On the Complexity of Existential Pebble Games

Phokion G. Kolaitis and Jonathan Panttaja ⋆

Computer Science Department
University of California, Santa Cruz
kolaitis@cs.ucsc.edu jpanttaj@cs.ucsc.edu

Abstract. Existential k-pebble games, $k \geq 2$, are combinatorial games played between two players, called the Spoiler and the Duplicator, on two structures. These games were originally introduced in order to analyze the expressive power of Datalog and related infinitary logics with finitely many variables. More recently, however, it was realized that existential k-pebble games have tight connections with certain consistency properties which play an important role in identifying tractable classes of constraint satisfaction problems and in designing heuristic algorithms for solving such problems. Specifically, it has been shown that strong k-consistency can be established for an instance of constraint satisfaction if and only if the Duplicator has a winnning strategy for the existential k-pebble game between two finite structures associated with the given instance of constraint satisfaction. In this paper, we pinpoint the computational complexity of determining the winner of the existential k-pebble game. The main result is that the following decision problem is EXPTIME-complete: given a positive integer k and two finite structures $\mathbf{A}$ and $\mathbf{B}$, does the Duplicator win the existential k-pebble game on $\mathbf{A}$ and $\mathbf{B}$? Thus, all algorithms for determining whether strong k-consistency can be established (when k is part of the input) are inherently exponential.

1 Introduction and Summary of Results

Combinatorial games are a basic tool for analyzing logical definability and delineating the expressive power of various logics. Typically, a logic L is decomposed into a union $L = \bigcup_{k \geq 1} L(k)$ of fragments according to some syntactic parameter, such as quantifier rank, pattern of quantification, or number of variables. With each fragment $L(k)$, one then seeks to associate a natural combinatorial game $\mathcal{G}(k)$ that captures $L(k)$-equivalence. Specifically, the desired game $\mathcal{G}(k)$ is played between two players, called the *Spoiler* and the *Duplicator*, on two structures $\mathbf{A}$ and $\mathbf{B}$, and has the following property: the Duplicator has a winning strategy for $\mathcal{G}(k)$ on $\mathbf{A}$ and $\mathbf{B}$ if and only if $\mathbf{A}$ and $\mathbf{B}$ satisfy the same $L(k)$-sentences. In the case of first-order logic FO, each such fragment is the set $\mathrm{FO}(k)$ of all first-order sentences of quantifier rank at most k, and the game $\mathcal{G}(k)$ is the k-move Ehrenfeucht-Fraïssé-game. Moreover, in the case of the infinitary logic $\mathrm{L}^{\omega}_{\infty\omega}$ with finitely many variables, each such fragment is the infinitary logic $\mathrm{L}^{k}_{\infty\omega}$ with k variables, $k \geq 1$, and the corresponding game is the k-pebble game. As is well known, k-pebble games have turned out to be an indispensable tool in the study of logics with fixed-point operators in finite model theory (see [7] for a survey).

⋆ Research of both authors was partially supported by NSF grant IIS-9907419.

Each game $\mathcal{G}(k)$ as above gives rise to the decision problem of determining the winner of this game: given two finite structures $\mathbf{A}$ and $\mathbf{B}$, does the Duplicator have a winning strategy for $\mathcal{G}(k)$ on $\mathbf{A}$ and $\mathbf{B}$? It is easy to show that, for every $k \geq 1$, determining the winner of the k-move Ehrenfeucht-Fraïssé-game is in LOGSPACE (this is a consequence of the fact that each equivalence class of $\mathrm{FO}(k)$-equivalence is first-order definable). The state of affairs, however, is quite different for the k-pebble games. Indeed, Grohe [7] established that, for each $k \geq 2$, determining the winner of the k-pebble game is a P-complete problem, that is, complete for polynomial-time under logarithmic-space reductions. It is also natural to consider the decision problem that arises by taking the parameter k as part of the input (in addition to the structures $\mathbf{A}$ and $\mathbf{B}$). Pezzoli [13] investigated the computational complexity of this problem for the Ehrenfeucht-Fraïssé-game and showed that it is PSPACE-complete. In other words, Pezzoli showed that the following problem is PSPACE-complete: given a positive integer k and two finite structures $\mathbf{A}$ and $\mathbf{B}$, does the Duplicator have a winning strategy for the k-move Ehrenfeucht-Fraïssé-game on $\mathbf{A}$ and $\mathbf{B}$? Thus, when the number of moves is part of the input, an exponential jump occurs in determining the winner of the Ehrenfeucht-Fraïssé-game. It is conjectured that a similar exponential jump in complexity holds for the k-pebble game, when k is part of the input. Specifically, the conjecture is that the following problem is EXPTIME-complete: given a positive integer k and two finite structures $\mathbf{A}$ and $\mathbf{B}$, does the Duplicator have a winning strategy for the k-pebble on $\mathbf{A}$ and $\mathbf{B}$? To date, this conjecture remains unsettled.

In this paper we investigate the computational complexity of the decision problems associated with the class of existential k-pebble games (or, in short, $(\exists, k)$-pebble games), which are an asymmetric variant of the k-pebble games. These games were introduced in [11] as a tool for studying the expressive power of Datalog and of the existential positive infinitary logic $\exists\mathrm{L}^{\omega}_{\infty\omega}$ with finitely many variables. More precisely, $\exists\mathrm{L}^{\omega}_{\infty\omega}$ is the collection of all $\mathrm{L}^{\omega}_{\infty\omega}$-formulas containing all atomic formulas and closed under existential quantification, infinitary conjunction $\bigwedge$, and infinitary disjunction $\bigwedge$. Clearly, $\exists\mathrm{L}^{\omega}_{\infty\omega} = \bigcup_{k\geq 1} \exists\mathrm{L}^{k}_{\infty\omega}$, where $\exists\mathrm{L}^{k}_{\infty\omega}$ is the collection of all $\exists\mathrm{L}^{\omega}_{\infty\omega}$-formulas with at most k distinct variables. The differences between the $(\exists, k)$-pebble game and the k-pebble game played on two structures $\mathbf{A}$ and $\mathbf{B}$ are that in the $(\exists, k)$-pebble game: (1) the Spoiler always plays on $\mathbf{A}$; and (2) the Duplicator strives to maintain a partial homomorphism, instead of a partial isomorphism. The main result of this paper is that determining the winner of the $(\exists, k)$-pebble game, when k is part of the input, is an EXPTIME-complete problem. In contrast, for each fixed $k \geq 2$, determining the winner of $(\exists, k)$-pebble game turns out to be a P-complete problem. Before commenting on the technique used to establish the main result, we discuss the motivation for investigating this problem and the implications of our main result.

Although $(\exists, k)$-pebble games were originally used in database theory and finite model theory, in recent years they turned out to have applications to the study of constraint satisfaction. Numerous problems in several different areas of artificial intelligence and computer science can be modeled as constraint satisfaction problems [4]. In full generality, an instance of the CONSTRAINT SATISFACTION PROBLEM consists of a set of variables, a set of possible values, and a set of constraints on tuples of variables; the question is to determine whether there is an assignment of values to the variables that satisfies the given constraints. Alternatively, as first pointed out by Feder and Vardi [6], the CONSTRAINT SATISFACTION PROBLEM can be identified with the HOMOMORPHISM PROBLEM: given two relational structures $\mathbf{A}$ and $\mathbf{B}$, is there a homomorphism h from $\mathbf{A}$ to $\mathbf{B}$? Intuitively, the structure $\mathbf{A}$ represents the variables and the tuples of variables that participate in constraints, the structure $\mathbf{B}$ represents the domain of values and the tuples of values that the constrained tuples of variables are allowed to take, and the homomorphisms from $\mathbf{A}$ to $\mathbf{B}$ are precisely the assignments of values to variables that satisfy the constraints. The CONSTRAINT SATISFACTION PROBLEM is NP-complete, since it contains BOOLEAN SATISFIABILITY, COLORABILITY, CLIQUE, and many other prominent NP-complete problems as special cases. For this reason, there has been an extensive pursuit of both tractable cases of the CONSTRAINT SATISFACTION PROBLEM and heuristic algorithms for this problem. In this pursuit, a particularly productive approach has been the introduction and systematic use of various *consistency* concepts that make explicit additional constraints implied by the original constraints. The *strong k-consistency* property is the most important one among them; intuitively, this property holds when every partial solution on fewer than k variables can be extended to a solution on k variables [5]. Closely related to this is the process of "establishing strong k-consistency", which is the question of whether additional constraints can be added to a given instance of the CONSTRAINT SATISFACTION PROBLEM in such a way that the resulting instance is strongly k-consistent and has the same space of solutions as the original one. Algorithms for establishing strong k-consistency play a key role both in identifying tractable cases of constraint satisfaction and in designing heuristics for this class of problems [2,5].

In [12], a tight connection was shown to exist between strong k-consistency properties and $(\exists, k)$-pebble games. Specifically, it turns out that strong k-consistency can be established for a given instance of the CONSTRAINT SATISFACTION PROBLEM if and only if the Duplicator has a winning strategy for the $(\exists, k)$-pebble game on the structures $\mathbf{A}$ and $\mathbf{B}$ forming the instance of the HOMOMORPHISM PROBLEM that is equivalent to the given instance of the CONSTRAINT SATISFACTION PROBLEM. This connection was fruitfully exploited in [3], where it was shown that the tractability of certain important cases of constraint satisfaction follows from the fact that the existence of a solution is equivalent to whether the Duplicator can win the $(\exists, k)$-pebble game for some fixed k. Note that, for every fixed k, there is a polynomial-time algorithm to determine whether, given two finite structures $\mathbf{A}$ and $\mathbf{B}$, the Duplicator has a winning strategy for the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$ (this had been already observed in [11]). Nonetheless, since many heuristics for constraint satisfaction require testing whether strong k-consistency can be established for arbitrarily large k's, it is important to identify the inherent computational complexity of determining the winner in the $(\exists, k)$-pebble game, when k is part of the input. It is not hard to verify that this problem is solvable in time $O(n^{2k})$, that is, in time exponential in k. Moreover, it was conjectured in [12] that a matching lower bound exists, which means that the following problem is EXPTIME-complete: given a positive integer k and two finite structures $\mathbf{A}$ and $\mathbf{B}$, does the Duplicator have a winning strategy for the $(\exists, k)$-pebble on $\mathbf{A}$ and $\mathbf{B}$?

In this paper, we prove this conjecture by showing that another pebble game, which was known to be EXPTIME-complete, has a polynomial-time reduction to the $(\exists, k)$-

pebble game. Specifically, Kasai, Adachi, and Iwata [8] introduced a pebble game, which we will call the KAI game, and showed that it is EXPTIME-complete via a direct reduction from polynomial-space alternating Turing machines (recall that $\mathrm{APSPACE} = \mathrm{EXPTIME}$ [1]). Our reduction of the KAI game to the $(\exists, k)$-pebble game is quite involved and requires the construction of elaborate combinatorial gadgets. In describing this reduction and establishing its correctness, we will adopt the setup and terminology used by Grohe [7] in showing that, for every $k \geq 2$, the k-pebble game is P-complete. Some of the basic gadgets in our reduction already occurred in Grohe's reduction. However, we will also need to explicitly construct other much more sophisticated gadgets that will serve as "switches" with special properties in the reduction. We note that Grohe also used highly sophisticated gadgets that were graphs with certain homogeneity properties. Grohe's gadgets, however, have size exponential in k and, hence, they cannot be used in a polynomial-time reduction when k is part of the input (this is also the reason why Grohe's reduction does not show that the k-pebble game is EXPTIME-complete, when k is part of the input). An immediate consequence of our main result is that determining whether strong k-consistency can be established, when k is part of the input, is an EXPTIME-complete problem and, thus, inherently exponential. Moreover, this explains why all known algorithms for establishing strong k-consistency are exponential in k (even ones considered to be "optimal", see [2]).

