The Complexity of Simulation and (Exotic) Matrix Multiplications

Massimo Cairo
Università di Trento
massimo.cairo@unitn.it

Romeo Rizzi
Università di Verona
romeo.rizzi@univr.it

May 10, 2016

Abstract

We study the complexity of computing the simulation preorder of finite transition systems, a crucial problem in model checking of temporal logic, showing that it is strongly related to some variants of matrix multiplication.

We first show that any $O(n^{\alpha})$-time algorithm for $n$-states transition systems can be used to compute the product of two $n \times n$ boolean matrices in $O(n^{\alpha})$ time. This reduction is the first evidence of the difficulty to get a truly subcubic combinatorial algorithm for the simulation preorder, and holds even restricting the problem to acyclic systems. For acyclic Kripke structures, we show an algorithm that employs fast (boolean) matrix multiplication and runs in $n^{\omega+o(1)}$ time (where $\omega < 2.4$ is the exponent of matrix multiplication). Moreover, we exhibit $O(n^2)$-size canonical certificates that can be checked by verifying a constant number of $n \times n$ standard matrix multiplications over the integers, i.e., in $O(n^2)$ (randomized) time.

For cyclic structures, we give some evidence that the problem might possibly be harder. We define the max-semi-boolean matrix multiplication (MSBMM) as the matrix multiplication on the semi-ring $(\max, \times)$ where one of the two matrices contains only 0’s and 1’s. We obtain $O(n^2)$-size canonical certificates for cyclic Kripke structures that can be checked by verifying a constant number of $n \times n$ MSBMMs. Then, we show that verifying a $n \times n$ MSBMM can be reduced to verifying the simulation preorder in a $O(n \log n)$-states Kripke structure. Hence, for any $\alpha \geq 2$, if MSBMM can be verified in $O(n^\alpha)$ time, then the simulation preorder admits certificates that can be checked in $O(n^\alpha)$ time, otherwise the simulation preorder does not admit a $O(n^\alpha)$-time algorithm.
1 Introduction

In the context of model checking, the simulation preorder of a transition system provides an abstraction that allows to reduce the state space while preserving the satisfiability of a large class of temporal logic formulas [3]. In the case of Kripke structures, the transition system is described as a graph, whose vertices are labeled and represent states of the system, and edges represent transitions between states. The simulation preorder is defined co-inductively: a state \(s\) simulates a state \(t\) whenever \(t\) and \(s\) are labeled in the same way and, for every transition from \(s\) to \(s'\), there is a transition from \(t\) to \(t'\) such that \(t'\) simulates \(s'\).

The problem of computing the simulation preorder on finite Kripke structures has been studied thoroughly, and a large family of algorithms have been proposed. Let \(n\) be the number of states and \(m\) the number of transitions (where we assume \(n \leq m\)). Polynomial algorithms have been presented in [1, 5, 6], improved to \(O(mn)\) time independently in [12] and [2]. More recently, a new parameter has been introduced for the analysis of the running time, namely, the number \(n^*\) of equivalence classes in the simulation preorder relation. New algorithms have been proposed [10, 11, 16, 17, 15, 4] that run faster when \(n^*\) is much smaller than \(n\).

In this work we address the following concerns. First of all, the above algorithms all require \(\Omega(n^3)\) time in the worst case, yet no argument is given to show why this running time seems to be required. Secondarily, previous algorithms do not provide explicit certificates, and no procedure has been proposed to check the simulation preorder more efficiently than computing it from scratch. We provide some answers to both these questions.

First, we show that the Simulation problem on an \(n\)-state Kripke structure is at least as hard as \(n \times n\) boolean matrix multiplication. This gives a good reason why obtaining a truly subcubic algorithm seems to be hard, without relying on “algebraic” techniques such as those employed to get fast matrix multiplication. To prove this lower bound, we do not rely on the possibility that transition systems may be cyclic: actually, our reduction uses acyclic transition systems of constant depth. It is interesting to study acyclic transition systems on their own, since, to the best of our knowledge, no asymptotically faster algorithm for the acyclic case is known with respect to the cyclic case. We show that, employing fast matrix multiplication, a truly subcubic algorithm for the acyclic case is possible. Specifically, if two \(n \times n\) boolean matrices can be multiplied in \(n^{\omega+o(1)}\) time (known to be true for \(\omega \approx 2.4\) [7]), then the simulation preorder of acyclic \(n\)-states Kripke structures can be computed in \(n^{\omega+o(1)}\) time. Together with the previous result, this shows that the simulation problem in acyclic Kripke structures is essentially equivalent to boolean matrix multiplication. We also obtain \(O(n^2)\)-size canonical certificates that can be checked by verifying a boolean matrix multiplication. By transforming this boolean matrix multiplication into a standard matrix multiplication, these certificates can be verified in (randomized) \(O(n^2)\) time [9, 13, 14].

In the cyclic case, we provide \(O(n^2)\)-size canonical certificates, too. In this case, however, they are checked using a more general variant of matrix multiplication. We introduce the max-semi-boolean matrix multiplication (MSBMM), a kind of matrix multiplication between a matrix of numbers and a boolean matrix, where the outer operation is max, and the columns of the boolean matrix act as a mask, selecting which values of the other matrix should be taken into account and which should be ignored. The MSBMM can be defined equivalently as the matrix multiplication on the semi-ring \((\max, \times)\), where one of the two matrices contains only zeros (for false) and ones (for true). This variant of matrix multiplication is more general than boolean matrix multiplication (consider the case where both matrices contain only zeros and ones) and, to the best of our knowledge, no \(n^{\omega+o(1)}\)-time algorithm is known to either compute or verify its result. We show that the verification of the MSBMM between two \(n \times n\) matrices can be reduced to computing the simulation preorder in a \(O(n \log n)\)-states Kripke structure. Hence, if MSBMM does not admit
a $\tilde{O}(n^\alpha)$-time verification algorithm, for some $\alpha \geq 2$, then then simulation preorder cannot be computed in $\tilde{O}(n^\alpha)$ time.

In this work, we study the simulation preorder considering an equivalent two-player game, which we call two-pebble game. Determining the winner in these games is equivalent to determining whether a state simulates another in a Kripke structure. We say that this game is a pseudo-infinite game, since its plays can be infinite, but the winning condition is non-trivial only on finite plays. Many of our results on simulation, including a self-certifying $O(nm)$-time algorithm for cyclic Kripke structures, arise naturally from a more general analysis of pseudo-infinite games.

# 2 Preliminaries

**Simulation.** A labeled directed graph is a structure $T = (V, E, L)$ where $(V, E)$ is a directed graph, and $L : V \to \Lambda$ is a labeling function on vertices.

A binary relation $R \subseteq V \times V$ is a simulation if, for every $(u, v) \in R$, we have that:

- $L(u) = L(v)$, and
- for every edge $(u, u') \in E$ there is an edge $(v, v') \in E$ such that $(u', v') \in R$.

For $u, v \in V$, we say that $v$ simulates $u$ (written $u \preceq v$) if there exists a simulation $R$ with $(u, v) \in R$. The relation $\preceq$ is a preorder relation, called the simulation preorder of $T$. The problem SIMULATION of size $n = |V|$ asks to compute the relation $\preceq$ over $V \times V$. In the ACYCLIC variant, $(V, E)$ is required to be acyclic.

**Pseudo-infinite games.** Let $A$ and $B$ denote the two players Alice and Bob. For a player $P \in \{A, B\}$, denote by $1 - P$ the other player (i.e., $1 - A := B$ and $1 - B := A$).

An arena is a structure $\mathcal{G} = (V, \mathcal{E}, V^A, V^B)$ consisting of a directed graph $(V, \mathcal{E})$ and a partition $(V^A, V^B)$ of $V$. The set $V(\mathcal{G}) = V$ is the set of configurations, $\mathcal{E}(\mathcal{G}) = \mathcal{E} \subseteq V \times V$ is the set of moves, and $V^P(\mathcal{G}) = V^P$ are the configurations where player $P$ holds the turn. We assume all these sets to be finite. We write $\sigma \rightarrow_P \sigma'$ if $(\sigma, \sigma') \in \mathcal{E}(\mathcal{G})$.

