A Semantic Approach to Interpolation[⋆]

Andrei Popescu[⋆⋆], Traian Florin Şerbănuţă[⋆⋆⋆], and Grigore Roşu

Department of Computer Science,
University of Illinois at Urbana-Champaign
{popescu2, tserban2, grosu}@cs.uiuc.edu

Abstract. Interpolation results are investigated for various types of formulae. By shifting the focus from syntactic to *semantic interpolation*, we generate, prove and classify a series of interpolation results for first-order logic. A few of these results non-trivially generalize known interpolation results. All the others are new.

1 Introduction

Craig interpolation is a landmark result in first-order logic [7]. In its original formulation, it says that given sentences Γ_1 and Γ_2 such that $\Gamma_1 \vdash \Gamma_2$, there is some sentence Γ whose non-logical symbols occur in *both* Γ_1 and Γ_2, called an *interpolant*, such that $\Gamma_1 \vdash \Gamma$ and $\Gamma \vdash \Gamma_2$. This well-known result can also be rephrased as follows: given first-order signatures Σ_1 and Σ_2, a Σ_1-sentence Γ_1 and a Σ_2-sentence Γ_2 such that $\Gamma_1 \models_{\Sigma_1 \cup \Sigma_2} \Gamma_2$, there is some $(\Sigma_1 \cap \Sigma_2)$-sentence Γ such that $\Gamma_1 \models_{\Sigma_1} \Gamma$ and $\Gamma \models_{\Sigma_2} \Gamma_2$.

One naturally looks for this property in logical systems other than first-order logic. The conclusion of studying various extensions of first-order logic was that "interpolation is indeed [a] rare [property in logical systems]" ([2], page 68). We are going to show in this paper that the situation is totally different when one looks in the opposite direction, at restrictions of first-order logic. There are simple logics, such as equational logic, where the interpolation result does *not* hold for sentences, but it holds for *sets of sentences* [27]. For this reason, as well as for reasons coming from theoretical software engineering, in particular from specification theory and modularization [3, 13, 14, 9], it is quite common today to state interpolation more generally, in terms of sets of sentences Γ_1, Γ_2, and Γ. This is also the approach that we follow in this paper.

We call our approach to interpolation "semantic" because we shift the problem of finding syntactic interpolants Γ to a problem of finding appropriate *classes*

[⋆] Supported by NSF grants CCF-0234524, CCF-044851, CNS-0509321.
[⋆⋆] Also: Institute of Mathematics "Simion Stoilow" of the Romanian Academy, Bucharest; and Fundamentals of Computer Science, Faculty of Mathematics, University of Bucharest.
[⋆⋆⋆] Also: Fundamentals of Computer Science, Faculty of Mathematics, University of Bucharest.

of models, which we call *semantic interpolants.* We present a precise characterization for *all* the semantic interpolants of a given instance $\Gamma_1 \models_{\Sigma_1 \cup \Sigma_2} \Gamma_2$, as well as a general theorem ensuring the existence of semantic interpolants closed under generic closure operators. Not all semantic interpolants correspond to sets of sentences. However, when semantic interpolants are closed under certain operators, they become *axiomatizable*, thus corresponding to some sets of sentences. Following the nice idea of using Birkhoff-like axiomatizability to prove the Craig interpolation for equational logics in [27], a similar semantic approach was investigated in [25], but it was only applied there to obtain Craig interpolation results for categorical generalizations of equational logics. A similar idea is exploited in [9], where interpolation results are presented in an institutional [17] setting. While the institution-independent interpolation results in [9] can potentially be applied to various particular logics, their instances still refer to just one type of sentence: the one the particular logic comes with.

The conceptual novelty of our semantic approach to interpolation in this paper is to keep the restrictions on Γ_1, Γ_2, and Γ, or more precisely the ones on their corresponding classes of models, *independent.* This way, surprising and interesting results can be obtained with respect to the three types of sentences involved. By considering several combinations of closure operators allowed by our parametric semantic interpolation theorem, we provide many interpolation results;[1] some of them generalize known results, but most of them are new. For example, we show that if the sentences in Γ_1 are first-order while the ones in Γ_2 are universally quantified Horn clauses (UHC's), then those in the interpolant Γ can be chosen to be UHC's too. Surprisingly, sometimes the interpolant is strictly simpler than Γ_1 and Γ_2. For example, we show that the following choices of the type of sentences in the interpolant Γ are possible (see also the table on page 316, lines 6, 13 and 20):

– Γ_1-universal and Γ_2-positive (i.e., contains only formulae without negations) imply that Γ consists only of universally quantified disjunction of atoms;
– Γ_1-UHC's and Γ_2-positive imply that Γ has only universally quantified atoms;
– Γ_1- finitary formulae and Γ_2- infinitary universally quantified disjunctions of atoms imply Γ- (finitary) universally quantified disjunctions of atoms.

Some Motivation. Craig interpolation has applications in various areas of computer science. Such an area is formal specification theory (see [19, 14]). For structured specifications [3, 29], interpolation ensures a good, compositional, behavior of their semantics [3, 5, 25]. In choosing a logical framework for specifications, one has to find the right balance between expressive power and amenable computational aspects. Therefore, an intermediate choice between the "extremes", full first-order logic and equational logic, might be desirable. We enable (at least partially) such intermediate logics (e.g., the *positive*- or $(\forall\vee)$- logic) as specification frameworks, by showing that they have the interpolation property. Moreover, the

[1] Other results obtained with this technique, including some for second-order and higher-order logic, can be found in the technical report [24].

very general nature of our results w.r.t. signature morphisms sometimes allows one to enrich the class of morphisms used for renaming usually up to arbitrary morphisms, freeing specifications from unnatural constraints, like injectivity of renaming/translation. Some technical details about the applications of our results to formal specifications can be found in Section 5.

Automatic reasoning is another area where interpolation is important and where our results contribute. There, *putting theories together* while still taking advantage, inside their union language, of their available decision procedures [21, 23], relies on interpolation in a crucial way. Moreover, interpolation provides a heuristic to "divide and conquer" a proving task: in order to show $\Gamma_1 \models_{\Sigma_1 \cup \Sigma_2} \Gamma_2$, find some Γ over the syntax $\Sigma_1 \cap \Sigma_2$ and prove the two "simpler" tasks $\Gamma_1 \models_{\Sigma_1} \Gamma$ and $\Gamma \models_{\Sigma_2} \Gamma_2$. For some simpler sub-logics of first-order logic, such as propositional calculus, where there is a finite set of semantically different sentences over any given signature, one can use interpolation also as a disproof technique: if for each $(\Sigma_1 \cap \Sigma_2)$-sentence Γ (there is only a finite number of them) at least one of $\Gamma_1 \models_{\Sigma_1} \Gamma$ or $\Gamma \models_{\Sigma_2} \Gamma_2$ fails, then $\Gamma_1 \models_{\Sigma_1 \cup \Sigma_2} \Gamma_2$ fails. The results of the present paper, although not effectively constructing interpolants, provide information about the existence of interpolants *of a certain type*, helping reducing the space of search. For instance, according to one of the cases of our main result, Theorem 2, the existence of a positive interpolant Γ is ensured by the fact that *either* one of Γ_1 or Γ_2 is positive (lines 2, 3 of table on page 316).

