Inapproximability of the Standard Pebble Game and Hard to Pebble Graphs

Erik D. Demaine, Quanquan C. Liu

MIT CSAIL, Cambridge, Massachusetts

Abstract. Pebble games are single-player games on DAGs involving placing and moving pebbles on nodes of the graph according to a certain set of rules. The goal is to pebble a set of target nodes using a minimum number of pebbles. In this paper, we present a possibly simpler proof of the result in [4] and strengthen the result to show that it is PSPACE-hard to determine the minimum number of pebbles to an additive $n^{1/3-\varepsilon}$ term for all $\varepsilon > 0$, which improves upon the currently known additive *constant* hardness of approximation [4] in the standard pebble game. We also introduce a family of explicit, constant indegree graphs with n nodes where there exists a graph in the family such that using $0 < k < \sqrt{n}$ pebbles requires $\Omega((n/k)^k)$ moves to pebble in both the standard and black-white pebble games. This independently answers an open question summarized in [14] of whether a family of DAGs exists that meets the upper bound of $O(n^k)$ moves using constant k pebbles with a different construction than that presented in [1].

1 Introduction

Pebble games were originally introduced to study compiler operations and programming languages. For such applications, a DAG represents the computational dependency of each operation on a set of previous operations and pebbles represent register allocation. Minimizing the amount of resources allocated to perform a computation is accomplished by minimizing the number of pebbles placed on the graph [16]. The *standard pebble game* (also known as the *black pebble game*) is traditionally used to model such behavior. In the standard pebble game, one is given a DAG, $G = (V, E)$, with n nodes and constant indegree and told to perform a set of *pebbling moves* that places, removes, or slides pebbles around the nodes of G.

The premise of such games is given some input modeled by *source* nodes $S \subseteq V$ one should compute some set of outputs modeled as target nodes $T \subseteq V$. In terms of G, S is typically the set of nodes without incoming edges and T is typically the set of nodes without outgoing edges. The rules of the standard pebble game are as follows:

© Springer International Publishing AG 2017

F. Ellen et al. (Eds.): WADS 2017, LNCS 10389, pp. 313–324, 2017.
DOI: 10.1007/978-3-319-62127-2_27

Standard Pebble Game
Input: Given a DAG, $G_{n,\delta} = (V, E)$. Let $\mathsf{pred}(v) = \{u \in V : (u, v) \in E\}$. Let $S \subseteq V$ be the set of sources of G and $T \subseteq V$ be the set of targets of G. Let $\mathcal{P} = \{P_0, \ldots, P_\tau\}$ be a valid pebbling strategy that obeys the following rules where P_i is a set of nodes containing pebbles at timestep i and $P_0 = \emptyset$ and $P_\tau = \{T\}$. Let $\mathsf{Peb}(G, \mathcal{P}) = \max_{i \in [\tau]}\{|P_i|\}$.

Rules:

1. At most one pebble can be placed or removed from a node at a time.
2. A pebble can be placed on any source, $s \in S$.
3. A pebble can be removed from any vertex.
4. A pebble can be placed on a non-source vertex, v, at time i if and only if its direct predecessors are pebbled, $\mathsf{pred}(v) \in P_{i-1}$.
5. A pebble can slide from vertex v to vertex w at time i if and only if $(v, w) \in E$ and $\mathsf{pred}(w) \in P_{i-1}$.

Goal: Determine $\min_{\mathcal{P}}\{\mathsf{Peb}(G, \mathcal{P})\}$ using a valid strategy $\mathcal{P}$.

In addition to the standard pebble game, other pebble games are useful for studying computation. The *red-blue pebble game* is used to study I/O complexity [12], the *reversible pebble game* is used to model reversible computation [3], and the *black-white pebble game* is used to model non-deterministic straight-line programs [5]. Although we will be proving a result about the black-white pebble game in Section 4, we will defer introducing the rules of the game to our full paper [6] since the black-white pebble game is not central to the main results of this paper.

Much previous research has focused on proving lower and upper bounds on the *pebbling space cost* (i.e. the maximum number of pebbles over time) of pebbling a given DAG under the rules of each of these games. For all of the aforementioned pebble games (except the red-blue pebble game since it relies on a different set of parameters), any DAG can be pebbled using $O(n/\log n)$ pebbles [9,11,15]. Furthermore, there exist DAGs for each of the games that require $\Omega(n/\log n)$ pebbles [9,11,15].