A play on $\mathcal{G}$ is a finite or infinite walk on the directed graph $(V, \mathcal{E})$. More specifically, a play $\pi$ on $\mathcal{G}$ of length $|\pi| \in \mathbb{N} \cup \{\infty\}$, from the initial configurations $V_0 \subseteq V$, is a sequence of configurations $\sigma_i(\pi)$ for $0 \leq i < |\pi| + 1$ such that $\sigma_0(\pi) \in V_0$ and $\sigma_i(\pi) \rightarrow_\mathcal{G} \sigma_{i+1}(\pi)$ for every $i < |\pi|$. We write $\pi \uparrow$ if $|\pi| = \infty$ and $\pi \downarrow \mathcal{U}$ if $|\pi| = \ell \in \mathbb{N}$ and $\sigma_\ell(\pi) \in \mathcal{U}$ (we omit $\ell$ and $\mathcal{U}$ in this notation as needed).

A positional strategy on $G$ for player $P \in \{A, B\}$ is a function $s^P : V^P \to V \cup \{\bot\}$ such that if $s^P(\sigma) = \sigma' \in V$ then $\sigma \rightarrow_\mathcal{G} \sigma'$. A positional strategy $s^P$ prescribes the next move of player $P$ when she holds the turn: player $P$ moves from $\sigma$ to $s^P(\sigma)$ if $s^P(\sigma) \neq \bot$ and stops on $\sigma$ if $s^P(\sigma) = \bot$.

A play $\pi$ is consistent with $s^P$ if $\sigma_{i+1}(\pi) = s^P(\sigma_i(\pi))$ whenever $i < |\pi|$ with $\sigma_i(\pi) \in V^P$, and $s^P(\sigma_\ell(\pi)) = \bot$ if $\pi \downarrow \mathcal{U} V^P$. Given a property $\mathcal{P}$ for a generic play $\pi$, player $P$ guarantees $\mathcal{P}$ on $\mathcal{G}$ from $V_0 \subseteq V$, with the strategy $s^P$, if every play $\pi$ on $\mathcal{G}$ from $V_0$ consistent with $s^P$ satisfies the property $\mathcal{P}$.

A pseudo-infinite game is a pair $(\mathcal{G}, \mathcal{F})$, where $\mathcal{F} = (\mathcal{F}^A, \mathcal{F}^B)$ is a partition of $V$. For a play $\pi$ on $\mathcal{G}$, player $P$ wins $\pi$ on $(\mathcal{G}, \mathcal{F})$ if $\pi \downarrow \mathcal{F}^P$. Infinite plays are neither won nor lost by any player. Player $P$ survives $\pi$ if either $P$ wins $\pi$ or $\pi \uparrow$.
Certificates for pseudo-infinite games. Let \((G, F)\) be a pseudo-infinite game and \(U \subseteq V(G)\). Define the set \(f^P_{G,F}(U) \subseteq V(G)\) as follows

\[
\sigma \in f^P_{G,F}(U) \iff \begin{cases} 
\sigma \in F^P \lor \bigvee_{\sigma \rightarrow \sigma'} \sigma' \in U & \text{if } \sigma \in V^P(G) \\
\sigma \in F^P \land \bigwedge_{\sigma \rightarrow \sigma'} \sigma' \in U & \text{if } \sigma \in V^{1-P}(G)
\end{cases}
\]

where as usual \(\bigvee \emptyset = \text{false} \) and \(\bigwedge \emptyset = \text{true}\). Observe that \(P\) guarantees, with a positional strategy \(s^P\), that if \(\sigma_i(\pi) \in f^P_{G,F}(U)\) then either \(\pi \downarrow F^P\) or \(\sigma_{i+1}(\pi) \in U\). It is sufficient to take \(s^P(\sigma) = \perp\) if \(\sigma \in V^P(G) \cap F^P\) and \(s^P(\sigma) \in U\) if \(\sigma \in f^P_{G,F}(U) \cap V^P(G) \cap F^{1-P}\) (by definition of \(f^P_{G,F}\), a move \(\sigma \rightarrow \sigma'\) with \(\sigma' \in U\) exists). Notice that \(f\) is monotone, i.e., \(f^P_{G,F}(U') \subseteq f^P_{G,F}(U)\) if \(U' \subseteq U\).

We say that \(U\) is stable on \((G, F)\) for \(P\) if \(U \subseteq f^P_{G,F}(U)\). If \(U\) is stable, then player \(P\) guarantees with \(s^P\) to survive from \(U\): indeed, for every play \(\pi\) from \(U\) consistent with \(s^P\), we have \(\sigma_0(\pi) \in U\) and if \(\sigma_i(\pi) \in U \subseteq f^P_{G,F}(U)\) then either \(\pi \downarrow F^P\) or \(\sigma_{i+1}(\pi) \in U\). Notice that \(f^P_{G,F}(U) = V(G) \setminus f^{1-P}_{G,F}(V(G) \setminus U)\) by de Morgan laws.

A potential on \(G\) is a function \(p : V(G) \to \mathbb{N} \cup \{\infty\}\). Given a potential \(p\), we define the potential \(g^P_{G,F}(p)\) as follows

\[
g^P_{G,F}(p)(\sigma) = \begin{cases} 
\min \{ h^P_F(\sigma) \} \cup \{ 1 + p(\sigma') | \sigma \rightarrow \sigma' \} & \text{if } \sigma \in V^P(G) \\
\max \{ h^P_F(\sigma) \} \cup \{ 1 + p(\sigma') | \sigma \rightarrow \sigma' \} & \text{if } \sigma \in V^{1-P}(G)
\end{cases}
\]

where

\[
h^P_F(\sigma) := \begin{cases} 
0 & \text{if } \sigma \in F^P \\
\infty & \text{if } \sigma \in F^{1-P}
\end{cases}
\]

and \(1 + \infty = \infty\). Observe that \(g^P_{G,F}(p)\) can be defined equivalently by the equations \(\{ \sigma | g^P_{G,F}(p)(\sigma) < k + 1 \} = f^P_{G,F}(\{ \sigma | p(\sigma) < k \})\) for \(k \in \mathbb{N}\). Player \(P\) guarantees, with a positional strategy \(s^P\), that if \(g^P_{G,F}(p)(\sigma_i(\pi)) < \infty\) then either \(\pi \downarrow F^P\) or \(p(\sigma_{i+1}(\pi)) < g^P_{G,F}(p)(\sigma_i(\pi))\).

A potential \(p\) is decreasing for \(P\) if \(g^P_{G,F}(p)(\sigma) \leq p(\sigma)\) for every \(\sigma \in V(G)\). If \(p\) is decreasing for \(P\), then \(P\) guarantees to win with \(s^P\) from \(\{ \sigma_0 | p(\sigma_0) < \infty \}\). Indeed, for any play \(\pi\) consistent with \(s^P\) such that \(p(\sigma_0(\pi)) = p_0 < \infty\), the value \(p(\sigma_i(\pi))\) decreases strictly with \(i\) so \(|\pi| \leq p_0\) and \(\pi \downarrow F^P\).

Winning rank. Define recursively \(W^P_0 = 0\), and \(W^P_{<k+1} = f^P_{G,F}(W^P_k)\) for every \(k \in \mathbb{N}\). Clearly \(W^P_0 \subseteq W^P_1\), and inductively \(W^P_{<k} \subseteq W^P_{<k+1}\) for every \(k \in \mathbb{N}\) since \(f^P_{G,F}\) is monotone. Define the sets \(W^P_k = W^P_{<k+1} \setminus W^P_{<k}\) for \(k \in \mathbb{N}\), observing that they are pairwise disjoint, and let \(r^P_{G,F}(\sigma) = k\) if \(\sigma \in W^P_k\) for some \(k\), and \(r^P_{G,F}(\sigma) = \infty\) otherwise. The potential \(r^P_{G,F}\) (called winning rank of \(P\)) is the unique potential \(p\) such that \(g^P_{G,F}(p) = p\). Indeed, \(\{ \sigma | r^P_{G,F}(\sigma) < k + 1 \} = f^P_{G,F}(\{ \sigma | r^P_{G,F}(\sigma) < k \})\) for every \(k \in \mathbb{N}\) by construction, so \(g^P_{G,F}(r^P_{G,F}) = r^P_{G,F}\). Suppose to have a distinct solution \(g^P_{G,F}(p) = p\) and take the smallest \(k\) such that \(\{ \sigma | p(\sigma) = k \} \neq W^P_k\). Then \(\{ \sigma | p(\sigma) < k \} = W^P_{<k}\) so \(\{ \sigma | p(\sigma) < k + 1 \} = f^P_{G,F}(\{ \sigma | p(\sigma) < k \}) = f^P_{G,F}(W^P_{<k}) = W^P_{<k+1}\) and \(\{ \sigma | p(\sigma) = k \} = \{ \sigma | p(\sigma) < k + 1 \} \setminus W^P_{<k} = W^P_k\).