Technical Preliminaries. For simplifying the exposition, set-theoretical foundational issues are ignored in this paper.[2] Given a class $\mathcal{D}$, let $\mathcal{P}(\mathcal{D})$ denote the collection of all subclasses of $\mathcal{D}$. For any $\mathcal{C} \in \mathcal{P}(\mathcal{D})$, let $\overline{\mathcal{C}}$ denote $\mathcal{D} \setminus \mathcal{C}$, that is, the class of all elements in $\mathcal{D}$ which are not in $\mathcal{C}$. Also, given $\mathcal{C}_1, \mathcal{C}_2 \in \mathcal{P}(\mathcal{D})$ let $[\mathcal{C}_1, \mathcal{C}_2]$ denote all classes $\mathcal{C}$ which include $\mathcal{C}_1$ and are included in $\mathcal{C}_2$.

An *operator* on $\mathcal{D}$ is a mapping $F : \mathcal{P}(\mathcal{D}) \to \mathcal{P}(\mathcal{D})$. Let $Id_{\mathcal{D}}$ denote the identity operator. For any operator F on $\mathcal{D}$, let $Fixed(F)$ denote the collection of all *fixed points* of F, that is, $\mathcal{C} \in Fixed(F)$ iff $F(\mathcal{C}) = \mathcal{C}$. An operator F on $\mathcal{D}$ is a *closure operator* iff it is *extensive* ($\mathcal{C} \subseteq F(\mathcal{C})$), *monotone* (if $\mathcal{C}_1 \subseteq \mathcal{C}_2$ then $F(\mathcal{C}_1) \subseteq F(\mathcal{C}_2)$) and *idempotent* ($F(F(\mathcal{C})) = F(\mathcal{C})$).

Given a relation $\mathcal{R}$ on $\mathcal{D}$, let $\mathcal{R}$ also denote the operator on $\mathcal{D}$ associated with $\mathcal{R}$, assigning to each $\mathcal{C} \in \mathcal{P}(\mathcal{D})$ the class of all elements from $\mathcal{D}$ in relation with elements in $\mathcal{C}$, that is, $\mathcal{R}(\mathcal{C}) = \{c' \in \mathcal{D} \mid (\exists c \in \mathcal{C})\ c\ \mathcal{R}\ c'\}$. Notice that the operator associated to a reflexive and transitive relation is a closure operator.

Given two classes $\mathcal{C}$ and $\mathcal{D}$ and a mapping $\mathcal{U} : \mathcal{C} \to \mathcal{D}$, we let $\mathcal{U}$ also denote the mapping $\mathcal{U} : \mathcal{P}(\mathcal{C}) \to \mathcal{P}(\mathcal{D})$ defined by $\mathcal{U}(\mathcal{C}') = \{\mathcal{U}(c) \mid c \in \mathcal{C}'\}$ for any $\mathcal{C}' \in \mathcal{P}(\mathcal{C})$. Also, we let $\mathcal{U}^{-1} : \mathcal{P}(\mathcal{D}) \to \mathcal{P}(\mathcal{C})$ denote the mapping defined by $\mathcal{U}^{-1}(\mathcal{D}') = \{c \in \mathcal{C} \mid \mathcal{U}(c) \in \mathcal{D}'\}$ for any $\mathcal{D}' \in \mathcal{P}(\mathcal{D})$. Given two mappings $U, V : \mathcal{P}(\mathcal{C}) \to \mathcal{P}(\mathcal{D})$, we say that U *is included in* V, written $U \sqsubseteq V$, iff $U(\mathcal{C}') \subseteq V(\mathcal{C}')$ for any $\mathcal{C}' \in \mathcal{P}(\mathcal{C})$.

[2] Yet, it is easy to see that references to collections of classes could be easily avoided.

We write the composition of mappings in "diagrammatic order": if $f : A \to B$ and $g : B \to C$ then $f;g$ denotes their composition, regardless of whether f and g are mappings between sets, classes, or collections of classes.

Definition 1. *We say mappings (between classes) $\mathcal{U}$, $\mathcal{V}$, $\mathcal{U}'$, $\mathcal{V}'$ (see diagram) form a* ***commutative square*** *iff $\mathcal{V}'\,;\mathcal{U} = \mathcal{U}'\,;\mathcal{V}$. A commutative square is a* ***weak amalgamation square*** *iff for any $b \in \mathcal{B}$ and $c \in \mathcal{C}$ such that $\mathcal{U}(b) = \mathcal{V}(c)$, there exists some $a' \in \mathcal{A}'$ such that $\mathcal{V}'(a') = b$ and $\mathcal{U}'(a') = c$.*

$$\begin{tikzcd}
& \mathcal{A} & \\
\mathcal{B} \arrow[ur, "\mathcal{U}"] & & \mathcal{C} \arrow[ul, "\mathcal{V}"'] \\
& \mathcal{A}' \arrow[ul, "\mathcal{V}'"] \arrow[ur, "\mathcal{U}'"'] &
\end{tikzcd}$$

We call this amalgamation square "weak" because a' is not required to be unique.

2 First-Order Logic and Classical Interpolation Revisited

First-Order Logic. A (many-sorted) first-order signature is a triple (S, F, P) consisting of a set S of sort symbols, a set F of function symbols, and a set P of relation symbols. Each function or relation symbol comes with a string of argument sorts, called *arity*, and for function symbols, a result sort. $F_{w \to s}$ denotes the set of function symbols with arity w and result sort s, and P_w the set of relation symbols with arity w. Given a signature Σ, the class of *Σ-models*, *Mod*(Σ) consists of all first-order structures A interpreting each sort symbol s as a non-empty[3] set A_s, each function symbol σ as a function A_σ from the product of the interpretations of the argument sorts to the interpretation of the result sort, and each relation symbol π as a subset A_π of the product of the interpretations of the argument sorts.

The set of *Σ-sentences*, *Sen*(Σ), consists of the usual first-order sentences built from equational and relational atoms by iterative application of logical connectives and quantifiers. The satisfaction of sentences by models ($A \models \gamma$) is the usual Tarskian satisfaction defined inductively on the structure of the sentences. The satisfaction relation can be extended to a relation $\models$ between classes of models $\mathcal{M} \subseteq \mathit{Mod}(\Sigma)$ and sets of sentences $\Gamma \subseteq \mathit{Sen}(\Sigma)$: $\mathcal{M} \models \Gamma$ iff $A \models \gamma$ for all $A \in \mathcal{M}$ and $\gamma \in \Gamma$. This further induces two operators $\_^* : \mathcal{P}(\mathit{Sen}(\Sigma)) \to \mathcal{P}(\mathit{Mod}(\Sigma))$ and $\_^* : \mathcal{P}(\mathit{Mod}(\Sigma)) \to \mathcal{P}(\mathit{Sen}(\Sigma))$, defined by $\Gamma^* = \{A \mid \{A\} \models \Gamma\}$ and $\mathcal{M}^* = \{\gamma \mid \mathcal{M} \models \{\gamma\}\}$ for each $\Gamma \subseteq \mathit{Sen}(\Sigma)$ and $\mathcal{M} \subseteq \mathit{Mod}(\Sigma)$. The two operators $\_^*$ form a Galois connection between $(\mathcal{P}(\mathit{Sen}(\Sigma)), \subseteq)$ and $(\mathcal{P}(\mathit{Mod}(\Sigma)), \subseteq)$. The two composition operators $\_^*\,;\_^*$ are denoted $\_^\bullet$ and are called *deduction closure* (the one on sets of sentences) and *axiomatizable hull* (the one on classes of models). We call *elementary classes* the classes of models closed under $\_^\bullet$ and *theories* the sets of sentences closed under $\_^\bullet$. If $\Gamma, \Gamma' \subseteq \mathit{Sen}(\Sigma)$, we say that *$\Gamma$ semantically deduces Γ'*, written $\Gamma \models \Gamma'$, iff $\Gamma^* \subseteq \Gamma'^*$.