It turns out that finding a strategy to optimally pebble a graph in the standard pebble game is computationally difficult even when each vertex is allowed to be pebbled only once. Specifically, finding the minimum number of black pebbles needed to pebble a DAG in the standard pebble game is PSPACE-complete [8] and finding the minimum number of black pebbles needed in the one-shot case is NP-complete [16]. In addition, finding the minimum number of pebbles in both the black-white and reversible pebble games have been recently shown to be both PSPACE-complete [4,10]. But the result for the black-white pebble game is proven for unbounded indegree [10]. A key open question in the field is whether hardness results can be obtained for constant indegree graphs for the black-white pebble game. However, whether it is possible to find good approximate solutions to the minimization problem has barely been studied. In fact, it was not known

until this paper whether it is hard to find the minimum number of pebbles within even a non-constant *additive* factor [4]. The best known multiplicative approximation factor is the very loose $\Theta(n/\log n)$ which is the pebbling space upper bound [11], leaving much room for improvement.

Our results deal primarily with the standard pebble game, but we believe that the techniques could be extended to show hardness of approximation for other pebble games. We prove the following:

Theorem 1. *The minimum number of pebbles needed in the standard pebble game on DAGs with maximum indegree* 2 *is PSPACE-hard to approximate to within an additive* $n^{1/3-\varepsilon}$ *for any* $\varepsilon > 0$.

In addition to determining the pebbling space cost, we sometimes also care about pebbling time which refers to the number of operations (placements, removals, or slides) that a strategy uses. For example, such a situation arises if we care not only about the memory used in computation but also the time of computation. It is previously known that there exists a family of graphs such that, given $\Theta(\frac{n}{\log n})$ pebbles, one is required to use $\Omega(2^{\Theta(\frac{n}{\log n})})$ moves to pebble any graphs with n nodes in the family [13].

Less is known about the trade-offs when a small number (e.g. constant k) of pebbles is used until the very recent, independent result presented in [1]. It can be easily shown through a combinatorial argument that the maximum number of moves necessary using $k = O(1)$ pebbles to pebble n nodes is $O(n^k)$ [14]. It is an open question whether it is possible to prove a time-space trade-off such that using $k = O(1)$ pebbles requires $\Omega(n^k)$ time. In this paper, we resolve this open question for both the standard pebble and the black-white pebble games using an independent construction from that presented in [1].

Theorem 2. *There exists a family of graphs with* n *vertices and maximum indegree* 2 *such that* $\Omega((\frac{n-k^2}{k})^k)$ *moves are necessary to pebble any graph with* n *vertices in the family using* $k < \sqrt{n}$ *pebbles in both the standard and black-white pebble games.*

In particular, when $k = O(1)$, the number of moves necessary to pebble a graph in the family is $\Theta(n^k)$.

The organization of the paper is as follows. First, in Section 2, we provide the definitions and terminology we use in the remaining parts of the paper. Then, in Section 3, we provide a proof for the inapproximability of the standard pebble game to an $n^{1/3-\varepsilon}$ additive factor.

In Section 4, we present our hard to pebble graph families using $k < \sqrt{n}$ pebbles and prove that the family takes $\Omega(n^k)$ moves to pebble in both the standard and black-white pebble games when $k = O(1)$.

Finally, in Section 5, we discuss some open problems resulting from this paper.

2 Definitions and Terminology

In this section, we define the terminology we use throughout the rest of the paper. All of the pebble games we consider in this paper are played on directed acyclic graphs (DAGs). In this paper, we only consider DAGs with maximum indegree 2. We define such a DAG as $G = (V, E)$ where $|V| = n$ and $|E| = m$.

The purpose of any pebble game is to pebble a set of targets $T \subseteq V$ using minimum number of pebbles. In all pebble games we consider, a player can always place a pebble on any source node, $S \subseteq V$. Usually, S consists of all nodes with indegree 0 and T consists of all nodes with outdegree 0.