We have that if \(r^P_{G,F}(\sigma) < \infty\) then \(r^P_{G,F}(\sigma) < |V|\), since \(W^P_{k+1} \neq \emptyset\) implies \(W^P_k \neq 0\).

The winning set of \(P\) is \(W^P(G,F) = \{ \sigma | r^P_{G,F}(\sigma) < \infty \}\). Since \(r^P_{G,F}\) is decreasing, \(P\) guarantees to win from \(W^P(G,F)\). The surviving set of \(P\) is \(S^P(G,F) = \{ \sigma | r^P_{G,F}(\sigma) = \infty \} = V(G) \setminus W^{1-P}(G,F)\). Observe that \(S^P(G,F)\) is stable for \(P\), and actually \(S^P(G,F) = f^P_{G,F}(S^P(G,F))\). Thus, \(P\) guarantees to survive from \(S^P(G,F)\).
Composing the winning rank. Let \( \delta^+(\sigma) \) be the number of moves \( \sigma \to G \sigma' \), and let \( c_k(\sigma) \leq \delta^+(\sigma) \) be the number of moves \( \sigma \to G \sigma' \) such that \( \sigma' \in \mathcal{W}^p_k \). We can characterize \( \mathcal{W}^p_{k+1} \) as follows: for \( \sigma \in \mathcal{V}_p \), we have \( \sigma \in \mathcal{W}^p_{k+1} \) if \( \sigma \in \mathcal{F}^P \) or \( c_k(\sigma) > 0 \), while for \( \sigma \in \mathcal{V}_{1-p} \), we have \( \sigma \in \mathcal{W}^p_{k+1} \) if \( \sigma \in \mathcal{F}^P \) and \( c_k(\sigma) = \delta^+(\sigma) \). The winning rank can be computed in linear time with the following algorithm. We maintain a counter \( \mathcal{V} \to \mathbb{N} \). Start with \( c(\sigma) = c_0(\sigma) = 0 \) for every \( \sigma \in \mathcal{V} \), and compute the set \( \mathcal{W}^0_0 = (\mathcal{V} \cap \mathcal{F}^P) \cup \{ \sigma \in \mathcal{V}^{1-p} \cap \mathcal{F}^P \mid \delta^+(\sigma) = 0 \} \) in \( O(|\mathcal{V}|) \) time. Then, for each \( k = 1, \ldots, |\mathcal{V}| - 1 \), compute \( c_k \) and \( \mathcal{W}^k_k \) as follows: for each move \( \sigma \to G \sigma' \) with \( \sigma' \in \mathcal{W}^p_{k-1} \), increase the value of \( c(\sigma) \) by one. At the end of this process, \( c(\sigma) = c_k(\sigma) \) for every \( \sigma \in \mathcal{V} \). If a configuration \( \sigma \) satisfies for the first time the condition \( c_k(\sigma) > 0 \) (if \( \sigma \in \mathcal{V}^P \)) or \( c_k(\sigma) = \delta^+(\sigma) \) (if \( \sigma \in \mathcal{V}^{1-p} \cap \mathcal{F}^P \)), then add \( \sigma \) to \( \mathcal{W}^p_k \). Visiting any single move takes constant time. Since each move is visited at most once, the total time is \( O(|\mathcal{V}| + |\mathcal{E}|) \).

Observe that the winning rank can be verified in linear time and logarithmic space by checking \( g^\mathcal{F}_G(\sigma) = g^\mathcal{F}_G(\sigma) \) for every configuration \( \sigma \in \mathcal{V} \). We obtain the following.

**Theorem 1.** The winning and surviving sets of a pseudo-infinite game can be computed in linear time, producing a linear-size canonical certificate verifiable in linear time and logarithmic space.

Two-pebble games. Let \( G_A \) and \( G_B \) be directed graphs, where \( G_P = (V_P, E_P) \) for each player \( P \in \{A, B\} \). In the following we write \( u \rightarrow P u' \) if \( (u, u') \in E_P \). The two-pebble arena on \( (G_A, G_B) \) is the arena \( G(G_A, G_B) = G = (V, E, V^A, V^B) \) defined as follows. The set of configurations is \( \mathcal{V} = V(G_A, G_B) = V^A \cup V^B \) where \( V^P = \{P\} \times V_P \times V_{1-P} \). For each configuration \( (P, u, v) \in V^P \) and for each edge \( u \rightarrow P u' \), there is a move \( (P, u, v) \rightarrow G (1-P, v, u') \). A two-pebble arena can be interpreted as follows. There are two pebbles, one controlled by player \( A \) and the other controlled by player \( B \), which are moved in turn by the two players along the edges of \( G_A \) and \( G_B \) respectively. In the configuration \( (P, u, v) \in V^P \), the pebble of \( P \) is located on \( u \in V_P \), the pebble of \( 1-P \) is located on \( v \in V_{1-P} \), and player \( P \) has to move next.

Let \( n_P = |V_P| \) and \( m_P = |E_P| \). Observe that \( |\mathcal{V}| = 2n_A n_B, |\mathcal{E}| = n_A m_B + n_B m_A \), and \( \mathcal{G}(G_A, G_B) \) can be constructed in time \( O(n_A n_B + n_A m_B + n_B m_A) \).

Let \( \mathcal{F} = (\mathcal{F}^A, \mathcal{F}^B) \) be a partition of \( \mathcal{V} \). The two-pebble game \( (G(G_A, G_B), \mathcal{F}) \) of size \( n_A \times n_B \) is the pseudo-infinite game \( (G, \mathcal{F}) \) on the two-pebble arena \( G = G(G_A, G_B) \). The problem Two-Pebble Winning Set (2PWS) asks to compute the set \( \mathcal{W}^P = \mathcal{W}^P(G, \mathcal{F}) \).

In the Acyclic variant, \( G_A \) and \( G_B \) are both required to be acyclic. In the Semi-Acyclic variant, we require at least one of \( G_A \) or \( G_B \) to be acyclic. In the Equivalence variant, we require \( (G(G_A, G_B), \mathcal{F}) \) to be equivalence-restricted, that is, \( G_A = G_B = (V, E) \) and \( \mathcal{F}^B = \{(A, u, v) \in V \mid u \sim v \} \) where \( \sim \) is an equivalence relation. Moreover, in this last variant we are interested in computing only \( \mathcal{W}^A \) and not \( \mathcal{W}^B \).

**Reductions.** The following lemma shows that Simulation is equivalent to Equivalence 2PWS.

**Lemma 2.** Let \( T = (V, E, L) \) be a labeled directed graph. Define \( G = (V, E), \mathcal{G} = G(G, G), \mathcal{F}^B = \{(A, u, v) \in V(G) \mid L(u) = L(v)\} \) and \( \mathcal{F}^A = V(G) \setminus \mathcal{F}^B \). For any \( u, v \in V \) we have \( u \sim v \) iff \( (A, u, v) \in \mathcal{S}^B(G, \mathcal{F}) \).

**Proof.** Suppose \( u_0 \leq v_0 \) and take a simulation \( R \) such that \( (u_0, v_0) \in R \). Define \( \mathcal{U} = \{(u, v) \mid (u, v) \in R\} \cup \{(B, v, u') \mid \exists u \text{ such that } (u, v) \in R \text{ and } (u, u') \in E\} \).

