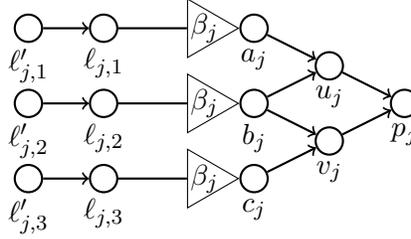


Figure 15: Clause j .

In this subsection, focus on the gadget $G := G(C_j)$ constructed for the j^{th} clause C_j . The gadget G behaves like a disjunction in the sense that at least one literal is assigned TRUE if, and only if, $\beta_j + 3$ additional pebbles are needed to surround p_j .

Lemma 7.13 (True Clause, upper bound). Fix a partial assignment ρ . Assume none of the literals $\ell_{j,1}, \ell_{j,2}, \ell_{j,3}$ is assigned $*$, and at least one of them is assigned TRUE. Let S_j be their canonical nodes. Consider the region $R_j := \text{anc}_G(p_j) \setminus \text{anc}_G(S_j)$ beyond the canonical nodes. Then $\text{RPeb}_{R_j}^S(p_j) \leq \beta_j + 3$.

Proof. Note that the j^{th} clause gadget is symmetric in the three literals $\ell_{j,1}, \ell_{j,2}, \ell_{j,3}$ when restricting attention to $\text{anc}_G(\{a_j, b_j, c_j\}) \setminus \text{anc}_G^*(\{\ell'_{j,1}, \ell'_{j,2}, \ell'_{j,3}\})$. If at least one of the literals $\ell_{j,1}, \ell_{j,2}, \ell_{j,3}$ is assigned TRUE, we claim that it takes at most $\beta_j + 3$ pebbles over R_j to leave pebbles only on $\{a_j, b_j, c_j\}$; afterwards p_j can be surrounded by pebbling u_j and v_j (note that $\beta_j + 3 \geq \underbrace{2}_{u_j, v_j} + \underbrace{3}_{a_j, b_j, c_j} = 5$).

To prove the claim, note that if $\ell_{j,1}$ is assigned TRUE, then the intersection of R_j with the turnpike from $\ell_{j,1}$ to a_j is precisely the nodes *properly* in the turnpike; if $\ell_{j,1}$ is assigned FALSE, then the intersection is precisely the nodes in the turnpike. This holds similarly for literals $\ell_{j,2}$ and $\ell_{j,3}$. By symmetry over $\text{anc}_G(\{a_j, b_j, c_j\}) \setminus \text{anc}_G^*(\{\ell'_{j,1}, \ell'_{j,2}, \ell'_{j,3}\})$, assume $\ell_{j,1}$ is assigned TRUE, and each of $\ell_{j,2}, \ell_{j,3}$ is assigned TRUE or FALSE. Consider the following strategy to place three pebbles on $\{a_j, b_j, c_j\}$, where pebbles outside of R_j are not counted:

1. Persistently pebble the turnpike from $\ell_{j,2}$ to b_j . It takes at most $\beta_j + 2$ pebbles over R_j by Lemma 6.9.
2. Persistently pebble the turnpike from $\ell_{j,3}$ to c_j . Now only b_j and c_j have pebbles over R_j . It takes at most $\underbrace{\beta_j + 2}_{anc(c_j)} + \underbrace{1}_{b_j} = \beta_j + 3$ pebbles over R_j by Lemma 6.9.
3. Persistently pebble the turnpike from $\ell_{j,1}$ to a_j . Since $\ell_{j,1}$ is outside of R_j , it takes at most $\beta_j + 1$ pebbles over the intersection of the turnpike and R_j by Lemma 6.9, for a total of $\underbrace{\beta_j + 1}_{anc(a_j)} + \underbrace{2}_{b_j, c_j} = \beta_j + 3$ pebbles.

Now a_j, b_j and c_j have pebbles. □

Lemma 7.14 (False Clause, lower bound). *Fix a partial assignment ρ . Assume all of the literals $\ell_{j,1}, \ell_{j,2}$ and $\ell_{j,3}$ are assigned FALSE. Let $S_j := \{\ell'_{j,1}, \ell'_{j,2}, \ell'_{j,3}\}$ be their canonical nodes. Consider the region $R_j := anc_G(p_j) \setminus anc_G(S_j)$ beyond the canonical nodes. Then $RPeb_{R_j}^S(p_j) \geq \beta_j + 4$.*

Proof. Note that G consists of a pyramid whose sources are attached to three turnpikes. Since all literals are assigned FALSE, the intersection of R_j with each of the turnpike is precisely the nodes in the turnpike.

Fix an induced subgraph $F \subseteq G$ (for instance $F =$ the turnpike from $\ell_{j,1}$ to a_j) having a unique sink. Say a pebbling configuration on F is *v-locked* if $RPeb_F(\mathbb{P}) = RPeb^V(F)$, that is if in order to reach the empty configuration it is necessary to pass through a configuration with $RPeb^V(F)$ pebbles. In particular, any configuration with a pebble on the sink of F is v-locked on F . Also, if a configuration is v-locked on F , then there is a pebble on F . Given a pebbling configuration on G , say a_j (resp. b_j, c_j) is v-locked if the configuration is v-locked on the turnpike from $\ell_{j,1}$ to a_j (resp. from $\ell_{j,2}$ to b_j , from $\ell_{j,3}$ to c_j).

With locking in mind, consider a “projected” configuration on the pyramid defined as follows. Given a pebbling configuration \mathbb{P} on G , its projection to the pyramid is $proj(\mathbb{P}) := (\{u_j, v_j, p_j\} \cap \mathbb{P}) \cup \{t \in \{a_j, b_j, c_j\} : t \text{ is v-locked under } \mathbb{P}\}$. Note that given a strategy on G , its (configuration-wise) projection to the pyramid is a legal strategy on the pyramid.

We are interested in the truncated paths $\tilde{\pi}$ on the pyramid (i.e., source to sink paths excluding the sink, which are in bijection to the edges $(a_j, u_j), (b_j, u_j), (b_j, v_j), (c_j, v_j)$), and in particular whether they are blocked under $proj(\mathbb{P})$. A truncated path $\tilde{\pi}$ is *blocked* under $proj(\mathbb{P})$ if $\tilde{\pi} \cap proj(\mathbb{P}) \neq \emptyset$.

Consider a strategy on G to surround p_j . Its projection to the pyramid is a strategy on the pyramid to surround p_j . At the beginning, all truncated paths on the pyramid are not blocked; at the end, all truncated paths are blocked. Consider the first time that all truncated paths on the pyramid are blocked: in the strategy projected on the pyramid, this must be the result of pebbling a source node a_j, b_j , or c_j . By symmetry (in the rest of this argument), assume a_j is pebbled, then there are at least two more pebbles on the pyramid in the projected configuration. Since a_j is being pebbled in the projected strategy, a_j is getting v-locked on G (i.e., restricting attention to the turnpike from $\ell_{j,1}$ to a_j , there are as many pebbles as the visiting price of the turnpike), accounting for $\beta_j + 2$ pebbles in the intersection of R_j and the turnpike to a_j by Lemma 6.9. The two other pebbles in the projected strategy each account for one more pebble over R_j , for a total of $\beta_j + 4$ pebbles. □

The lower bound shown in Lemma 7.14 holds for clauses which are falsified. We now prove a weaker lower bound on the surrounding price for the satisfied clauses, which matches the upper bound.

Lemma 7.15 (Any Clause, lower bound). *Fix a partial assignment ρ . Assume none of literals $\ell_{j,1}, \ell_{j,2}$, or $\ell_{j,3}$ is assigned *. Let S_j be their canonical nodes. Consider the region $R_j := anc_G(p_j) \setminus anc_G(S_j)$ beyond the canonical nodes. Then $RPeb_{R_j}^S(p_j) \geq \beta_j + 3$.*

Proof. Follow the proof of Lemma 7.14 to define v-locked, projection to the pyramid, truncated paths on the pyramid, and blocking on the pyramid. The only difference is that, since some literal can be assigned TRUE, the intersection of R_j with some of the turnpike can be the nodes *properly* in the turnpike.

Consider a strategy on G to surround p_j . Its projection to the pyramid is a strategy on the pyramid to surround p_j . As in the proof of Lemma 7.14, consider the first time that all truncated paths on the pyramid are blocked, which in the projected strategy must be the result of pebbling a source node, say, a_j . And there are at least two more pebbles on the pyramid in the projected configuration. Since a_j is being pebbled in the projected strategy, a_j is getting v-locked on G (i.e., restricting attention to the turnpike from $\ell_{j,1}$ to a_j , there are as many pebbles as the visiting price of the turnpike), accounting for $\beta_j + 1$ pebbles in the intersection of R_j and the turnpike to a_j by Lemma 6.9 (note that node $\ell_{j,1}$ may be outside of R_j if literal $\ell_{j,1}$ is in TRUE position). The two other pebbles in the projected strategy each account for one more pebble over R_j , for a total of $\beta_j + 3$ pebbles. \square

Assuming that the literal gadgets are in the position corresponding to ρ , we represent “whether ρ falsifies C_j ” through an increase in the persistent price of the sink of the corresponding gadget, i.e., if ρ satisfies C_j , then the persistent price would be a certain number ($\beta_j + 4$); but if ρ falsifies C_j , then the persistent price would be one plus that number ($\beta_j + 5$). The difference in pebbling prices can be succinctly expressed using the Iverson bracket notation $\llbracket \rho \text{ falsifies } C_j \rrbracket$.

Corollary 7.16 (Clause Gadget). Fix a partial assignment ρ . Assume none of literals $\ell_{j,1}$, $\ell_{j,2}$, or $\ell_{j,3}$ is assigned $*$. Let S_j be their canonical nodes. Consider the region $R_j := \text{anc}_G(p_j) \setminus \text{anc}_G(S_j)$ beyond the canonical nodes. Then $\text{RPeb}_{R_j}(p_j) = \beta_j + 4 + \llbracket \rho \text{ falsifies } C_j \rrbracket$.

Proof. If ρ satisfies C_j , i.e., at least one literal is assigned TRUE, then $\text{RPeb}_{R_j}(p_j) = \beta_j + 4$ because persistent price is one plus surrounding price (Proposition 2.1), and the upper bound (Lemma 7.13) matches the lower bound (Lemma 7.15). If ρ falsifies C_j , i.e., all literals are assigned FALSE, then $\text{RPeb}_{R_j}(p_j) = \beta_j + 5$ because it has to increase (Lemma 7.14) but not by more than one (Lemma 6.10). \square

7.4 Conjunction Gadget

To construct a gadget for the conjunction of m clauses, it suffices to repeatedly compose a gadget for the conjunction two smaller gadgets, using the *conjunction gadget* represented in Figure 16.

Construction 7.17 (Conjunction Gadget). Assume two gadgets G_1 and G_2 with unique sinks are constructed. Construct the *conjunction gadget* of weight r of G_1 and G_2 , denoted $\Lambda_r(G_1, G_2)$, as follows. Call z_1 the sink of G_1 , z_2 the sink of G_2 . Construct nodes d_1, d_2, d_3, d_4, e , and edges $(d_1, d_3), (d_2, d_3), (d_4, e)$. Add a turnpike of toll r from z_1 to d_1 , a turnpike of toll $r - 1$ from z_2 to d_2 , and a turnpike of toll $r - 2$ from d_3 to d_4 .

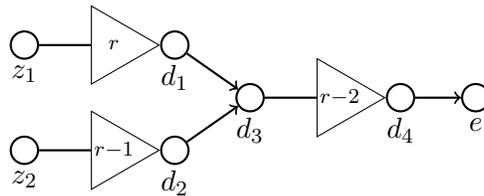


Figure 16: Conjunction gadget of weight r of G_1 and G_2 .

For the gadget $G := \Lambda_r(G_1, G_2)$, the nodes in the gadget are $V(G) = \text{anc}(e) \setminus (\text{anc}^*(z_1) \cup \text{anc}^*(z_2))$. We want to analyze pebbling prices restricted to a certain region \tilde{R} (in the final gadget containing the conjunction gadgets) which will be a superset of the nodes of G .

Lemma 7.18 (True Conjunction). Fix a region \check{R} where $\check{R} \supseteq V(G)$. If $RPeb_{\check{R}}(z_1) \leq r+1$ and $RPeb_{\check{R}}(z_2) \leq r$, then $RPeb_{\check{R}}^S(e) \leq r+2$.

Proof. Consider the following strategy to visit d_4 (equivalently, surround e) using at most $r+2$ pebbles:

- Persistently pebble z_1 , persistently pebble the turnpike from z_1 to d_1 , persistently unpebble z_1 . Now d_1 has a pebble. Over \check{R} , the first sub-step takes at most $r+1$ pebbles, the second sub-step takes at most $r+2$ pebbles by Lemma 6.9, and the third sub-step takes at most $\underbrace{(r+1)}_{anc(z_1)} + \underbrace{1}_{d_1} = r+2$ pebbles.
- Persistently pebble z_2 , persistently pebble the turnpike from z_2 to d_2 , persistently unpebble z_2 . Now d_1 and d_2 have pebbles. Over $anc(d_2) \cap \check{R}$, the first sub-step takes at most r pebbles, the second sub-step takes at most $(r-1) + 2 = r+1$ pebbles by Lemma 6.9, and the third sub-step takes at most $\underbrace{r}_{anc(z_2)} + \underbrace{1}_{d_2}$ pebbles. Over \check{R} , this step takes at most $\underbrace{r+1}_{anc(d_2)} + \underbrace{1}_{d_1} = r+2$ pebbles.
- Pebble d_3 , persistently pebble the turnpike from d_3 to d_4 . Now d_1, d_2, d_3, d_4 have pebbles. Over $\check{R}' := (anc(e) \cap \check{R}) \setminus (anc(d_1) \cup anc(d_2))$, the first sub-step takes 1 pebble, the second sub-step takes at most $(r-2) + 2 = r$ pebbles by Lemma 6.9. Over \check{R} , this step takes at most $\underbrace{r}_{\check{R}'} + \underbrace{2}_{d_1, d_2}$ pebbles. \square

Lemma 7.19 (Any Conjunction). Fix a region \check{R} where $\check{R} \supseteq V(G)$. We have the following:

1. $RPeb_{\check{R}}^S(e) \geq r+2$; and
2. if $RPeb_{\check{R}}(z_1) \leq r+2$ and $RPeb_{\check{R}}(z_2) \leq r+1$, then $RPeb_{\check{R}}^S(e) \leq r+3$.

Proof. For Item (1), note that any strategy to surround e must visit d_1 , and $RPeb_{\check{R}}^V(d_1) \geq r+2$ by Lemma 6.9. Item (2) follows from the proof of Lemma 7.18 to visit d_4 (equivalently, surround e):

1. persistently pebble z_1 , persistently pebble the turnpike from z_1 to d_1 , persistently unpebble z_1 . Now d_1 has a pebble. This step takes at most $r+3$ pebbles over \check{R} .
2. persistently pebble z_2 , persistently pebble the turnpike from z_2 to d_2 , persistently unpebble z_2 . Now d_1 and d_2 have pebbles. This step takes at most $r+3$ pebbles over \check{R} .
3. pebble d_3 , persistently pebble the turnpike from d_3 to d_4 . Now d_1, d_2, d_3, d_4 have pebbles. This step takes at most $r+2$ pebbles over \check{R} . \square

Lemma 7.20 (False Conjunction). Fix a region \check{R} where $\check{R} \supseteq V(G)$. We have the following:

1. if $RPeb_{\check{R}}(z_1) \geq r+2$, then $RPeb_{\check{R}}^S(e) \geq r+3$; and
2. if $RPeb_{\check{R}}(z_2) \geq r+1$, then $RPeb_{\check{R}}^S(e) \geq r+3$.

Proof. Fix a turnpike $T \subseteq G$, say from s to z (for instance if T is the turnpike from z_1 to d_1 , then $s = z_1$ and $z = d_1$). Say a pebbling configuration on T is *s-locked* if the pebbles cannot be cleared without using $RPeb_{\check{R}}^V(T)$ pebbles including one on s (when restricted to T); that is, if the empty configuration cannot be reached without entering a configuration which has $RPeb_{\check{R}}^V(T)$ pebbles and contains s . In particular, any configuration with a pebble on the sink of T is s-locked on T by Lemmas 6.7 and 6.9. Also, if a configuration is s-locked on T , then there is a pebble on some node *properly* in the turnpike (there is a pebble on $V(T) \setminus \{s\}$).

