2015 30th Annual ACM/IEEE Symposium on Logic in Computer Science

Interpolation with Decidable Fixpoint Logics

Michael Benedikt
University of Oxford

Balder ten Cate
UC Santa Cruz and LogicBlox Inc

Michael Vanden Boom
University of Oxford

***Abstract*—A logic satisfies Craig interpolation if whenever one formula ϕ_1 in the logic entails another formula ϕ_2 in the logic, there is an intermediate formula — one entailed by ϕ_1 and entailing ϕ_2 — using only relations in the common signature of ϕ_1 and ϕ_2. Uniform interpolation strengthens this by requiring the interpolant to depend only on ϕ_1 and the common signature. A uniform interpolant can thus be thought of as a minimal upper approximation of a formula within a subsignature. For first-order logic, interpolation holds but uniform interpolation fails. Uniform interpolation is known to hold for several modal and description logics, but little is known about uniform interpolation for fragments of predicate logic over relations with arbitrary arity. Further, little is known about ordinary Craig interpolation for logics over relations of arbitrary arity that have a recursion mechanism, such as fixpoint logics.**

In this work we take a step towards filling these gaps, proving interpolation for a decidable fragment of least fixpoint logic called unary negation fixpoint logic. We prove this by showing that for any fixed k, uniform interpolation holds for the k-variable fragment of the logic. In order to show this we develop the technique of reducing questions about logics with tree-like models to questions about modal logics, following an approach by Grädel, Hirsch, and Otto. While this technique has been applied to expressivity and satisfiability questions before, we show how to extend it to reduce interpolation questions about such logics to interpolation for the μ-calculus.

I. Introduction

A desirable property of a logic is *interpolation*: if one formula ϕ_1 entails another ϕ_2, there is a formula in the common signature that is entailed by the first and entails the second. It implies the well-known *Beth Definability Property* as well as its extension, the *Projective Beth Definability Property* stating that implicit specifications can be converted to explicit ones, which is important in knowledge representation [1] and for rewriting queries in terms of views in databases [2], [3]. Interpolation has many other applications, both in simplifying definitions and in verification [4].

It is even more desirable to have *effective* interpolation: given ϕ_1 and ϕ_2, one can determine if entailment holds and compute an interpolant if it does. An even stronger goal is to prove effective *uniform interpolation* theorems. In uniform interpolation, the interpolating formula depends only on ϕ_1 and the signature of ϕ_2. Uniform interpolation can be thought of as stating that ϕ_1 has a minimal approximation from above in the signature of ϕ_2.

The first interpolation result was proven for first-order logic by Craig [5]; the undecidability of validity in first-order logic implies that this cannot be made effective in the sense above. Effective interpolation results were later shown for fragments of first-order logic, such as the guarded negation fragment [6], as well as a number of modal and description logics. On the negative side, it is known that first-order logic does not have uniform interpolation [7].

The situation is much less clear for logics that contain recursion. The standard way to capture recursion in a logic is via a fixpoint operator. One adds second-order variables $X, Y, \ldots$ which are allowed in atomic formulas. These variables are bound not by second-order quantifiers, but by $[\mathbf{lfp}\, X, \boldsymbol{x}.\phi(X, \boldsymbol{x}, \boldsymbol{y})]$ where X is a second-order variable of arity $|\boldsymbol{x}|$. In the setting of modal logic, the addition of fixpoints gives one the μ-calculus, which can express important reachability properties of a labelled transition system. The greater expressiveness of fixpoint logics makes them a natural candidate for the stronger uniform interpolation property. D'Agostino and Hollenberg [8] showed that in fact the μ-calculus has uniform interpolation. Uniform interpolation has also been shown for other modal and description logics [9], [10]. Little is known about interpolation, uniform or otherwise, effective or otherwise, for fixpoint logics over general relational structures, where relations can have arbitrary arity.

Effective interpolation results are relevant only for logics where entailment is decidable. This is not the case for LFP, the fixpoint extension of first-order logic. However, several decidable fragments of LFP are known, including *guarded fixpoint logic* (GFP), *guarded negation fixpoint logic* (GNFP), and *unary negation fixpoint logic* (UNFP). All of these achieve decidability by restricting the use of either quantification or negation.

Our main result is effective uniform interpolation for unary fixpoint queries of some fixed *width* k, denoted UNFP^k (the width of a UNFP formula is, roughly, a bound on the maximal number of free variables in any subformula). In particular, this result provides a decidable extension of the description logic $\mathcal{ALCO}$ having effective uniform interpolation with respect to relation symbols. This contrasts with an earlier conjecture in [10, Section 7] that no such extension could have uniform interpolation with respect to both relations and constants.

From effective uniform interpolation for UNFP^k, we can deduce effective Craig interpolation for the full unary negation fixpoint language UNFP. The fact that we have effective interpolation for UNFP (and hence, e.g., the ability to effectively convert implicit to explicit definitions) is significant in that UNFP is quite expressive, subsuming recursive query languages such as Monadic Datalog (with stratified negation).

Our approach for proving uniform interpolation will work via reducing logics that always admit "tree-like models" — such as the guarded and unary negation fragments — to modal logics. This technique was introduced by Grädel, Hirsch and

1043-6871/15 $31.00 © 2015 IEEE
DOI 10.1109/LICS.2015.43

Otto [11]. In this approach we start with an input problem in some logic over unrestricted structures in some relational signature. Making use of the tree-like model property for the logic, we apply a "forward mapping" that translates to a corresponding problem in modal logics. Given a solution to the problem in the modal logic setting, we then apply a "backward mapping" to get a corresponding solution back in the setting of the original logic. [11] applied this technique to prove that GFP is precisely the "guarded-bisimulation" invariant fragment of a "guarded" second-order logic, in analogy to the fact that the μ-calculus is the bisimulation-invariant fragment of monadic second-order logic [12]. We will apply it to interpolation, using as our solution in the modal setting the interpolation result of D'Agostino and Hollenberg mentioned above.

We supplement our effective interpolation theorems with several negative results, showing the limitations of interpolation. In particular, we show that one cannot hope to extend the uniform interpolation result to full UNFP, and one can not extend even ordinary interpolation to the larger language of guarded negation fixpoint logic.

Due to space limitations, most proofs are deferred to the full version of this paper.

II. Preliminaries

A. Notation and conventions

We use $\boldsymbol{x}, \boldsymbol{y}, \ldots$ (respectively, $\boldsymbol{X}, \boldsymbol{Y}, \ldots$) to denote vectors of first-order (respectively, second-order) variables. For a formula ϕ, we write $\phi(\boldsymbol{x})$ to indicate that the free first-order variables in ϕ are among $\boldsymbol{x}$. If we want to emphasize that there are also free second-order variables $\boldsymbol{X}$, we write $\phi(\boldsymbol{x}, \boldsymbol{X})$. We often use α to denote atomic formulas, and for such formulas, if we write $\alpha(\boldsymbol{x})$ then we assume that the free variables in α are precisely $\boldsymbol{x}$.

A formula ϕ is assumed to be given in the standard tree representation of a formula, and the *size* of ϕ, denoted $|\phi|$, is the number of symbols in ϕ. We will sometimes represent ϕ using a node-labelled DAG (directed acyclic graph). The nodes represent formulas, and the edge relation connects a formula to its subformulas. The size of a DAG representation is the number of nodes and edges in the DAG.

B. Basics of unary negation and guarded logics

The *Unary Negation Fragment* of FO [13] (denoted UNF) is built up inductively according to the grammar:

$$\phi ::= R\,\boldsymbol{t} \mid \exists x.\phi \mid \phi \vee \phi \mid \phi \wedge \phi \mid \neg\phi(x)$$

where R is either a relation symbol or the equality relation, and $\boldsymbol{t}$ is a tuple over variables and constants. Notice that any use of negation must occur only on formulas with at most one free variable. The *Guarded Negation Fragment* GNF extends UNF by allowing $\alpha(\boldsymbol{x}) \wedge \neg\phi(\boldsymbol{x})$, where α is an atomic relation or equality that contains all of the free variables of the negated formula. Such an atomic relation is a *guard* of the formula. GNF is also related to the *Guarded Fragment* (GF), typically defined via the grammar:

$$\phi ::= R\,\boldsymbol{t} \mid \exists \boldsymbol{x}.(\alpha(\boldsymbol{x}\boldsymbol{y}) \wedge \phi(\boldsymbol{x}\boldsymbol{y})) \mid \phi \vee \phi \mid \phi \wedge \phi \mid \neg\phi(\boldsymbol{x})$$

where R is either a relation symbol or the equality relation, α is an atomic relation or equality relation, and $\boldsymbol{t}$ is a tuple over variables and constants. Here it is the quantification that is guarded, rather than negation. As in GNF, we allow equality guards by default. Note that GNF subsumes GF sentences and UNF formulas.

The fixpoint extensions of these logics (denoted GNFP, UNFP, and GFP) extend the base logic by adding to the grammars the additional production rule

$$[\mathbf{lfp}\, X, \boldsymbol{x}.\, \mathrm{gdd}(\boldsymbol{x}) \wedge \phi(\boldsymbol{x}, X, \boldsymbol{Y})](\boldsymbol{t})$$

where (i) X only appears positively in ϕ, (ii) second-order variables like X cannot be used as guards and (iii) $\mathrm{gdd}(\boldsymbol{x})$ expresses that $\boldsymbol{x}$ is guarded by an atom from the original signature (it can be understood as an abbreviation for the disjunction of existentially quantified relational atoms involving all of the variables in $\boldsymbol{x}$).[1] In UNFP, there is an additional requirement that $\boldsymbol{x}$ has at most one variable, so only unary or 0-ary predicates can be defined. GNFP subsumes both GFP sentences and UNFP formulas. These logics are all contained in LFP, the fixpoint extension of FO.

The semantics for the existential quantifier and boolean connectives is standard. We briefly review the semantics of $[\mathbf{lfp}\, X, \boldsymbol{x}.\, \mathrm{gdd}(\boldsymbol{x}) \wedge \phi(\boldsymbol{x}, X, \boldsymbol{Y})](\boldsymbol{t})$. Since $\phi(\boldsymbol{x}, X, \boldsymbol{Y})$ is monotone in X, it induces an operator $U \mapsto \mathcal{O}_\phi^{\mathfrak{A},\boldsymbol{V}}(U) := \{\boldsymbol{a} : \mathfrak{A}, U, \boldsymbol{V} \models \mathrm{gdd}(\boldsymbol{a}) \wedge \phi(\boldsymbol{a}, X, \boldsymbol{Y})\}$ on every structure $\mathfrak{A}$ with valuation $\boldsymbol{V}$ for $\boldsymbol{Y}$, and this operator has a least fixpoint. Given some ordinal β, the *fixpoint approximant* $\phi^\beta(\mathfrak{A}, \boldsymbol{V})$ of ϕ on $\mathfrak{A}, \boldsymbol{V}$ is defined such that

$$\begin{aligned} \phi^0(\mathfrak{A}, \boldsymbol{V}) &:= \emptyset \\ \phi^{\beta+1}(\mathfrak{A}, \boldsymbol{V}) &:= \mathcal{O}_\phi^{\mathfrak{A},\boldsymbol{V}}(\phi^\beta(\mathfrak{A}, \boldsymbol{V})) \\ \phi^\beta(\mathfrak{A}, \boldsymbol{V}) &:= \bigcup_{\beta' < \beta} \phi^{\beta'}(\mathfrak{A}, \boldsymbol{V}) \quad \text{where } \beta \text{ is a limit ordinal.} \end{aligned}$$

We let $\phi^\infty(\mathfrak{A}, \boldsymbol{V}) := \bigcup_\beta \phi^\beta(\mathfrak{A}, \boldsymbol{V})$ denote the least fixpoint based on this operation. Thus, $[\mathbf{lfp}\, X, \boldsymbol{x}.\, \mathrm{gdd}(\boldsymbol{x}) \wedge \phi(\boldsymbol{x}, X, \boldsymbol{Y})]$ defines a new predicate named X of arity $|\boldsymbol{x}|$, and $\mathfrak{A}, \boldsymbol{V}, \boldsymbol{a} \models [\mathbf{lfp}\, X, \boldsymbol{x}.\, \mathrm{gdd}(\boldsymbol{x}) \wedge \phi(\boldsymbol{x}, X, \boldsymbol{Y})](\boldsymbol{x})$ iff $\boldsymbol{a} \in \phi^\infty(\mathfrak{A}, \boldsymbol{V})$. If $\boldsymbol{V}$ is empty or understood in context, we just write $\phi^\infty(\mathfrak{A})$.