Given two signatures $\Sigma = (S, F, P)$ and $\Sigma' = (S', F', P')$, a signature morphism $\phi : \Sigma \to \Sigma'$ is a triple $(\phi^{st}, \phi^{op}, \phi^{rl})$ mapping the three components in a compatible way. (When there is no danger of confusion, we denote each

[3] Birkhoff-style axiomatizability, which will be used intensively in this paper, depends on the non-emptiness of carriers [27].

of the mappings $\phi^{st}, \phi^{op}, \phi^{rl}$ by ϕ.) Sentence translations rename the sorts, function-, and relation- symbols. For each signature morphism $\phi : \Sigma \to \Sigma'$, the reduct $A'\restriction_\phi$ of a Σ'-model A' is the Σ-model defined by $(A'\restriction_\phi)_\alpha = A'_{\phi(\alpha)}$ for each sort, function, or relation symbol α in the domain signature of ϕ. Let $Mod(\phi) : Mod(\Sigma') \to Mod(\Sigma)$ denote the mapping $A' \mapsto A'\restriction_\phi$. Satisfaction relation has the important property that it is "invariant under change of notation" [17], i.e., for each $\gamma \in Sen(\Sigma)$ and $A' \in Mod(\Sigma')$, $A' \models \phi(\gamma)$ iff $A'\restriction_\phi \models \gamma$.

Interpolation. The original formulation of interpolation [7] is in terms of signature intersections and unions, that is, w.r.t. squares which are pushouts of signature inclusions. However, subsequent advances in modularization theory [3, 13, 14, 9, 4] showed the need of arbitrary pushout squares or even weak amalgamation squares. A general formulation of interpolation is the following:

Definition 2. *Assume a commutative square of signature morphisms (see diagram) and two sets of sentences $\Gamma_1 \subseteq Sen(\Sigma_1)$, $\Gamma_2 \subseteq Sen(\Sigma_2)$ such that $\phi'_2(\Gamma_1) \models_{\Sigma'} \phi'_1(\Gamma_2)$ (i.e., Γ_1 implies Γ_2 on the "union language" Σ'). An* ***interpolant*** *for Γ_1 and Γ_2 is a set $\Gamma \subseteq Sen(\Sigma)$ such that $\Gamma_1 \models_{\Sigma_1} \phi_1(\Gamma)$ and $\phi_2(\Gamma) \models_{\Sigma_2} \Gamma_2$.*

$$\begin{tikzcd}
 & \Sigma \arrow[dl, "\phi_1"'] \arrow[dr, "\phi_2"] & \\
\Sigma_1 \arrow[dr, "\phi'_2"'] & & \Sigma_2 \arrow[dl, "\phi'_1"] \\
 & \Sigma' &
\end{tikzcd}$$

The following two examples show that, without further restrictions on signature morphisms, an interpolant Γ may not be found with *the same type* of sentences as Γ_1 and Γ_2, but with more general ones. In other words, there are sub-first-order logics which do not admit Craig Interpolation within themselves but in a larger (sub-)logic. The first example below shows a square in unconditional equational logic which does not admit unconditional interpolants, but admits a conditional one:

Example 1. Consider the following pushout of algebraic signatures, as in [25]: $\Sigma = (\{s\}, \{d_1, d_2 : s \to s\})$, $\Sigma_1 = (\{s\}, \{d_1, d_2, c : s \to s\})$, $\Sigma_2 = (\{s\}, \{d : s \to s\})$, $\Sigma' = (\{s\}, \{d, c : s \to s\})$, all morphisms mapping the sort s into itself, ϕ_1 and ϕ_2 mapping d_1 and d_2 into themselves and into d, respectively, ϕ'_2 mapping d_1 and d_2 into d and c into itself, and ϕ'_1 mapping d into itself.

Take $\Gamma_1 = \{(\forall x)d_2(x) = c(d_1(x)),\ (\forall x)d_1(d_2(x)) = c(d_2(x))\}$ and $\Gamma_2 = \{(\forall x)d(d(x)) = d(x)\}$ to be sets of Σ_1-equations and of Σ_2-equations, respectively. It is easy to see that Γ_1 implies Γ_2 in the "union language", i.e., $\phi'_2(\Gamma_1) \models \phi'_1(\Gamma_2)$. But Γ_1 and Γ_2 have no equational Σ-interpolant, because the only equational Σ-consequences of Γ_1 are the trivial ones, of the form $(\forall X)t = t$ with t a Σ-term (since all the nontrivial Σ_1-consequences of Γ_1 contain the symbol c). Yet, Γ_1 and Γ_2 have a conditional-equational interpolant, e.g., $\{(\forall x)d_1(x) = d_2(x) \Rightarrow d_1(x) = d_1(d_1(x))\}$.

The following example shows a situation in which the interpolant cannot even be conditional-equational; it can be a first-order, though:

Example 2. Consider the same pushout of signatures as in previous example and take $\Gamma_1 = \{(\forall x)d_2(x) = d_1(c(x)),\ (\forall x)d_1(d_2(x)) = d_2(c(x))\}$ and

$\Gamma_2 = \{(\forall x)d(d(x)) = d(x)\}$. Again, $\phi_2'(\Gamma_1) \models \phi_1'(\Gamma_2)$. But now Γ_1 and Γ_2 have no conditional-equational Σ-interpolant either, because all nontrivial conditional equations we can infer from Γ_1 contain c (to see this, think in terms of the deduction system for conditional equational logic). Nevertheless, Γ_1 and Γ_2 have a first-order interpolant, e.g., $\{(\forall x)d_1(x) = d_2(x) \Rightarrow (\forall y)d_1(y) = d_1(d_1(y))\}$.

An obstacle to interpolation inside the desired type of sentences in the examples above is the lack of injectivity of ϕ_2 on operation symbols; injectivity on both sorts and operation symbols implies conditional equational interpolation [26].

A counterexample given in [4] shows that not even first-order logic admits interpolation without making additional requirements on the square morphisms. We shall shortly prove that for a pushout square to have first-order interpolation, it is sufficient that it has *one* of the morphisms injective *on sorts*. This is, up to our knowledge, the most general known effective criterion for a pushout to have first-order interpolation.