Given a pebbling configuration on G , say d_1 (resp. d_2, d_4) is s-locked if the configuration is s-locked on on the turnpike from z_1 to d_1 (resp. from z_2 to d_2 , from d_3 to d_4).

Fix an induced subgraph $F \subseteq G$ having a unique sink v (for instance $F = \text{anc}(z_1)$ and $v = z_1$). Say a pebbling configuration on F is p -locked if $\text{RPeb}_F(\mathbb{P}) = \text{RPeb}_{\check{R}}(F)$; this is if the pebbles cannot be cleared without using $\text{RPeb}_{\check{R}}(F)$ pebbles (when restricted to F). In particular, the configuration with just a single pebble on v over \check{R} is p -locked on F . Also, if a configuration is p -locked on F , then there is a pebble on $F \cap \check{R}$. Given a pebbling configuration on G , say z_1 (resp. z_2) is p -locked if the configuration is p -locked on $\text{anc}(z_1)$ (resp. $\text{anc}(z_2)$).

Claim 7.21 (s-locked implies p-locked). *Fix any turnpike T on G , say of toll r from s to z . Assume $\text{RPeb}_{\check{R}}(s) \geq r + 2$. In any pebbling that uses at most $r + 2$ pebbles over $\text{anc}(z) \cap \check{R}$, if z is s-locked then s is p -locked.*

Proof. Assume z starts to get s-locked. By definition of s-locked, there are $r + 2$ pebbles on the turnpike T , and s has one of the pebbles. Since at most $r + 2$ pebbles are used over \check{R} , only s has pebble over $\text{anc}(s) \cap \check{R}$, so s is p -locked. Until z is not s-locked, there is one pebble *properly* in the turnpike (i.e., over $V(T) \setminus \{s\}$). Since unlocking s requires $\text{RPeb}_{\check{R}}(s) \geq r + 2$ pebbles over $\text{anc}(s) \cap \check{R}$, until z stops being s-locked, s remains p -locked. \square

Fix a strategy to surround node e , which at some time t_3 must pebble or unpebble d_3 . At time t_3 , both node d_1 and node d_2 have pebbles, hence both node d_1 and node d_2 are s-locked. At the beginning, both node d_1 and node d_2 are not s-locked. Let t_1 (resp. t_2) be the earliest time before time t_3 such that node d_1 (resp. d_2) remains s-locked between time t_1 and time t_3 . Thus node d_1 (resp. d_2) is s-locked at time t_1 (resp. t_2).

Claim 7.22 (s-locked). *If the pebbling uses at most $r + 2$ pebbles over \check{R} to surround e , then*

- (i) $t_1 < t_2$; and
- (ii) at time t_2 , there is exactly one pebble over $\text{anc}(d_1) \cap \check{R}$.

Proof. For Item (i), if $t_2 < t_1$, then at time t_1 there is a pebble on the turnpike from z_2 to d_2 (as node d_2 is already s-locked), and there are $r + 2$ pebbles on the turnpike from z_1 to d_1 by definition of s-locked, for a total of $r + 3$ pebbles over \check{R} .

For Item (ii), there is at least one pebble on the turnpike from z_1 to d_1 as node d_1 is already s-locked, and there is at most one pebble over $\text{anc}(d_1) \cap \check{R}$ since there are at least $(r - 1) + 2 = r + 1$ pebbles on the turnpike from z_2 to d_2 by definition of s-locked, and we assumed that at most $r + 2$ pebbles are used over \check{R} . \square

For Item (1), assume $\text{RPeb}_{\check{R}}(z_1) \geq r + 2$. If at most $r + 2$ pebbles are used over \check{R} to surround node e , then Claim 7.22 shows that at time t_2 , node d_1 is s-locked (as $t_1 < t_2$), and there is exactly one pebble on $\text{anc}(d_1) \cap \check{R}$. Since $\text{RPeb}_{\check{R}}(z_1) \geq r + 2$ and the turnpike from z_1 to d_1 has toll r , Claim 7.21 says that when node d_1 is s-locked (which is the case at time t_2), node z_1 is p -locked. Therefore at time t_2 , there are two pebbles over $\text{anc}(d_1) \cap \check{R}$: one *properly* on the turnpike from z_1 to d_1 (since d_1 is s-locked); and one on $\text{anc}(z_1) \cap \check{R}$ (since z_1 is p -locked). This contradiction shows that $\text{RPeb}_{\check{R}}^S(e) \geq r + 3$.

For Item (2), assume $\text{RPeb}_{\check{R}}(z_2) \geq r + 1$. Fix a strategy using at most $r + 2$ pebbles over \check{R} to surround node e , i.e., to visit node d_4 . At the end, node d_4 is s-locked; at the beginning, node d_4 is not s-locked. Let t_4 be the earliest time such that node d_4 remains s-locked since t_4 until the end. Thus node d_4 is s-locked at time t_4 .

Redefine t_3 if necessary, assume it is the last time before t_4 such that node d_3 is pebbled or unpebbled. Then time t_1 and time t_2 are defined (as above) relative to this t_3 , giving $t_1 < t_2 < t_3 < t_4$ (the first inequality is by Claim 7.22).

Note that at time t_3 , node d_3 is being pebbled: to see this, we know that at time t_4 , the turnpike from d_3 to d_4 is being s-locked, so there is a pebble on node d_3 by definition of s-locked. Since there is no pebble move on node d_3 after time t_3 and before time t_4 , it follows that d_3 is being pebbled at time t_3 , and there is a pebble on node d_3 between time t_3 and time t_4 .

We know that both node d_1 and node d_2 are s-locked at time t_3 . In fact, they remain s-locked between time t_3 and time t_4 : to see this, note that to make node d_1 not s-locked takes $r + 2$ pebbles over $\text{anc}(d_1) \cap \check{R}$, but there are two pebbles outside this region (one on the turnpike from z_2 to d_2 , and one on d_3), which cannot be done with at most $r + 2$ pebbles over \check{R} . Likewise, note that to make node d_2 not s-locked takes $(r - 1) + 2 = r + 1$ pebbles over $\text{anc}(d_2) \cap \check{R}$, but there are two pebbles outside this region (one on the turnpike from z_1 to d_1 , and one on d_3), which cannot be done with at most $r + 2$ pebbles over \check{R} . Therefore, node d_1 is s-locked from time t_1 to time t_4 , and node d_2 is s-locked from time t_2 to time t_4 .

At time t_1 , the turnpike from z_1 to d_1 is getting s-locked, so by definition of s-locked there are $r + 2$ pebbles on the turnpike. By assumption, at most $r + 2$ pebbles are used over \check{R} , so there is no pebble over $\text{anc}(d_2) \cap \check{R}$ at time t_1 . Restrict attention to the sub-strategy \mathcal{P}' between time t_1 and time t_4 . In the sub-strategy \mathcal{P}' , node d_1 remains s-locked, so at most $(r + 2) - 1 = r + 1$ pebbles can be used over $\text{anc}(d_2) \cap \check{R}$. Since $\text{RPeb}_{\check{R}}(z_2) \geq (r - 1) + 2$ and the turnpike from z_1 to d_1 has toll $r - 1$, Claim 7.21 says that when node d_2 is s-locked, node z_2 is p-locked. At time t_4 ,

- node d_1 is s-locked, so there is a pebble properly in the turnpike from z_1 to d_1 ;
- node d_2 is s-locked, hence node z_2 is p-locked, so there is a pebble properly in the turnpike from z_2 to d_2 , and a pebble over $\text{anc}(z_2) \cap \check{R}$; and
- there are $(r - 2) + 2 = r$ pebbles in the turnpike from d_3 to d_4 .

This accounts for $1 + 2 + r = r + 3$ pebbles over \check{R} . This contradiction shows that $\text{RPeb}_{\check{R}}^S(e) \geq r + 3$. \square

Recall that we are going to represent the unsatisfiability of a clause by increased persistent prices, i.e., let q_j be the condition that clause C_j is satisfied, and \bar{q}_j be its negation, then $\text{RPeb}_{\check{R}_j}(p_j) = \beta_j + 4 + \llbracket \bar{q}_j \rrbracket$ by Corollary 7.16.

Corollary 7.23 (Conjunction Gadget). *Fix a region \check{R} where $\check{R} \supseteq V(G)$. Assume that for some conditions q_1 and q_2 it holds that $\text{RPeb}_{\check{R}}(z_1) = r + 1 + \llbracket \bar{q}_1 \rrbracket$ and $\text{RPeb}_{\check{R}}(z_2) = r + \llbracket \bar{q}_2 \rrbracket$. Then $\text{RPeb}_{\check{R}}(e) = r + 3 + \llbracket \bar{q}_1 \wedge \bar{q}_2 \rrbracket$.*

Proof. If both q_1 and q_2 are TRUE, then $\text{RPeb}_{\check{R}}(e) = r + 3$ because persistent price is one plus surround price (Proposition 2.1), and the upper bound (Lemma 7.18) matches the lower bound (Item (1) of Lemma 7.19). If q_1 or q_2 is FALSE, then $\text{RPeb}_{\check{R}}(e) = r + 4$ because the lower bound increases (Lemma 7.20) to match the new upper bound (Item (2) of Lemma 7.19). \square

7.5 CNF Gadget

Assume $\Gamma = C_1 \wedge C_2 \wedge \dots \wedge C_m$ is a conjunction of m clauses. Let $\Gamma_k := \bigwedge_{1 \leq j \leq k} C_j$ be the conjunction of the first k clauses. We will construct a gadget $F_k := G(\Gamma_k)$ for Γ_k with increasing k by successive conjunction of two smaller gadgets, then the CNF gadget for Γ is $G(\Gamma) = G(\Gamma_m)$.

For $1 \leq j \leq m$, let $\beta_j := 2j$, then $\beta_j \geq 2$.

Construction 7.24 (CNF Gadget). Assume for each clause C_j , $1 \leq j \leq m$, a clause gadget $G(C_j)$ is constructed. Let $\Gamma_k := \bigwedge_{1 \leq j \leq k} C_j$ be the conjunction of the first k clauses when $0 \leq k \leq m$. Construct a partial CNF gadget F_k for increasing k as follows. Construct $F_0 := T_7$ as the graph with $\text{RPeb}^V(T_7) = \text{RPeb}(T_7) = 7$ given by Lemma 6.4. Then for $1 \leq k \leq m$, construct $F_k := \Lambda_{\beta_k}(F_{k-1}, G(C_k))$ be

the conjunction gadget of weight β_k of the previous partial CNF gadget (F_{k-1}) and the gadget of clause k ($G(C_k)$). The CNF gadget $G(\Gamma)$ for Γ is F_m .

For the gadget $G := G(\Gamma)$, the nodes in the gadget $V(G)$ contains the nodes of $F_0 = T_7$, the nodes of all clause gadgets $G(C_j)$, and the nodes of all intermediate conjunction gadgets.

Lemma 7.25 (CNF Gadget). *Let ρ be an assignment, and let S be the canonical nodes in all variable gadgets according to ρ (Definition 7.11). Let $\check{R} := V(G) \setminus \text{anc}(S)$ be the region beyond the canonical nodes of ρ . Then $\text{RPeb}_{\check{R}}(F_m) = \beta_m + 7 + \llbracket \rho \text{ falsifies } \Gamma \rrbracket$.*

Proof. For $1 \leq j \leq m$, let q_j be the condition that clause j (C_j) is satisfied by ρ . For $0 \leq k \leq m$, let q'_k be the condition that the first k clauses (Γ_k) are satisfied by ρ . We show by induction that $\text{RPeb}_{\check{R}}(F_k) = \beta_k + 7 + \llbracket q'_k \rrbracket$.

When $k = 0$, Γ_k is satisfied vacuously, so q'_0 is TRUE. The base case holds as $\text{RPeb}_{\check{R}}(F_0) = \text{RPeb}(T_7) = 7$.

For the general case $1 \leq k \leq m$, plug $r = \beta_k + 4$ into Corollary 7.23. Since induction hypothesis gives $\text{RPeb}_{\check{R}}(F_{k-1}) = \beta_{k-1} + 7 + \llbracket q'_{k-1} \rrbracket = \beta_k + 5 + \llbracket q'_{k-1} \rrbracket$, and Corollary 7.16 gives $\text{RPeb}_{\check{R}}(G(C_k)) = \beta_k + 4 + \llbracket q_k \rrbracket$, it follows that $\text{RPeb}_{\check{R}}(F_k) = \beta_k + 7 + \llbracket q'_k \rrbracket$, because $q'_k = q'_{k-1} \wedge q_k$. \square

7.6 Existential Quantifier Gadget

Assume that we already have the gadget $G(\phi_{i-1})$ and that the i^{th} inner-most quantifier is existential, i.e., $Q_i = \exists$. This quantifier refers to x_i , we set the parameter $r_i := \gamma_i - 2$ for the corresponding variable gadget. We construct $G(\phi_i)$ as follows.

Construction 7.26 (Existential Quantifier Gadget). Let q_{i-1} denote the sink of $G(\phi_{i-1})$. Construct nodes f_i, g_i, q_i , and edges $(x_i, g_i), (\bar{x}_i, g_i), (f_i, q_i), (g_i, q_i)$. Add a turnpike of toll $\gamma_i - 5$ from q_{i-1} to f_i .

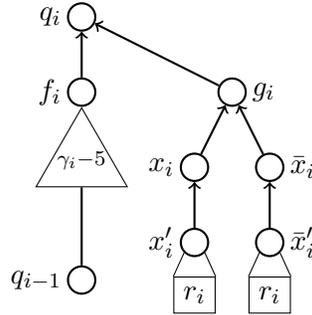


Figure 17: Existentially quantified variable $\exists x_i$.

For the gadget $G := G(\phi_i)$, the nodes in the gadget $V(G)$ contains the nodes in the previous gadget $V(G(\phi_{i-1}))$, the new nodes f_i, g_i, q_i and the nodes in the turnpike from q_{i-1} to f_i .

Lemma 7.27 (Existential Upper Bound). *Assume Lemma 7.3 holds for $i - 1$.*

1. If $\phi_i \upharpoonright_\rho$ is TRUE, then $\text{RPeb}_R^S(q_i) \leq \gamma_i - 1$.
2. If $\phi_i \upharpoonright_\rho$ is FALSE, then $\text{RPeb}_R^S(q_i) \leq \gamma_i$.

Proof. For Item (1), by assumption $\phi_i \upharpoonright_\rho = \exists x_i \phi_{i-1} \upharpoonright_\rho$ is TRUE, so there is an assignment of x_i to TRUE or to FALSE to satisfy ϕ_{i-1} , i.e., $\phi_{i-1} \upharpoonright_{\rho_1}$ is TRUE or $\phi_{i-1} \upharpoonright_{\rho_0}$ is TRUE. Assume the former by symmetry. Since $\phi_{i-1} \upharpoonright_{\rho_1}$ is TRUE, by the assumption that Lemma 7.3 holds for $i-1$, we have $RPeb_{\check{R}_1}(q_{i-1}) = \gamma_{i-1} = \gamma_i - 3$. Consider the following strategy to surround q_i with at most $\gamma_i - 1$ pebbles over \check{R} :

- (i) Put x_i into the canonical TRUE position with $r_i + 1 = \gamma_i - 1$ pebbles by Lemma 7.10. Now node x_i and node \bar{x}'_i have pebbles;
- (ii) Persistently pebble node q_{i-1} using at most $\gamma_i - 3$ pebbles over the new region \check{R}_1 . Now x_i , \bar{x}'_i and q_{i-1} have pebbles. Over the old region \check{R} , at most $(\gamma_i - 3) + 2 = \gamma_i - 1$ pebbles are used (this step is legal as $\check{R} = \check{R}_1 \cup (\text{anc}(x_i) \cup \text{anc}(\bar{x}'_i))$ and x_i and \bar{x}'_i have pebbles);
- (iii) Persistently pebble the turnpike from q_{i-1} to f_i . Now x_i , \bar{x}'_i , q_{i-1} and f_i have pebbles. At most $(\gamma_i - 5) + 2 = \gamma_i - 3$ pebbles are used over the turnpike (including the pebble on node q_{i-1}) by Lemma 6.9, so at most $(\gamma_i - 3) + 2 = \gamma_i - 1$ pebbles are used over \check{R} ; and
- (iv) Pebble node \bar{x}_i then node g_i . Now the six nodes x_i , \bar{x}'_i , \bar{x}_i , q_{i-1} , f_i , g_i have pebbles.

