Theoretical Informatics and Applications Will be set by the publisher
Informatique Théorique et Applications

TIME AND SPACE COMPLEXITY OF REVERSIBLE PEBBLING *

Richard Královič[1]

Abstract. This paper investigates one possible model of reversible computations, an important paradigm in the context of quantum computing. Introduced by Bennett, a reversible pebble game is an abstraction of reversible computation that allows to examine the space and time complexity of various classes of problems. We present a technique for proving lower and upper bounds on time and space complexity for several types of graphs. Using this technique we show that the time needed to achieve optimal space for chain topology is $\Omega(n \lg n)$ for infinitely many n and we discuss time-space trade-offs for chain. Further we show a tight optimal space bound for the binary tree of height h of the form $h + \Theta(\lg^* h)$ and discuss space complexity for the butterfly. These results give an evidence that reversible computations need more resources than standard computations. We also show an upper bound on time and space complexity of the reversible pebble game based on the time and space complexity of the standard pebble game, regardless of the topology of the graph.

1991 Mathematics Subject Classification. 68Q10, 68Q25.

Keywords and phrases: Reversible computations, pebbling.

* ***SUPPORTED IN PART BY THE SLOVAK GRANT AGENCY GRANT VEGA 1/0131/03.***

[1] Department of Computer Science
Faculty of Mathematics, Physics and Informatics
Comenius University
Mlynská Dolina
842 48 Bratislava
Slovakia
e-mail: rkralovi@dcs.fmph.uniba.sk

1. Introduction

The standard pebble game was introduced as a graph-theoretic model for analysing time-space complexity of deterministic computations. In this model, values to be computed are represented by vertices of a directed acyclic graph. An edge from a vertex b to a vertex a represents the fact that for computing the value a, the value b has to be already known. Computation is modelled by placing and removing pebbles representing memory locations on/from the vertices. A pebble placed on a certain vertex expresses the fact that the value of this vertex is already computed and stored in the memory.

The pebble game is important for problems that can be represented by acyclic graphs. Then, in order to get a time-efficient space-restricted computation it is often useful to study the time-space complexity of the pebbling game on the corresponding class of graphs.

Various modifications of the pebbling game have been studied in connection with different models of computations (pebble game with black and white pebbles for nondeterministic computations, pebble game of two players for alternating computations, pebble game with red and blue pebbles for input-output complexity analysis, pebble game with labels for database serializability testing, etc., see [8]). The standard pebble game played on dynamic graphs was studied as a model of incremental computations (see [9] and [10]).

The model of reversible computations is interesting in connection with quantum computing. Since the basic laws of quantum physics are reversible, the quantum computation has to be reversible, too. That means that each state of the computation has to uniquely define both the *following* and the *preceding* state of the computation.

Another motivation for studying the model of reversible computation follows from the fact that reversible operations are not known to require any heat dissipation. With continuing miniaturisation of computing devices, reduction of the energy dissipation becomes very important. Both these reasons for studying reversible computations are mentioned in [2], [5], [6] and [11].

Reversible pebble game is a modification of the standard pebble game for modelling reversible computations that enables to analyse time and space complexity and time-space trade-offs of reversible computations. It was introduced by Charles H. Bennett in [1] and further analysed e.g. in [4], [5] and [12].

In this paper three basic classes of directed acyclic graphs (dags) are considered: the chain, the complete binary tree and the butterfly. These topologies represent the structure of many important problems. For example simple sequential computation of maximum of an array can be described by the chain topology and recursive "divide and conquer" algorithms (e.g. recursive mergesort) can be described by the complete binary tree topology.

It is evident that minimal space complexity of the standard pebble game on a chain of length n is $O(1)$, minimal time complexity is $O(n)$ and minimal space and time complexities can both be achieved simultaneously. Concerning the reversible pebble game, it was proved in [5] that the minimal space complexity on the

chain topology is $O(\lg n)$ and the time complexity for optimal space complexity is $O(n^{\lg 3})$.

We prove a lower bound on time complexity of computations using minimal space, stating that the optimal time complexity in this case cannot be $o(n \lg n)$ i.e. it has to be $\Omega(n \lg n)$ for infinitely many n.

As proven in [7], minimal space complexity for standard pebble game on a complete binary tree of height h is $h + 1$. We show a tight space bound for reversible pebble game on a complete binary tree of the form $h + \Theta(\lg^* h)$. These results give an evidence that more resources are needed for reversible computation in comparison with irreversible computation.

We also show an upper bound on the time and space complexity of the reversible pebble game based on the complexity of the standard pebble game, regardless of the topology: let time t and space p be sufficient for the standard pebble game on an arbitrary graph G and let time t' and space p' be sufficient for the reversible pebble game on a chain of length t. Then time t' and space $p \cdot p'$ are sufficient for the reversible pebble game on G. This result is a generalisation of results from [1] about reversible simulation of irreversible computations, expressed in terms of the reversible pebble game.

2. Preliminaries

The Reversible Pebble Game is played on directed acyclic graphs. Let G be a dag. A *configuration on* G is a set of vertices covered by pebbles. For a configuration C, the formula $C(v) = 1$ denotes the fact that in C the vertex v is covered by a pebble. Similarly, $C(v) = 0$ means the vertex v is uncovered. We denote the number of pebbles used in a configuration C by $\#(C)$. An *empty configuration* on G is denoted by $\mathrm{E}(G)$. Empty configuration is a configuration without pebbles. The rules of the *Reversible pebble game* are the following:

R1: A pebble can be placed on a vertex v if and only if all direct predecessors of the vertex v are covered by pebbles.

R2: A pebble can be removed from a vertex v if and only if all direct predecessors of the vertex v are covered by pebbles.

The reversible pebble game differs from the standard pebble game in the rule R2 – in the standard pebble game, pebbles can be removed from any vertex at any time.

An ordered pair of configurations on a dag G such that the second one follows from the first one by applying one of these rules is called a *transition*.

For our purposes a transition can be also a pair of two identical configurations. Such a transition is called *trivial*. Configurations that form a nontrivial transition always differ in a state of exactly one vertex.

An important property of a transition in a reversible pebble game is its *symmetry*. From the rules of the game it follows that if (C_1, C_2) forms a transition, then also (C_2, C_1) forms a transition.

A *computation on a graph* G is a sequence of configurations on G such that each successive pair forms a transition. Let $\mathcal{K}$ be a computation then $\mathcal{K}(i)$ denotes the i-th configuration in the computation $\mathcal{K}$. A computation $\mathcal{K}$ is a *complete computation* if and only if the first and the last configurations of $\mathcal{K}$ are empty (i.e. $\#(\mathcal{K}(1)) = \#(\mathcal{K}(n)) = 0$ where n is the length of $\mathcal{K}$) and for each vertex v there exists a configuration C in $\mathcal{K}$ such that v is covered in C. A complete computation on G is an abstraction of a successful solution of the problem represented by G.

We are interested in *space* and *time* complexities of a computation $\mathcal{K}$. *Space* of $\mathcal{K}$ (denoted as $\mathrm{S}(\mathcal{K})$) is the number of pebbles needed to perform the computation – that is the maximum number of pebbles used over all configurations of $\mathcal{K}$. *Time* of a computation $\mathcal{K}$ (denoted as $\mathrm{T}(\mathcal{K})$) is the number of nontrivial transitions in $\mathcal{K}$. Note that $\mathrm{T}(\mathcal{K})$ can be less than the length of $\mathcal{K}$ in case that $\mathcal{K}$ contains trivial transitions. By removing all trivial transitions from $\mathcal{K}$ we obtain a computation of length $1 + \mathrm{T}(\mathcal{K})$.

The *minimal space* of the reversible pebble game on a dag G (denoted as $\mathrm{S}_{\min}(G)$) is the minimum of $\mathrm{S}(\mathcal{K})$ over all complete computations $\mathcal{K}$ on G. The *time* $\mathrm{T}(G, s)$ of the reversible pebble game on the dag G with *at most s pebbles* is the minimum of $\mathrm{T}(\mathcal{K})$ over all complete computations $\mathcal{K}$ on G such that $\mathrm{S}(\mathcal{K}) \leq s$.

Let $\mathcal{G}$ be a class of dags. Let $\{\mathcal{G}_n \mid n \in \mathbb{N}\}$ be a system of subclasses of $\mathcal{G}$, that forms a partition of $\mathcal{G}$. Then the *minimal space function* $\mathrm{S}_{\min}(n)$ of a class $\mathcal{G}$ is the maximum of $\mathrm{S}_{\min}(G)$ over all dags in the subclass $\mathcal{G}_n$. The *time function* $\mathrm{T}(n, s)$ is the maximum of $\mathrm{T}(G, s)$ over all dags G in the subclass $\mathcal{G}_n$.

2.1. Operations on Computations

For proving upper and lower bounds on time and space complexities of the reversible pebble game it is useful to reason formally about reversible computations. In order to do so we develop an algebraic model of computations. In this section we introduce some operations for constructing and modifying computations.

For changing the state of a particular vertex in a configuration we use the operation *Put*. $Put(C, v, 1)$ denotes a configuration obtained from a configuration C by pebbling a vertex v. Similarly $Put(C, v, 0)$ denotes a configuration obtained from C by unpebbling vertex v. See also definition 6.1 in Appendix A.

An important property of reversible computations is the following one: Let G be a dag, G' be a subgraph of G, and $\mathcal{K}$ be a computation on G. If we remove all vertices not in G' from all configurations of $\mathcal{K}$ we obtain a computation on G'. The correctness of such construction is immediate – no rule of reversible pebble game can be violated by removing a vertex from all configurations of a computation. Another important fact is that removing some configurations from the beginning and the end of a reversible computation does not violate any property of a reversible computation, either.

Hence, we can introduce an operator for a "restriction" of a computation: $\mathrm{Rst}(\mathcal{K}, i, j, V')$ denotes a computation obtained from $\mathcal{K}$ by discarding vertices other than those in V' and configurations other than those numbered i to j. We use $\mathrm{Rst}(\mathcal{K}, i, j)$ when no vertices should be removed.

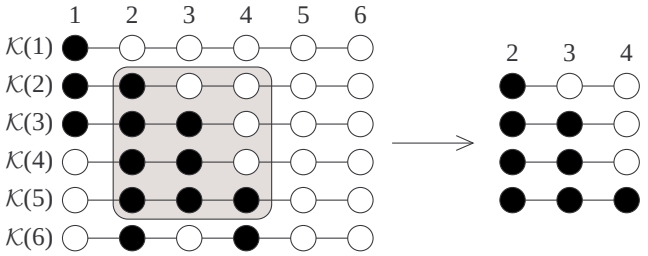

FIGURE 1. An example of a restriction – $\mathrm{Rst}(\mathcal{K}, 2, 5, \{2, 3, 4\})$

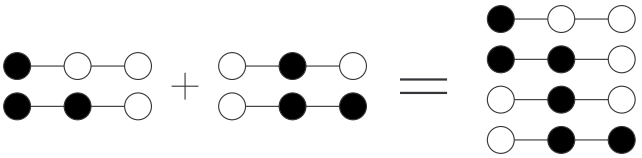

FIGURE 2. An example of a join

For an example of a restriction see Figure 1. A formal definition can be found in Appendix A, Definition 6.2.

It follows from the symmetry of the rules of the reversible pebble game that reversing a reversible computation does not violate the reversible computation property. We can therefore introduce an operator $\mathrm{Rev}(\mathcal{K})$ that reverses the order of configurations in the computation $\mathcal{K}$ (see Definition 6.3).

Now we introduce operations that are in some sense inverse to restriction.

Let $\mathcal{K}_1$ and $\mathcal{K}_2$ be computations on a dag G such that the last configuration of $\mathcal{K}_1$ and the first configuration of $\mathcal{K}_2$ form a transition. Then we can first execute $\mathcal{K}_1$ and subsequently execute $\mathcal{K}_2$. In this way we obtain new computation $\mathcal{K}_1 + \mathcal{K}_2$. This computation will be called *a join* of computations $\mathcal{K}_1$ and $\mathcal{K}_2$ (see Definition 6.4 and Figure 2).

The join of two computations is an inverse operation to restriction by removing configurations. Now we introduce an inverse operation to the restriction performed by removing vertices.

Let V_1, V_2 be disjoint sets of vertices of a dag G. Let C be a configuration on a subgraph of G induced by V_1 and let $\mathcal{K}$ be a computation on a subgraph induced by V_2. We construct a computation on a subgraph of G induced by $V_1 \cup V_2$ by adding C to each configuration in $\mathcal{K}$. This computation will be denoted as $C \cdot \mathcal{K}$ – *a merge* of computation $\mathcal{K}$ with configuration C.

To ensure that the constructed computation is correct we enforce the following constraint: if there is an edge in G from $u \in V_1$ to $v \in V_2$ the vertex u must be pebbled in C.

For an example of merge, see Figure 3; see also Definition 6.5.

Any computation on a graph G can be applied on any graph G' that is isomorphic to G. We will denote a computation $\mathcal{K}$ *applied to* the graph G' as $\mathcal{K}|G'$. See also Definition 6.6.