We also write ϕ^β for the fixpoint approximations obtained by unfolding the fixpoint β times. That is, $\phi^0 := \bot$, $\phi^{\beta+1}(\boldsymbol{x}) := \mathrm{gdd}(\boldsymbol{x}) \wedge \phi[\phi^\beta(\boldsymbol{y})/X(\boldsymbol{y})]$, and if β is a limit ordinal, $\phi^\beta := \bigvee_{\beta' < \beta} \phi^{\beta'}$. This formula defines the β-approximation of the fixpoint process based on $[\mathbf{lfp}\, X, \boldsymbol{x}.\, \mathrm{gdd}(\boldsymbol{x}) \wedge \phi(\boldsymbol{x}, X, \boldsymbol{Y})]$. In general these formulas live in an infinitary version of the logic that allows conjunctions and disjunctions over arbitrary sets of formulas. However, for the logics that we are considering, if β is finite, then ϕ^β remains in the same logic.

[1] In GFP and UNFP, omitting $\mathrm{gdd}(\boldsymbol{x})$ does not change the expressivity of the logic; however, in GNFP this guardedness condition must be explicitly enforced.

We will sometimes utilize *simultaneous fixpoints*, or *vectorial fixpoints*, of the form $[\mathbf{lfp}\, X_i, \boldsymbol{x}_i.S](\boldsymbol{t})$ where

$$S = \begin{cases} X_1, \boldsymbol{x}_1 := \mathrm{gdd}(\boldsymbol{x}_1) \wedge \phi_1(\boldsymbol{x}_1, X_1, \ldots, X_j, \boldsymbol{Y}) \\ \vdots \\ X_j, \boldsymbol{x}_j := \mathrm{gdd}(\boldsymbol{x}_j) \wedge \phi_j(\boldsymbol{x}_j, X_1, \ldots, X_j, \boldsymbol{Y}) \end{cases}$$

is a system of formulas ϕ_i where $X_1, \ldots, X_j$ occur positively, and satisfy the same requirements for the body of the fixpoint formulas as before. Such a system can be viewed as defining a monotone operation on vectors of relations, and $[\mathbf{lfp}\, X_i, \boldsymbol{x}_i.S](\boldsymbol{t})$ expresses that $\boldsymbol{t}$ is a tuple in the i-th component of the least fixpoint defined by this operation. Allowing simultaneous fixpoints does not change the expressivity of these logics since they can be eliminated in favor of traditional fixpoints using the Bekič principle [14], with a possible exponential blow-up in the size of the formula, and only a polynomial blow-up if a DAG-representation is used.

These logics are expressive: modal logic is contained in each of these logics (even without fixpoints), every *union of conjunctive queries* (UCQ) is expressible in UNF and GNF, and every GF sentence can be expressed in GNF [15]. Nevertheless, these logics are decidable and have nice model theoretic properties (see Theorem 2). In the first part of the paper we will focus primarily on UNFP. This includes Monadic Datalog and its extension with stratified negation. UNFP can also be viewed as an expressive generalization of many description logics: the concept language of many description logics is contained in UNFP. In particular, this holds for $\mathcal{ALCIO}_{reg}$, the extension of the basic description logic $\mathcal{ALC}$ with inverse roles, individuals, and the regular role operators [16]. UNFP also subsumes the hybrid μ-calculus, which contains some description logics [17].

We pause to give a simple example of UNFP expressing a reachability property.

Example 1. *Consider structures over a single unary relation P and a binary relation R. Then*

$$\neg \exists y. \left((y = y) \wedge \neg [\mathbf{lfp}\, Y, y.Py \vee \exists z.(Ryz \wedge Yz)](y) \right)$$

expresses that every element can R-reach an element where P holds. This sentence is in UNFP *and* GFP*.*

It is often helpful to consider the formulas in a normal form. For instance, UNFP formulas in *normal form* can be generated using the following grammar:

$$\phi ::= \bigvee_i \exists \boldsymbol{x}. \bigwedge_j \psi_{ij} \mid [\mathbf{lfp}\, X, x.\phi(x, X, \boldsymbol{Y})](t)$$
$$\psi ::= R\,\boldsymbol{t} \mid X\,t \mid \phi(x) \mid \neg\phi(x).$$

The idea is that each UNFP formula in normal form is built from fixpoint predicates and UCQ-shaped formulas, where each conjunct in a CQ-shaped subformula is an atom, a normalized subformula with at most one free variable, or the negation of a normalized subformula with at most one free variable. Every UNFP-formula ϕ can be converted into this form in a canonical way, with an exponential blow-up in size. The *width* of ϕ is the maximum number of free variables in any subformula of its equivalent normal form (obtained in this canonical way). We remark that this is different than the normal form used in [13], but the width of a UNFP-formula under the two definitions is identical. We denote by UNFP^k the set of UNFP-*formulas of width at most k*.

The following theorem summarizes the decidability and model theoretic results about UNFP and GNFP that we will make use of.

Theorem 2 ([13],[15])**.** *Satisfiability and finite satisfiability are* 2EXPTIME*-complete for* GNF *and* GNFP *(and hence for* UNF *and* UNFP*).*

GNF *(and hence* UNF*) has the finite-model property: if ϕ is satisfiable, then ϕ is satisfiable in a finite structure. This does not hold for* UNFP *or* GNFP.

GNFP *(and hence* UNFP*) has the tree-like model property: if ϕ is satisfiable, then ϕ is satisfiable over structures of bounded tree-width (in fact, tree-width $\mathrm{width}(\phi) - 1$).*

C. Interpolation

Given a logic $\mathcal{L}$ and a relational signature σ, possibly including constants, we write $\mathcal{L}[\sigma]$ to denote the logic over this signature σ. We write $\mathrm{rel}(\sigma)$ for the set of relations and $\mathrm{con}(\sigma)$ for the set of constants in σ.

For formula φ_L and φ_R over signatures σ_L and σ_R, we write $\varphi_\mathrm{L} \models \varphi_\mathrm{R}$ (φ_L *entails* φ_R) if every model of the antecedent φ_L is a model of the consequent φ_R, and we say this entailment is a *validity*.

An *interpolant* for such a validity is a formula θ for which $\varphi_\mathrm{L} \models \theta$ and $\theta \models \varphi_\mathrm{R}$, and θ mentions only relations in $\mathrm{rel}(\sigma_\mathrm{L}) \cap \mathrm{rel}(\sigma_\mathrm{R})$.

We say a logic $\mathcal{L}$ has *Craig interpolation* if any validity $\varphi_\mathrm{L} \models \varphi_\mathrm{R}$ for φ_L and φ_R in $\mathcal{L}$, has an interpolant θ in $\mathcal{L}$. Craig famously proved that FO has Craig interpolation [5]. In [6], the authors demonstrated that GNF even has *effective Craig interpolation*: the GNF interpolant can be constructed from the GNF validity.

We say $\mathcal{L}$ has *uniform interpolation* if given some φ_L over signature σ_L and some subsignature σ' of σ_L, there is a formula θ in $\mathcal{L}$ that is an interpolant for all validities $\varphi_\mathrm{L} \models \varphi_\mathrm{R}$ with $\mathrm{rel}(\varphi_\mathrm{L}) \cap \mathrm{rel}(\varphi_\mathrm{R}) \subseteq \mathrm{rel}(\sigma')$. That is, the interpolant depends only on the antecedent and the common signature, rather than on the particular consequent. It is clear that uniform interpolation implies Craig interpolation.

D. Main results

The main result of this work is the following:

Theorem 3 (Uniform interpolation for UNFP^k)**.** *Let φ_L be a sentence in* UNFP$^k[\sigma]$*. Then for all signatures σ' with $\mathrm{rel}(\sigma') \subseteq \mathrm{rel}(\sigma)$ and $\mathrm{con}(\sigma') = \mathrm{con}(\sigma)$, there is a sentence $\chi \in$* UNFP$^k[\sigma']$ *such that*

- $\varphi_\mathrm{L} \models \chi$*, and*
- $\chi \models \varphi_\mathrm{R}$ *for all sentences $\varphi_\mathrm{R} \in$* UNFP$^k[\sigma_\mathrm{R}]$ *where $\varphi_\mathrm{L} \models \varphi_\mathrm{R}$ and $\mathrm{rel}(\sigma) \cap \mathrm{rel}(\sigma_\mathrm{R}) \subseteq \mathrm{rel}(\sigma')$ and $\mathrm{con}(\sigma_\mathrm{R}) \subseteq \mathrm{con}(\sigma')$.*

Furthermore, a DAG-representation for χ of size at most doubly exponential in $|\varphi_\mathrm{L}|$ can be constructed in 2EXPTIME.

We emphasize that the subsignature σ' for the uniform interpolant can only restrict the relations, not the constants, in σ. We discuss this more in Section IV-B.

An immediate corollary of Theorem 3 is that UNFP has Craig interpolation.

Corollary 4 (Craig interpolation for UNFP). *Let $\varphi_{\mathrm{L}} \in \mathrm{UNFP}[\sigma_{\mathrm{L}}]$ and $\varphi_{\mathrm{R}} \in \mathrm{UNFP}[\sigma_{\mathrm{R}}]$ be sentences such that $\varphi_{\mathrm{L}} \models \varphi_{\mathrm{R}}$. Then there is a sentence $\chi \in \mathrm{UNFP}^K[\sigma']$ such that $\varphi_{\mathrm{L}} \models \chi$ and $\chi \models \varphi_{\mathrm{R}}$, where $\mathrm{rel}(\sigma') := \mathrm{rel}(\sigma_{\mathrm{L}}) \cap \mathrm{rel}(\sigma_{\mathrm{R}})$, $\mathrm{con}(\sigma') := \mathrm{con}(\sigma_{\mathrm{L}}) \cup \mathrm{con}(\sigma_{\mathrm{R}})$, and $K := \max\{\mathrm{width}(\varphi_{\mathrm{L}}), \mathrm{width}(\varphi_{\mathrm{R}})\}$. Furthermore, a DAG-representation for χ of size at most doubly exponential in the size of φ_{L} and φ_{R} can be constructed in* 2EXPTIME.

E. Building blocks

In order to prove our results we will make use of GSO, a rich logic extending UNFP. We will also use two logics, MSO and L_μ, that will be interpreted over restricted signatures, so we review here some important prior results.

Guarded second-order logic. *Guarded second-order logic* (denoted GSO) over a signature σ is a fragment of second-order logic in which second-order quantification is interpreted only over guarded relations, i.e. over relations where every tuple in the relation is guarded by some predicate from σ. We refer the interested reader to [11] for more background and some equivalent definitions of this logic.

The logics UNFP, GNFP, and GFP considered in this paper can all be translated into GSO.

Proposition 5. *Given $\phi \in \mathrm{GNFP}[\sigma]$, we can construct an equivalent $\phi' \in \mathrm{GSO}[\sigma]$.*

Proof sketch: By structural induction on ϕ. The interesting case is for the least fixpoint. If $\phi(\boldsymbol{y}) = [\mathbf{lfp}\, X, \boldsymbol{x}.\, \mathrm{gdd}(\boldsymbol{x}) \wedge \psi(X, \boldsymbol{x})](\boldsymbol{y})$ then

$$\phi'(\boldsymbol{y}) := \forall X.[(\forall \boldsymbol{x}.((\mathrm{gdd}(\boldsymbol{x}) \wedge \psi'(X, \boldsymbol{x})) \to X\boldsymbol{x})) \to X\boldsymbol{y}]$$

where second-order quantifiers range over guarded relations. ■

Transition systems and their logics. A special kind of signature is a *transition system signature*, of the form $\tilde{\sigma}$ consisting of unary predicates (corresponding to a set of propositions $\mathit{props}(\tilde{\sigma})$) and binary predicates (corresponding to a set of actions $\mathit{actions}(\tilde{\sigma})$). A structure for such a signature is a *transition system*. Trees allowing both edge-labels and node-labels have a natural interpretation as transition systems.

