Abstract
The finite satisfiability problem for guarded fixpoint logic is decidable.

Keywords: guarded fragment, guarded fixpoint logic, finite satisfiability

1. Introduction
The Guarded Fragment is a robustly decidable syntactic fragment of first-order logic, which has received much attention since its conception thirteen years ago [1]. It has also seen a number of variants and extensions being adopted in diverse fields of computer science. One of the most powerful extensions to date, Guarded Fixpoint Logic, was introduced by Grädel and Walukiewicz in [6], where it was shown that the satisfiability problem of Guarded Fixpoint Logic is decidable and computationally no more complex than for the Guarded Fragment. Grädel and Walukiewicz also observed that, unlike the Guarded Fragment, but like the modal $\mu$-calculus with backward modalities, which it extends, Guarded Fixpoint Logic does not have the finite model property. Therefore, there is a finite satisfiability decision problem: decide if a formula has a finite model. Grädel and Walukiewicz left the decidability of this decision problem open. Here we claim this inheritance.

Main Theorem 1. It is decidable whether or not a given guarded fixpoint sentence $\varphi$ is finitely satisfiable. The problem is $2\text{ExpTime}$-complete, but $\text{ExpTime}$-complete for formulas of bounded width.

The proof combines three ingredients:

Email address: {vbarany,bojan}@mimuw.edu.pl (Vince Bárány, Mikołaj Bojańczyk)
The connection between guarded fixpoint logic and alternating automata [6].

Decidability of finite emptiness of alternating automata on undirected graphs [3].

A recent development in the finite model theory of guarded logics [2].

In what follows, no intricate knowledge of either [3] or [2] is required, the results of these papers are used simply as black boxes. We merely combine them in a straightforward manner: [3] provides the algorithm and the construction of [2] proves its correctness.

Here is the plan of the paper. In Section 2 we define guarded fixpoint logic. In Section 3, we define alternating automata on undirected graphs, and state the result from [3]. In Section 4 we show the relation between guarded fixpoint logic and alternating automata, following [6]. In the last section, we present the algorithm, and prove its correctness using [2].

2. Guarded Fixpoint Logic

The guarded fragment of first-order logic comprises only formulas with a restricted pattern of “guarded quantification” and otherwise inherits the semantics of first-order logic. Guarded quantification takes the form $\exists \bar{y}R(\bar{x}\bar{y}) \land \varphi(\bar{x}\bar{y})$ or, dually, $\forall \bar{y}R(\bar{x}\bar{y}) \rightarrow \varphi(\bar{x}\bar{y})$ where $R(\bar{x}\bar{y})$ is a positive literal acting as a guard by effectively restricting the variables $\bar{x}$ to range only over those tuples occurring in the appropriate positions in the atomic relation $R$. Here it is meant that $\bar{x}\bar{y}$ include all free variables of $\varphi$. Guarded quantification can be understood as a generalisation of polyadic modalities of modal logic. Indeed, the guarded fragment was conceived precisely with this analogy in mind [1], therefore it is no coincidence that the model theory of the guarded fragment bears such a strong resemblance to that of modal logic [7].

Guarded Fixpoint logic is obtained by extending the guarded fragment of first-order logic with least- and greatest fixpoint constructs. Its syntax can be defined by the following scheme

\[
\varphi ::= R(\bar{x}) \mid \varphi' \land \varphi'' \mid \neg \varphi' \mid \exists \bar{y} \left( R(\bar{x}\bar{y}) \land \varphi(\bar{x}\bar{y}) \right) \mid Z(\bar{z}) \mid \left[ \text{LFP} \ Z, \bar{z} . \varphi'''(Z, \bar{z}) \right](\bar{x}) \mid \left[ \text{GFP} \ Z, \bar{z} . \varphi'''(Z, \bar{z}) \right](\bar{x})
\]

where $R$ is an arbitrary atomic relation symbol, $Z$ a fixpoint variable, where all free variables of subformulas $\varphi', \varphi'', \varphi'''$ are among those indicated, and
\( \varphi''' \) is required to be monotonic in \( Z \). The semantics is straightforward and standard: the least (or greatest) fixpoint of a formula \( \varphi'''(Z, \vec{z}) \) on a given structure is the wrt. set inclusion least (resp. greatest) relation \( S \) satisfying \( S(\vec{a}) \leftrightarrow \varphi'''(S, \vec{a}) \) for all \( \vec{a} \) on the structure. Crucially, fixpoint variables and fixpoint formulas are not allowed to stand as guard in a guarded quantification, only atomic relation symbols may act as guards. Furthermore, within sentences it can be assumed wlog. that in the matrix \( \varphi'''(Z, \vec{z}) \) of a fixpoint formula the tuple of free variables \( \vec{z} \) is explicitly guarded [6].