We also address the computational complexity of determining who wins the $(\exists, k)$-pebble game, when k is a fixed positive integer. Kasif [10] showed that determining whether strong 2-consistency can be established is a P-complete problem. From this and the aforementioned connection between strong k-consistency and the $(\exists, k)$-pebble game [12], it follows that determining who wins the $(\exists, 2)$-pebble game is a P-complete problem. Here we give a direct proof to the effect that, for every fixed $k \geq 2$, determining who wins the $(\exists, k)$-pebble game is a P-complete problem. This is done via a reduction from the MONOTONE CIRCUIT VALUE PROBLEM, which we present first as a warm-up to the reduction of the KAI game to the $(\exists, k)$-game, when k is part of the input. Due to space limitations, here we present only outlines of these reductions; complete proofs can be found in the full version of the paper, which is available at http://www.cs.ucsc.edu/˜kolaitis/papers/.

2 The Existential k-Pebble Game

Let $\mathbf{A}$ and $\mathbf{B}$ be two relational structures over the same vocabulary. A *homomorphism h from $\mathbf{A}$ to $\mathbf{B}$* is a mapping $h : A \rightarrow B$ from the universe A of $\mathbf{A}$ to the universe B of $\mathbf{B}$ such that, for every relation $R^{\mathbf{A}}$ of $\mathbf{A}$ and every tuple $(a_1, \ldots, a_m) \in R^{\mathbf{A}}$, we have that $(h(a_1), \ldots, h(a_m)) \in R^{\mathbf{B}}$. A *partial homomorphism from $\mathbf{A}$ to $\mathbf{B}$* is a homomorphism from a substructure of $\mathbf{A}$ to a substructure of $\mathbf{B}$.

Let $k \geq 2$ be a positive integer. The *existential k-pebble game* (or, in short, the *$(\exists, k)$-pebble game*) is played between two players, the Spoiler and the Duplicator, on two relational structures $\mathbf{A}$ and $\mathbf{B}$ according to the following rules: each player has k pebbles labeled $1, \ldots, k$; on the i-th move of a round of the game, $1 \leq i \leq k$, the Spoiler places a pebble on an element a_i of A, and the Duplicator responds by placing the pebble with the same label on an element b_i of B. The Spoiler wins the game at the end of that round, if the correspondence $a_i \mapsto b_i$, $1 \leq i \leq k$, is not a homomorphim between the substructures of $\mathbf{A}$ and $\mathbf{B}$ with universes $\{a_1, \ldots, a_k\}$ and $\{b_1, \ldots, b_k\}$, respectively. Otherwise, the Spoiler removes one or more pebbles, and a new round of the game begins. The Duplicator wins the $(\exists, k)$-pebble game if he has a *winning strategy*, that is to say, a systematic way that allows him to sustain playing "forever", so that the Spoiler can never win a round of the game.

To illustrate this game (and its asymmetric character), let $\mathbf{K}_m$ be the m-clique, that is, the complete undirected graph with m nodes. For every $k \geq 2$, the Duplicator wins the $(\exists, k)$-pebble game on $\mathbf{K}_k$ and $\mathbf{K}_{k+1}$, but the Spoiler wins the $(\exists, k+1)$-pebble game on $\mathbf{K}_{k+1}$ and $\mathbf{K}_k$. As another example, let $\mathbf{L}_s$ be the s-element linear order, $s \geq 2$. If $m < n$, then the Duplicator wins the $(\exists, 2)$-pebble game on $\mathbf{L}_m$ and $\mathbf{L}_n$, but the Spoiler wins the $(\exists, 2)$-pebble game on $\mathbf{L}_n$ and $\mathbf{L}_m$.

Note that the above description of a winning strategy for the Duplicator in the $(\exists, k)$-pebble game is rather informal. The concept of a winning strategy can be made precise, however, in terms of families of partial homomorphisms with appropriate properties. Specifically, a *winning strategy for the Duplicator in the existential k-pebble game on* $\mathbf{A}$ *and* $\mathbf{B}$ is a nonempty family $\mathcal{F}$ of partial homomorphisms from $\mathbf{A}$ to $\mathbf{B}$ such that:

1. For every $f \in \mathcal{F}$, the domain $\mathrm{dom}(f)$ of f has at most k elements.
2. $\mathcal{F}$ is *closed under subfunctions*, which means that if $g \in \mathcal{F}$ and $f \subseteq g$, then $f \in \mathcal{F}$.
3. $\mathcal{F}$ has the *k-forth property*, which means that for every $f \in \mathcal{F}$ with $|\mathrm{dom}(f)| < k$ and every $a \in A$ on which f is undefined, there is a $g \in \mathcal{F}$ that extends f and is defined on a.

Intuitively, the second condition provides the Duplicator with a "good" move when the Spoiler removes a pebble from an element of $\mathbf{A}$, while the third condition provides the Duplicator with a "good" move when the Spoiler places a pebble on an element of $\mathbf{A}$.

3 The $(\exists, k)$-pebble game is P-Complete

In this section, we show that, for every $k \geq 2$, determining the winner of the $(\exists, k)$-pebble game is a P-complete problem. We do this by constructing a reduction from the MONOTONE CIRCUIT VALUE PROBLEM (MCV) in the style of Grohe [7], but with different gadgets. In this reduction, the structures will be undirected graphs with ten unary predicates, called *colors*. So, we actually prove that, for every $k \geq 2$, the $(\exists, k)$-pebble game restricted to such structures is P-complete.

The following concepts and terminology come from Grohe [7].

1. In an undirected graph with colors, a *distinguished pair* of vertices is a pair of vertices that are of the same color, and that color is not used for any other vertex in the graph.
2. A *position* of the $(\exists, k)$-pebble game on A and B, is a set P of ordered pairs such that $P \subseteq A \times B$ and $|P| \leq k$. Often, we will omit the ordered pair notation and use the shorthand $ab \in P$ to mean that $(a, b) \in P$.
3. A *strategy* for the Spoiler is simply a mapping from positions to moves which tells the Spoiler how to play given the current position.

4. We say that the Spoiler *can reach a position P' from another position P of the $(\exists, k)$-pebble game on A and B* if the Spoiler has a strategy for the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$ such that, starting from position P, either he wins the game or after a number of moves the game is in a position P'' such that $P' \subseteq P''$.
   This concept will be used to combine strategies of the Spoiler on different gadgets in order to construct strategies for the combined game.
5. We say that the Duplicator *can avoid a position P' from another position P of the $(\exists, k)$-pebble game on A and B* if the Duplicator has a winning strategy for the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$ such that starting from position P, position P' never occurs.

For each gadget used in the reduction there will be two pieces, one for the Spoiler's structure and one for the Duplicator's structure. For gadget X, we call the Spoiler's side X_S, the Duplicator's side X_D, and the pair (X_S, X_D) simply X.

3.1 The Gadgets H and I

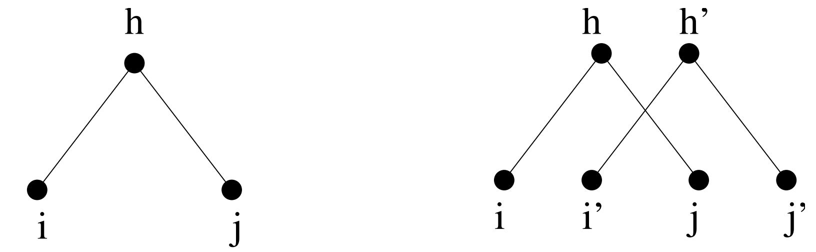

Fig. 1. H Gadget based on the one from [7]. H_S is on the left and H_D is on the right.

The graphs H_D and I_D, which are both based on gadgets from [7], are going to be used for **and** nodes and **or** nodes respectively. H_D, as seen in Figure 1, consists of six vertices h, h', i, i', j, j'. These six vertices form three distinguished pairs, (h, h'), (i, i'), and (j, j'). There are edges from h to i, and h to j, and edges from h' to i' and h' to j'. This graph has only one non-identity automorphism, which we will call swi, that maps any vertex a to a' and any vertex a' to a. H_S is simply the subgraph of H_D determined by h, i, j. Starting from position hh', the Spoiler can reach both ii' and jj' in the $(\exists, k)$-pebble game on (H_S, H_D).

I_D, seen in figure 2, has ten vertices. It contains the three distinguished pairs (h, h'), (i, i'), and (j, j'), plus four additional nodes which we will name by their connections to the other vertices. These nodes are hij, $h'ij'$, $h'i'j$, and $hi'j'$. This graph has three non-identity automorphisms, which we will refer to as fix_i, fix_j, and fix_h. These automorphism fix i, j, and h respectively, while switching the other two. By playing according to these automorphisms, the Duplicator can avoid either ii' or jj' from hh' but not both in the $(\exists, k)$-pebble game.

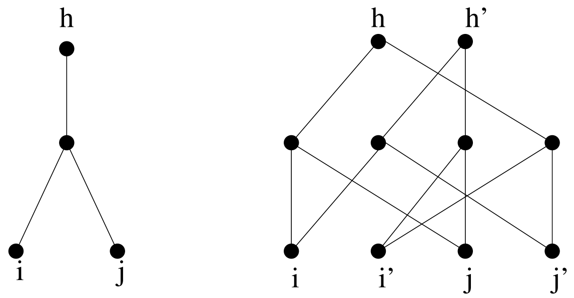

Fig. 2. I Gadget based on the one from [7]. I_S is on the left and I_D is on the right.

3.2 Single Input One-Way Switches

The Single Input One-Way Switches are used to restrict the ways in which the Spoiler can win the game. The basic intuition is that the Spoiler can only make progress in one particular direction; moreover, to do so he must use all of his pebbles.

This lemma is similar to Lemma 14 from [7], adapted to the $(\exists, k)$-pebble game.

Lemma 1. *For every $k \geq 2$ there exists a pair of graphs O^k_S and O^k_D with $O^k_S \subset O^k_D$, $\{x, x', y, y'\} \subset V(O^k_D)$, xx' and yy' distinguished pairs of vertices, and $\{x, y\} \subset V(O^k_S)$, such that:*

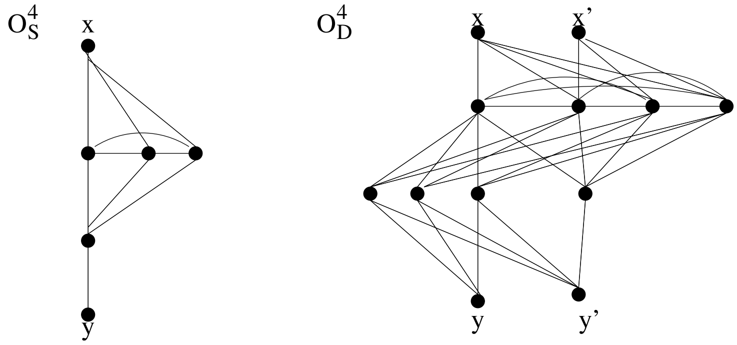

Fig. 3. Single Input One-Way Switch O^4

1. *The Spoiler can reach yy' from xx' in the $(\exists, k)$-pebble game on (O^k_S, O^k_D).*
2. *There exist two disjoint sets of positions of the $(\exists, k)$-pebble game on (O^k_S, O^k_D), called Pretrapped and Trapped positions such that:*
   (a) Pretrapped and Trapped positions are partial homomorphisms
   (b) The Duplicator can avoid positions that are not Trapped and not Pretrapped from Pretrapped positions

(c) The Duplicator can avoid positions that are not Trapped from Trapped positions
(d) The position $\{xx'\}$ is Pretrapped
(e) If P is Pretrapped and $|P| < k$, then $P \cup \{yy\}$ is Pretrapped
(f) The positions $\{yy\}$ and $\{yy'\}$ are Trapped
(g) If P is Trapped and $|P| < k$, then $P \cup \{xx\}$ is Trapped

We call (O^k_S, O^k_D) the Single Input One-Way Switch.

Corollary 2. *The Spoiler can reach $\{(y, y')\}$ from $\{(x, x')\}$ in the $(\exists, k)$-pebble game on the Single Input One-Way Switch, but not in the $(\exists, k-1)$-pebble game on the Single Input One-Way Switch.*

Proof. The first part of the Corollary is simply a restatement of condition 1 from Lemma 1. Assume for the sake of contradiction that the Spoiler can reach $\{(x, x')\}$ from $\{(y, y')\}$ in the $(\exists, k-1)$-pebble game on the Single Input One-Way Switch. Then, from the position $\{(x, x'), (y, y)\}$ of the $(\exists, k)$-pebble game, the Spoiler could reach the position $\{(y, y'), (y, y)\}$ by ignoring the pebbles on (y, y) and playing the $(\exists, k-1)$-pebble game. The position $\{(x, x'), (y, y)\}$ is Pretrapped, but $\{(y, y'), (y, y)\}$ is neither Pretrapped nor Trapped. This is a contradiction because the Duplicator can avoid such positions from Pretrapped positions.