3 Semantic Interpolation

The interpolation problem, despite its syntactic nature, can be regarded *semantically*, on classes of models. Indeed, by the sentence-model duality and the satisfaction condition, we have that:

- $\phi_2'(\Gamma_1) \models \phi_1'(\Gamma_2)$ iff $\phi_2'(\Gamma_1)^* \subseteq \phi_1'(\Gamma_2)^*$ iff $Mod(\phi_2')^{-1}(\Gamma_1^*) \subseteq Mod(\phi_1')^{-1}(\Gamma_2^*)$;
- $\Gamma_1 \models \phi_1(\Gamma)$ iff $\Gamma_1^* \subseteq \phi_1(\Gamma)^*$ iff $\Gamma_1^* \subseteq Mod(\phi_1)^{-1}(\Gamma^*)$;
- $\phi_2(\Gamma) \models \Gamma_2$ iff $\phi_2(\Gamma)^* \subseteq \Gamma_2^*$ iff $Mod(\phi_2)^{-1}(\Gamma^*) \subseteq \Gamma_2^*$.

Therefore, the interpolation property can be restated only in terms of inclusions between classes of models. If Γ is an interpolant of Γ_1 and Γ_2, we will call Γ^* a *semantic interpolant* of Γ_1^* and Γ_2^*. These suggest defining the following broader notion of "semantic interpolation":

Definition 3. *Consider the commutative diagram on the right, together with some $\mathcal{M} \in \mathcal{P}(\mathcal{B})$ and $\mathcal{N} \in \mathcal{P}(\mathcal{C})$ such that ${\mathcal{V}'}^{-1}(\mathcal{M}) \subseteq {\mathcal{U}'}^{-1}(\mathcal{N})$. We say that $\mathcal{K} \in \mathcal{P}(\mathcal{A})$ is a* ***semantic interpolant*** *of $\mathcal{M}$ and $\mathcal{N}$ iff $\mathcal{M} \subseteq \mathcal{U}^{-1}(\mathcal{K})$ and $\mathcal{V}^{-1}(\mathcal{K}) \subseteq \mathcal{N}$.*

$$\begin{tikzcd}
 & \mathcal{A} & \\
\mathcal{B} \arrow[ur, "\mathcal{U}"] & & \mathcal{C} \arrow[ul, "\mathcal{V}"'] \\
 & \mathcal{D} \arrow[ul, "\mathcal{V}'"] \arrow[ur, "\mathcal{U}'"'] &
\end{tikzcd}$$

If we take $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$ to be $Mod(\Sigma)$, $Mod(\Sigma_1)$, $Mod(\Sigma_2)$, $Mod(\Sigma')$ and $\mathcal{U}$, $\mathcal{V}$, $\mathcal{U}'$, $\mathcal{V}'$ to be $Mod(\phi_1)$, $Mod(\phi_2)$, $Mod(\phi_1')$, $Mod(\phi_2')$, respectively, we obtain the concrete first-order case. The connection between semantic interpolation and classical logical interpolation holds only when one considers classes which are *elementary*, i.e., specified by sets of sentences, and the interpolant is also elementary. Rephrasing the interpolation problem semantically allows us to adopt the following "divide and conquer" approach, already sketched in [25]:

1. Find as many semantic interpolants as possible without caring whether they are axiomatizable or not (note that "axiomatizable" will mean "elementary" only within first-order logic, but we shall consider other logics as well);
2. Then, by imposing diverse axiomatizability closures on the two starting classes of models, try to obtain a closed interpolant.

Let $\mathcal{I}(\mathcal{M},\mathcal{N})$ denote the collection of all semantic interpolants of $\mathcal{M}$ and $\mathcal{N}$. The following gives a precise characterization of semantic interpolants together with a general condition under which they exist.

Proposition 1. *Under the hypothesis of Definition 3:*

1. $\mathcal{I}(\mathcal{M},\mathcal{N}) = [\mathcal{U}(\mathcal{M}), \overline{\mathcal{V}(\overline{\mathcal{N}})}]$;
2. *If the square is a weak amalgamation square then* $\mathcal{I}(\mathcal{M},\mathcal{N}) \neq \emptyset$.

Definition 4. *Given two classes $\mathcal{C}$ and $\mathcal{D}$, a mapping $\mathcal{U} : \mathcal{C} \to \mathcal{D}$ and a pair of operators $F = (F_{\mathcal{C}}, F_{\mathcal{D}})$, we say that $\mathcal{U}$* ***preserves fixed points of*** *F if $\mathcal{U}(Fixed(F_{\mathcal{C}})) \subseteq Fixed(F_{\mathcal{D}})$, that is, for any fixed point of $F_{\mathcal{C}}$ we obtain through $\mathcal{U}$ a fixed point of $F_{\mathcal{D}}$; also we say that $\mathcal{U}$* ***lifts*** *F if $F_{\mathcal{D}} \,; \mathcal{U}^{-1} \sqsubseteq \mathcal{U}^{-1} \,; F_{\mathcal{C}}$, that is, for any $\mathcal{D}' \in \mathcal{P}(\mathcal{D})$ and any $c \in \mathcal{C}$, if $\mathcal{U}(c) \in F_{\mathcal{D}}(\mathcal{D}')$ then $c \in F_{\mathcal{C}}(\mathcal{U}^{-1}(\mathcal{D}'))$.*

The intuition for the word "lifts" used above comes from the case of the operators $F_{\mathcal{C}}$ and $F_{\mathcal{D}}$ being given by binary relations.

The following theorem is at the heart of all our subsequent results. It gives general criteria under which a weak amalgamation square admits semantic interpolants *closed* under certain generic operators.

Theorem 1. *Consider a weak amalgamation square as in diagram and pairs of operators $F = (F_{\mathcal{B}}, F_{\mathcal{A}})$ and $G = (G_{\mathcal{C}}, G_{\mathcal{A}})$ such that:*

1. $F_{\mathcal{A}} \,; G_{\mathcal{A}} \,; F_{\mathcal{A}} = F_{\mathcal{A}} \,; G_{\mathcal{A}}$;
2. *$G_{\mathcal{C}}$ and $G_{\mathcal{A}}$ are closure operators;*
3. *$\mathcal{U}$ preserves fixed points of F;*
4. *$\mathcal{V}$ lifts G.*

$$\begin{tikzcd} & \mathcal{A} & \\ \mathcal{B} \arrow[ur, "\mathcal{U}"] & & \mathcal{C} \arrow[ul, "\mathcal{V}"'] \\ & \mathcal{D} \arrow[ul, "\mathcal{V}'"] \arrow[ur, "\mathcal{U}'"'] & \end{tikzcd}$$

Then for each $\mathcal{M} \in Fixed(F_{\mathcal{B}})$ and $\mathcal{N} \in Fixed(G_{\mathcal{C}})$ such that ${\mathcal{V}'}^{-1}(\mathcal{M}) \subseteq {\mathcal{U}'}^{-1}(\mathcal{N})$, $\mathcal{M}$ and $\mathcal{N}$ have a semantic interpolant $\mathcal{K}$ in $Fixed(F_{\mathcal{A}}) \cap Fixed(G_{\mathcal{A}})$.

The operators above will be conveniently chosen in the next section to be closure operators characterizing axiomatizable classes of models. The two types of axiomatizability that we consider as attached to F and G need not be the same, i.e., the classes $\mathcal{M}$ and $\mathcal{N}$ need not be axiomatizable by the same type of first-order sentences. And in the most fortunate cases, as we shall see below, the interpolant is able to capture and even strengthen the properties of both classes.