We will be interested in two logics over transition systems signatures. One is *monadic second-order logic* (denoted MSO) — where second-order quantification is only over unary relations. MSO is contained in GSO, because unary relations are trivially guarded.

While MSO and GSO can be interpreted over arbitrary signatures, there are logics that have syntax specific to transition system signatures. One is the *modal μ-calculus* (denoted L_μ), an extension of modal logic with fixpoints. Given a transition system signature $\tilde{\sigma}$, formulas $\phi \in \mathrm{L}_\mu[\tilde{\sigma}]$ can be generated using the grammar

$$\phi ::= P \mid X \mid \phi \wedge \phi \mid \neg\phi \mid \langle\rho\rangle\phi \mid \mu X.\phi$$

where $P \in \mathit{props}(\tilde{\sigma})$, $\rho \in \mathit{actions}(\tilde{\sigma})$. The formulas $\mu X.\phi$ are required to use the variable X only positively in ϕ, and the semantics define a least-fixpoint operation based on ϕ. For instance, $\mu Y.(P \vee \langle E_R\rangle Y) \in \mathrm{L}_\mu$ holds at an element s in a transition system over a binary relation R and unary relation P iff s can R-reach an element where P holds. As usual, we refer to $\langle\rho\rangle\phi$ as a diamond modality, and $\mu X.\phi$ as a least fixpoint. Using negation, we also have the dual operators, the box modality $[\rho]\phi$ and the greatest fixpoint $\nu X.\phi$. We refer the reader to [18] for the formal semantics, and a survey of results. It is easy to see that L_μ can be translated into MSO.

Bisimulation games and logics. The logic L_μ applies to transition system signatures, and lies within MSO. Similarly the logics UNFP and GSO apply to arbitrary-arity signatures, with UNFP lying within GSO. It is easy to see that in both cases the containment is proper. In each case, what distinguishes the smaller logic from the larger is *invariance* under certain equivalences, called bisimulations.

For instance, the classical *bisimulation game* between transition systems $\mathfrak{A}$ and $\mathfrak{B}$ defines an equivalence relation over structures for a transition system signature $\tilde{\sigma}$. Positions in the game consist of pairs $(a, b) \in \mathrm{dom}(\mathfrak{A}) \times \mathrm{dom}(\mathfrak{B})$ that agree on all propositions in $\mathit{props}(\tilde{\sigma})$. From such a position (a, b), Spoiler first chooses which structure to move in, say $\mathfrak{A}$ (the choice of $\mathfrak{B}$ is symmetric). Spoiler then selects some a' with $(a, a') \in E_\rho^{\mathfrak{A}}$; if this is not possible, then the game terminates, and Duplicator wins. Duplicator must then choose some b' with $(b, b') \in E_\rho^{\mathfrak{B}}$ such that a' and b' agree on all propositions in $\mathit{props}(\tilde{\sigma})$. If this is not possible, the game terminates and Duplicator loses. Otherwise, the game proceeds from (a', b'). If the game never terminates then Duplicator wins. If Duplicator has a winning strategy in the bisimulation game starting from (a, b), we say $\mathfrak{A}, a$ and $\mathfrak{B}, b$ are *bisimilar*. We say two tree structures are bisimilar if they are bisimilar from their roots.

We say a formula ϕ is *$\tilde{\sigma}$-bisimulation-invariant* if it does not distinguish between $\tilde{\sigma}$-bisimilar transition systems. It is straightforward to check that $\mathrm{L}_\mu[\tilde{\sigma}]$-formulas are $\tilde{\sigma}$-bisimulation invariant. We will make use of a stronger result of Janin and Walukiewicz [12] that the μ-calculus is the bisimulation-invariant fragment of MSO (we state it here for trees because of how we use this later).

Theorem 6 ($\mathrm{L}_\mu \equiv$ bisimulation invariant MSO [12]). *Let $\tilde{\sigma}$ be a transition system signature. A class of trees is definable in $\mathrm{L}_\mu[\tilde{\sigma}]$ iff it is definable in $\mathrm{MSO}[\tilde{\sigma}]$ and closed under $\tilde{\sigma}$-bisimulation within the class of all $\tilde{\sigma}$-trees. Moreover, the translation between these logics is effective.*

There are variants of bisimulation and bisimulation games for the guarded logics mentioned earlier (see [19] and [15]). We describe here the UN^k*-bisimulation game* between σ-structures $\mathfrak{A}$ and $\mathfrak{B}$ corresponding to $\mathrm{UN}^k[\sigma]$*-bisimulation of width* k. Let $\mathfrak{A}$ and $\mathfrak{B}$ be σ-structures. We say f is a *partial unary-rigid homomorphism* if f is a partial homomorphism from $\mathfrak{A}$ to $\mathfrak{B}$ or $\mathfrak{B}$ to $\mathfrak{A}$ (relative to σ), and the restriction of f to any single element in its domain is a partial isomorphism.

Positions in the $\mathrm{UN}^k[\sigma]$ bisimulation game between $\mathfrak{A}$ and $\mathfrak{B}$ are partial unary-rigid homomorphisms f with $|\mathrm{dom}(f)| \leq k$. In general, we call these *bag positions*, since they represent a bag of at most k elements from each structure. In the special case when $|\mathrm{dom}(f)| \leq 1$, f is also called an *interface position*. We say the *active structure* is the structure containing $\mathrm{dom}(f)$.

The initial position is an empty partial homomorphism (an interface position). Starting in an interface position f, one round of the game consists of the following:

- Spoiler selects k elements $\boldsymbol{d}$ in the active structure (these elements need not be distinct, but they must include any elements in $\mathrm{dom}(f)$);
- Duplicator chooses $\boldsymbol{d}'$ in the other structure such that $g : \boldsymbol{d} \mapsto \boldsymbol{d}'$ is a partial unary-rigid homomorphism consistent with f (i.e. $f(c) = g(c)$ for all $c \in \mathrm{dom}(f)$).

Duplicator immediately loses if this is not possible. Otherwise, the game proceeds from the bag position g. Starting in a bag position g, one round of the game consists of the following:

- Spoiler collapses to at most one element $d \in \mathrm{dom}(g)$, and chooses h to be $d \mapsto g(d)$ or $g(d) \mapsto d$ (in case Spoiler collapses to the empty set, h is an empty partial homomorphism from $\mathfrak{A}$ to $\mathfrak{B}$ or vice versa).

The game proceeds from the interface position h.

If Duplicator has a winning strategy in this game, then we say that $\mathfrak{A}$ and $\mathfrak{B}$ are $\mathrm{UN}^k[\sigma]$-*bisimilar*. We say a sentence ϕ is $\mathrm{UN}^k[\sigma]$-*bisimulation invariant* if it does not distinguish between $\mathrm{UN}^k[\sigma]$-bisimilar structures. That is, if $\mathfrak{A}$ is $\mathrm{UN}^k[\sigma]$-bisimilar to $\mathfrak{B}$, then $\mathfrak{A} \models \phi$ iff $\mathfrak{B} \models \phi$. It is straightforward to show that $\mathrm{UNFP}^k[\sigma]$ sentences are $\mathrm{UN}^k[\sigma]$-bisimulation invariant.

Proposition 7. *Assume Duplicator has a winning strategy in the* $\mathrm{UN}^k[\sigma]$*-bisimulation game between* $\mathfrak{A}$ *and* $\mathfrak{B}$. *If* φ *is a* $\mathrm{UNFP}^k[\sigma]$ *sentence in normal form, then* $\mathfrak{A} \models \varphi$ *iff* $\mathfrak{B} \models \varphi$.

In the course of this work, we will prove a converse to this, an analog of the Janin-Walukiewicz theorem, showing that $\mathrm{UNFP}^k[\sigma]$ captures the $\mathrm{UN}^k[\sigma]$-bisimulation invariant subset of GSO (see Theorem 24).

Interpolation over transition systems. A motivation for this work was the main prior example of effective uniform interpolation for a fixpoint logic, D'Agostino and Hollenberg's result that L_μ has uniform interpolation [8]. Although it is not emphasized in their work, the proof they describe yields effective uniform interpolation.

Theorem 8 (Uniform interpolation for L_μ [8])**.** *Let* $\tilde{\sigma}$ *be a transition system signature, and let* $\tilde{\varphi}_\mathrm{L} \in \mathrm{L}_\mu[\tilde{\sigma}]$. *Then for all signatures* $\tilde{\sigma}'$ *such that* $\mathrm{rel}(\tilde{\sigma}') \subseteq \mathrm{rel}(\tilde{\sigma})$ *there is a formula* $\theta \in \mathrm{L}_\mu[\tilde{\sigma}']$ *such that*

- $\tilde{\varphi}_\mathrm{L} \models \theta$, *and*
- $\theta \models \tilde{\varphi}_\mathrm{R}$ *for all* $\tilde{\varphi}_\mathrm{R} \in \mathrm{L}_\mu[\tilde{\sigma}_\mathrm{R}]$ *where* $\tilde{\varphi}_\mathrm{L} \models \tilde{\varphi}_\mathrm{R}$ *and* $\mathrm{rel}(\tilde{\sigma}) \cap \mathrm{rel}(\tilde{\sigma}_\mathrm{R}) \subseteq \mathrm{rel}(\tilde{\sigma}')$.

Furthermore, θ *can be obtained effectively from* φ_L *and* $\tilde{\sigma}'$.

III. Uniform Interpolation for UNFP^k

A. Overview

The goal in this section is to prove Theorem 3, effective uniform interpolation for UNFP^k, but without the elementary bounds. As a by-product of this first "inefficient" version, we will get an analogue of the Janin-Walukiewicz theorem for UNFP^k (see Theorem 24). For now, we will only consider signatures without constants or equality. In the next section we will refine the construction to get the elementary bounds and we will describe how to adapt the arguments to handle features like constants and equality.

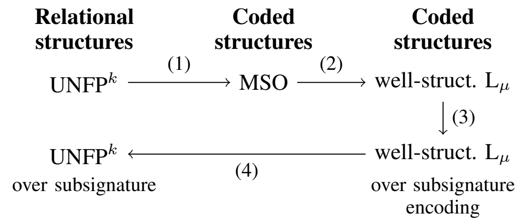

Fig. 1. Proof structure for effective uniform interpolation

We first give the idea behind our proof. The proof structure is presented in Figure 1.

Because of the tree-like model property for UNFP, we can restrict to structures of bounded tree-width. In fact, if k is the width of the UNFP-formula, then by Theorem 2 we can restrict to structures of tree-width $k-1$.

We now want to convert a UNFP^k sentence into a logical formula defining the trees that encode its tree-like models. This requires two steps: we must first argue that the set of codes is a regular set of trees, and we must come up with a corresponding "regular" representation of it. We call the representation in MSO the *naïve forward mapping step* (1). Secondly we have a *simplification step* (2): we use the fact that the original sentence was in UNFP^k, and hence UN^k-bisimulation invariant, to simplify the regular representation into a L_μ-formula, of a particular "well-structured" form, defined later.

Next is the L_μ*-interpolation step* (3): we apply uniform interpolation for L_μ in the coded world to get a new L_μ formula representing the interpolant we want on these tree codes. Finally, there is the *backward mapping step* (4): we translate this L_μ-formula back into UNFP^k, which yields a uniform interpolant for the original UNFP^k sentence.

It turns out that in order to actually transform from L_μ to UNFP^k it helps to have stronger properties on the tree-like models and coded structures. We capture this idea in the notion of a "shrewd unravelling" and "shrewd tree" below.

B. Schema for coding structures by trees

It is well-known that structures of tree-width $k-1$ can be interpreted by labeled trees over an alphabet that depends only on the signature of the structure and k. But in this work we will need to choose a special kind of encoding — for example, we will need that the decoding can be done within UNFP^k.

Fix some signature σ and some $k \in \mathbb{N}$. We now define a signature $\tilde{\sigma}_k$ for structures that encode tree decompositions of σ-structures of tree-width at most $k-1$. Informally, these coded tree structures will mimic the structure of a UN-bisimulation game of width k, alternating between *interface* *nodes* representing positions with at most one element, and *bag nodes* representing positions with k elements. Edges between nodes will indicate the relationship between the names of elements in each node.

