Game Quantification on Automatic Structures and Hierarchical Model Checking Games *

Łukasz Kaiser
kaiser@informatik.rwth-aachen.de

Mathematische Grundlagen der Informatik, RWTH Aachen
Ahornstasse 55, 52074 Aachen, Germany

Abstract. Game quantification is an expressive concept and has been studied in model theory and descriptive set theory, especially in relation to infinitary logics. Automatic structures on the other hand appear very often in computer science, especially in program verification. We extend first-order logic on structures on words by allowing to use an infinite string of alternating quantifiers on letters of a word, the game quantifier. This extended logic is decidable and preserves regularity on automatic structures, but can be undecidable on other structures even with decidable first-order theory. We show that in the presence of game quantifier any relation that allows to distinguish successors is enough to define all regular relations and therefore the game quantifier is strictly more expressive than first-order logic in such cases. Conversely, if there is an automorphism of atomic relations that swaps some successors then we prove that it can be extended to any relations definable with game quantifier. After investigating it's expressiveness, we use game quantification to introduce a new type of combinatorial games with multiple players and imperfect information exchanged with respect to a hierarchical constraint. It is shown that these games on finite arenas exactly capture the logic with game quantifier when players alternate their moves but are undecidable and not necessarily determined in the other case. In this way we define the first model checking games with finite arenas that can be used for model checking first-order logic on automatic structures.

1 Introduction

Game quantification, the use of infinite strings of quantifiers $Q_1x_1Q_2x_2\ldots$ with $Q_i = \forall$ or $\exists$, is an intuitive and expressive concept and has interesting connections to model theory, infinitary logics and descriptive set theory [10]. A formula with game quantifiers, e.g.

$$(\exists x_1 \forall y_1 \exists x_2 \forall y_2 \ldots) R(x_1, y_1, x_2, y_2, \ldots),$$

where R is a set of infinite sequences, is normally interpreted using Gale-Stewart games. In the corresponding game $G(\exists\forall, R)$ two players alternatively choose

* This research has been partially supported by the European Community Research Training Network "Games and Automata for Synthesis and Validation" (GAMES)

elements of the structure and the first player wins (and the formula is true) if the resulting sequence belongs to R.

Traditionally game quantification was investigated on open or closed sets R, i.e. sets that are defined as infinite disjunctions or conjunctions of finitary relations, $R(\overline{x}) = \bigvee_n R_n(x_1, \ldots, x_n)$. In such cases the formulas with alternating quantifiers can be identified with the monotone open game quantifier $\mathcal{G}^\exists$ or the dual closed game quantifier $\mathcal{G}^\forall$. The duality of these quantifiers ($X \in \mathcal{G}^\exists \iff X \notin \mathcal{G}^\forall$) results from the determinacy of Gale-Stewart games for open and closed sets [7], which was extended by Martin to any Borel set [11].

We are going to introduce game quantification for presentations of automatic structures, i.e. for structures over finite or infinite sequences of letters chosen from a finite alphabet where each relation R is recognised by a finite Muller automaton. Automatic structures, for example Presburger arithmetic, are often used in computer science. They appear in verification problems, have decidable first-order theory [4] and are actively investigated (see [9, 1] and references therein). Automatic relations are Borel, so we can use the duality result mentioned before, but we look more closely at the games that appear in this setting. It turns out that we can not only bring the formulas to negation normal form, but we can as well give a computable procedure to construct the automaton recognising the set defined by any formula with game quantifiers and thus show that such formulas are decidable.

The expressive power of game quantification is traditionally compared to infinitary logics over the structure of elements and is most evident in the formula that allows to compare order types of two elements with respect to given orderings. In our case the alphabet is finite and therefore our reference point will be first-order logic over finite or infinite sequences of letters, i.e. over the considered presentation of an automatic structure. It turns out that a formula similar in some way to the one comparing order types allows us to compare the length of common prefixes of words. Using this we are able to show that on some automatic structures game quantification is indeed stronger than first-order logic and we investigate its expressiveness in more detail. On the other hand, it follows from the decidability result that the logic with game quantifier collapses to first-order logic on complete-automatic structures.

To gain deeper insight into definability in the presence of game quantifier on weaker automatic structures we look for automorphisms of structures that are invariant for the logic we study. Similar to the action of permutations of ω on countable models of sentences in infinitary logic studied by invariant descriptive set theory, we define a family of inductive automorphisms where permutation of the alphabet is applied on each position separately and show that these are invariant for the logic with game quantification. This completes the picture of the dependency between expressibility in logic with game quantification and possibility to distinguish different successors.

After analysing the logic with game quantifier we define a family of multiplayer Muller games with imperfect information shared in a hierarchical way. Such games, even when played on a small arenas, can be used to model complex

interactions between players and can be used for model checking. Expressing the semantic of a logic by means of games has proved fruitful for developing model checking algorithms [8], especially for μ-calculus which corresponds to parity games [6]. Additionally, the game semantic is quite intuitive and we use multi-player Muller games with imperfect information [2], which is interesting as these types of games have so far not been widely used for model-checking.

We start investigating this class of games by showing that they are not necessarily determined and undecidable if players are not forced to alternate their moves. On the other hand, when players alternate moves we prove the exact correspondence between the games and the logic with game quantification. More precisely, the games can be used as model checking games on automatic structures for first-order logic with game quantifier and at the same time the winning region can be defined in this logic. It follows that deciding the winner is non-elementary in this case. Still, we argue that these games can give rise to efficient algorithms for model checking on complex structures, since recently developed algorithms for games with semiperfect information [5] could be used in practical cases.

2 Preliminaries

In this paper we will be working mainly with structures on words, finite or infinite sequences of letters from a finite alphabet Σ. We denote by Σ^* the set of finite words over Σ and by Σ^ω the set of infinite words, $\Sigma^{\leq\omega} = \Sigma^* \cup \Sigma^\omega$. We normally assume that Σ is fixed and that it contains at least two elements, in our examples usually $\Sigma = \{a, b\}$, and when we need an element not in the alphabet we denote it by $\square \notin \Sigma$.

Let us fix the notation used for operations on words. For any sequence or word w let us denote by $w|_n$ the finite word composed of the first n letters of w, with $w|_0 = \varepsilon$, the empty word or sequence, and by $w[n]$ the nth letter or element of w for $n = 1, 2, \ldots$. We say that $v \sqsubseteq w$ if v is a prefix of w and in such case we denote by $w - v$ the word u such that $v \cdot u = w$. For an infinite word $w \in \Sigma^\omega$ the set of letters that appear infinitely often in this word is denoted by $\mathrm{Inf}(w)$. We sometimes extend all the notations presented here to vectors of words, so for example if $\overline{x}$ is a tuple of words then $\overline{x}[n]$ is a tuple consisting of the nth letter of each word in $\overline{x}$.

2.1 Automatic Structures

We are going to analyse inductive structures modelled over finite and infinite words, so formally we consider the following structure:

$$(\Sigma^{\leq\omega}, R_1, \ldots, R_K),$$

where each relation R_i has arity $\mathrm{ar}(i)$, so $R_i \subseteq (\Sigma^{\leq\omega})^{\mathrm{ar}(i)}$. Sometimes we want the relations to be recognised by automata and in such cases we will consider

them as ω-languages over the tuple-alphabet extended with $\Box$ for finite words, $\otimes R_i \subseteq ((\Sigma \cup \{\Box\})^{\mathrm{ar}(i)})^\omega$.

To define the relations $\otimes R_i$ we have to compose infinite words over the tuple-alphabet $(\Sigma \cup \{\Box\})^{\mathrm{ar}(i)}$ from finite and infinite words over Σ. In such case, if we have a number of words $w^1 = x_1^1 x_2^1 \ldots$ and so up to $w^k = x_1^k x_2^k \ldots$, then we denote the composed word by $\otimes\overline{w} =$

$$w^1 \otimes \ldots \otimes w^k = \begin{bmatrix} x_1^1 \\ \vdots \\ x_1^k \end{bmatrix} \begin{bmatrix} x_2^1 \\ \vdots \\ x_2^k \end{bmatrix} \ldots \in ((\Sigma \cup \{\Box\})^k)^\omega,$$

whereas if some w^l was finite, $w^l = x_1^l x_2^l \ldots x_L^l$, then we prolong it with $\Box^\omega$, i.e. $x_{L+i}^l = \Box$. This allows us to define $\otimes R_i$ with respect to R_i by $R_i(w_1, \ldots, w_k) \iff \otimes R_i(w_1 \otimes \ldots \otimes w_k)$.