4 New Interpolation Results

We next give a series of interpolation results for various types of first-order sentences.[4] Our semantic approach exploits axiomatizability results; since these results use model (homo)morphisms, we first briefly recall some definitions.

Given two Σ-models A and B, a *morphism* $h : A \to B$ is an S-sorted function $(h_s : A_s \to B_s)_{s \in S}$ that commutes with operations and preserves relations.

[4] See the technical report [24] for more interpolation results.

Models and model morphisms form a category denoted $\mathit{Mod}(\Sigma)$ too (just like the class of models), with composition defined as sort-wise function composition. For each signature morphism ϕ, the mapping $Mod(\phi)$ can be naturally extended to a functor. A *surjective* (*injective*) morphism is a morphism which is surjective (injective) on each sort. Because of the weak form of commutation imposed on morphisms w.r.t. the relational part of models, relations and functions do not behave similarly along arbitrary morphisms, but only along closed ones: a morphism $h : A \to B$ is called *closed* if the relation preservation condition holds in the "iff" form, that is, for each predicate symbol π, $(a_1, \ldots, a_n) \in A_\pi$ iff $(h_{s_1}(a_1), \ldots, h_{s_n}(a_n)) \in B_\pi$. A morphism $h : A \to B$ is called *strong* if the target relations are covered through h by the source relation, that is, for each predicate symbol π and $(b_1, \ldots, b_n) \in B_\pi$, there exists $(a_1, \ldots, a_n) \in A_\pi$ such that $(h_{s_1}(a_1), \ldots, h_{s_n}(a_n)) = (b_1, \ldots, b_n)$. Closed injective morphisms and strong surjective morphisms naturally capture the notions of embedding and homomorphic image respectively.

We next define some types of first-order sentences.

- $\mathcal{FO}$: first-order sentences;
- $\mathcal{P}os$: *positive* sentences, that is, constructed inductively from atomic formulae by means of any first-order constructs, except negation;
- $\forall$: sentences $(\forall x_1, x_2, \ldots, x_k)e$, where e is a quantifier free formula;
- $\exists$: sentences $(\exists x_1, x_2, \ldots, x_k)e$, where e is a quantifier free formula;
- $\mathcal{UH}$, *universal Horn clauses*, that is, $(\forall x_1, x_2, \ldots, x_k)(e_1 \wedge e_2 \ldots \wedge e_p) \Rightarrow e$, with e_i, e atomic formulae;
- $\mathcal{UA}$, *universal atoms*, that is, $(\forall x_1, x_2, \ldots x_k)e$, where e is an atomic formula;
- $\forall\vee$, universally quantified disjunctions of atoms, i.e., $(\forall x_1, x_2, \ldots, x_k)(e_1 \vee e_2 \ldots \vee e_p)$ where e_i are atomic formulae;
- $\mathcal{FO}_\infty, \mathcal{UH}_\infty, \forall\vee_\infty$, the infinitary extensions of $\mathcal{FO}, \mathcal{UH}, \forall\vee$, respectively; in the former case, infinite conjunction and disjunction is allowed; in the latter two cases, $e_1 \wedge e_2 \ldots \wedge e_p$ and $e_1 \vee e_2 \ldots \vee e_p$ are replaced by any possibly infinite sentence- conjunction and disjunction respectively.

We assume the reader familiar with some basic model theoretic notions, such as submodels, products, filtered products, ultraproducts, ultrapowers and ultraradicals (see, e.g., [6, 20]). Recall though that, given a family $(A_i)_{i\in I}$ of models and a filter $\mathcal{F}$ on I (i.e., a Boolean filter $\mathcal{F} \subseteq 2^I$), the *filtered product* of $(A_i)_{i\in I}$ over $\mathcal{F}$, denoted $\prod_\mathcal{F} A_i$, has the carrier $B = \prod_{i\in I} A_i/{\equiv}$, where $\equiv$ is given by $(a_i)_{i\in I} \equiv (b_i)_{i\in I}$ iff $\{i \in I \mid a_i = b_i\} \in \mathcal{F}$. Operations are defined by $B_\sigma((a_i^1)_{i\in I}/{\equiv}, \ldots, (a_i^n)_{i\in I}/{\equiv}) = (B_\sigma(a_i^1, \ldots, a_i^n))_{i\in I}/{\equiv}$ and $B_\pi = \{((a_i^1)_{i\in I}/{\equiv}, \ldots, (a_i^n)_{i\in I}/{\equiv}) \mid \{i \in I \mid (a_i^1, \ldots, a_i^n) \in A_{i\pi}\} \in \mathcal{F}\}$. If $\mathcal{F}$ is an *ultrafilter*, then $\prod_\mathcal{F} A_i$ is said to be an *ultraproduct*; moreover, if all A_i's are equal to some model A, then $\prod_\mathcal{F} A_i$ is written $A^I/_\mathcal{F}$ and said to be the *ultrapower* of A over $\mathcal{F}$; in this latter case, A is said to be an *ultraradical* of $A^I/_\mathcal{F}$.
Consider the following binary relations on Σ-models:

- A ***S*** B iff B is isomorphic to a *submodel* of A;
- A ***Ext*** B iff B is isomorphic to an *extension* of A, i.e., to a model C such that A is a submodel of C;

- A ***H*** B iff there exists a *surjective* morphism between A and B;
- A ***Hs*** B iff there exists a *strong surjective* morphism between A and B;
- A ***Ur*** B iff B is an *ultraradical* of a model isomorphic to A, i.e., if A is isomorphic to an *ultrapower* of B

Recall that any binary relation, in particular the ones on $Mod(\Sigma)$ above, has an associated operator bearing the same name. Besides these operators, we shall also consider the operators P, Fp, and Up on $Mod(\Sigma)$ defined below:

- $P(\mathcal{M}) = \mathcal{M} \cup \{$all *products* of models in $\mathcal{M}\}$;
- $Fp(\mathcal{M}) = \mathcal{M} \cup \{$all *filtered products* of models in $\mathcal{M}\}$;
- $Up(\mathcal{M}) = \mathcal{M} \cup \{$all *ultraproducts* of models in $\mathcal{M}\}$;

All these constructions are considered up to isomorphism; for instance, the operator *Up* "grabs" into the class not only the ultraproducts standardly constructed as quotients of direct products, but all models isomorphic to them.

The next proposition collects some known axiomatizability results. For details, the reader is referred to [6] (Section 5.2), [20] (Sections 25 and 26), [1], [22], and [9]. Below, e.g., the pair $(\mathcal{UA}, \{S, H, P\})$ corresponds to the famous Birkhoff Theorem (a class of algebras is equationally axiomatizable iff it is closed under subalgebras, homomorphic images, and products) and the pair $(\mathcal{FO}, \{Up, Ur\})$ corresponds to the Keisler-Shelah Theorem (a class of first-order models is elementary iff it is closed under ultrapowers and ultraradicals).