Guarded fixpoint logic naturally extends the modal \( \mu \)-calculus with backward modalities. As such it can axiomatise (the necessarily infinite) well-founded DAGs having no sink nodes via the following canonical example.

\[
\exists xy E(x, y) \land \forall xy \left( E(x, y) \rightarrow [\text{LFP } Z, z . \forall v E(v, z) \rightarrow Z(v)](x) \land \exists w E(y, w) \right)
\]

Guarded logics possess a very appealing model theory in which guarded bisimulation plays a similarly central role as does bisimulation for modal logics. Consider two relational structures \( \mathfrak{A}_0 \) and \( \mathfrak{A}_1 \) over the same relational signature. Consider a family \( Z \) of partial isomorphisms of the form

\[
a : A_0 \rightarrow A_1
\]

where \( A_0 \) is a guarded subset of \( \mathfrak{A}_0 \), and consequently, also \( A_1 \) is a guarded subset of \( \mathfrak{A}_1 \). The family \( Z \) is called a guarded bisimulation if for every \( a : A_0 \rightarrow A_1 \) in the family \( Z \), every \( i \in \{0, 1\} \) and every guarded subset \( B_i \) of \( \mathfrak{A}_i \), there is a partial isomorphism

\[
c : C_0 \rightarrow C_1
\]

in the family \( Z \) such that \( B_i \subseteq C_i \) and \( a \) agrees with \( c \).

Guarded fixpoint formulas are invariant under guarded bisimulation. More formally, suppose that \( \varphi \) is a formula of guarded fixpoint logic and \( Z \) is a guarded bisimulation of structures \( \mathfrak{A}_0 \) and \( \mathfrak{A}_1 \). Suppose that \( \eta \) is a valuation of free variables of \( \varphi \) to elements of \( \mathfrak{A}_0 \), and \( a \in Z \) is a partial isomorphism whose domain contains the range of \( \eta \). Then \( \varphi \) holds in \( \mathfrak{A}_0 \) under the valuation \( \eta \) if and only if \( \varphi \) holds in \( \mathfrak{A}_1 \) under the valuation \( a \circ \eta \). In particular, guarded bisimilar structures satisfy the same guarded fixpoint sentences.

The guarded fragment has been characterised as the guarded-bisimulation-invariant fragment of first-order logic, most recently even in the context of finite structures [8]. Similarly, guarded fixpoint logic is characterised as the guarded-bisimulation-invariant fragment of guarded second-order logic [7].
3. Alternating two-way automata

In this section, we introduce alternating automata on undirected graphs.

A similar model, namely alternating two-way automata on infinite trees, were used by Grädel and Walukiewicz [6] in their decision procedure for satisfiability of guarded fixpoint logic. They reduced satisfiability to the emptiness problem for alternating two-way automata on infinite trees. The latter problem is known to be decidable, by Vardi [9].

A two-way alternating automaton on infinite trees can move its head to the parent, or to the children. In this paper, instead of automata on trees, we consider automata on undirected graphs. In an undirected graph, the automaton can only ask to follow an edge, and there is no notion of forward or backward. This is much in the spirit of [6], where the automata on trees would not care about the distinction between parent or child edges.

3.1. Definition of alternating automata on undirected graphs

We now define the automaton model that we use. An alternating automaton on undirected graphs is defined by: an input alphabet Σ, a set of states $Q$, a partition $Q = Q_\forall \cup Q_\exists$, an initial state $q_I$, a ranking function $\Omega : Q \to \mathbb{N}$ for the parity acceptance condition, and a transition relation

$$\delta \subseteq Q \times \Sigma \times \{\epsilon, \text{edge}\} \times Q.$$ 

An input to the automaton is an undirected graph whose nodes are labelled by Σ, and a designated node $v$ of the graph. The automaton accepts an input graph $G$ from an initial node $v_0$ if player $\exists$ wins the parity game defined below.