A *sequential pebbling strategy*, $\mathcal{P} = [P_0, \ldots, P_\tau]$ is a series of configurations of pebbles on G where each P_i is a set of pebbled vertices $P_i \subseteq V$. P_i follows from P_{i-1} by the rules of the game and $P_0 = \emptyset$ and $P_\tau = T$. Then, by definition, $|P_i|$ is the number of pebbles used in configuration P_i. For a *sequential* strategy, $|P_{i-1}| - 1 \leq |P_i| \leq |P_{i-1}| + 1$ for all $i \in [\tau] = [1, \ldots, \tau]$ (i.e. at most one pebble can be placed, removed, or slid on the graph at any time). In this paper, we only consider sequential strategies.

Given any strategy $\mathcal{P}$ for pebbling G, the *pebbling space cost*, $\mathsf{Peb}(G, \mathcal{P})$, of $\mathcal{P}$ is defined as the maximum number of pebbles used by the strategy at any time: $\mathsf{Peb}(G, \mathcal{P}) = \max_{i \in [\tau]}\{|P_i|\}$.

The *minimum pebbling space cost* of G, $\mathsf{Peb}(G)$, is defined as the smallest space cost over the set of all valid strategies, $\mathbb{P}$, for G:

Definition 1 (Minimum Pebbling Space Cost).

$$\mathsf{Peb}(G) = \min_{\mathcal{P} \in \mathbb{P}}\{\mathsf{Peb}(G, \mathcal{P})\}.$$

The *pebbling time cost*, $\mathsf{Time}(G, \mathcal{P}, s) = |\mathcal{P}| - 1$, of a strategy $\mathcal{P}$ using s pebbles is the number of moves used by the strategy. The *minimum pebbling time cost* of any strategy that has pebbling space cost s is the minimum number of moves used by any such strategy.

Definition 2 (Minimum Pebbling Time Cost).

$$\mathsf{Time}(G, s) = \min_{\mathcal{P}' \in \{\mathcal{P} \in \mathbb{P} : |\mathcal{P}| \leq s\}}\{\mathsf{Time}(G, \mathcal{P}', s)\} \geq n.$$

3 Inapproximability of the Standard Pebble Game

In this section, we provide an alternative proof of the result presented in [4] that the standard pebble game is inapproximable to any constant additive term. Then, we show that our proof technique can be used to show our main result stated in Theorem 1. We make modifications to the proof presented by [8] to obtain our main result. A quick explanation of the relevant results presented in [8] can be found in our full paper [6].

3.1 Inapproximability to $n^{1/3-\varepsilon}$ additive term for any $\varepsilon > 0$

We now prove our main result. For our reduction we modify the variable, clause, and quantifier gadgets in [8] to produce a gap reduction from the PSPACE-hard problem, QBF.

Important Graph Components Before we dive into the details of our construction, we first mention two subgraphs and the properties they exhibit.

The first graph is the *pyramid graph* (please refer to [6] for a figure), Π_h with height h, which requires a number of pebbles that is equal to the height, h, of the pyramid to pebble [8]. Therefore, in order to pebble the apex of such a graph, at least h pebbles must be available. As in [8], we depict such pyramid graphs by a triangle with a number indicating the height (hence number of pebbles) needed to pebble the pyramid (see Figure 1 for an example of the triangle symbolism).

We make use of the following definition and lemma (restated and adapted) from [8] in our proofs:

Definition 3 (Frugal Strategy [8]). *A pebbling strategy, $\mathcal{P}$, is frugal if the following are true:*

1. *Suppose vertex $v \in G$ is pebbled for the first time at time t'. Then, for all times, $t > t'$, some path from v to q_1 (the only target node) contains a pebble.*
2. *At all times after v is pebbled for the last time, all paths from v to q_1 contain a pebble.*
3. *The number of pebble placements on any vertex $v \in G$ where $v \neq q_1$, is bounded by the number of pebble placements on $\mathsf{pred}(v)$.*

Lemma 1 (Normal Pebbling Strategy [8]). *If the target vertex is not inside a pyramid, Π_h, and each of the vertices in the bottom level of the pyramid has at most one predecessor each, then any pebbling strategy can be transformed into a* normal *pebbling strategy without increasing the number of pebbles used. A normal pebbling strategy is one that is frugal and after the first pebble is placed on any pyramid, Π_h, no placements of pebbles occurs outside Π_h until the apex of Π_h is pebbled and all other pebbles are removed from Π_h.*