Proposition 2. *If the pair (T, Ops), consisting of a type T of Σ-sentences and a set Ops of operators on $Mod(\Sigma)$, is one of $(\mathcal{FO}, \{Up, Ur\})$, $(\mathcal{P}os, \{Up, Ur, H\})$, $(\forall, \{S, Up\})$, $(\exists, \{Ext, Up, Ur\})$, $(\mathcal{UH}, \{S, Fp\})$, $(\mathcal{UA}, \{S, H, P\})$, $(\forall\vee, \{Hs, S, Up\})$, $(\mathcal{UH}_\infty, \{S, P\})$, $(\forall\vee_\infty, \{Hs, S\})$, then $\mathcal{M} \subseteq Sen(\Sigma)$ is of the form Γ^* with $\Gamma \subseteq T$ iff $\mathcal{M}$ is a fixed point of all the operators in Ops.*

Consider the following "syntactic" properties for a morphism $\phi : \Sigma \to \Sigma'$, where $\Sigma = (S, F, P)$ and $\Sigma' = (S', F', P')$:

(IS) ϕ is ***injective on sorts***;

(IR) ϕ is ***injective on relation symbols***;

(I) ϕ is ***injective*** on sorts, operation- and relation- symbols

(RS) there are no operation symbols in $F' \setminus \phi(F)$, having the ***result sort*** in $\phi(S)$.

Proposition 3. *For each signature morphism $\phi : \Sigma \to \Sigma'$,*

1. *$Mod(\phi)$ preserves fixed points of P, Fp, Up;*
2. (I) *$\Rightarrow$ $Mod(\phi)$ lifts S, H, Hs and preserves fixed points of Ext* [9].
3. (IS) *and* (RS) *$\Rightarrow$ $Mod(\phi)$ preserves fixed points of S, Hs, and lifts Ext;*
4. (IS), (IR) *and* (RS) *$\Rightarrow$ $Mod(\phi)$ preserves fixed points of H;*
5. (IS) *$\Rightarrow$ $Mod(\phi)$ lifts Ur;*

The table below states interpolation results for diverse types of sentences. It should be read as: given a weak amalgamation square of signatures as in Definition 2 and $\Gamma_1 \subseteq Mod(\Sigma_1)$, $\Gamma_2 \subseteq Mod(\Sigma_2)$, if Γ_1 and Γ_2 are sentences of the indicated types such that $\phi_2'(\Gamma_1) \models \phi_1'(\Gamma_2)$, then there exists an interpolant Γ of the indicated type; the semantic conditions under which this situation holds are given in the $Mod(\phi_1)$- and $Mod(\phi_2)$- columns of the table, with the meaning that $Mod(\phi_1)$ *preserves* fixed points of the indicated operator and $Mod(\phi_2)$ *lifts* the indicated operator. (*Id* is the identity operator.) These semantic conditions are implied by the syntactic conditions listed in the ϕ_1- and ϕ_2- columns; "any" means that no restriction is posed on the signature morphism.

	Γ_1 Type	Γ_2 Type	Γ Type	$Mod(\phi_1)$ preserves	$Mod(\phi_2)$ lifts	ϕ_1	ϕ_2
1	$\mathcal{FO}$	$\mathcal{FO}$	$\mathcal{FO}$	Up	Ur	any	(IS)
2	$\mathcal{FO}$	$\mathcal{P}os$	$\mathcal{P}os$	Up	$H ; Ur$	any	(I)
3	$\mathcal{P}os$	$\mathcal{FO}$	$\mathcal{P}os$	$Up ; H$	Ur	(IS), (IR), (RS)	(IS)
4	$\mathcal{FO}$	$\forall$	$\forall$	Up	S	any	(I)
5	$\forall$	$\mathcal{FO}$	$\forall$	$Up ; S$	Id	(IS), (RS)	any
6	$\forall$	$\mathcal{P}os$	$\forall\vee$	$Up ; S$	Hs	(IS), (RS)	(I)
7	$\mathcal{FO}$	$\exists$	$\exists$	Up	$Ext; Ur$	any	(IS), (RS)
8	$\exists$	$\mathcal{FO}$	$\exists$	$Up ; Ext$	Ur	(I)	(IS)
9	$\mathcal{FO}$	$\mathcal{UH}$	$\mathcal{UH}$	Fp	S	any	(I)
10	$\mathcal{UH}$	$\mathcal{FO}$	$\mathcal{UH}$	$Fp ; S$	Id	(IS), (RS)	any
11	$\mathcal{UH}$	$\mathcal{UA}$	$\mathcal{UA}$	P	$S ; H$	any	(I)
12	$\mathcal{UA}$	$\mathcal{FO}$	$\mathcal{UA}$	$P ; S ; H$	Id	(IS), (IR), (RS)	any
13	$\mathcal{UH}$	$\mathcal{P}os$	$\mathcal{UA}$	$P ; S$	H	(IS),(RS)	(I)
14	$\mathcal{FO}$	$\forall\vee$	$\forall\vee$	Up	$S ; Hs$	any	(I)
15	$\forall\vee$	$\mathcal{FO}$	$\forall\vee$	$Up ; S ; Hs$	Id	(IS), (RS)	any
16	$\mathcal{UH}_\infty$	$\mathcal{UA}$	$\mathcal{UA}$	P	$S ; H$	any	(I)
17	$\mathcal{UH}_\infty$	$\mathcal{FO}_\infty$	$\mathcal{UH}_\infty$	$P ; S$	Id	(IS), (RS)	any
18	$\mathcal{FO}_\infty$	$\forall\vee_\infty$	$\forall\vee_\infty$	Id	$S ; Hs$	any	(I)
19	$\forall\vee_\infty$	$\mathcal{FO}_\infty$	$\forall\vee_\infty$	$S ; Hs$	Id	(IS), (RS)	any
20	$\mathcal{FO}$	$\forall\vee_\infty$	$\forall\vee$	Up	$S ; Hs$	any	(I)

Theorem 2. *The results stated in this table hold, i.e., in each of the 20 cases, if ϕ_1 and ϕ_2 satisfy the indicated properties, Γ_1 and Γ_2 have the indicated types and $\phi_2'(\Gamma_1) \models \phi_1'(\Gamma_2)$, then there exists an interpolant Γ of the indicated type.*

Let us discuss the results listed in the table above. The syntactic conditions on signature morphisms are in many cases weaker than, or equal to, injectivity (I). In fact, if we consider only relational languages, i.e., without operation symbols, *all* the conditions are so (because (RS) becomes vacuous). As for operation symbols, it is interesting to note that (RS) comprises the principle of data encapsulation expressed in algebraic terms [16]. As also suggested by the examples in Section 2, it seems that the degree of generality that one can allow on signature morphism increases with the expressive power of a logic. For instance, line 1 says that first-order interpolation holds whenever the righthand morphism is injective on sorts (and, in fact, since in full first-order logic Craig interpolation is equivalent to the symmetrical property of Robinson consistency,[5] *either one* of the morphisms being injective on sorts would do). On the other hand, universal Horn clauses (lines 9 and 10), and then universal atoms (lines 11, 12, 13) require stronger and stronger assumptions on the signature morphisms. Our results say more than interpolation within a certain type T of sentences: the interpolant has type T provided *one* of the starting sets has type T. Particularly interesting results are listed in lines 6, 13, and 20, where the interpolant strictly “improves” the type of both sides. Regarding finiteness of Γ, as noted in [9], it is easy to see that if Γ_2 is finite, by compactness of first-order logic, Γ can be chosen to be also finite in our cases of finitary sub-first-order logics. On the other hand, the finiteness of Γ_1 does not necessarily imply the finite axiomatizability of Γ^* (this follows by a famous theorem due to Kleene).