Because of Corollary 2, the Spoiler has to use all of his pebbles in order to make progress. The "One-Way" aspect of the Switch lies in the fact that $\{(y, y')\}$ is Trapped, and $\{(x, x')\}$ is not. This means that the Duplicator can avoid $\{(x, x')\}$ from $\{(y, y')\}$.

3.3 Twisted Switches

The Twisted Switch consists of an H gadget, an I gadget, and two Single Input One-Way Switches in parallel. We use a Twisted Switch in the reduction to initialize the game. The construction of the Twisted Switch is the same as that in Grohe [7], except that we substitute a One-Way Switch in place of what Grohe calls a Threshold Switch.

The following Lemma introduces a set of positions of the Single Input One-Way Switch, called Switched positions. Within the Twisted Switch, the Duplicator uses Switched positions instead of Trapped and Pretrapped positions on the Single Input One-Way Switch.

Lemma 3. *There is a set of positions of the $(\exists, k)$-pebble game on O^k, called Switched positions, such that*

1. *$\{xx', yy\}$ and $\{xx, yy'\}$ are Switched.*
2. *If P is Switched, then either $P \cup \{xx', yy\}$ or $P \cup \{xx, yy'\}$ is Switched.*
3. *The Duplicator can avoid positions that are not Switched from Switched positions in the $(\exists, k)$-pebble game.*

Lemma 4. *On the Twisted Switch, the Spoiler can reach $\{yy'\}$ from $\{xx'\}$ in the $(\exists, k)$-pebble game, and there exists a set of positions of the $(\exists, k)$-pebble game called Twisted positions such that*

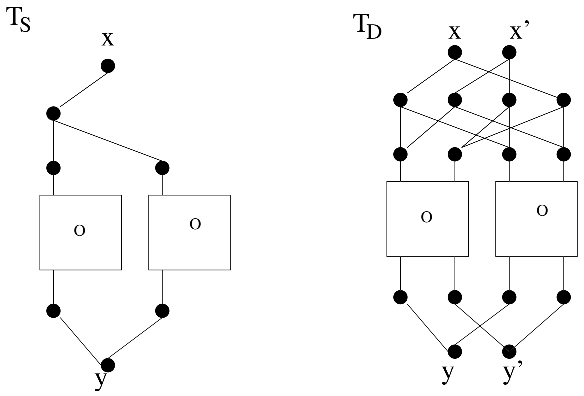

Fig. 4. Twisted Switch [7]

1. *The Duplicator can avoid non-Twisted positions from Twisted positions*
2. $\{yy\}$*, and* $\{yy'\}$*, are Twisted positions.*
3. *If* P *is a Twisted position, then* $P \cup \{xx'\}$ *is a Twisted position.*

The Twisted Switch will be used to initialize the game. In order to do this, $x \in T_S$ and $x' \in T_D$ are colored the same color, while $x \in T_D$ is colored a different color. Thus, if the Spoiler plays on x, then the Duplicator must play on x'. From here the Spoiler can play to $\{(y, y')\}$. The Twisted positions allow the Duplicator to avoid $\{(x, x)\}$, which is a losing position.

3.4 Reduction from MCV to $(\exists, k)$-pebble game

Theorem 5. *For every fixed* $k \geq 2$*, determining the winner of the* $(\exists, k)$*-pebble game is P-Complete.*

Proof. (Outline) Given a monotone circuit C, an assignment of values to the input nodes, and a node v, we construct a graph C_D. For each node a in the circuit, there are three choices.

1. If a is an input node, then C_D contains vertices a and a'. If the value of a in C is false, then we color a' black.
2. If a is an **and** node with parents b and c, then C_D contains nodes a and a', a copy of H_D, called H_a with h and h' identified with a and a', as well as two copies of O^k_D, one of which has x and x' connected to i and i', and y and y' connected to b and b' called O_{ab}. The other copy connects j and j' to c and c' in a similar manner and is called O_{ac}.
3. If a is an **or** node with parents b and c, then C_D contains nodes a and a', a copy of I_D called I_a with h and h' identified with a and a', as well as two copies of O^k_D, one of which has x and x' connected to i and i', and y and y' connected to b and b' called O_{ab}. The other copy connects j and j' to c and c' in a similar manner and is called O_{ac}.

In any case, a is colored white.

The construction of C_S is similar, except that we do not add a' and we use the Spoiler version of each gadget and switch instead of the Duplicator version.

Also, there is a Twisted Switch T, such that x is colored a fresh color in C_S, and x' is colored the same color in C_D. Also, in C_S, y in T is connected to v, while in C_D, y is connected to v and y' is connected to v'.

The Spoiler plays from the top of the Twisted Switch through the graph attempting to reach a false input. Because of the properties of the H and I gadgets and the Single Input One-Way Switch , the Spoiler can do this if the value of the goal node is false. If the value of the goal node is true, then the Duplicator can play along indefinitely. He does this by choosing a path down the simulated circuit which leads to a true input. If the Spoiler attempts to depart from the intended mode of play, the Duplicator can use the properties of Trapped and Pretrapped strategies to arrive at a partial homomorphism that is a subset of the identity. ∎

4 The $(\exists, k)$-pebble game is EXPTIME-Complete

Kasai, Adachi and Iwata [8] showed that the following pebble game, which (to avoid confusion in the terminology) we will call the KAI game, is EXPTIME-complete. The game is played between two players, called Player I and Player II. An instance of the KAI game is a quadruple (X, S, R, t), where X is a set of nodes, $R \subseteq X^3$ is a set of *rules*, $S \subseteq X$ is the initial position of the game, and $t \in X$ is the *goal*. There are as many pebbles as nodes in the initial position S; at the start of the game, each node in S has a pebble on it. The two players take turns and in each turn they slide a pebble as follows: if $(x, y, z) \in R$ is a rule, then the current player may slide a pebble from x to z, if there are pebbles on x and y, but not z. The first player to place a pebble on the goal t wins. The problem is, given (X, R, S, t), does Player I have a winning strategy?

We will show that the $(\exists, k)$-pebble game is EXPTIME complete by exhibiting a reduction from the KAI game. The reduction procedes by constructing an instance of the $(\exists, k)$-pebble game such that the Spoiler and the Duplicator simulate the instance of the KAI game. In particular, the Spoiler can only win by simulating a winning strategy for Player I in the KAI game. If there is no winning strategy, then the Spoiler does not gain any advantage by not simulating the KAI game.

In order to perform this simulation, we use a Twisted Switch (Section 3.3) to initialize the game, and new switches to allow Player I and Player II to choose rules, to apply the rules, and to force the Spoiler to simulate the game.

4.1 The Gadgets H^m and I^m

These gadgets allow the Spoiler and Duplicator to choose rules of the KAI game to use. In both the H^m and I^m gadgets, the nodes y^i, y^i_0, y^i_1 are colored the same color, but the color of y^i is different that the color of y^j for $i \neq j$.

Lemma 6. *For every $k \geq 2$, in the $(\exists, k)$-pebble game on (H^m_S, H^m_D), from a position $\{xx_j\}$, $j \in \{0, 1\}$, the Spoiler can reach $\{y^i y^i_j\}$ for any i.*

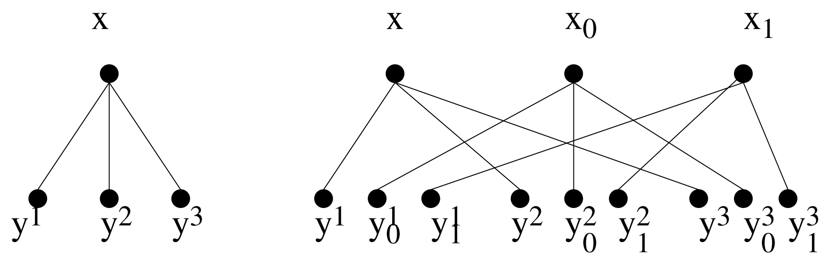

Fig. 5. H^3 Gadget

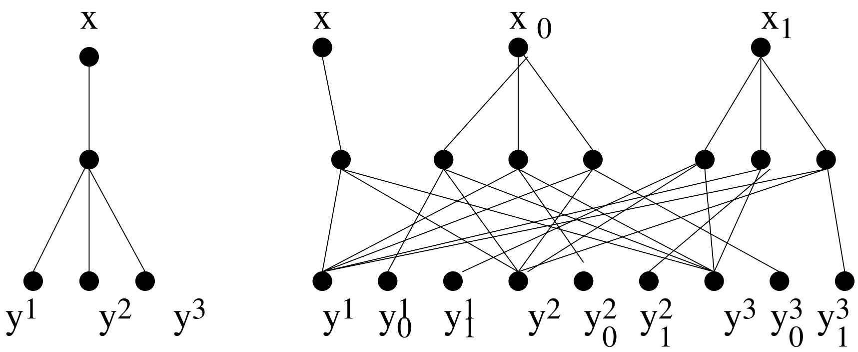

Fig. 6. I^3 Gadget

Lemma 7. *For every $k \geq 2$, in the $(\exists, k)$-pebble game on I_S^m, I_D^m, from a position $\{xx_j\}$, $j \in \{0, 1\}$, the Duplicator can choose any $1 \leq i \leq m$, and avoid $\{y^l y^l_j\}$ for $l \neq i$.*

4.2 Multiple Input One-Way Switches for the $(\exists, k)$-pebble game

The idea of the Multiple Input One-Way Switch is to restrict the Spoiler's potential winning strategies. We simulate each node x^i in the KAI game by using three nodes in the Duplicator's graph, x_0^i, x_1^i, x^i. These correspond to not having pebble on x^i in the simulated game, having a pebble on x^i in the simulated game, and no information about x^i, respectively. In the Multiple Input One-Way Switch, the Spoiler can only make progress if he has information about each node in the simulated game. Also, if the Spoiler attempts to play backwards through the Switch, he will end up with no information about any nodes in the simulated game.

Lemma 8. *For every $k \geq 2$, there exists a pair of graphs M_S^k, and M_D^k such that*

$$\{x^1, x_0^1, x_1^1, \ldots, x^{k-1}, x_0^{k-1}, x_1^{k-1}, y^1, y_0^1, y_1^1, \ldots, y^{k-1}, y_0^{k-1}, y_1^{k-1}\} \subset V(M_D^k),$$

$\{x^1, \ldots, x^{k-1}, y^1, \ldots, y^{k-1}\} \subset V(M_S^k)$ *and the following properties hold:*