The other important subgraph is the *road graph* (see [6] for figure), R_w with width w, which requires a number of pebbles that is the width of the graph to pebble *any* of the outputs [7,14]. Therefore, we state as an immediately corollary of their proof:

Corollary 1 (Road Graph Pebbling). *To pebble $O \subseteq \{o_1, \ldots, o_w\}$ of the outputs of R_w, with a valid strategy, $\mathcal{P} = [P_0, \ldots, P_\tau]$ where $P_\tau = O$, requires $w + |O| - 1$ pebbles.*

We define a *regular* pebbling strategy for road graphs similarly to the *normal* pebbling strategy for pyramids.

Lemma 2 (Regular Pebbling Strategy). *If each input, $i_j \in \{i_1, \ldots, i_w\}$, to the road graph has at most* 1 *predecessor, any pebbling strategy can be transformed into a* regular *pebbling strategy without increasing the number of pebbles used. A regular pebbling strategy is one that is frugal and after the first pebble is placed on any road graph, R_w, no placements of pebbles occurs outside R_w until a set of desired outputs, $O \subseteq \{o_1, \ldots, o_w\}$, of R_w all contain pebbles and all other pebbles are removed from R_w.*

We immediately obtain the following corollary from Lemmas 1 and 2:

Corollary 2. *Any pebbling strategy, $\mathcal{P}$, can be transformed into a pebbling strategy, $\mathcal{P}'$, that is normal and regular if no target vertices lie inside a pyramid or road graph and each input node to either the pyramid or road graph has at most one predecessor.*

Modified Graph Constructions We first describe the changes we made to each of the gadgets used in the PSPACE-completeness proof presented by [8] and then prove our inapproximability result using these gadgets in Section 3.1. Given a QBF instance, $B = Q_1 x_1 \cdots Q_u x_u F$, with c clauses, we create the following gadgets:

Variable Nodes: We replace all variable nodes in the proof provided in [8] with road graphs each of width K. The modified variable nodes are shown in Figure 1. Each variable node as in the original proof by [8] has 3 possible configurations which are also shown in Figure 1.

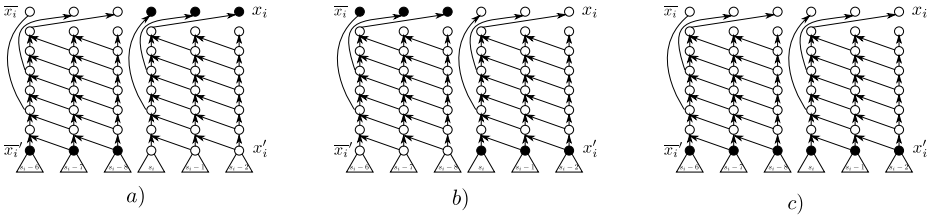

Fig. 1: Modified variable gadget with 3 possible configurations using the road graph subgraph previously described. Here $K = 3$. a) x_i is True. b) $\overline{x_i}$ is True. c) Double false.

Quantifier Blocks: Each universal and existential quantifier blocks are also modified to account for the new variable nodes. See Figure 2 and Figure 3 which depict the new quantifier gadgets that use the new variable nodes. Note that instead of each quantifier gadget requiring 3 total pebbles, each gadget requires $3K$ pebbles to remain on each block before the clauses are pebbled. The basic idea is to expand all nodes $a_i, b_i, c_i ...etc.$ into a path of length K to account for each of the K copies of x_i and each of the K copies of $\overline{x_i}$. Each $s_i = s_{i-1} - 3K$ and $s_1 = 3Ku + 3K + 1$.

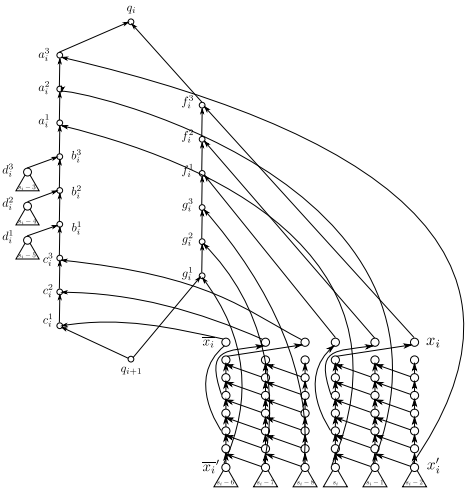

Fig. 2: Modified universal quantifier block. Here $K = 3$.