For Item (2), since $\phi_{i-1} \upharpoonright_{\rho_1}$ is FALSE, by the assumption that Lemma 7.3 holds for $i-1$, we have $RPeb_{\check{R}_1}(q_{i-1}) = \gamma_{i-1} + 1 = (\gamma_i - 3) + 1 = \gamma_i - 2$. We run the same strategy as in Item (1) to surround q_i , using at most γ_i pebbles over \check{R} (only Step (ii) uses one more pebble). \square

Lemma 7.28 (Existential Lower Bound). *Assume Lemma 7.3 holds for $i-1$.*

1. $RPeb_{\check{R}}^S(q_i) \geq \gamma_i - 1$.
2. If $RPeb_{\check{R}}^S(q_i) \leq \gamma_i - 1$, then $\phi_i \upharpoonright_\rho$ is TRUE.

Proof. Fix a strategy to surround q_i using at most $\gamma_i - 1$ pebbles over \check{R} . At the end, there is a pebble on the turnpike from q_{i-1} to f_i , and there is a pebble on g_i . Let t_2 be the earliest time such that since t_2 there is at least one pebble on the turnpike from q_{i-1} to f_i . Let t_3 be the earliest time such that since t_3 there is a pebble on g_i .

At time t_3 , node g_i is being pebbled, so both nodes x_i and \bar{x}_i have pebbles, and none of literal x_i or \bar{x}_i is in * position. Let t_0 (resp. t_1) be the last time before time t_3 such that literal \bar{x}_i (resp. literal x_i) has a transition (Definition 7.6). Note that neither literal \bar{x}_i nor x_i can have a transition after time t_3 : to make a transition for literal \bar{x}_i (resp. x_i) takes $r_i = \gamma_i - 2$ pebbles on the ancestors of node \bar{x}'_i (resp. x'_i) by Lemma 7.7, but there is a pebble on the other literal gadget, i.e., on $L_{r_i}(x_i)$ (resp. $L_{r_i}(\bar{x}_i)$), and there is a pebble on g_i , which cannot be done with at most $\gamma_i - 1$ pebbles over \check{R} . So time t_0 (resp. t_1) is in fact the last time that literal \bar{x}_i (resp. x_i) has a transition, and literal \bar{x}_i (resp. x_i) is not in * position since t_0 (resp. t_1).

Assume $t_0 < t_1$ by symmetry (in the rest of this argument).

Claim 7.29 (Clearance). *At time t_1 , there is no pebble over $\check{R}_0 = \check{R} \setminus (\text{anc}(x'_i) \cup \text{anc}(\bar{x}_i))$.*

Proof. At time t_1 , there is a transition of literal x_i , accounting for $r_i = \gamma_i - 2$ pebbles on $\text{anc}(x'_i)$ by Lemma 7.7. And there is a pebble on the other literal gadget $L_{r_i}(\bar{x}_i)$, because literal \bar{x}_i is not in * position. This accounts for at least $\gamma_i - 1$ pebbles over \check{R} . \square

The proof of Claim 7.29 establishes Item (1).

By Claim 7.29, we know that since time t_1 literal x_i is not in TRUE position, hence in FALSE position. As there is no transition of literals \bar{x}_i or x_i after time t_1 , there is a pebble on $\text{anc}(\bar{x}_i)$ and a pebble on $\text{anc}(x'_i)$ since time t_1 . Over the region \check{R}_0 , there are at most $(\gamma_i - 1) - 2 = \gamma_i - 3$ pebbles since time t_1 . Note that region \check{R}_0 is associated with the $(i-1)$ -assignment ρ_0 .

By Claim 7.29, we have $t_0 < t_1 < t_2$, where t_2 is defined (in the first paragraph of this proof) as the earliest time since which there is at least one pebble on the turnpike from q_{i-1} to f_i . Note that at time $t_2 - 1$ there is no pebble on the turnpike from q_{i-1} to f_i , but at the end there is a pebble on f_i . The sub-strategy since time $t_2 - 1$ visits f_i when restricted to the turnpike from q_{i-1} to f_i , so by Lemma 6.7, there is a time t_4 after t_2 such that there are $(\gamma_i - 5) + 2 = \gamma_i - 3$ pebbles over the turnpike, including one pebble on q_{i-1} . As a result, over the region $\check{R}_0 \cap \text{anc}(q_{i-1})$, there is only one pebble at time t_4 , which is on node q_{i-1} . The sub-strategy from time t_1 to t_4 persistently pebble node q_{i-1} over the region $\check{R}_0 \cap \text{anc}(q_{i-1})$ using $\gamma_i - 3 = \gamma_{i-1}$ pebbles, so $\text{RPeb}_{\check{R}_0}^S(G(\phi_{i-1})) \leq \gamma_{i-1}$. By the assumption that Lemma 7.3 holds for $i - 1$, we know $\phi_{i-1} \upharpoonright_{\rho_0}$ is TRUE. As a result, $\exists x_i \phi_{i-1} \upharpoonright_{\rho} = \phi_i \upharpoonright_{\rho}$ is TRUE, giving Item (2). \square

Lemma 7.30 (Existential Quantifier Gadget). *Assume that Lemma 7.3 holds for $i - 1$. We have $\text{RPeb}_{\check{R}}^S(G(\phi_i)) = \gamma_i + \llbracket \phi_i \upharpoonright_{\rho} \text{ is FALSE} \rrbracket$.*

Proof. Since persistent price is one plus surrounding price (Proposition 2.1), it suffices to show that $\text{RPeb}_{\check{R}}^S(G(\phi_i)) = \gamma_i - 1 + \llbracket \phi_i \upharpoonright_{\rho} \text{ is FALSE} \rrbracket$. If $\phi_i \upharpoonright_{\rho}$ is TRUE, then $\text{RPeb}_{\check{R}}^S(G(\phi_i)) = \gamma_i - 1$, as the upper bound (Lemma 7.27) matches the lower bound (Lemma 7.28). If $\phi_i \upharpoonright_{\rho}$ is FALSE, then $\text{RPeb}_{\check{R}}^S(G(\phi_i)) = \gamma_i$, as the upper bound (Lemma 7.27) matches the lower bound (Lemma 7.28). \square

7.7 Universal Quantifier Gadget

Assume that we already have the gadget $G(\phi_{i-1})$ and that the i^{th} inner-most quantifier is existential, i.e., $Q_i = \forall$. This quantifier refers to x_i , we set the parameter $r_i := \gamma_i - 3$ for the corresponding variable gadget. We construct $G(\phi_i)$ as follows.

Construction 7.31 (Universal Quantifier Gadget). Let q_{i-1} denote the sink of $G(\phi_{i-1})$. Construct nodes $f'_i, \bar{f}'_i, f_i, \bar{f}_i, g_i, \bar{g}_i, h_i, \bar{h}_i, q_i$, and edges $(x_i, f'_i), (\bar{x}_i, \bar{f}'_i), (f_i, h_i), (g_i, \bar{h}_i), (\bar{x}_i, \bar{f}'_i), (x'_i, f'_i), (f_i, \bar{h}_i), (\bar{g}_i, \bar{h}_i), (h_i, q_i), (\bar{h}_i, q_i)$. Add a turnpike of toll $\gamma_i - 6$ from f'_i to f_i , a turnpike of toll $\gamma_i - 6$ from \bar{f}'_i to \bar{f}_i , a turnpike of toll $\gamma_i - 7$ from q_{i-1} to g_i , and a turnpike of toll $\gamma_i - 7$ from q_{i-1} to \bar{g}_i .

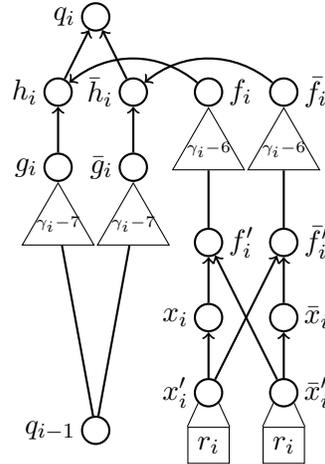


Figure 18: Universally quantified variable $\forall x_i$.

For the gadget $G := G(\phi_i)$, the nodes in the gadget $V(G)$ contains the nodes in the previous gadget $V(G(\phi_{i-1}))$, the new nodes $f'_i, \bar{f}'_i, f_i, \bar{f}_i, g_i, \bar{g}_i, h_i, \bar{h}_i, q_i$, and the nodes in the four turnpikes.

Lemma 7.32 (One-Sided Upper Bound). *Assume Lemma 7.3 holds for $i - 1$.*

1. If $\phi_{i-1} \upharpoonright_{\rho_1}$ is TRUE, then using at most $\gamma_i - 2$ pebbles over \check{R} , we can leave pebbles on nodes $x_i, \bar{x}'_i, f_i, q_{i-1}, g_i$.
2. If $\phi_{i-1} \upharpoonright_{\rho_0}$ is TRUE, then using at most $\gamma_i - 2$ pebbles over \check{R} , we can leave pebbles on nodes $\bar{x}_i, x'_i, \bar{f}_i, q_{i-1}, \bar{g}_i$.
3. If $\phi_{i-1} \upharpoonright_{\rho_1}$ is FALSE, then using at most $\gamma_i - 1$ pebbles over \check{R} , we can leave pebbles on nodes $x_i, \bar{x}'_i, f_i, q_{i-1}, g_i$.
4. If $\phi_{i-1} \upharpoonright_{\rho_0}$ is FALSE, then using at most $\gamma_i - 1$ pebbles over \check{R} , we can leave pebbles on nodes $\bar{x}_i, x'_i, \bar{f}_i, q_{i-1}, \bar{g}_i$.

Proof. For Item (1), since $\phi_{i-1} \upharpoonright_{\rho_1}$ is TRUE, by the assumption that Lemma 7.3 holds for $i - 1$, we have $RPeb_{\check{R}_1}(q_{i-1}) = \gamma_{i-1} = \gamma_i - 5$. Consider the following strategy to leave pebbles on $x_i, \bar{x}'_i, f_i, q_{i-1}, g_i$ using at most $\gamma_i - 2$ pebbles over \check{R} :

- (i) Put x_i into the canonical TRUE position with $r_i + 1 = \gamma_i - 2$ pebbles by Lemma 7.10. Now node x_i and node \bar{x}'_i have pebbles;
- (ii) Pebble f'_i , persistently pebble the turnpike from f'_i to f_i , then unpebble f'_i . Now x_i, \bar{x}'_i and f_i have pebbles. Over the turnpike from f'_i to f_i , at most $(\gamma_i - 6) + 2 = \gamma_i - 4$ pebbles are used by Lemma 6.9. Over \check{R} , at most $(\gamma_i - 4) + \underbrace{2}_{x_i, \bar{x}'_i} = \gamma_i - 2$ pebbles are used.
- (iii) Persistently pebble node q_{i-1} using at most $\gamma_i - 5$ pebbles over the new region \check{R}_1 . Now x_i, \bar{x}'_i, f_i and q_{i-1} have pebbles. Over the old region \check{R} , at most $\underbrace{\gamma_i - 5}_{\check{R}_1} + \underbrace{3}_{x_i, \bar{x}'_i, f_i} = \gamma_i - 2$ pebbles are used (this step is legal since $\check{R} = \check{R}_1 \cup (\text{anc}(x_i) \cup \text{anc}(\bar{x}'_i))$ and since x_i and \bar{x}_i have the only pebbles on $\text{anc}(x_i) \cup \text{anc}(\bar{x}'_i)$);
- (iv) Persistently pebble the turnpike from q_{i-1} to g_i . Now $x_i, \bar{x}'_i, f_i, q_{i-1}$ and g_i have pebbles. Over the turnpike from q_{i-1} to g_i , at most $(\gamma_i - 7) + 2 = \gamma_i - 5$ pebbles are used by Lemma 6.9. Over \check{R} , at most $(\gamma_i - 5) + \underbrace{3}_{x_i, \bar{x}'_i, f_i} = \gamma_i - 2$ pebbles are used.

Item (2) is symmetric to Item (1).

For Item (3), since $\phi_{i-1} \upharpoonright_{\rho_1}$ is FALSE, by the assumption that Lemma 7.3 holds for $i - 1$, we have $RPeb_{\check{R}_1}(q_{i-1}) = \gamma_{i-1} + 1 = (\gamma_i - 5) + 1 = \gamma_i - 4$. We run the same strategy as in Item (1) to leave pebbles on $x_i, \bar{x}'_i, f_i, q_{i-1}, g_i$, using at most $\gamma_i - 1$ pebbles over \check{R} (only Step (iii) uses one more pebble).

Item (4) is symmetric to Item (3). \square

Lemma 7.33 (Universal Upper Bound). Assume Lemma 7.3 holds for $i - 1$.

1. If $\phi_i \upharpoonright_{\rho}$ is TRUE, then $RPeb_{\check{R}}^S(q_i) \leq \gamma_i - 1$.
2. If $\phi_i \upharpoonright_{\rho}$ is FALSE, then $RPeb_{\check{R}}^S(q_i) \leq \gamma_i$.

Proof. For Item (1), by assumption $\phi_i \upharpoonright_{\rho} = \forall x_i \phi_{i-1} \upharpoonright_{\rho}$ is TRUE, so assigning x_i to TRUE and to FALSE both satisfy ϕ_{i-1} , i.e., $\phi_{i-1} \upharpoonright_{\rho_1}$ is TRUE and $\phi_{i-1} \upharpoonright_{\rho_0}$ is TRUE. Consider the following strategy to surround q_i with at most $\gamma_i - 1$ pebbles over \check{R} :

- (i) Run Item (1) of Lemma 7.32 to pebble nodes $x_i, \bar{x}'_i, f_i, q_{i-1}, g_i$, using at most $\gamma_i - 2$ pebbles over \check{R} .

- (ii) Pebble h_i . Now the six nodes $x_i, \bar{x}'_i, f_i, q_{i-1}, g_i, h_i$ have pebbles.
- (iii) Run the reverse of Item (1) of Lemma 7.32 to remove pebbles from nodes $x_i, \bar{x}'_i, f_i, q_{i-1}, g_i$. Now node h_i has a pebble. Over \check{R} , at most $\gamma_i - 2 + \underbrace{1}_{h_i} = \gamma_i - 1$ pebbles are used.
- (iv) Run Item (2) of Lemma 7.32 to pebble nodes $\bar{x}_i, x'_i, \bar{f}_i, q_{i-1}, \bar{g}_i$. Now the six nodes $h_i, \bar{x}_i, x'_i, \bar{f}_i, q_{i-1}, \bar{g}_i$ have pebbles. Over \check{R} , at most $\gamma_i - 2 + \underbrace{1}_{h_i} = \gamma_i - 1$ pebbles are used.
- (v) Pebble \bar{h}_i to surround q_i . Seven nodes have pebbles.

For Item (2), we run the same strategy as in Item (1) to surround q_i , using at most γ_i pebbles over \check{R} (each of Steps (i), (iii), (iv) may use one more pebble by Items (3) and (4) of Lemma 7.32). \square

Lemma 7.34 (Universal Lower Bound). *Assume Lemma 7.3 holds for $i - 1$.*

1. $RPeb_{\check{R}}^S(q_i) \geq \gamma_i - 1$.
2. *If $RPeb_{\check{R}}^S(q_i) \leq \gamma_i - 1$, then $\phi_i \upharpoonright_\rho$ is TRUE.*

Proof. Fix a strategy to surround q_i using at most $\gamma_i - 1$ pebbles over \check{R} . Let $\check{R}_f := \{h_i\} \cup anc(f_i) \setminus anc^*(f'_i)$ be the region to augment h_i to the turnpike from f'_i to f_i , and $\check{R}_{\bar{f}} := \{\bar{h}_i\} \cup anc(\bar{f}_i) \setminus anc^*(\bar{f}'_i)$ be the region to augment \bar{h}_i to the turnpike from \bar{f}'_i to \bar{f}_i . At the end, the region \check{R}_f has a pebble (on h_i) and the region $\check{R}_{\bar{f}}$ has a pebble (on \bar{h}_i). Let t_1 (resp. t_0) be the earliest time such that since time t_1 (resp. t_0) the region \check{R}_f (resp. $\check{R}_{\bar{f}}$) has pebble.

