An Institutional Foundation for the $\mathbb{K}$ Semantic Framework

Claudia Elena Chiriţă[1]([✉]) and Traian Florin Şerbănuţă[2]

[1] Department of Computer Science, Royal Holloway University of London,
Egham, UK
`claudia.elena.chirita@gmail.com`
[2] Faculty of Mathematics and Computer Science, University of Bucharest,
Bucharest, Romania
`traian.serbanuta@fmi.unibuc.ro`

Abstract. We advance an institutional formalisation of the logical systems that underlie the $\mathbb{K}$ semantic framework and are used to capture both structural properties of program configurations through pattern matching, and changes of configurations through reachability rules. By defining encodings of matching and reachability logic into the institution of first-order logic, we set the foundation for integrating $\mathbb{K}$ into logic graphs of heterogeneous institution-based specification languages such as HETCASL. This will further enable the use of the $\mathbb{K}$ tool with other existing formal specification and verification tools associated with HETS.

1 Introduction

The $\mathbb{K}$ framework [19] is an executable semantic framework based on rewriting and used for defining programming languages, computational calculi, type systems and formal-analysis tools. It was developed as an alternative to the existing operational-semantics frameworks and over the years has been employed to define actual programming languages, to study runtime verification methods and to develop analysis tools such as type checkers, type inferencers, model checkers and verifiers based on Hoare-style assertions. A comprehensive overview of the framework can be found in [20]. Its associated tool [6] enables the development of modular and executable definitions of languages, and moreover, it allows the user to test programs and to explore their behaviour in an exhaustive manner, facilitating in this way the design of new languages. Driven by recent developments on the theoretical foundations of the $\mathbb{K}$ semantic framework [18,21] and on the established connections with other semantic frameworks and formal systems such as reduction semantics, Hoare logic and separation logic, we propose an institutional formalisation [10] of the logical systems on which the $\mathbb{K}$ framework is based: matching and reachability logic. This would allow us to extend the usage of $\mathbb{K}$ by focusing on its potential as a formal specification language, and furthermore, through its underlying logics, to establish rigorous mathematical relationships between $\mathbb{K}$ and other similar languages, enabling the integration of their verification tools and techniques.

Matching logic [18] is a formal system used to express properties about the structure of mathematical objects and language constructs, and to reason about them by means of pattern matching. Its sentences, called patterns, are built in an inductive manner, similar to the terms of first-order logic, using operation symbols provided by a many-sorted signature, as well as Boolean connectives and quantifiers. The semantics is defined in terms of multialgebras, which interpret patterns as subsets of their carriers. This leads to a ternary satisfaction relation between patterns, multialgebras and elements (or states) of multialgebras.

Unlike first-order logic, matching logic is difficult to formalise faithfully as an institution due to the ternary nature of its satisfaction relation and to the fact that patterns are classified by sorts, much in the way the sentences of branching temporal logics are classified into state or path sentences and evaluated accordingly. We overcome these limitations by relying on the concept of stratified institution developed in [2], which extends institutions with an abstract notion of model state and defines a parameterised satisfaction relation that takes into account the states of models. We further develop this concept by adding classes, which are determined by signatures, associated with sentences, and parameterise both the stratification of models and the satisfaction relation. We show that both matching and computation-tree logic can be described as stratified institutions with classes, and we adapt the canonical construction of an ordinary institution from a stratified one presented in [2] to take into consideration the role of classes.

The main advantage of using stratified institutions with classes to formalise matching logic is that we can extend the construction of reachability logic described in [21] from matching to other logical systems. Reachability logic is a formalism for program verification through which transition systems that correspond to the operational semantics of programming languages can be described using reachability rules; these rules rely on patterns and generalise Hoare triples in order to specify transitions between program configurations (similarly to term-rewrite rules). Therefore, reachability logic can be seen as a language-independent alternative to the axiomatic semantics and proof systems particular to each language. We define an abstract institution of reachability logic over an arbitrary stratified institution with classes such that by instantiating this parameter with matching logic we recover the original notion of reachability.

This paper is based on the Master's thesis of the first author [5], which additionally contains detailed proofs of the results presented herein.

2   Preliminaries

2.1   Institution Theory

The concept of institution [10] formalises the intuitive notion of logic by abstracting the alphabet, syntax, semantics and satisfaction relation. In the following, we assume familiarity with the basics of category theory. The reader is referred to the book [14] of Mac Lane and Eilenberg for further reading.

Definition 1. *An institution* $\mathbf{I} = (\mathbb{S}ig^{\mathbf{I}}, \mathrm{Sen}^{\mathbf{I}}, \mathrm{Mod}^{\mathbf{I}}, \models^{\mathbf{I}})$ *consists of*

– *a category* $\mathbb{S}ig^{\mathbf{I}}$ *whose objects are called* signatures,