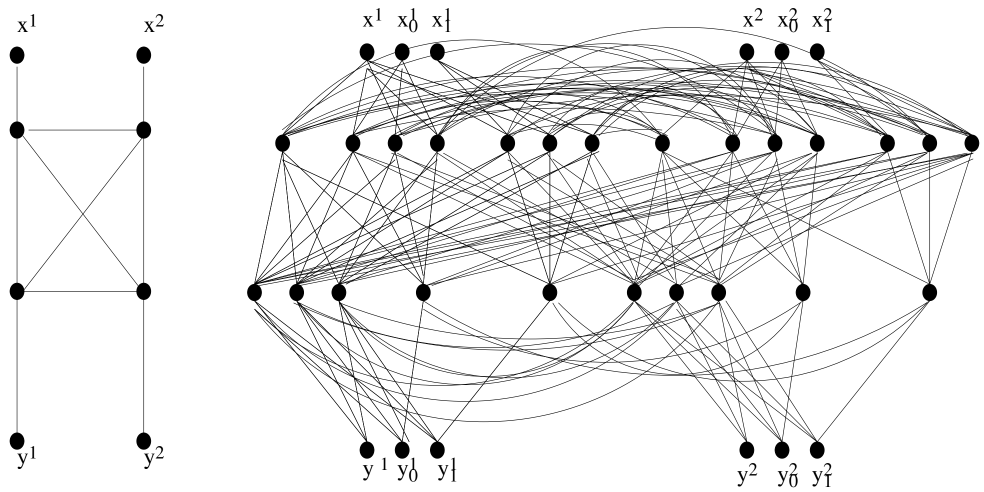

Fig. 7. A subgraph of M^4

1. *From a position $\{x^i x^i_{j_i} | 1 \le i \le k-1, j_i \in \{0,1\}\}$, the Spoiler can reach the position $\{y^i y^i_{j_i} | 1 \le i \le k-1, j_i \in \{0,1\}\}$ in the $(\exists, k)$-pebble game on M^k_S and M^k_D.*
2. *There exist two disjoint sets of positions of the $(\exists, k)$-pebble game on (M^k_S, M^k_D), called Pretrapped and Trapped positions such that:*
   - *(a) Pretrapped and Trapped positions are partial homomorphisms*
   - *(b) The Duplicator can avoid positions that are not Pretrapped and not Trapped from Pretrapped positions*
   - *(c) The Duplicator can avoid positions that are not Trapped from Trapped positions*
   - *(d) From any position $P = \{x^i a | 1 \le i \le k-1\}$ where $|\{x^i x^i_j \in P | j \in \{0,1\}\}| < k-1$, the Duplicator can avoid $y^i y^i_j$ for all $1 \le i \le k-1, j \in \{0,1\}$.*
   - *(e) All positions that are subsets of positions of the form $\{x^i x^i_{j_i} | 1 \le i \le k-1, j_i \in \{0,1\}\}$, are PreTrapped.*
   - *(f) If P is Pretrapped and $|P| < k$, then $P \cup \{y^i y^i\}$ is Pretrapped for all i*
   - *(g) Any position in which all of the Spoiler's pebbles are on nodes y^i, is Trapped.*
   - *(h) If P is Trapped and $|P| < k$, then $P \cup \{x^i x^i\}$ is Trapped for all i*

Moreover, $|V(M^k_S)|$ is $O(k^2)$ and $|V(M^k_D)|$ is $O(k^2)$.

4.3 The Rule Gadget

The Rule gadgets are used to simulate a move of the KAI game. One rule gadget causes the Spoiler to lose if the rule gadget corresponds to a rule that cannot be applied, and another causes the Duplicator to lose if the rule cannot be applied.

Lemma 9. *If the rule gadget RS^n does not correspond to a legal rule, that is, if one of xx_1, yy_1, zz_0 is not in P, then the Duplicator can avoid $z'z'_1$ in the $(\exists, k)$-pebble game on RS^n.*

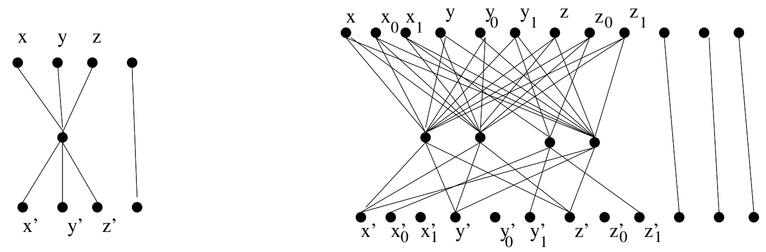

Fig. 8. The Rule Gadget RS^n that penalizes the Spoiler for choosing a bad rule

Lemma 10. *If the rule gadget RD^n does not correspond to a legal rule, that is, if one of xx_1, yy_1, zz_0 is not in P, then the Spoiler can play to $z'z'_0$, which causes the Duplicator to lose.*

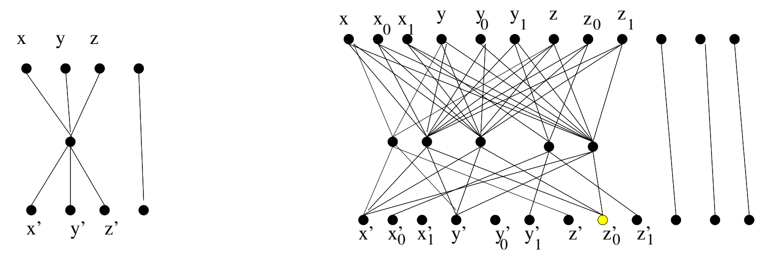

Fig. 9. The Rule Gadget RD^n that penalizes the Duplicator for choosing a bad rule

4.4 Winning the Game

For each gadget, we define its *boundary* to be the set of nodes which are used to connect that gadget to other gadgets. For M^k, H^m, and I^m, the x and y nodes form the boundary. For RS^n and RD^n, the boundary consists of all the nodes except for the middle nodes. For the Twisted Switch, the boundary consists of y in T_S and y, y' in T_D. In the Twisted Switch, x and x' are never connected to any other gadgets.

Lemma 11. *For each gadget other than the Twisted Switch, starting from a position that is a subset of the identity on the boundary, the Duplicator can avoid any position that is not a subset of the identity on the boundary.*

By combining this lemma with the properties of the Multiple Input One-Way Switch, we obtain a sufficient condition for the Duplicator to win the $(\exists, k)$-pebble game.

4.5 Reduction from KAI Game to $(\exists, k)$-pebble game

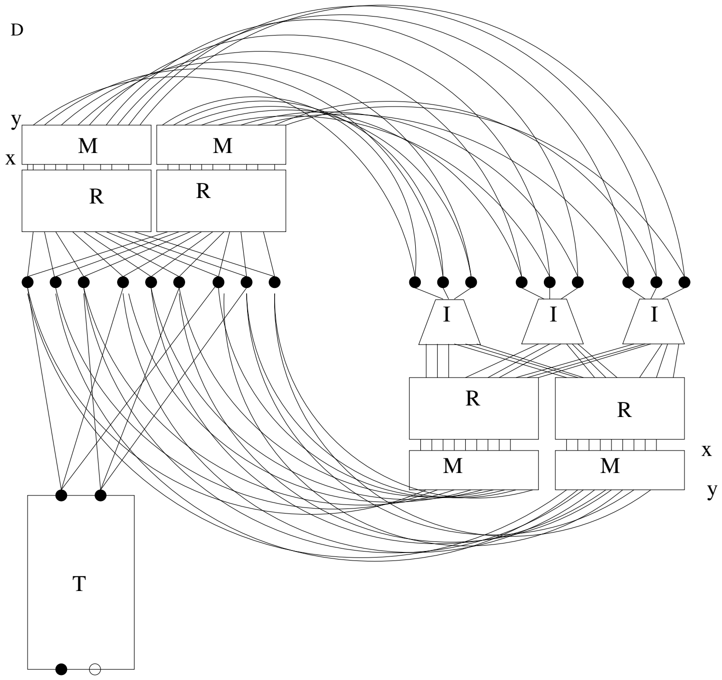

Fig. 10. This is component decomposition of the Duplicator's graph for the reduction

Theorem 12. *Determining the winner of the $(\exists, k)$-pebble game with k part of the input is EXPTIME-Complete.*

Proof. (Outline) We will give a polynomial-time reduction from the KAI Game to the $(\exists, k)$-pebble game. Given an instance (X, S, R, t) of the KAI game, we form an instance of the $(\exists, k)$-pebble game as follows.

The Duplicator's graph and the Spoiler's graph each have two sides. One side represents Player I's turn in the KAI game, while the other side represents Player II's turn.

First, we build Player I's side of the graph. For each $x^i \in X$, we form three nodes in D, called xs^i, xs^i_0, xs^i_1. These three nodes correspond to specific information about the simulated KAI game. If there is a pebble on xs^i_1, then there is a pebble on x in the KAI game, and xs^i_0 corresponds to no pebble on x. A pebble on xs^i in the Duplicator's graph means that the Spoiler has made a mistake. For each $(x, y, z) \in R$, construct a rule gadget to penalize the Spoiler, connected to each xs^i, xs_0^i, xs_1^i by H^m gadgets. The other end is then connected to a Multiple Input One-Way Switch .

On the other half of the graph, there are another set of nodes xd^i, xd_0^i, xd_1^i. Connecting the xd nodes to Duplicator punishing rule gadgets are a set of I^m gadgets where $m = |R|$. Given any two of these I^m gadgets A and B, there is an edge between $A.y_j^i$ and $B.y_k^i$ for all i, j, k, and an edge between $A.y^i$ and $B.y^i$ for all i. We then connect a Multiple Input One-Way Switch to each of the rule gadgets in the obvious way. The outputs of these are connected back around to the first set of state nodes.

We use a Twisted Switch to set up the initial positions. If $x^i \in S$, then there is an edge from y' of the Twisted Switch to xs_1^i otherwise there is an edge from y' to xs_0^i. There is an edge from y to xs^i for every i. We then color x' a unique color. In addition we give t_1 a unique color so that if the Spoiler can reach tt_1, then the Duplicator loses.

The Spoiler's graph is constructed in a similar way. The Spoiler gets xs^i and xd^i for every $x^i \in X$. Also, for each rule and each side, there is a corresponding Multiple Input One-Way Switch followed by a rule gadget. For the Twisted Switch, we color x the same color that we colored x' in the Duplicator's graph.

The two players now simulate the KAI game by playing the $(\exists, k)$-pebble game on these structures. The Spoiler initializes the game by playing on the Twisted Switch. From here, the Spoiler uses the H^m gadgets to choose a rule, then plays through the RS^n gadget and the Multiple Input One-Way Switch to simulate the move of Player I in the KAI game. Then, the Spoiler continues to play through Player II's side of the structures, allowing the Duplicator to choose a rule to simulate, and applying that rule. If Player I has a winning strategy for the KAI game, then this simulation process will eventually simulate Player I placing a pebble on t. Because of the coloring of the structures, this causes the Spoiler to win the $(\exists, k)$-pebble game. The more difficult step is showing that if Player I does not have a winning strategy, then the Duplicator wins the $(\exists, k)$-pebble game. If the Spoiler plays nice and simulates the KAI game, then the Duplicator can avoid t, by playing a smart strategy for the KAI game. If the Spoiler does not play nice, and departs from the simulation, then, because of the properties of the gadgets, the Duplicator can play along indefinitely. ∎

As pointed out in Section 3, the structures used in the reduction of MCV to the $(\exists, k)$-pebble game with fixed k were undirected graphs with a fixed number (ten) of colors. In contrast, the structures used in the preceding reduction of the KAI game to the $(\exists, k)$-game with k part of the input are undirected graphs with a number of colors that is linear in the size of the input. It is an interesting technical problem to exhibit a reduction of the KAI game to the $(\exists, k)$-game with k part of the input in which the structures are undirected graphs with a fixed number of colors.

5 Concluding Remarks

Although in this paper we focused on the $(\exists, k)$-pebble game because of its connections to constraint satisfaction, in database theory there is also interest in the *one-to-one* $(\exists, k)$*-pebble game*, which is the variant of the $(\exists, k)$-pebble game in which the Duplicator strives to maintain one-to-one homomorphisms (see [11]). A perusal of the two reductions presented here reveals that in both these reductions the structures constructed have the property that the Duplicator wins the $(\exists, k)$-pebble game if and only if the Duplicator wins the one-to-one $(\exists, k)$-pebble game. Consequently, determining the winner in the one-to-one $(\exists, k)$-pebble is P-complete for every fixed $k \geq 2$, and is EXPTIME-complete when k is part of the input.

Several problems remain open in this area. Kasai and Iwata [9] proved that the number of pebbles used in the KAI game gives rise to a strict hierarchy on the time complexity of that game. Thus, it is natural to ask whether a similar strict hierarchy result can be proved for the $(\exists, k)$-pebble, for fixed k. This amounts to showing that, for each fixed $k \geq 2$, determining the winner of the $(\exists, k)$-pebble game is not solvable in time $O(n^s)$ for any $s < 2k$. Finally, it remains an open problem to establish that the (two-sided) k-pebble game with k part of the input is an EXPTIME-complete problem.