The arena of the parity game consists of pairs of the form $(v, q)$, where $v$ is a node of $G$, and $q$ is a state of the automaton. The initial position in the arena is $(v_0, q_I)$. The rank of a position $(v, q)$, as used by the parity condition, is $\Omega(q)$. Let $u$ be a node of the input graph, and let $a \in \Sigma$ be its label. In the arena of the game, there is an edge from $(u, q)$ to $(w, p)$ if:

- there is a transition $(q, a, \epsilon, p)$ and $u = w$; or
- there is a transition $(q, a, \text{edge}, p)$ and there is an edge in the input graph from $u$ to $w$.

Some alternating automata on undirected graphs accept infinite graphs, but no finite graphs. Therefore, it makes sense to consider the decision problem: does an automaton accept a finite graph? This problem was shown decidable in [3, 4].
**Theorem 2** ([3, 4]). *Given an alternating automaton on undirected graphs it is decidable in exponential time in the number of states of the automaton, whether or not it accepts some finite graph.*

Formally, [3, 4] considered two-way automata on directed graphs with the automaton having transitions corresponding to: staying in the same node, moving forward along an edge, and moving backward along an edge. The two-way model is more general than the one for undirected graphs.

### 3.2. Undirected bisimulation

We write $\text{nodes}(G)$ for the nodes of a graph $G$. Consider two undirected graphs $G_0$ and $G_1$, with node labels. An undirected bisimulation is a set

$$Z \subseteq \text{nodes}(G_0) \times \text{nodes}(G_1)$$

with the following properties. If $(v_0, v_1)$ belongs to $Z$, then the node labels of $v_0$ and $v_1$ are the same. Also, for any $i \in \{0, 1\}$ and node $w_i$ connected to $v_i$ by an edge, there exits a node $w_{1-i}$ connected to $v_{1-i}$ by an edge and such that $(w_0, w_1) \in Z$.

We say that node $v_0$ in a graph $G_0$ is bisimilar to node $v_1$ in a graph $G_1$ if there is an undirected bisimulation that contains the pair $(v_0, v_1)$. In this case, for every alternating automaton on undirected graphs, the automaton accepts $G_0$ from $v_0$ if and only if it accepts $G_1$ from $v_1$.

### 3.3. Undirected unraveling

Consider an undirected graph $G$ and $v$ a node of $G$. The undirected unraveling of $G$ from $v$ is the graph $T$, whose nodes are paths in the graph $G$ that begin in $v$, and edges are placed between a path and the same path without the last node. The undirected unraveling is a tree. We write

$$\pi: \text{nodes}(T) \to \text{nodes}(G)$$

for the function that maps a path to its target node. If $G$ has node labels, then one labels the nodes of $T$ according to their images under $\pi$. Then, the graph of $\pi$ is an undirected bisimulation between $T$ and $G$.

### 4. Tabloids

In this section, we introduce a notion of tabloids largely in line with the tableaux of Grädel and Walukiewicz [6], but allowing tabloids to take the shape of arbitrary undirected graphs.
**Tabloid.** Fix a relational signature \( \Sigma \) and a set \( K \) of constant names. A **tabloid** over signature \( \Sigma \) and constants \( K \) is an undirected graph, where every node \( v \) is equipped with two labels: a set \( K_v \subseteq K \), called the **constants** of \( v \), and an atomic type \( \tau_v \) over \( K_v \), called the **type of** \( v \). If nodes \( v \) and \( w \) are connected by an edge in the graph, then the types \( \tau_v \) and \( \tau_w \) should agree over the constants from \( K_v \cap K_w \).

**A structure from a tree tabloid.** Consider a tabloid \( T \) whose underlying graph is a tree. Then we define a \( \Sigma \)-structure \( A(T) \) as follows. The universe of \( A(T) \) is built using pairs \((v,c)\), where \( v \) is a vertex of \( G \) and \( c \) is a constant of \( v \). The universe consists not of these pairs, but of their equivalence classes under the following equivalence relation: \((v,c)\) and \((v',c')\) are equivalent if \( c = c' \) and \( c \) occurs in the label of every node on the undirected path connecting \( v \) and \( v' \) in \( G \). The path is unique, because the underlying graph is a tree. We write \([v,c]\) for an equivalence class of such a pair. A tuple \(|[v_1,c_1],\ldots,[v_n,c_n]|\) satisfies a relation \( R \in \Sigma \) in \( A(T) \) if there is some node \( v \) such that

\[
[v,c_1] = [v_1,c_1], \ldots, [v,c_n] = [v_n,c_n]
\]

(1)

and \( R(c_1,\ldots,c_n) \) is in the atomic type \( \tau_v \). Because \( T \) is a tree, this definition does not depend on the choice of \( v \), since the set of nodes \( v \) such that (1) holds is connected. It is, however, unclear how to extend this construction to cyclic tabloids.