To speak about presentations of ω-automatic structures we will use Muller automata to recognise ω-regular languages. A (deterministic) *Muller automaton* over $\Gamma = \Sigma \cup \{\Box\}$ is a tuple $\mathcal{A} = (Q, \delta, q_0, \mathcal{F})$ where Q is a finite set of states, δ is a state transition function $\delta : Q \times \Gamma \to Q$, $q_0 \in Q$ is the initial state and $\mathcal{F} \subseteq \mathcal{P}(Q)$ is the acceptance condition. A *run* of $\mathcal{A}$ on $w \in \Gamma^\omega$ is a sequence

$$\rho_\mathcal{A}(w) = q_0, q_1, \ldots \in Q^\omega \text{ such that } q_i = \delta(q_{i-1}, w[i]).$$

The word w is *accepted* by $\mathcal{A}$ if the set of states appearing infinitely often during the run is in the acceptance condition, also when $\mathrm{Inf}(\rho_\mathcal{A}(w)) \in \mathcal{F}$, and a language $L \subseteq \Gamma^\omega$ is ω-regular if there is a Muller automaton $\mathcal{A}$ that accepts exactly the words $w \in L$. A structure is automatic, or actually, as we consider only structures on words, is a presentation of an automatic structure, if for each relation R_i in this structure the language $\otimes R_i$ is ω-regular over $(\Sigma \cup \{\Box\})^{\mathrm{ar}(i)}$.

You should note that since we allow both finite and infinite words all our words when interpreted over $\Sigma \cup \{\Box\}$ have the property that if a $\Box$ appears then $\Box^\omega$ follows.

2.2 Alternating Automata

We have introduced the standard notion of automata, but we still need to present alternating Muller automata which are an important tool in our proofs. The intuition behind alternating automata is that, unlike in the deterministic case when only one run on a given word is possible, we have more possibilities of transitions from each state for a given letter. Moreover, we do not only want to accept when there *exists* an accepting run among all possible ones (nondeterministic automata), or when *all* possible runs are accepting (universal automata), but we want to be able to alternate the conditions with respect to states of the automaton, so to have both existential and universal branching choices.

To define alternating automata we have to consider, for a given set of states$Q$, the set $\mathcal{B}^+(Q)$ of all *positive boolean formulas* over Q. By definition $\mathcal{B}^+(Q)$ is the set of all boolean formulas built using elements of Q, the junctors $\wedge$ and $\vee$ and

the constants $\top$ (true) and $\bot$ (false). We will say that a subset $X \subseteq Q$ *satisfies* a formula $\varphi \in \mathcal{B}^+(Q)$ if φ is satisfied by the assignment that assigns true to all elements of X and false to $Q \setminus X$.

An *alternating (Muller) automaton* is then a tuple $\mathcal{A} = (Q, \delta, q_0, \mathcal{F})$ where as before Q is the set of states, q_0 is the initial state, $\mathcal{F} \subseteq \mathcal{P}(Q)$ is the acceptance condition, but this time δ does not point to a single next state but specifies a whole boolean condition,

$$\delta \ : \ Q \times \Gamma \to \mathcal{B}^+(Q).$$

Intuitively a *correct run* of $\mathcal{A}$ on a word w is an infinite tree labelled with Q where the successors of each node form a satisfying set for the boolean condition related to the state in this node and to the corresponding letter in w.

To capture this intuition formally we will represent runs — infinite trees — as sets of branches of the tree, so a run $R \subseteq Q^\omega$. For a run R we define a *branching set* after a finite word u as the set of all letters appearing in words prolonging u,

$$b_R(u) = \{c \in Q \ : \ \exists v \ u \cdot c \cdot v \in R\}.$$

In this formalisation R is a correct run of $\mathcal{A}$ on the word w when for each $u \in R$ and each prefix $u|_i$ the branching set after that prefix satisfies the corresponding boolean constraint,

$$b_R(u|_i) \text{ satisfies } \delta(u[i], w[i]).$$

We can now define that $\mathcal{A}$ accepts a word w if there is a correct non-empty run R on w starting from q_0 such that each branch $r \in R$ is accepted, $\mathrm{Inf}(r) \in \mathcal{F}$. Of course a language $L \subseteq \Gamma^\omega$ is recognised by an alternating automaton if the automaton accepts exactly the words $w \in L$.

Alternating automata may seem to be more powerful than deterministic ones and it is often much easier to express problems in terms of alternating automata than in terms of deterministic ones, but the following theorem guarantees that we can always replace an alternating automaton with a deterministic one.

Theorem 1. *Every language recognised by an alternating Muller automaton is ω-regular, i.e. there exists a deterministic Muller automaton that recognises it.*

The above theorem can be proved by expressing acceptance of alternating automata in monadic second-order logic on infinite word ($S1S$) and then going back from the logic to automata [3]. However, more effort must be done to see that the deterministic automaton in the above theorem does not need to be bigger than double exponential in the size of the alternating one [12].

3 Game Quantifier on Automatic Structures

We want to extend first-order logic to make explicit use of the inductive structure of the words and therefore let us introduce $\Game$, the game quantifier. The meaning of the formula $\Game xy \ \varphi(x, y)$ is that φ can be satisfied when the arguments are

constructed stepwise by two players, i.e. first the first letter of x, then the first letter of y given by the second player, another letter of x by the first player and so on. Formally the play will be infinite so to capture finite words we have to define it on $\Gamma = \Sigma \cup \{\Box\}$ by

$$\begin{aligned}\eth xy\ \varphi(x,y) \iff\ & (\exists \text{ well-formed } f\ :\ \Gamma^* \times \Gamma^* \to \Gamma)\\ & (\forall \text{ well-formed } g\ :\ \Gamma^* \times \Gamma^* \to \Gamma)\ \varphi(x_{fg}, y_{fg}),\end{aligned}$$

where x_{fg} and y_{fg} are the Σ-words constructed inductively using f and g up to the first appearance of $\Box$,

$$\begin{aligned}x_{fg}[n+1] &= f(x_{fg}|_n, y_{fg}|_n),\\ y_{fg}[n+1] &= g(x_{fg}|_{n+1}, y_{fg}|_n),\end{aligned}$$

and well-formedness means that if any of the functions f resp. g outputs $\Box$ then the word x_{fg} resp. y_{fg} is considered to be finite and the function must then continue to output $\Box$ infinitely, formally h is well-formed when

$$h(w,u) = \Box \implies (\forall w' \sqsupseteq w)\ (\forall u' \sqsupseteq u)\ h(w',u') = \Box.$$

Please note that this direct definition coincides with the traditional one that uses infinite string of quantifiers,

$$\eth xy\ \varphi(x,y) \iff (\exists a_1 \forall b_1 \exists a_2 \forall b_2 \ldots)\ \varphi(a_1 a_2 \ldots, b_1 b_2 \ldots).$$

Moreover, using our notation, $\eth xy\ \varphi(x)$ is equivalent to $\exists x\ \varphi(x)$ as we can always forget opponent moves and play letters from x or conversely use any g to obtain the witness x. Similarly $\eth xy\ \varphi(y)$ is equivalent to $\forall y\ \varphi(y)$. Thus, we do not need to consider the standard quantifiers when the game quantifier is present.

On some structures it is possible to encode a pair of words into a single one, but that is not always the case. Therefore we might sometimes need to use the game quantifier with more variables:

$$\begin{aligned}&\eth x_1 \ldots x_k y_1 \ldots y_m\ \varphi(\overline{x}, \overline{y}) \iff\\ &\quad (\exists f\ :\ (\Gamma^*)^k \times (\Gamma^*)^m \to \Gamma^k)\ (\forall g\ :\ (\Gamma^*)^k \times (\Gamma^*)^m \to \Gamma^m)\ \varphi(\overline{x}_{fg}, \overline{y}_{fg}),\end{aligned}$$

where again the functions must be well–formed in each column and

$$\overline{x}_{fg}[n+1] = f(\overline{x}_{fg}|_n, \overline{y}_{fg}|_n),\ \overline{y}_{fg}[n+1] = g(\overline{x}_{fg}|_{n+1}, \overline{y}_{fg}|_n).$$

As an example of the use of game quantifier let us consider the following relation R given by the formula:

$$R(u,w,s,t) := \eth xy\ (y = u \to x = s) \wedge (y = w \to x = t).$$

We claim, that R means that the common prefix of s and t is longer than the common prefix of u and w. Denoting by $v \sqcap r$ the common prefix of v and r and by $|v|$ the length of v we can say, that

$$R(u,w,s,t) \equiv |u \sqcap w| < |s \sqcap t|$$

for $u \neq w$ and $s \neq t$. The way we think about evaluating such formula is by means of a game played by two players – the Verifier for x and the Falsifier for y. To see the above equivalence, let us assume that indeed the common prefix of s and t is longer than the common prefix of u and w. In this case the Falsifier will have to choose $y = u$ or $y = w$ before the Verifier chooses if $x = s$ or if $x = t$, and therefore the Verifier is going to win. In the other case, the Falsifier can make the formula false as he knows if $x = s$ or if $x = t$ before choosing whether $y = u$ or $y = w$.