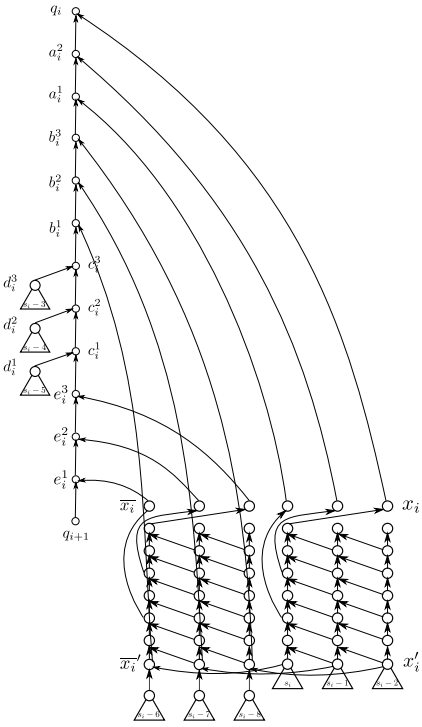

Fig. 3: Modified existential quantifier block. Here $K = 3$.

Clause Gadgets: Each clause gadget is modified to be a pyramid of height $3K + 1$ where the bottom layer is connected to 2 nodes from two different literals. Therefore, for a given clause (l_i, l_j, l_k), K nodes are connected to l_i and l_j, K nodes to l_j and l_k, and K nodes to l_i and l_k. See Figure 4 for an example of the modified clause gadget.

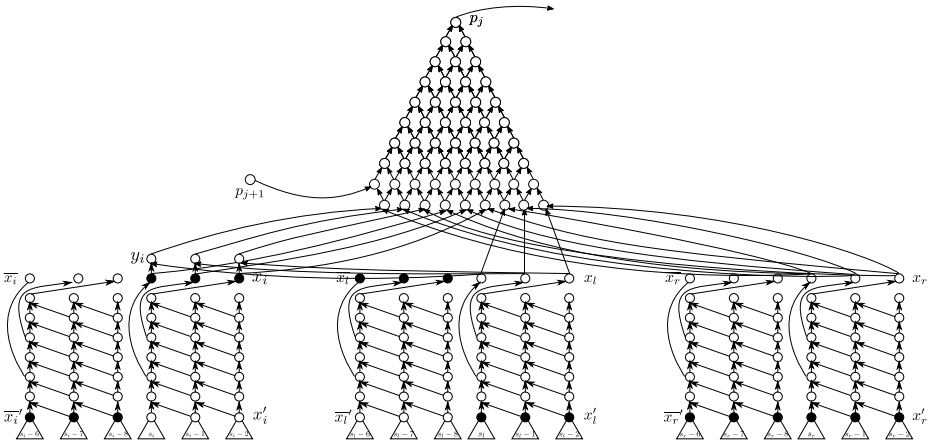

Fig. 4: Modified clause gadget. The clause here is (x_i, x_l, x_r) where $x_i = True$, $x_l = False$, and $x_r = False$. Here $K = 3$.

Proofs of the Construction We construct a graph G using the gadgets described above in Section 3.1 for any given QBF instance, $B = Q_1 x_1 \cdots Q_k x_u F$. In short, the proof relies on the fact that each quantifier gadget requires $2K$ pebbles to set the corresponding variable to true or false (i.e. the corresponding literals to true or false). An additional K pebbles need to remain on each quantifier in order to be able to repebble quantifiers when checking for universal variables' satisfaction. Furthermore, a clause would consist of modified pyramids of height $3K + 1$ connected to pairs of nodes from different literals. Following the proof in [8], the quantifier gadgets are pebbled first with $3Ku$ pebbles remaining on the quantifier gadgets. Then, the clauses are pebbled with $3K + 1$ pebbles.

If B is satisfiable, then clauses can be pebbled with $3K+1$ pebbles. Otherwise, $4K$ pebbles are needed to pebble one or more unsatisfied clauses in G, resulting in a gap of $K - 1$ pebbles between when B is satisfiable and unsatisfiable. Thus, if given an approximation algorithm that estimates the number of pebbles needed within additive $K - 1$, we can distinguish between the case when B is satisfiable (at most $3Km + 3K + 1$ pebbles are needed) and the case when B is unsatisfiable (when at least $3Km + 4K$ pebbles are needed).