References

1. A. K. Chandra, D. C. Kozen, and L. J. Stockmeyer. Alternation. *Journal of the ACM*, 28:114–233, January 1981.
2. M. C. Cooper. An optimal k-consistency algorithm. *Artificial Intelligence*, 41:89–95, 1989.
3. V. Dalmau, Ph. G. Kolaitis, and M. Y. Vardi. Constraint satisfaction, bounded treewidth, and finite-variable logics. In *Proc. of Eighth International Conference on Principles and Practice of Constraint Programming*, pages 310–326, 2002.
4. R. Dechter. Constraint networks. In S.C. Shapiro, editor, *Encyclopedia of Artificial Intelligence*, pages 276–285. Wiley, New York, 1992.
5. R. Dechter. From local to global consistency. *Artificial Intelligence*, 55:87–107, 1992.
6. T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: a study through Datalog and group theory. *SIAM Journal on Computing*, 28:57–104, 1998.
7. M. Grohe. Equivalence in finite-variable logics is complete for polynomial time. *Combinatorica*, 19(4):507–523, 1999.
8. T. Kasai, A. Adachi, and S. Iwata. Classes of pebble games and complete problems. *SIAM Journal of Computing*, 8(4):574–586, 1979.
9. T. Kasai and S. Iwata. Gradually intractable problems and nondeterminitstic log-space lower bounds. *Mathematical Systems Theory*, 18:153–170, 1985.
10. S. Kasif. On the parallel complexity of some constraint satisfaction problems. In *Proc. of Fifth National Conference on Artificial Intelligencee*, volume 1, pages 349–353, 1986.
11. Ph. G. Kolaitis and M. Y. Vardi. On the expressive power of Datalog: Tools and a case study. *Journal of Computer and System Sciences*, 51:110–134, 1995.
12. Ph. G. Kolaitis and M. Y. Vardi. A game-theoretic approach to constraint satisfaction. In *Proc. of the Seventeenth National Conference on Artificial Intelligence*, pages 175–181, 2000.
13. E. Pezzoli. Computational complexity of Ehrenfeucht-Fraïssé games on finite structures. In *Computer Science Logic. 12th International Workshop, CSL'98.*, pages 159–170, 1999.

A Proofs

In the proofs we use $Im(P)$, P a position of the game, to refer to the nodes in the Duplicator's graph which are contained in P. This comes from the fact that many of the positions we will be discussing are partial homomorphisms.

Lemma 1. *For every $k \geq 2$ there exists a pair of graphs O^k_S, and O^k_D with $O^k_S \subset O^k_D$, $\{x, x', y, y'\} \subset V(O^k_D)$, xx' and yy' distinguished pairs of vertices, and $\{x, y\} \subset V(O^k_S)$, such that:*

1. *The Spoiler can reach yy' from xx' in the $(\exists, k)$-pebble game on (O^k_S, O^k_D).*
2. *There exist two disjoint sets of positions of the $(\exists, k)$-pebble game on (O^k_S, O^k_D), called Pretrapped and Trapped positions such that:*
   (a) *Pretrapped and Trapped positions are partial homomorphisms*
   (b) *The Duplicator can avoid positions that are not Trapped and not Pretrapped from Pretrapped positions*
   (c) *The Duplicator can avoid positions that are not Trapped from Trapped positions*
   (d) *The position $\{xx'\}$ is Pretrapped*
   (e) *If P is Pretrapped and $|P| < k$, then $P \cup \{yy\}$ is Pretrapped*
   (f) *The positions $\{yy\}$ and $\{yy'\}$ are Trapped*
   (g) *If P is Trapped and $|P| < k$, then $P \cup \{xx\}$ is Trapped*

Proof. First, we define O^k_S and O^k_D. Then, we show that these graphs have the desired properties.

$$\begin{aligned}
V(O^k_D) = &\{x, x', y, y'\} \cup \\
&\{1, 2\} \times \{0, 1, 2, \ldots, k-1\} \\
E(O^k_D) = &\{(x, (1, a))\} \cup \\
&\{(x', (1, a))|a > 0\} \cup \\
&\{((1, a), (1, b))|a \neq b\} \\
&\{((1, a), (2, b))|b \neq a\} \cup \\
&\{((1, 0), (2, 0))\} \cup \\
&\{((2, a), y)|a > 0\} \cup \\
&\{((2, a), y'\}
\end{aligned}$$

$$\begin{aligned}
V(O^k_S) = &\{x, y\} \cup \\
&\{1\} \times \{1, 2, \ldots k-1\} \cup \\
&\{(2, 0)\} \\
E(O^k_S) = &\{(x, (1, a))\} \cup \\
&\{((1, a), (1, b))|a \neq b\} \\
&\{((1, a), (2, 0))\} \cup \\
&\{((2, 0), y)\}
\end{aligned}$$

We color each row of vertices a unique color so that when the Spoiler places a pebble on row 2, the Duplicator cannot place a pebble on row 1. These graphs can be seen in figure 3.

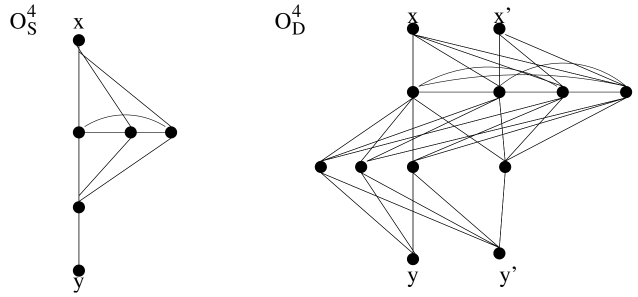

Fig. 11. Single Input One-way Switch

The graph O^k_D has the property that given any set of $m < k-1$ vertices of the form $\{(2,a)|a > 0\}$, there are exactly $k-1-m$ vertices $\{(1,b)|b > 0\}$ such that these vertices form a complete bipartite graph with sets of size m and $k-1-m$. We will use this fact later in describing the strategy of the Duplicator.

In order to prove the first item, we must exhibit a sequence of moves for the Spoiler in the $(\exists, k)$-pebble game on (O^k_S, O^k_D) starting in position $\{xx'\}$ such that either the Duplicator loses or the position $\{yy'\}$ occurs. The Spoiler first places $k-1$ pebbles on $\{(1,a)|a > 0\}$. Since these are all adjacent to the pebble on x, the Duplicator must place $k-1$ pebbles adjacent to x'. Since these are all adjacent to each other, the Duplicator cannot place any of these pebbles on the same vertex. The Duplicator is forced to place these pebbles on $\{(1,a)|a > 0\}$. The Spoiler now picks up the pebbles on xx' and places a pebble on $(2,0)$. This pebble is adjacent to every pebble the Spoiler placed in the previous row, so the Duplicator must place his pebble on $(2,0)$. Now the Spoiler picks up any pair of pebbles from row 1 and places a pebble on y. The Duplicator must place a pebble on either y or y', but it must be adjacent the pebble placed in row 2. Since $(2,0)$ is not connected to y, the Duplicator must play on y', arriving at the position $\{yy'\}$.

Now we must show the existence of the two disjoint sets of positions. We first define a set of partial homomorphisms, and then split it to form the Pretrapped and Trapped positions. Given a partial homomorphism $P|O^k_S \to O^k_D$, we define $R_{i,+}(P) = \{(i,a)|\exists u, (u,(i,a)) \in P \wedge a > 0\}$, and $R_{i,0}(P) = \{(i,a)|\exists u, (u,(i,a)) \in P \wedge a = 0\}$. Also, let $T(P) = R_{1,0}(P) \cup R_{2,0}(P)$. Let S be the set of all partial homomorphisms $P|O^k_S \to O^k_D$ such that:

1. $|P| \leq k$
2. $|T(P)| > 0 \Rightarrow xx' \notin P$
3. $\neg(xx' \in P \wedge yy' \in P)$

We say that a position P is Pretrapped if $P \in S$ and $xx' \in P$, and P is Trapped if $P \in S$ and $xx' \notin P$.

Now it remains to prove that the sets of Pretrapped and Trapped positions have the desired properties. By definition, Pretrapped and Trapped positions are partial homomorphisms. Clearly any subset of a Trapped position is Trapped and any subset of a Pretrapped position is either Trapped or Pretrapped. The position $\{xx'\} \in S$, so $\{xx'\}$ is Pretrapped. The positions $\{yy\}$ and $\{yy'\}$ are in S, so they are Trapped.

Assume that P is Pretrapped, then $|T(P)| = 0$, and every pebble that the Duplicator has placed on row 2 is on the positive side and is therefore connected to y. This means that $P \cup \{yy\}$ is a partial homomorphism, and, since we have not changed any of the criteria for being in S or removed xx', $P \cup \{yy\}$ is Pretrapped.

Assume that P is Trapped, then $xx' \notin P$. There is no vertex adjacent to x' that is not also adjacent to x, so when the Duplicator must play on one of them, it is always acceptable to play on x. This means that $P \cup \{xx\}$ is Trapped.

Now we have to show that for every position $P \in S$ such that $|P| < k$, and for every $u \in V(O_S^k)$, there is a $v \in V(O_D^k)$ such that $P \cup \{uv\} \in S$ and if P was Trapped, then so is $P \cup \{uv\}$.

Case 1: If $u = x$, then $v = x$. Clearly P was Trapped, and therefore so is $P \cup \{xx\}$.

Case 2: If $u = (1, a)$, then if $|R_{1,+}(P)| + |R_{2,+}(P)| < k-1$, then there exists $(1, b)$, $b > 0$, such that $(1, b)$ is adjacent to every vertex in $R_{2,+}(P)$, so choose $v = (1, b)$. This is still a partial homomorphism. It is also still in S because we have not affected the criteria for membership. If $|R_{1,+}(P)| + |R_{2,+}(P)| = k - 1$, then $v = (1, 0)$. Since $|R_{1,+}(P)| + |R_{2,+}(P)| = k - 1$ we know that $|T(P)| = 0$, and $xx' \notin P$. Thus $P \cup \{uv\} \in S$.

Case 3: If $u = (2, 0)$, then if $|R_{1,+}(P)| < k - 1$, then there exists $(2, b)$, $b > 0$, such that $(2, b)$ is adjacent to every vertex in $R_{1,+}(P)$, so choose $v = (2, b)$. This is still a partial homomorphism. It is also still in S because we have not affected the criteria for membership. If $|R_{1,+}(P)| = k - 1$, then $v = (2, 0)$. Since $|R_{1,+}(P)| = k - 1$ we know that $|T(P)| = 0$, and $xx' \notin P$. Thus $P \cup \{uv\} \in S$.

Case 4: If $u = y$, then, if P is Pretrapped, $P \cup \{yy\}$ is Pretrapped, so $v = y$. If P is Trapped, then check $R_{2,0}(P)$. If $|R_{2,0}(P)| = 0$, then $v = y$, else $v = y'$.

Corollary 2. *The Spoiler can reach $\{(y, y')\}$ from $\{(x, x')\}$ in the $(\exists, k)$-pebble game on the Single Input One-Way Switch, but not in the $(\exists, k-1)$-pebble game on the Single Input One-Way Switch.*

Proof. The first part of the Corollary is simply a restatement of condition 1 from Lemma 1. Assume for the sake of contradiction that the Spoiler can reach $\{(x, x')\}$ from $\{(y, y')\}$ in the $(\exists, k - 1)$-pebble game on the Single Input One-Way Switch. Then, from the position $\{(x, x'), (y, y)\}$ of the $(\exists, k)$-pebble game, the Spoiler could reach the position $\{(y, y'), (y, y)\}$, by ignoring the pebbles on (y, y) and playing the $(\exists, k-1)$-pebble game. The position $\{(x, x'), (y, y)\}$ is Pretrapped, but $\{(y, y'), (y, y)\}$ is neither Pretrapped nor Trapped. This is a contradiction because the Duplicator can avoid such positions from Pretrapped positions.

Lemma 3. *There is a set of positions on O^k, called switched positions, such that*

1. $\{xx', yy\}$ and $\{xx, yy'\}$ are Switched.
2. If P is Switched, then either $P \cup \{xx', yy\}$ or $P \cup \{xx, yy'\}$ is Switched.
3. The Duplicator can avoid positions that are not Switched from Switched positions in the $(\exists, k)$-pebble game.

Proof. Let the set of Switched positions be the set of all $P \in S$, such that if $yy \in P$, then $P \cup \{xx'\} \in S$.