3.1 Basic Properties of FO+∋

The two most important properties of FO+∋ that interest us are the decidability of it on ω-automatic structures and the existence of negation normal form, which semantically corresponds to the determinacy of the underlying games.

To be able to clearly state the existence of negation normal form let us introduce another variation of game quantifier, namely one where it is the Falsifier who makes the moves first. Formally, let

$$\ni^{\forall} xy\ \varphi(x,y) \iff (\exists f\ :\ \Gamma^* \times \Gamma^* \to \Gamma)\ (\forall g\ :\ \Gamma^* \times \Gamma^* \to \Gamma)\ \varphi(x^{\forall}_{fg}, y^{\forall}_{fg}),$$

where again the functions must be well-formed and this time the words are constructed in reverse order,

$$y^{\forall}_{fg}[n+1] = g(x^{\forall}_{fg}|_n, y^{\forall}_{fg}|_n),\ x^{\forall}_{fg}[n+1] = f(x^{\forall}_{fg}|_n, y^{\forall}_{fg}|_{n+1}).$$

If we denote the game quantifier introduced before by $\ni^{\exists}$ then the intended relation that leads to negation normal form can be stated as follows:

$$\ni^{\exists} xy\ \varphi(x,y) \equiv \neg \ni^{\forall} yx\ \neg\varphi(x,y).$$

Please note that when the relation of prefixing with a letter is present, the quantifier $\ni^{\forall}$ is superfluous and can be eliminated by adding one arbitrary letter,

$$\ni^{\forall} xy\ \varphi(x,y) \iff \ni^{\exists} zy\ \exists x\ z = ax \wedge \varphi(x,y).$$

To verify this equivalence, please note that on the right side the Verifier must start with an a and later play a strategy that satisfies φ, so the same strategy without the first a can be used on the left side. Conversely, if Verifier's strategy on the left side is given then playing an a and later the same strategy is winning for the right side.

To prove decidability and the existence of negation normal form we actually need one crucial lemma, namely that if we begin with ω-regular relations then anything defined in the FO+∋ logic remains ω-regular. The proof relies on the fact that, when used on an automaton, the game quantifier indeed constructs a game and changes the automaton to an alternating one.

Lemma 1. *If the relation $R(\overline{x}, \overline{y}, \overline{z})$ is ω-regular over $\overline{x} \otimes \overline{y} \otimes \overline{z}$ then the relation $S(\overline{z}) \iff \ni \overline{xy}\ R(\overline{x}, \overline{y}, \overline{z})$ is ω-regular over $\otimes \overline{z}$.*

Proof. Let us take the deterministic automaton $\mathcal{A}_R$ for R over $\overline{x} \otimes \overline{y} \otimes \overline{z}$ and construct an alternating automaton $\mathcal{A}_S$ for S over $\otimes\overline{z}$ in the following way. The set of states, acceptance condition and initial state remain the same and the new transition relation is defined by

$$\delta_S(q, \overline{z}) = \bigvee_{\overline{x} \in \Gamma^k} \bigwedge_{\overline{y} \in \Gamma^l} \delta_R(q, \overline{x} \otimes \overline{y} \otimes \overline{z}),$$

where k is the number of elements of $\overline{x}$ and l is the number of elements of $\overline{y}$.

By definition, the semantic of the relation S is

$$S(\overline{z}) \iff (\exists f \ : \ (\Gamma^*)^k \times (\Gamma^*)^l \to \Gamma^k)(\forall g \ : \ (\Gamma^*)^k \times (\Gamma^*)^l \to \Gamma^l) \ \varphi(\overline{x}_{fg}, \overline{y}_{fg}, \overline{z}).$$

One can see that the function f in this definition corresponds to the choice of the letters for x in the boolean formula when selecting the run of the alternating automaton and that the function g corresponds to the choice of the branch of the run, as all need to be accepted. ∎

This lemma immediately gives us decidability of FO+∋ on automatic structures and also allows us to use determinacy of Muller games for the proof for game quantifier inversion.

Corollary 1. *FO+∋ is decidable on ω-automatic structures, all relations definable in it are ω-automatic and for a fixed number of quantifier alternations it has elementary complexity.*

Corollary 2. *For each FO+∋ formula φ on every automatic structure $\mathfrak{A}$*

$$\mathfrak{A}, \overline{z} \models \ni^{\exists} \overline{xy} \ \varphi(\overline{x}, \overline{y}, \overline{z}) \iff \mathfrak{A}, \overline{z} \models \neg \ni^{\forall} \overline{yx} \ \neg\varphi(\overline{x}, \overline{y}, \overline{z}).$$

The last corollary follows from the determinacy of finitely coloured Muller games. You should note that because of $\overline{z}$ the game arena itself might be infinite, but the number of colours depends only on the size of Muller automaton for φ and is therefore finite. As was already mentioned the determinacy of Muller games can be derived from a more general result by Martin [11] which can be used to generalise the corollary to a wider class of structures, namely all where the relations are Borel sets.

3.2 Expressive Power of Game Quantification

Some automatic structures are known to be *complete*, meaning that every regular relation over such structure can be defined in first-order logic. For structures over finite and infinite words the canonical example of such structure is the binary tree, $\mathcal{T} = (\{a, b\}^{\leq\omega}, \sigma_a, \sigma_b, \sqsubseteq, \mathrm{el})$, where σ_a and σ_b denote a and b-successors of a finite word (i.e. $\sigma_a(u, ua)$), $\sqsubseteq$ is the prefix relation and $\mathrm{el}(x, y)$ means that x and y have equal length. Each automatic and ω-automatic relation over $\{a, b\}^{\leq\omega}$ can be described by an FO formula over this structure, so since FO+∋ relations are automatic by Lemma 1, then FO+∋ is as strong as FO in such case.

This situation changes when $\sqsubseteq$ and el are not given as then FO+∋ can be used to define them using just σ_a and σ_b and is therefore complete and stronger than FO.

Fact 1. *On the structure* $(\{a,b\}^{\leq\omega}, \sigma_a, \sigma_b)$ *the logic FO+∋ can define all regular relations and is therefore stronger than FO.*

Proof. First let us recall a few basic formulas that we are going to use. As we have already shown in the example we can use the game quantifier to talk about the length of common prefix of words, i.e. for $u \neq w, s \neq t$ we can say $|s \sqcap t| < |u \sqcap w|$ and the other variants with $\leq, =, \geq$ and $>$ are expressible using boolean combinations and argument permutations.

To say that x is a prefix of y we are going to say that no word $z \neq x$ has longer common prefix with x than y,

$$x \sqsubseteq y \;\equiv\; (x = y) \;\vee\; (\forall z \neq x)\; |x \sqcap z| \leq |x \sqcap y|.$$

To define equal length we will again use the $|s \sqcap t| < |u \sqcap w|$ relation to define that $|x| \leq |y|$. We will say that for any $x' \neq x$ there is an $y' \neq y$ that has common prefix with y not shorter that the common prefix of x' and x:

$$|x| \leq |y| \;\equiv\; (\forall x' \neq x)\; (\exists y' \neq y)\; |x \sqcap x'| \leq |y \sqcap y'|.$$

We can of course use boolean combinations to define $|x| = |y|$ and in this way both $\sqsubseteq$ and el are defined and thus any regular relation can be defined. ∎

You should note that the above proof does not make any use of the successor relations to define $\sqsubseteq$ and el.

Let us now take a weaker structure, namely $(\{a,b\}^{\leq\omega}, S_a)$ where $S_a(x,y)$ is any relation with the property that for each $x \in \{a,b\}^*$ it holds $S_a(x, xa)$ but $S_a(x, xb)$ does *not* hold. We did not specify how the relation S_a behaves on words of bigger difference in length, but this can be compensated for using $\sqsubseteq$ and el. Therefore with game quantifier the relation S_a is enough to express successors in the following way:

$$|x| = |y| + 1 \equiv |x| < |y| \wedge \forall z\; |z| < |y| \rightarrow |z| \leq |x|,$$

$$\sigma_a(x) = (y \equiv S_a(x,y) \wedge |y| = |x| + 1),\; \sigma_b(x) = (y \equiv \neg S_a(x,y) \wedge |y| = |x| + 1).$$

When one considers encoding natural numbers as binary words and analysing such structure, it is necessary to have a relation EQ that defines the equality between numbers as opposed to equality over words which might have redundant zeros, $\mathrm{EQ}(x,y) \equiv (x = n0^k$ and $y = n0^l)$. You can see that the relation EQ, definable in the natural presentation of numbers, satisfies the constraints that we put on S_0. Therefore the game quantifier is enough to define all regular relations in the binary presentation of $(\mathbb{N}, =)$. This can as well be used to define $+$ in such presentation so if we add some stronger non-regular relation then model checking becomes undecidable.