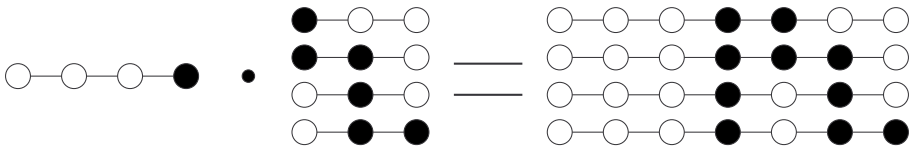

FIGURE 3. An example of a merge

3. Chain Topology

The simplest topology for a pebble game is a *chain*. A chain with n vertices (denoted as $Ch(n)$) is a dag $Ch(n) = (V, E)$ where $V = \{1 \dots n\}$ and $E = \{(i-1, i) | i \in \{2 \dots n\}\}$. This topology is an abstraction of a simple straightforward computation where the result of the step $n+1$ can be computed from the result of the step n alone.

In this section we discuss the optimal space complexity of the reversible pebble game on the chain topology – the minimal space function $\mathrm{S}_{\min}(n)$ for Ch, where the subclass Ch_n contains only a chain $Ch(n)$. We will also discuss lower and upper bounds for the optimal time complexity of the space optimal pebbling – the time function $\mathrm{T}(n, \mathrm{S}_{\min}(n))$ and an upper bound on the time-space tradeoff for the chain topology.

3.1. Optimal Space for the Chain Topology

For determining the space complexity of the reversible pebble game on the chain topology we examine the maximum length of the chain that can be pebbled by p pebbles. We denote this length by $\mathrm{S}^{-1}(p)$. It holds that $\mathrm{S}^{-1}(p) = \max\{m \mid \exists \mathcal{K} \in \mathbb{C}_{Ch(m)} : \mathrm{S}(\mathcal{K}) \leq p\}$ where $\mathbb{C}_{Ch(m)}$ is the set of all complete computations on $Ch(m)$.

The reversible pebble game on the chain topology was studied in connection with reversible simulation of irreversible computations. C. H. Bennett suggested in [1] a pebbling strategy whose special case has space complexity $\Theta(\lg n)$. The space optimality of this algorithm was proved in [4], [5] and [6]. This result is cited in the following theorem.

Theorem 3.1. *It holds that* $\mathrm{S}^{-1}(p) = 2^p - 1$. *Therefore for minimal space function of the chain topology* $\mathrm{S}_{\min}(n)$ *it holds*

$$\mathrm{S}_{\min}(n) = \Theta(\lg n)$$

3.2. Optimal Time and Space for the Chain Topology

To show that the optimal time and the optimal space complexity cannot be achieved simultaneously we examine the following question: Given the minimal number of pebbles needed to perform the complete computation, what is the minimal time needed to perform it? This minimal time for space optimal reversible computation is described by the function $\mathrm{T}(n, \mathrm{S}_{\min}(n))$.

Upper bound on time for the space optimal reversible computation on a chain topology follows from the results presented in [5]. As for lower bounds, we prove that for infinitely many n it holds $\mathrm{T}(n, \mathrm{S}_{\min}(n)) = \Omega(n \lg n)$. Hence the function $\mathrm{T}(n, \mathrm{S}_{\min}(n))$ must be larger than the optimal time $\mathrm{T}(n)$, and therefore it makes sense to discuss tradeoffs between time and space complexity.

An important fact is that any time optimal computation must be symmetric. The time of its subcomputation to the point where the vertex with the maximal number is pebbled is half of the time of the whole computation.

Lemma 3.2. *Let $\mathcal{K}$ be a complete computation of length l on Ch_n, $\mathrm{S}(\mathcal{K}) = \mathrm{S}_{\min}(n)$, $\mathrm{T}(\mathcal{K}) = \mathrm{T}(n, \mathrm{S}_{\min}(n))$. Let $i = \min\{i | i \in \{1 \dots l\} \wedge \mathcal{K}(i)(n) = 1\}$. Then it holds that*

$$\mathrm{T}(\mathrm{Rst}(\mathcal{K}, 1, i)) = \mathrm{T}(\mathrm{Rst}(\mathcal{K}, i, l)) = \frac{\mathrm{T}(\mathcal{K})}{2}$$

This lemma obviously follows from the reversibility of $\mathcal{K}$. A detailed proof is in Appendix B.

We shall need another auxiliary lemma: Consider a space optimal complete computation on $Ch_{\mathrm{S}^{-1}(p+1)}$ (note that it uses exactly $p+1$ pebbles). For each configuration in this computation consider the position of the first pebbled vertex. Maximum among these values is $\mathrm{S}^{-1}(p) + 1$.

Lemma 3.3. *Let $n = \mathrm{S}^{-1}(p+1)$. Let $\mathcal{K}$ be a complete computation on Ch_n of length l such that $\mathrm{S}(\mathcal{K}) = p + 1$. It holds that*

$$\max\{\min\{j | j \in \{1 \dots n\} \wedge \mathcal{K}(i)(j) = 1\} | i \in \{1 \dots l\}\} = \mathrm{S}^{-1}(p) + 1$$

This lemma follows from Corollary 3 of [4]. An alternative proof can be found in Appendix B.

Now we prove a lower bound on time for the space optimal computation. We consider only chains of length $\mathrm{S}^{-1}(p)$. A chain has length $\mathrm{S}^{-1}(p)$ for some p if and only if it can be pebbled by p pebbles but each longer chain requires more than p pebbles. We prove a recurrent inequality that gives a lower bound for $\mathrm{T}(\mathrm{S}^{-1}(p), p)$:

Theorem 3.4. $\mathrm{T}(\mathrm{S}^{-1}(p+1), p+1) \geq 2\mathrm{S}^{-1}(p) + 2 + 2\mathrm{T}(\mathrm{S}^{-1}(p), p)$

Proof. Let $\mathcal{K}$ be a time optimal complete computation on $Ch_{\mathrm{S}^{-1}(p+1)}$ such that $\mathrm{S}(\mathcal{K}) = p + 1$. Clearly $\mathrm{T}(\mathcal{K}) = \mathrm{T}(\mathrm{S}^{-1}(p+1), p+1)$. We prove that $\mathrm{T}(\mathcal{K}) \geq 2\mathrm{S}^{-1}(p) + 2 + 2\mathrm{T}(\mathrm{S}^{-1}(p), p)$.

Let l be the length of $\mathcal{K}$ and $n = \mathrm{S}^{-1}(p)$. According to Theorem 3.1 it holds $\mathrm{S}^{-1}(p+1) = 2n + 1$. We define i as follows:

$$i = \min\{i | i \in \{1 \dots l\} \wedge \mathcal{K}(i)(2n+1) = 1\}$$

By Lemma 3.2 it holds $\mathrm{T}(\mathrm{Rst}(\mathcal{K}, 1, i)) = \frac{1}{2}\mathrm{T}(\mathcal{K})$. We prove that the time needed for the first half of $\mathcal{K}$ is at least $n + 1 + \mathrm{T}(n, p)$. The main idea of the proof is to partition the computation into parts such that each transition in the computation

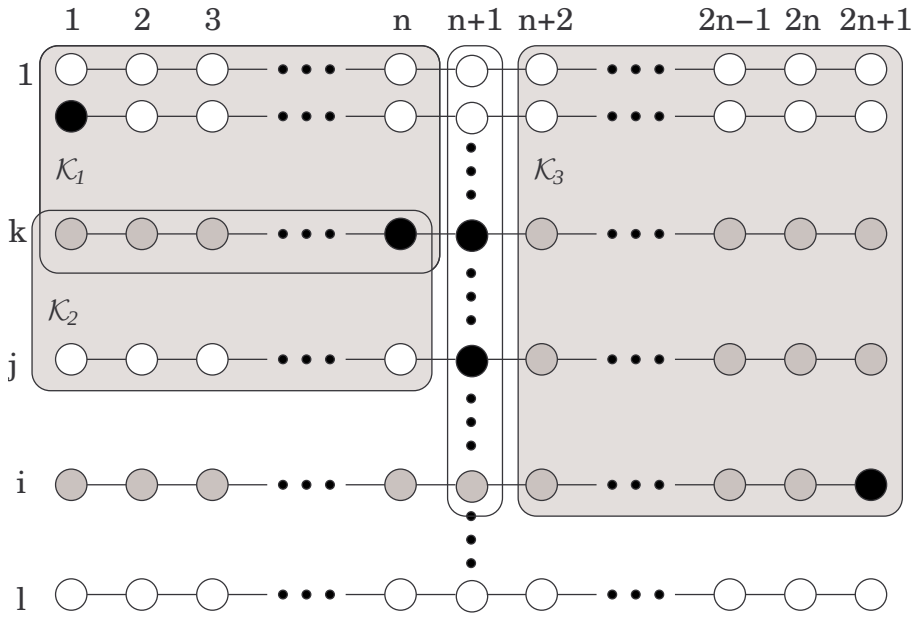

FIGURE 4. Computations $\mathcal{K}$, $\mathcal{K}_1$, $\mathcal{K}_2$ and $\mathcal{K}_3$. Gray circles are vertices with unknown state.

occurs in at most one part. The time of the computation then has to be at least the sum of the times of the parts.

From Lemma 3.3 it follows that in $\mathcal{K}(k)$ at least one of the first $n+1$ vertices is pebbled for each k and that there exists j such that in $\mathcal{K}(j)$ the first pebbled vertex is the vertex number $n+1$. We can assume w.l.o.g. that $j \leq i$ (otherwise we can replace $\mathcal{K}$ by $\mathrm{Rev}(\mathcal{K})$). Let k be a configuration such that $\mathcal{K}(k-1)(n+1) = 0$ and $(\forall q : k \leq q \leq j)\mathcal{K}(q)(n+1) = 1$. Clearly $\mathcal{K}(k-1)(n) = \mathcal{K}(k)(n) = 1$.

Let $\mathcal{K}_1 = \mathrm{Rst}(\mathcal{K}, 1, k, \{1 \ldots n\})$, $\mathcal{K}_2 = \mathrm{Rst}(\mathcal{K}, k, j, \{1 \ldots n\})$ and $\mathcal{K}_3 = \mathrm{Rst}(\mathcal{K}, 1, i, \{n+2 \ldots 2n+1\})$. See Figure 4.

It holds $\mathrm{T}(\mathrm{Rst}(\mathcal{K}, 1, i)) = \mathrm{T}(\mathrm{Rst}(\mathcal{K}, 1, i, \{1 \ldots n\})) + \mathrm{T}(\mathrm{Rst}(\mathcal{K}, 1, i, \{n+1\})) + \mathrm{T}(\mathcal{K}_3)$, because each nontrivial transition has to occur in exactly one of the three parts of $\mathcal{K}$. Analogically it holds that $\mathrm{T}(\mathrm{Rst}(\mathcal{K}, 1, i, \{1 \ldots n\}) = \mathrm{T}(\mathcal{K}_1) + \mathrm{T}(\mathcal{K}_2) + \mathrm{T}(\mathrm{Rst}(\mathcal{K}, j, i, \{1 \ldots n\}))$. Therefore

$$\mathrm{T}(\mathrm{Rst}(\mathcal{K}, 1, i)) \geq \mathrm{T}(\mathcal{K}_1) + \mathrm{T}(\mathcal{K}_2) + \mathrm{T}(\mathcal{K}_3) + \mathrm{T}(\mathrm{Rst}(\mathcal{K}, 1, i, \{n+1\}))$$

Trivially, $\mathrm{T}(\mathrm{Rst}(\mathcal{K}, 1, i, \{n+1\})) \geq 1$ and $\mathrm{T}(\mathcal{K}_1) \geq n$.

$\mathrm{Rev}(\mathcal{K}_2) + \mathcal{K}_2$ is a complete computation on Ch_n. Since $(\forall q : k \leq q \leq j)$ $\mathcal{K}(q)(n+1) = 1$, it holds that $\mathrm{S}(\mathrm{Rev}(\mathcal{K}_2) + \mathcal{K}_2) = p$. Therefore

$$2\mathrm{T}(\mathcal{K}_2) = \mathrm{T}(\mathrm{Rev}(\mathcal{K}_2) + \mathcal{K}_2) \geq \mathrm{T}(n, p)$$

Similarly, $\mathcal{K}_3 + \mathrm{Rev}(\mathcal{K}_3)$ is a complete computation on a graph isomorphic to Ch_n. Due to Lemma 3.3, $\mathrm{S}(\mathcal{K}_3 + \mathrm{Rev}(\mathcal{K}_3)) = p$. Hence we have

$$2\mathrm{T}(\mathcal{K}_3) = \mathrm{T}(\mathcal{K}_3 + \mathrm{Rev}(\mathcal{K}_3)) \geq \mathrm{T}(n, p)$$

Putting everything together yields

$$\mathrm{T}(\mathrm{Rst}(\mathcal{K}, 1, i, \{1 \ldots n\}) \geq 1 + n + \mathrm{T}(n, p)$$

Thus, $\mathrm{T}(\mathcal{K}) \geq 2\mathrm{S}^{-1}(p) + 2 + 2\mathrm{T}(\mathrm{S}^{-1}(p), p)$. □

Corollary 3.5. $\mathrm{T}(n, \mathrm{S}_{\min}(n)) = O(n^{\log_2 3})$, $\mathrm{T}(n, \mathrm{S}_{\min}(n)) \neq o(n \lg n)$,

Proof. The upper bound was presented in [5]. By solving the recurrence proved in the preceding theorem we obtain that $T(n, \mathrm{S}_{\min}(n)) = \Omega(n \lg n)$ for $n = 2^p - 1$. Since this function is a restriction of $T(n, \mathrm{S}_{\min}(n))$, the function $T(n, \mathrm{S}_{\min}(n))$ cannot be[1] $o(n \lg n)$. □

3.3. Upper Bound on Time-Space Tradeoff for Chain Topology

In the previous section the time complexity of a reversible pebbling for space optimal computations was analysed. Now we discuss the time complexity for computations that are not space optimal.