Clearly $\{xx', yy\}$ and $\{xx, yy'\}$ are Switched. If P is Switched, then either P is Trapped or P is Pretrapped. If P is Pretrapped, then $P \cup \{xx'\}$ is also Pretrapped because $xx' \in P$, and $P \cup \{yy\}$ is Pretrapped as proven earlier. Therefore $P \cup \{xx', yy\}$ is Switched. If P is Trapped, then either yy is in P or not. If $yy \in P$, then $P \cup xx' \in S$, and therefore $P \cup \{xx', yy\}$ is Switched. If $yy \notin P$, then $P \cup \{xx, yy'\} \in S$ and $P \cup \{xx, yy'\}$ is Switched.

Based on the earlier proof, if $P \in S$, then the Duplicator can play to some $P' \in S$. We only have to show that the Duplicator can maintain the property that if $yy \in P$, then $P \cup \{xx'\} \in S$. If $P \cup \{xx', yy\}$ is Switched, then the Duplicator plays according to a Trapped strategy on $P \cup \{xx'\}$. If $P \cup \{xx, yy'\}$ is Switched, then the Duplicator plays according to a Trapped strategy on $P \cup \{yy'\}$.

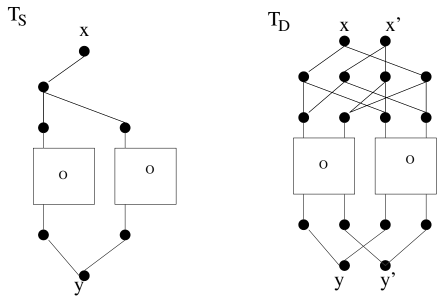

Fig. 12. Twisted Switch [7]

Lemma 4. *On the Twisted Switch, The Spoiler can reach $\{yy'\}$ from $\{xx'\}$ in the $(\exists, k)$-pebble game, and there exists a set of positions of the $(\exists, k)$-pebble game called Twisted positions such that*

1. *The Duplicator can avoid non Twisted positions from Twisted positions*
2. *$\{yy\}$, and $\{yy'\}$, are Twisted positions.*
3. *If P is a Twisted position, then $P \cup \{xx'\}$ is a Twisted position.*

Proof. It is easy to see that the Spoiler can reach yy' from xx'. The Spoiler plays to either ii' or jj' in H, and the traverses one of the One-way switches, and reaches yy'.

Let the Twisted positions be all those positions that meet one of the following criteria:

1. $P_H \subset swi$, $P_I \subset fix_i$, P_{O_l} is Switched, $P_{O_r} \cup \{xx, yy\}$ is Trapped.
2. $P_H \subset swi$, $P_I \subset fix_j$, P_{O_r} is Switched, $P_{O_l} \cup \{xx, yy\}$ is Trapped.
3. $P_H \subset fix$, $P_I \subset fix_i$, P_{O_r} is Switched, $P_{O_l} \cup \{xx, yy\}$ is Trapped.
4. $P_H \subset fix$, $P_I \subset fix_j$, P_{O_l} is Switched, $P_{O_r} \cup \{xx, yy\}$ is Trapped.

Clearly, $\{yy\}$ and $\{yy'\}$ are Twisted. Assume P is twisted. We know that $P_I \subset fix_i$ or $P_I \subset fix_j$. Since xx' is in both of these automorphisms, $P \cup \{xx'\}$ is Twisted.

Given any Twisted position P, it falls in one of the four categories above. On each part of the graph, the Duplicator simply plays according to the appropriate strategy.

Theorem 5. *The $(\exists, k)$-pebble game with k fixed is P-Complete*

Proof. This reduction is modified only slightly from the reduction in [7]. The reduction proceeds as follows. Given a circuit C, an assignment of values to the input nodes, and a node v, we construct a graph C_D. For each node a in the circuit, there are three choices.

1. If a is an input node, then C_D contains vertices a and a'. If the value of a in C is false, then we color a' black.
2. If a is an **and** node with parents b and c, then C_D contains nodes a and a', a copy of H_D, called H_a with h and h' identified with a and a', as well as two copies of O_D^k, one of which has x and x' connected to i and i', and y and y' connected to b and b' called O_{ab}. The other copy connects j and j' to c and c' in a similar manner and is called O_{ac}.
3. If a is an **or** node with parents b and c, then C_D contains nodes a and a', a copy of I_D called I_a with h and h' identified with a and a', as well as two copies of O_D^k, one of which has x and x' connected to i and i', and y and y' connected to b and b' called O_{ab}. The other copy connects j and j' to c and c' in a similar manner and is called O_{ac}.

In any case, a is colored white.

The construction of C_S is similar, except that we do not add a' and we use the Spoiler version of each gadget and switch instead of the Duplicator version.

Also, there is a Twisted Switch T, such that x is colored a fresh color in C_S, and x' is colored the same color in C_D. Also, in C_S, y in T is connected to v, while in C_D, y is connected to v and y' is connected to v'.

Now we want to prove that if the value of node v in C is false then the Spoiler wins the $(\exists, k)$-pebble game on (C_S, C_D) and if the value of node v is true then the Duplicator wins the $(\exists, k)$-pebble game on (C_S, C_D). First, assume that the value of v is false. We must show that the Spoiler wins the $\exists k$-pebble game. First, the Spoiler can play starting on xx', and play to vv'. The proof is by induction on the height of v. Assume that the height of v is 0. This means that v is an input node. Since we are assuming that the value of v is false, by construction, v and v' are different colors, so the Spoiler wins. Now, assume that the height of v is n and for every node u of height less than n, if the value of u is false, then the Spoiler wins. If v is an and node with inputs b and c, then one of the inputs of v is false. Assume, without loss of generality, that b is false. In the $(\exists, k)$-pebble game on (H_S, H_D), the Spoiler can reach ii' from

hh'. From this position the Spoiler can reach bb' along O^k_S. By induction, the Spoiler wins the game starting from bb'. The argument is analogous for the or nodes.

Now that we have shown that if the value of v is false, then the Spoiler wins the $(\exists, k)$-pebble game, we must show that if the value of v is true, then the Duplicator wins the $(\exists, k)$-pebble game.

A position P is safe if the position on T is twisted and there exists a node a such that:

1. For every one-way switch O adjacent to aa', $P_O \cup \{yy\}$ is Trapped, and everywhere else P acts as the identity; or
2. For every one-way switch O adjacent to aa', $P_O \cup \{yy'\}$ is Trapped, and
   (a) a is an and node with parents b and c, and $P_{H_a} \subseteq swi$, $P_{O_{ab}} \cup \{xx'\}$ is Pretrapped, and $P_{O_{ac}} \cup \{xx'\}$ is Pretrapped.
   (b) a is an or node with parents b and c with one of the following:
      i. b is true and $P_{I_a} \subseteq fix_j$, $P_{O_{ab}} \cup \{xx'\}$ is Pretrapped, and $P_{O_{ac}} \subseteq id$.
      ii. c is true and $P_{I_a} \subseteq fix_i$, $P_{O_{ac}} \cup \{xx'\}$ is Pretrapped, and $P_{O_{ab}} \subseteq id$.
   (c) a is an input node with $aa \notin P$.

We now have to show that the set of safe positions is a winning strategy for the Duplicator. Any safe position is a partial isomorphism. The position vv' is clearly a safe position. Since all of the criteria for being safe involve subset properties, any subset of a safe position is safe. Now, given any safe position P, we must show that the Duplicator can play into another safe position. Assume P satisfies case 1. The Duplicator plays according to the identity everywhere except the one-way switches adjacent to a. On one of these subgraphs O, the Duplicator plays according to a Trapped strategy on $O \cup \{yy\}$ if $|P_O| < k - 1$, and according to a Trapped strategy on O otherwise.

If P satisfies case 2, then on each one-way switch O adjacent to a, the Duplicator plays according to a Trapped strategy on $P_O \cup \{yy'\}$.

If P satisfies case 2a, then we play a Trapped strategy on $O_{ab} \cup \{xx'\}$ and $O_{ac} \cup \{xx'\}$, unless there are already $k-1$ pebbles placed on one of these subgraphs, in which case we just play according to a Trapped strategy on that subgraph. As before we play according to the identity everywhere outside the vicinity of a, and according to to an automorphism on H_a.

The duplicator plays similarly for the other cases.

This reduction is clearly in L, because we replace each gate in the circuit with a gadget whose size depends solely on what type of gate it is. We can easily keep track of the indices for the loop in a small amount of space. We have used 10 different colors for the reduction. We use three for the H and I gadgets, four for the rows of the One-way Switches, one for the input of the Twisted Switch, one to encode the values of the circuit inputs, and one base color for everything else.

Lemma 6. *In the $(\exists, k)$-pebble game on H^m_S, H^m_D, from a position $\{xx_j\}$, $j \in \{0, 1\}$, the Spoiler can reach $\{y^i y^i_j\}$ for any i.*

Proof. Clear based on vertex coloring.

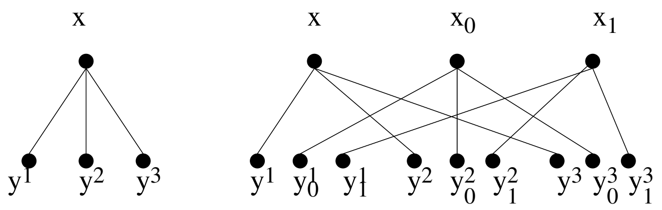

Fig. 13. H^m Gadget

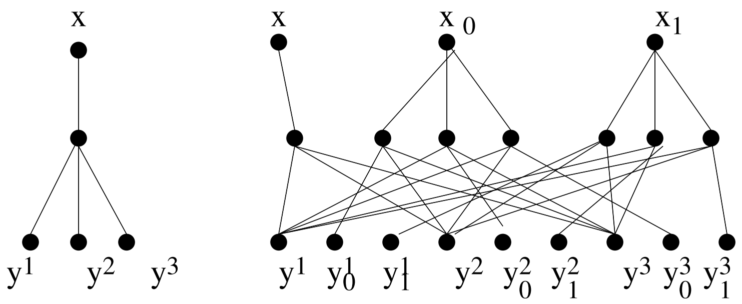

Fig. 14. I^m Gadget

Lemma 7. *In the $(\exists, k)$-pebble game on I^m_S, I^m_D, from a position $\{xx_j\}$, $j \in \{0,1\}$, the Duplicator can choose any $1 \leq i \leq m$, and avoid $\{y^l y^l_j\}$ for $l \neq i$.*

Proof. In a similar way to that used on the smaller I gadget, the Duplicator chooses an intermediate step which forces the Spoiler to play a certain way in order to carry information forward in the game.