Corollary 3. *On the binary presentation of* $(\mathbb{N}, =)$ *the logic FO+∋ can define all regular relations and therefore the binary presentations of* $(\mathbb{N}, =), (\mathbb{N}, s), (\mathbb{N}, <), (\mathbb{N}, +)$ *are complete-automatic for FO+∋.*

Corollary 4. *The logic FO+∋ is undecidable on the binary presentation of Skolem arithmetic* $(\mathbb{N}, \cdot)$.

3.3 Inductive Automorphisms

After analysing what can be expressed in FO+$\eth$ we want to look for methods to establish what relations can *not* be expressed in this logic. For example one could ask if a^ω can be expressed in FO+$\eth$ without any relations other than equality of words. We are going to develop a general method to answer such questions by showing that there is a class of automorphisms of a structure that extend to all relations definable in FO+$\eth$.

First of all please note that not all automorphisms extend to relations definable in FO+$\eth$. For example you can take the bijection of $\Sigma^{\leq\omega}$ that swaps a^ω with b^ω and leaves other elements untouched. The relation $|s \sqcap t| < |u \sqcap w|$ is definable in FO+$\eth$ just with equality, but you can see that this bijection does not extend to an automorphism of the set with this relation as

$$|b^\omega \sqcap ab^\omega| < |a^\omega \sqcap ab^\omega| \text{ but } |a^\omega \sqcap ab^\omega| > |b^\omega \sqcap ab^\omega|.$$

To define the class of *inductive automorphisms* that do extend to relations definable in FO+$\eth$ we are going to restrict the bijections of $\Sigma^{\leq\omega}$ only to a special form.

Definition 1. *The bijection* $\pi : \Sigma^{\leq\omega} \to \Sigma^{\leq\omega}$ *is* inductive *when it does not change the length of the words,* $|\pi(u)| = |u|$ *for every word* u*, and additionally there exists a family of permutations*

$$\{\pi_w\}_{w\in\Sigma^*} \quad \pi_w : \Sigma \to \Sigma,$$

so that for each word u *with at least* n *letters the* n*th letter of* $\pi(u)$ *is given by the appropriate permutation*

$$\pi(u)[n] = \pi_{u|_{n-1}}(u[n]).$$

Please note that the inverse automorphism ϕ^{-1} of any inductive automorphism ϕ is again inductive as inverse permutations $\{\pi_w^{-1}\}$ can be used.

It turns out that if we restrict our attention only to an automorphism ϕ that is an inductive bijection then the structure can be extended with any FO+$\eth$ definable relation and ϕ will still be an automorphism of the extended structure.

Theorem 2. *If* ϕ *is an inductive automorphism of a structure* $(\Sigma^{\leq\omega}, R_1, \ldots, R_k)$ *and* R *is a relation definable by an FO+*$\eth$ *formula,* $R(\overline{x}) \iff \varphi(\overline{x})$ *for some* $\varphi \in$ FO+$\eth$*, then* ϕ *is an automorphism of the extended structure* $(\Sigma^{\leq\omega}, R_1, \ldots, R_k, R)$.

Proof. Clearly when we proceed by induction on the structure of formulas it is enough to consider the inductive step for game quantifier, i.e. to show that if for a formula φ it holds that $\varphi(\overline{x},\overline{y},\overline{z}) \iff \varphi(\overline{\phi}(x),\overline{\phi}(y),\overline{\phi}(y))$ then for $\psi(\overline{z}) = \eth xy\ \varphi(\overline{x},\overline{y},\overline{z})$ it holds $\psi(\overline{z}) \iff \psi(\overline{\phi}(z))$ (the converse follows with the inverse automorphism ϕ^{-1}).

To show the above let us first define for any strategies f of the Verifier and g of the Verifier used in $\eth xy\ \varphi(\overline{x},\overline{y},\overline{z})$ the transposed strategies f_ϕ, g_ϕ in the following way:

$$f_\phi(x,y) = \pi_{\phi^{-1}(x)}h(\phi^{-1}(x),\phi^{-1}(y)),\ g_\phi(x,y) = \pi_{\phi^{-1}(y)}h(\phi^{-1}(x),\phi^{-1}(y)),$$

where π_w is the permutation for word w associated with ϕ. You should observe that when the players play with strategies f_ϕ, g_ϕ then the resulting words are exactly the images of the words that result from using f and g,

$$x_{f_\phi g_\phi} = \phi(x_{fg}),\ y_{f_\phi g_\phi} = \phi(y_{fg}).$$

In this way we can use the winning strategy f for the first player in $\psi(\overline{z})$ and play with f_ϕ in $\psi(\overline{\phi}(z))$. If the opponent chooses to play g then at the end the formula $\varphi(\overline{x}_{f_\phi g}, \overline{y}_{f_\phi g}, \overline{\phi}(z))$ will be evaluated, but

$$\varphi(\overline{x}_{f_\phi g}, \overline{y}_{f_\phi g}, \overline{\phi}(z)) \equiv \varphi(\overline{\phi}(x_{f g_{\phi^{-1}}}), \overline{\phi}(y_{f g_{\phi^{-1}}}), \overline{\phi}(z)) \equiv \varphi(\overline{x}_{f g_{\phi^{-1}}}, \overline{y}_{f g_{\phi^{-1}}}, \overline{z}),$$

which is true as f is winning against any strategy, in particular against $g_{\phi^{-1}}$. ∎

The above theorem gives a general method to investigate definability in FO+∋. For example we can answer the question we stated at the beginning and say that a^ω is not definable in FO+∋ just with equality, because a bijection of $\Sigma^{\leq\omega}$ that swaps the first letter is an inductive bijections and moves a^ω to ba^ω. Together with the fact proved in the previous section that a relation distinguishing successors is enough to define all regular relations in FO+∋ we get a detailed picture of what can and what can not be defined in this logic.

4 Muller Games with Information Levels

To define model checking games that capture first-order and game quantification on automatic structures we need to go beyond two-player perfect information games and use multi-player games with imperfect information. Therefore these games will be played by two coalitions, I and II, each consisting of N players,

$$\Pi = (1,\mathrm{I}), (2,\mathrm{I}), \ldots, (N,\mathrm{I}), (1,\mathrm{II}), (2,\mathrm{II}), \ldots, (N,\mathrm{II}),$$

taking actions described as letters in Σ. The arena of the game is therefore given by the pairwise disjoint sets of positions belonging to each player $V_{1,\mathrm{I}}, \ldots, V_{N,\mathrm{I}}$, $V_{1,\mathrm{II}}, \ldots, V_{N,\mathrm{II}}$ and the function μ defining the moves. Positions of coalition I are denoted by $V_{\mathrm{I}} = V_{1,\mathrm{I}} \cup \ldots \cup V_{N,\mathrm{I}}$ and of coalition II by $V_{\mathrm{II}} = V_{1,\mathrm{II}} \cup \ldots \cup V_{N,\mathrm{II}}$ with all positions $V = V_{\mathrm{I}} \cup V_{\mathrm{II}}$. In any position v the player can choose an action a from Σ and then move to the position $\mu(v,a)$ as $\mu : V \times \Sigma \to V$. The objective of the game is given by a Muller winning condition $\mathcal{F}$.

The *(general) Muller game with information levels* or *hierarchical Muller game* is therefore given by the tuple

$$(V_{1,\mathrm{I}}, \ldots, V_{N,\mathrm{I}}, V_{1,\mathrm{II}}, \ldots, V_{N,\mathrm{II}},\ \mu,\ \mathcal{F} \subseteq \mathcal{P}(V)).$$

In such game *play actions* are the sequence of actions taken by the players during a play, so formally it is an infinite word $\alpha \in \Sigma^\omega$. The play corresponding to play actions α and starting in position v_0 is an infinite sequence of positions resulting from taking the moves as described by α,

$$\pi_\alpha(v_0) = v_0 v_1 \ldots \iff v_i = \mu(v_{i-1}, \alpha[i]),\quad i = 1, 2, \ldots.$$

During the play $\pi_\alpha(v_0)$ we encounter a sequence of players that take the moves and let us denote this sequence by $\Pi_\alpha(v_0) = p_0 p_1 \ldots \Leftrightarrow v_i \in V_{p_i}$.

When we want to play the game each of the $2N$ players has to decide on a strategy $s_p : \Sigma^* \to \Sigma$. In a game with perfect information we would say that play actions α are coherent with the strategy s_p in a play starting in v_0 when for each move i taken by player p, also $v_i \in V_p$, the action taken is given by the strategy acting on the history of actions, $\alpha[i+1] = s_p(\alpha|_i)$.