Assume $t_0 < t_1$ by symmetry (in the rest of this argument). At time $t_1 - 1$, there is no pebble on \check{R}_f , and there are pebbles on nodes x_i and \bar{x}'_i (so that node f'_i can be pebbled at time t_1). Hence literal \bar{x}_i is in FALSE position, and literal x_i is not in * position. Note that there is no transition of literals x_i or \bar{x}_i since time t_1 : to make a transition for literal x_i (resp. \bar{x}_i) takes $r_i = \gamma_i - 3$ pebbles over $anc(x'_i)$ (resp. $anc(\bar{x}'_i)$) by Lemma 7.7, but there is a pebble on $\check{R}_{\bar{f}}$, a pebble on \check{R}_f , and a pebble on the other literal gadget $L_{r_i}(\bar{x}_i)$ (resp. $L_{r_i}(x_i)$); thus a transition cannot be done with at most $\gamma_i - 1$ pebbles over \check{R} . As such, there is a pebble on $anc(x_i)$ and a pebble on $anc(\bar{x}'_i)$ since time t_1 .

Because node f_i must be visited before node h_i can be pebbled, by Lemma 6.9, at some later time $t_2 > t_1$ there are $(\gamma_i - 6) + 2 = \gamma_i - 4$ pebbles on the turnpike from f'_i to f_i . At time t_2 over \check{R} , there are at least

$$\underbrace{\gamma_i - 4}_{\check{R}_f \setminus \{h_i\}} + \underbrace{1}_{\check{R}_{\bar{f}}} + \underbrace{2}_{anc(\{x_i, \bar{x}'_i\})} = \gamma_i - 1 \quad (7.5)$$

pebbles, giving Item (1).

At time t_2 , there is no pebble on $\{h_i\} \cup \check{R}_g$ where $\check{R}_g := anc(g_i) \setminus anc(\{x_i, \bar{x}'_i\})$, as all $\gamma_i - 1$ pebbles over \check{R} are on $\check{R}_f \setminus \{h_i\}$, $\check{R}_{\bar{f}}$ or $anc(\{x_i, \bar{x}'_i\})$ by (7.5). Over $\check{R} \setminus \check{R}_g$, there are at least

$$\underbrace{1}_{\check{R}_f} + \underbrace{1}_{\check{R}_{\bar{f}}} + \underbrace{2}_{anc(\{x_i, \bar{x}'_i\})} = 4$$

pebbles since time t_2 , so at most $(\gamma_i - 1) - 4 = \gamma_i - 5$ pebbles over \check{R}_g . Because node g_i must be visited before node h_i can be pebbled, by Lemma 6.7, at some later time $t_3 > t_2$ there are $(\gamma_i - 7) + 2 = \gamma_i - 5$ pebbles on the turnpike from q_{i-1} to g_i , including one on node q_{i-1} . Recall the region $\check{R}_1 = \check{R} \setminus (anc(x_i) \cup anc(\bar{x}'_i))$ that is associated with the $(i - 1)$ -assignment ρ_1 . At time t_3 , there is only one pebble over $\check{R}_1 \cap anc(q_{i-1})$,

which is on node q_{i-1} . The sub-strategy from time t_2 to t_3 persistently pebble node q_{i-1} over the region $\check{R}_1 \cap \text{anc}(q_{i-1})$ using $\gamma_i - 5 = \gamma_{i-1}$ pebbles, so $\text{RPeb}_{\check{R}_1}(G(\phi_{i-1})) \leq \gamma_{i-1}$. By the assumption that Lemma 7.3 holds for $i - 1$, we know $\phi_{i-1} \upharpoonright_{\rho_1}$ is TRUE.

We claim that at time t_2 , node \bar{h}_i has a pebble (by modifying the argument in the previous paragraph). Let $\check{R}_h := (\{h_i\} \cup \text{anc}(g_i)) \setminus \text{anc}(q_{i-1})$ be nodes properly in the turnpike from q_{i-1} to g_i plus h_i , and $\check{R}_{\bar{h}} := (\{\bar{h}_i\} \cup \text{anc}(\bar{g}_i)) \setminus \text{anc}(q_{i-1})$ be nodes properly in the turnpike from q_{i-1} to \bar{g}_i plus \bar{h}_i .

Claim 7.35 (Persistence). *At time t_2 , node \bar{h}_i has a pebble.*

Proof. For otherwise, at time t_2 , there is no pebble on \check{R}_h or $\check{R}_{\bar{h}}$, as all $\gamma_i - 1$ pebbles over \check{R} are on $\check{R}_f \setminus \{h_i\}$, $\check{R}_{\bar{f}} \setminus \{\bar{h}_i\}$ or $\text{anc}(\{x_i, \bar{x}'_i\})$ by (7.5). Let t_3 (resp. t_4) be the earliest time such that since time t_3 (resp. t_4) the region \check{R}_h (resp. $\check{R}_{\bar{h}}$) has pebble.

Claim 7.36 (No Double Persistence). *Since t_3 , there are at least two pebbles over $\check{R}_f \cup \check{R}_h$.*

Proof. Note that \check{R}_f consists of the turnpike from f'_i to f_i plus node h_i , and \check{R}_h consists of nodes properly in the turnpike from q_{i-1} to g_i plus node h_i .

Fix an induced subgraph $F \subseteq G$ (for instance $F =$ the turnpike from f'_i to f_i) having a unique sink. Say a pebbling configuration \mathbb{P} is v -locked on F if $\text{RPeb}_F(\mathbb{P}) = \text{RPeb}^V(F)$; this is if the pebbles cannot be cleared without using $\text{RPeb}^V(F)$ pebbles (when restricted to F). In particular, any configuration with a pebble on the sink of F is v -locked on F . Also, if a configuration is v -locked on F , then there is a pebble on F . Given a pebbling configuration on G , we say that f_i (resp. g_i) is v -locked if the configuration is v -locked on the turnpike from f'_i to f_i (resp. from q_{i-1} to g_i).

Assume for contradiction that at some time $t_7 \geq t_3$ there is only one pebble over $\check{R}_f \cup \check{R}_h$, which must be on h_i by the inclusion-exclusion principle. Let t_5 be the earliest time before t_7 such that there is a pebble on node h_i from t_5 to t_7 . We know $t_0 < t_1 < t_3 < t_5 < t_7$. At time t_5 node h_i is pebbled, so node g_i and node f_i each has a pebble, and both are v -locked. At time t_7 , nodes properly in the two turnpikes have no pebbles, and nodes g_i and f_i are not v -locked. Let t_6 be the earliest time after t_5 such that one of the turnpikes is not v -locked, then $t_5 < t_6 < t_7$.

- If the turnpike from f'_i to f_i stops being v -locked at time t_6 , then at time $t_6 - 1$ there are $(\gamma_i - 6) + 2 = \gamma_i - 4$ pebbles over the turnpike from f'_i to f_i by Lemma 6.9. Over \check{R} , there are

$$\underbrace{\gamma_i - 4}_{\text{turnpike from } f'_i \text{ to } f_i} + \underbrace{1}_{\text{turnpike from } q_{i-1} \text{ to } g_i} + \underbrace{1}_{h_i} + \underbrace{1}_{\check{R}_{\bar{f}}} + \underbrace{2}_{\text{anc}(\{x_i, \bar{x}'_i\})} = \gamma_i + 1$$

pebbles, contradicting that at most $\gamma_i - 1$ pebbles are used over \check{R} .

- If the turnpike from q_{i-1} to g_i stops being v -locked at time t_6 , then at time $t_6 - 1$ there are $(\gamma_i - 7) + 2 = \gamma_i - 5$ pebbles over the turnpike from q_{i-1} to g_i by Lemma 6.9. Over \check{R} , there are

$$\underbrace{\gamma_i - 5}_{\text{turnpike from } q_{i-1} \text{ to } g_i} + \underbrace{1}_{\text{turnpike from } f'_i \text{ to } f_i} + \underbrace{1}_{h_i} + \underbrace{1}_{\check{R}_{\bar{f}}} + \underbrace{2}_{\text{anc}(\{x_i, \bar{x}'_i\})} = \gamma_i$$

pebbles, contradicting that at most $\gamma_i - 1$ pebbles are used over \check{R} . \square

Assume $t_3 < t_4$ by symmetry (in the rest of Claim 7.35). At time $t_4 - 1$, there is no pebble on $\check{R}_{\bar{h}}$. By Lemma 6.9, at some later time $t_5 > t_4$ there are $(\gamma_i - 7) + 2 = \gamma_i + 5$ pebbles on the turnpike from q_{i-1} to \bar{g}_i . Over \check{R} , there are at least

$$\underbrace{\gamma_i - 5}_{\text{turnpike from } q_{i-1} \text{ to } \bar{g}_i} + \underbrace{2}_{\check{R}_f \cup \check{R}_h} + \underbrace{1}_{\check{R}_{\bar{f}}} + \underbrace{2}_{\text{anc}(\{x_i, \bar{x}'_i\})} = \gamma_i$$

pebbles, contradicting that at most $\gamma_i - 1$ pebbles are used over \check{R} . \square

Claim 7.35 shows that there is a pebble on \bar{h}_i at time t_2 , hence there is no pebble on the turnpike from \bar{f}'_i to \bar{f}_i by (7.5). Let t_3 be the earliest time before t_2 such that the only pebble on $\check{R}_{\bar{f}}$ is on \bar{h}_i , and let t_6 be the earliest time before t_3 such that \bar{h}_i has a pebble from time t_6 to t_3 . At time t_3 , there are pebbles on nodes x'_i and \bar{x}_i (so that node \bar{f}'_i can be unpebbled at time $t_3 - 1$), hence literal x_i is in FALSE position, and literal \bar{x}_i is not in * position. Note that there is no transition of literals x_i or \bar{x}_i between time t_6 and $t_3 - 1$: to make a transition for literal x_i (resp. \bar{x}_i) takes $r_i = \gamma_i - 3$ pebbles over $anc(x'_i)$ (resp. $anc(\bar{x}'_i)$) by Lemma 7.7, but there is a pebble on \bar{h}_i , a pebble on $\check{R}_{\bar{f}} \setminus \{\bar{h}_i\}$, and a pebble on the other literal gadget $L_{r_i}(\bar{x}_i)$ (resp. $L_{r_i}(x_i)$); thus a transition cannot be done with at most $\gamma_i - 1$ pebbles over \check{R} . As such, there is a pebble on $anc(x'_i)$ and a pebble on $anc(\bar{x}_i)$ between time t_6 and t_3 .

At time t_6 , node \bar{g}_i and node \bar{f}_i each has a pebble (so that node \bar{h}_i can be pebbled at time $t_6 - 1$). At time t_3 , the turnpike from \bar{f}'_i to \bar{f}_i has no pebble. By Lemma 6.9, there is a time t_4 between t_6 and t_3 such that there are $(\gamma_i - 6) + 2 = \gamma_i - 4$ pebbles on the turnpike from \bar{f}'_i to \bar{f}_i . We know $t_6 < t_4 < t_3 < t_2$. At time t_4 over \check{R} , there are at least

$$\underbrace{\gamma_i - 4}_{\check{R}_{\bar{f}} \setminus \{\bar{h}_i\}} + \underbrace{1}_{\bar{h}_i} + \underbrace{2}_{anc(\{x'_i, \bar{x}_i\})} = \gamma_i - 1 \quad (7.6)$$

pebbles.

At time t_4 , there is no pebble on $\check{R}_{\bar{g}}$ where $\check{R}_{\bar{g}} := anc(\bar{g}_i) \setminus anc(\{x'_i, \bar{x}_i\})$, as all $\gamma_i - 1$ pebbles over \check{R} are on $\check{R}_{\bar{f}}$ or $anc(\{x'_i, \bar{x}_i\})$ by (7.6). Over $\check{R} \setminus \check{R}_{\bar{g}}$, there are at least

$$\underbrace{1}_{\check{R}_{\bar{f}} \setminus \{\bar{h}_i\}} + \underbrace{1}_{\bar{h}_i} + \underbrace{2}_{anc(\{x'_i, \bar{x}_i\})} = 4$$

pebbles between time t_6 and $t_3 - 1$, so at most $(\gamma_i - 1) - 4 = \gamma_i - 5$ pebbles over $\check{R}_{\bar{g}}$. Because node \bar{g}_i is visited at time t_6 and the turnpike from q_{i-1} to \bar{g}_i has no pebble at time t_4 , by Lemma 6.7, at some time t_5 between t_6 and t_4 there are $(\gamma_i - 7) + 2 = \gamma_i - 5$ pebbles on the turnpike from q_{i-1} to \bar{g}_i , including one on node q_{i-1} . Recall the region $\check{R}_0 = \check{R} \setminus (anc(x'_i) \cup anc(\bar{x}_i))$ that is associated with the $(i - 1)$ -assignment ρ_0 . At time t_5 , there is only one pebble over $\check{R}_0 \cap anc(q_{i-1})$, which is on node q_{i-1} . The (reverse of the) substrategy from time t_5 to t_4 persistently pebble node q_{i-1} over the region $\check{R}_0 \cap anc(q_{i-1})$ using $\gamma_i - 5 = \gamma_{i-1}$ pebbles, so $RPeb_{\check{R}_0}(G(\phi_{i-1})) \leq \gamma_{i-1}$. By the assumption that Lemma 7.3 holds for $i - 1$, we know $\phi_{i-1} \upharpoonright_{\rho_0}$ IS TRUE.

As a result, $\forall x_i \phi_{i-1} \upharpoonright_{\rho} = \phi_i \upharpoonright_{\rho}$ IS TRUE, giving Item (2). \square

Lemma 7.37 (Universal Quantifier Gadget). *Assume that Lemma 7.3 holds for $i - 1$. We have $RPeb_{\check{R}}(G(\phi_i)) = \gamma_i + \llbracket \phi_i \upharpoonright_{\rho} \text{ IS FALSE} \rrbracket$.*

Proof. Since persistent price is one plus surrounding price (Proposition 2.1), it suffices to show that $RPeb_{\check{R}}^S(G(\phi_i)) = \gamma_i - 1 + \llbracket \phi_i \upharpoonright_{\rho} \text{ IS FALSE} \rrbracket$. If $\phi_i \upharpoonright_{\rho}$ IS TRUE, then $RPeb_{\check{R}}^S(G(\phi_i)) = \gamma_i - 1$, as the upper bound (Lemma 7.33) matches the lower bound (Lemma 7.34). If $\phi_i \upharpoonright_{\rho}$ IS FALSE, then $RPeb_{\check{R}}(G(\phi_i)) = \gamma_i$, as the upper bound (Lemma 7.33) matches the lower bound (Lemma 7.34). \square

8 Product Construction for Reversible Pebbling

The part of the proof of Theorem 3.4 that deals with reversible pebbling uses as a black box the construction in Theorem 3.5 for reversible pebbling. Now we state it again and we give its full proof.

Theorem 8.1. *Given two graphs G_1 and G_2 , there is a polynomial-time constructible graph $\mathcal{R}(G_1, G_2)$ of size $3|G_1| \cdot |G_2|$ with reversible pebbling price $RPeb(\mathcal{R}(G_1, G_2)) = RPeb(G_1) + RPeb(G_2) + 1$.*

We want to construct a graph \mathcal{R} to inherit structures from two graphs, which are called respectively the *exterior graph* G_1 and the *interior graph* G_2 . Intuitively, for every node in G_1 , we will construct a *block* with the structure of G_2 : in each such block, for every node in G_2 we create a *cell* of three nodes, where different cells are connected according to the exterior graph G_1 and the interior graph G_2 .