Bennett's pebbling strategy introduced in [1] can be used to derive upper bounds on time-space tradeoff for chain topology. This strategy pebbles chain of length ml^k in time $m(2l-1)^k$ using $m + k(l-1)$ pebbles. Choosing $m = 1$ and $k = \lg_l n$ for a fixed l we obtain a tradeoff in the following form: space $O(\lg n)$ versus time $O(n^{\frac{\lg(2l-1)}{\lg l}})$. Choosing $m = 1$ and $l = \sqrt[k]{n}$ for a fixed k we obtain a tradeoff in the form: space $O(\sqrt[k]{n})$ versus time $O(n)$. See also [3] where these results are formulated in terms of reversible simulation of irreversible computation with the emphasis on expressing constant factors of the asymptotic terms.

As an example of our proof technique we present an alternative proof of the second tradeoff.

It is obvious that for any complete computation $\mathcal{K}$ on Ch_n it holds $\mathrm{T}(\mathcal{K}) \geq 2n$, because each vertex has to be pebbled at least once and unpebbled at least once. So it is easy to see that the space of a complete computation $\mathcal{K}$ such that $\mathrm{T}(\mathcal{K}) = 2n$ is exactly n.

Now we will analyse the space complexity of complete computations on Ch_n that are running in time at most $c \cdot n$. We denote $\mathrm{S}^{-1}(c, p)$ the maximal length of a chain, that can be pebbled by p pebbles in time at most c times its length. Formally, $\mathrm{S}^{-1}(c, p) = \max\{n | \exists \mathcal{K} \in \mathbb{C}_{Ch(n)}, \mathrm{S}(\mathcal{K}) \leq p \wedge \mathrm{T}(\mathcal{K}) \leq cn\}$ where $\mathbb{C}_{Ch(n)}$ is the set of all complete computations on Ch_n.

Theorem 3.6. *It holds that* $\mathrm{S}^{-1}(2^k, p) \geq \binom{p}{k}$.

Proof. We prove the statement by induction on p. The base case $\mathrm{S}^{-1}(2^k, 1) \geq \binom{1}{k}$ holds trivially.

For $k = 1$ the inequality $\mathrm{S}^{-1}(2^1, p) \geq \binom{p}{1}$ holds. (It is easy to make a complete computation $\mathcal{K}$ on Ch_p satisfying $\mathrm{S}(\mathcal{K}) = p$ and $\mathrm{T}(\mathcal{K}) = 2p$.)

[1] Note that the function $T(n, \mathrm{S}_{\min}(n))$ is not monotonic, therefore we cannot formulate the result in a form $T(n, \mathrm{S}_{\min}(n)) = \Omega(n \lg n)$. We can only state that $T(n, \mathrm{S}_{\min}(n)) = \Omega(n \lg n)$ holds for infinitely many n.

Let $p > 1$ and $k > 1$. Let us assume by induction that $\mathrm{S}^{-1}(2^{k-1}, p-1) \geq \binom{p-1}{k-1}$ and $\mathrm{S}^{-1}(2^k, p-1) \geq \binom{p-1}{k}$. We prove that $\mathrm{S}^{-1}(2^k, p) \geq \binom{p}{k}$.

We construct a complete computation $\mathcal{K}$ on a chain of length $\mathrm{S}^{-1}(2^{k-1}, p-1) + 1 + \mathrm{S}^{-1}(2^k, p-1)$ as follows: pebble $\mathrm{S}^{-1}(2^{k-1}, p-1)$ vertices using $p-1$ pebbles in time $2^{k-1}\mathrm{S}^{-1}(2^{k-1}, p-1)$ and put the p-th pebble on the vertex number $\mathrm{S}^{-1}(2^{k-1}, p-1) + 1$. Then pebble next $\mathrm{S}^{-1}(2^k, p-1)$ vertices using p pebbles in time $2^k\mathrm{S}^{-1}(2^k, p-1)$. Finally, unpebble first $\mathrm{S}^{-1}(2^{k-1}, p-1)$ and remove the p-th pebble.

To formally describe this computation, let $\mathcal{K}_1$ be a complete computation on $Ch_{\mathrm{S}^{-1}(2^{k-1}, p-1)}$ such that $\mathrm{S}(\mathcal{K}_1) \leq p-1$ and $\mathrm{T}(\mathcal{K}_1) \leq 2^{k-1}\mathrm{S}^{-1}(2^{k-1}, p-1)$. Let us denote the length of $\mathcal{K}_1$ by l_1. Let i be such that $\mathcal{K}_1(i)(\mathrm{S}^{-1}(2^{k-1}, p-1)) = 1$. Let $\mathcal{K}_2$ be a complete computation on $Ch_{\mathrm{S}^{-1}(2^k, p-1)}$ such that $\mathrm{S}(\mathcal{K}_2) \leq p-1$ and $\mathrm{T}(\mathcal{K}_2) \leq 2^k\mathrm{S}^{-1}(2^k, p-1)$.

Let G_1 be a graph $(\{\mathrm{S}^{-1}(2^{k-1}, p-1)+1\}, \emptyset)$. Let G_2 be a graph obtained from $Ch_{\mathrm{S}^{-1}(2^k, p-1)}$ by renaming its vertices to $\mathrm{S}^{-1}(2^{k-1}, p-1) + 2, \ldots, \mathrm{S}^{-1}(2^{k-1}, p-1) + 1 + \mathrm{S}^{-1}(2^k, p-1)$.

The computation $\mathcal{K}$ can be defined as follows (see also Figure 5):

$$\begin{aligned}
\mathcal{K} = {} & \mathrm{Rst}(\mathcal{K}_1, 1, i) \cdot \mathrm{E}(G_1) \cdot \mathrm{E}(G_2) + \\
& + \mathrm{Rst}(\mathcal{K}_1, i, l_1) \cdot \mathrm{Put}(\mathrm{E}(G_1), \mathrm{S}^{-1}(2^{k-1}, p-1) + 1, 1) \cdot \mathrm{E}(G_2) + \\
& + (\mathcal{K}_2 | G_2) \cdot \mathrm{E}(Ch_{\mathrm{S}^{-1}(2^{k-1}, p-1)}) \cdot \mathrm{Put}(\mathrm{E}(G_1), \mathrm{S}^{-1}(2^{k-1}, p-1) + 1, 1) + \\
& + \mathrm{Rev}(\mathrm{Rst}(\mathcal{K}_1, i, l_1)) \cdot \mathrm{Put}(\mathrm{E}(G_1), \mathrm{S}^{-1}(2^{k-1}, p-1) + 1, 1) \cdot \mathrm{E}(G_2) + \\
& + \mathrm{Rev}(\mathrm{Rst}(\mathcal{K}_1, 1, i)) \cdot \mathrm{E}(G_1) \cdot \mathrm{E}(G_2)
\end{aligned}$$

Clearly $\mathcal{K}$ is a complete computation on $Ch_{\mathrm{S}^{-1}(2^{k-1}, p-1)+1+\mathrm{S}^{-1}(2^k, p-1)}$, $\mathrm{S}(\mathcal{K}) \leq p$ and $\mathrm{T}(\mathcal{K}) \leq 2\mathrm{T}(\mathcal{K}_1) + 2 + \mathrm{T}(\mathcal{K}_2) \leq 2^k\mathrm{S}^{-1}(2^{k-1}, p-1) + 2 + 2^k\mathrm{S}^{-1}(2^k, p-1) \leq 2^k(\mathrm{S}^{-1}(2^{k-1}, p-1) + 1 + \mathrm{S}^{-1}(2^k, p-1))$. Therefore $\mathrm{S}^{-1}(2^k, p) \geq \mathrm{S}^{-1}(2^{k-1}, p-1) + 1 + \mathrm{S}^{-1}(2^k, p-1)$.

By applying the induction hypothesis we have $\mathrm{S}^{-1}(2^k, p) \geq \binom{p-1}{k-1} + \binom{p-1}{k} = \binom{p}{k}$. □

Corollary 3.7. *Let k be fixed. $O(\sqrt[k]{n})$ pebbles are sufficient for a complete time $O(n)$ computation on Ch_n.*

4. Binary Tree Topology

In this section we shall discuss the space complexity of the reversible pebble game on complete binary trees. A *complete binary tree* of height 1 (denoted by $\mathrm{Bt}(1)$) is a graph containing one vertex and no edges. A complete binary tree of height $h > 1$ (denoted as $\mathrm{Bt}(h)$) consists of a root vertex and two subtrees, that are complete binary trees of height $h-1$.

This topology represents a class of problems, where the result can be computed from two different subproblems.

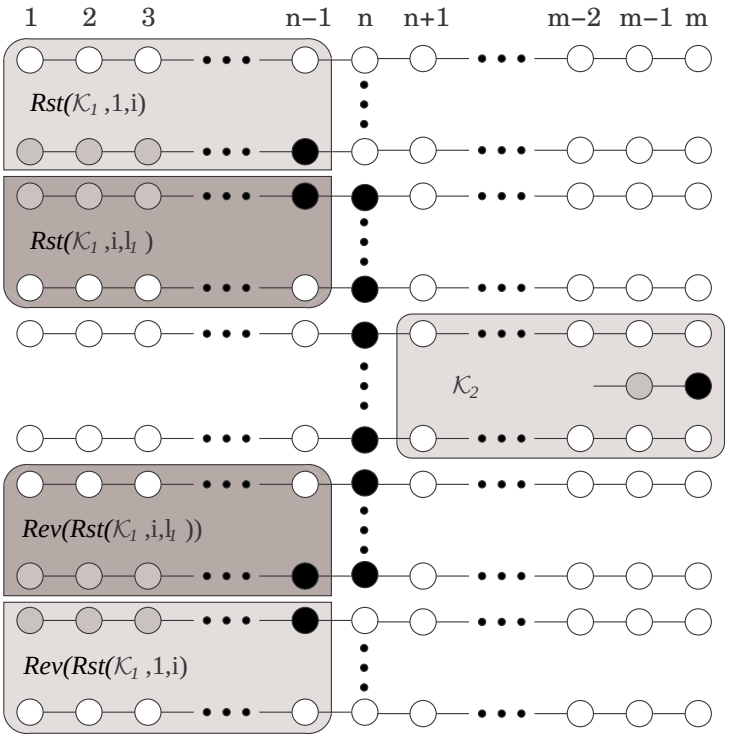

FIGURE 5. A computation $\mathcal{K}$, where $n = \mathrm{S}^{-1}(2^{k-1}, p-1)+1$ and $m = \mathrm{S}^{-1}(2^{k-1}, p-1) + 1 + \mathrm{S}^{-1}(2^k, p-1)$.

Let T be a complete binary tree. We denote the root vertex of T as $\mathrm{R}(T)$, the left subtree of T as $\mathrm{Lt}(T)$ and the right subtree of T as $\mathrm{Rt}(T)$.

As mentioned in Section 2, we denote the minimal number of pebbles needed to perform a complete computation on $\mathrm{Bt}(h)$ by $\mathrm{S}_{\min}(h)$. To simplify some of the proofs we also consider the minimal number of pebbles needed to perform a computation from the empty configuration to a configuration where only the root is pebbled.

Definition 4.1. Let $\mathcal{K}$ be a computation of length l on $\mathrm{Bt}(h)$. Let $\mathcal{K}(1) = \mathrm{E}(\mathrm{Bt}(h))$ and $\mathcal{K}(l) = \mathrm{Put}(\mathrm{E}(\mathrm{Bt}(h)), \mathrm{R}(\mathrm{Bt}(h)), 1)$. Then $\mathcal{K}$ is called a *semicomplete computation*.

The minimal number of pebbles needed to perform a semicomplete computation on $\mathrm{Bt}(h)$ (e.g. $\min\{\mathrm{S}(\mathcal{K})\}$, where $\mathcal{K}$ is a semicomplete computation) will be denoted as $\mathrm{S}'_{\min}(h)$.

We shall use the following inequalities between $\mathrm{S}_{\min}(h)$ and $\mathrm{S}'_{\min}(h)$. These inequalities show that the difference between the number of pebbles needed to perform a complete and a semicomplete computation is small. Therefore, we can estimate the space complexity of the binary tree topology by estimating the number of pebbles needed to perform a semicomplete computation.

Lemma 4.2. $\mathrm{S}_{\min}(h) + 1 \geq \mathrm{S}'_{\min}(h) \geq \mathrm{S}_{\min}(h)$

By performing a complete computation without removing a pebble from the root vertex we can obtain a semicomplete computation with space $\mathrm{S}_{\min}(h)+1$, hence $\mathrm{S}_{\min}(h)+1 \geq \mathrm{S}'_{\min}(h)$. By joining a semicomplete computation with its reverse we obtain a semicomplete computation, hence $\mathrm{S}'_{\min}(h) \geq \mathrm{S}_{\min}(h)$. The formal proof can be found in Appendix B.