But since the players do not have perfect information, we assume additionally that for each player p there is a function ν_p that extracts from the history of play actions the information visible for this player. More precisely $\nu_p : (\Sigma \times \Pi)^* \to \Sigma^*$ extracts the information visible to player p from the history of play actions together with players that took the moves. Therefore play actions α in a play starting in v_0 are coherent with s_p when for each i such that $v_i \in V_p$ it holds

$$\alpha[i+1] = s_p(\nu_p((a_1, p_0)(a_2, p_1)\ldots(a_i, p_{i-1}))),$$

where $\alpha = a_1, a_2, \ldots$ and $\Pi_\alpha(v_0) = p_0, p_1, \ldots$.

The above definition of views of play history is very general and we will use only a concrete special case of *hierarchical* view functions. The hierarchical information views are defined so that in each coalition the player i is able to see the moves of players $1, \ldots, i$ in both coalitions, but can not see the moves of players with numbers $j > i$. More formally $\nu_{i,c}((a_1, p_0)(a_2, p_1)\ldots) = a_{i_1}, a_{i_2}, \ldots$ when the indices i_k are precisely those for which $p_{i_k - 1} = (j, d)$ with $j \leq i$.

To define when coalition I wins such a hierarchical game we can not require from coalition I to put forth their winning strategies before II does (as usual in such definitions), because as you saw the player with higher numbers have strictly more information and their advantage would be lost if they disclosed their strategies too early. Therefore we use the following definition that requires that strategies are given stepwise, level by level going through the levels of information visibility.

Definition 2. *Coalition* I *wins the hierarchical game*

$$(V_{1,\mathrm{I}}, \ldots, V_{N,\mathrm{I}}, V_{1,\mathrm{II}}, \ldots, V_{N,\mathrm{II}},\ \mu,\ \mathcal{F})$$

starting from position v_0 when the following condition holds. There exists a strategy $s_{1,\mathrm{I}}$ for player $1, \mathrm{I}$ such that for each strategy $s_{1,\mathrm{II}}$ of player $1, \mathrm{II}$ there exists a strategy $s_{2,\mathrm{I}}$ such that for each strategy $s_{2,\mathrm{II}}$ ... there exists a strategy $s_{N,\mathrm{I}}$ such that for each strategy $s_{N,\mathrm{II}}$ the play actions sequence α, starting from v_0 and coherent with all strategies $s_{1,\mathrm{I}}, s_{1,\mathrm{II}}, \ldots, s_{N,\mathrm{I}}, s_{N,\mathrm{II}}$, results in a play $\pi_\alpha(v_0)$ winning for I, *i.e. such that* $\mathrm{Inf}(\pi_\alpha(v_0)) \in \mathcal{F}$.

As you can expect, the definition for coalition II is dual, i.e. it says that there exists a $s_{1,\mathrm{II}}$ so that for all $s_{1,\mathrm{I}}$, ..., the play is not winning, $\mathrm{Inf}(\pi_\alpha(v_0)) \notin \mathcal{F}$.

4.1 Example of a Hierarchical Game

To get some intuition on what kind of behaviour can be captured with hierarchical games let us consider the simple example depicted on Figure 1.

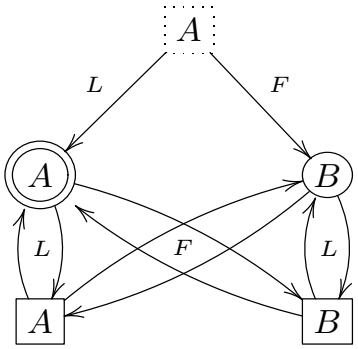

Fig. 1. Example of a Hierarchical Muller Game.

You can think that this game is played using a coin with two sides, A and B, and each of the players can choose to flip the coin (F) or leave it as it is (L). Formally in this game we have 4 players, two in each coalition, the top position belongs to $2, \mathrm{II}$ and the two bottom positions belong to $1, \mathrm{II}$. The game proceeds as follows: first the second player of the coalition II chooses either to flip the coin or to leave it as it is. Then only the other two players play by either flipping the coin or leaving it intact. We will say that coalition I wins if the A side is seen infinitely often in positions where I move in the game, as marked on Figure 1.

To illustrate the importance of the hierarchical information levels let us consider two variants of this game. In the first one, the bottom strongly connected component belongs to players on the same information levels, i.e. to $1, \mathrm{II}$ and $1, \mathrm{I}$. You can see that in this scenario coalition II can win, because first the player $2, \mathrm{II}$ can flip the coin to B and latter the player $1, \mathrm{II}$ can play by always repeating the last move of player $1, \mathrm{I}$.

In the other variant let the player for I have more information, i.e. let the bottom strongly connected component belong to $1, \mathrm{II}$ and $2, \mathrm{I}$, with $V_{1,\mathrm{I}} = \emptyset$. Now coalition I can win because the strategy of player $2, \mathrm{I}$ is given after the strategy of $1, \mathrm{II}$ is set. Therefore I can assure that the coin will be flipped after each two moves infinitely often, which guarantees that I holds the coin on A side infinitely often independent of the first move of $2, \mathrm{II}$.

4.2 Alternating Moves in Hierachical Games

In general, determining the winner of hierarchical games is undecidable, what can be proved by reducing the Post correspondence problem. Let us state it as a theorem (proved in Appendix A).

Theorem 3. *The question whether coalition* I *wins in a general Muller game with information levels is undecidable.*

Moreover, let us show on Figure 2(a) a simple example of a hierarchical game that is not determined. The positions of coalition I are round, the positions of

coalition II are square and there are two levels of information, the positions on the upper level are dotted. The component W is winning for coalition I and L is losing. When you analyse this game you will see that on the lower information level the player has to predict the move of the opponent to win, i.e. his strategy has to start with an a exactly if the opponent starts with an a. This of course leads to a non-determined game as each player can counter the strategy of the opponent once it is known.

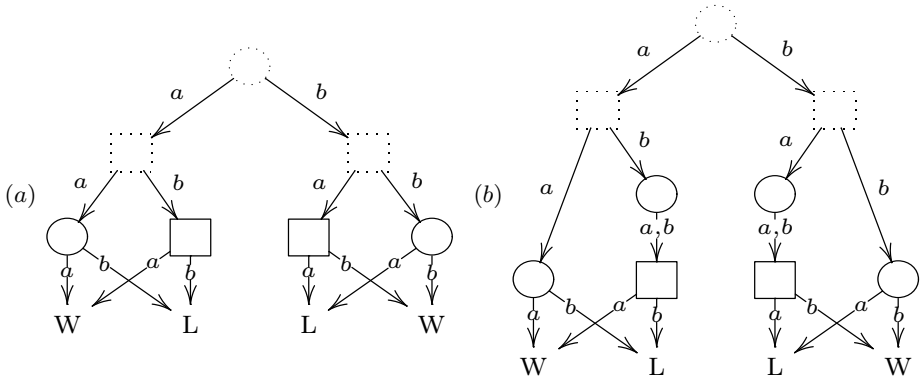

Fig. 2. (a) Non-determined General Hierarchical Game, (b) Similar Determined Game.

Both non-determinsm and undecidability can be countered by restricting to games where players alternate their moves. More precisely let an *alternating game with information levels* be such a game, where for each letter $a \in \Sigma$ and each level $i = 1, \ldots, N$ the following alternation conditions hold:

$$v_i \in V_{i,\mathrm{I}} \implies \mu(v_i, a) \in V_{i,\mathrm{II}},\ v_i \in V_{i,\mathrm{II}} \implies \mu(v_i, a) \in V_{(i \bmod N)+1,\mathrm{I}}.$$

To see that alternation of moves really helps you can look at Figure 2(b) where two positions were added that may seem useless, but that make the game determined. To convince yourself that in the extended game coalition II can indeed win, try to take the following strategy of player $1, \mathrm{II}$. Let him play always the opposite move to the one that was taken before by player $1, \mathrm{I}$. For player $2, \mathrm{II}$ take the following strategy — if player $1, \mathrm{I}$ declared that he will play a first then play b and else play a first. You can check that these strategies are indeed winning for II, but this is possible only because constructing the strategy for $1, \mathrm{II}$ we knew the first letter played by $1, \mathrm{I}$.

5 Model Checking with Hierarchical Games

To connect the logic FO+∋ to the games with information levels let us restrict our attention only to such games where players alternate their moves in order of information visibility, as defined above.