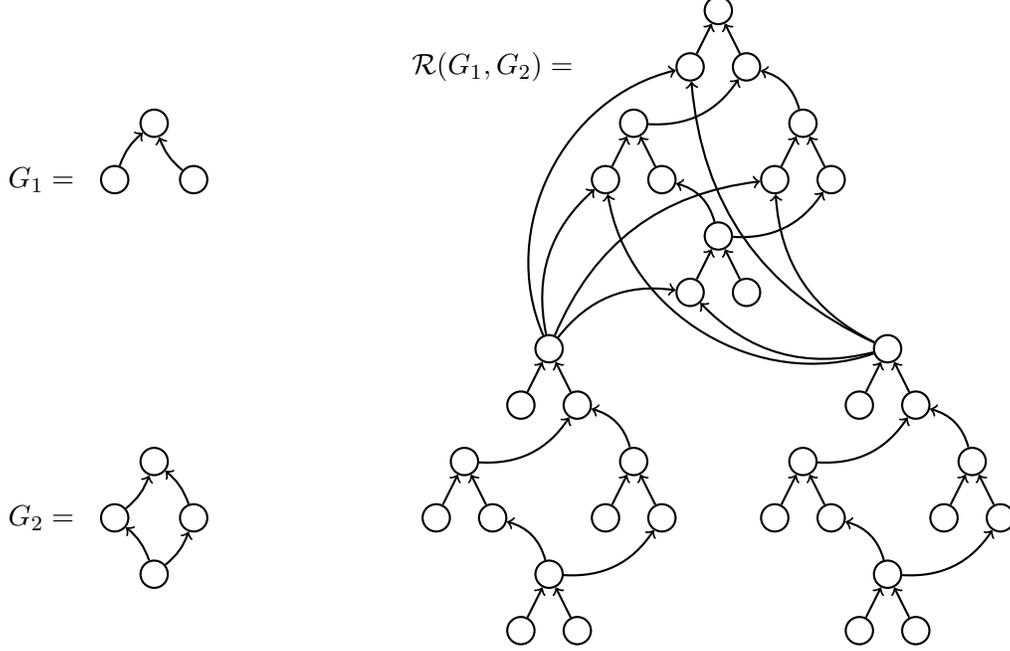


Figure 19: Example of Construction 8.2: product of a pyramid of height 1 and a rhombus.

Construction 8.2 (Product for reversible pebbling). Fix two graphs G_1 and G_2 , and denote z_2 as the unique sink of G_2 . Construct a graph $\mathcal{R} := \mathcal{R}(G_1, G_2)$ as follows. For every node $(v_1, v_2) \in V(G_1) \times V(G_2)$, create three nodes $(v_1, v_2)_{out}, (v_1, v_2)_{ext}, (v_1, v_2)_{int}$. Add an edge from the exterior node to the output node, i.e., from node $(v_1, v_2)_{ext}$ to $(v_1, v_2)_{out}$; add an edge from the interior node to the output node, i.e., from node $(v_1, v_2)_{int}$ to $(v_1, v_2)_{out}$. The exterior node supports the structure of the exterior graph G_1 , in the sense that for every predecessor w_1 of v_1 in G_1 , we create an edge from the sink the w_1 -block of G_2 , i.e., from node $(w_1, z_2)_{out}$ to node $(v_1, v_2)_{ext}$. The interior node supports the structure of the interior graph G_2 , in the sense that for every predecessor w_2 of v_2 in G_2 , we create an edge from the output node of w_2 , i.e., from node $(v_1, w_2)_{out}$ to node $(v_1, v_2)_{int}$.

Formally, $V(\mathcal{R}) := \{(v_1, v_2)_{out}, (v_1, v_2)_{ext}, (v_1, v_2)_{int} : v_1 \in V(G_1), v_2 \in V(G_2)\}$, and $E(\mathcal{R}) := E_{out} \cup E_{ext} \cup E_{int}$, where $E_{out} := \{((v_1, v_2)_{ext}, (v_1, v_2)_{out}), ((v_1, v_2)_{int}, (v_1, v_2)_{out}) : v_1 \in V(G_1), v_2 \in V(G_2)\}$, $E_{ext} := \{((w_1, z_2)_{out}, (v_1, v_2)_{ext}) : v_1 \in V(G_1), v_2 \in V(G_2), w_1 \in pred_{G_1}(v_1)\}$, and $E_{int} := \{((v_1, w_2)_{out}, (v_1, v_2)_{int}) : v_1 \in V(G_1), v_2 \in V(G_2), w_2 \in pred_{G_2}(v_2)\}$.

Clearly, if G_1 and G_2 each has in-degree at most two and a unique sink, then so does the resulting graph \mathcal{R} . Note that the graph \mathcal{R} partitions into $|V(G_1)|$ blocks, namely, for each $v_1 \in V(G_1)$, the v_1 -block is the subgraph of \mathcal{R} induced over the node set $\{(v_1, v_2)_{out}, (v_1, v_2)_{ext}, (v_1, v_2)_{int} : v_2 \in V(G_2)\}$. Each such

block further partitions into $|V(G_2)|$ cells, namely, in the v_1 -block, for each $v_2 \in V(G_2)$, the (v_1, v_2) -cell is the subgraph of the v_1 -block induced over the node set $\{(v_1, v_2)_{\text{out}}, (v_1, v_2)_{\text{ext}}, (v_1, v_2)_{\text{int}}\}$.

Finally, given a configuration \mathbb{P}' on \mathcal{R} , say the v_1 -block (resp. the (v_1, v_2) -cell) is *surrounded* if any exterior node of the v_1 -block (resp. the interior node of the (v_1, v_2) -cell) is surrounded in \mathbb{P}' . Note that the v_1 -block is surrounded iff *every* exterior node of the v_1 -block is surrounded.

Lemma 8.3 (Upper Bound). $RPeb(\mathcal{R}(G_1, G_2)) \leq RPeb(G_1) + RPeb(G_2) + 1$.

Proof. Fix a persistent pebbling \mathcal{P}_1 of G_1 using $RPeb(G_1)$ pebbles, and a persistent pebbling \mathcal{P}_2 of G_2 using $RPeb(G_2)$ pebbles. We will construct a persistent pebbling \mathcal{P}' of $\mathcal{R}(G_1, G_2)$ using $RPeb(G_1) + RPeb(G_2) + 1$ pebbles.

We claim that the persistent price of each block of \mathcal{R} is at most $RPeb(G_2) + 2$. For any $v_1 \in V(G_1)$, to persistently pebble the v_1 -block (assuming the v_1 -block is surrounded), simulate \mathcal{P}_2 as follows: whenever \mathcal{P}_2 pebbles a node $v_2 \in V(G_2)$, the simulating pebbling has a phase to persistently pebble the (v_1, v_2) -cell, and whenever \mathcal{P}_2 unpebbles a node $v_2 \in V(G_2)$, then the simulating pebbling has a phase to persistently unpebble the (v_1, v_2) -cell. If the current configuration in \mathcal{P}_2 is \mathbb{P} , and the configuration in the simulating pebbling at the end of a phase is \mathbb{P}' , then the simulating pebbling maintains the phase-invariant that $\mathbb{P}' = \{(v_1, v_2)_{\text{out}} : v_2 \in \mathbb{P}\}$. Note that the simulating pebbling is legal: since the pebbling \mathcal{P}_2 is legal, when v_2 is pebbled or unpebbled it is surrounded in the current configuration \mathbb{P} , so the (v_1, v_2) -cell is surrounded in the simulating configuration \mathbb{P}' and the interior node can be pebbled or unpebbled; and we assume that the v_1 -block is surrounded, so the exterior node can be pebbled or unpebbled. The simulating pebbling uses at most two more pebbles (on the exterior node and the interior node of each cell), for at most $RPeb(G_2) + 2$ pebbles over the v_1 -block.

Then the resulting graph $\mathcal{R}(G_1, G_2)$ can be persistently pebbled by simulating \mathcal{P}_1 as follows. Whenever \mathcal{P}_1 pebbles a node $v_1 \in V(G_1)$, the simulating pebbling \mathcal{P}' has a stage to persistently pebble the v_1 -block; whenever \mathcal{P}_1 unpebbles a node $v_1 \in V(G_1)$, the simulating pebbling \mathcal{P}' has a stage to persistently unpebble the v_1 -block. If the current configuration in \mathcal{P}_1 is \mathbb{P} , and the configuration in the simulating pebbling \mathcal{P}' at the end of a stage is \mathbb{P}' , then the simulating pebbling maintains the stage-invariant that $\mathbb{P}' = \{(v_1, z_2)_{\text{out}} : v_1 \in \mathbb{P}\}$. Note that the simulating pebbling \mathcal{P}' is legal: since the pebbling \mathcal{P}_1 is legal, when v_1 is pebble or unpebbled it is surrounded in the current configuration \mathbb{P} , so the v_1 -block is surrounded in the simulating configuration \mathbb{P}' , so the v_1 -block can be persistently pebbled or unpebbled in a stage of \mathcal{P}' . When the v_1 -block is pebbled or unpebbled in a stage of \mathcal{P}' , there are at most $RPeb(G_1) - 1$ pebbles on other blocks of \mathcal{R} , and there are at most $RPeb(G_2) + 2$ pebbles in the v_1 -block, for a total of $RPeb(G_1) + RPeb(G_2) + 1$ pebbles. \square

To prove the lower bound we extract simultaneous peblings of G_1 and G_2 from any pebbling of \mathcal{R} and use the known pebbling prices of G_1 and G_2 to argue that some configuration needs many pebbles. We do so by projecting a pebbling of \mathcal{R} into G_1 and G_2 as the skeleton of a pebbling and then filling the gaps between configurations with a legal sequence of pebbling moves.

Lemma 8.4 (Lower Bound). $RPeb(\mathcal{R}(G_1, G_2)) \geq RPeb(G_1) + RPeb(G_2) + 1$.

Proof. Fix any persistent pebbling $\mathcal{P}' = (\mathbb{P}'_0, \mathbb{P}'_1, \dots, \mathbb{P}'_\tau)$ of $\mathcal{R}(G_1, G_2)$. For every $v_1 \in V(G_1)$, we are going to simulate a pebbling \mathcal{P}^{v_1} on G_2 based (essentially) on the configurations of \mathcal{P}' over the v_1 -block. From the family of peblings $\{\mathcal{P}^{v_1}\}_{v_1 \in V(G_1)}$, we then simulate a persistent pebbling \mathcal{P} on G_1 .

In more detail, for each $v_1 \in V(G_1)$ we define a mapping $Int^{v_1} : \mathcal{R} \rightarrow G_2$ and we view the sequence of configurations $(Int^{v_1}(\mathbb{P}'_t))_{t \in [0, \tau]}$ as the skeleton of a pebbling of G_2 . We fill the gaps between configurations according to the algorithm described below to obtain a legal pebbling \mathcal{P}^{v_1} . Similarly we define a mapping $Ext : \mathcal{R} \rightarrow G_1$ to construct the pebbling \mathcal{P} on G_1 .

To describe the mappings we need some definitions. Given a configuration \mathbb{P}' in \mathcal{P}' of $\mathcal{R}(G_1, G_2)$, its projection to the output (resp. exterior, interior) nodes of the v_1 -block is $proj_{\text{out}}^{v_1}(\mathbb{P}') := \{v_2 \in V(G_2) : (v_1, v_2)_{\text{out}} \in \mathbb{P}'\}$ (resp. $proj_{\text{ext}}^{v_1}(\mathbb{P}') := \{v_2 \in V(G_2) : (v_1, v_2)_{\text{ext}} \in \mathbb{P}'\}$, $proj_{\text{int}}^{v_1}(\mathbb{P}') := \{v_2 \in V(G_2) : (v_1, v_2)_{\text{int}} \in \mathbb{P}'\}$).

The closure $clos(\mathbb{P}') \subseteq V(\mathcal{R}(G_1, G_2))$ is the smallest set of nodes containing \mathbb{P}' that is closed under pebble placements on interior or output nodes; equivalently, $clos(\mathbb{P}')$ can be generated by the following algorithm: Start with \mathbb{P}' , while there is a node $v \in V(\mathcal{R}(G_1, G_2))$ which is an interior node $v = (v_1, v_2)_{\text{int}}$ or an output node $v = (v_1, v_2)_{\text{out}}$ such that v is surrounded by, but not in, the subset of nodes having pebbles, pebble v . Note that the closure of a v_1 -block does not depend on other blocks as they are connected only through exterior nodes.

For brevity, given a graph G and a subset $U \subseteq V(G)$ of vertices, denote $unsur_G(U) := \{v \in V(G) : pred_G(v) \not\subseteq U\}$ as the subset of nodes in G not surrounded by U .

The block mapping is $I^{w_1}(\mathbb{P}') := proj_{\text{out}}^{w_1}(\mathbb{P}') \cup (proj_{\text{int}}^{w_1}(\mathbb{P}') \cap unsur_{G_2}(proj_{\text{out}}^{w_1}(\mathbb{P}')))$

We define the interior mapping $Int^{w_1}(\mathbb{P}')$ to be $I^{w_1}(\mathbb{P}')$ if the w_1 -block is surrounded, and $I^{w_1}(clos(\mathbb{P}'))$ if the w_1 -block is not surrounded.

Given a configuration \mathbb{P}' , its p-projection is $proj^P(\mathbb{P}') := \{v_1 \in V(G_1) : Int^{v_1}(\mathbb{P}') \text{ is p-locked}\}$, and v-projection is $proj^V(t) := \{v_1 \in V(G_1) : Int^{v_1}(\mathbb{P}') \text{ is v-locked}\}$.

Finally the exterior mapping is $Ext(\mathbb{P}') := proj^P(\mathbb{P}') \cup (proj^V(\mathbb{P}') \cap unsur_{G_1}(proj^P(\mathbb{P}')))$.

We abuse the notation for mappings from configurations in \mathcal{P}' and write $f(t)$ to mean $f(\mathbb{P}'_t)$. In addition we define $\mathbb{P}^{v_1}(t) = Int^{v_1}(\mathbb{P}'_t)$ and $\mathbb{P}(t) = Ext(\mathbb{P}'_t)$. Note that there can be multiple pebble moves between, say, $\mathbb{P}(t-1)$ and $\mathbb{P}(t)$.

We construct the pebbings $\{\mathcal{P}^{v_1}\}_{v_1 \in V(G_1)}$ and \mathcal{P} according to Algorithm 8.8, which are legal by Claim 8.9. Note that \mathcal{P} is a persistent pebbling of G_1 : $\mathbb{P}^{w_1}(\tau) = I^{w_1}(clos(\tau))$, which is \emptyset if w_1 is not the sink of G_1 , and $\{z_2\}$ if w_1 is the sink of G_1 , so $\mathbb{P}(\tau) = Ext(\tau) = \{z_1\}$, the sink of G_1 . Then the lower bound follows from Claim 8.13. \square

The algorithm to construct the remaining configurations in the pebbings $\{\mathcal{P}^{v_1}\}_{v_1 \in V(G_1)}$ and \mathcal{P} , given \mathcal{P}' , is essentially to insert and remove missing pebbles in topological order whenever two configuration are different, and make such a pebbling go through a specific configuration in the exterior case. We give an explicit description below and in Claim 8.9 we prove that it is equivalent to this implicit description.

Definition 8.5 (Reasonable pebbling). Given a configuration \mathbb{P} , a set of vertices to pebble T_+ and a set of vertices to unpebble T_- , the *reasonable pebbling* is the following pebbling:

- start with \mathbb{P} ;
- for each $v \in T_+$ in a topological order, pebble v ;
- for each $v \in T_-$ in a reverse topological order, unpebble v .

Furthermore, the reasonable pebbling between two configurations $\mathbb{P}_1, \mathbb{P}_2$ is the reasonable pebbling with $\mathbb{P} = \mathbb{P}_1$, $T_+ = \mathbb{P}_2 \setminus \mathbb{P}_1$, and $T_- = \mathbb{P}_1 \setminus \mathbb{P}_2$.

Claim 8.6 (Legality). Assume $\mathbb{P}, T_+, T_- \subseteq V(G)$ are given. If for every $v \in T_+ \cup T_-$, we have $pred_G(v) \subseteq \mathbb{P} \cup T_+$, then the reasonable pebbling is legal.