In this construction, K can be any polynomial function of u where u is the number of variables in B and c is the number of clauses (in other words, $K = u^a c^b$

for any constants a and b). The total minimum number of pebbles necessary is $O(Ku)$ and the total number of nodes in the graph is $O(K^3(u+c))$.

We first prove that the number of pebbles needed to pebble each quantifier gadget is $3K$ and $3K$ pebbles remain on the quantifier blocks throughout the pebbling of the clauses.

Lemma 3. *Every regular and normal strategy, $\mathcal{P}'$, must be one where each quantifier gadget must be pebbled with $3K$ pebbles before the clauses are pebbled. Furthermore, each quantifier gadget must be pebbled wth $3K$ pebbles when q_u is pebbled.*

Next we prove that provided $3Ku$ pebbles stay on the quantifier blocks, each unsatisfied clause requires $4K$ pebbles.

Lemma 4. *Given a clause gadget, C_i, its corresponding variable, c_i is true if and only if C_i can be pebbled with $3K+1$ pebbles. Furthermore, if c_i is false and all literals in C_i are set in the false configuration, then at least $4K$ pebbles are necessary to pebble the clause.*

Given the previous proofs, we now prove the following key lemmas:

Lemma 5. *Given G which is constructed from the provided QBF instance, $B = Q_1x_1 \cdots Q_ux_uF$, using our modified reduction in Section 3.1, B is satisfiable if and only if $\mathsf{Peb}(G) \leq 3Ku + 3K + 1$.*

Before, we prove the next crucial lemma (Lemma 7), we first prove the following lemma which will help us prove Lemma 7:

Lemma 6. *Let N_i be the configuration such that some number of pebbles are on the first $i-1$ quantifier blocks and the i-th is being pebbled. Therefore, N_{u+1} is the configuration when some number of pebbles are on all u quantifier blocks and the first clause gadget is being pebbled. There does not exist a frugal strategy, $\mathcal{P}$, that can pebble our reduction construction, G, such that N_i contains less than $s - s_i$ pebbles on the first $i-1$ quantifier blocks when the i-th quantifier block or when the first clause is being pebbled.*

Lemma 7. *Given G which is constructed from the provided QBF instance, $B = Q_1x_1 \cdots Q_ux_uF$, using our modified reduction in Section 3.1, B is unsatisfiable if and only if $\mathsf{Peb}(G) \geq 3Ku + 4K$.*

Proof of Inapproximability Using Lemmas 5 and 7, we prove that it is PSPACE-hard to approximate the minimum number of black pebbles needed given a DAG, G, to an additive $n^{1-\varepsilon}$ factor.

Theorem 3 (Restatement of Theorem 1). *The minimum number of pebbles needed in the standard pebble game on DAGs with maximum indegree 2 is PSPACE-hard to approximate to additive $n^{1/3-\varepsilon}$ for $\varepsilon > 0$.*

Proof. From Lemmas 5 and 7, the cost of pebbling a graph constructed from a satisfiable B is at most $3Ku + 3K + 1$ whereas the cost of pebbling a graph constructed from an unsatisfiable B is at least $3Ku + 4K$. As we can see, the aforementioned reduction is a gap-producing reduction with a gap of $K - 1$ pebbles. Then, all that remains to be shown is that for any $\varepsilon > 0$, it is the case that $K \geq (K^3(u + c))^{(1/3-\varepsilon)}$. (Note that for $\varepsilon > 1/3$, setting K to any positive integer achieves this bound.) Suppose we set $K = \max(u, c)^a$ where $a > 0$. Given an $0 < \varepsilon \leq 1/3$, $a = \frac{1/3-\varepsilon}{1+3\varepsilon} \geq 0$ precisely when ε is in the stated range.

For values of $a \geq 0$, we can duplicate the clauses and variables gadgets so that u and c are large enough such that $K = \max(u, c)^a \geq 2$. Let $d = \max(u, c)$. Then, we need d to be large enough so that $d^a \geq 2$ (i.e. we want d^a to be some integer). Then, we can set $d \geq 2^{1/a}$. Thus, we can duplicate the number of variables and clauses so that $d \geq 2^{\frac{1+3\varepsilon}{1/3-\varepsilon}}$.