In an alternating game every infinite play actions sequence can be divided into sequences of $2N$ actions, each taken by a different player,

$$\alpha = a_1^{1,\mathrm{I}} a_1^{1,\mathrm{II}} a_1^{2,\mathrm{I}} a_1^{2,\mathrm{II}} \dots a_1^{N,\mathrm{I}} a_1^{N,\mathrm{II}} a_2^{1,\mathrm{I}} \dots a_2^{N,\mathrm{II}} a_3^{1,\mathrm{I}} \dots .$$

Let the $2N$-*split* of these play actions be the tuple of words played by each of the players,

$$\mathrm{split}_{2N}(\alpha) = (a_1^{1,\mathrm{I}} a_2^{1,\mathrm{I}} \dots,\ \{a_i^{1,\mathrm{II}}\},\ \dots,\ \{a_i^{N,\mathrm{I}}\},\ \{a_i^{N,\mathrm{II}}\}).$$

You should note that, since the set of plays starting from a fixed v_0 that are winning for I is ω-regular, then also the set of corresponding $2N$-splits of play actions is ω-regular. This can be seen by taking only each $2N$th state of the Muller automaton recognising the plays and making a product with Σ^{2N} to store the states that were omitted from the path in the original automaton. For an alternating hierarchical Muller game G let us denote the $2N$ary relation recognising the $2N$-split of plays winning for I by

$$W_{\mathrm{I}}^{G,v_0}(a_1, \dots, a_{2N}) \Leftrightarrow \forall \alpha\ (\mathrm{split}_{2N}(\alpha) = (a_1, \dots, a_{2N}) \Rightarrow \mathrm{Inf}(\pi_\alpha(v_0)) \in \mathcal{F}^G).$$

The definition for coalition II is analogous, just with $\mathrm{Inf}(\pi_\alpha(v_0)) \notin \mathcal{F}^G$.

You can now note that the condition that coalition I (resp. II) wins in an alternating hierarchical Muller game can be expressed in FO+∋ using the relation W_{I}^{G,v_0}, which results in the following theorem, proved in Appendix B.

Theorem 4. *For any alternating game with information levels G and the relations W_{I}^{G,v_0} and W_{II}^{G,v_0} defined as above, coalition* I *(resp.* II*) wins the game G starting from v_0 exactly if the following formula holds in $(\Sigma^\omega, W_{\mathrm{I}}^{G,v_0})$:*

$$\ni x_1 y_1 \dots \ni x_N y_N\ W_{\mathrm{I}}^{G,v_0}(x_1, y_1, \dots, x_N, y_N).$$

After we captured winning in alternating games in FO+∋ let us do the converse and construct a model checking game for a given FO+∋ formula on an automatic structure. At first we will restrict ourself to formulas in the form

$$\varphi = \ni x_1 y_1 \ni x_2 y_2 \dots \ni x_N y_N\ R(x_1, y_1, \dots, x_N, y_N)$$

and just construct a game so that the *split* of the winning plays will allow us to use the previous theorem.

The construction can be understood intuitively as prefixing each variable with all possible letters in the order of information hierarchy and making a step of the automaton when all variables are prefixed. To define these games precisely let us take the automaton for R, namely $\mathcal{A}_R = (Q, q_0, \delta, \mathcal{F}_R)$, and construct the

model checking game G_φ for φ in the following way. For each even tuple of letters $c_1, d_1, c_2, d_2, \ldots, c_M, d_M$, with $0 \leq M < N$, and for every state $q \in Q$, we will have in our game the position

$$R^q(c_1x_1, d_1y_1, \ldots, c_Mx_M, d_My_M, x_{M+1}, \ldots, y_N), \tag{1}$$

and for each uneven tuple $c_1, d_1, c_2, d_2, \ldots, c_M, d_M, c_{M+1}$, $0 \leq M < N$, the position

$$R^q(c_1x_1, \ldots, d_My_M, c_{M+1}x_{M+1}, y_{M+1}, \ldots, y_N). \tag{2}$$

In each of these positions the next move is made by the player corresponding to the next variable that is not yet prefixed by a letter, e.g. in position 1 it is the player $M+1$ of coalition I who makes the move for x_{M+1} and in position 2 it is the player $M+1$ of coalition II. We can now formally define the set of positions for players on each level i as $V_{i,\mathrm{I}} = Q \times \Sigma^{2(i-1)}$, $V_{i,\mathrm{II}} = Q \times \Sigma^{2i-1}$.

The moves in G_φ are defined in an intuitive way — the player chooses a letter to prefix his variable with, so for $0 \leq M < N$

$$\mu(R^q(c_1x_1, \ldots, d_My_M, x_{M+1}, \ldots, y_N), c_{M+1}) =$$

$$= R^q(c_1x_1, \ldots, d_My_M, c_{M+1}x_{M+1}, y_{M+1}, \ldots, y_N),$$

and for $0 \leq M < N-1$

$$\mu(R^q(c_1x_1, \ldots, c_{M+1}x_{M+1}, y_{M+1}, \ldots, y_N), d_{M+1}) =$$

$$= R^q(c_1x_1, \ldots, c_{M+1}x_{M+1}, d_{M+1}y_{M+1}, x_{M+2}, \ldots, y_N).$$

The only special case is the final position $R^q(c_1x_1, d_1y_1, \ldots, c_Nx_N, y_N)$. When the player N, II chooses the final letter d_N then it will not be appended, but instead all the prefixing letters will be removed and the state of the automaton will be changed (here $\overline{\alpha} = c_1d_1 \ldots c_Nd_N$):

$$\mu(R^q(c_1x_1, d_1y_1, \ldots, c_Nx_N, y_N), d_N) = R^{\delta(q,\overline{\alpha})}(x_1, y_1, \ldots, x_N, y_N).$$

The winning condition $\mathcal{F}$ in the game is defined to correspond to the acceptance condition $\mathcal{F}_R$ of the automaton for R in such way, that we look only at the state component of each position.

To see that the game G_φ is indeed the model checking game for φ we can use Theorem 4 again, just observe that the $2N$-split of the winning paths in G_φ is exactly the relation R, $W_{\mathrm{I}}^{G_\varphi, R^{q_0}(x_1,y_1,\ldots,x_N,y_N)} = R$.

In this way we know how to construct the model checking game for formulas in simple form. As we have seen, any formula in FO+$\Im$ can be written in negation normal form and additionally, by renaming variables, it can be reduced to prenex normal form. Let us therefore consider now a general formula in the form $\varphi = \Im x_1y_1\Im x_2y_2 \ldots \Im x_Ny_N\, \psi(x_1, y_1, \ldots, x_N, y_N)$, where ψ is in negation normal form and does not contain quantifiers. Let us construct the game G_φ inductively with respect to ψ.

In the case of $\psi(\overline{x}) = R(\overline{x})$ or $\psi(\overline{x}) = \neg R(\overline{x})$ the solution was already presented, when considering $\neg R$ we just have to complement the acceptance condition of the automaton for R. Let us show how to construct the game for boolean connectives, i.e. for $\psi_1 \wedge \psi_2$ and for $\psi_1 \vee \psi_2$. We want to adhere to the usual convention of model checking games and to have only one additional position for any junctor. The game for $\psi_1 \circ \psi_2$, where $\circ = \wedge, \vee$, is therefore constructed as follows: we take the two games for ψ_1 and ψ_2 and we add one more position on higher level of information that has two possible moves — to the starting position of ψ_1 and to the starting position of ψ_2. The new position belongs to coalition I when $\circ = \vee$ and to coalition II when $\circ = \wedge$ and in both cases the other coalition does not play on that information level. With the construction described above we face a problem, as the game is not strictly alternating any more, but it is not a significant obstacle. An example of a model checking game and the way to overcome this technical problem can be seen in Appendix C.

To formally prove that the resulting games are indeed model checking games for formulas with boolean connectives you can just replace the connectives with a new variable and the formula with one relation where only the first letter of connective-variables is considered. Then the automata for such relation corresponds to the defined game and Theorem 4 can be used again.

The exact correspondence of alternating hierarchical games and FO+∂ makes it possible to use our knowledge about this logic. In particular we can transfer the results about complexity including the non-elementary lower bound on deciding first-order logic on automatic structures.

Corollary 5. *The question whether coalition* I *(resp.* II*) wins in an alternating Muller game with information levels on a finite arena is decidable, non-elementary when the number of levels is not fixed and it can be done in* 2K-EXPTIME *for K information levels.*

The possibility to get negation normal form for FO+∂ can as well be translated and gives the proof of determinacy of alternating hierarchical games.

Corollary 6. *Alternating Muller games with information levels are determined.*

6 Conclusions and Future Work

We described how game quantification can be used on automatic structures and the resulting logic turned out to be very interesting. It is decidable and the defined relations remain regular, which might be used in the study of presentations of automatic structures. On the other hand the logic is strictly more expressive than first-order on some weaker structures. Most notably on the binary tree and on presentations of natural numbers it is possible to define all regular relations when game quantification is allowed. The methods that we used, for example inductive automorphisms, might be extended to morphisms between presentations of the same automatic structure and used to study intrinsic regularity.

On the other hand, it might be interesting to ask what is the expressive power of FO+∂ on formulas with just one game quantifier, i.e. $\partial \overline{xy}\ \varphi(\overline{x}, \overline{y})$ where φ

is quantifier-free. Such formulas may be more expressive than just existential or universal fragment of first-order logic even on complete-automatic structures and can be decided with double exponential complexity.