**Labelling with a formula.** Consider a tree tabloid \( T \) over a set of constants \( K \) and a relational signature \( \Sigma \). Let \( \varphi \) be a formula over \( \Sigma \). We define a new tree, call it \( T_{\varphi} \), which has the same nodes as \( T \), but more information in its labels. Consider a node \( v \) of \( T \) with constants \( K_v \), a subformula \( \psi \) of \( \varphi \), and a function \( \eta \) that maps free variables of \( \psi \) to constants in \( K_v \). For \( v \) and \( \eta \), define a valuation \([\eta]_v\), which maps free variables of \( \psi \) to elements of the structure \( A(T) \), by setting

\[
[\eta]_v(x) = [v,\eta(x)].
\]

Define the \( \varphi \)-type of the node \( v \) to be the set of pairs \((\psi,\eta)\) as above, such that the formula \( \psi \) is valid in \( A(T) \) under the valuation \([\eta]_v\). The set of \( \varphi \)-types is finite, and depends on \( K \) and \( \varphi \). Call this set \( \Gamma_{\varphi,K} \). Define \( T_{\varphi} \) to be the tree with the same nodes and edges as \( T \), but where every node is labelled by its \( \varphi \)-type.

The next theorem follows from Grädel and Walukiewicz in [6].
Theorem 3. Let $\varphi$ be a guarded fixpoint sentence of width $n$. Let $K$ be a set of $2n$ constants. One can compute an alternating two-way automaton $A_\varphi$ on graphs labelled by $\Gamma_{\varphi,K}$, such that for every tree tabloid $T$ over constants $K$,

$$T_\varphi \text{ is accepted by } A \iff A(T) \models \varphi.$$ 

The number of states in $A$, and the time to compute the automaton, are $O(|\varphi| \cdot \exp(n))$.

5. Algorithm for finite satisfiability

We now propose the algorithm for finite satisfiability of guarded fixpoint logic. Given a formula $\varphi$, we compute the automaton $A_\varphi$ using Theorem 3. Then, we test if the automaton $A_\varphi$ accepts some finite graph, using Theorem 2. This section is devoted to proving the correctness of this procedure as claimed in the next proposition.

Proposition 4. A formula $\varphi$ of guarded fixpoint logic has a finite model if, and only if, the associated automaton $A_\varphi$ accepts a finite graph.

5.1. From a finite accepted graph to a finite model

In this section, we prove that if the automaton $A_\varphi$ accepts a finite graph, then the formula $\varphi$ is satisfied in some finite structure.

Lemma 5. Let $G$ be a finite tabloid and $T$ its undirected unraveling. Then $A(T)$ has a guarded bisimulation of finite index over its set of guarded tuples.

Proof. Let $T$ be the undirected unravelling of $G$ and $\pi : \text{nodes}(T) \to \text{nodes}(G)$ the natural projection from $T$ onto $G$. Whenever $\pi(v) = \pi(w)$ then (i) $v$ and $w$ have the same label, and (ii) rooting $T$ in either $v$ or in $w$ results in isomorphic trees. Let $Z$ be the set of partial isomorphisms

$$([v, c_1], \ldots, [v, c_r]) \mapsto ([w, c_1], \ldots, [w, c_r])$$

ranging over pairs $v, w$ with the same image under $\pi$, and tuples of constants $c_1, \ldots, c_r$ such that $A(T) \models R([v, c_1], \ldots, [v, c_r])$ for some $R \in \Sigma$. By (i) we have that for any constants $c_1, \ldots, c_n$ and relation $R$ from $\Sigma$

$$A(T) \models R([v, c_1], \ldots, [v, c_r]) \iff A(T) \models R([w, c_1], \ldots, [w, c_r]).$$

From (ii) it follows that $Z$ is a guarded bisimulation. It has finite index because the image of $\pi$ is finite and $c_1, \ldots, c_r$ have finitely many possible values. \qed
Transforming an arbitrary guarded bisimulation of finite index into a finite model is a problem that was solved in [2].