Lemma 4.3. $\mathrm{S}'_{\min}(h+1) = \mathrm{S}_{\min}(h)+2$

Proof. At first we prove that $\mathrm{S}_{\min}(h)+2 \geq \mathrm{S}'_{\min}(h+1)$. By the previous lemma, $\mathrm{S}_{\min}(h)+1 \geq \mathrm{S}'_{\min}(h)$. Let $\mathcal{K}_1$ be a semicomplete computation on $\mathrm{Bt}(h)$ that uses no more than $\mathrm{S}_{\min}(h)+1$ pebbles. Let $\mathcal{K}_2$ be a complete computation on $\mathrm{Bt}(h)$ of length l that uses no more than $\mathrm{S}_{\min}(h)$ pebbles.

Using the computations $\mathcal{K}_1$ and $\mathcal{K}_2$ we construct a semicomplete computation $\mathcal{K}_3$ on $\mathrm{Bt}(h+1)$ that uses at most $\mathrm{S}_{\min}(h)+2$ pebbles.

Let i be the minimal number such that $\mathcal{K}_2(i)(\mathrm{R}(\mathrm{Bt}(h))) = 1$. We define the computation $\mathcal{K}_3$ as follows (see also Figure 6a):

$$
\begin{aligned}
\mathcal{K}_3 = \quad & \big(\mathcal{K}_1 | \mathrm{Lt}(\mathrm{Bt}(h+1))\big) \cdot \mathrm{E}\big(\mathrm{Rt}(\mathrm{Bt}(h+1)) \cup \mathrm{R}(\mathrm{Bt}(h+1))\big) + \\
& + \big(\mathrm{Rst}(\mathcal{K}_2, 1, i) | \mathrm{Rt}(\mathrm{Bt}(h+1))\big) \cdot \\
& \quad \cdot \mathrm{Put}\Big(\mathrm{E}\big(\mathrm{Lt}(\mathrm{Bt}(h+1)) \cup \mathrm{R}(\mathrm{Bt}(h+1))\big), \mathrm{R}\big(\mathrm{Lt}(\mathrm{Bt}(h+1))\big), 1\Big) + \\
& + \big(\mathrm{Rst}(\mathcal{K}_2, i, l) | \mathrm{Rt}(\mathrm{Bt}(h+1))\big) \cdot \\
& \quad \cdot \mathrm{Put}\Big(\mathrm{E}\big(\mathrm{Lt}(\mathrm{Bt}(h+1))\big), \mathrm{R}\big(\mathrm{Lt}(\mathrm{Bt}(h+1))\big), 1\Big) \cdot \\
& \quad \cdot \mathrm{Put}\Big(\mathrm{E}\big(\mathrm{R}(\mathrm{Bt}(h+1))\big), \mathrm{R}(\mathrm{Bt}(h+1)), 1\Big) + \\
& + \Big(\mathrm{Rst}\big(\mathcal{K}_2, 1, i, \mathrm{Bt}(h) \setminus \mathrm{R}(\mathrm{Bt}(h))\big) \mid \mathrm{Lt}(\mathrm{Bt}(h+1)) \setminus \mathrm{R}\big(\mathrm{Lt}(\mathrm{Bt}(h+1))\big)\Big) \cdot \\
& \quad \cdot \mathrm{Put}\Big(\mathrm{E}\big(\mathrm{R}(\mathrm{Lt}(\mathrm{Bt}(h+1)))\big), \mathrm{R}\big(\mathrm{Lt}(\mathrm{Bt}(h+1))\big), 1\Big) \cdot \\
& \quad \cdot \mathrm{Put}\Big(\mathrm{E}\big(\mathrm{R}(\mathrm{Bt}(h+1)) \cup \mathrm{Rt}(\mathrm{Bt}(h+1))\big), \mathrm{R}(\mathrm{Bt}(h+1)), 1\Big) + \\
& + \big(\mathrm{Rst}(\mathcal{K}_2, i, l) | \mathrm{Lt}(\mathrm{Bt}(h+1))\big) \cdot \\
& \quad \cdot \mathrm{Put}\Big(\mathrm{E}\big(\mathrm{R}(\mathrm{Bt}(h+1)) \cup \mathrm{Rt}(\mathrm{Bt}(h+1))\big), \mathrm{R}(\mathrm{Bt}(h+1)), 1\Big)
\end{aligned}
$$

Clearly, $\mathcal{K}_3$ is a semicomplete computation on $\mathrm{Bt}(h+1)$ and it holds that

$$\mathrm{S}(\mathcal{K}_3) \leq \max(\mathrm{S}(\mathcal{K}_1), \mathrm{S}(\mathcal{K}_2)+2) \leq \mathrm{S}_{\min}(h)+2$$

Therefore the first inequality holds.

Now we prove that $\mathrm{S}_{\min}(h)+2 \leq \mathrm{S}'_{\min}(h+1)$ holds. Let $\mathcal{K}$ be a semicomplete computation of length l on $\mathrm{Bt}(h+1)$, such that $\mathrm{S}(\mathcal{K}) = \mathrm{S}'_{\min}(h+1)$.

We show that a complete computation on $\mathrm{Bt}(h)$ that uses at most $\mathrm{S}'_{\min}(h+1)-2$ pebbles can be obtained by an appropriate restriction of $\mathcal{K}$.

Let i be the minimal number such that $(\forall k : i \leq k \leq l)\mathcal{K}(k)(\mathrm{R}(\mathrm{Bt}(h+1))) = 1$. Let j be the minimal number such that $i \leq j \leq l$ and it holds:

$$\#\Big(\mathrm{Rst}\big(\mathcal{K}, j, j, \mathrm{Lt}(\mathrm{Bt}(h+1))\big)\Big) = 0 \quad \vee \quad \#\Big(\mathrm{Rst}\big(\mathcal{K}, j, j, \mathrm{Rt}(\mathrm{Bt}(h+1))\big)\Big) = 0$$

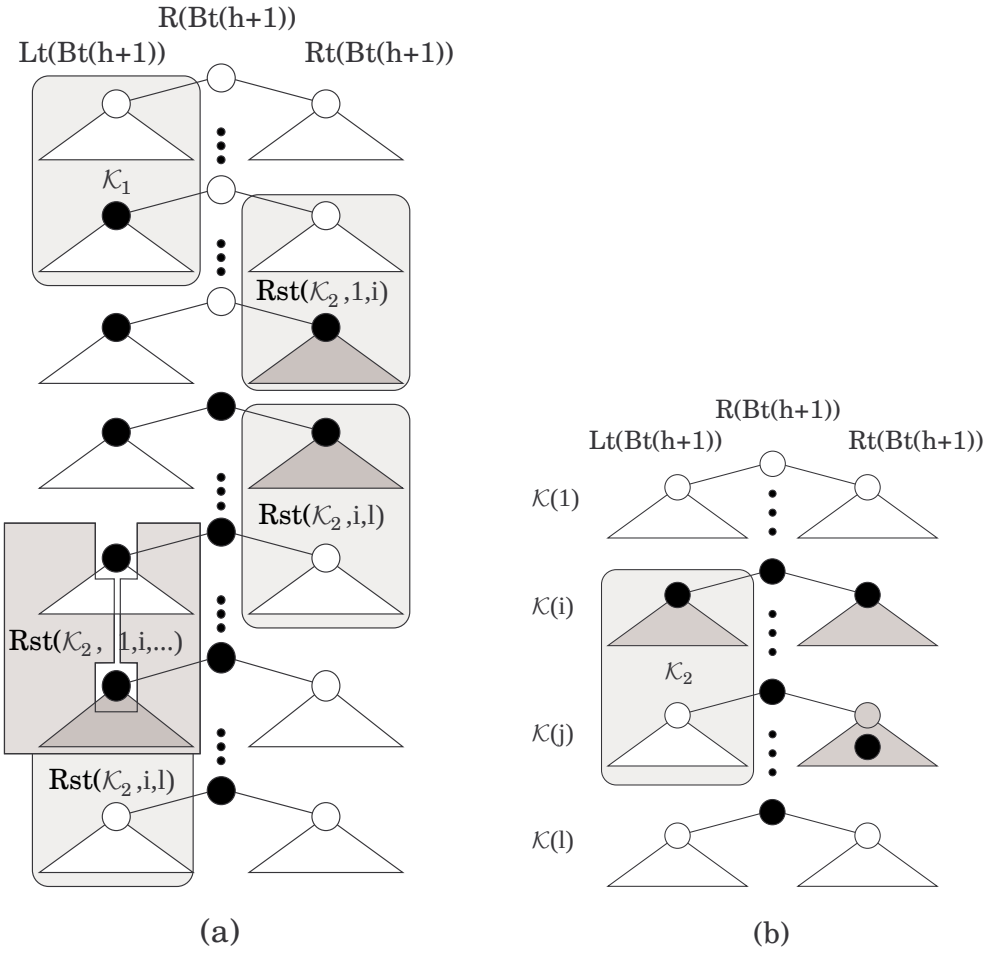

FIGURE 6. a) The proof of $\mathrm{S}_{\min}(h) + 2 \geq \mathrm{S}'_{\min}(h+1)$ – the construction of the computation $\mathcal{K}_3$. b) The proof of $\mathrm{S}_{\min}(h) + 2 \leq \mathrm{S}'_{\min}(h+1)$ – the computation $\mathcal{K}_2$.

We may assume without loss of generality that $\#(\mathrm{Rst}(\mathcal{K}, j, j, \mathrm{Lt}(\mathrm{Bt}(h+1)))) = 0$.

Let us consider the following computation $\mathcal{K}_2$ (see also Figure 6b):

$$\mathcal{K}_2 = \mathrm{Rst}\big(\mathcal{K}, i, j, \mathrm{Lt}(\mathrm{Bt}(h+1))\big)$$

$\mathcal{K}_2$ is a computation on a graph $\mathrm{Rt}(\mathrm{Bt}(h+1))$ which is isomorphic to $\mathrm{Bt}(h)$. Since the last configuration of $\mathcal{K}_2$ is empty and in the first configuration of $\mathcal{K}_2$ the root vertex is pebbled, it holds that $\mathrm{Rev}(\mathcal{K}_2) + \mathcal{K}_2$ is a complete computation on the tree of height h.

In all configurations $\mathcal{K}(k)$ for $i \leq k \leq j$ the root $\mathrm{R}(\mathrm{Bt}(h))$ is pebbled. Also, there is at least one vertex pebbled in $\mathrm{Rt}(\mathrm{Bt}(h))$. Hence, the space of the computation $\mathcal{K}_2$ can not exceed $\mathrm{S}(\mathcal{K}) - 2$. Therefore $\mathrm{S}_{\min}(h) \leq \mathrm{S}(\mathcal{K}_2) \leq \mathrm{S}'_{\min}(h+1) - 2$.

□

As a corollary to the preceding lemmas we obtain the following useful facts:

Corollary 4.4. $\mathrm{S}'_{\min}(h)$ *equals to* h *plus the number of different numbers* i *such that* $i < h$ *and* $\mathrm{S}'_{\min}(i) = \mathrm{S}_{\min}(i)$. *Furthermore, for* $h_1 \geq h_2$ *it holds that* $\mathrm{S}'_{\min}(h_1) - \mathrm{S}'_{\min}(h_2) \geq h_1 - h_2$.

4.1. Tight Space Bound for Binary Tree Topology

In the following considerations we use a function $\mathrm{S}'^{-1}_{\min}$. The value $h = \mathrm{S}'^{-1}_{\min}(p)$ denotes the maximal height of a binary tree that can be pebbled by a semicomplete computation that uses at most $h + p$ pebbles. Formally

$$\mathrm{S}'^{-1}_{\min}(p) = \max\{h | \exists \mathcal{K} \in \ \mathrm{Sc}_h \wedge \mathrm{S}(\mathcal{K}) = h + p\}$$

where Sc_h is the set of all semicomplete computations on $\mathrm{Bt}(h)$.

From the definition of $\mathrm{S}'^{-1}_{\min}(p)$ and Corollary 4.4 we easily obtain the following lemma:

Lemma 4.5. *For each* h, p *such that* $\mathrm{S}'_{\min}(h) = h + p$ *it holds that* $\mathrm{S}'^{-1}_{\min}(p-1) < h \leq \mathrm{S}'^{-1}_{\min}(p)$.

Furthermore let $f(p)$ *be a nondecreasing function such that* $\mathrm{S}'^{-1}_{\min}(p) \leq f(p)$. *Then it holds that* $\mathrm{S}'_{\min}(h) \geq h + f^{-1}(h)$ *for each* h.

Similarly, let $g(p)$ *be a nondecreasing function such that* $\mathrm{S}'^{-1}_{\min}(p) \geq g(p)$. *Then it holds that* $\mathrm{S}'_{\min}(h) \leq h + g^{-1}(h) + 1$ *for each* h.

Now we prove the upper (lower) bound on $\mathrm{S}'^{-1}_{\min}(p)$. From this lemma we obtain a lower (upper) bound on $\mathrm{S}'_{\min}(h)$ and therefore also a lower (upper) bound on $\mathrm{S}_{\min}(h)$.