Lemma 8. *For every $k \geq 2$ there exists a pair of graphs M^k_S, and M^k_D with*

$$\{x^1, x^1_0, x^1_1, \ldots, x^{k-1}, x^{k-1}_0, x^{k-1}_1, y^1, y^1_0, y^1_1, \ldots, y^{k-1}, y^{k-1}_0, y^{k-1}_1\} \subset V(M^k_D),$$

and $\{x^1, \ldots, x^{k-1}, y^1, \ldots, y^{k-1}\} \subset V(M^k_S)$, such that:

1. *From a position $\{x^i x^i_{j_i} | 1 \leq i \leq k-1, j_i \in \{0,1\}\}$, the Spoiler can reach the position $\{y^i y^i_{j_i} | 1 \leq i \leq k-1, j_i \in \{0,1\}\}$ in the $\exists k$-pebble game on M^k_S and M^k_D.*
2. *There exist two disjoint sets of positions of the $\exists k$-pebble game on (M^k_S, M^k_D), called Pretrapped and Trapped positions such that:*
   (a) *Pretrapped and Trapped positions are partial homomorphisms*
   (b) *The Duplicator can avoid positions that are not Pretrapped and not Trapped from Pretrapped positions*
   (c) *The Duplicator can avoid positions that are not Trapped from Trapped positions*
   (d) *From any position $P = \{x^i a | 1 \leq i \leq k-1\}$ where $|\{x^i x^i_j \in P | j \in \{0,1\}\}| < k-1$, the Duplicator can avoid $y^i y^i_j$ for all $1 \leq i \leq k-1, j \in \{0,1\}$.*

(e) All positions that are subsets of positions of the form $\{x^i x^i_{j_i} | 1 \leq i \leq k-1, j_i \in \{0,1\}\}$, are Pretrapped.
(f) If P is Pretrapped and $|P| < k$, then $P \cup \{y^i y^i\}$ is Pretrapped for all i
(g) Any position in which all of the Spoiler's pebbles on nodes y^i, is Trapped.
(h) If P is Trapped and $|P| < k$, then $P \cup \{x^i x^i\}$ is Trapped for all i

Proof. First, we define M^k_S and M^k_D. Then, we show that these graphs have the desired properties.

$$
\begin{aligned}
V(M^k_D) = {} & \{x^1, x^1_0, x^1_1, \ldots, x^{k-1}, x^{k-1}_0, x^{k-1}_1, y^1, y^1_0, y^1_1, \ldots, y^{k-1}, y^{k-1}_0, y^{k-1}_1\} \cup \\
& \{A^i_j | 1 \leq i \leq k-1, -(k-1) \leq j \leq k-1\} \cup \\
& \{B^i_j | 1 \leq i \leq k-1, -(k-1) \leq j \leq 1\} \\
E(M^k_D) = {} & \{(x^i, A^i_j) | 1 \leq i \leq k-1, -(k-1) \leq j \leq k-1\} \cup \\
& \{(x^i_0, A^i_j) | 1 \leq i \leq k-1, -(k-1) \leq j \leq -1\} \cup \\
& \{(x^i_1, A^i_j) | 1 \leq i \leq k-1, 1 \leq j \leq k-1\} \cup \\
& \{(A^i_j, A^l_m) | i \neq l, |j| \neq |m|\} \cup \\
& \{(B^i_j, B^l_m) | i \neq l, (j < 0 \wedge m < 0) \vee (j \geq 0 \wedge m \geq 0)\} \cup \\
& \{(A^i_j, B^l_m) | j \neq 0, -(k-1) \leq m \leq -1, |j| \neq ((l+m) \bmod k) + 1\} \cup \\
& \{(A^i_0, B^l_m)\} \cup \\
& \{(A^i_j, B^l_0) | (j < 0) \vee (i \neq l)\} \cup \\
& \{(A^i_j, B^l_1) | (j > 0) \vee (i \neq l)\} \cup \\
& \{(B^i_j, y^i) | 1 \leq i \leq k-1, -(k-1) \leq j \leq -1\} \cup \\
& \{(B^i_j, y^i_j) | 1 \leq i \leq k-1, j \in \{0,1\}\} \cup \\
& \{(B^i_j, y^i_k) | 1 \leq i \leq k-1, -(k-1) \leq j \leq -1, k \in \{0,1\}\}
\end{aligned}
$$

$$
\begin{aligned}
V(M^k_S) = {} & \{x^1, \ldots, x^{k-1}, y^1, \ldots, y^{k-1}\} \cup \\
& \{A^i | 1 \leq i \leq k-1\} \cup \\
& \{B^i | 1 \leq i \leq k-1\} \\
E(M^k_S) = {} & \{(x^i, A^i)\} \cup \\
& \{(A^i, A^j) | i \neq j\} \cup \\
& \{(A^i, B^j)\} \cup \\
& \{(B^i, B^j) | i \neq j\} \cup \\
& \{(B^i, y^i)\}
\end{aligned}
$$

One can picture this graph as a set of switches that are similar to the Single Input One-Way Switch, which are connected to each other. In particular, each individual switch needs to be colored a separate color. Figure A shows two such switches for

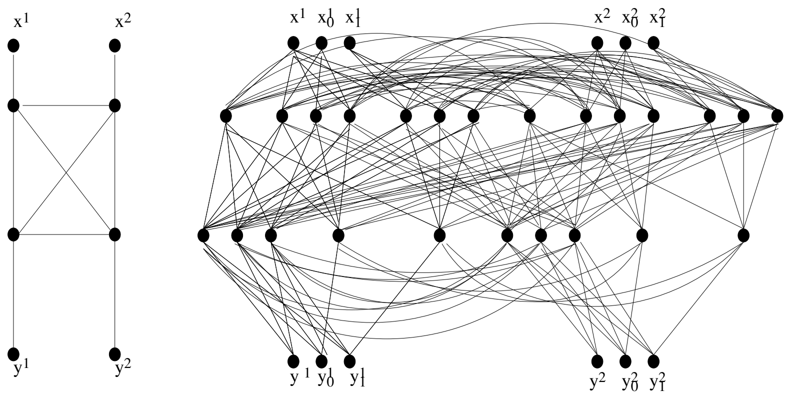

Fig. 15. A subgraph of M^4

the game on 4 pebbles. There would be one more in the full Multiple Input One-Way Switch, connected to each of these two in the same way.

From a position $P = \{x^i x^i_{j_i} | 1 \leq i \leq k-1, j_i \in \{0,1\}\}$, the Spoiler can take his extra pebble and place it on A^1. We know that x^1_0 is only connected to A^1_i where $i < 0$ and x^1_1 is only connected to A^1_i where $i > 0$. So, if $x^1 x^1_0 \in P$, the Duplicator must place a pebble on some A^1_i, $i < 0$, and vice versa. Now, the Spoiler picks up the pebble on x^1 and places it on A^2. Similarly, the Duplicator must play negative if her pebble is on x^2_0 and positive if her pebble is on x^2_1. There is an additional constraint now, because if in the previous move the players added $A^1 A^1_i$ to P, the Duplicator cannot play a pebble on A^2_i now, because there is no edge in M^k_D between A^1_i and A^2_i. After $k-1$ such moves, P' will consist of $x^{k-1} x^{k-1}_{j_{k-1}}$ and for each other i, $A^i A^i_{l_i}$, where $|l_i| \neq |l_j|$ for all $i \neq j$ and the signs of the l_i match the initial position.

The Spoiler next picks up the pebble on x^{k-1} and places it on B^1. For any $B^l_m \in M^k_D$, $m < 0$, there is a j, $|j| > 0$, such that $(A^i_j, B^l_m) \notin E(M^k_D)$ for any i. This means that the Duplicator must place a pebble on either B^1_0, or B^1_1 depending on the sign of A^1_j. Next, the Spoiler picks of the pebble on A^1 and places it on B^2. The Duplicator must respond by playing a pebble on B^2_0 or B^2_1, depending on the sign of A^2_j. This sequence is continued.

From here it is easy to see that the Spoiler can reach $\{y^i y^i_{j_i} | 1 \leq i \leq k-1, j_i \in \{0,1\}\}$.

Now we must exhibit a set of positions with the appropriate properties.

Let S be the set of all partial homomorphisms, P, from M_S to M_D such that:

1. $\exists i, j$ such that $x^j_i \in Im(P) \Rightarrow \nexists l, m$ such that $B^l_m \in Im(P)$
2. $\forall i, y^i_0 \in Im(P) \Rightarrow \nexists j < 0$ such that $a^i_j \in Im(P)$
3. $\forall i, y^i_1 \in Im(P) \Rightarrow \nexists j > 0$ such that $a^i_j \in Im(P)$

P is Pretrapped if $\exists i, j$ such that $x^j_i \in Im(P)$ and $P \in S$.

P is Trapped if $P \in S$ and P is not Pretrapped.

The last four properties of these sets of positions are easy to see. The difficulty arises in proving that given any Trapped position, the Duplicator can play so that the position remains Trapped, and given any Pretrapped position, the Duplicator can play so that the position is either Trapped or Pretrapped. We prove that given any position in S, the Duplicator can play so that the new position is also in S and if the position was Trapped then the new position is Trapped. Given a position in S, there are four basic types of moves that the Spoiler can make.

Case 1: $u = x^i$. The Duplicator can always play on x^i.

Case 2: $u = A^i$. If $x^i x^i_j \notin P$ for any j, then the Duplicator can play on A^i. Assume $x^i x^i_0 \in P$ (The case of $x^i x^i_1 \in P$ is symmetric). We need to find node A^i_j, such that $j < 0$, and A^i_j is connected to all of the chosen nodes on the A and B levels. Since there are strictly fewer than $k-1$ such nodes, there is a choice such that the partial homomorphism property is maintained.

Case 3: $u = B^i$. If $\exists l, m$ such that $x^l_m \in Im(P)$, then $exists j < 0$ such that $P \cup \{(b^i, b^i_j)\}$ is a partial homomorphism. If $\exists l, m \geq 0$ such that $b^l_m \in Im(P)$ then either b^i_0 or b^i_1 will maintain the partial homomorphism. Let $A = \{a^l_m \in Im(P) | m \neq 0\}$. If $|A| = k-1$ then $\exists j \neq 0$ such that $a^i_j \in A$. Then, if $j > 0$ the duplicator plays b^i_1 otherwise the duplicator plays b^i_0.

Case 4: $u = y^i$: If P is Pretrapped, then the Duplicator can play y^i. If P is Trapped, then the duplicator can play one of y^i, y^i_0, y^i_1.

Now we show that using this strategy, the Duplicator will avoid $y^i y^i_j$ for all $1 \leq i \leq k-1, j \in \{0,1\}$ from any position $P = \{x^i a | 1 \leq i \leq k-1\}$ where $|\{x^i x^i_j \in P | j \in \{0,1\}\}| < k-1$. We know that P is Pretrapped, so $P \cup \{y^i y^i_j | 1 \leq i \leq k-1, j \in \{0,1\}$ is Pretrapped. For sake of contradiction, assume that the Spoiler can play to $\{y^i y^i_j\}$ for some i, j. This means that the Duplicator would have played a pebble on B^i_l for some $l \geq 0$. There are two reasons that the Duplicator will play a pebble on B^i_l with $l \geq 0$: either there is an i', l' such that $B^{i'}_{l'}$ is in $Im(P')$; or P' contains $k-1$ pairs of the form $A^q A^q_m$ with $m \neq 0$. In the first case we can recurse to the second case. Now it is easy to see that the Duplicator would never actually play to this position, because for some i, the pair $x^i x^i \in P$. If the Spoiler plays on A^i, the Duplicator will play on A^i_0.

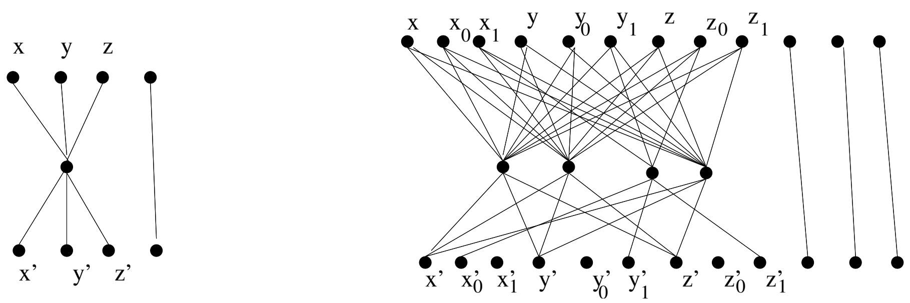

Fig. 16. The Rule Gadget that penalizes the Spoiler for choosing a bad rule

Lemma 9. *If the rule gadget RS^n does not correspond to a legal rule, then the Duplicator can play to $z'z'$.*

Proof. Each rule gadget corresponds to a rule $R(x, y, z)$. If $\{(x, x_1), (y, y_1), (z, z_0)\} \subseteq P$, then the Spoiler can reach $\{(x', x'_0), (y', y'_1), (z', z'_1)\}$. Otherwise, if the Spoiler plays on x', y', z', the Duplicator is forced to play on x', y', z'.

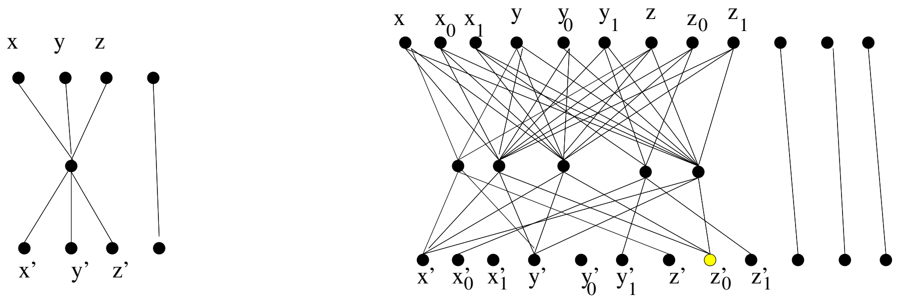

Fig. 17. The Rule Gadget that penalizes the Duplicator for choosing a bad rule

Lemma 10. *If the rule gadget RD^n does not correspond to a legal rule, then the Spoiler can play to zz_0, which causes the Duplicator to lose.*

Proof. If $\{(x, x_1), (y, y_1), (z, z_0)\} \subseteq P$, then the Spoiler can play to reach $\{(x', x'_0), (y', y'_1), (z', z'_1)\}$. If $\{(x, x), (y, y), (z, z)\} \subseteq P$, then play goes to $\{(x', x'), (y', y'), (z', z')\}$. Otherwise, The Spoiler can play to $\{(x', x'), (y', y'), (z', z'_0)\}$ which causes the Duplicator to lose because z'_0 is colored a distinct color.