**Theorem 6** ([2, Theorem 6]). Suppose that $\mathfrak{A}$ is a structure that has a guarded bisimulation of finite index over its set of guarded tuples. Then $\mathfrak{A}$ is guarded bisimilar to a finite structure.

### 5.2. From a finite model to a finite accepted graph

In this section, we prove that if $\varphi$ is satisfied in some finite structure then the automaton $\mathcal{A}_{\varphi}$ accepts some finite graph.

Let $\mathfrak{A}$ be a finite model of $\varphi$. We define a finite tabloid $G$ as follows. Choose a set $K$ of $2^n$ constants, where $n$ is the width of $\varphi$. Nodes of $G$ are injections $\chi : A \to K$, where $A$ is a guarded subset of the universe of $\mathfrak{A}$. The reason why we chose $K$ to have $2^n$ elements is that any guarded subset has at most $n$ elements. The set of constants of a node $\chi$ is the image $\chi(A)$. The type of $\chi$ is the image, under $\chi$, of the type of $A$ in $\mathfrak{A}$. We connect nodes $\chi$ and $\chi'$ if $\chi \cup \chi'$ is an injective function. This includes the case of them having disjoint domains and images.

Let $T$ be the undirected unraveling of $G$, and let $\pi : \text{nodes}(T) \to \text{nodes}(G)$ be the projection obtained from the tree unraveling. For each $v \in \text{nodes}(T)$, $\pi(v)$ is an injection $\chi_v : A_v \to K_v$ from a guarded set $A_v$ of elements of $\mathfrak{A}$ to the set $K_v$ of constant names in the label of $v$. Let $\gamma_v : K_v \to \mathfrak{A}(T)$ map each $c \in K_v$ to $[v,c]$. Then $\gamma_v \circ \chi_v$ is a partial isomorphism between guarded subsets of $\mathfrak{A}$ and $\mathfrak{A}(T)$ for each $v \in \text{nodes}(T)$, and it is not difficult to check that $\{\gamma_v \circ \chi_v \mid v \in \text{nodes}(T)\}$ forms a guarded bisimulation between $\mathfrak{A}$ and $\mathfrak{A}(T)$.

Because satisfaction of formulas of guarded fixpoint logic is preserved under guarded bisimulation, it follows that $\mathfrak{A}(T)$ satisfies the formula $\varphi$. Therefore, by Theorem 3, the tree $T_{\varphi}$ is accepted by the automaton $\mathcal{A}_{\varphi}$.

The result is still not complete, because $T_{\varphi}$ is an infinite tree, and not a finite graph. The tree $T_{\varphi}$ has the same nodes as $T$, so we can apply the projection $\pi$ to its nodes as well.

**Claim 7.** If $v, w$ are nodes of $T$ with the same image under $\pi$, then they have the same labels in $T_{\varphi}$.

**Proof.** Because labels in $T_{\varphi}$ are $\varphi$-types, we need to show that if $\pi(v) = \pi(w)$, then the $\varphi$-types are the same in $v$ and $w$. Let $K_v$ be the set of constants in the label of $v$ in $T$, which is the same as the set of constants in the label of $w$.
in $T$. Let $\psi$ be a subformula of $\varphi$, and let $\eta$ be a function from free variables of $\psi$ to $K_v$. We need to prove that $(\psi, \eta)$ holds in $v$ if and only if it holds in $w$. But this follows thanks to the bisimulation $Z$ defined in Lemma 5.

By the above claim, it makes sense to assign labels from $\Gamma_{\varphi,K}$ to nodes of $G$: for each node $v$, choose the label used by all nodes of $T_\varphi$ that are mapped by $\pi$ to $v$. Call $G_\varphi$ the resulting graph. The tree $T_\varphi$ is the undirected unraveling of $G_\varphi$. Therefore, the automaton $A$ must also accept $G_\varphi$, because alternating automata on undirected graphs are invariant under undirected unravelings. Since $G_\varphi$ has the same nodes as $G$, it is finite. This completes the proof of Proposition 4, thereby also our Main Theorem 1.

References