Therefore, for every $\varepsilon > 0$, we can construct a graph with a specific K calculated from ε such that it is PSPACE-hard to find an approximation within an additive $n^{1/3-\varepsilon}$ factor where n is the number of nodes in the graph.

4 Hard to Pebble Graphs for Constant k Pebbles

It is long known that the maximum number of moves necessary to pebble any graph with constant k pebbles is $O(n^k)$. (Note that the maximum number of moves necessary to pebble any graph is either $O(n^{k-1})$ or $O(n^k)$ depending on whether or not sliding is allowed. Here, we allow sliding in all of our games. The bound of $O(n^{k-1})$ proven in [14] is one for the case when sliding is not allowed.) The upper bound of $O(n^k)$ for any constant k number of pebbles submits to a simple combinatorial proof adapted from [14]) to account for sliding. However, to the best of the author's knowledge, examples of such families of graphs that require $O(n^k)$ moves to pebble using k pebbles did not exist until very recently in an independent work [1]. In this section, we present an independent, simple to construct family of graphs that require $\Theta(n^k)$ time for constant k number of pebbles in both the standard and black-white pebble games. We further reduce the indegree of nodes in this family of graphs to 2 and show that our results still hold. Furthermore, we show this family of graphs to exhibit a steep time-space trade-off (from exponential in k to linear) even when k is not constant. Such families of graphs could potentially have useful applications in cryptography in the domain of proofs of space and memory-hard functions [2].

We construct the following family of graphs, $\mathbb{H}_{n,\delta}$, below with n nodes and indegree δ and show that for constant k pebbles, the number of steps it takes to pebble the graph $H_{n,k} \in \mathbb{H}_{n,\delta}$ with k pebbles is $\Omega(n^k)$. We also show a family of graphs, $\mathbb{H}_{n,2}$ with indegree 2 that shows the same asymptotic tradeoff.

We construct the family of graphs in the following way.

Definition 4. *Given a set of n nodes and maximum number of pebbles k where $k < \sqrt{n}$, we lexicographically order the nodes (from 1 to n) and create the following set of edges between the nodes where directed edges are directed from v_i to v_j where $i < j$:*

1. v_i *and* v_{i+1} *for all* $i \in [k, n]$
2. v_i *and* v_j *for all* $i \in [l-1]$ *for all* $2 \leq l \leq k$ *and* $j \in \{f(l) + 2r - 2\}$ *for all* $r \in [\frac{n-k}{2k} + 1]$ *where* $f(l) = k + (l-1)(\frac{n-k}{k}) + 1$.
3. v_i *and* v_j *for all* $i = f(l) - 1$ *and* $j \in \{f(l) + 2r - 1\}$ *for all* $i \in [\frac{n-k}{2k} + 1]$ *where* $l \in [1, k-1]$.

The target node (the only sink) is v_n. *Note that the sources in our construction are* v_j *for all* $j \in [1, k]$.

Due to the space constraints, we leave all proofs of the properties of the graph family as well as an example figure of a member of the family in our full paper [6].

5 Open Problems

There are a number of open questions that naturally follow the content of this paper.

The first obvious open question is whether the techniques introduced in this paper can be tweaked to allow for a PSPACE-hardness of approximation to an $n^{1-\varepsilon}$ additive factor for any $\varepsilon > 0$. We note that the trivial method of attempting to reduce the size of the subgraph gadgets used in the variables (i.e. use a different construction than the road graph such that less than K^3 nodes are used) is not sufficient since the number of nodes in the graph is still $\Theta(K^3(u+c))$. This is not to say that such an approach is not possible; simply that more changes need to be made to all of the other gadgets. The next logical step is to determine whether $\mathsf{Peb}(G)$ can be approximated to a constant 2 factor multiplicative approximation.

Another open question is whether the techniques introduced in this paper can be applied to show hardness of approximation results for other pebble games such as the black-white or reversible pebble games. The main open question in the topic of hardness of approximation of pebble games is whether the standard pebble game can be approximated to any factor smaller than $n/\log n$ or whether the games are PSPACE-hard to approximate to any constant factor, perhaps even logarithmic factors.