Proof. For any $v \in T_+$, right before v is pebbled, we know $pred_G(v)$ have pebbles since $pred_G(v) \subseteq \mathbb{P} \cup T_+$ and the pebble placement on T_+ proceeds in a topological order. So all pebble placements on T_+ are legal.

For any $v \in T_-$, right before v is unpebbled, we know $pred_G(v)$ have pebbles since $pred_G(v) \subseteq \mathbb{P} \cup T_+$ and the pebble placement on T_- proceeds in a reverse topological order. So all pebble removals on T_- are legal. \square

Corollary 8.7 (Legality). *Assume $\mathbb{P}_1, \mathbb{P}_2 \subseteq V(G)$ are given. Assume the sets $T_+ := \mathbb{P}_2 \setminus \mathbb{P}_1$ and $T_- := \mathbb{P}_1 \setminus \mathbb{P}_2$ satisfy that for every $v \in T_+ \cup T_- = \mathbb{P}_1 \triangle \mathbb{P}_2$, we have $\text{pred}_G(v) \subseteq \mathbb{P}_1 \cup T_+ = \mathbb{P}_1 \cup \mathbb{P}_2$, then the reasonable pebbling over \mathbb{P}_1, T_+, T_- is legal.*

Proof. Apply Claim 8.6 on \mathbb{P}_1, T_+, T_- . □

Algorithm 8.8. *As Claim 8.9 shows, we only need to consider the case when an output node $(v_1, v_2)_{\text{out}}$ is pebbled or unpebbled to get to \mathbb{P}'_t in \mathcal{P}' . In this case we run the following steps in sequence:*

- (a) *if the v_1 -block is surrounded in \mathbb{P}'_t , we insert the reasonable pebbling between $I^{v_1}(t-1)$ and $I^{v_1}(t)$ into \mathcal{P}^{v_1} .*
- (b) *if $v_2 = z_2$,*
 - (i) *if $(v_1, v_2)_{\text{out}}$ is pebbled to get to \mathbb{P}'_t in \mathcal{P}' , for each successor w_1 of v_1 such that the w_1 -block is surrounded in \mathbb{P}'_t , we insert the reasonable pebbling between $I^{w_1}(\text{clos}(t-1))$ and $I^{w_1}(t)$ into \mathcal{P}^{w_1} .*
 - (ii) *if $(v_1, v_2)_{\text{out}}$ is unpebbled to get to \mathbb{P}'_t in \mathcal{P}' , for each successor w_1 of v_1 such that the w_1 -block is surrounded in \mathbb{P}'_{t-1} , we insert the reasonable pebbling between $I^{w_1}(t-1)$ and $I^{w_1}(\text{clos}(t))$ into \mathcal{P}^{w_1} .*
- (c) *if $\text{Ext}(t-1) \neq \text{Ext}(t)$, let $D := \text{proj}^V(t-1) \cup \text{proj}^V(t)$. We insert the reasonable pebbling with $T_+ := D \setminus \text{Ext}(t-1)$, and $T_- := D \setminus \text{Ext}(t)$ into \mathcal{P} .*

Claim 8.9 (Correctness). *The pebbblings $\{\mathcal{P}^{v_1}\}_{v_1 \in V(G_1)}$ and \mathcal{P} are legal.*

Proof. At the beginning $t = 0$, we know $\mathbb{P}'_0 = \emptyset$, so for any $w_1 \in V(G_1)$ we have $I^{w_1}(0) = \emptyset$ and $I^{w_1}(\text{clos}(0)) = \emptyset$, matching $\mathbb{P}^{w_1}(0) = \emptyset$. Also $\text{proj}^P(0) = \emptyset = \text{proj}^V(0)$, and $\mathbb{P}(0) = \emptyset = \text{Ext}(0)$.

For any two consecutive configurations we show that either they are the same or the algorithm inserted a reasonable pebbling between them, which is legal by Claim 8.11.

When $t > 0$, if an exterior node or an interior node is pebbled or unpebbled to get to \mathbb{P}'_t in \mathcal{P}' , fix any $w_1 \in V(G_1)$. Note that the w_1 -block is surrounded in \mathbb{P}'_{t-1} iff it is surrounded in \mathbb{P}'_t .

If the w_1 -block is surrounded, then $\text{proj}_{\text{out}}^{w_1}(\cdot)$ is the same at \mathbb{P}'_{t-1} and \mathbb{P}'_t , and $\text{proj}_{\text{int}}^{w_1}(\cdot) \cap \text{unsur}_{G_2}(\text{proj}_{\text{out}}^{w_1}(\cdot))$ is also the same at \mathbb{P}'_{t-1} and \mathbb{P}'_t because a node being pebbled or unpebbled must be surrounded, and the predecessors of an interior node are output nodes. Hence $\mathbb{P}^{w_1}(t-1) = I^{w_1}(t-1) = I^{w_1}(t) = \mathbb{P}^{w_1}(t)$.

Otherwise the w_1 -block is not surrounded, then the pebble move is not on any exterior node in the w_1 -block, hence $\text{clos}(\cdot)$ is the same at \mathbb{P}'_{t-1} and \mathbb{P}'_t over the w_1 -block, so $I^{w_1}(\text{clos}(\cdot))$ is also the same at \mathbb{P}'_{t-1} and \mathbb{P}'_t .

Since $\text{Ext}(\cdot)$ only depends on $\{\mathbb{P}^{v_1}(t)\}_{v_1 \in V(G_1)}$, it holds that $\mathbb{P}_{t-1} = \mathbb{P}_t$ as well.

Focus on the case when an output node $(v_1, v_2)_{\text{out}}$ is pebbled or unpebbled to get to \mathbb{P}'_t in \mathcal{P}' , fix any $w_1 \in V(G_1)$.

- If $w_1 \neq v_1$ and either w_1 is not a successor of v_1 or $v_2 \neq z_2$, note that the w_1 -block is surrounded in \mathbb{P}'_{t-1} iff it is surrounded in \mathbb{P}'_t . Since $\mathbb{P}'_{t-1} = \mathbb{P}'_t$ over the w_1 -block, we have $I^{w_1}(t-1) = I^{w_1}(t)$ and $I^{w_1}(\text{clos}(t-1)) = I^{w_1}(\text{clos}(t))$.
- If $w_1 = v_1$, then the w_1 -block is surrounded in \mathbb{P}'_{t-1} iff it is surrounded in \mathbb{P}'_t . If the w_1 -block is not surrounded, since the pebble move is not on any exterior node in the w_1 -block, we have $\text{clos}(\cdot)$ is the same at \mathbb{P}'_{t-1} and \mathbb{P}'_t over the w_1 -block, so $I^{w_1}(\text{clos}(\cdot))$ is also the same at \mathbb{P}'_{t-1} and \mathbb{P}'_t .

Otherwise the w_1 -block is surrounded and Step (a) inserts a reasonable pebbling into \mathcal{P}^{w_1} .

- If w_1 is a successor of v_1 and $v_2 = z_2$ the w_1 -block cannot be surrounded in both \mathbb{P}'_{t-1} and \mathbb{P}'_t .
 If the w_1 -block is unsurrounded in both \mathbb{P}'_{t-1} and \mathbb{P}'_t , since $\mathbb{P}'_{t-1} = \mathbb{P}'_t$ over the w_1 -block, we have $I^{w_1}(\text{clos}(t-1)) = I^{w_1}(\text{clos}(t))$.
 Otherwise w_1 is surrounded in exactly one of \mathbb{P}'_{t-1} and \mathbb{P}'_t and Step (b) inserts a reasonable pebbling into \mathcal{P}^{w_1} .

Finally, Step (c) inserts a reasonable pebbling into \mathcal{P} as needed. \square

Claim 8.10 (Exterior Symmetry). *In Step (c), we have $\text{Ext}(t-1) \cup T_+ = D = \text{Ext}(t) \cup T_-$.*

Proof. Since $\text{proj}^P(t) \subseteq \text{proj}^V(t)$, we have $\text{Ext}(t) = \text{proj}^P(t) \cup (\text{proj}^V(t) \cap \text{unsur}_{G_1}(\text{proj}^P(t))) \subseteq \text{proj}^V(t) \subseteq D$ and likewise $\text{Ext}(t-1) \subseteq \text{proj}^V(t-1) \subseteq D$. \square

Claim 8.11 (Legality). *All reasonable pebblings are legal.*

Proof. At Step (a) an output node $(v_1, v_2)_{\text{out}}$ is pebbled or unpebbled to get to \mathbb{P}'_t in \mathcal{P}' and the v_1 -block is surrounded, hence $\mathbb{P}^{v_1}(t-1) = I^{v_1}(t-1)$. By Corollary 8.7 it suffices to show that for $w_2 \in I^{v_1}(t) \triangle I^{v_1}(t-1)$, we have $\text{pred}_{G_2}(w_2) \subseteq I^{v_1}(t) \cup I^{v_1}(t-1)$. Recall $I^{v_1}(\cdot) = \text{proj}_{\text{out}}^{v_1}(\cdot) \cup (\text{proj}_{\text{int}}^{v_1}(\cdot) \cap \text{unsur}_{G_2}(\text{proj}_{\text{out}}^{v_1}(\cdot)))$.

Assume $w_2 \in I^{v_1}(t) \setminus I^{v_1}(t-1)$, reversing the roles of $t-1$ and t otherwise. If $w_2 \in \text{proj}_{\text{out}}^{v_1}(t) \setminus I^{v_1}(t-1)$, then $w_2 \in \text{proj}_{\text{out}}^{v_1}(t) \setminus \text{proj}_{\text{out}}^{v_1}(t-1)$, i.e., $w_2 = v_2$ and $(v_1, w_2)_{\text{out}}$ is being pebbled to get to \mathbb{P}'_t in \mathcal{P}' . Since \mathcal{P}' is legal, $(v_1, w_2)_{\text{int}}$ has a pebble in \mathbb{P}'_{t-1} , i.e., $w_2 \in \text{proj}_{\text{int}}^{v_1}(t-1)$, so w_2 is surrounded by $\text{proj}_{\text{out}}^{v_1}(t-1) \subseteq I^{v_1}(t-1)$ as needed.

Otherwise $w_2 \in (\text{proj}_{\text{int}}^{v_1}(t) \cap \text{unsur}_{G_2}(\text{proj}_{\text{out}}^{v_1}(t))) \setminus I^{v_1}(t-1)$, since $\text{proj}_{\text{int}}^{v_1}(t) = \text{proj}_{\text{int}}^{v_1}(t-1)$, we know $w_2 \in \text{unsur}_{G_2}(\text{proj}_{\text{out}}^{v_1}(t))$ but $w_2 \notin \text{unsur}_{G_2}(\text{proj}_{\text{out}}^{v_1}(t-1))$, so w_2 is a successor of v_2 and $(v_1, v_2)_{\text{out}}$ is being unpebbled to get to t in \mathcal{P}' and $\text{pred}_{G_2}(w_2) \subseteq \text{proj}_{\text{out}}^{v_1}(t-1) \subseteq I^{v_1}(t-1)$ as needed.

At Step (b)(i) an output node $(v_1, v_2)_{\text{out}}$ is pebbled to get to \mathbb{P}'_t in \mathcal{P}' , and w_1 is a successor of v_1 such that the w_1 -block is surrounded in \mathbb{P}'_t . Note that the w_1 -block is not surrounded in \mathbb{P}'_{t-1} , so $\mathbb{P}^{w_1}(t-1) = I^{w_1}(\text{clos}(t-1))$. Since the pebble move to get to \mathbb{P}'_t in \mathcal{P}' is not in the w_1 -block, we have $I^{w_1}(\text{clos}(t-1)) = I^{w_1}(\text{clos}(t))$. By Corollary 8.7 it suffices to show that for $w_2 \in I^{w_1}(t) \triangle I^{w_1}(\text{clos}(t))$, we have $\text{pred}_{G_2}(w_2) \subseteq I^{w_1}(t) \cup I^{w_1}(\text{clos}(t))$.

- Assume $w_2 \in I^{w_1}(t) \setminus I^{w_1}(\text{clos}(t))$. Since $\text{proj}_{\text{out}}^{w_1}(t) \subseteq \text{proj}_{\text{out}}^{w_1}(\text{clos}(t)) \subseteq I^{w_1}(\text{clos}(t))$, we can assume $w_2 \in (\text{proj}_{\text{int}}^{w_1}(t) \cap \text{unsur}_{G_2}(\text{proj}_{\text{out}}^{w_1}(t))) \setminus I^{w_1}(\text{clos}(t))$. Since $\text{proj}_{\text{int}}^{w_1}(t) \subseteq \text{proj}_{\text{int}}^{w_1}(\text{clos}(t))$, we know w_2 is surrounded by $\text{proj}_{\text{out}}^{w_1}(\text{clos}(t)) \subseteq I^{w_1}(\text{clos}(t))$ as needed.
- Assume $w_2 \in I^{w_1}(\text{clos}(t)) \setminus I^{w_1}(t)$. We claim that $w_2 \in \text{proj}_{\text{int}}^{w_1}(\text{clos}(t))$: if $w_2 \in (\text{proj}_{\text{int}}^{w_1}(\text{clos}(t)) \cap \text{unsur}_{G_2}(\text{proj}_{\text{out}}^{w_1}(\text{clos}(t)))) \setminus I^{w_1}(t)$, then we are done; otherwise $w_2 \in \text{proj}_{\text{out}}^{w_1}(\text{clos}(t)) \setminus I^{w_1}(t)$, then $w_2 \notin \text{proj}_{\text{out}}^{w_1}(t)$ and thus by definition of $\text{clos}(\cdot)$ we have $w_2 \in \text{proj}_{\text{int}}^{w_1}(\text{clos}(t))$ as claimed. Now if $w_2 \in \text{proj}_{\text{int}}^{w_1}(t)$, then w_2 is surrounded by $\text{proj}_{\text{out}}^{w_1}(t) \subseteq I^{w_1}(t)$ as needed; otherwise $w_2 \notin \text{proj}_{\text{int}}^{w_1}(t)$, then from $w_2 \in \text{proj}_{\text{int}}^{w_1}(\text{clos}(t))$ and by definition of $\text{clos}(\cdot)$ we have w_2 is surrounded by $\text{proj}_{\text{out}}^{w_1}(\text{clos}(t)) \subseteq I^{w_1}(\text{clos}(t))$ as needed.

To see that Step (b)(ii) is legal, note that it is the reverse of Step (b)(i).

At Step (c) an output node $(v_1, v_2)_{\text{out}}$ is pebbled or unpebbled to get to \mathbb{P}'_t in \mathcal{P}' . By Claim 8.6 it suffices to show that for $w_1 \in (D \setminus \text{Ext}(t-1)) \cup (D \setminus \text{Ext}(t))$, we have $\text{pred}_{G_1}(w_1) \subseteq \text{Ext}(t-1) \cup T_+ = D$, where

the last equality is due to Claim 8.10. Recall that $Ext(\cdot) = proj^P(\cdot) \cup (proj^V(\cdot) \cap unsur_{G_1}(proj^P(\cdot)))$ and $D = proj^V(t-1) \cup proj^V(t)$.

Fix any $w_1 \in (D \setminus Ext(t-1)) \cup (D \setminus Ext(t))$. Swap the roles of t and $t-1$ if necessary, we can assume $w_1 \in D \setminus Ext(t)$. If $w_1 \in proj^V(t) \setminus Ext(t)$, then w_1 is surrounded by $proj^P(t) \subseteq Ext(t) \subseteq D$ as needed.