We prove that \( \mathcal{U} \) is stable on \( (G, \mathcal{F}^B) \), so \( (A, u_0, v_0) \in \mathcal{U} \subseteq \mathcal{S}^B(G, \mathcal{F}) \). Take \( (A, u, v) \in \mathcal{U} \). Since \( (u, v) \in R \), we have \( L(u) = L(v) \) by definition of simulation, so \( (A, u, v) \in \mathcal{F}^B \). Moreover, for
every \((A, u, v) \rightarrow_{G} (B, v, u')\) we have \((u, u') \in E\) so \((B, v, u') \in U\) by construction. Now take any \((B, v, u') \in U\) and let \(u \in V\) be such that \((u, v) \in R\) and \((u, u') \in E\). By definition of simulation, there is a \(v' \in V\) such that \((v, v') \in E\) and \((u', v') \in R\). Hence, \((B, v, u') \rightarrow_{G} (A, u', v')\) with \((A, u', v') \in U\).

(\(\iff\)) We prove that the relation \(R = \{(u, v) \mid (A, u, v) \in S^B(G, F)\}\) is a simulation. Suppose \((u, v) \in R\) so \((A, u, v) \in S^B(G, F)\). Observe that \((A, u, v) \in F^B\), so \(L(u) = L(v)\), otherwise Bob loses the play \((A, u, v) \downarrow\). For any edge \((u, u') \in E\), we have \((A, u, v) \rightarrow_{G} (B, v, u')\) and, since \((A, u, v) \in S^B(G, F)\), also \((B, v, u') \in S^B(G, F)\). However, since \((B, v, u') \notin F_B\), then there exists a \(v' \in V\) such that \((B, v, u') \rightarrow_{G} (A, u', v')\) and \((A, u', v') \in S^B(G, F)\). In particular, \((v, v') \in E\) and \((u', v') \in R\).

\(\square\)

Remark 3. The arena \(G = G(G, G)\) has \(O(n^2)\) configurations, \(O(nm)\) moves, and can be constructed in \(O(nm)\) time, where \(n = |V|\) and \(m = |E|\) assuming \(m \geq n\). By Theorem 1, the winning rank \(r^B_{G,F}\) can be computed in \(O(|V(G)| + |E(G)|) = O(nm)\) time. Hence, \textsc{Simulation} can be computed in \(O(nm)\) time, producing \(r^B_{G,F}\) as a canonical certificate.

The following lemma shows that the equivalence-restricted variants of two-pebble games are not simpler than the general versions.

Lemma 4. For any (acyclic) two-pebble game \((G(G_A, G_B), F)\) of size \(n_A \times n_B\), there exists an (acyclic) equivalence-restricted two-pebble game \((G(G, G), F')\) of size \(n = O(n_A + n_B)\), constructible in \(O(n^2)\) time, and a map \(f: V(G_A, G_B) \rightarrow V(G, G)\), computable in \(O(1)\) time, such that \((P, u, v) \in W^A(G(G_A, G_B), F)\) iff \((P, u, v) \in W^A(G(G, G), F')\).

Proof. Let \(G = (V, E)\), where \(V = V_A \cup V_B \cup \{v^* \mid v \in V_A\}\) and \(E\) contains all the edges in \(E_A \cup E_B\) plus the following extra edges:

- \((u, u^*)\) for every \(u \in V_A\),
- \((v, u)\) for every \((B, v, u) \in F_B\),
- \((v, u^*)\) for every \((A, u, v) \in F_B\).

Let \(x \sim y\) for every \(x, y \in V_A \cup V_B\), \(u^* \sim u^*\) for \(u \in V_A\) and \(x \not\sim y\) in any other case. Define \(G = G(G_A, G_B)\), \(G' = G(G, G)\), \(F' = (F^A, F^B)\), \(F^B = \{(A, u, v) \in V \times V \mid u \sim v\}\) and \(F^{IA} = V(G') \setminus F^B\). We prove that \(S^B(G, F) = S^B(G', F') \cap V(G)\).

Define the potential \(p\) on \(G'\) as follows

\[
\begin{align*}
p(P, u, v) &= r^A_{G', F'}(P, u, v) + 1 & \text{for } (P, u, v) \in V(G) \\
p(B, v, u^*) &= 0 & \text{for } (A, u, v) \in F^A \\
p(P, x, y) &= \infty & \text{in any other case}
\end{align*}
\]

Observe that \(p\) is decreasing on \((G', F')\) for \(A\). Thus \(W^A(G, F) \subseteq \{(P, x, y) \mid p(P, x, y) < \infty\} \subseteq W^A(G', F')\). Now, define the set \(U \subseteq V(G')\) as follows

\[
\begin{align*}
(P, u, v) \in U & \text{ for } (P, u, v) \in S^B(G, F) \\
(B, v, u^*) \in U & \text{ for } (A, u, v) \in F^B \\
(A, u, u), (A, u^*, u^*) \in U & \text{ for } u \in V_A \\
(B, u, u') \in U & \text{ for } (u, u') \in E_A \\
(P, x, y) \notin U & \text{ in any other case}
\end{align*}
\]

Observe that \(U\) is stable on \((G', F')\) for \(B\). Thus \(S^B(G, F) \subseteq U \subseteq S^B(G', F')\).

The map \(f: V(G_A, G_B) \rightarrow V(G, G)\) is the inclusion. \(\square\)
Boolean matrix multiplication. Given an $n_1 \times n_2$ boolean matrix $B_1$ and an $n_2 \times n_3$ boolean matrix $B_2$, their boolean product is the $n_1 \times n_3$ boolean matrix $B_1 \ast B_2$ defined by:

$$(B_1 \ast B_2)[i, j] = \bigvee_{k=1}^{n_2} B_1[i, k] \land B_2[k, j].$$

The problem BOOLEAN MATRIX MULTIPLICATION (BMM) of size $n_1 \times n_2 \times n_3$ asks to compute $B_1 \ast B_2$ given $B_1$ and $B_2$. It is folklore that BMM can be reduced to a standard matrix multiplication over integers, of the same size. If $n_1, n_2, n_3 \leq n$ then BMM can be computed in $n^{\omega+o(1)}$ time, where $\omega < 2.4$ is the exponent of matrix multiplication \cite{7}. Moreover, a standard matrix multiplication can be verified in (randomized) $O(n^2)$ time \cite{9, 13, 14}. If the output of the standard matrix multiplication is provided as a certificate, then BMM can be also checked in $O(n^2)$ time.

Semi-boolean matrix multiplications. Given an $n_1 \times n_2$ matrix of numbers\footnote{Integers, reals, or elements of any totally ordered set.} $A$ and an $n_2 \times n_3$ boolean matrix $B$, their min- and max-semi-boolean products are the $n_1 \times n_3$ matrices $A \ast_{\text{min}} B$ and $A \ast_{\text{max}} B$ defined as follows:

$$(A \ast_{\text{min}} B)[i, j] = \min \{A[i, k] \mid k = 1, \ldots, n_2 \text{ and } B[k, j] \text{ is true}\},$$

$$ (A \ast_{\text{max}} B)[i, j] = \max \{A[i, k] \mid k = 1, \ldots, n_2 \text{ and } B[k, j] \text{ is true}\}. $$

The problems MIN- and MAX-SEMI-BOOLEAN MATRIX MULTIPLICATION (MSBMM) of size $n_1 \times n_2 \times n_3$ ask to compute $A \ast_{\text{min}} B$ and $A \ast_{\text{max}} B$ given $A$ and $B$. The min and max versions are clearly equivalent since $A \ast_{\text{min}} B = -(A \ast_{\text{max}} B)$.

In the DISTINCT variant of MSBMM, we require $A[i, k] \neq A[i, k']$ for $k \neq k'$. Observe that, to solve MSBMM, we can replace $A[i, k]$ with its rank in the set $\{A[i, k] \mid k = 1, \ldots, n_2\}$, breaking ties arbitrarily, and we get an equivalent DISTINCT MSBMM problem.