5 Applications to Formal Specification

Craig interpolation is a an important/desired property in many areas. Next we consider some applications of our interpolation results to formal specification and module algebra.

In formalisms for modularization [3, 13, 29], modules are built by composing other modules via specific operations. One typically starts with *flat* (or *basic*) modules, which are pairs (Σ, Γ) comprising a signature Σ and a set of Σ-sentences Γ. According to [3], one of the most natural semantics of modules, also called *flat semantics*, is given by their corresponding theories; for example the semantics of a basic module (Σ, Γ) is the theory $(\Sigma, \Gamma^\bullet)$. Diverse operations are used to build up structured theories, among which the *export* (or *information hiding*) and *combination* (or *sum*) operators [3] (or [13]), $\Box$ and $+$. $\Box$ restricts the interface of the theory (Σ, Γ) to common symbols of Σ' and Σ, while $+$ just puts together two theories in their union signature. Formally, for each signature Σ' and theory (Σ, Γ), let $\Sigma'\Box(\Sigma, \Gamma)$ be $(\Sigma' \cap \Sigma, \iota^{-1}(\Gamma))$, where $\iota : \Sigma' \cap \Sigma \hookrightarrow \Sigma$; and for theories (Σ_1, Γ_1) and (Σ_2, Γ_2), let $(\Sigma_1, \Gamma_1) + (\Sigma_2, \Gamma_2)$ be $(\Sigma_1 \cup \Sigma_2, (\Gamma_1 \cup \Gamma_2)^\bullet)$. A very desirable property of specification frameworks is the following *restricted distributivity law*:

$$\Sigma'\Box((\Sigma_1, \Gamma_1) + (\Sigma_2, \emptyset^\bullet)) = (\Sigma'\Box(\Sigma_1, \Gamma_1)) + (\Sigma'\Box(\Sigma_2, \emptyset^\bullet))$$

As discussed in [3, 13], full distributivity does not typically hold. It is shown in [3] that, in first-order logic, restricted distributivity is implied by interpolation. Their proof is rather logic-independent, so it works for any logic that has first-order signatures and satisfies interpolation. In particular, it works for all the sublogics of (finitary or infinitary) first-order logic appearing in the table that precedes Theorem 2. Thus our interpolation results show that the restricted

[5] This is not true however for our examples of sub-first-order logics.

distributivity law holds in module algebra developed within many logical frameworks intermediate between full first-order logic and equational logic.

Another application to formal specifications relies on the fact that interpolation entails a *compositional behavior* of the semantics of structured specifications, by ensuring that the two alternative semantics, the flat and the structured ones, coincide. There are good reasons to *not* always consider the flat semantics of module expressions, but rather to *keep the structure* of modules [29, 5, 18]. In the case of hiding, $\Sigma'\Box(\Sigma, \Gamma)$ provides more information than $(\Sigma', \Gamma^\bullet \cap Sen(\Sigma'))$: (1) Γ might be finite, showing that $\Gamma^\bullet$, maybe unlike $\Gamma^\bullet \cap Sen(\Sigma')$, is finitely presented; (2) while the theory of all Σ'-reducts of (Σ, Γ) (i.e., all visible parts of the possible implementations of the theory) is indeed $\Gamma^\bullet \cap Sen(\Sigma')$, usually not any model of $\Gamma^\bullet \cap Sen(\Sigma')$ is a Σ-reduct of a model of (Σ, Γ); hence the theory does not describe precisely the intended semantics on classes of models.

To understand the role played by interpolation, consider the situation when a module $\Sigma'\Box(\Sigma, \Gamma)$ is imported and its interface (Σ') renamed via a signature morphism $j : \Sigma' \to \Sigma''$ in the importing context. The flat semantics of the renamed module is $(\Sigma'', j(\Gamma^\bullet \cap Sen(\Sigma'))^\bullet)$. On the other hand, the renamed module itself might be regarded construcively as an information hiding module whose interface is Σ'' and whose base module is a consistent renaming of (Σ, Γ). This is achieved by taking the pushout $(\Sigma'' \hookrightarrow \Sigma_0, j_0 : \Sigma \to \Sigma_0)$ of $(\Sigma' \hookrightarrow \Sigma, j : \Sigma' \to \Sigma'')$, yielding the new module $\Sigma''\Box(\Sigma_0, j_0(\Gamma))$. One can show *using interpolation* that the modular and the flat semantics are equivalent, that is, $j(\Gamma^\bullet \cap Sen(\Sigma'))^\bullet = j_0(\Gamma)^\bullet \cap Sen(\Sigma'')$. This desirable semantical equivalence is shown by our results to hold for several first-order sublogics. More precisely, lines 3,5,15 in the table preceding Theorem 2 show that the framework may be restricted to positive-, universal-, or [universal quantification of atom disjunction]- logics. Moreover, line 19 shows the same thing for the [universal quantification of possibly infinite atom disjunction]-logic. According to these results, the renaming morphism j can be allowed to be injective *on sorts* in the case of positive logic and *any* morphism in the other three cases. Note that lines 2,4,14,18 list results complementary to the above, and generalize those in [9]. These latter results relax the requirements not on the renaming morphism, but on the *hiding morphism* (allowing one to replace the inclusion $\Sigma' \hookrightarrow \Sigma$ with an arbitrary signature morphism).

Within a specification framework, one should not commit herself to a particular kind of first-order sub-logic, but rather use the available power of expression on a by-need basis, keeping flexible the border between expressive power and effective/efficient decision or computation. The issue of coexistence of different logical systems brings up a third application of our results. The various logical systems that one would like to use should not be simply "swallowed" by a richer universal logic that encompasses them all, but rather integrated using logic translations. This methodology, which is the meta-logical counterpart of keeping structured (i.e., unflattened) the specifications themselves, is followed for instance in CafeOBJ [11, 12]. The underlying logical structure of this system can be formalized as a *Grothendieck institution* [8], which provides a means of building specifications inside the minimal needed logical system. The framework is initially presented as an *indexed institution*, i.e., a family of logical systems with translations between them, and then flattened by a Grothendieck construction.

Lifting interpolation from the component institutions to the Grothendieck institution was studied in [10]; a criterion is given there for lifting interpolation, consisting mainly of three conditions: (1) that the component institutions have interpolation (for some designated pushouts of signatures); (2) that the involved institution comorphisms have interpolation; (3) that each pullback in the index category yields an interpolating square of comorphisms. We give just one example showing that, via the above conditions, some of our interpolation results can be used for putting together in a consistent way two very interesting logical systems: (finitary) first-order logic ($\mathcal{FO}$) and the logic of universally quantified possibly infinite disjunctions of atoms ($\forall\bigvee_\infty$). While the former is a well-established logic, the latter has the ability of expressing some important properties, not expressible in the former, such as *accessibility* of models, e.g., $(\forall x)(x = 0 \vee x = s(0) \vee x = s(s(0)) \vee \ldots)$ for natural numbers. If one combines the expressive power of these two logics, initiality conditions are also available, e.g., the above accessibility condition ("no junk") can be complemented with the "no confusion" statement $\neg \bigvee_{i,j\in\mathbb{N},i<j} s^i(0) = s^j(0)$. Since the two logical systems have the same signatures, condition (2) above is trivially satisfied. Moreover, our results stated in lines 1 and 19 of the table preceding Theorem 2 ensure condition (1) for some very wide class of signature pushouts. Finally, condition (3) is fulfilled by the result in line 20, which states that formulae from the two logics have interpolants *in their intersection logic*, that of universally quantified (finite) conjunctions of atoms.