Lemma 4.6. *Let* $h_1 = \mathrm{S}'^{-1}_{\min}(p)$, $h_2 = \mathrm{S}'^{-1}_{\min}(p+1)$. *Then the following inequality holds:*

$$h_2 - h_1 \leq 2^{h_1 + p + 1}$$

Proof. A configuration on a binary tree is called *open* if there exists a path from the root to some leaf of the tree such that no pebble is placed on this path. Otherwise, the configuration is called *closed*.

From the assumption $h_2 = \mathrm{S}'^{-1}_{\min}(p+1)$ it follows that there exists some semicomplete computation $\mathcal{K}$ of length l on $\mathrm{Bt}(h_2)$ such that $\mathrm{S}(\mathcal{K}) = h_2 + p + 1$ (see Figure 7a). Let i be the first configuration of $\mathcal{K}$ such that the root vertex is pebbled in $\mathcal{K}(j)$ for all $j \geq i$ (i.e. $i = \min\{i' | (\forall j \geq i')\mathcal{K}(j)(\mathrm{R}(\mathrm{Bt}(h_2))) = 1\}$).

Since $\mathcal{K}$ is a reversible computation, it holds that

$$\mathcal{K}(i)\Big(\mathrm{R}\big(\mathrm{Lt}(\mathrm{Bt}(h_2))\big)\Big) = \mathcal{K}(i)\Big(\mathrm{R}\big(\mathrm{Rt}(\mathrm{Bt}(h_2))\big)\Big) = 1$$

Therefore, the configuration obtained from $\mathcal{K}(i)$ by unpebbling the root vertex (denoted as $\mathrm{Put}\big(\mathcal{K}(i), \mathrm{R}(\mathrm{Bt}(h_2)), 0\big)$) is a closed configuration. Since $\mathrm{Put}\big(\mathcal{K}(l), \mathrm{R}(\mathrm{Bt}(h_2)), 0\big)$ is an empty configuration, it is also an open configuration. Let j be the minimal number such that $j \geq i$ and $\mathrm{Put}(\mathcal{K}(j), \mathrm{R}(\mathrm{Bt}(h_2)), 0)$ is open.

Since $\mathrm{Put}(\mathcal{K}(j), \mathrm{R}(\mathrm{Bt}(h_2)), 0)$ is open and $\mathrm{Put}(\mathcal{K}(j-1), \mathrm{R}(\mathrm{Bt}(h)), 0)$ is closed and $\mathcal{K}$ is a reversible computation, there exists exactly one path in $\mathcal{K}(j)$ from the root to a leaf such that no pebble is placed on it. We can assume without loss of generality that this path is $\mathrm{R}(\mathrm{Bt}(h_2)), \mathrm{R}(\mathrm{Rt}(\mathrm{Bt}(h_2))), \mathrm{R}(\mathrm{Rt}^2(\mathrm{Bt}(h_2))), \ldots,$ $\mathrm{R}(\mathrm{Rt}^{h_2-1}(\mathrm{Bt}(h_2)))$. We call this set of vertices the *right path*.

Now we prove that for each k such that $h_2 \geq k \geq h_1 + 2$ and for each r such that $i \leq r < j$ it holds $\#(\mathcal{K}(r)(\mathrm{Lt}(\mathrm{Rt}^{h_2-k}(\mathrm{Bt}(h_2))))) > 0$. This part of the proof is illustrated by Figure 7b. We need this fact to "bind" $h_2 - h_1 - 1$ pebbles so that they can not be used on the vertices of the right path.

Assume, that this conjecture is false and let k be the maximal number violating it. Let r be a maximal number such that $i \leq r < j$ and

$$\#\Big(\mathcal{K}(r)\big(\mathrm{Lt}(\mathrm{Rt}^{h_2-k}(\mathrm{Bt}(h_2)))\big)\Big) = 0 \quad \vee \quad \#\Big(\mathcal{K}(r)\big(\mathrm{Rt}(\mathrm{Rt}^{h_2-k}(\mathrm{Bt}(h_2)))\big)\Big) = 0$$

Our assumption ensures that such r exists. Note that only one clause of the disjunction can be satisfied because $\mathcal{K}$ is a reversible computation. Without loss of generality let $\#\Big(\mathcal{K}(r)\big(\mathrm{Lt}(\mathrm{Rt}^{h_2-k}(\mathrm{Bt}(h_2)))\big)\Big) = 0$. Since $\mathrm{Put}\big(\mathcal{K}(r), \mathrm{R}(\mathrm{Bt}(h_2)), 0\big)$ is closed, at least one vertex from $\mathrm{R}(\mathrm{Rt}(\mathrm{Bt}(h_2)))$, $\mathrm{R}(\mathrm{Rt}^2(\mathrm{Bt}(h_2)))$, …, $\mathrm{R}(\mathrm{Rt}^{h_2-k}($ $\mathrm{Bt}(h_2)))$ is pebbled in the configuration $\mathcal{K}(r)$. All these vertices are unpebbled in the configuration $\mathcal{K}(j)$. Let q be the minimal number such that $q > r$ and all above mentioned vertices are unpebbled in $\mathcal{K}(q)$. Clearly $q \leq j$. Since $\mathcal{K}$ is a reversible computation it holds that

$$\mathcal{K}(q-1)\Big(\mathrm{R}\big(\mathrm{Rt}^{h_2-k}(\mathrm{Bt}(h_2))\big)\Big) = 1$$

$$\mathcal{K}(q)\Big(\mathrm{R}\big(\mathrm{Rt}^{h_2-k}(\mathrm{Bt}(h_2))\big)\Big) = 0$$

$$\mathcal{K}(q-1)\Big(\mathrm{R}\big(\mathrm{Lt}(\mathrm{Rt}^{h_2-k}(\mathrm{Bt}(h_2)))\big)\Big) = \mathcal{K}(q)\Big(\mathrm{R}\big(\mathrm{Lt}(\mathrm{Rt}^{h_2-k}(\mathrm{Bt}(h_2)))\big)\Big) = 1$$

Now consider the computation $\mathcal{K}' = \mathrm{Rst}(\mathcal{K}, r, q-1, \mathrm{Lt}(\mathrm{Rt}^{h_2-k}(\mathrm{Bt}(h_2))))$. Computation $\mathcal{K}' + \mathrm{Rev}(\mathcal{K}')$ is a complete computation on $\mathrm{Lt}(\mathrm{Rt}^{h_2-k}(\mathrm{Bt}(h_2))))$ (this graph is isomorphic to $\mathrm{Bt}(k-1)$). Let us consider the space of this computation. For each $z \in \{r \ldots q-1\}$ in $\mathcal{K}(z)$ the root vertex of $\mathrm{Bt}(h_2)$ is pebbled. By the choice of k, there is at least one vertex pebbled in each of the following subgraphs: $\mathrm{Lt}(\mathrm{Bt}(h_2)), \mathrm{Lt}(\mathrm{Rt}(\mathrm{Bt}(h_2))), \ldots, \mathrm{Lt}(\mathrm{Rt}^{h_2-k-1}(\mathrm{Bt}(h_2)))$. By the choice of q, at least one vertex is pebbled in the upper part of right path – at least one of the following vertices is pebbled: $\mathrm{R}(\mathrm{Rt}(\mathrm{Bt}(h_2))), \ldots, \mathrm{R}(\mathrm{Rt}^{h_2-k}(\mathrm{Bt}(h_2)))$. By the choice of r, there is at least one vertex pebbled in $\mathrm{Rt}^{h_2-k+1}(\mathrm{Bt}(h_2))$. Hence it holds $\mathrm{S}(\mathcal{K}' + \mathrm{Rev}(\mathcal{K}')) = \mathrm{S}(\mathcal{K}') \leq \mathrm{S}(\mathcal{K}) - (3 + h_2 - k) = k + p - 2$.

From our assumption it follows that the space of any semicomplete computation on $\mathrm{Bt}(k-1)$ is at least $k + p$. From Lemma 4.2 it follows that the space of any complete computation on $\mathrm{Bt}(k-1)$ is at least $k + p - 1$, which is a contradiction.

Now consider the computation $\mathcal{K}_2$ (see Figure 7a):

$$\mathcal{K}_2 = \mathrm{Rst}(\mathcal{K}, i, j, \mathrm{R}(\mathrm{Rt}(\mathrm{Bt}(h_2))) \cup \mathrm{R}(\mathrm{Rt}^2(\mathrm{Bt}(h_2))) \cup \ldots \cup \mathrm{R}(\mathrm{Rt}^{h_2-h_1-1}(\mathrm{Bt}(h_2))))$$

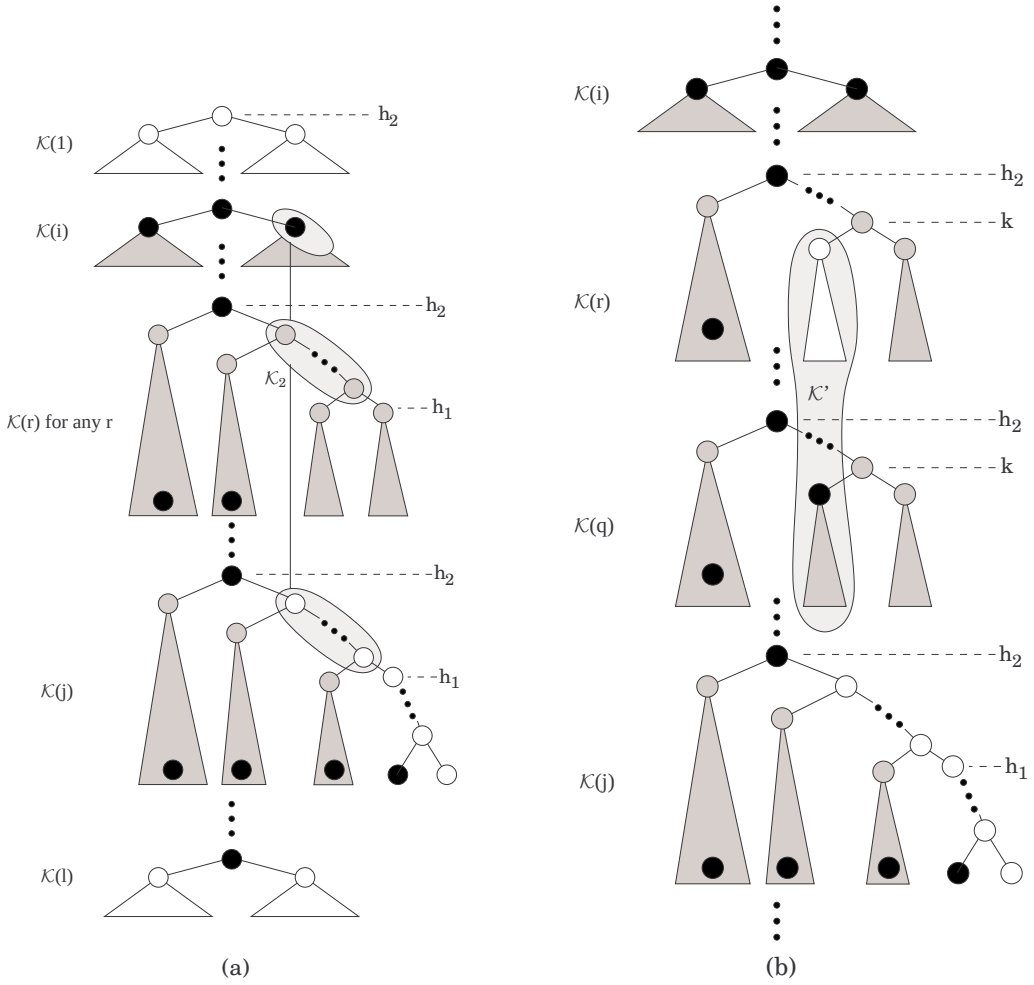

FIGURE 7. a) The main framework of the proof of Lemma 4.6. b) The proof of the auxiliary property by contradiction.

It is a computation on an upper part of the right path, which is graph isomorphic to $Ch(h_2-h_1-1)$. Vertex $\mathrm{R}(\mathrm{Rt}(\mathrm{Bt}(h_2)))$ is pebbled in the first configuration of $\mathcal{K}_2$ and no vertex is pebbled in the last configuration of $\mathcal{K}_2$. Therefore $\mathrm{Rev}(\mathcal{K}_2)+\mathcal{K}_2$ is a complete computation on a graph isomorphic to $Ch(h_2-h_1-1)$.

Since for each $h_2 \geq k \geq h_1+2$ and for each $i \leq r \leq j$ it holds that $\#(\mathcal{K}(r)(\mathrm{Lt}(\mathrm{Rt}^{h_2-k}(\mathrm{Bt}(h_2))))) > 0$ and $\mathcal{K}(r)(\mathrm{R}(\mathrm{Bt}(h))) = 1$, we can estimate upper bound for the space of $\mathcal{K}_2$ to be: $\mathrm{S}(\mathcal{K}_2) \leq (h_2+p+1)-(1+h_2-h_1-1) = h_1+p+1$. Using the upper bound on space of the chain topology (Theorem 3.1) we have $h_2-h_1-1 \leq 2^{h_1+p+1}-1$.