Formally, we define $\tilde{\sigma}_k$ as follows.

- There is a unary relation $I \in \tilde{\sigma}_k$, which will be used to indicate interface nodes.
- There are unary relations $D_n \in \tilde{\sigma}_k$ for $n \in \{0, 1, k\}$, which will be used to indicate the number of elements represented at each node.
- For every relation $R \in \sigma$ of arity n and every sequence $\boldsymbol{i} = i_1 \dots i_n$ of n indices taken from $\{1, \dots, k\}$, there is a unary relation $R_{\boldsymbol{i}} \in \tilde{\sigma}_k$, which will be used to indicate that the tuple of elements indexed by $\boldsymbol{i}$ at that node is in R. For instance, if $R \in \sigma$ is a binary relation and $k = 2$, then we would have $R_{11}, R_{12}, R_{21}, R_{22} \in \tilde{\sigma}_k$. For a sequence $\boldsymbol{i} = i_1 \dots i_n$, we say that $R_{\boldsymbol{i}}$ uses *indices* $i_1 \dots i_n$.
- For every partial 1-1 map ρ from $\{1, \dots, k\}$ to $\{1, \dots, k\}$ with $\mathrm{dom}(\rho) = \{1\}$ or $\mathrm{rng}(\rho) = \{1\}$, there is a binary relation $E_\rho \in \tilde{\sigma}_k$. These will be used to indicate the relationship between elements in neighboring nodes.

Let $\mathit{Props}_k := \mathit{props}(\tilde{\sigma}_k)$ (respectively, $\mathit{Actions}_k := \mathit{actions}(\tilde{\sigma}_k)$) denote the collection of unary relations (respectively, binary relations) in $\tilde{\sigma}_k$. Likewise, let Props_k^1 denote the set of unary relations from $\tilde{\sigma}_k$ that use only one index. We will sometimes think of $\tilde{\sigma}_k$-structures as transitions systems over propositions Props_k and actions $\mathit{Actions}_k$. This means node labels come from $\mathcal{P}(\mathit{Props}_k)$ and edge labels from $\mathit{Actions}_k$.

Let $\mathcal{T}$ be a $\tilde{\sigma}_k$-tree. We say a node v is an *empty interface node* (respectively, *non-empty interface node*) if $v \in I$ and $v \in D_0$ (respectively, $v \in I$ and $v \in D_1$). We say v is a *bag node* if $v \notin I$ and $v \in D_k$. We say $\mathcal{T}$ is *consistent* if it satisfies certain natural conditions that ensure that these $\tilde{\sigma}_k$-structures correspond to a σ-structure: for instance, if $(u, v) \in E_\rho$, then for all $R \in \sigma$ of arity r and for all $\boldsymbol{i} \in \mathrm{dom}(\rho)^r$, $u \in R_{\boldsymbol{i}}$ iff $v \in R_{\rho(\boldsymbol{i})}$. We also enforce some simple properties about the structure of the coded trees, for instance, requiring that the root is an empty interface node, and nodes at even (respectively, odd) depths are interface nodes (respectively, bag nodes).

The tree decompositions of every σ-structure of tree-width $k - 1$ can be encoded in consistent $\tilde{\sigma}_k$-trees, and every consistent $\tilde{\sigma}_k$-tree corresponds to an actual σ-structure. Given a consistent $\tilde{\sigma}_k$-tree $\mathcal{T}$ with nodes $v, w \in \mathrm{dom}(\mathcal{T})$ and indices $i, j \in \{1, \dots, k\}$, we say (v, i) is equivalent to (w, j) if the i-th element in node v corresponds to the j-th element in node w, based on the edge label mappings between nodes in the tree. Let $[v, i]$ denote the equivalence class based on this equivalence relation. Using this, we can define the *decoding* of $\mathcal{T}$ to be the σ-structure $\mathfrak{D}(\mathcal{T})$ with universe $\{[v, i] : v \in \mathrm{dom}(\mathcal{T}) \text{ and } i \in \{1, \dots, k\}\}$ and $R^{\mathfrak{D}(\mathcal{T})}([v_1, i_1], \dots, [v_r, i_r])$ iff there is some node $w \in \mathrm{dom}(\mathcal{T})$ such that $w \in R_{j_1 \dots j_r}$ and $[w, j_m] = [v_m, i_m]$ for all $m \in \{1, \dots, r\}$. For v a node in a consistent tree $\mathcal{T}$ with $v \in D_n$, we write $\mathrm{elem}(v)$ for the vector of elements $a_1 \cdots a_n$ in $\mathfrak{D}(\mathcal{T})$ induced by v in $\mathcal{T}$.

The conditions for a consistent tree are definable in FO, and are $\tilde{\sigma}_k$-bisimulation invariant.

Lemma 9. *Within the class of $\tilde{\sigma}_k$-trees, the class of consistent $\tilde{\sigma}_k$-trees is $\tilde{\sigma}_k$-bisimulation invariant and definable in* $\mathrm{FO}[\tilde{\sigma}_k]$.

There is also a connection between bisimilarity of consistent trees and UN^k-bisimilarity of the corresponding structures.

Proposition 10. *If consistent $\tilde{\sigma}_k$-trees $\mathcal{T}$ and $\mathcal{T}'$ are $\tilde{\sigma}_k$-bisimilar, then $\mathfrak{D}(\mathcal{T})$ is $\mathrm{UN}^k[\sigma]$-bisimilar to $\mathfrak{D}(\mathcal{T}')$.*

C. Coding structures shrewdly

While the decoding of a consistent tree is well-defined, there are many ways to code a given structure, depending on the particular decomposition used. Below we will define particular codings based on unravellings.

The $\mathrm{UN}^k[\sigma]$-*unravelling* of a σ-structure $\mathfrak{A}$ is defined as follows. Consider the set Π of finite sequences of the form $X_0 Y_1 X_1 \dots Y_m$ or $X_0 Y_1 X_1 \dots Y_m X_m$, where each X_i is a tuple of elements of $\mathfrak{A}$ of size at most 1, each Y_i is a tuple of elements of $\mathfrak{A}$ of size k, and Y_{i+1} contains every element in X_i and X_{i+1}. We can assume $X_0 = \emptyset$. Each $\pi \in \Pi$ represents the projection to $\mathfrak{A}$ of a play in the $\mathrm{UN}^k[\sigma]$-bisimulation game between $\mathfrak{A}$ and some other structure, starting from the empty position.

We want to define a $\tilde{\sigma}_k$-tree $\mathcal{T} = \mathcal{T}^k(\mathfrak{A})$ based on this. Sequences $\pi \in \Pi$ can be arranged in a tree structure in the obvious way based on the prefix order. We interpret the predicates in $\tilde{\sigma}_k$ as discussed earlier. For all nodes π ending in a position $Y = a_1 \dots a_n$,

- $\pi \in I^{\mathcal{T}}$ iff π ends in an interface position, i.e. π is of the form $X_0 Y_1 X_1 \dots Y_m X_m$,
- $\pi \in D_j^{\mathcal{T}}$ iff $j = n$,
- $\pi \in R_{\boldsymbol{i}}^{\mathcal{T}}$ for $\boldsymbol{i} = i_1 \dots i_r$ iff $(a_{i_1}, \dots, a_{i_r}) \in R^{\mathfrak{A}}$,
- $(\pi, \pi') \in E_\rho^{\mathcal{T}}$ iff $\pi' = \pi Z$ and either (i) ρ is empty and there are no shared elements in Y and Z, or (ii) $\rho(i) = j$ for i (respectively, j) the index of the shared element in Y (respectively, Z).

The $\mathrm{UN}^k[\sigma]$-*unravelling* of $\mathfrak{A}$ is defined to be $\mathfrak{D}(\mathcal{T}^k(\mathfrak{A}))$, and its tree decomposition is encoded by $\mathcal{T}^k(\mathfrak{A})$.

Usually, $\mathcal{T}^k(\mathfrak{A})$ would be a suitable canonical coding of an infinite structure, allowing one to perform a forward and backward mapping as described in the overview. Indeed, an unravelling similar to this, but based on guarded bisimulation, is used in [11]. However, we were unable to do the backward mapping step when using this coding based on UN^k-unravelling, because this unravelling does not provide a close enough correspondence with the power — and limitations — of UNFP^k.

We overcome this difficulty by defining a "shrewd" unravelling. The idea is that a single tuple of elements in the original structure $\mathfrak{A}$ has many copies in the unravelling of $\mathfrak{A}$, as usual. However, we take this further, by including even more copies of certain parts of the structure, with variations to these copies that UNFP^k cannot distinguish. We capture the desired property of the coded structures in the following definition, and then define the corresponding unravelling that will yield coded trees like this. For $\tau, \tau' \subseteq Props_k$, we write $\tau' \subseteq_1 \tau$ if $\tau' \subseteq \tau$ and $\tau' \cap Props^1_k = \tau \cap Props^1_k$.

Definition 11 (Shrewd tree)**.** *We say a consistent $\tilde{\sigma}_k$-tree $\mathcal{T}$ is* shrewd *if it satisfies the following property. For all interface nodes v, if w is a ρ-child of v and τ is the set of unary predicates from $\tilde{\sigma}_k$ that hold at w, then for any $\tau' \subseteq_1 \tau$, there is a ρ-child w' of v such that*

- *τ' describes exactly the collection of unary predicates that hold at w', and*
- *the subtrees rooted at w and w' are isomorphic (ignoring w and w').*

We now define the *shrewd* $\mathrm{UN}^k[\sigma]$*-unravelling*. Given a σ-structure $\mathfrak{A}$ and some sequence $\boldsymbol{a} = a_1 \dots a_n$ of elements from $\mathfrak{A}$ (of size at most k), we define $\tau(\boldsymbol{a}) \subseteq Props_k$ such that $R_{\boldsymbol{i}} \in \tau(\boldsymbol{a})$ for $\boldsymbol{i} = i_1 \dots i_r$ iff $(a_{i_1}, \dots, a_{i_r}) \in R^{\mathfrak{A}}$. Now consider the set Π of finite sequences of the form $X_0(Y_1,\tau_1)X_1 \dots (Y_m,\tau_m)$ or $X_0(Y_1,\tau_1)X_1 \dots (Y_m,\tau_m)X_m$, where each X_i is a tuple of elements of $\mathfrak{A}$ of size at most 1, each Y_i is a tuple of elements of $\mathfrak{A}$ of size k, Y_{i+1} contains every element in X_i and X_{i+1}, and $\tau_i \subseteq_1 \tau(Y_i)$.

We can define a $\tilde{\sigma}_k$-tree $\mathcal{T} = \mathcal{S}^k(\mathfrak{A})$ based on this. Sequences $\pi \in \Pi$ can be arranged in a tree structure in the obvious way based on the prefix order. We interpret the predicates in $\tilde{\sigma}_k$ as follows. For all nodes π ending in a position of the form Y or (Y,τ) for $Y = a_1 \dots a_n$,

- $\pi \in I^{\mathcal{T}}$ iff π ends in an interface position Y,
- $\pi \in D^{\mathcal{T}}_j$ iff $j = n$,
- $\pi \in R^{\mathcal{T}}_{\boldsymbol{i}}$ iff π ends in the interface position Y and $R_{\boldsymbol{i}} \in \tau(\boldsymbol{a})$, or π ends in the bag position (Y,τ) and $R_{\boldsymbol{i}} \in \tau$,
- $(\pi,\pi') \in E^{\mathcal{T}}_\rho$ iff π' is of the form πZ or $\pi(Z,\tau)$ and either (i) ρ is empty and there are no shared elements in Y and Z, or (ii) $\rho(i) = j$ for i (respectively, j) the index of the shared element in Y (respectively, Z).

The *shrewd* UN^k*-unravelling* of $\mathfrak{A}$ is $\mathfrak{D}(\mathcal{S}^k(\mathfrak{A}))$, and its key properties are stated in the following lemma.