6 Related Work and Concluding Remarks

The idea of using axiomatizability properties for proving Craig interpolation first appeared, up to our knowledge, in [27] in the case of many-sorted equational logic. Then [25] generalized this to an arbitrary pullback of categories, by considering some Birkhoff-like operators on those categories, with results applicable to different versions of equational logic. An institution-independent relationship between Birkhoff-like axiomatizability and Craig interpolation was depicted in [9], using a concept of *Birkhoff institution.* If we disregard combination of logics and flatten to the least logic, the results in lines 2,4,14,18 of the table preceding Theorem 2 can be also found in [9]. Our Theorem 1 generalizes the previous "semantic" results, bringing the technique of semantic interpolation, we might say, up to its limit. The merit of Theorem 1 is that it provides general conditions under which a semantic interpolant has a syntactic counterpart (i.e., it is axiomatizable). This theorem solves only half of the interpolation problem; concrete lifting and preserving conditions, as well as certain inclusions between operators, still have to be proved. Thus, in this paper, we provide a general methodology for proving interpolation results. We have also followed this methodology working out many concrete examples. The list of sub-first-order-logics that fit our framework is of course open for other suitably axiomatizable logics; and so are the possible combinations between these logics, which might guarantee interpolants even simpler than the types of formulae of *both* logics, as shown by some of our results. Regarding our combined interpolation results, it is worth pointing out that they are not overlapped with, but rather complementary to, the ones in [10] for Grothendieck institutions. There, some combined interpolation properties are previously assumed, in order to ensure interpolation in the resulted larger logical system.

An interesting fact to investigate would be to which extent can syntactically-obtained interpolation results "compete" with our semantic results. While it is true that the syntactic proofs are usually constructive, they do not seem to provide information on the type of the interpolant comparable to what we gave here. In particular, since the diverse Gentzen systems for first-order logic with equality have only partial cut elimination [15], an appeal to the non-equality version of the language, by adding appropriate axioms for equality in the theory, is needed; moreover, dealing with function symbols requires a further appeal to an encoding of functions as relations, again with the cost of adding some axioms. All these transformations make even some presumably very careful syntactic proofs rather indirect and obliterating, and sometimes place the interpolant way outside the given subtheory - this is the reason why an interpolation theorem for equational logic was not known until a separate, specific proof was given in [28]. Yet, comparing and paralleling (present or future) semantic and syntactic proofs seems fruitful for deepening our understanding of Craig interpolation, this extremely complex and resourceful, purely syntactic and yet surprisingly semantic, property of logical systems.

References

1. H. Andréka and I. Németi. A general axiomatizability theorem formulated in terms of cone-injective subcategories. In *Universal Algebra*, volume 29, pages 13–35. 1982.
2. J. Barwise and J. Feferman. *Model-Theoretic Logics*. Springer, 1985.
3. J. Bergstra, J. Heering, and P. Klint. Module algebra. *Journal of the Association for Computing Machinery*, 37(2):335–372, 1990.
4. T. Borzyszkowski. Generalized interpolation in CASL. *Inf. Process. Lett.*, 76(1-2):19–24, 2000.
5. T. Borzyszkowski. Logical systems for structured specifications. *Theoretical Computer Science*, 286(2):197–245, 2002.
6. C. C. Chang and H. J. Keisler. *Model Theory*. North Holland, Amsterdam, 1973.
7. W. Craig. Linear reasoning. A new form of the Herbrand-Gentzen Theorem. *Journal of Symbolic Logic*, 22:250–268, 1957.
8. R. Diaconescu. Grothendieck institutions. *Appl. Categorical Struct.*, 10(4):383–402, 2002.
9. R. Diaconescu. An institution-independent proof of Craig interpolation theorem. *Studia Logica*, 77(1):59–79, 2004.
10. R. Diaconescu. Interpolation in Grothendieck institutions. *Theoretical Computer Science*, 311:439–461, 2004.

11. R. Diaconescu and K. Futatsugi. *CafeOBJ Report.* World Scientific, 1998. AMAST Series in Computing, volume 6.
12. R. Diaconescu and K. Futatsugi. Logical foundations of CafeOBJ. *Theoretical Computer Science*, 285:289–318, 2002.
13. R. Diaconescu, J. Goguen, and P. Stefaneas. Logical support for modularization. In G. Huet and G. Plotkin, editors, *Logical Environments*, pages 83–130. Cambridge, 1993.
14. T. Dimitrakos and T. Maibaum. On a generalized modularization theorem. *Information Processing Letters*, 74(1–2):65–71, 2000.
15. J. H. Gallier. *Logic for computer science. Foundations of automatic theorem proving.* Harper & Row, 1986.
16. J. Goguen. Types as theories. In *Topology and Category Theory in Computer Science*, pages 357–390. Oxford, 1991.
17. J. Goguen and R. Burstall. Institutions: Abstract model theory for specification and programming. *Journal of the ACM*, 39(1):95–146, January 1992.
18. J. Goguen and G. Roşu. Composing hidden information modules over inclusive institutions. In *From Object Orientation to Formal Methods: Dedicated to the memory of Ole-Johan Dahl*, volume 2635 of *LNCS*, pages 96–123. Springer, 2004.
19. D. G. J. Bicarregui, T. Dimitrakos and T. Maibaum. Interpolation in practical formal development. *Logic Journal of the IGPL*, 9(1):231–243, 2001.
20. J. D. Monk. *Mathematical Logic.* Springer-Verlag, 1976.
21. G. Nelson and D. C. Oppen. Simplification by cooperating decision procedures. *ACM Trans. Program. Lang. Syst.*, 1(2):245–257, 1979.
22. I. Németi and I. Sain. Cone-implicational subcategories and some Birkhoff-type theorems. In *Universal Algebra*, volume 29, pages 535–578. 1982.
23. D. C. Oppen. Complexity, convexity and combinations of theories. *Theoretical Computer Science*, 12:291–302, 1980.
24. A. Popescu, T. Şerbănuţă, and G. Roşu. A semantic approach to interpolation. Technical Report UIUCDCS-R-2005-2643, University of Illinois at Urbana-Champaign.
25. G. Roşu and J. Goguen. On equational Craig interpolation. *Journal of Universal Computer Science*, 6:194–200, 2000.
26. P. H. Rodenburg. Interpolation in conditional equational logic. *Fundam. Inform.*, 15(1):80–85, 1991.
27. P. H. Rodenburg. A simple algebraic proof of the equational interpolation theorem. *Algebra Universalis*, 28:48–51, 1991.
28. P. H. Rodenburg and R. van Glabbeek. An interpolation theorem in equational logic. Technical Report CS-R8838, CWI, 1988.
29. D. Sannella and A. Tarlecki. Specifications in an arbitrary institution. *Information and Control*, 76:165–210, 1988.