□

We have proven an upper bound on $\mathrm{S}'^{-1}_{\min}(p)$. Now we prove the lower bound:

Lemma 4.7. *Let* $h_1 = \mathrm{S}'^{-1}_{\min}(p)$, $h_2 = \mathrm{S}'^{-1}_{\min}(p+1)$. *Then the following inequality holds:*

$$h_2 - h_1 \geq 2^{h_1+p-2}$$

Proof. We prove by induction that for each $k \in \{h_1 + 1, \ldots, h_1 + 2^{h_1+p-2}\}$ there exists a semicomplete computation $\mathcal{K}$ on $Bt(k)$ such that $\mathrm{S}(\mathcal{K}) \leq k + p + 1$. This implies that $h_2 \geq h_1 + 2^{h_1+p-2}$.

The base case is $k = h_1 + 1$. By assumption there exists a semicomplete computation $\mathcal{K}'$ on $\mathrm{Bt}(h_1)$ such that $\mathrm{S}(\mathcal{K}') = h_1 + p$. After applying $\mathcal{K}'$ to $\mathrm{Lt}(\mathrm{Bt}(k))$ and $\mathrm{Rt}(\mathrm{Bt}(k))$, pebbling $\mathrm{R}(\mathrm{Bt}(k))$ and applying reversed $\mathcal{K}'$ to $\mathrm{Lt}(\mathrm{Bt}(k))$ and $\mathrm{Rt}(\mathrm{Bt}(k))$ we obtain a semicomplete computation on $\mathrm{Bt}(k)$ that uses at most $h_1 + p + 2 = k + p + 1$ pebbles.

Let us assume that the induction hypothesis holds for each $h \in \{h_1 + 1, \ldots, k - 1\}$. We construct a computation $\mathcal{K}$ on $\mathrm{Bt}(k)$ as follows: At first we sequentially apply a space optimal semicomplete computation on subgraphs $\mathrm{Lt}(\mathrm{Bt}(k))$, $\mathrm{Lt}(\mathrm{Rt}(\mathrm{Bt}(k)))$, $\mathrm{Lt}(\mathrm{Rt}^2(\mathrm{Bt}(k)))$, $\ldots$, $\mathrm{Lt}(\mathrm{Rt}^{k-h_1-2}(\mathrm{Bt}(k)))$, $\mathrm{Lt}(\mathrm{Rt}^{k-h_1-1}(\mathrm{Bt}(k)))$ and $\mathrm{Rt}^{k-h_1}(\mathrm{Bt}(k))$. By the induction hypothesis, the space of a semicomplete computation on $\mathrm{Lt}(\mathrm{Rt}^i(\mathrm{Bt}(k)))$ is at most $(k-i-1)+p+1$ for $i \leq k - h_1 - 2$. By the assumption, the space of a semicomplete computation on $\mathrm{Lt}(\mathrm{Rt}^{k-h_1-1}(\mathrm{Bt}(k)))$ and $\mathrm{Rt}^{k-h_1}(\mathrm{Bt}(k))$ is at most $h_1 + p$. Therefore the space of this part of $\mathcal{K}$ is at most $k + p$.

In the second part of $\mathcal{K}$, we perform a space optimal complete computation on a chain consisting of the vertices $\mathrm{R}(\mathrm{Bt}(k))$, $\mathrm{R}(\mathrm{Rt}(\mathrm{Bt}(k)))$, $\mathrm{R}(\mathrm{Rt}^2(\mathrm{Bt}(k)))$, $\ldots$, $\mathrm{R}(\mathrm{Rt}^{k-h_1-1}(\mathrm{Bt}(k)))$. Due to the Theorem 3.1, the space of this part is less than $\lceil \log_2(k-h_1+1) \rceil + k - h_1 + 1 \leq \lceil \log_2(k-h_1) \rceil + k - h_1 + 2$. Since $k \leq h_1 + 2^{h_1+p-2}$, it holds that $\lceil \log_2(k - h_1) \rceil + k - h_1 + 2 \leq k + p$.

The third part of the computation $\mathcal{K}$ is the reversed first part.

Hence $\mathcal{K}$ is a complete computation on $\mathrm{Bt}(k)$ and $\mathrm{S}(\mathcal{K}) \leq k + p$ and therefore $\mathrm{S}_{\min}(k) \leq k + p$. Using Lemma 4.2, $\mathrm{S}'_{\min}(k) \leq k + p + 1$. Therefore there exists a semicomplete computation on $\mathrm{Bt}(k)$ with space less than $k + p + 1$. □

Now we make some approximations and by putting everything together we obtain an asymptotically tight bound for $\mathrm{S}_{\min}(h)$:

Lemma 4.8. *For $p \geq 2$ it holds that $2^{\mathrm{S}'^{-1}_{\min}(p)} \leq \mathrm{S}'^{-1}_{\min}(p+1) \leq 2^{3\,\mathrm{S}'^{-1}_{\min}(p)}$.*

Proof. Let $h_1 = \mathrm{S}'^{-1}_{\min}(p)$, $h_2 = \mathrm{S}'^{-1}_{\min}(p+1)$. From Lemma 4.6 it follows $h_2 - h_1 \leq 2^{h'+p+1}$, which is equivalent to $\mathrm{S}'^{-1}_{\min}(p+1) \leq 2^{\mathrm{S}'^{-1}_{\min}(p)+p+1} + \mathrm{S}'^{-1}_{\min}(p)$.

From the definition of $\mathrm{S}'^{-1}_{\min}$ and from Corollary 4.4 it follows that $\mathrm{S}'^{-1}_{\min}(p) \geq p + 1$. Therefore $2^{\mathrm{S}'^{-1}_{\min}(p)+p+1} + \mathrm{S}'^{-1}_{\min}(p) \leq 2^{3\,\mathrm{S}'^{-1}_{\min}(p)}$ for $p \geq 1$. Hence, the second inequality holds.

From Lemma 4.7 it follows $h_2 - h_1 \geq 2^{h_1+p-2}$, which is equivalent to $\mathrm{S}'^{-1}_{\min}(p+1) \geq 2^{\mathrm{S}'^{-1}_{\min}(p)+p-2} + \mathrm{S}'^{-1}_{\min}(p)$. Therefore for $p \geq 2$ it holds $\mathrm{S}'^{-1}_{\min}(p+1) \geq 2^{\mathrm{S}'^{-1}_{\min}(p)}$. □

Theorem 4.9. *For the minimal space function of the complete binary tree topology $\mathrm{S}_{\min}(h)$ it holds that $\mathrm{S}_{\min}(h) = h + \Theta(\lg^*(h))$, where the function $\lg^*(x)$ is defined as follows: $\lg^*(x) = 0$ for $x \leq 0$, $\lg^*(x) = 1 + \lg^*(\lg(x))$ otherwise.*

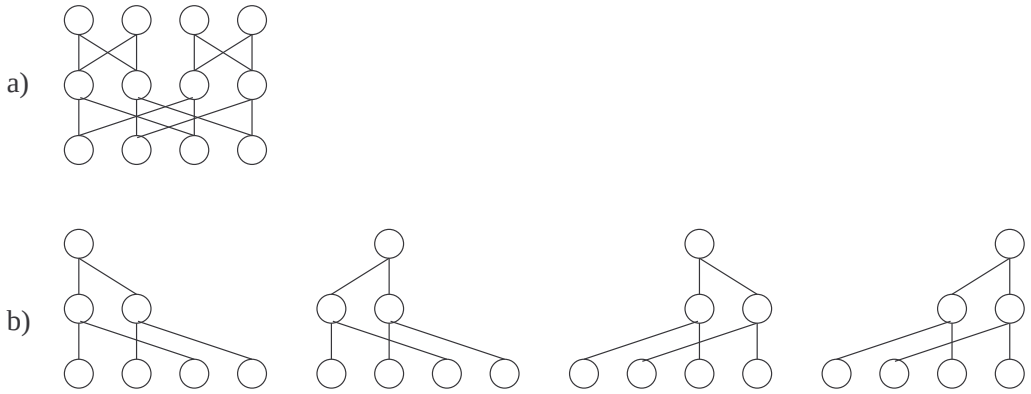

FIGURE 8. a) The butterfly graph of order 3. b) The decomposition of the butterfly graph of order 3 into 4 complete binary trees of height 3.

Proof. From the previous lemma it follows, that $\mathrm{S}'^{-1}_{\min}(p) \leq \overbrace{8^{8^{\cdots 8}}}^{p}$ and $\mathrm{S}'^{-1}_{\min}(p) \geq \overbrace{2^{2^{\cdots 2}}}^{p}$. Applying Lemma 4.5 we obtain $\mathrm{S}'_{\min}(h) = h + \Omega(\lg^*(h))$ and $\mathrm{S}'_{\min}(h) = h + O(\lg^*(h))$. Hence, $\mathrm{S}'_{\min}(h) = h + \Theta(\lg^*(h))$ holds. From Lemma 4.2 it follows, that $\mathrm{S}_{\min}(h) = h + \Theta(\lg^*(h))$. □

4.2. Extension to Butterflies

The butterfly graphs are an important class of graphs to study, as they poses the superconcentrator property and form an inherent structure of some important problems in numerical computations, such as the discrete FFT.

The butterfly graph of order d (denoted by $\mathrm{Bf}(d)$) is the graph $G = (V, E)$, which $V = \{1 \dots d\} \times \{0 \dots 2^{d-1} - 1\}$ and $E = \big\{\big((i,j),(i+1,j)\big), \big((i,j),(i+1,j \oplus 2^{i-1})\big) | 1 \leq i < d, 0 \leq j \leq 2^{d-1} - 1\big\}$, where $\oplus$ denotes bitwise exclusive or. For an example of a butterfly graph, see Figure 8a. This graph can be decomposed into 2^{d-1} complete binary trees of height d. The root of the i-th tree is the vertex $(1, i)$ and this tree contains all vertices, that can be reached from the root (see figure 8b). Note that these trees are not disjoint. We denote these trees as $T_1, \dots T_d$.

The decomposition property implies, that the minimal space complexity of a complete computation on the butterfly graph of order d cannot be lower than the minimal space complexity on the complete binary tree of height d:

Lemma 4.10. *It holds that* $\mathrm{S}(\mathrm{Bf}(d)) \geq \mathrm{S}(\mathrm{Bt}(d))$.

Proof. Let us assume the contrary. Let $\mathcal{K}$ be a complete computation of length l on $\mathrm{Bf}(d)$ such that $\mathrm{S}(\mathcal{K}) < \mathrm{S}(\mathrm{Bt}(d))$. By restricting this computation to any binary tree from the decomposition (for example to T_1) we obtain a complete computation on $\mathrm{Bt}(d)$ with space complexity less than $\mathrm{S}(\mathrm{Bt}(d))$. Formally, $\mathrm{S}(\mathrm{Rst}(\mathcal{K}, 1, l, T_1)) < \mathrm{S}(\mathrm{Bt}(d))$, which is a contradiction. □

On the other side, by sequentially applying complete computations to all binary trees obtained by the decomposition of the butterfly graph, we obtain a complete computation on it. Hence, we can construct a complete computation on the butterfly graph of order d with space complexity equal to the minimal space complexity of the binary tree of height d:

Lemma 4.11. *It holds that* $\mathrm{S}(\mathrm{Bf}(d)) \leq \mathrm{S}(\mathrm{Bt}(d))$.

Proof. Let $\mathcal{K}$ be a complete computation on $\mathrm{Bt}(d)$ such that $\mathrm{S}(\mathcal{K}) = \mathrm{S}(\mathrm{Bt}(d))$. Consider the computation $\mathcal{K}'$ defined as follows:

$$\mathcal{K}' = \mathcal{K}|T_1 + \mathcal{K}|T_2 + \ldots + \mathcal{K}|T_d$$

Computation $\mathcal{K}'$ is a complete computation on $\mathrm{Bf}(h)$ such that $\mathrm{S}(\mathcal{K}') = \mathrm{S}(\mathcal{K}) = \mathrm{S}(\mathrm{Bt}(d))$. □

Therefore the minimal space complexity of the butterfly topology equals to the minimal space complexity of the binary tree topology:

Theorem 4.12. *For the minimal space for the butterfly graph of order d it holds that* $\mathrm{S}_{\min}(d) = d + \Theta(\lg^*(d))$.

Proof. This theorem is an immediate consequence of Lemmas 4.10 and 4.11 and Theorem 4.9. □

5. Reversible simulation of irreversible computation

In previous sections we have discussed complexity aspects of the reversible pebble game on various topologies. Now we shall bound the time and space complexity of the reversible pebble game by the time and space complexity of the standard irreversible pebble game, regardless of the topology.

This result expresses the idea of the reversible simulation of irreversible computations, which was introduced by [1], in a pebbling abstraction.