With regard to hard to pebble graphs, we wonder if our graph family could be improved to show $\Omega(n^k)$ for any $0 < k \leq n/\log n$. This would be interesting because to the best of the authors' knowledge we do not yet know of any graph families that exhibit sharp (asymptotically tight) time-space trade-offs for this entire range of pebble number.

References

1. Joël Alwen, Susanna F. de Rezende, Jakob Nordström, and Marc Vinyals. Cumulative space in black-white pebbling and resolution. In *Innovations in Theoretical Computer Science, ITCS 2017, Berkeley, CA, USA, 9-11 January, 2017*, 2017.
2. Joël Alwen and Vladimir Serbinenko. High parallel complexity graphs and memory-hard functions. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 595–603, 2015. Available from: `http://doi.acm.org/10.1145/2746539.2746622`.

3. Charles H. Bennett. Time/space trade-offs for reversible computation. *SIAM J. Comput.*, 18(4):766–776, August 1989. Available from: http://dx.doi.org/10.1137/0218053.
4. Siu Man Chan, Massimo Lauria, Jakob Nordström, and Marc Vinyals. Hardness of approximation in PSPACE and separation results for pebble games. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 466–485, 2015. Available from: http://dx.doi.org/10.1109/FOCS.2015.36.
5. Stephen Cook and Ravi Sethi. Storage requirements for deterministic / polynomial time recognizable languages. In *Proceedings of the Sixth Annual ACM Symposium on Theory of Computing*, STOC '74, pages 33–39, New York, NY, USA, 1974. ACM. Available from: http://doi.acm.org/10.1145/800119.803882.
6. Erik D. Demaine and Quanquan C. Liu. Inapproximability of the standard pebble game and hard to pebble graphs. *CoRR*, 2017.
7. Peter Emde Boas and Jan Leeuwen. *Theoretical Computer Science 4th GI Conference: Aachen, March 26–28, 1979*, chapter Move rules and trade-offs in the pebble game, pages 101–112. Springer Berlin Heidelberg, Berlin, Heidelberg, 1979. Available from: http://dx.doi.org/10.1007/3-540-09118-1_12.
8. John R. Gilbert, Thomas Lengauer, and Robert Endre Tarjan. The pebbling problem is complete in polynomial space. In *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC '79, pages 237–248, New York, NY, USA, 1979. ACM. Available from: http://doi.acm.org/10.1145/800135.804418.
9. John R. Gilbert and Robert E Tarjan. Variations of a pebble game on graphs. Technical report, Stanford, CA, USA, 1978.
10. Philipp Hertel and Toniann Pitassi. The PSPACE-completeness of black-white pebbling. *SIAM J. Comput.*, 39(6):2622–2682, April 2010. Available from: http://dx.doi.org/10.1137/080713513.
11. John Hopcroft, Wolfgang Paul, and Leslie Valiant. On time versus space. *J. ACM*, 24(2):332–337, April 1977. Available from: http://doi.acm.org/10.1145/322003.322015.
12. Hong Jia-Wei and H. T. Kung. I/O complexity: The red-blue pebble game. In *Proceedings of the Thirteenth Annual ACM Symposium on Theory of Computing*, STOC '81, pages 326–333, New York, NY, USA, 1981. ACM. Available from: http://doi.acm.org/10.1145/800076.802486.
13. Thomas Lengauer and Robert Endre Tarjan. Upper and lower bounds on time-space tradeoffs. In *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC '79, pages 262–277, New York, NY, USA, 1979. ACM. Available from: http://doi.acm.org/10.1145/800135.804420.
14. Jakob Nordstrom. New wine into old wineskins: A survey of some pebbling classics with supplemental results. 2015.
15. Wolfgang J. Paul, Robert Endre Tarjan, and James R. Celoni. Space bounds for a game on graphs. In *Proceedings of the Eighth Annual ACM Symposium on Theory of Computing*, STOC '76, pages 149–160, New York, NY, USA, 1976. ACM. Available from: http://doi.acm.org/10.1145/800113.803643.
16. Ravi Sethi. Complete register allocation problems. *SIAM J. Comput.*, 4(3):226–248, 1975. Available from: http://dx.doi.org/10.1137/0204020.