Lemma 11. *For each gadget other than the Twisted Switch, starting from a position that is a subset of the identity on the boundary, the Duplicator can avoid any position that is not a subset of the identity on the other boundary.*

Proof. For H^m, I^m, RS^n and RD^n this is clear because for each X in this set, $X_S \subset X_D$. On the Multiple Input One-Way Switch, this follows from the properties proven in Lemma 8.

Theorem 12. *Determining the winner of the $(\exists, k)$-pebble game with k part of the input is EXPTIME-Complete.*

Proof. We perform a reduction from the KAI Game to the $(\exists, k)$-pebble game. An instance of the KAI Game consists of (X, S, R, t). Given this, we form an instance of the $(\exists, k)$-pebble game as follows.

The Duplicator's graph and the Spoiler's graph each have two sides. One side represents Player I's turn in the KAI game, while the other side represents Player II's turn.

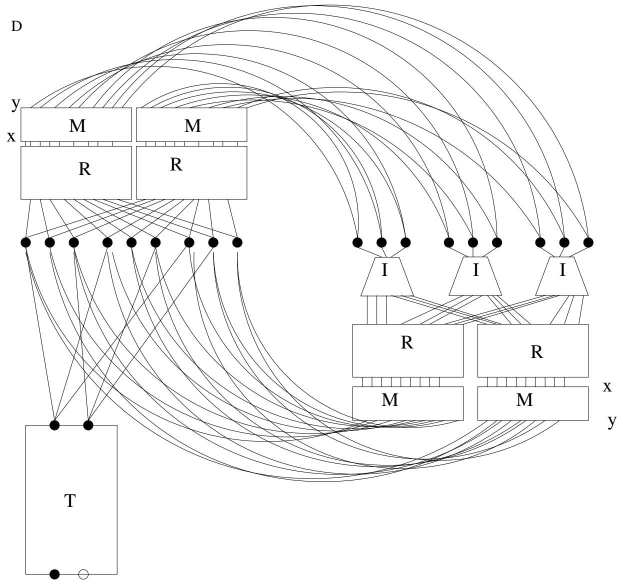

Fig. 18. This is component decomposition of the Duplicator's graph for the reduction

First, we build Player I's side of the graph. For each $x^i \in X$, we form three nodes in D, called xs^i, xs^i_0, xs^i_1. These three nodes correspond to specific information about the simulated KAI game. If there is a pebble on xs^i_1, then there is a pebble on x in the KAI game, and xs^i_0 corresponds to no pebble on x. A pebble on xs in the Duplicator's graph means that the Spoiler has made a mistake. For each $(x, y, z) \in R$, construct a rule gadget to penalize the Spoiler, connected to each xs, xs_0, xs_1 by H^m gadgets. The other end is then connected to a multi input one way switch.

On the other half of the graph, there are another set of nodes, xd^i, xd^i_0, xd^i_1. Connecting the xd nodes to Duplicator punishing rule gadgets are a set of I^m gadgets where $m = |R|$. Given any two of these I^m gadgets A and B, there is an edge between $A.y^i_j$ and $B.y^i_k$ for all i, j, k, and an edge between $A.y^i$ and $B.y^i$ for all i. We then connect a Multiple Input One-Way Switch to each of the rule gadgets in the obvious way. The outputs of these are connected back around to the first set of state nodes.

We use a Twisted Switch to set up the initial positions. If $x^i \in S$, then there is an edge from y' of the Twisted Switch to xs^i_1 otherwise there is an edge from y' to xs^i_0. There is an edge from y to xs^i for every i. We then color x' a unique color.

The Spoiler's graph is constructed in a similar way. The Spoiler gets xs^i and xd^i for every $x^i \in X$. Also, for each rule and each side, there is a corresponding Multiple Input One-Way Switch followed by a rule gadget. For the Twisted Switch, we color x the same color that we colored x' in the Duplicator's graph.

First, we show that if Player I has a forced win in the KAI game, then the Spoiler can win the $(\exists, k)$-pebble game. A forced win consists of a strategy such that on Player

I's turn, there is a legal rule such that for every choice Player II makes, Player I arrives back at a position with a forced win.

If the Spoiler plays on x in the Twisted Switch, the Duplicator must play on x'. This allows the Spoiler to play the initial position of the KAI game. Since we know that there is a rule that corresponds to a forced win strategy for Player I in the KAI game, the Spoiler can use the H^m gadgets to choose that rule. Then, the Spoiler can traverse the Multiple Input One-Way Switch, maintaining the values of the nodes. Since this rule was chosen from a forced win strategy for Player I, we know that it is a legal rule to apply, so the Spoiler can play to the xd with the value of xd corresponding to the simulated game after the chosen rule has been applied.

Now, the Spoiler continues play along the I^m gadgets. We know that the Spoiler can reach some rule along these gadgets, but the Duplicator gets to pick which one. The Duplicator must pick a single rule because of the connections among the outputs of the I^m gadgets. This means that the Spoiler can reach the Multiple Input One-Way Switch that corresponds to some rule, with all of the information intact. Now, the Spoiler plays through the Multiple Input One-Way Switch and the rule gadget to return to the xs side of the structure. Since the Spoiler is playing according to a forced win strategy, there is a forced win strategy from this position.

Eventually, the Spoiler will place a pebble on td, and the Duplicator will be forced to place a pebble on td_1. Since these are different colors, the Spoiler wins.

Now, assuming that Player I does not have a forced win in the KAI game, we must exhibit a winning strategy for the Duplicator.

Consider a configuration $C = (O, r)$ of the KAI game where O is the set of nodes which have pebbles on them and r is a boolean representing which player's turn it is. We call a configuration of the KAI game critical for a position P of the $(\exists, k)$-pebble game if one of the following is true.

1. The number of active nodes is $|S|$, it is the Spoiler's turn and
   (a) For each Multiple Input One-Way Switch MD^j on Player II's side, $P|_{MD^j} \cup \{y^i y_1^i | x^i \in O\} \cup \{y^i y_0^i | x^i \notin O\}$ is Trapped.
   (b) For each Multiple Input One-Way Switch MS^j on Player I's side that corresponds to a rule (x, y, z) such that $x, y \in O$ and $z \notin O$, $P|_{MS^j} \cup \{x^i x_1^i | x^i \in O$ $\{x\} \cup \{z\}\} \cup \{x^i x_0^i | x^i \notin O \setminus \{x\} \cup \{z\}\}$ is Pretrapped.
   (c) For each Multiple Input One-Way Switch MS^j on Player I's side that corresponds to a rule (x, y, z) such that either $x \notin O$, $y \notin O$, or $z \in O$, then $P|_{MS^j} \cup \{y^i y_1^i | x^i \in O \setminus \{x, y, z\}\} \cup \{y^i y_0^i | x^i \notin O \cup \{x, y, z\}\} \cup \{y^i y^i | x^i \in \{x, y, z\}\}$ is Pretrapped.
2. The number of active nodes is less than $|S|$, it is the Spoiler's turn and there exists x^k such that
   (a) For each Multiple Input One-Way Switch MD^j on Player II's side, $P|_{MD^j} \cup \{y^i y_1^i | x^i \in O\}$ is Trapped, $x^k \notin O$, $P|_{MD^j} \cup \{x^k x^k\}$ is Trapped.
   (b) For each Multiple Input One-Way Switch MS^j on Player I's side that corresponds to a rule (x, y, z) such that $x, y \in O$ and $z \notin O$, $P|_{MS^j} \cup \{x^i x_1^i | x^i \in O \setminus \{x\}\}$ is Pretrapped, and $x_k \notin O$, $P|_{MD^j} \cup \{x^k x^k\}$ is Pretrapped.
   (c) For each Multiple Input One-Way Switch MS^j on Player I's side that corresponds to a rule (x, y, z) such that either $x \notin O$, $y \notin O$, or $z \in O$, then $P|_{MS^j} \cup \{y^i y_1^i | x^i \in O \setminus \{x, y, z\}\} \cup \{y^i y^i | x^i \in \{x, y, z\}\}$ is Pretrapped.

3. The number of active nodes is $|S|$, it is the Duplicator's turn and
   (a) For each Multiple Input One-Way Switch MS^j on Player I's side, $P|_{MS^j} \cup \{y^i y_1^i | x^i \in O\} \cup \{y^i y_0^i | x^i \notin O\}$ is Trapped.
   (b) For each Multiple Input One-Way Switch MD^j on Player II's side that corresponds to a rule (x, y, z) such that $x, y \in O$ and $z \notin O$, $P|_{MD^j} \cup \{x^i x_1^i | x^i \in O \setminus \{x\} \cup \{z\}\} \cup \{x^i x_0^i | x^i \notin O \setminus \{x\} \cup \{z\}\}$ is Pretrapped.
   (c) For each Multiple Input One-Way Switch MD^j on Player II's side that corresponds to a rule (x, y, z) such that either $x \notin O$, $y \notin O$, or $z \in O$, then $P|_{MD^j} \cup \{y^i y_1^i | x^i \in O \setminus \{x, y, z\}\} \cup \{y^i y_0^i | x^i \notin O \cup \{x, y, z\}\} \cup \{y^i y^i | x^i \in \{x, y, z\}\}$ is Pretrapped.
4. The number of active nodes is less than $|S|$, it is the Duplicator's turn and there exists x^k such that
   (a) For each Multiple Input One-Way Switch MS^j on Player I's side, $P|_{MS^j} \cup \{y^i y_1^i | x^i \in O\}$ is Trapped, $x^k \notin O$, $P|_{MS^j} \cup \{x^k x^k\}$ is Trapped.
   (b) For each Multiple Input One-Way Switch MD^j on Player II's side that corresponds to a rule (x, y, z) such that $x, y \in O$ and $z \notin O$, $P|_{MD^j} \cup \{x^i x_1^i | x^i \in O \setminus \{x\}\}$ is Pretrapped, and $x_k \notin O$, $P|_{MD^j} \cup \{x^k x^k\}$ is Pretrapped.
   (c) For each Multiple Input One-Way Switch MD^j on Player II's side that corresponds to a rule (x, y, z) such that either $x \notin O$, $y \notin O$, or $z \in O$, then $P|_{MD^j} \cup \{y^i y_1^i | x^i \in O \setminus \{x, y, z\}\} \cup \{y^i y^i | x^i \in \{x, y, z\}\}$ is Pretrapped.

Let the safe positions be the set of partial homomorphisms P such that $P|_T$ is Twisted and there is a configuration of the KAI game that is not on a winning path for Player I, which is critical for P. Notice that any position where the empty configuration is critical is a winning position for the Duplicator.

What we need to show is that for any safe position, the Duplicator can play in such a way as to avoid non-safe positions. The empty position is safe. As an example, consider a safe position P in which the configuration $(O, 1)$ is critical for P. If the Spoiler plays in any MD^j, then the Duplicator plays according to a Trapped strategy on MD^j. If the Spoiler plays on T, then the Duplicator plays according to a Twisted strategy on T. If the Spoiler plays on RS^i, the Duplicator has no choices. If the Spoiler plays on MS^j, then the Duplicator plays according to a Pretrapped strategy on MS^j. If $|O| < |S|$, then as proven earlier, the Duplicator can avoid $y^i y_j^i$ on all MS^j. Also, if MS^j corresponds to a rule that is illegal in configuration O, then the Duplicator can also avoid $y^i y_j^i$ for all MS^j.

The Duplicator plays similarly on the other side of the graph when a configuration $(O, 2)$ is critical for P.