Lemma 12. *For all σ-structures $\mathfrak{A}$, $\mathcal{S}^k(\mathfrak{A})$ is a shrewd consistent $\tilde{\sigma}_k$-tree, and $\mathfrak{D}(\mathcal{S}^k(\mathfrak{A}))$ is* $\mathrm{UN}^k[\sigma]$*-bisimilar to $\mathfrak{A}$.*

We remark that none of the logics considered in this paper can enforce that a given consistent $\tilde{\sigma}_k$-tree $\mathcal{T}$ is shrewd or that $\mathcal{T}$ is the shrewd-unravelling of some structure $\mathfrak{A}$ (i.e. that $\mathcal{T} = \mathcal{S}^k(\mathfrak{A})$), but this is not problematic for our purposes.

D. Forward and backward mapping between relational structures and their encodings as trees

We are now ready to state the naïve forward mapping step mentioned in the overview, moving between coded $\tilde{\sigma}_k$-structures and σ-structures. This mapping will be from $\mathrm{GSO}[\sigma]$ to $\mathrm{MSO}[\tilde{\sigma}_k]$.

Theorem 13 ($\mathrm{GSO}[\sigma]$ to $\mathrm{MSO}[\tilde{\sigma}_k]$)**.** *Let ψ be a sentence in* $\mathrm{GSO}[\sigma]$*. For all k, we can construct a sentence $\psi^{\rightarrow} \in$* $\mathrm{MSO}[\tilde{\sigma}_k]$ *such that for all consistent $\tilde{\sigma}_k$ trees $\mathcal{T}$, $\mathfrak{D}(\mathcal{T}) \models \psi$ iff $\mathcal{T} \models \psi^{\rightarrow}$.*

Note that the mapping "plays well" with bisimulations.

Corollary 14. *If $\psi \in$* $\mathrm{GSO}[\sigma]$ *is a sentence that is invariant under* $\mathrm{UN}^k[\sigma]$*-bisimulation, then $\psi^{\rightarrow}$ is $\tilde{\sigma}_k$-bisimulation invariant on consistent $\tilde{\sigma}_k$-trees.*

Proof: Assume $\mathcal{T}$ and $\mathcal{T}'$ are $\tilde{\sigma}_k$-bisimilar consistent $\tilde{\sigma}_k$-trees with roots ϵ and ϵ', respectively. Then using Theorem 13 and Proposition 10, we have $\mathcal{T}, \epsilon \models \psi^{\rightarrow}$ iff $\mathfrak{D}(\mathcal{T}) \models \psi$ iff $\mathfrak{D}(\mathcal{T}') \models \psi$ iff $\mathcal{T}', \epsilon' \models \psi^{\rightarrow}$. ∎

Once we have moved from a UNFP^k formula over structures to an MSO formula on their encodings, the next step is the invariant simplification step, which gets us from bisimulation-invariant MSO on encodings to an L_μ formula on encodings. In fact, it is helpful at this stage to simplify to what we call "well-structured" L_μ. We say a L_μ-formula is *well-structured* if there is some set $Q = \{q_1, \dots, q_n\}$ with corresponding fixpoint variables X_{q_i} and fixpoints $\lambda_i \in \{\mu, \nu\}$ such that the formula is of the form

$$\lambda_n X_{q_n} \dots \lambda_1 X_{q_1} . \begin{pmatrix} \delta_{q_1} \\ \vdots \\ \delta_{q_n} \end{pmatrix} \quad \text{with}$$

$$\delta_q := \bigvee_{\tau} \left[\left(\bigwedge_{P \in \tau} P \wedge \bigwedge_{P \in Props_k \setminus \tau} \neg P \right) \wedge \delta_{q,\tau} \right]$$

$$\delta_{q,\tau} := \bigvee_{S} \left(\bigwedge_{(\rho,r) \in S} \langle \rho \rangle X_r \wedge \bigwedge_{\rho \in Actions_k} [\rho] \bigvee_{(\rho,r) \in S} X_r \right)$$

where the outer disjunction in δ_q ranges over some collection of $\tau \in \mathcal{P}(Props_k)$, and the outer disjunction in $\delta_{q,\tau}$ ranges over some collection of $S \in \mathcal{P}(Actions_k \times Q)$. We also require that there is some i such that the fixpoint predicates $X_{q_1}, \dots, X_{q_i}$ only contain bag nodes, and $X_{q_{i+1}}, \dots, X_{q_n}$ only contain interface nodes, by requiring that the corresponding τ in δ_q include or omit the proposition I as appropriate. The simplification step is captured in the following theorem.

Theorem 15. *Let $\psi \in$* $\mathrm{MSO}[\tilde{\sigma}_k]$ *be $\tilde{\sigma}_k$-bisimulation invariant over the class of $\tilde{\sigma}_k$-trees. Then we can effectively obtain a well-structured* L_μ*-formula ψ' such that for all $\tilde{\sigma}_k$-trees $\mathcal{T}$, $\mathcal{T} \models \psi$ iff $\mathcal{T} \models \psi'$.*

Proof: Apply the Janin-Walukiewicz theorem (Theorem 6), and then take advantage of the equivalence between L_μ and a form of automata called *μ-automata* [20]. The structure in the formula comes from the structure of the transition function of the automaton. ∎

Uniform interpolation of well-structured L_μ-formulas can then be done, with the help of Theorem 8; the well-structured form again follows by observing that the interpolation result in [8] actually uses the μ-automata mentioned earlier.

Theorem 16. *Let $\psi \in \mathrm{L}_\mu[\tilde{\sigma}]$ be a well-structured* L_μ*-formula. Let $\tilde{\sigma}'$ be a subsignature of $\tilde{\sigma}$. We can construct a well-structured $\theta \in \mathrm{L}_\mu[\tilde{\sigma}']$ such that θ is a uniform interpolant for ψ and $\tilde{\sigma}'$.*

Finally, the backward step moves from L_μ-formulas on encodings, back to UNFP^k on relational structures. This is the step that requires the most work.

Theorem 17 ($L_\mu[\tilde{\sigma}_k]$ to $\text{UNFP}^k[\sigma]$)**.** *Let ψ be a well-structured formula in $L_\mu[\tilde{\sigma}_k]$. We can construct a sentence $\psi^{\leftarrow} \in \text{UNFP}^k[\sigma]$ such that for all σ-structures $\mathfrak{B}$, $\mathfrak{B} \models \psi^{\leftarrow}$ iff $\mathcal{S}^k(\mathfrak{B}) \models \psi$.*

Proof sketch: Fix $\psi \in L_\mu[\tilde{\sigma}_k]$. We want to produce the $\text{UNFP}^k[\sigma]$ sentence $\psi^{\leftarrow}$. Unfortunately, the well-structured L_μ-formula cannot be translated directly into UNFP, since we may need to use non-unary negation in the translation of some $\bigwedge_{P \in Props_k \setminus \tau} \neg P$ (for instance, if $R_{21} \in Props_k \setminus \tau$, then $\neg R_{21}$ would become the non-unary negation $\neg R x_2 x_1$). Hence we first transform into a "UNFP-safe" version that is easier to work with; this transformation takes advantage of the fact that we are working over shrewd consistent trees (see Claim 18). Once we are in this form, we then give the backward mapping (Claim 19).

The UNFP-safe L_μ formulas restrict indices and other problematic constructs that would lead to negation in the backward mapping. Let $\text{indices}(\chi)$ denote the outermost indices in the formula, up to the next occurrences of a modality. We say an $L_\mu[\tilde{\sigma}_k]$ formula is UNFP-*safe for interface nodes* (respectively, UNFP-*safe for bag nodes*) if

- there are no explicit ν-fixpoints or box-modalities (they have been rewritten using negation, μ-fixpoints, and diamond modalities);
- negation is only applied to subformulas χ where $|\text{indices}(\chi)| \leq 1$;
- for every subformula χ in the scope of an even (respectively, odd) number of modalities, $\text{indices}(\chi)$ is contained in $\{1\}$ (respectively, $\{1, \ldots, k\}$);
- every fixpoint subformula or fixpoint variable is in the scope of an even (respectively, odd) number of modalities (i.e. fixpoints reference only interface nodes).

In general, we would not be able to convert an arbitrary L_μ-formula to an equivalent UNFP-safe form. However, because we are working over shrewd consistent trees, we can convert any well-structured L_μ formula to a UNFP-safe form. For instance, we can show that we can replace some instances of $\bigwedge_{P \in Props_k \setminus \tau} \neg P$ that are under a diamond modality with just $\bigwedge_{P \in Props_k^1 \setminus \tau} \neg P$.

Claim 18. *Let $\psi' \in L_\mu[\tilde{\sigma}_k]$ be a well-structured formula. Then we can effectively obtain a* UNFP-*safe formula φ such that for all shrewd consistent trees $\mathcal{T}$, $\mathcal{T} \models \psi'$ iff $\mathcal{T} \models \varphi$.*

Once we have this UNFP-safe L_μ formula, we can translate into UNFP^k. The proof proceeds by induction on the structure of the formula, so we must handle free variables. For each fixpoint variable X we introduce two fixpoint variables X_1 and X_0 to handle non-empty and empty interface nodes, respectively; we define $X^{\leftarrow} := (X_0, X_1)$. A set J of nodes in $\mathcal{S}^k(\mathfrak{B})$ is a UNFP-*safe valuation for a free variable X* if it (i) contains only interface nodes, and (ii) if it contains an empty interface node then it contains every empty interface node. We write $J^{\leftarrow}$ for its representation in $\mathfrak{B}$, i.e. $J^{\leftarrow} := (J_0, J_1)$ where J_1 is defined to be $\{\text{elem}(v) : v \in J \text{ and } |\text{elem}(v)| = 1\}$, and J_0 is $\top$ (respectively, $\bot$) if J contains all empty interface node (respectively, contains no empty interface nodes). The strengthened result for formulas with free variables is captured in the following claim.

Claim 19. *Let $\varphi \in L_\mu[\tilde{\sigma}_k]$ be* UNFP-*safe for interface nodes (respectively, bag nodes) with free second-order variables $\boldsymbol{X}$. We can construct* $\text{UNFP}[\sigma]$-*formulas $\varphi_0^{\leftarrow}(\boldsymbol{X}^{\leftarrow})$, $\varphi_1^{\leftarrow}(x_1, \boldsymbol{X}^{\leftarrow})$, and $\varphi_k^{\leftarrow}(x_1, \ldots, x_k, \boldsymbol{X}^{\leftarrow})$ of width k such that for all σ-structures $\mathfrak{B}$, for all* UNFP-*safe valuations $\boldsymbol{J}$ of $\boldsymbol{X}$, and for all interface nodes (respectively, bag nodes) v in $\mathcal{S}^k(\mathfrak{B})$ with $|\text{elem}(v)| = m$,*

$$\mathfrak{B}, \text{elem}(v), \boldsymbol{J}^{\leftarrow} \models \varphi_m^{\leftarrow} \text{ iff } \mathcal{S}^k(\mathfrak{B}), v, \boldsymbol{J} \models \varphi.$$

Moreover, if $\text{indices}(\varphi) = \{i_1, \ldots, i_n\}$, then $\text{free}(\varphi_m^{\leftarrow}) = \{x_{i_1}, \ldots, x_{i_n}\}$, and any strict subformula in $\varphi_m^{\leftarrow}$ that begins with an existential quantifier and is not directly below another existential quantifier has at most one free variable.

We construct the formulas as follows.