3 Acyclic Simulation

3.1 Hardness

Consider the two-pebble game $(G(G_A, G_B), F)$, where $G_P = (V_P, E_P)$ and $F = (F^A, F^B)$, defined as follows. Let $V_A = \{x_1, \ldots, x_n\} \cup \{y_1, \ldots, y_p\}$ and $V_B = \{z_1, \ldots, z_m\}$. Let $E_A = \{(x_i, y_k) \mid B_1[i, k] \text{ is true}\}$ and $E_B = \emptyset$. Finally, let $F^A = \{(B, z_j, y_k) \mid B_2[k, j] \text{ is true}\}$ and $F^B = V(G_A, G_B) \setminus F^A$.

Since $G_B$ has no edges, the only plays on $G$ from $(A, x_i, z_j)$ for $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$ are $(A, x_i, z_j) \downarrow$, and $(A, x_i, z_j) \rightarrow (B, z_j, y_k) \downarrow$ for $k \in \{1, \ldots, p\}$. Clearly, $A$ guarantees to win on $(G, F)$ from $(A, x_i, z_j)$ iff $(A, x_i, z_j) \rightarrow G(B, z_j, y_k)$ for some $k$ such that $(B, z_j, y_k) \in F^A$, that is, iff

$$\bigvee_{k=1}^{p} (x_i, y_k) \in E_A \land (B, z_j, y_k) \in F^A$$

$$= \bigvee_{k=1}^{p} B_1[i, k] \land B_2[k, j]$$

$$= (B_1 \ast B_2)[i, j].$$

We obtain the following.
Hence, we can compute Lemma 6. We obtain the following.

If any of $3.3$ Verification and certificates

Nevertheless, the whole set $U$ is computed in $O(\alpha)$ time, for some $\alpha \geq 2$, then Boolean Matrix Multiplication of size $n \times n \times n$ can be computed in $O(\alpha)$ time.

3.2 Employing boolean matrix multiplication

Let $(G, A, G_B, F)$ be a two-pebble game where $G_P = (V_P, E_P)$ for $P \in \{A, B\}$ and $|V_A|, |V_B| \leq n$. Let $G = G(A, G_B) = (V, E, A, V_B)$ be the corresponding two-pebble arena. For a given set $U \subseteq V$, we show how to compute $f_{G, F}^P(U) \cap V^P$ using a BMM. To obtain $f_{G, F}^P(U) \cap V^P$, first compute $f_{G, F}^P(V \setminus U) \cap V^P$, and then apply de Morgan laws:

$$f_{G, F}^P(U) \cap V^P = \left[ V \setminus f_{G, F}^P(V \setminus U) \right] \cap V^P = \left[ f_{G, F}^P(V \setminus U) \cap V^P \right].$$

Let $V_A = \{v_1^A, \ldots, v_{n_A}^A\}$ and $V_B = \{v_1^B, \ldots, v_{n_B}^B\}$. Define the following matrices:

$E_P$: an $n \times n$ boolean matrix where $E_P[i, j] = \text{true}$ if $v_i^P \rightarrow P v_j^P$.

$U_P$: an $n \times n$ boolean matrix where $U_P[i, j] = \text{true}$ if $(1 - P, v_j^P, v_i^P) \in U$.

We have

$$(P, v_i^P, v_j^P) \in f_{G, F}^P(U) \iff (P, v_i^P, v_j^P) \in F^P \cup \bigvee_{k=1}^{n_P} v_i^P \rightarrow P v_k^P \land (1 - P, v_j^P, v_k^P) \in U$$

$$\iff (P, v_i^P, v_j^P) \in F^P \cup \bigvee_{k=1}^{n_P} E_P[i, k] \land U_P[k, j]$$

$$\iff (P, v_i^P, v_j^P) \in F^P \cup (E_P \star U_P)[i, j].$$

Hence, we can compute $f_{G, F}^P(U) \cap V^P$ by computing the BMM $E_P \star U_P$ with only $O(n^2)$ overhead. We obtain the following.

**Lemma 6.** For a given set $U \subseteq V$, the set $f_{G, F}^P(U)$ can be computed by solving two BMMs of size at most $n \times n \times n$, and only $O(n^2)$ extra time.

3.3 Verification and certificates

If any of $G_A$ and $G_B$ is acyclic, then $G = G(A, G_B)$ is also acyclic and the set $S^P(G, F) = W^P(G, F)$ is the unique solution of the equation $f_{G, F}^P(U) = U$. To verify that $U = S^P(G, F)$ for a given set $U \subseteq V(G)$, it is sufficient to verify that this equation holds, which, by Lemma 6, is equivalent to verifying two BMMs.

This technique can be applied to verify (semi-)acyclic 2PWS, but cannot be used to verify acyclicity. Indeed, by Lemma 2, the solution of $\text{Simulation}$ coincides with the set $S^B(G, F) \cap V^A$ in the corresponding 2PWS problem, but does not provide the set $S^B(G, F) \cap V^B$. Nevertheless, the whole set $S^B(G, F)$ is a certificate for acyclicity that can be checked by verifying two BMMs. Moreover, if we provide the output of the corresponding standard integer matrix multiplications in the certificate, then it can be checked in (randomized) $O(n^2)$ time. This is summarized in the following.
Theorem 7. Acyclic and Semi-Acyclic 2PWS of size $n \times n$ can be verified by verifying two boolean matrix multiplications of size $n \times n \times n$, with only $O(n^2)$ extra time.

Acyclic Simulation of size $n$ and (semi-)Acyclic 2PWS of size $n \times n$ admit a $O(n^2)$-size certificate that can be verified by verifying two integer matrix multiplications of size $n \times n \times n$, which can be done in (randomized) $O(n^2)$ time.

3.4 Subcubic algorithm

In this section we give an algorithm for Acyclic Simulation of size $n$ running in time $n^{\omega+o(1)}$, where $\omega$ is the exponent of matrix multiplication.

Dicut decomposition of arenas. A dicut of a directed graph $(V, E)$ is a partition $(V_T, V_H)$ of $V$ such that there are no edges from $V_H$ to $V_T$, i.e., $E \cap (V_H \times V_T) = \emptyset$.

Let $G$ be an arena and $U \subseteq V(G)$. The sub-arena of $G$ induced by $U$ is $G[U]$ where $V^P(G[U]) = V^P(G) \cap U$ for $P \in \{A, B\}$ and $E(G[U]) = E(G) \cap (U \times U)$.

Consider a pseudo-infinite game $(G, F)$ on the arena $G = (V, E, \mathcal{V}^A, \mathcal{V}^B)$, and let $(V_T, V_H)$ be a dicut of $(V, E)$. Let $F^P_H = F^P \cap V_H$ for $P \in \{A, B\}$ and let $S^P_H = S^P(G[V_H], F_H)$.

Lemma 8. We have

$$S^P(G, F) = S^P_H \cup S^P(G|V_T, F_T)$$

where $F_T = f^P_{G,F}(S^P_H) \cap V_T$ and $F^{-1}_T = V_T \setminus F_T$.

Proof. Let $G_H := G[V_H]$, $G_T := G[V_T]$, $S^P := S^P(G, F)$ and $S^P_T := S^P(G_T, F_T)$.

For any $U \subseteq V$ and $\sigma \in V_H \cap V^P$ we have

$$\sigma \in f^P_{G_H,F_H}(U \cap V_H) \iff \sigma \in F^P_H \lor \biglor_{\sigma \rightarrow \sigma'} \sigma' \in U \cap V_H$$

$$\iff \sigma \in F^P \lor \biglor_{\sigma \rightarrow \sigma'} \sigma' \in U$$

$$\iff \sigma \in f^P_{G,F}(U)$$

so $f^P_{G_H,F_H}(U \cap V_H) \cap V^P = f^P_{G,F}(U) \cap V_H \cap V^P$. We obtain $f^P_{G_H,F_H}(U \cap V_H) = f^P_{G,F}(U) \cap V_H$ by applying de Morgan laws.