Theorem 5.1. *Let G be an arbitrary dag. Let G can be pebbled by standard (irreversible) computation with p pebbles in time t. Let $\mathcal{K}$ be a complete reversible computation on the chain Ch_t. Then G can be pebbled by a reversible computation with $p \cdot \mathrm{S}(\mathcal{K})$ pebbles in time at most $\mathrm{T}(\mathcal{K})$.*

Proof. By assumption there exists some irreversible computation $\mathcal{K}_s$ on the graph G, such that $\mathrm{T}(\mathcal{K}_s) = t$ and $\mathrm{S}(\mathcal{K}_s) = p$. Without loss of generality we can assume that the length of $\mathcal{K}$ is $\mathrm{T}(\mathcal{K}) + 1$. We construct a complete reversible computation $\mathcal{K}'$ on the graph $G = (V, E)$ of length $\mathrm{T}(\mathcal{K}) + 1$ as follows:

$$\big(\forall i : 1 \leq i \leq \mathrm{T}(\mathcal{K}) + 1\big)\big(\forall v \in V\big)$$

$$\mathcal{K}'(i)(v) = 1 \Leftrightarrow \big(\exists j : 1 \leq j \leq \mathrm{T}(\mathcal{K}_s)\big)\big(\mathcal{K}(i)(j) = 1 \wedge \mathcal{K}_s(j)(v) = 1\big)$$

Hence, we define $\mathcal{K}'(i)$ as a "superposition" of all configurations in $\mathcal{K}_s$ that are "marked" in $\mathcal{K}(i)$ (see also Figure 9).

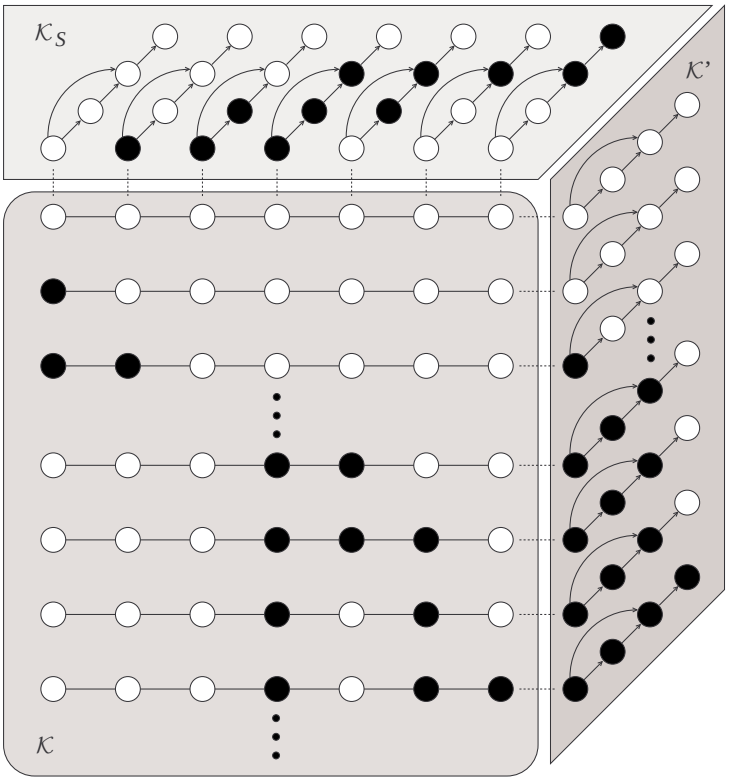

FIGURE 9. Reversible simulation of irreversible computation.

Now we prove that $\mathcal{K}'$ is a reversible computation. At first we prove that two successive configurations can not differ in two vertices.

Assume that the conjecture does not hold: $\mathcal{K}'(i-1)(a) \neq \mathcal{K}'(i)(a)$ and $\mathcal{K}'(i-1)(b) \neq \mathcal{K}'(i)(b)$ for some i and some $a \neq b$. Clearly there exists at most one j such that $\mathcal{K}(i-1)(j) \neq \mathcal{K}(i)(j)$. From the definition of $\mathcal{K}'$ it follows that $\mathcal{K}_s(j)(a) = \mathcal{K}_s(j)(b) = 1$ (otherwise $\mathcal{K}'(i-1)(a) = \mathcal{K}'(i)(a)$ or $\mathcal{K}'(i-1)(b) = \mathcal{K}'(i)(b)$). Since $\mathcal{K}_s(1)$ is an empty configuration, $j \neq 1$. Hence, it holds that $\mathcal{K}_s(j-1)(a) = 1 \vee \mathcal{K}_s(j-1)(b) = 1$, otherwise $\mathcal{K}_s$ is not an (irreversible) computation. Without loss of generality $\mathcal{K}_s(j-1)(a) = 1$. Because $\mathcal{K}$ is a reversible computation, $\mathcal{K}(i-1)(j-1) = \mathcal{K}(i)(j-1) = 1$. Therefore $\mathcal{K}'(i-1)(a) = \mathcal{K}'(i)(a) = 1$, which is a contradiction.

Next we prove, that if two successive configurations in $\mathcal{K}'$ differ in a vertex a, then all direct predecessors of a are pebbled in these configurations.

Again, let us assume that the conjecture does not hold. Hence there exist $a, b \in V$ and i such that $(b, a) \in E$, $\mathcal{K}'(i-1)(a) \neq \mathcal{K}'(i)(a)$ and $\mathcal{K}'(i)(b) = 0$ (note that we have already proven that $\mathcal{K}'(i-1)(b) = \mathcal{K}'(i)(b)$). As was already stated, there exists at most one j such that $\mathcal{K}(i-1)(j) \neq \mathcal{K}(i)(j)$. It also holds that $\mathcal{K}_s(j)(a) = 1$, $j \neq 1$ and $\mathcal{K}(i-1)(j-1) = \mathcal{K}(i)(j-1) = 1$. This implies that $\mathcal{K}_s(j-1)(a) = 0$. Since $\mathcal{K}_s$ is an (irreversible) computation, $\mathcal{K}_s(j-1)(b) = 1$. Therefore $\mathcal{K}'(i-1)(b) = \mathcal{K}'(i)(b) = 1$ which is a contradiction.

We have proven that $\mathcal{K}'$ is a reversible computation. Since $\mathcal{K}(1) = \mathcal{K}(\mathrm{T}(\mathcal{K})+1)$ are empty configurations, $\mathcal{K}'(1) = \mathcal{K}'(\mathrm{T}(\mathcal{K})+1)$ are empty configurations, too.

For each vertex $v \in V$ there exists j such that $\mathcal{K}_s(j)(v) = 1$. For this j there exists some i such that $\mathcal{K}(i)(j) = 1$. Therefore $\mathcal{K}'(i)(v) = 1$. This implies that $\mathcal{K}'$ is a complete computation.

In each configuration of $\mathcal{K}'$ there can be at most $p \cdot \mathrm{S}(\mathcal{K})$ vertices pebbled. Therefore G can be pebbled by the reversible computation using $\mathrm{S}(\mathcal{K}') \leq \mathrm{S}(\mathcal{K}_s) \cdot \mathrm{S}(\mathcal{K}) = p \cdot \mathrm{S}(\mathcal{K})$ pebbles.

Since $\mathcal{K}'$ is a computation of length $\mathrm{T}(\mathcal{K}) + 1$, time of $\mathcal{K}'$ is at most $\mathrm{T}(\mathcal{K})$. □

Applying various pebbling strategies for chain topology to Theorem 5.1, we obtain various upper bounds on time and space complexity of the reversible pebble game.

Corollary 5.2. *Let G be arbitrary an dag. Let G can be pebbled by the standard computation using p pebbles in time t. Then G can be pebbled by the reversible computation in time $O(t^{\log_2 3})$ using $p \cdot O(\log_2 t) = O(p^2)$ pebbles.*

Proof. This result can be obtained by applying the results from Theorem 3.1 and Corollary 3.5 to Theorem 5.1. It does not make sense to consider computations using p pebbles of length over 2^p. We can thus assume $\log_2 t \leq p$.

This result is a pebbling formulation of the results from [1] about reversible simulation of irreversible computations. □

6. Conclusion

In this paper we have defined an abstract model for reversible computations – the reversible pebble game. We have described a technique for proving time and space complexity bounds for this game and presented a lower bound on time for optimal space for the chain topology (valid for infinitely many n) and a tight optimal space bound for the binary tree topology. These results imply that reversible computations require more resources than standard irreversible computations, e.g. for a space complexity of the chain of length n it is $\Theta(1)$ vs. $\Theta(\lg n)$ and for a space complexity of the binary tree of height h it is $h + \Theta(\log^*(h))$ vs. $h + \Theta(1)$.

We have also presented an upper bound on time and space complexity of the reversible pebble game based on the time and space complexity of the standard pebble game.

For further research it would be interesting to examine the time complexity of the reversible pebble game for the tree and the butterfly topology and to consider other important topologies, for example the pyramids.

Acknowledgement: I would like to thank Peter Ružička for introducing me into the area of reversible pebbling and for his valuable advice and consultations.

References

[1] Bennett, C. H.: Time-space trade-offs for reversible computation. SIAM J. Comput., 18(1989), pp. 766–776

[2] Buhrman, H., Tromp J. and Vitányi, P.: Time and space bounds for reversible simulation. Proc. ICALP 2001, LNCS 2076, Springer-Verlag, 2001
[3] Levine, R. Y. and Sherman, A. T.: A note on Bennett's time-space tradeoff for reversible computation. SIAM J. Computing, 19(4):673–677, 1990
[4] Li, M., Tromp, J. and Vitányi P. M. B.: Reversible simulation of irreversible computation. Physica D 120 (1998), pp. 168–176
[5] Li, M. and Vitányi P. M. B.: Reversibility and adiabatic computation: trading time and space for energy. Proceedings of the Royal Society of London Ser. A, 452:1–21, 1996.
[6] Li, M. and Vitányi P. M. B.: Reversible simulation of irreversible computation. Proc. 11th IEEE Conference on Computational Complexity, Philadelphia, Pennsylvania, May 24–27, 1996
[7] Paterson, M. S. and Hewitt, C. E.: Comparative Schematology, MAC Conf. on Concurrent Systems and Parallel Computation, 1970, pp. 119–127
[8] Ružička, P.: Pebbling – The Technique for Analysing Computation Efficiency. SOFSEM'89, 1989, pp. 205–224
[9] Ružička, P. and Waczulík, J.: Pebbling Dynamic Graphs in Minimal Space. Theoretical Informatics and Applications, vol. 28, n. 6, 1994, pp. 557–565
[10] Ružička, P. and Waczulík, J.: On Time-Space Trade-Offs in Dynamic Graph Pebbling. MFCS'93, LNCS 711, Springer-Verlag, 1993, pp. 671–681
[11] Williams, R.: Space-Efficient Reversible Simulations, DIMACS REU report, July 2000
[12] Zavarský, A.: On the Cost of Reversible Computations: Time-Space Bounds on Reversible Pebbling. Manuscript, 1998

Appendix A – Formal definitions

This appendix contains formal definitions of operations on computations are presented.

Definition 6.1. Let $G = (V, E)$ be a dag and let C be a configuration on G. Let $u, v \in V$ and $h \in \{0, 1\}$. Then $\mathrm{Put}(C, v, h)$ is a configuration on G defined as follows:

- $\mathrm{Put}(C, v, h)(u) = C(u)$, if $u \neq v$;
- $\mathrm{Put}(C, v, h)(u) = h$, if $u = v$.

Definition 6.2. Let $G = (V, E)$ be a dag, $V' \subseteq V$. Let $\mathcal{K}$ be a computation of length n on G. A *Restriction* $\mathcal{K}' = \mathrm{Rst}(\mathcal{K}, i, j, V')$ of the computation $\mathcal{K}$ to an interval $\{i \dots j\}$ $(1 \leq i \leq j \leq n)$ and to a subgraph $G' = (V', E \cap (V' \times V'))$ is a computation $\mathcal{K}'$ of length $j - i + 1$ on G' defined as follows:

$$(\forall k \in \{1 \dots j - i + 1\})(\forall v \in V')\mathcal{K}'(k)(v) = \mathcal{K}(i + k - 1)(v)$$

We use the notation $\mathrm{Rst}(\mathcal{K}, i, j)$ when no vertices are removed (e.g., $\mathrm{Rst}(\mathcal{K}, i, j) = \mathrm{Rst}(\mathcal{K}, i, j, V)$ for the graph $G = (V, E)$).

For an example of a restriction, see Figure 1.

Definition 6.3. Let $\mathcal{K}$ be a computation on G of length n. Then the *reverse* of the computation $\mathcal{K}$ (denoted as $\mathrm{Rev}(\mathcal{K})$) is a computation on G defined as follows:

$$\mathrm{Rev}(\mathcal{K})(i) = \mathcal{K}(n + 1 - i)$$

Definition 6.4. Let $\mathcal{K}_1$ and $\mathcal{K}_2$ be computations on a dag G and let $\mathcal{K}_1$ and $\mathcal{K}_2$ have length n_1 and n_2 respectively. Let $\mathcal{K}_1(n_1)$ and $\mathcal{K}_2(1)$ form a transition. Then the *join* of computations C_1 and C_2 (denoted by $\mathcal{K}_1 + \mathcal{K}_2$) is a computation on G of length $n_1 + n_2$ defined as follows:

- $(\mathcal{K}_1 + \mathcal{K}_2)(i) = \mathcal{K}_1(i)$, if $i \leq n_1$
- $(\mathcal{K}_1 + \mathcal{K}_2)(i) = \mathcal{K}_2(i - n_1)$, if $i > n_1$

It is clear that this definition is correct. Configurations $(\mathcal{K}_1 + \mathcal{K}_2)(n_1)$ and $(\mathcal{K}_1 + \mathcal{K}_2)(n_1 + 1)$ form a transition by the assumption. All other successive pairs of configurations form transitions because $\mathcal{K}_1$ and $\mathcal{K}_2$ are computations.