- Assume $\varphi = I$. Then $\varphi_0^{\leftarrow} := \top$, $\varphi_1^{\leftarrow} := \top$, and $\varphi_k^{\leftarrow} := \bot$.
- Assume $\varphi = D_j$. Then $\varphi_m^{\leftarrow} := \begin{cases} \top & \text{if } m = j, \\ \bot & \text{otherwise.} \end{cases}$
- Assume $\varphi = R_{i_1 \ldots i_n}$. Then $\varphi_0^{\leftarrow} := \bot$, $\varphi_1^{\leftarrow} := R x_{i_1} \ldots x_{i_n}$, and $\varphi_k^{\leftarrow} := R x_{i_1} \ldots x_{i_n}$.
- Assume $\varphi = X$. Then $\varphi_0^{\leftarrow} := X_0$, $\varphi_1^{\leftarrow} := X_1 x_1$, and $\varphi_k^{\leftarrow} := \bot$.
- The translation commutes with $\vee$, $\wedge$, and $\neg$. For the negation case, the definition of UNFP-safety and the inductive hypothesis ensures that the resulting formulas are in UNFP.
- Assume $\varphi = \langle \rho \rangle \chi$. Then
$$\varphi_0^{\leftarrow} := \exists y_1 \ldots y_k. \left(\chi_k^{\leftarrow}(y_1, \ldots, y_k)\right),$$
$$\varphi_1^{\leftarrow} := \exists y_1 \ldots y_k. \left(x_1 = y_{\rho(1)} \wedge \chi_k^{\leftarrow}(y_1, \ldots, y_k)\right),$$
$$\varphi_k^{\leftarrow} := \begin{cases} \chi_1^{\leftarrow}(x_i) & \text{if } \text{dom}(\rho) = \{i\} \\ \chi_0^{\leftarrow} & \text{if } \text{dom}(\rho) = \emptyset \end{cases}.$$
- Assume $\varphi = \mu Y. \chi(\boldsymbol{X}, Y)$. Then $\varphi_0^{\leftarrow}$ and $\varphi_1^{\leftarrow}$ are the first and second components of the vectorial fixpoint
$$\varphi^{\leftarrow} := \mathbf{lfp} \begin{pmatrix} Y_0 \\ Y_1, y_1 \end{pmatrix} . \begin{pmatrix} \chi_0^{\leftarrow}(\boldsymbol{X}^{\leftarrow}, Y_0, Y_1) \\ \chi_1^{\leftarrow}(y_1, \boldsymbol{X}^{\leftarrow}, Y_0, Y_1) \end{pmatrix}$$
and $\varphi_k^{\leftarrow} := \bot$.

It can be checked that the constructed formulas have width k.

This concludes the proof sketch of Theorem 17, the backward mapping going from L_μ to UNFP. ■

E. Putting it all together

We are now ready to give an initial algorithm for effective uniform interpolation for UNFP^k, without elementary bounds.

Fix $\varphi_L \in \text{UNFP}^k[\sigma]$, and fix some $\sigma' \subseteq \sigma$. Let $\tilde{\sigma}_k$ and $\tilde{\sigma}'_k$ be the signatures of tree representations for σ-structures and σ'-structures, respectively.

1) *Naïve forward mapping step*: Convert φ_L to GSO using Proposition 5, and then obtain $\varphi_L{}^{\rightarrow} \in \text{MSO}[\tilde{\sigma}_k]$ using Theorem 13. This is $\tilde{\sigma}_k$-bisimulation invariant over consistent $\tilde{\sigma}_k$-trees by Corollary 14.

2) *Simplification step:* Let $\gamma_{\sigma,k}$ be an FO$[\tilde{\sigma}_k]$-sentence expressing the consistency conditions for the trees over $\tilde{\sigma}_k$, using Lemma 9. Then $\gamma_{\sigma,k} \wedge \varphi_{\mathrm{L}}{}^{\rightarrow}$ is $\tilde{\sigma}_k$-bisimulation invariant over all $\tilde{\sigma}_k$-trees, so we can obtain well-structured $\tilde{\varphi}_{\mathrm{L}} \in \mathrm{L}_\mu[\tilde{\sigma}_k]$ by applying Theorem 15 to $\gamma_{\sigma,k} \wedge \varphi_{\mathrm{L}}{}^{\rightarrow}$.
3) L_μ *interpolation step*: Obtain well-structured $\theta \in \mathrm{L}_\mu[\tilde{\sigma}'_k]$ by applying Theorem 8 to $\tilde{\varphi}_{\mathrm{L}}$ and $\tilde{\sigma}'_k$.
4) *Backward mapping step*: Obtain $\theta^{\leftarrow} \in \mathrm{UNFP}^k[\sigma']$ using Theorem 17. Set $\chi := \theta^{\leftarrow}$.

First we prove that $\varphi_{\mathrm{L}} \models \chi$. Let $\mathfrak{B}$ be a σ-structure and assume $\mathfrak{B} \models \varphi_{\mathrm{L}}$. Then $\mathfrak{D}(\mathcal{S}^k(\mathfrak{B})) \models \varphi_{\mathrm{L}}$ (since φ_{L} is $\mathrm{UN}^k[\sigma]$-bisimulation invariant), so $\mathcal{S}^k(\mathfrak{B}) \models \varphi_{\mathrm{L}}{}^{\rightarrow}$ by Theorem 13. Since $\mathcal{S}^k(\mathfrak{B})$ is a consistent $\tilde{\sigma}_k$-tree, $\mathcal{S}^k(\mathfrak{B}) \models \gamma_{\sigma,k} \wedge \varphi_{\mathrm{L}}{}^{\rightarrow}$, so $\mathcal{S}^k(\mathfrak{B}) \models \tilde{\varphi}_{\mathrm{L}}$. By Theorem 8, this means that $\mathcal{S}^k(\mathfrak{B}) \models \theta$ (now viewed as a $\tilde{\sigma}'_k$ structure). Finally, by Theorem 17, this means that $\mathfrak{B} \models \theta^{\leftarrow}$.

Next, assume that $\varphi_{\mathrm{L}} \models \varphi_{\mathrm{R}}$ for some $\varphi_{\mathrm{R}} \in \mathrm{UNFP}^k[\sigma_{\mathrm{R}}]$ where $\sigma \cap \sigma_{\mathrm{R}} \subseteq \sigma'$. Let $\sigma'' = \sigma \cup \sigma_{\mathrm{R}}$. Let $\tilde{\varphi}_{\mathrm{L}} \in \mathrm{L}_\mu[\tilde{\sigma}_k]$ as above. Let $\tilde{\varphi}_{\mathrm{R}} \in \mathrm{L}_\mu[\tilde{\sigma}_{\mathrm{R},k}]$ be the formula obtained by applying Theorem 6 to $\gamma_{\sigma_{\mathrm{R}},k} \rightarrow \varphi_{\mathrm{R}}{}^{\rightarrow}$, where $\varphi_{\mathrm{R}}{}^{\rightarrow}$ is obtained using Theorem 13. We aim to prove that $\chi \models \varphi_{\mathrm{R}}$.

We first prove that $\tilde{\varphi}_{\mathrm{L}} \models \tilde{\varphi}_{\mathrm{R}}$ over all $\tilde{\sigma}''_k$-trees. Assume $\mathcal{T} \models \tilde{\varphi}_{\mathrm{L}}$, where $\mathcal{T}$ is a $\tilde{\sigma}''_k$-tree. Then $\mathcal{T} \models \gamma_{\sigma,k} \wedge \varphi_{\mathrm{L}}{}^{\rightarrow}$ so we know that $\mathcal{T}$ is a consistent $\tilde{\sigma}_k$-tree. If $\mathcal{T}$ is not $\tilde{\sigma}_{\mathrm{R},k}$-consistent, then $\mathcal{T} \models \gamma_{\sigma_{\mathrm{R}},k} \rightarrow \varphi_{\mathrm{R}}{}^{\rightarrow}$, so $\mathcal{T} \models \tilde{\varphi}_{\mathrm{R}}$ and we are done. Otherwise, $\mathcal{T}$ is both $\tilde{\sigma}_k$-consistent and $\tilde{\sigma}_{\mathrm{R},k}$-consistent, which is enough to conclude that it is a consistent $\tilde{\sigma}''_k$-tree. Hence, by Theorem 13, $\mathfrak{D}(\mathcal{T}) \models \varphi_{\mathrm{L}}$. By our initial assumption that $\varphi_{\mathrm{L}} \models \varphi_{\mathrm{R}}$, this means $\mathfrak{D}(\mathcal{T}) \models \varphi_{\mathrm{R}}$. By Theorem 13, this means that $\mathcal{T} \models \varphi_{\mathrm{R}}{}^{\rightarrow}$, so by weakening, $\mathcal{T} \models \gamma_{\sigma_{\mathrm{R}},k} \rightarrow \varphi_{\mathrm{R}}{}^{\rightarrow}$. This means that $\mathcal{T} \models \tilde{\varphi}_{\mathrm{R}}$.

Now we claim that $\tilde{\varphi}_{\mathrm{L}} \models \tilde{\varphi}_{\mathrm{R}}$ over all $\tilde{\sigma}''_k$-structures. Assume not. Then there is some $\tilde{\sigma}''_k$-structure $\mathfrak{A}$ such that $\mathfrak{A} \models \tilde{\varphi}_{\mathrm{L}} \wedge \neg\tilde{\varphi}_{\mathrm{R}}$. But by the tree-model property of L_μ [18], this means there is some $\tilde{\sigma}''_k$-tree $\mathcal{T}$ such that $\mathcal{T} \models \tilde{\varphi}_{\mathrm{L}} \wedge \neg\tilde{\varphi}_{\mathrm{R}}$, which contradicts the previous paragraph. This means that $\tilde{\varphi}_{\mathrm{L}} \models \tilde{\varphi}_{\mathrm{R}}$ over all $\tilde{\sigma}''_k$-structures.

Since $\tilde{\varphi}_{\mathrm{L}} \models \tilde{\varphi}_{\mathrm{R}}$ we can apply Theorem 8 to see that $\theta \models \tilde{\varphi}_{\mathrm{R}}$ (over all $\tilde{\sigma}_{\mathrm{R},k}$-structures).

Now we are ready to show that $\chi \models \varphi_{\mathrm{R}}$. If $\mathfrak{B} \models \theta^{\leftarrow}$ for a σ_{R}-structure $\mathfrak{B}$, then $\mathcal{S}^k(\mathfrak{B}) \models \theta$ by Theorem 17. By the previous paragraph, this implies that $\mathcal{S}^k(\mathfrak{B}) \models \tilde{\varphi}_{\mathrm{R}}$ and hence $\mathcal{S}^k(\mathfrak{B}) \models \gamma_{\sigma_{\mathrm{R}},k} \rightarrow \varphi_{\mathrm{R}}{}^{\rightarrow}$. But this means that $\mathfrak{D}(\mathcal{S}^k(\mathfrak{B})) \models \varphi_{\mathrm{R}}$ by Theorem 17. Since $\mathfrak{B}$ and $\mathfrak{D}(\mathcal{S}^k(\mathfrak{B}))$ are $\mathrm{UN}^k[\sigma_{\mathrm{R}}]$-bisimilar, this means that $\mathfrak{B} \models \varphi_{\mathrm{R}}$ as desired.

This completes the proof of uniform interpolation for UNFP^k, without the elementary bounds. We will come back to this technique using the naïve forward mapping in order to prove Theorem 24.

IV. Refining and Extending the Construction

A. Obtaining elementary bounds

The prior approach using the naïve forward mapping yields a non-elementary upper bound on the size of the uniform interpolant and the time complexity of constructing it. This is due to the fact that we pass through MSO, and converting from MSO to L_μ can result in a non-elementary blow-up [21].

We improve the complexity of this uniform interpolation algorithm in this section by avoiding GSO and MSO, and using automata on trees instead (see Figure 2).

$$\begin{tikzcd}
\textbf{Relational structures} & \textbf{Coded structures} & \textbf{Coded structures} \\
\mathrm{UNFP}^k \arrow[r, "(1)"] & \text{2-way } \mu\text{-aut.} \arrow[r, "(2)"] & \text{well-struct. } \mathrm{L}_\mu \arrow[d, "(3)"] \\
\mathrm{UNFP}^k \text{ over subsignature} & & \text{well-struct. } \mathrm{L}_\mu \text{ over subsignature encoding} \arrow[ll, "(4)"]
\end{tikzcd}$$

Fig. 2. Proof structure for elementary uniform interpolation

The automata that we use are designed to work on trees with arbitrary branching, so they cannot refer to specific children of a node. This is different than traditional tree automata on binary trees that can refer to the left child and right child. To start, we also allow 2-way movement.