Game quantification made it possible to define an expressive class of model checking games that we used for checking first-order logic on automatic structures. These games use multiple players and imperfect information in a novel way and might be used to derive more efficient algorithms for verification, especially if the efficient algorithms from [5] can be generalised to hierarchical games.

References

1. V. Barany, *Invariants of Automatic Presentations and Semi-Synchronous transductions*, Proceedings of STACS 06, vol. 3884 of Lecture Notes in Computer Science, pp. 289-300, 2006.
2. J. C. Bradfield, *Parity of Imperfection* or *Fixing Independence*, vol. 2803 of Lecture Notes in Computer Science, pp. 72-85, 2003.
3. J. R. Büchi, *On Decision Method in Restricted Second Order Arithmetic*, Proceedings of the International Congress on Logic, Methodology and Philosophy of Science, pp. 1-11, 1962.
4. A. Blumensath, E. Grädel, *Finite Presentations of Infinite Structures: Automata and Interpretations*, vol. 37 of Theory of Computing Systems, pp. 641-674, 2004.
5. K. Chatterjee, T. A. Henzinger, *Semiperfect-Information Games*, Proceedings of FSTTCS 2005, vol. 3821 of Lecture Notes in Computer Science, pp. 1-18, 2005.
6. A. Emerson, C. Jutla, *Tree automata, mu-calculus and determinacy*, in Proc. 32nd IEEE Symp. on Foundations of Computer Science, pp. 368-377, 1991.
7. D. Gale, F.M. Stewart. *Infinite Games with Perfect Information*, in *H. W. Kuhn, A. W. Tucker* eds. Contributions to the Theory of Games, Volume II, pp. 245-266, Annals of Mathematics Studies, Princeton University Press, 1953.
8. E. Grädel, *Model Checking Games*, Proceedings of WOLLIC 02, vol. 67 of Electronic Notes in Theoretical Computer Science, 2002.
9. B. Khoussainov, A. Nerode, *Automatic Presentations of Structures*, Proceedings of LCC 94, vol. 960 of Lecture Notes in Computer Science, pp. 367-392, 1995.
10. Ph. G. Kolaitis *Game Quantification*, in *J. Barwise, S. Feferman* eds. Model Theoretic Logics, pp. 365-422, 1985
11. D. Martin, *Borel determinacy*, Annals of Mathematics (102), pp. 336-371, 1975.
12. S. Miyano, T. Hayashi, *Alternating Finite Automata on ω-words*, vol. 32 of Theoretical Computer Science, pp. 321-330, 1984.

A Proof of Undecidability of Hierarchical Muller Games.

Theorem 5. *The question whether coalition* I *wins in a general Muller game with information levels is undecidable.*

Proof. Let us reduce the Post correspondence problem for $\overline{u} = u_1, \ldots, u_K$ and $\overline{v} = v_1, \ldots, v_K$, where $u_i, v_i \in \{a, b\}^*$, to the problem if I wins in the hierarchical game $G_{\overline{u},\overline{v}}$. The possible actions in $G_{\overline{u},\overline{v}}$ are $\Sigma = \{a, b, \Box, 1, 2, \ldots, K\}$ and they will roughly correspond to the players choosing letters in words u_i, v_i, special delimiter $\Box$, and choosing which word to play next.

In constructing $G_{\overline{u},\overline{v}}$ we are going to use sub-games, such that for a given word u the sub-game enforces that u is played, else the player that moves loses. It is easy to construct such sub-game, it has one more position than u has letters and if the wrong letter is chosen then it leads to a position where the player looses. There is one outgoing edge in the sub-game and it is the one taken when the last letter u is played. On Figure 3(a) there is an example sub-game for $u = aba$.

Constructing the game $G_{\overline{u},\overline{v}}$ we start with a position belonging to player $3, \mathrm{II}$ with two possible (non-losing) moves. In this position the coalition II can decide if the test will be done for the words $\overline{u}$ or for the words $\overline{v}$. All other positions will be on lower level of information, so that coalition I will never be able to deduce in what component the play is taking place.

Each of the two components for $\overline{u}$ and for $\overline{v}$ starts with a position of player $2, \mathrm{I}$ where this player chooses if he wants to play some word $1, \ldots, K$ or the special symbol $\Box$. If the special symbol is chosen, the player $1, \mathrm{I}$ must play the same symbol and return back to the position, where $2, \mathrm{II}$ chooses a word. When a word number L is chosen, then in each of the components first the word v_L and then u_L is played. But in the $\overline{u}$ component, it is the player $2, \mathrm{II}$ that must play v_L and $1, \mathrm{I}$ must play u_L, and in the component for $\overline{v}$ it is the other way round, $1, \mathrm{I}$ must play v_L. Later the game returns to the position where $2, \mathrm{I}$ chooses a word. The complete game, using the sub-games for u_i and v_i, is depicted on Figure 3(b). The winning condition is as follows: the special symbol $\Box$ must be chosen by $2, \mathrm{I}$ infinitely often, and there must also be another $L \neq \Box$ that is played infinitely often.

Let us first look at the game and show that if there is a solution for the Post correspondence problem then I has winning strategies for $G_{\overline{u},\overline{v}}$. Indeed, let $i_1, i_2, \ldots, i_M$ be the indices for the solution of the correspondence problem, so $u_{i_1} u_{i_2} \ldots u_{i_M} = v_{i_1} v_{i_2} \ldots v_{i_M}$. Let the player $2, \mathrm{I}$ play so that he chooses first i_1, then i_2, and so on up to i_M, then the special symbol $\Box$ and then again i_1 and so on. The player $1, \mathrm{I}$ is going to play the letters from the word $u_{i_1} u_{i_2} \ldots u_{i_M}$ in turn and then $\Box$ and then again the letters $u_{i_1} u_{i_2} \ldots u_{i_M}$ and $\Box$ and so on. Please note that it does not matter in which component the play is taking place, the player $1, \mathrm{I}$ will never play the wrong letter and the player $2, \mathrm{I}$ will choose $\Box$ and non-$\Box$ infinitely often, so coalition I is going to win.

To prove the converse, namely that if there is a winning strategy for I then the correspondence problem has a solution, you have to observe two facts. First

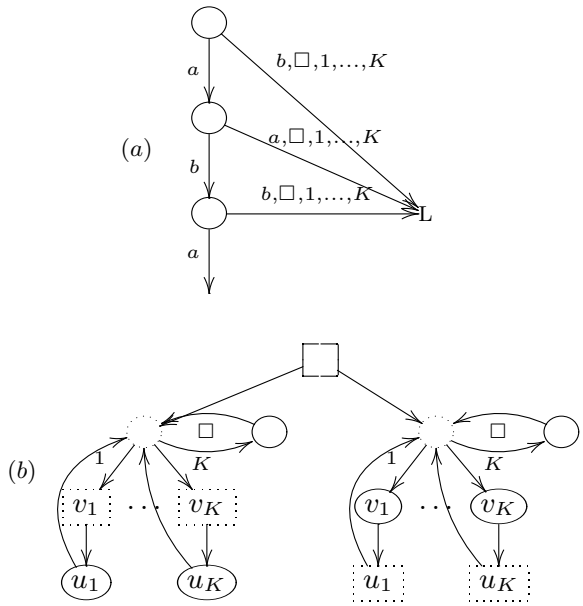

Fig. 3. (a)Example sub-game for $u = aba$, (b) Complete Game $G_{\overline{u},\overline{v}}$.

of all, 2, I can never deduce in which component the play is taking place, because what he can see after each of his moves is the same in both components. Second, □ must be played and can played by 2, I only if up to that point the words played would have the same length in both components. This is because playing □ makes I loose everywhere except for the special position, □ can not be played in a sub-game for any word.

Formally, let us first fix the only sensible strategy for 2, II, namely that when the last number played is L and it is the recently played action, then play v_L, and if there were other actions $\{a, b\}^*$ taken after the last L then play u_L. Please note that the player 2, II in fact knows in which component the play takes place, even if the move of 3, II is not visible for him. With this strategy fixed, the condition that coalition I has a winning strategy for $G_{\overline{u},\overline{v}}$ means that there exists a strategy σ_1 for player 1, I and a strategy σ_2 for player 2, I such that the play corresponding to these two strategies and the fixed one for 2, II is in the winning condition independent of the component chosen by 3, II.