Let C be a configuration on a dag G. Then we can consider the configuration C to be a computation of length 1, so that $\mathcal{K}(1) = C$. Therefore we can also join a computation with a configuration and vice versa.

For an example of a join, see Figure 2.

Definition 6.5. Let $G = (V, E)$ be a dag, $V_1 \subseteq V$, $V_2 \subseteq V$, $V_1 \cap V_2 = \emptyset$. Let C be a configuration on the graph $(V_2, E \cap (V_2 \times V_2))$. Let $\{(w, v) | v \in V_1 \wedge w \in V_2 \wedge C(w) = 0 \wedge (w, v) \in E\} = \emptyset$. Let $\mathcal{K}$ be a computation of length n on the graph $(V_1, E \cap (V_1 \times V_1))$. The *computation $\mathcal{K}$ merged with the configuration C* (denoted by $\mathcal{K} \cdot C$) is a computation on the graph $(V_1 \cup V_2, E \cap ((V_1 \cup V_2) \times (V_1 \cup V_2)))$ of length n defined as follows:

- $(\mathcal{K} \cdot C)(i)(v) = \mathcal{K}(i)(v)$, if $v \in V_1$
- $(\mathcal{K} \cdot C)(i)(v) = C(v)$, if $v \in V_2$

This definition is clearly correct. By adding the same configuration to all configurations of some computation $\mathcal{K}$, there is only one way to violate the rules of the reversible pebble game: if some among the added direct predecessors of a vertex, which the pebble is laid on or removed from, are not pebbled. But this is prohibited by the assumption of the definition.

For an example of a merge, see Figure 3.

Definition 6.6. Let $\mathcal{K}$ be a computation of length n on a dag G and G' be a dag isomorphic to G. Let φ is the isomorphism between G and G'. Then a computation $\mathcal{K}$ *applied to* the graph G' (denoted as $\mathcal{K}|G'$) is a computation on G' of length n such that $(\mathcal{K}|G')(i)(v) = \mathcal{K}(i)(\varphi(v))$ for all $1 \leq i \leq n$ and for all vertices v of G.

Appendix B – Formal proofs

In this appendix formal proofs of some lemmas and theorems are presented.

Proof of Lemma 3.2. Assume that the first equality does not hold. W.l.o.g. we can assume that $\mathrm{T}(\mathrm{Rst}(\mathcal{K}, 1, i)) < \mathrm{T}(\mathrm{Rst}(\mathcal{K}, i, l))$. Otherwise we can consider $\mathrm{Rev}(\mathcal{K})$ instead of $\mathcal{K}$. Now let us consider the computation $\mathcal{K}' = \mathrm{Rst}(\mathcal{K}, 1, i) + \mathrm{Rev}(\mathrm{Rst}(\mathcal{K}, 1, i))$ (see Figure 10). $\mathcal{K}'$ is a complete computation on Ch_n. Clearly $\mathrm{S}(\mathcal{K}') \leq \mathrm{S}(\mathcal{K})$ and $\mathrm{T}(\mathcal{K}') < \mathrm{T}(\mathcal{K})$. But this contradicts to $\mathrm{T}(\mathcal{K}) = \mathrm{T}(n, \mathrm{S}_{\min}(n))$.

We have proven that $\mathrm{T}(\mathrm{Rst}(\mathcal{K}, 1, i)) = \mathrm{T}(\mathrm{Rst}(\mathcal{K}, i, l))$. Clearly, $\mathrm{T}(\mathcal{K}) = \mathrm{T}(\mathrm{Rst}(\mathcal{K}, 1, i)) + \mathrm{T}(\mathrm{Rst}(\mathcal{K}, i, l))$, so the second equality is obvious. □

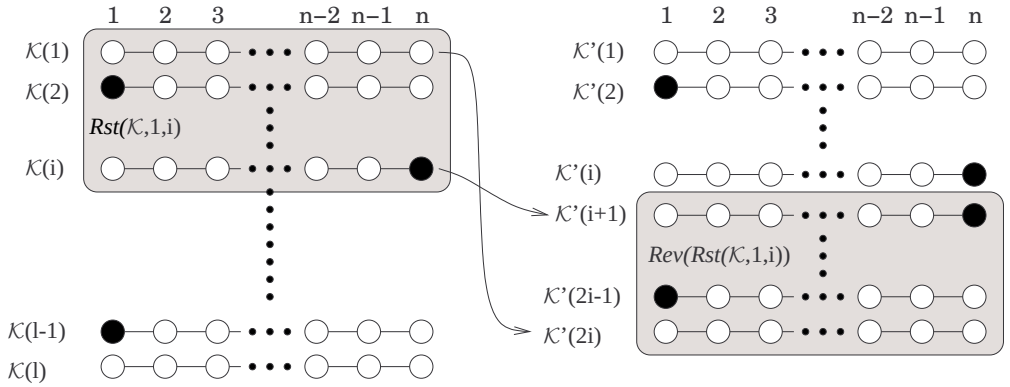

FIGURE 10. Proof of Lemma 3.2

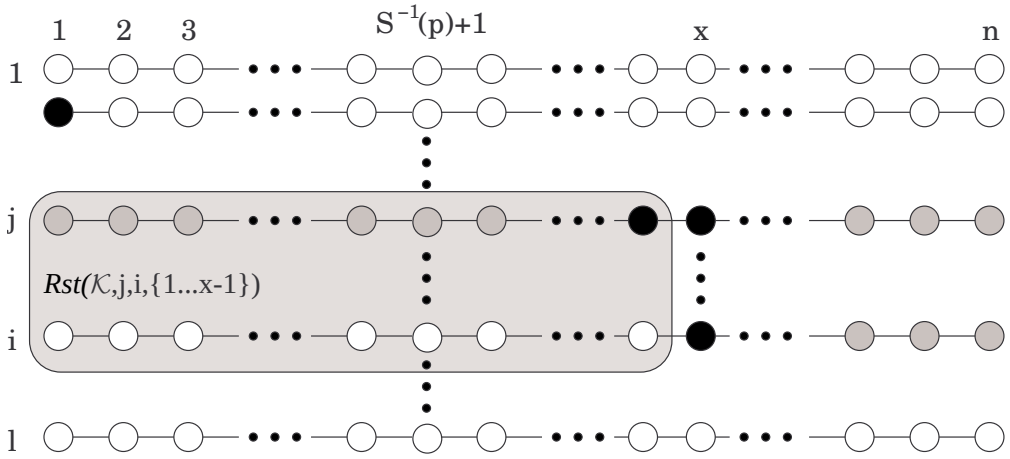

FIGURE 11. Computation $\mathcal{K}$. Gray circles are vertices with unknown state.

Proof of Lemma 3.3. Let $x = \max\{\min\{j | j \in \{1 \dots n\} \wedge \mathcal{K}(i)(j) = 1\} | i \in \{1 \dots l\}\}$. Theorem 3.1 states that $n = \mathrm{S}^{-1}(p+1) = 2\mathrm{S}^{-1}(p) + 1$. At first we show by contradiction that $x \leq \mathrm{S}^{-1}(p) + 1$.

Assume that $x > \mathrm{S}^{-1}(p) + 1$. Hence there exists some complete computation $\mathcal{K}$ on Ch_n of length l and some i $(1 \leq i \leq n)$ such that $\mathcal{K}(i)(x) = 1$ and $(\forall y : 1 \leq y < x)\ \mathcal{K}(i)(y) = 0$.

Let j be minimal number such that $(\forall k : j \leq k \leq i)\ \mathcal{K}(k)(x) = 1$. Clearly $j > 1$, therefore $\mathcal{K}(j-1)(x) = 0$. Since $\mathcal{K}$ is a reversible computation, it holds: (see Figure 11)

$$\mathcal{K}(j-1)(x-1) = \mathcal{K}(j)(x-1) = 1$$

Now consider the computation $\mathcal{K}_1$:

$$\mathcal{K}_1 = \mathrm{Rev}(\mathrm{Rst}(\mathcal{K}, j, i, \{1 \dots x-1\})) + \mathrm{Rst}(\mathcal{K}, j, i, \{1 \dots x-1\})$$

$\mathcal{K}_1$ is a complete computation on the chain of length $x-1$. Hence $\mathrm{S}(\mathcal{K}_1) < \mathrm{S}(\mathcal{K})$, because vertex x is pebbled in $\mathcal{K}$ in configurations $j \dots i$. That means that $\mathcal{K}_1$ is

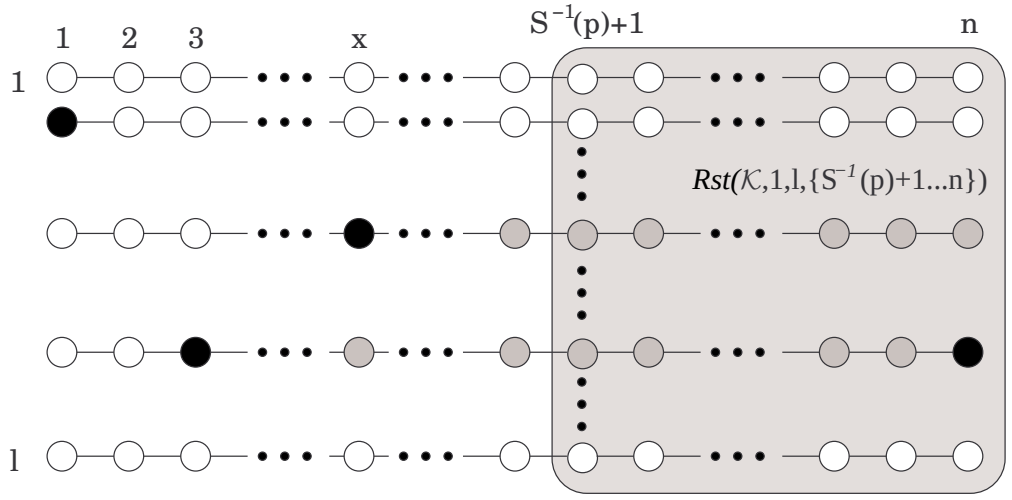

FIGURE 12. Computation $\mathcal{K}$. Gray circles are vertices with unknown state.

a complete computation on a chain longer than $\mathrm{S}^{-1}(p)$ with space complexity at most p, which is a contradiction.

Now we prove that $x \geq \mathrm{S}^{-1}(p) + 1$. Assume that $x < \mathrm{S}^{-1}(p) + 1$. Hence there exists some complete computation $\mathcal{K}$ on Ch_n of length l such that in each its configuration there is at least one vertex with number less than $\mathrm{S}^{-1}(p) + 1$ pebbled. (See Figure 12.)

Consider $\mathcal{K}_2 = \mathrm{Rst}(\mathcal{K}, 1, l, \{\mathrm{S}^{-1}(p) + 1, \ldots, n\})$. It holds that $\mathrm{S}(\mathcal{K}_2) < \mathrm{S}(\mathcal{K})$. But clearly $\mathcal{K}_2$ is a complete computation on graph isomorphic to the chain of length $\mathrm{S}^{-1}(p)+1$. Again, we have a complete computation with space complexity at most p on a chain longer than $\mathrm{S}^{-1}(p)$.

By proving both inequalities we proved $x = \mathrm{S}^{-1}(p) + 1$. □

Proof of Lemma 4.2. Let $\mathcal{K}$ be a semicomplete computation on $\mathrm{Bt}(h)$ such that $\mathrm{S}(\mathcal{K}) = \mathrm{S}'_{\min}(h)$. It holds that $\mathcal{K} + \mathrm{Rev}(\mathcal{K})$ is a complete computation on $\mathrm{Bt}(h)$ and $\mathrm{S}(\mathcal{K} + \mathrm{Rev}(\mathcal{K})) = \mathrm{S}(\mathcal{K}) \geq \mathrm{S}_{\min}(h)$. Therefore $\mathrm{S}'_{\min}(h) \geq \mathrm{S}_{\min}(h)$.

Now we prove the first inequality. Let $\mathcal{K}$ be a complete computation on $\mathrm{Bt}(h)$, such that $\mathrm{S}(\mathcal{K}) = \mathrm{S}_{\min}(h)$. Let l be the length of $\mathcal{K}$, let $\mathcal{K}(i)(\mathrm{R}(\mathrm{Bt}(h))) = 1$. Now consider a computation

$$\mathcal{K}_2 = \mathrm{Rst}(\mathcal{K}, 1, i) + \mathrm{Rst}(\mathcal{K}, i, l, \mathrm{Lt}(\mathrm{Bt}(h)) \cup \mathrm{Rt}(\mathrm{Bt}(h))) \cdot \mathrm{Put}(\mathrm{R}(\mathrm{Bt}(h)), \mathrm{R}(\mathrm{Bt}(h)), 1)$$

Clearly, $\mathcal{K}_2$ is a semicomplete computation on $\mathrm{Bt}(h)$ and $\mathrm{S}(\mathcal{K}_2) \leq \mathrm{S}(\mathcal{K})+1$. Therefore it holds

$$\mathrm{S}_{\min}(h) + 1 = \mathrm{S}(\mathcal{K}) + 1 \geq \mathrm{S}(\mathcal{K}_2) \geq \mathrm{S}'_{\min}(h)$$

□

Communicated by (The editor will be set by the publisher).
(The dates will be set by the publisher).