Formally, a *2-way alternating μ-automaton* $\mathcal{A}$ is a tuple $\langle \Sigma_p, \Sigma_a, Q_E, Q_A, q_0, \delta, \Omega \rangle$ where Σ_p and Σ_a are finite alphabets of propositions and actions, $Q := Q_E \cup Q_A$ is a finite set of states partitioned into states Q_E controlled by Eve and states Q_A controlled by Adam, and $q_0 \in Q$ is the initial state. The transition function has the form $\delta : Q \times \Sigma_p \rightarrow \mathcal{P}(\mathrm{Dir} \times \Sigma_a \times Q)$ where $\mathrm{Dir} = \{\uparrow, 0, \downarrow\}$ is the set of possible directions (up $\uparrow$, stay 0, down $\downarrow$). The acceptance condition is a parity condition specified by $\Omega : Q \rightarrow \mathrm{Pri}$, which maps each state to a priority in a finite set of priorities Pri.

We view $\mathcal{A}$ running on a tree $\mathcal{T}$ starting at node $v_0 \in \mathrm{dom}(\mathcal{T})$ as a game $\mathcal{G}(\mathcal{A}, \mathcal{T}, v_0)$. The arena is $Q \times \mathrm{dom}(\mathcal{T})$, and the initial position is (q_0, v_0). From a position (q, v) with $q \in Q_E$ (respectively, $q \in Q_A$), Eve (respectively Adam) selects $(d, a, r) \in \delta(q, \mathcal{T}(v))$. If $d = 0$, then the game continues from position (r, v). Otherwise, Eve (respectively, Adam) selects an a-neighbor w of v in direction d and the game continues from position (r, w).

A *play* in $\mathcal{G}(\mathcal{A}, \mathcal{T}, v_0)$ is a sequence $(q_0, v_0), (q_1, v_1), (q_2, v_2), \ldots$ of moves in the game. Such a *play is winning* for Eve if the *parity condition* is satisfied: the maximum priority that occurs infinitely often in $\Omega(q_0), \Omega(q_1), \ldots$ is even.

A *strategy* for one of the players is a function that returns the next choice for that player given the history of the play. Choosing a strategy for both players fixes a play in $\mathcal{G}(\mathcal{A}, \mathcal{T}, v_0)$. A play π is *compatible* with a strategy ζ if there is a strategy ζ' for the other player such that ζ and ζ' yield π. A *strategy is winning* for Eve if every play compatible with it is winning.

We write $L(\mathcal{A})$ for the set of trees $\mathcal{T}$ such that Eve has a winning strategy in $\mathcal{G}(\mathcal{A}, \mathcal{T}, \epsilon)$, where ϵ is the root of $\mathcal{T}$.

The idea is to redo the forward mapping, but now go directly from UNFP^k to a 2-way alternating μ-automaton. For brevity in the following theorem (and the rest of this section), we give only bounds on the output size, not the running time of the algorithm. However the proofs will show that the worst-case running time is bounded by a polynomial in the output size.

Theorem 20 (UNFP$^k[\sigma]$ to 2-way alternating μ-automata)**.** *Let ψ be a sentence in* UNFP$^k[\sigma]$*. We can construct a 2-way alternating μ-automaton $\mathcal{A}_\psi$ such that for all consistent $\tilde{\sigma}_k$ trees $\mathcal{T}$, $\mathfrak{D}(\mathcal{T}) \models \psi$ iff $\mathcal{T} \models L(\mathcal{A}_\psi)$ and the size of $\mathcal{A}_\psi$ is doubly exponential in $|\psi|$, but the number of states and priorities of $\mathcal{A}_\psi$ is at most singly exponential in $|\psi|$.*

Proof sketch: We need to test whether ψ holds in the decoding of the input tree $\mathcal{T}$. We construct the automaton by induction on the structure of ψ. The idea behind the construction is to allow Eve to guess an annotation of $\mathcal{T}$ with information about which unary subformulas of ψ hold, and then run an automaton that checks ψ with the help of these annotations. In order to prevent Eve from cheating with her guesses about the subformulas, Adam is also allowed to launch automata that check Eve's claims about the subformulas.

Likewise, testing $\mathfrak{D}(\mathcal{T}) \models [\mathbf{lfp}\, Y, y.\chi(y, Y)](a)$ can be viewed as a game between Adam and Eve which starts with $y = a$ and proceeds as follows:

- Eve chooses some valuation for Y such that $\mathfrak{D}(\mathcal{T}) \models \chi(y, Y)$ (she loses if this is not possible), then
- Adam chooses some new $y \in Y$ (he loses if this is not possible), and then the game proceeds to the next turn.

If the game never terminates then Adam is declared the winner. We can implement this game as an automaton running on $\mathcal{T}$, where Eve guesses an annotation of $\mathcal{T}$ with the valuation of Y in the current round, and then simulates the inductively-defined automaton checking $\chi(y, Y)$. Adam can challenge any b in the set Y chosen by Eve by launching another copy of the automaton checking χ starting from the node carrying b. Correctness is enforced by using the parity condition: an odd priority is used if Adam challenges a **lfp** fixpoint. ∎

The new simplification step then converts this 2-way alternating μ-automaton to a well-structured L$_\mu$-formula.

Theorem 21 (2-way μ-automata to well-structured L$_\mu[\tilde{\sigma}_k]$)**.** *Let $\mathcal{A}$ be a 2-way alternating μ-automaton on $\tilde{\sigma}_k$ trees. We can construct a well-structured* L$_\mu[\tilde{\sigma}_k]$*-formula ψ such that for all consistent $\tilde{\sigma}_k$ trees $\mathcal{T}$, $\mathfrak{D}(\mathcal{T}) \models \psi$ iff $\mathcal{T} \in L(\mathcal{A})$. The size of ψ is at most $|\mathcal{A}|^{f(m)}$ where m is the number of states and priorities of $\mathcal{A}$ and f is a polynomial function independent of $\mathcal{A}$.*

This is accomplished by converting to a μ-automaton [20] — which can be seen as the 1-way nondeterministic counterpart to the 2-way alternating automata described earlier — using a variation of a construction in [22].

Uniform interpolation of well-structured L$_\mu$-formulas can be done without a blow-up in the size of the formula.

Theorem 22. *Let $\psi \in$ L$_\mu[\tilde{\sigma}]$ be a well-structured L$_\mu$-formula. Let $\tilde{\sigma}'$ be a subsignature of $\tilde{\sigma}$. We can construct a well-structured $\theta \in$ L$_\mu[\tilde{\sigma}']$ such that θ is a uniform interpolant for ψ and $\tilde{\sigma}'$ and is of size at most $|\psi|$.*

As with the previous step, this is also accomplished using μ-automata. Finding the uniform interpolant corresponds to taking a projection of the automaton language based on the (encoding) of the desired subsignature. Indeed, this is how uniform interpolation for μ-calculus is proven (see [8, Theorem 3.3 and Corollarly 3.4]). This does not increase the size of the automaton, or the corresponding well-structured L$_\mu$-formula.

Finally, the backward step is now from well-structured L$_\mu$ to UNFP^k.

Theorem 23 (well-structured L$_\mu[\tilde{\sigma}_k]$ to UNFP$^k[\sigma]$)**.** *Let $\psi \in$ UNFP$^k[\tilde{\sigma}_k]$ be well-structured. We can construct a DAG-representation of a sentence $\psi^{\leftarrow} \in$ UNFP$^k[\sigma]$ such that for all σ-structures $\mathfrak{B}$, $\mathfrak{B} \models \psi^{\leftarrow}$ iff $\mathcal{S}^k(\mathfrak{B}) \models \psi$, and the size of the DAG-representation of $\psi^{\leftarrow}$ is polynomial in the size of ψ.*

The proof of the backward mapping step is actually the same as before, noting that the conversion starting from a well-structured L$_\mu$-formula can be done in polynomial time.

We now summarize this improved uniform interpolation algorithm (see Figure 2). Let $\varphi \in$ UNFP$^k[\sigma]$, and let $\sigma' \subseteq \sigma$ be the target subsignature for the uniform interpolant.

1) *Forward mapping step*: Apply Theorem 20 to obtain a 2-way alternating μ-automaton $\mathcal{A}$ for φ over $\tilde{\sigma}_k$-trees, of size doubly exponential in $|\varphi|$, but with the number of states and priorities only singly exponential in $|\varphi|$.
2) *Simplification step*: Apply Theorem 21 to obtain a well-structured L$_\mu[\tilde{\sigma}_k]$-formula ψ equivalent to $\mathcal{A}$. Note that ψ is of size doubly exponential in $|\varphi|$, since the exponential blow-up in this step is only relative to the number of states and priorities in $\mathcal{A}$.
3) *Interpolation step*: Apply Theorem 22 to ψ and $\tilde{\sigma}'_k$ to obtain a well-structured L$_\mu[\tilde{\sigma}'_k]$-formula θ. This is at most the size of ψ, so it is still doubly exponential in $|\varphi|$.
4) *Backward mapping step*: Obtain a DAG-representation of the UNFP$^k[\sigma']$ formula $\theta^{\leftarrow}$ equivalent to θ using Theorem 23. The DAG-representation is at most doubly exponential in $|\varphi|$. Set $\chi := \theta^{\leftarrow}$.

Overall, this means that the uniform interpolant χ has a DAG-representation that is at most doubly exponential in the size of φ, as stated in Theorem 3. As mentioned earlier, because the time complexity is bounded by the size of the output at each stage, this algorithm also yields the 2EXPTIME bound on uniform interpolation for UNFP^k.

B. Extension for unary formulas, constants, and equality

The uniform interpolation results can be easily extended to handle formulas with at most one free variable, instead of sentences. This requires only a slight change to the definition of consistent tree to allow the root to be a non-empty interface node, representing the free variable.

In order to handle signatures σ with constants, we must modify the encodings of the tree decompositions so the constants from σ are represented at every node in the tree encodings (so in particular, interface nodes contain every constant, plus at most one additional element). This comes from the fact that in order to have a connection between the UN$^k[\sigma]$-bisimulation game and satisfaction of UNFP$^k[\sigma]$

formulas with constants (as required for Proposition 7), we need information about constants at every position in the game.

Note that these constants do not adversely impact the forward and backward mapping. In particular, the constants do not lead to any difficulties in the backward mapping from L_μ to UNFP because constants can appear freely in UNFP-formulas. Indeed, as long as some formula ψ has at most one free variable, we can negate ψ regardless of the number of constants it mentions.

It is important to note that in our uniform interpolation result, the uniform interpolant may make use of any constants from the original signature. That is, although the uniform interpolant can restrict the relations that are used, it cannot restrict the constants that are used. This is unavoidable, at least for interpolation over formulas. We explain this more in the full version of this paper.

For equality, we must first convert formulas with equality to an equality normal form with very restricted uses of equality. This allows us to use the algorithms presented earlier, treating equality like any other relation. We provide more details on this equality extension in the full version of this paper.

C. *Characterization of* UNFP^k *as* UN^k*-bisimulation invariant fragment of* GSO

As a by-product of our translations, we can conclude that UNFP^k is the UN^k-bisimulation invariant fragment of GSO. This is in analogy to the fact that L_μ is the bisimulation invariant fragment of MSO (see Theorem 6), and that GFP is the "guarded bisimulation" invariant fragment of GSO [11].

Theorem 24 (UNFP^k $\equiv$ UN^k-bisimulation invariant GSO)**.** *Every sentence ψ in* GSO$[\sigma]$ *that is invariant under* UN$^k[\sigma]$*-bisimulation is effectively equivalent to a sentence φ in* UNFP$^k[\sigma]$.

Proof: If we start from a UNFP^k-sentence, then we can translate into UNFP^k-bismilar GSO (even MSO), using Propositions 5 and 7. For the other direction, fix some UN$^k[\sigma]$-bisimulation invariant sentence ψ in GSO$[\sigma]$.

1) *Naïve forward mapping step*: Obtain $\psi^\rightarrow$ in MSO$[\tilde{\sigma}_k]$ from ψ using Theorem 13. By Corollary 14, $\psi^\rightarrow$ is $\tilde{\sigma}_k$-bisimulation invariant on consistent $\tilde{\sigma}_k$-trees.
2) *Simplification step*: Let γ be an FO$[\tilde{\sigma}_k]$ sentence expressing the conditions for consistent trees, obtained using Lemma 9. Then $(\gamma \rightarrow \psi^\rightarrow)$ in MSO$[\tilde{\sigma}_k]$ is $\tilde{\sigma}_k$-bisimulation invariant within the class of all $\tilde{\sigma}_k$-trees, so we can obtain φ in $L_\mu[\tilde{\sigma}_k]$ from $(\gamma \rightarrow \psi^\rightarrow)$ using Theorem 6.
3) *Backward mapping step*: Obtain $\varphi^\leftarrow$ in UNFP$^k[\sigma]$ from φ using Theorem 17.