Otherwise $w_1 \in proj^V(t-1) \setminus proj^V(t)$, in particular $\mathbb{P}^{w_1}(t-1) \neq \mathbb{P}^{w_1}(t)$, so $w_1 \in \{v_1\} \cup succ_{G_1}(v_1)$ by design of the algorithm. Note that it suffices to show that the w_1 -block is surrounded in \mathbb{P}'_{t-1} or in \mathbb{P}'_t : if the w_1 -block is surrounded in \mathbb{P}' , for any $u_1 \in pred_{G_1}(w_1)$ the sink z_2 of the inner graph G_2 satisfies $z_2 \in proj^{u_1}_{out}(\mathbb{P}') \subseteq I^{u_1}(\mathbb{P}') \subseteq Int^{u_1}(\mathbb{P}')$, so $u_1 \in proj^V(\mathbb{P}') \subseteq D$ as needed. If $w_1 = v_1$, then in Step (a) the w_1 -block is surrounded in \mathbb{P}'_t as needed. Otherwise $w_1 \in succ_{G_1}(v_1)$, then in Step (b) the w_1 -block is surrounded in \mathbb{P}'_t (in Step (b)(i)) or in \mathbb{P}'_{t-1} (in Step (b)(ii)) as needed. \square

Claim 8.12 (No Spurious Projections). *For any $w_1 \in V(G_1)$ and $w_2 \in V(G_2)$, if w_2 has a pebble in any configuration of \mathcal{P}^{w_1} created between time $t-1$ and time t , i.e., between $\mathbb{P}^{w_1}(t-1)$ and $\mathbb{P}^{w_1}(t)$, then the (w_1, w_2) -cell has a pebble both in \mathbb{P}'_{t-1} and in \mathbb{P}'_t in \mathcal{P}' .*

Proof. Let us reword the claim. Define $proj^{w_1}_{any}(\mathbb{P}') := proj^{w_1}_{out}(\mathbb{P}') \cup proj^{w_1}_{ext}(\mathbb{P}') \cup proj^{w_1}_{int}(\mathbb{P}')$, note that $proj^{w_1}_{any}(t-1) = proj^{w_1}_{any}(t)$, and we want to show $\mathbb{P}^{w_1}(t-1) \cup T_+^{w_1} \subseteq proj^{w_1}_{any}(t)$.

Note that for any \mathbb{P}' and $w_1 \in V(G_1)$ we have $I^{w_1}(\mathbb{P}') \subseteq proj^{w_1}_{any}(\mathbb{P}')$. Moreover, we have $I^{w_1}(clos(\mathbb{P}')) \subseteq proj^{w_1}_{any}(\mathbb{P}')$: for any $w_2 \in I^{w_1}(clos(\mathbb{P}'))$, if $w_2 \in proj^{w_1}_{out}(clos(\mathbb{P}'))$, then either $w_2 \in proj^{w_1}_{out}(\mathbb{P}') \subseteq proj^{w_1}_{any}(\mathbb{P}')$ as needed, or $w_2 \notin proj^{w_1}_{out}(\mathbb{P}')$, then by definition of $clos(\cdot)$ we have $w_2 \in proj^{w_1}_{ext}(\mathbb{P}') \subseteq proj^{w_1}_{any}(\mathbb{P}')$ as needed. Otherwise $w_2 \in proj^{w_1}_{int}(clos(\mathbb{P}')) \cap unsur_{G_2}(proj^{w_1}_{out}(clos(\mathbb{P}')))$, then it follows that $w_2 \in proj^{w_1}_{int}(\mathbb{P}') \subseteq proj^{w_1}_{any}(\mathbb{P}')$ as needed; for otherwise $w_2 \notin proj^{w_1}_{int}(\mathbb{P}')$, thus by definition of $clos(\cdot)$ we have $pred_{G_2}(w_2) \subseteq proj^{w_1}_{out}(clos(\mathbb{P}'))$, which contradicts $w_2 \in unsur_{w_1}(proj^{w_1}_{out}(clos(\mathbb{P}')))$.

To prove the claim it is enough to recall that the set $\mathbb{P}^{w_1}(t-1) \cup T_+^{w_1}$ equals $I^{w_1}(\mathbb{P}_{t-1}) \cup I^{w_1}(\mathbb{P}_t)$ in Step (a) because the w_1 -block is surrounded, it equals $I^{w_1}(clos(\mathbb{P}_{t-1})) \cup I^{w_1}(\mathbb{P}_t)$ in Step (b)(i), and it equals $I^{w_1}(\mathbb{P}_{t-1}) \cup I^{w_1}(clos(\mathbb{P}_t))$ in Step (b)(ii). All three sets are contained in $proj^{w_1}_{any}(t)$ by the previous paragraph. \square

Claim 8.13 (Legal implies Lower Bound). *Assume that the pebbings $\{\mathcal{P}^{v_1}\}_{v_1 \in V(G_1)}$ and \mathcal{P} are legal, and \mathcal{P} is a persistent pebbling of G_1 . Then there is a configuration \mathbb{P}'_β in \mathcal{P}' of $\mathcal{R}(G_1, G_2)$ having at least $RPeb(G_1) + RPeb(G_2) + 1$ pebbles.*

Proof. Since \mathcal{P} is a legal persistent pebbling of G_1 , there is a configuration \mathbb{P}_b having at least $RPeb(G_1)$ pebbles. By definition of the algorithm \mathbb{P}_b is created in Step (c) when an output node $(v_1, v_2)_{out}$ is pebbled or unpebbled to get to \mathbb{P}'_t in \mathcal{P}' , so the configuration \mathbb{P}_b is between $\mathbb{P}(t-1)$ and $\mathbb{P}(t)$. Let \mathbb{P}'_β be \mathbb{P}'_t if the output node $(v_1, v_2)_{out}$ is pebbled, and \mathbb{P}'_{t-1} if the output node $(v_1, v_2)_{out}$ is unpebbled, then $\mathbb{P}'_\beta = \mathbb{P}'_{t-1} \cup \mathbb{P}'_t$.

Note that for any $u_1 \in \mathbb{P}_b$ we know that between $\mathbb{P}^{u_1}(t-1)$ and $\mathbb{P}^{u_1}(t)$, some configuration in \mathcal{P}^{u_1} is not empty, because $u_1 \in Ext(t-1) \cup Ext(t) \subseteq D = proj^V(t-1) \cup proj^V(t)$. It follows from Claim 8.12 that there is at least one pebble in every u_1 -block of \mathbb{P}'_β .

If $proj^P(t-1) \neq proj^P(t)$, that is if there is $w_1 \in proj^P(t-1) \triangle proj^P(t)$, then we are done. Since exactly one of $\mathbb{P}^{w_1}(t-1)$ and $\mathbb{P}^{w_1}(t)$ is p-locked, there is a configuration between them having at least $RPeb(G_2)$ pebbles, and by Claim 8.12 at least $RPeb(G_2)$ cells of the w_1 -block have pebbles. Summing up, in \mathbb{P}'_β , there are

$$\underbrace{RPeb(G_2)}_{w_1 \text{ block}} + \underbrace{RPeb(G_1) - 1}_{\text{other blocks}}$$

cells having pebbles, and among them the (v_1, v_2) -cell has three pebbles, for a total of $RPeb(G_1) + RPeb(G_2) + 1$ pebbles as needed.

Assume that $proj^P(t-1) = proj^P(t)$ instead. Recall $Ext(\cdot) = proj^P(\cdot) \cup (proj^V(\cdot) \cap unsur_{G_1}(proj^P(\cdot)))$. In Step (c) we know $Ext(t-1) \neq Ext(t)$, therefore at least one of $D_+ := Ext(t) \setminus Ext(t-1)$ and $D_- := Ext(t-1) \setminus Ext(t)$ is not empty. Swap t and $t-1$ if necessary, by their symmetry in the rest of this lemma, assume $D_- \neq \emptyset$. Fix $w_1 \in D_- = Ext(t-1) \setminus Ext(t)$.

Since $proj^P(t-1) \setminus Ext(t) \subseteq proj^P(t-1) \setminus proj^P(t) = \emptyset$, we can assume that $w_1 \in (proj^V(t-1) \cap unsur_{G_1}(proj^P(t-1))) \setminus Ext(t)$.

In particular $w_1 \in (proj^V(t-1) \setminus proj^V(t)) \cap unsur_{G_1}(proj^P(t))$. Let $u_1 \in pred_{G_1}(w_1)$ be such that $u_1 \notin proj^P(t-1) = proj^P(t)$.

Now $w_1 \in proj^V(t-1) \setminus proj^V(t)$, and as in the proof of Claim 8.11, we know that the w_1 -block is surrounded in \mathbb{P}'_{t-1} or in \mathbb{P}'_t . This implies that the sink of the u_1 -block $(u_1, z_2)_{out}$ has a pebble, and so $z_2 \in \mathbb{P}^{u_1}(t-1)$ or $z_2 \in \mathbb{P}^{u_1}(t)$. But u_1 is not in the p-projection at either time, so there is some other pebble other than z_2 in $\mathbb{P}^{u_1}(t-1)$ and $\mathbb{P}^{u_1}(t)$. By Claim 8.12, at least 2 cells of the u_1 -block have pebbles.

Since $w_1 \in proj^V(t-1) \setminus proj^V(t)$, exactly one of $\mathbb{P}^{w_1}(t-1)$ and $\mathbb{P}^{w_1}(t)$ is v-locked so there is a configuration of \mathcal{P}^{w_1} between them having at least $RPeb^V(G_2)$ pebbles, and by Claim 8.12 at least $RPeb^V(G_2)$ cells of the w_1 -block have pebbles. Summing up, in \mathbb{P}'_β , there are

$$\underbrace{RPeb^V(G_2)}_{w_1 \text{ block}} + \underbrace{2}_{u_1 \text{ block}} + \underbrace{RPeb(G_1) - 2}_{\text{other blocks}} \geq RPeb(G_1) + RPeb(G_2) - 1$$

cells having pebbles, and among them the (v_1, v_2) -cell has three pebbles, for a total of $RPeb(G_1) + RPeb(G_2) + 1$ pebbles as needed. \square

This product construction does not work for the standard black pebbling: there is no single constant C such that for any two graphs G_1 and G_2 , $Peb(\mathcal{R}(G_1, G_2)) = Peb(G_1) + Peb(G_2) + C$, therefore we need to develop a different construction in the next section.

Indeed, if we take G_1 and G_2 to be the singleton graph, which has pebbling price 1, the product construction is the pyramid of height 1, which has pebbling price 3. This gives a value of 1 for C .

If we take G_1 and G_2 to be the path of length 1, which has pebbling price 2, the product construction has pebbling price 4. This gives a value of 0 for C .

If we take G_1 and G_2 to be the pyramid of height 1, which has pebbling price 3, the product construction has pebbling price 5. An optimal strategy is to pebble the sinks of the two G_2 copies corresponding to sources in G_1 , then pebble all the exterior nodes of the remaining copy of G_2 , unpebble the sinks and finish the pebbling. This gives a value of -1 for C .

9 Product Construction for Standard Pebbling

The part of the proof of Theorem 3.4 that deals with standard pebbling uses as a black box the construction in Theorem 3.5 for standard pebbling. Now we state it again and we give its full proof.

Theorem 9.1. *Given two graphs G_1 and G_2 of standard pebbling price at least 3, there is a polynomial-time constructible graph $\mathcal{S}(G_1, G_2)$ of size $|G_1|(2|G_1| + |G_2|)$ with standard pebbling price $Peb(\mathcal{S}(G_1, G_2)) = Peb(G_1) + Peb(G_2) - 1$.*

For the rest of the section we fix G_1 and G_2 to be two single sink directed acyclic graphs, with sinks z_1 and z_2 , respectively. We set $p_1 := Peb(G_1)$ and $p_2 := Peb(G_2)$, and we assume that p_2 is at least 3.

Construction 9.2 (Product for standard pebbling). A *centipede* of length ℓ is a path of length ℓ where all nodes but the source have an extra predecessor. Formally, it is the graph with vertices $\{r_0, \dots, r_\ell\}$, $\{s_1, \dots, s_\ell\}$ and edges $\{(r_{i-1}, r_i) : i \in [\ell]\} \cup \{(s_i, r_i) : i \in [\ell]\}$.

As the first step we define the graph \hat{G}_2 from G_2 as follows: \hat{G}_2 is the union of G_2 and of a centipede of length $|G_1|$, where we identify the sink of G_2 with the vertex r_0 of the centipede. Observe that the pebbling price of \hat{G}_2 is $\max(3, p_2) = p_2$.

The graph $\mathcal{S}(G_1, G_2)$ is as follows. For every vertex v_1 of G_1 we make a copy of \hat{G}_2 , which we call the v_1 -block. Then, for every edge (u_1, v_1) in G_1 , we add edges from the sink of the u_1 -block to all the sources of the centipede in the v_1 -block. Formally, $\mathcal{S}(G_1, G_2)$ is the graph with vertices $\{(v_1, v_2) : v_1 \in V(G_1), v_2 \in V(\hat{G}_2)\}$ and edges $\{((v_1, u_2), (v_1, v_2)) : v_1 \in V(G_1), (u_2, v_2) \in E(G_2)\}, \{((u_1, r_{|G_1|}), (v_1, s_i)) : (u_1, v_1) \in E(G_1), i \in [|G_1|])\}$.

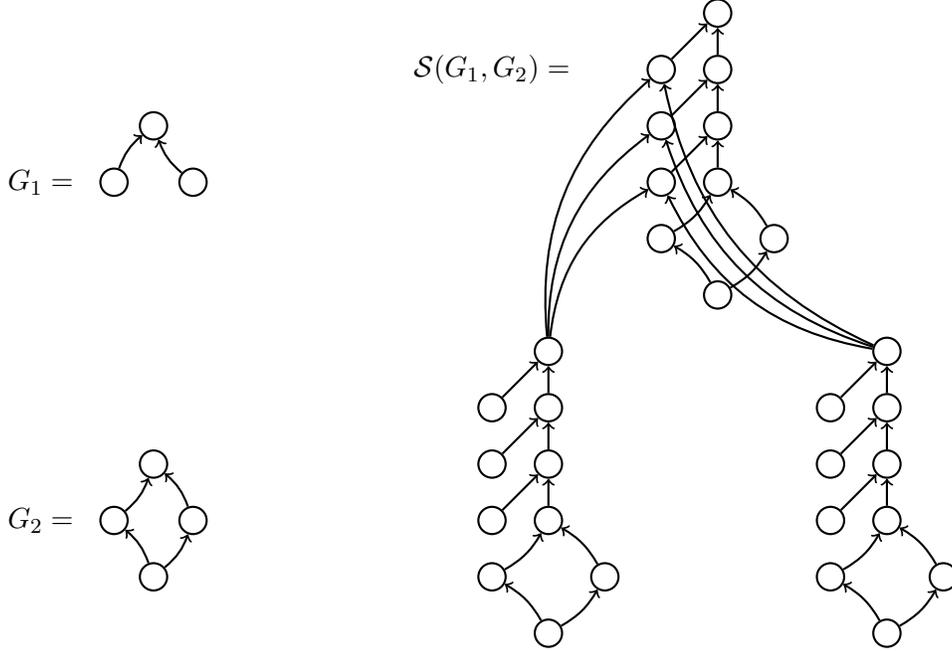


Figure 20: Example of Construction 9.2: product of a pyramid of height 1 and a rhombus.

Lemma 9.3. *The standard pebbling price of $\mathcal{S}(G_1, G_2)$ is at most $p_1 + p_2 - 1$.*

Proof. To pebble $\mathcal{S}(G_1, G_2)$ we simulate a strategy for pebbling G_1 with p_1 pebbles. When a pebble is placed in v_1 , we pebble the sink of the v_1 -block using a strategy for \hat{G}_2 in p_2 pebbles. When a pebble is removed from v_1 , we remove the pebble from the sink of the v_1 -block. When we put a pebble in some v_1 -block there are at most $p_1 - 1$ other non empty blocks and they all have exactly one pebble, therefore this strategy for $\mathcal{S}(G_1, G_2)$ is within the pebbling limit. \square

Lemma 9.4. *The standard pebbling price of $\mathcal{S}(G_1, G_2)$ is at least $p_1 + p_2 - 1$.*

Proof. Given a pebbling $\mathcal{P}^S = (\mathbb{P}_0^S, \mathbb{P}_1^S, \dots, \mathbb{P}_\tau^S)$ for the graph $\mathcal{S}(G_1, G_2)$ we construct a pebbling \mathcal{P} for G_1 in such a way that if the space of \mathcal{P}^S is less than $p_1 + p_2 - 1$ then \mathcal{P} has space less than p_1 , which is impossible.