For any $U_T \subseteq V_T$ and $\sigma \in V_T \cap V^P$ we have

$$\sigma \in f^P_{G_T,F_T}(U_T) \iff \sigma \in F^P \lor \biglor_{\sigma \rightarrow \sigma'} \sigma' \in U_T$$

$$\iff \sigma \in f^P_{G,F}(S^P_H) \lor \biglor_{\sigma \rightarrow \sigma'} \sigma' \in U_T$$

$$\iff \sigma \in F^P \lor \biglor_{\sigma \rightarrow \sigma'} \sigma' \in S^P_H \lor \biglor_{\sigma \rightarrow \sigma'} \sigma' \in U_T$$

$$\iff \sigma \in F^P \lor \biglor_{\sigma \rightarrow \sigma'} \sigma' \in U_T \cup S^P_H$$

$$\iff \sigma \in f^P_{G,F}(U_T \cup S^P_H)$$

so $f^P_{G_T,F_T}(U_T) \cap V^P = f^P_{G,F}(U_T \cup S^P_H) \cap V_T \cap V^P$. We obtain $f^P_{G_T,F_T}(U_T) = f^P_{G,F}(U_T \cup S^P_H) \cap V_T$ by applying de Morgan laws.

We prove the following facts:
1. \( S^P_H \cup S^P_T \) is stable on \((G, \mathcal{F})\):

\[
S^P_H \cup S^P_T = f_{\mathcal{F}_H}^P(S^P_H) \cup f_{\mathcal{F}_T}^P(S^P_T) \\
= [f_{\mathcal{F}_H}^P(S^P_H) \cap \mathcal{V}_H] \cup [f_{\mathcal{F}_T}^P(S^P_T \cup S^P_H) \cap \mathcal{V}_T] \\
\subseteq f_{\mathcal{F}_T}^P(S^P_T \cup S^P_H)
\]

so \( S^P_H \cup S^P_T \subseteq S^P \).

2. \( S^P \cap \mathcal{V}_H \) is stable on \((G_H, \mathcal{F}_H)\):

\[
S^P \cap \mathcal{V}_H = f_{\mathcal{F}_H}^P(S^P) \cap \mathcal{V}_H \\
= f_{\mathcal{F}_H}^P(S^P) \cap \mathcal{V}_H
\]

so \( S^P \cap \mathcal{V}_H \subseteq S^P_H \) and, together with (1), \( S^P \cap \mathcal{V}_H = S^P_H \).

3. \( S^P \cap \mathcal{V}_T \) is stable on \((G_T, \mathcal{F}_T)\):

\[
S^P \cap \mathcal{V}_T = f_{\mathcal{F}_T}^P(S^P) \cap \mathcal{V}_T \\
= f_{\mathcal{F}_T}^P(S^P) \cap \mathcal{V}_T \\
= f_{\mathcal{F}_T}^P(S^P \cap \mathcal{V}_T)
\]

so \( S^P \cap \mathcal{V}_T \subseteq S^P_T \) and, together with (1), \( S^P \cap \mathcal{V}_T = S^P_T \).

The equality \( S^P = S^P_H \cup S^P_T \) follows.

**Algorithm for two-pebble games.** Let \((V^T_A, V^H_A)\) be a dicut of \(G_A\) such that \(|V^T_A|, |V^H_A| \leq \lfloor n/2 \rfloor\). Such a dicut can be easily obtained from a topological sort of \(G_A\), splitting at about half. Observe that the dicut \((V^T_A, V^H_A)\) induces a dicut \((\mathcal{V}_T, \mathcal{V}_H)\) of \(G = G(G_A, G_B)\) where \(\mathcal{V}_X = \mathcal{V}(G_A[V^X_A], G_B)\) for \(X \in \{T, H\}\). To compute \(S^P(G, \mathcal{F})\), we first compute \(S^P_H := S^P(G[\mathcal{V}_H], \mathcal{F}_H)\) recursively. Then, we compute \(f_{\mathcal{F}_T}^P(S^P_H)\) with two BMMs and set \(\mathcal{F}_T^P = f_{\mathcal{F}_T}^P(S^P_H) \cap \mathcal{V}_T\). Next, we compute \(S^P(G[\mathcal{V}_T], \mathcal{F}_T)\) recursively. Finally, we apply the formula

\[
S^P(G, \mathcal{F}) = S^P_H \cup S^P(G[\mathcal{V}_T], \mathcal{F}_T)
\]

given by Lemma 8.

At each recursive call we swap the players \(A\) and \(B\), so that the running time \(T(n_A, n_B)\) satisfies the recurrence

\[
T(n_A, n_B) \leq 2 \cdot T(n_A, \lfloor n_A/2 \rfloor) + (n_A + n_B)^{\omega+o(1)} \\
\leq 4 \cdot T(\lfloor n_A/2 \rfloor, \lfloor n_B/2 \rfloor) + (n_A + n_B)^{\omega+o(1)}.
\]

Under the assumption \(n_A, n_B \leq n\) and \(\omega \geq 2\), we get \(T(n) := T(n, n) \leq n^{\omega+o(1)}\). (If \(\omega = 2\), we get an extra logarithmic factor which is accounted for in the \(n^{o(1)}\) term.)

**Theorem 9.** Acyclic Simulation and Acyclic Two-Pebble Winning Set can be computed in \(n^{\omega+o(1)}\) time, for any \(\omega\) such that boolean matrix multiplication can be solved in \(n^{\omega+o(1)}\) time.
4 Cyclic Simulation

4.1 Certificates

Recall that the winning rank \( r_{G,F}^P \) is the only solution of the equation \( r_{G,F}^P = g_{G,F}^P(r_{G,F}^P) \), so it can be verified by checking that this equation holds.

We show that the computation of \( g_{G,F}^P \) for a two-pebble arena \( G = G(A,G,B) \) can be transformed into a pair of DISTINCT MSBMM. Let \( V_A = \{ v_1^A, \ldots, v_{n_A}^A \} \) and \( V_B = \{ v_1^B, \ldots, v_{n_B}^B \} \). For \( P \in \{ A,B \} \), define the following matrices:

- **\( E_P \):** an \( n_P \times n_P \) boolean matrix where \( E_P[i,j] = \text{true} \) if \( v_i^P \rightarrow P v_j^P \).
- **\( P_P \):** an \( n_P \times n_{1-P} \) matrix where \( P_P[i,j] = \min \{ (1 - P)^{-1} v_i^{1-P}, v_j^P \} \).

For \( (P,v_i^P,v_j^{1-P}) \in F^{1-P} \) we have \( h_F^P = \infty \) and

\[
    g_{G,F}^P(p)(P,v_i^P,v_j^{1-P}) = \min_{v_i' \rightarrow P v_j'^P} p(1 - P) v_i^{1-P}, v_j^P = \min \{ P_P[k,j] \mid k = 1, \ldots, n_P \text{ and } E_P[i,k] = \text{true} \} = (P_P \ast \min E_P)[i,j]
\]

and for \( (P,v_i^P,v_j^{1-P}) \in F^P \) we have \( h_F^P = 0 \) and

\[
    g_{G,F}^{1-P}(p)(P,v_i^P,v_j^{1-P}) = \max_{v_i' \rightarrow P v_j'^P} p(1 - P) v_i^{1-P}, v_j^P = \max \{ P_P[k,j] \mid k = 1, \ldots, n_P \text{ and } E_P[i,k] = \text{true} \} = (P_P \ast \max E_P)[i,j].
\]

Notice that all the other cases are either trivial or can be reduced to one of the two above. By transforming the multiplications \( P_P \ast \min E_P \) and \( P_P \ast \max E_P \) to their DISTINCT version, we obtain the following.

**Theorem 10.** Simulation of size \( n \) and Two-Pebble Winning Set of size \( n \times n \) admit \( O(n^2) \)-size certificates that can be verified by verifying two DISTINCT Max-Semi-Boolean Matrix Multiplications of size \( n \times n \times n \), and only \( O(n^2) \) extra time.

4.2 Hardness

In this section we present a reduction from the problem of verifying DISTINCT Max-Semi-Boolean Matrix Multiplication to the problem of verifying Two-Pebble Winning Set.