Let us first concentrate on the strategy σ_2. Since, according to the winning condition, □ can not be played infinitely often and in each component the only possible answer to □ is again □, let us assume without loss of generality that the first move taken by σ_2 is not □ and let it be L_1. The play after L_1 goes through v_{L_1} and u_{L_1} and let us denote by L_2 the next move of 2, I, i.e.

$$L_1 = \sigma_2(\varepsilon),\ L_2 = \sigma_2(L_1 v_{L_1} u_{L_1}).$$

Let us continue the play and denote the next moves of I by $L_2, \ldots, L_M$, up to the point where he plays a $\Box$, more formally

$$L_{i+1} = \sigma_2(L_1 v_{L_1} u_{L_1} L_2 \ldots L_i v_{L_i} u_{L_i}),\ L_{M+1} = \Box.$$

Since we are able to extract the sequence $L_1, \ldots, L_M$ of moves of $2, \mathrm{I}$ from his winning strategy, let us now look at the player $1, \mathrm{I}$. This is the only player on information level 1, so he sees only his own previous moves and instead of the strategy σ_1 we can say that he plays a word $t \in \{a, b, \Box\}^\omega$ such that

$$t[i] = \sigma_1(t|_{i-1}) \text{ for } i = 0, 1, \ldots.$$

Due to the structure of the game, no $\Box$ can be played by $1, \mathrm{I}$ before $2, \mathrm{I}$ decides to play $\Box$ and then $\Box$ must be played. Therefore if w is the prefix of t up to the first $\Box$, then we know that w is exactly the word played by $1, \mathrm{I}$ while $2, \mathrm{I}$ played the moves $L_1, \ldots, L_M$. But due to the structure of the game, coalition II can decide if $w = u_{L_1} \ldots u_{L_M}$ or if $w = v_{L_1} \ldots v_{L_M}$ and w has to be good for both cases, it must therefore be the solution for the Post correspondence problem. ∎

B Expressing Alternating Hierarchical Games in FO+∋

Theorem. *For any alternating game with information levels*

$$G = (V_{1,\mathrm{I}}, \ldots, V_{N,\mathrm{I}}, V_{1,\mathrm{II}}, \ldots, V_{N,\mathrm{II}}, \mu, \mathcal{F})$$

and position $v_0 \in V_{1,\mathrm{I}}$, coalition I *wins G starting from v_0 when*

$$\varphi_{\mathrm{I}} = \ni x_1 y_1 \ldots \ni x_N y_N\ W_{\mathrm{I}}^{G,v_0}(x_1, y_1, \ldots, x_N, y_N)$$

holds, and coalition II *wins G starting from v_0 when*

$$\varphi_{\mathrm{II}} = \ni^\forall y_1 x_1 \ldots \ni^\forall y_N x_N\ W_{\mathrm{II}}^{G,v_0}(x_1, y_1, \ldots, x_N, y_N)$$

holds, where the relations W_{I}^{G,v_0} and W_{II}^{G,v_0} are defined as

$$W_{\mathrm{I}}^{G,v_0}(a_1, \ldots, a_{2N}) \Leftrightarrow \forall \alpha\ (\mathrm{split}_{2N}(\alpha) = (a_1, \ldots, a_{2N}) \Rightarrow \mathrm{Inf}(\pi_\alpha(v_0)) \in \mathcal{F}^G),$$

$$W_{\mathrm{II}}^{G,v_0}(a_1, \ldots, a_{2N}) \Leftrightarrow \forall \alpha\ (\mathrm{split}_{2N}(\alpha) = (a_1, \ldots, a_{2N}) \Rightarrow \mathrm{Inf}(\pi_\alpha(v_0)) \notin \mathcal{F}^G).$$

Proof. When you read the definition of when a coalition i wins a game with information levels you can see, that it has similar semantic structure to the formula φ_i. To fix the notation, let us read the definition, which says that there is a strategy σ_1 for player on level 1 of coalition i so that for any counter-strategy ρ_1 there exists a strategy σ_2 and so on up to σ_N so that for all ρ_N the resulting play must be good for coalition i. On the other hand, the formula φ_i, according to the definition of $\ni$, says that there is a function f_1 so that for all functions

g_1 there is a function f_2 and so on up to a function f_N so that for all g_N if you construct the words according to $\overline{f}$ and $\overline{g}$, then they form a $2N$-split of a play that is good for coalition i.

As the final condition in both cases above is equivalent, due to the definition of W_i^{G,v_0}, the only problem is to show how to relate the functions f_i and the strategies σ_i. Here the alternation of the game plays a crucial role, as we are going to rely on the fact, that it is always known which player takes actions where. Therefore, when a function f is given we can construct a strategy σ_f by looking only at the moves taken by by the players on the current level of information and applying f to them. Conversely, when a strategy σ on level L is given, you can compute the function f_σ by collecting every action the strategy σ has taken as the first argument and then collecting every action the opponent on level L has taken as the second argument. Even though we are not explicitly writing it here, you should be aware that since in an alternating game coalition I makes moves before coalition II, then in case if $i = \mathrm{I}$ the first argument will be as long as the second, whereas in case of $i = \mathrm{II}$ it will be longer. That corresponds exactly to the use of $\partial^\forall$ quantifiers in the formula for coalition II.

You can see that the correspondence between functions in the formula and strategies in the game that we fixed is injective, i.e. if $f \neq g$ then $\sigma_f \neq \sigma_g$. Therefore if coalition i wins the game G then indeed the formula φ_i must be true with $f_k = f_{\sigma_k}$, because else you could take the functions g_l and construct the counter-strategies $\rho_l = \sigma_{g_l}$.

On the other hand, even if the correspondence is not formally bijective, you should note that once all functions f_i, g_i with $i < k$ are given then during the construction of the strategy σ_{f_k} for function f_k the complete history is known. Therefore if we had some counter-strategy ρ_l then the function $g_l = f_{\rho_l}$ would also disprove the formula φ_i. ∎

C Example of Model Checking Game

To illustrate the construction of model checking games let us look at a simple formula $\exists x\ (R_1(x) \wedge R_2(x))$ with $R_1 = \{a^\omega\}$ and $R_2 = \{a, b\}^\omega \backslash \{a^\omega\}$. You can see that both the automaton for R_1 and the one for R_2 has two states and the transition function is identical too — on any b the automata go from q_0 to q_1 and stay there infinitely. Just the acceptance conditions differ, with $\mathcal{F}_1 = \{\{q_0\}\}$ and $\mathcal{F}_2 = \{\{q_1\}\}$.

On Figure 4(a) you can see the game for this formula, with dumb moves for the second player, as $\exists x\ \varphi(x) \equiv \partial xy\ \varphi(x)$. You should remember that this is actually a 4-player game and the top position belongs to player $2, \mathrm{II}$. Since the formula is false, coalition II wins this game, because for I to win the player $1, \mathrm{I}$ would have to present a strategy to visit both of the double-circled vertices infinitely often without knowing in which branch he is, and that is impossible.

To see that the broken alternation is not a problem, let us formally say that the game for $\psi_1 \circ \psi_2$ is not like the one depicted on Figure 4(a), but rather like the one depicted on Figure 4(b), where dumb moves are added to make the

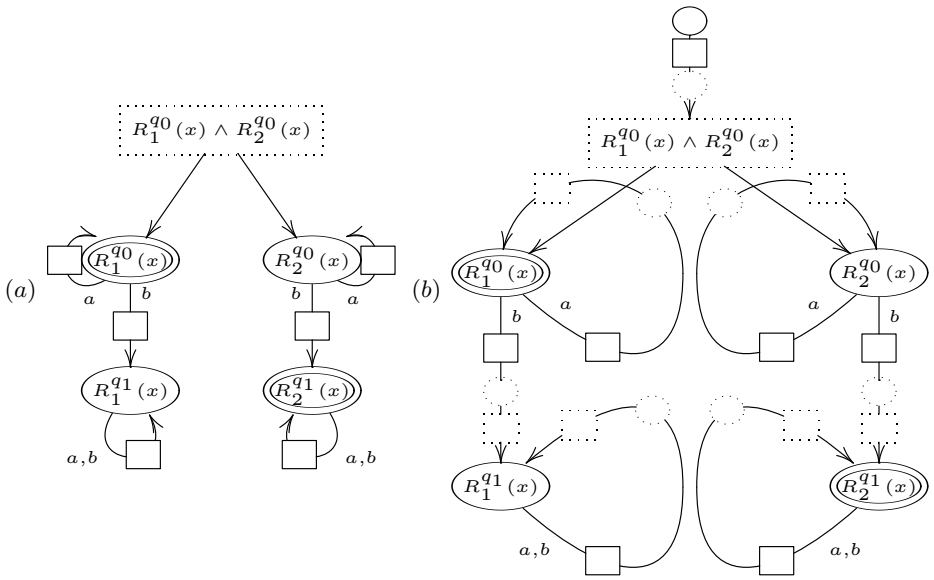

Fig. 4. (a) Game for $\exists x\ (R_1(x) \wedge R_2(x))$, (b) Equivalent Alternating Game.

game alternating. It is clear that winning strategies in these two games can be transferred as in each move on each level of visibility the players know how many moves on the other levels were made.