In particular for any configuration \mathbb{P}_t^S in \mathcal{P}^S , we build a sequence $\mathcal{P}^{[t]}$ of pebbling configurations for G_1 , such that the final pebbling \mathcal{P} of G_1 is the concatenation of $\mathcal{P}^{[0]}, \mathcal{P}^{[1]}, \dots, \mathcal{P}^{[\tau]}$. While we build these sequences of configurations, we say that a vertex $v_1 \in V(G_1)$ is *active* if in the last configuration built so far v_1 does not have a pebble while all of its predecessors do.

If \mathbb{P}_t^S follows from a pebbling removal after which some v_1 -block of $\mathcal{S}(G_1, G_2)$ becomes empty then $\mathcal{P}^{[t]}$ is the pebbling step that removes the pebble present on v_1 , if any, otherwise $\mathcal{P}^{[t]}$ is the empty sequence.

If \mathbb{P}_t^S is the result of a pebble placement after which some v_1 -block of $\mathcal{S}(G_1, G_2)$ contains p_2 pebbles, then $\mathcal{P}^{[t]}$ performs the following pebbling steps: place a pebble on each empty vertex $w_1 \in V(G_1)$ such that

1. the w_1 -block of $\mathcal{S}(G_1, G_2)$ contains a pebble in \mathbb{P}_t^S ; and
2. w_1 is active.

This process is repeated until there is no vertex in $V(G_1)$ that meets both conditions. In particular a pebbling placement in the pebbling \mathcal{P}^S can cause a long chain of pebbling placements in \mathcal{P} .

Pebbling \mathcal{P} is a legal standard pebbling of G_1 since removals are always legal, and pebbling placements are only done on active vertices by construction.

Claim 9.5. *Assume the pebbling \mathcal{P}^S uses at most $p_1 + p_2 - 1$ pebbles. If there is a pebble on $(v_1, r_{|G_1|})$ in \mathbb{P}_t^S , then there is a pebble on v_1 at the end of $\mathcal{P}^{[t]}$.*

Proof. We prove the claim by induction over a topological order of G_1 . Consider the earliest time t_1 such that \mathcal{P}^S has p_2 pebbles in the v_1 -block and there is a pebble in the v_1 -block during the whole interval $[t_1, t]$. Such a time exists because the v_1 -block is a copy of \hat{G}_2 , and p_2 pebbles are necessary to pebble its sink.

If v_1 is active at any point during the construction of $\mathcal{P}^{[t_1]}$, then v_1 is pebbled in that sequence and it is not removed afterwards. This is always the case if v_1 is a source of G_1 .

If v_1 is not active we assume that Claim 9.5 holds for all its predecessors. Time t_1 is the first time when there are p_2 pebbles in the v_1 -block since it has been empty. Therefore none of the successors of (v_1, z_2) in the v_1 -block has a pebble. Also, since at most $p_1 - 1 < |G_1|$ pebbles are outside the G_2 part of the v_1 -block, some vertex (v_1, s_i) in the centipede part of the v_1 -block has no pebble.

So far we discovered that at time t_1 there is a path $(v_1, s_i), (v_1, r_i), (v_1, r_{i+1}) \dots, (v_1, r_{|G_1|})$ with no pebbles, and that at time t vertex $(v_1, r_{|G_1|})$ has a pebble. Then it must be the case that vertex (v_1, s_i) is pebbled at some time t_2 where $t_1 < t_2 < t$, and furthermore at time t_2 there must be pebbles on $(u_1, r_{|G_1|})$ and $(w_1, r_{|G_1|})$, where u_1 and w_1 are the predecessors of v_1 in graph G_1 . By induction hypothesis u_1 and w_1 have a pebble at the end of $\mathcal{P}^{[t_2]}$, so v_1 is active at that point and, since the v_1 -block is not empty, it gets a pebble. Such pebble stays in place at least until the end of $\mathcal{P}^{[t]}$. \square

We can finally prove Lemma 9.4. Assume for the sake of contradiction that \mathcal{P}^S uses strictly less than $p_1 + p_2 - 1$ pebbles. Pebbling \mathcal{P} is a legal pebbling of G_1 which pebbles the sink z_1 , because of Claim 9.5 and the fact that \mathcal{P}^S pebbles vertex $(z_1, r_{|G_1|})$. Consider a configuration in which \mathcal{P} reaches its maximum number of pebbles. This configuration is at the end of a sequence $\mathcal{P}^{[t]}$ corresponding to a pebble placement in \mathcal{P}^S that causes some v_1 -block to have p_2 pebbles, since this is the only case in which a sequence $\mathcal{P}^{[t]}$ adds pebbles.

The corresponding configuration \mathbb{P}_t^S has p_2 pebbles in the v_1 -block and at most $p_1 - 2$ other non empty blocks by assumption. Empty blocks in \mathbb{P}_t^S corresponds to empty vertices in $\mathcal{P}^{[t]}$ by construction, so there are at most $p_1 - 1$ pebbles in all configurations in $\mathcal{P}^{[t]}$. This contradicts the fact that $\text{Peb}(G_1) = p_1$. \square

Observe that the only point where we used the fact that the length of a centipede is $|G_1|$ is to claim that there is one source without a pebble, so any length $u \geq \text{Peb}(G_1)$ would suffice. Since in general it is PSPACE-hard to compute $\text{Peb}(G_1)$, we settle for the trivial upper bound $|G_1|$.

10 Concluding Remarks

In this paper, we study the pebble game first introduced in [PH70] as well as the more restricted reversible pebble game in [Ben89], where by [Cha13a] the latter game is also equivalent to the Dymond–Tomba game [DT85] and the Raz–McKenzie game [RM99].

We establish that it is PSPACE-hard to approximate standard and reversible pebbling price up to any additive constant. To the best of our knowledge, these are the first hardness of approximation results for such pebble games, even for polynomial time. It would be very interesting to show stronger inapproximability results for pebbling price under stronger assumptions. On the one hand, we are only able to show additive hardness, but on the other hand our results hold for arbitrary algorithms using a polynomial amount of memory. It seems reasonable to believe that the problem should become much harder for algorithms restricted to polynomial time, but showing this seems like a challenging task—in some sense, it appears that pebbling might be so hard a problem that it is even hard to prove that it is hard.

Another challenging problem is to prove approximation hardness, or even just PSPACE-completeness, for the *black-white pebble game* [CS76] modelling nondeterministic computation. This game is a strict generalization of the standard (black) pebble game, and so intuitively it should be at least as hard, but the added option of placing nondeterministic white pebbles anywhere in the graph completely destroys locality and makes the reduction in [GLT80] break down. Hertel and Pitassi [HP10] showed a PSPACE-completeness result in the nonstandard setting when unbounded (and very large) fan-in is allowed. Essentially, the large fan-in makes it possible to lock down almost all pebbles in one place at a time (namely on the predecessors of a large fan-in vertex to be pebbled) and to completely rule out any use of white pebbles, reducing the whole problem to black pebbling (although this reduction, it should be stressed, is far from trivial). This approach does not work for bounded fan-in graphs, however, which is the standard setting studied in the 1970s and 80s and the setting that could potentially have interesting applications in, for instance, proof complexity.

We also show in this paper that standard black pebbling is asymptotically stronger than reversible pebbling by exhibiting families of DAGs over n vertices which have standard pebblings in space s but for which the reversible pebbling price is $\Omega(s \log n)$. Since any DAG on n vertices with standard pebbling price s can be reversibly pebbled in space $O(s^2 \log n)$, our separation is at most a linear factor (in $s \leq n$) off from the optimal. It would be interesting to determine how large the separation can be. We do not rule out the possibility that the separation we give might in fact be asymptotically optimal.

Acknowledgements

We are grateful to Anna Gál, Yuval Filmus, Toniann Pitassi, and Robert Robere for stimulating discussions on the topic of pebble games. A special thanks goes to Mladen Mikša, who participated in the initial stages of this work but somehow managed to avoid the pebbling addiction that seized the rest of us...

The first author performed part of this work while at Princeton University. The second, third and fourth authors were funded by the European Research Council under the European Union’s Seventh Framework Programme (FP7/2007–2013) / ERC grant agreement no. 279611. The third author was also supported by the Swedish Research Council grants 621-2012-5645 and 2016-00782, and by the Independent Research Fund Denmark grant 9040-00389B.

References

- [AS15] Joël Alwen and Vladimir Serbinenko. High parallel complexity graphs and memory-hard functions. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC ’15)*, pages 595–603, June 2015.

- [Ben73] Charles H. Bennett. Logical reversibility of computation. *IBM Journal of Research and Development*, 17(6):525–532, November 1973.
- [Ben89] Charles H Bennett. Time/space trade-offs for reversible computation. *SIAM Journal on Computing*, 18(4):766–776, August 1989.
- [BTV01] Harry Buhrman, John Tromp, and Paul Vitányi. Time and space bounds for reversible simulation. *Journal of Physics A: Mathematical and general*, 34:6821–6830, 2001. Preliminary version in *ICALP '01*.
- [CFLS95] Anne Condon, Joan Feigenbaum, Carsten Lund, and Peter W Shor. Probabilistically checkable proof systems and nonapproximability of PSPACE-hard functions. *Chicago Journal of Theoretical Computer Science*, 1995, October 1995. Preliminary version in *STOC '93*.
- [Cha73] Ashok K. Chandra. Efficient compilation of linear recursive programs. In *Proceedings of the 14th Annual Symposium on Switching and Automata Theory (SWAT '73)*, pages 16–25, 1973.
- [Cha13a] Siu Man Chan. Just a pebble game. In *Proceedings of the 28th Annual IEEE Conference on Computational Complexity (CCC '13)*, pages 133–143, June 2013.
- [Cha13b] Siu Man Chan. *Pebble Games and Complexity*. PhD thesis, University of California at Berkeley, 2013.
- [Coo74] Stephen A. Cook. An observation on time-storage trade off. *Journal of Computer and System Sciences*, 9(3):308–316, 1974. Preliminary version in *STOC '73*.
- [CP14] Siu Man Chan and Aaron Potechin. Tight bounds for monotone switching networks via Fourier analysis. *Theory of Computing*, 10:389–419, October 2014. Preliminary version in *STOC '12*.
- [CS76] Stephen A. Cook and Ravi Sethi. Storage requirements for deterministic polynomial time recognizable languages. *Journal of Computer and System Sciences*, 13(1):25–37, 1976. Preliminary version in *STOC '74*.
- [DL17] Erik D. Demaine and Quanquan C. Liu. Inapproximability of the standard pebble game and hard to pebble graphs. In *Proceedings of the 15th International Symposium on Algorithms and Data Structures (WADS '17)*, pages 313–324, July 2017.
- [DNW05] Cynthia Dwork, Moni Naor, and Hoeteck Wee. Pebbling and proofs of work. In *Proceedings of the 25th Annual International Cryptology Conference (CRYPTO '05)*, volume 3621 of *Lecture Notes in Computer Science*, pages 37–54. Springer, August 2005.
- [DT85] Patrick W. Dymond and Martin Tompa. Speedups of deterministic machines by synchronous parallel machines. *Journal of Computer and System Sciences*, 30(2):149–161, April 1985. Preliminary version in *STOC '83*.
- [FNPW10] Yuval Filmus, Jakob Nordström, Toniann Pitassi, and Yu Wu. Unpublished note, 2010.
- [FPRC13] Yuval Filmus, Toniann Pitassi, Robert Robere, and Stephen A Cook. Average case lower bounds for monotone switching networks. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS '13)*, pages 598–607, November 2013.
- [GLT80] John R. Gilbert, Thomas Lengauer, and Robert Endre Tarjan. The pebbling problem is complete in polynomial space. *SIAM Journal on Computing*, 9(3):513–524, August 1980. Preliminary version in *STOC '79*.

REFERENCES

- [HP10] Philipp Hertel and Toniann Pitassi. The PSPACE-completeness of black-white pebbling. *SIAM Journal on Computing*, 39(6):2622–2682, April 2010. Preliminary version in *FOCS '07*.
- [HPV77] John Hopcroft, Wolfgang Paul, and Leslie Valiant. On time versus space. *Journal of the ACM*, 24(2):332–337, April 1977. Preliminary version in *FOCS '75*.
- [Krá04] Richard Královič. Time and space complexity of reversible pebbling. *RAIRO – Theoretical Informatics and Applications*, 38(02):137–161, April 2004.
- [Lin78] Andrzej Lingas. A PSPACE-complete problem related to a pebble game. In *Proceedings of the 5th Colloquium on Automata, Languages and Programming (ICALP '78)*, pages 300–321, 1978.
- [LMT00] Klaus-Jörn Lange, Pierre McKenzie, and Alain Tapp. Reversible space equals deterministic space. *Journal of Computer and System Sciences*, 60(2):354–367, April 2000.
- [LTV98] Ming Li, John Tromp, and Paul Vitányi. Reversible simulation of irreversible computation. *Physica D: Nonlinear Phenomena*, 120(1–2):168–176, September 1998.
- [LV96] Ming Li and Paul Vitányi. Reversibility and adiabatic computation: Trading time and space for energy. *Proceedings of the Royal Society of London, Series A*, 452(1947):769–789, April 1996.
- [Nor13] Jakob Nordström. Pebble games, proof complexity and time-space trade-offs. *Logical Methods in Computer Science*, 9(3):15:1–15:63, September 2013.
- [Nor20] Jakob Nordström. New wine into old wineskins: A survey of some pebbling classics with supplemental results. Manuscript in preparation. To appear in *Foundations and Trends in Theoretical Computer Science*. Current draft version available at <http://www.csc.kth.se/~jakobn/research/>, 2020.
- [PH70] Michael S. Paterson and Carl E. Hewitt. Comparative schematology. In *Record of the Project MAC Conference on Concurrent Systems and Parallel Computation*, pages 119–127, 1970.
- [Pip80] Nicholas Pippenger. Pebbling. Technical Report RC8258, IBM Watson Research Center, 1980. In *Proceedings of the 5th IBM Symposium on Mathematical Foundations of Computer Science*.
- [Pot10] Aaron Potechin. Bounds on monotone switching networks for directed connectivity. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS '10)*, pages 553–562, October 2010.
- [PTC77] Wolfgang J. Paul, Robert Endre Tarjan, and James R. Celoni. Space bounds for a game on graphs. *Mathematical Systems Theory*, 10:239–251, 1977.
- [RM99] Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Combinatorica*, 19(3):403–435, March 1999. Preliminary version in *FOCS '97*.
- [Seg07] Nathan Segerlind. The complexity of propositional proofs. *Bulletin of Symbolic Logic*, 13(4):417–481, December 2007.
- [Set75] Ravi Sethi. Complete register allocation problems. *SIAM Journal on Computing*, 4(3):226–248, September 1975.
- [SM73] Larry J. Stockmeyer and Albert R. Meyer. Word problems requiring exponential time (preliminary report). In *Proceedings of the 5th Annual ACM Symposium on Theory of Computing (STOC '73)*, pages 1–9, 1973.

- [SS77] Sowmitri Swamy and John E. Savage. Space-time trade-offs on the FFT-algorithm. Technical Report CS-31, Brown University, 1977.
- [SS79] John E. Savage and Sowmitri Swamy. Space-time tradeoffs for oblivious interger multiplications. In *Proceedings of the 6th International Colloquium on Automata, Languages and Programming (ICALP '79)*, pages 498–504, 1979.
- [SS83] Sowmitri Swamy and John E. Savage. Space-time tradeoffs for linear recursion. *Mathematical Systems Theory*, 16(1):9–27, 1983.
- [Tom78] Martin Tompa. Time-space tradeoffs for computing functions, using connectivity properties of their circuits. In *Proceedings of the 10th Annual ACM symposium on Theory of computing (STOC '78)*, pages 196–204, 1978.
- [VT89] H. Venkateswaran and Martin Tompa. A new pebble game that characterizes parallel complexity classes. *SIAM Journal on Computing*, 18(3):533–549, June 1989. Preliminary version in *FOCS '86*.
- [WAPL14] Yu Wu, Per Austrin, Toniann Pitassi, and David Liu. Inapproximability of treewidth and related problems. *Journal of Artificial Intelligence Research*, 49:569–600, April 2014.
- [Wil00] Ryan Williams. Space-efficient reversible simulations. Technical report, Cornell University, 2000. Available at http://web.stanford.edu/~rrwill/spacesim9_22.pdf.