An \( m \times m \) boolean matrix \( B \) and two \( n \times m \) matrices of numbers \( A \) and \( C \) are given, where \( A[i,k] \neq A[i,k'] \) for \( k \neq k' \). We want to check that, for every \( i \) and \( j \),

\[
    C[i,j] = (A \ast \max B)[i,j] = \max \{ A[i,k] \mid k = 1, \ldots, p \text{ and } B[k,j] \text{ is true} \}.
\]

Fixed \( 1 \leq i \leq n \) and \( 1 \leq j \leq m \), let \( k_{ij} \) be the only index such that \( A[i,k_{ij}] = C[i,j] \). If there is no such \( k_{ij} \), or \( B[k_{ij},j] \) is false, then clearly the answer is no. Otherwise, \( C[i,j] \leq (A \ast \max B)[i,j] \) for every \( i,j \). It remains to check that there is no triple \( (i,j,k) \) such that \( A[i,k] > C[i,j] \) with \( B[k,j] \) true. We call such a triple an invalid triangle.

We exhibit a two-pebble game \( (G_A,G_B,F) \) where Bob survives on some initial configurations iff there exists an invalid triangle. Let \( G_P = (V_P,E_P) \) and define \( V_A = \{1, \ldots, n\} \) and \( E_A = \{(i,i) \mid i \in V_A\} \).
Permutation networks. To describe the graph $G_B$, we first need to introduce the concept of permutation network. A permutation network \cite{19} of size $n$ has $n$ inlets $u_1,\ldots,u_n$, $n$ outlets $v_1,\ldots,v_n$ and a set of gates $S$. For each gate $s \in S$, there are four ports: two input ports $x_1^s, x_2^s$ and two output ports $y_1^s, y_2^s$. Let $O = \{u_1,\ldots,u_n\} \cup \bigcup_{s \in S} \{y_1^s, y_2^s\}$ and $I = \bigcup_{s \in S} \{x_1^s, x_2^s\} \cup \{v_1,\ldots,v_n\}$. The network has a set of wires $W \subseteq O \times I$, which form a bijective relation between $O$ and $I$. Finally, there is a function $f$ that takes in input a permutation $\pi: \{1,\ldots,n\} \to \{1,\ldots,n\}$ and output the subset of gates $f(\pi) \subseteq S$ which are active on permutation $\pi$. A gate connects each of the two inputs to an output: when a gate is active, its inputs get swapped. Given $\pi$, we define the directed graph $G_\pi = (I \cup O, W \cup T_\pi)$, where $T_\pi \subseteq I \times O$ contains the pairs of the form $(x_i^s,y_j^s)$ for $s \in S$ and $i,j \in \{1,2\}$ with $i = j$ if $s$ is inactive on $\pi$ and $i \neq j$ is $s$ is active on $\pi$. The property of the network is that for every permutation $\pi$, the graph $G_\pi$ is composed of $n$ vertex-disjoint paths $P_1,\ldots,P_n$, where $P_i$ goes from $u_i$ to $v_{\pi(i)}$. Waksman \cite{19} shows a construction of permutation networks of size $n = 2^k$ where $|S| = O(n \log n)$ and $f$ is computable in $O(n \log n)$ time.

Gadget graphs. Take a permutation network of size $n$. For every gate $s \in S$ we define the gate gadget graph (Fig. 1) as follows. The graph has eight vertices and eight edges: two input vertices $x_1^s$ and $x_2^s$ and two output vertices $y_1^s$ and $y_2^s$, corresponding to the ports of $s$, four guard vertices $z_{j,k}^s$ and eight edges $x_i^s \to z_{j,k}^s$ and $z_{j,k}^s \to y_i^s$, for $j,k \in \{1,2\}$. For a given permutation $\pi$, we define $K^s(\pi) = \{z_{11}^s, z_{22}^s\}$ if $s$ is active for $\pi$ and $K^s(\pi) = \{z_{12}^s, z_{21}^s\}$ otherwise. Observe that the only maximal paths in the gadget non passing through $K^s(\pi)$ are $x_1^s z_{12}^s y_2^s$, $x_2^s z_{21}^s y_1^s$ if $s$ is active and $x_1^s z_{11}^s y_1^s$, $x_2^s z_{22}^s y_2^s$ otherwise.

The permutation gadget graph $X$ of size $n$ (Fig. 2) contains the inlets $u_1,\ldots,u_n$ and the outlets $v_1,\ldots,v_n$ as vertices, a gate gadget for each gate $s \in S$, and all the wires $W$ as extra edges. For a given permutation $\pi$, we define $K^X(\pi) = \bigcup_{s \in S} K^s(\pi)$. Observe that the only maximal paths in the graph not passing through $K^X(\pi)$ are $P_1,\ldots,P_n$ where $P_i$ goes from $u_i$ to $v_{\pi(i)}$.

Game construction. We identify the $k$-th columns of $A$ with $\ell = k \in \{1,\ldots,m\}$ and the $j$-th column of $C$ with $\ell = m+j \in \{m+1,\ldots,2m\}$. For every $i$, let $\pi_i: \{1,\ldots,2m\} \to \{1,\ldots,2m\}$ be a permutation that sorts the indices $\ell \in \{1,\ldots,2m\}$ according to the value $A[i,k]$ for $\ell = k \in$
Figure 3: A depiction of the graph $G_B$ for $m = 4$. The dotted lines within each permutation gadget $X \in \{Y, Z\}$ represent maximal paths not passing through $K^X(\pi_i^X)$, and depend on the vertex $i$ in the graph $G_A$ of Alice where her pebble is located. The dashed line is an example of edge $w_{m+j} \rightarrow x_k$, for $j = 2$ and $k = 3$, which is present only if $B[3, 2]$ is true. The actual graph has a similar edge for every $k$ and $j$ such that $B[k, j]$ is true.

{1, \ldots, m} and $C[i, j]$ for $\ell = m + j \in \{m + 1, \ldots, 2m\}$, breaking ties in favor of $A$. Namely, $\pi_i$ is such that $A[i,k] > C[i,j]$ implies $\pi_i(k) > \pi_i(m+j)$ and $A[i,k] \leq C[i,j]$ implies $\pi_i(k) < \pi_i(m+j)$.

The graph $G_B$ contains distinct vertices $x_\ell, y_\ell, z_\ell, w_\ell$ for each $\ell \in \{1, \ldots, 2m\}$ and the following objects.

- A permutation gadget $Y$ of size $2m$, associated with the permutation family $\pi_i^Y := \pi_i$, with inlets $x_1, \ldots, x_{2m}$ and outlets $y_1, \ldots, y_{2m}$.
- For each $1 \leq \ell, \ell' \leq 2m$, the edge $y_\ell \rightarrow z_{\ell'}$ if $\ell' \leq \ell$.
- A permutation gadget $Z$ of size $2m$, associated with the family of inverse permutations $\pi_i^Z := \pi_i^{-1}$, with inlets $z_1, \ldots, z_{2m}$ and outlets $w_1, \ldots, w_{2m}$.
- For each $1 \leq j, k \leq m$, the edge $w_{m+j} \rightarrow x_k$ if $B[k, j]$ is true.

Then, we define $\mathcal{F}^A = \{(B) \times V_B \times V_A\} \cup \{(A,i,u) \mid u \in K^Y(\pi_i^Y) \cup K^Z(\pi_i^Z)\}$. The reduction is complete.

**Lemma 11.** If $A, B, C$ comprise an invalid triangle $(i,j,k)$, then $(A,i,x_k) \in S^B(G, F)$.

**Proof.** We provide the strategy $s_B$ with which Bob guarantees to survive from $(A,i,x_k)$. For each configuration $(B,u,i)$, if $u$ is a non-last vertex in a maximal path $P$ not passing through $K^X(\pi_i^X)$ in the permutation gadget $X \in \{Y, Z\}$, then let $s_B(B,u,i) = (A,i,u')$ where $u'$ is the next vertex in the path. Next, let $s_B(B,y_{\pi_i(k)},i) = (A,i,z_{\pi_i(m+j)})$, possible since $A[i,k] > C[i,j]$ so $\pi_i(k) > \pi_i(m+j)$, and $s_B(B,w_{m+k},i) = (A,i,x_j)$, possible since $B[k,j]$ is true. In all other cases, stop.