We must show that $\varphi^\leftarrow$ is the UNFP$^k[\sigma]$-sentence equivalent to ψ. Consider some σ-structure $\mathfrak{B}$. Assume that $\mathfrak{B} \models \psi$. Then $\mathfrak{D}(\mathcal{S}^k(\mathfrak{B})) \models \psi$ by Lemma 12 and UN$^k[\sigma]$-bisimulation invariance of ψ. Using Theorem 13, this means $\mathcal{S}^k(\mathfrak{B}) \models \psi^\rightarrow$. Moreover, $\mathcal{S}^k(\mathfrak{B}) \models \gamma \rightarrow \psi^\rightarrow$ since $\mathcal{S}^k(\mathfrak{B})$ is a consistent $\tilde{\sigma}_k$-tree. so $\mathcal{S}^k(\mathfrak{B}) \models \varphi$. Finally, by Theorem 17, $\mathfrak{B} \models \varphi^\leftarrow$ as desired. Similar reasoning shows that $\mathfrak{B} \models \varphi^\leftarrow$ implies $\mathfrak{B} \models \psi$. ∎

V. Failure of interpolation

In this section we will see that some natural extensions and variants of our main interpolation theorems fail.

A. *Failure of uniform interpolation*

Although we have shown that UNFP has Craig interpolation, it fails to have uniform interpolation.

Proposition 25. *Uniform interpolation fails for* UNFP*. In particular, there is a* UNF *antecedent with no uniform interpolant in* LFP*, even when the consequents are restricted to sentences in* UNF*. The variant of uniform interpolation where entailment is considered only over finite structures also fails for* UNFP.

Proof: There is a UNF sentence φ that expresses that unary relations R,G,B form a 3-coloring of a graph with edge relation E. Consider a uniform interpolant θ (in any logic) for the UNF sentence φ with respect to its UNF-consequences in the signature $\{E\}$. We claim that there cannot be an LFP formula equivalent to θ.

For all finite graphs G that are not 3-colorable, let ψ_G be the UNF sentence corresponding to the canonical conjunctive query of G over relation E — that is, if G consists of edges E mentioning vertices $v_1 \dots v_n$, ψ_G is $\exists v_1 \dots v_n \bigwedge_{e \in G, e=(v_i,v_j)} E(v_i, v_j)$. Then φ must entail $\neg\psi_G$, since the 3-coloring of a graph G' satisfying ψ_G is easily seen to induce a 3-coloring on G.

Now consider a finite graph G. If G is 3-colorable, then $G \models \varphi$, and hence $G \models \theta$. On the other hand, if G is not 3-colorable, then $G \models \psi_G$, so $G \models \neg\theta$ because φ entails $\neg\psi_G$ and thus, by the assumption on θ, θ entails $\neg\psi_G$. Therefore, θ holds in G iff G is 3-colorable.

Dawar [23] showed that 3-colorability is not expressible in the infinitary logic $\mathcal{L}^\omega_{\infty\omega}$ over finite structures. Since LFP can be translated into $\mathcal{L}^\omega_{\infty\omega}$ over finite structures, this implies that θ cannot be in LFP.

The above argument only makes use of the properties of θ over finite structures, and thus demonstrates the failure of the variant of uniform interpolation in the finite. ∎

Recall that we have trivial uniform interpolants in existential second-order logic, i.e. in NP. The previous arguments shows that interpolants for UNFP express NP-hard problems, and thus cannot be in any PTime language if PTime is not equal to NP. We remark that one could still hope to find uniform interpolants for UNFP by allowing the interpolants to live in a larger fragment that is still "tame", but we leave this as an open question.

Uniform interpolation also fails for GSO.

Proposition 26. *Uniform interpolation fails for* GSO*. In particular, there is a* GF *antecedent with no uniform interpolant in* GSO*, even when the consequents are restricted to sentences of* GF *(or* UNF*) of width 2.*

Proof sketch: Consider $\sigma = \{P, Q, R_1, R_2, S\}$, and let $\varphi \in$ GSO$[\sigma]$ be

$$\forall z.[Qz \rightarrow \exists xy.(Sxy \wedge R_1zx \wedge R_2zy)] \wedge$$
$$\forall xy.[Sxy \rightarrow \exists x'y'.(Sx'y' \wedge R_1xx' \wedge R_2yy' \wedge ((Px' \wedge Py') \vee (\neg Px' \wedge \neg Py')))]$$

which expresses that there is an infinite "ladder" starting at every Q-node (where S connects pairs of elements on the same rung, and R_i connects corresponding elements on different rungs) and the pair of elements on each rung agree on P. By adding an additional dummy guard G (of arity 4), we can write a GF sentence that implies the same property.

Then for each n, we can define over $\sigma' = \{P, Q, R_1, R_2\}$ a formula $\psi_n \in \mathrm{GSO}[\sigma']$,

$$[\exists x.(Qx \wedge \forall x_1 \dots x_n((\bigwedge_i R_1 x_i x_{i+1} \wedge x_1 = x) \rightarrow Px_n))] \rightarrow$$
$$[\exists y(Qy \wedge \exists y_1 \dots y_n(\bigwedge_i R_2 y_i y_{i+1} \wedge y_1 = y \wedge Py_n))]$$

which expresses that if there is some Q-position x such that every R_1-path of length n from x ends in a position satisfying P, then there is an R_2-path of length n from some Q-position y that ends in a position satisfying P. Note that for all n, $\varphi \models \psi_n$. Moreover, ψ_n can also be expressed in either GF or UNF of width 2.

Assume for the sake of contradiction that there is some uniform interpolant θ in $\mathrm{GSO}[\sigma']$. Since GSO coincides with MSO over trees ([24], as cited in [11]), this means we can construct a nondeterministic parity tree automaton $\mathcal{A}$ that recognizes precisely the language of tree structures satisfying θ. A pumping argument can then be used to construct a tree structure accepted by this automaton, and hence satisfying θ, that fails to satisfy some ψ_n. This contradicts $\theta \models \psi_n$. ■

It has been known for some time that GF fails to have even ordinary Craig interpolation [25], and hence fails to have uniform interpolation. The previous proposition shows that we cannot get uniform interpolants for GF even when we allow the uniform interpolants to come from GSO.

B. *Failure of Craig interpolation for* GNFP

It is natural to try to extend our results to the logic GNFP. Unfortunately, Craig interpolation fails for GNFP.

Proposition 27. *Craig interpolation fails for* GNFP. *In particular, there is an entailment of* GFP *sentences with no* GNFP *interpolant, even over finite structures.*

Proof sketch: Let φ' be

$$[\mathbf{lfp}\, X, xy.Gxyy \wedge (Rxy \vee \exists y'.(Gxy'y \wedge Rxy' \wedge Xy'y)](xx)$$

and let φ be $\forall x.(Qx \rightarrow \varphi')$. This sentence $\varphi \in \mathrm{GFP}[\sigma]$ over the signature $\sigma = \{G, Q, R\}$ implies that there is an R-loop from every Q-labelled element. Define the $\mathrm{GFP}[\sigma']$ sentence ψ over signature $\sigma' = \{P, Q, R\}$ to be

$$\forall x.((Qx \wedge Px) \rightarrow [\mathbf{lfp}\, X, x.\exists y.(Rxy \wedge (Py \vee Xy))](x))$$

which expresses that for all Q and P labelled elements x, there is an R-path from x leading to some node y with Py.

Now $\varphi \models \psi$, but it can be shown that any GNFP sentence θ over the common signature $\{Q, R\}$ cannot distinguish between a structure consisting of a single loop satisfying φ and ψ, and a structure built from "lassos" of sufficient size (depending only on the width of θ) where φ and ψ fail to hold. Hence, there can be no GNFP interpolant. ■

VI. Conclusion

In this work we have extended the "bootstrapping from modal logic" technique of [11] to show that the logic UNFP has interpolation. We also have shown that the richer logics, such as guarded negation fixpoint, do not have interpolation. It is still possible that there is a decidable logic containing GNFP which has interpolation. We also leave open the question of whether interpolation holds over finite structures for UNFP.

Acknowledgment

Benedikt and Vanden Boom are supported by EPSRC grants EP/H017690/1 and EP/L012138/1. ten Cate is supported by NSF grant IIS-1217869.

References

[1] B. ten Cate, E. Franconi, and I. Seylan, "Beth definability in expressive description logics," in *IJCAI*, 2011.
[2] A. Nash, L. Segoufin, and V. Vianu, "Views and queries: Determinacy and rewriting," *ACM TODS*, vol. 35, no. 3, 2010.
[3] M. Marx, "Queries determined by views: pack your views," in *PODS*, 2007.
[4] K. L. McMillan, "Applications of Craig Interpolation to model checking," in *CSL*, 2004.
[5] W. Craig, "Linear reasoning. A new form of the Herbrand-Gentzen theorem," *The Journal of Symbolic Logic*, vol. 22, no. 03, pp. 250–268, 1957.
[6] M. Benedikt, B. ten Cate, and M. Vanden Boom, "Effective interpolation and preservation in guarded logics," in *CSL-LICS*, 2014.
[7] L. Henkin, "An extension of the Craig-Lyndon interpolation theorem," *The Journal of Symbolic Logic*, vol. 28, no. 3, pp. 201–216, 1963.
[8] G. D'Agostino and M. Hollenberg, "Logical Questions Concerning The mu-Calculus: Interpolation, Lyndon and Los-Tarski," *The Journal of Symbolic Logic*, vol. 65, no. 1, pp. 310–332, 2000.
[9] C. Lutz, I. Seylan, and F. Wolter, "An automata-theoretic approach to uniform interpolation and approximation in the description logic EL," in *KR*, 2012.
[10] P. Koopmann and R. Schmidt, "Count and forget: Uniform interpolation of SHQ-ontologies," in *IJCAR*, 2014.
[11] E. Grädel, C. Hirsch, and M. Otto, "Back and forth between guarded and modal logics," *ACM TOCL*, vol. 3, no. 3, pp. 418–463, 2002.
[12] D. Janin and I. Walukiewicz, "On the expressive completeness of the propositional mu-calculus with respect to monadic second order logic," in *CONCUR*, 1996.
[13] B. ten Cate and L. Segoufin, "Unary negation," in *STACS*, 2011.
[14] A. Arnold and D. Niwiński, *Rudiments of mu-calculus*. North Holland, 2001.
[15] V. Bárány, B. ten Cate, and L. Segoufin, "Guarded negation," in *ICALP*, 2011.
[16] G. D. Giacomo, "Decidability of class-based knowledge representation formalisms," Ph.D. dissertation, Universitá di Roma "La Sapienza", 1995.
[17] U. Sattler and M. Y. Vardi, "The hybrid μ-calculus," in *IJCAR*, 2001.
[18] J. Bradfield and C. Stirling, "Modal mu-calculi," in *Handbook of Modal Logic*. Elsevier, 2007, pp. 721–756.
[19] H. Andréka, I. Németi, and J. van Benthem, "Modal languages and bounded fragments of predicate logic," *Journal of Philosophical Logic*, vol. 27, no. 3, pp. 217–274, 1998.
[20] D. Janin and I. Walukiewicz, "Automata for the modal mu-calculus and related results," in *MFCS*, 1995.
[21] L. J. Stockmeyer, "The complexity of decision problems in automata theory and logic," Ph.D. dissertation, Department of Electrical Engineering, MIT, 1974.
[22] M. Y. Vardi, "Reasoning about the past with two-way automata," in *ICALP*, 1998.
[23] A. Dawar, "A restricted second order logic for finite structures," *Inf. Comput.*, vol. 143, no. 2, pp. 154–174, 1998.
[24] B. Courcelle, "The expression of graph properties and graph transformations in monadic second-order logic," in *Handbook of Graph Grammars and Computing by Graph Transformations*, vol. 1, 1997, pp. 313–400.
[25] E. Hoogland, M. Marx, and M. Otto, "Beth definability for the guarded fragment," in *LPAR*, 1999.