Following this strategy, Bob moves from $x_k$ along the permutation gadget $Y$ until he reaches $y_{\pi_i(k)}$. Then, he moves to $z_{\pi_i(m+j)}$. Next, he moves along the permutation gadget $Z$ going from $z_{\pi_i(m+j)}$ to $w_{\pi_i^{-1}(\pi_i(m+j))} = w_{m+j}$. Finally, he moves back to $x_k$, closing a cycle. Since Bob never moves to a configuration in $\mathcal{F}^A$, he survives. \qed
Lemma 12. If \( A, B, C \) do not comprise an invalid triangle, then \( \mathcal{W}^A(\mathcal{G}, \mathcal{F}) = \mathcal{V}(\mathcal{G}) \).

Proof. Consider the strategy \( s_A \) where \( s_A(A, i, u) = \bot \) if \( (A, i, u) \in \mathcal{F}^A \) and \( s_A(A, i, u) = (B, u, i) \) otherwise. Take any play \( \pi \) consistent with \( s_A \). If \( \pi \downarrow \), since \( \mathcal{F}^B \subseteq \mathcal{V}^A(\mathcal{G}) \), then \( \pi \downarrow \mathcal{F}^A \) and Alice wins. Otherwise, \( \pi \) is infinite and never passes through a configuration in \( \mathcal{V}^A(\mathcal{G}) \cap \mathcal{F}^A \). We define a potential \( p: \mathcal{V}(\mathcal{G}) \to \mathbb{N} \) and show that \( p \) is non increasing along \( \pi \) and strictly decreases frequently, a contradiction.

Let \( P^Y_{i\ell} \) be the only maximal path in \( Y \) that goes from \( x_\ell \) to \( y_{\pi_i(\ell)} \) and does not contain any \( u \in K^Y(\pi^Y_i) \). For every vertex \( u \) along \( P^Y_{i\ell} \) (including \( x_\ell \) and \( y_{\pi_i(\ell)} \)), let \( p(A, i, u) = p(B, u, i) = \pi_i(\ell) \).

Let \( P^Z_{i\ell} \) be the only maximal path in \( Z \) that goes from \( z_{\pi_i(\ell)} \) to \( w_\ell \) and does not contain any \( u \in K^Z(\pi^Z_i) \). For every vertex \( u \) along \( P^Z_{i\ell} \) (including \( z_{\pi_i(\ell)} \) and \( w_\ell \)), let \( p(A, i, u) = p(B, u, i) = \pi_i(\ell) \).

The only possible moves are either along a path \( P^Y_{i\ell} \) or \( P^Z_{i\ell} \), where the potential remains constant by definition, or fall into one of the following two types:

1. \((B, y_\ell, i) \to (A, i, z_\ell)\) with \( p(A, i, z_\ell) = \ell' \leq \ell = p(B, y_\ell, i)\),

2. \((B, w_{m+j}, i) \to (A, i, x_k)\) with \( B[k,j] \) true.

In moves of type 2, we have \( p(B, w_{m+k}, i) = \pi_i(m+k) \) and \( p(A, i, x_j) = \pi_i(j) \). Since there are no invalid triangles and \( B[k,j] \) is true, necessarily \( A[i,k] \leq C[i,j] \) so \( \pi_i(m+k) < \pi_i(j) \). Furthermore, moves of type 2 occur frequently, since without these moves the configuration graph becomes acyclic. \[\Box\]

From Lemma 11 and Lemma 12 we obtain the following.

Theorem 13. If Two-Pebble Winning Set of size \( n_A \times n_B \) can be computed or verified in \( T(n_A, n_B) \) time, then Distinct Max-Semi-Boolean Matrix Multiplication of size \( n \times m \times m \) can be verified in \( O(T(n, m \log m)) \) time.

In particular, if Simulation of size \( n \) can be computed or verified in \( O(n^\alpha) \) time for some \( \alpha \geq 2 \), then Distinct Max-Semi-Boolean Matrix Multiplication of size \( n \times n \times n \) can be verified in \( O(n^\alpha \log n) \) time.

Note

After the submission of this manuscript for review, we found out about an \( O(n^{2+\omega/3}) \)-time algorithm [18] and a subsequent \( O(n^{(3+\omega)/2}) \)-time algorithm [8] to compute the product between two \( n \times n \) matrices over the \((\max, \min)\) semi-ring. Since our max-semi-boolean matrix multiplication can be reduced to a \((\max, \min)\)-product, where one matrix contains only \( +\infty \) and \(-\infty \) entries, an \( O(n^{(3+\omega)/2}) \)-time algorithm for MSBMM can be obtained. By what discussed in this document, this implies that, even on cyclic structures, the simulation preorder admits certificates that can be checked in truly subcubic \( O(n^{(3+\omega)/2}) \leq O(n^{2.792}) \) time.
References

[1] Bard Bloom. Ready simulation, bisimulation, and the semantics of CCS-like languages. 1989.

[2] Bard Bloom and Robert Paige. Transformational design and implementation of a new efficient solution to the ready simulation problem. *Science of Computer Programming*, 24(3):189–220, June 1995.

[3] Doron Bustan and Orna Grumberg. Simulation-based minimization. *ACM Transactions on Computational Logic*, 4(2):181–206, 2003.

[4] Gérard Cécé. Three simulation algorithms for labelled transition systems. pages 1–26, January 2013.

[5] Rance Cleaveland, Joachim Parrow, and Bernhard Steffen. The concurrency workbench: a semantics-based tool for the verification of concurrent systems. *ACM Transactions on Programming Languages and Systems*, 15(1):36–72, January 1993.

[6] Rance Cleaveland and Bernhard Steffen. A linear-time model-checking algorithm for the alternation-free modal μ-calculus. *Formal Methods in System Design*, 2(2):121–147, April 1993.

[7] Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. In *Proceedings of the nineteenth annual ACM conference on Theory of computing - STOC '87*, pages 1–6, New York, New York, USA, 1987. ACM Press.

[8] Ran Duan and Seth Pettie. Fast algorithms for (max, min)-matrix multiplication and bottleneck shortest paths. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '09, pages 384–391, Philadelphia, PA, USA, 2009. Society for Industrial and Applied Mathematics.

[9] Rusins Freivalds. Probabilistic machines can use less running time. In *Information Processing 77, Proceedings of IFIP Congress 77*, pages 839–842, 1977.

[10] Raffaella Gentilini, Carla Piazza, and Alberto Policriti. Simulation as coarsest partition problem. In *Tools and Algorithms for the Construction*, pages 415–430. 2002.

[11] Raffaella Gentilini, Carla Piazza, and Alberto Policriti. From bisimulation to simulation: Coarsest partition problems. *Journal of Automated Reasoning*, 31(1):73–103, 2003.

[12] Monika R. Henzinger, Thomas A. Henzinger, and Peter W. Kopke. Computing simulations on finite and infinite graphs. In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pages 453–462. IEEE Comput. Soc. Press, 1995.

[13] Tracy Kimbrel and Rakesh Kumar Sinha. A probabilistic algorithm for verifying matrix products using $O(n + 2)$ time and $\log 2n + O(1)$ random bits. *Information Processing Letters*, 45(2):107–110, February 1993.

[14] Ivan Korec and Jiří Wiedermann. Deterministic verification of integer matrix multiplication in quadratic time. In *SOFSEM 2014: Theory and Practice of Computer*, pages 375–382. 2014.

[15] Jasen Markovski. Saving time in a space-efficient simulation algorithm. In *2011 11th International Conference on Quality Software*, pages 244–251. IEEE, July 2011.
[16] Francesco Ranzato and Francesco Tapparo. A new efficient simulation equivalence algorithm. In 22nd Annual IEEE Symposium on Logic in Computer Science (LICS 2007), pages 171–180. IEEE, 2007.

[17] Francesco Ranzato and Francesco Tapparo. An efficient simulation algorithm based on abstract interpretation. *Information and Computation*, 208(1):1–22, January 2010.

[18] Virginia Vassilevska, Ryan Williams, and Raphael Yuster. All pairs bottleneck paths and max-min matrix products in truly subcubic time. *Theory of Computing*, 5(1):173–189, 2009.

[19] Abraham Waksman. A permutation network. *Journal of the ACM*, 15(1):159–163, 1968.