- a sentence functor $\mathrm{Sen}^{\mathbf{I}}\colon \mathbb{Sig}^{\mathbf{I}} \to \mathbb{Set}$ *giving for every signature Σ the set* $\mathrm{Sen}^{\mathbf{I}}(\Sigma)$ *of Σ-sentences and for every signature morphism $\phi\colon \Sigma \to \Sigma'$ the* sentence translation map $\mathrm{Sen}^{\mathbf{I}}(\phi)\colon \mathrm{Sen}^{\mathbf{I}}(\Sigma) \to \mathrm{Sen}^{\mathbf{I}}(\Sigma')$,
- a model functor $\mathrm{Mod}^{\mathbf{I}}\colon (\mathbb{Sig}^{\mathbf{I}})^{\mathrm{op}} \to \mathbb{Cat}$ *defining for every signature Σ the category* $\mathrm{Mod}^{\mathbf{I}}(\Sigma)$ *of Σ-models and Σ-model homomorphisms, and for every signature morphism ϕ the* reduct functor $\mathrm{Mod}^{\mathbf{I}}(\phi)\colon \mathrm{Mod}^{\mathbf{I}}(\Sigma') \to \mathrm{Mod}^{\mathbf{I}}(\Sigma)$,
- a satisfaction relation $\models^{\mathbf{I}}_{\Sigma} \subseteq |\mathrm{Mod}^{\mathbf{I}}(\Sigma)| \times \mathrm{Sen}^{\mathbf{I}}(\Sigma)$ *for every signature Σ,*

such that the satisfaction condition

$$M' \models^{\mathbf{I}}_{\Sigma'} \mathrm{Sen}^{\mathbf{I}}(\phi)(\rho) \quad iff \quad \mathrm{Mod}^{\mathbf{I}}(\phi)(M') \models^{\mathbf{I}}_{\Sigma} \rho$$

holds for any signature morphism $\phi\colon \Sigma \to \Sigma'$, Σ'-model M' and Σ-sentence ρ.

We may omit the sub/superscripts in the notations of institutions when there is no risk of confusion: for example, $\models^{\mathbf{I}}_{\Sigma}$ may be denoted by $\models$ if the institution $\mathbf{I}$ and the signature Σ are clear. The sentence translation $\mathrm{Sen}^{\mathbf{I}}(\phi)$ and the reduct functor $\mathrm{Mod}^{\mathbf{I}}(\phi)$ may also be denoted by $\phi(\_)$ and $\_\!\restriction_{\phi}$. When $M = M'\!\restriction_{\phi}$ we say that M is a ϕ-*reduct* of M' and that M' is a ϕ-*expansion* of M.

First-order logic constitutes a long-established example of an institution [10].

Example (Many-sorted first-order logic with equality(FOL)). Signatures. A (many-sorted) first-order signature (S, F, P) consists of a set S of *sorts*, a family F of sets $F_{w \to s}$, for $w \in S^*$ and $s \in S$, of *operation* symbols with arity w and sort s (when the arity is empty, $F_{\lambda \to s}$ denotes the set of *constants* of sort s), and a family P of sets P_w, for $w \in S^*$, of *relation* symbols with arity w indexed by arities. A signature (S, F, P) is *algebraic* when P is empty.

Signature Morphisms. The morphisms of signatures $\phi\colon (S, F, P) \to (S', F', P')$ consist of functions $\phi^{\mathrm{st}}\colon S \to S'$ between the sets of sorts, $\phi^{\mathrm{op}}_{w \to s}\colon F_{w \to s} \to F'_{\phi^{\mathrm{st}}(w) \to \phi^{\mathrm{st}}(s)}$, for $w \in S^*$ and $s \in S$, between the sets of operation symbols, and $\phi^{\mathrm{rel}}_w\colon P_w \to P'_{\phi^{\mathrm{st}}(w)}$, for $w \in S^*$, between the sets of relation symbols.

Models. For every signature (S, F, P), a model M interprets every sort symbol s as a set M_s, called the *carrier set of sorts*, every operation symbol $\sigma \in F_{w \to s}$ as a function $M_\sigma\colon M_w \to M_s$, where $M_w = M_{s_1} \times \cdots \times M_{s_n}$ for $w = s_1 \ldots s_n$, with $s_1, \ldots, s_n \in S$, and every relation symbol $\pi \in P_w$ as a subset $M_\pi \subseteq M_w$.

A homomorphism of (S, F, P)-models $h\colon M \to N$ is an indexed family of functions $\{h_s\colon M_s \to N_s \mid s \in S\}$ such that

- h is an (S, F)-algebra homomorphism, that is $h_s(M_\sigma(m)) = N_\sigma(h_w(m))$, for every $\sigma \in F_{w \to s}$ and $m \in M_w$, where $h_w\colon M_w \to N_w$ is the canonical component-wise extension of h to tuples, and
- $h_w(m) \in N_\pi$ if $m \in M_\pi$, i.e., $h_w(M_\pi) \subseteq N_\pi$, for every $\pi \in P_w$.

Model Reducts. For every signature morphism $\phi\colon \Sigma \to \Sigma'$, the *reduct* $M'\!\restriction_{\phi}$ of a Σ'-model M' is defined by $(M'\!\restriction_{\phi})_\alpha = M'_{\phi(\alpha)}$ for every sort, function, or relation symbol α from Σ. The reducts of homomorphisms are defined likewise.

Sentences. The sentences are usual first-order sentences built from equational and relational atoms by applying in an iterative manner Boolean connectives and first-order quantifiers. The existential and universal quantification are over sets of first-order variables, which are triples $\langle x, s(S, F, P)\rangle$ sometimes denoted by $x : s$, where x is the name of the variable and $s \in S$ is its sort: for any $(S, F \uplus X, P)$-sentence ρ, $\exists X.\rho$ and $\forall X.\rho$ are (S, F, P)-sentences, where $(S, F \uplus X, P)$ denotes the extension of (S, F, P) with the elements of X as new symbols of constants. Note that different variables in X should have different names.

Sentence Translations. Every signature morphism $\phi \colon (S, F, P) \to (S', F', P')$ induces a sentence translation $\mathrm{Sen}(\phi) \colon \mathrm{Sen}(S, F, P) \to \mathrm{Sen}(S', F', P')$ that is defined inductively on the structure of sentences and renames the symbols of (S, F, P) according to ϕ. For instance, the translation of an existentially quantified sentence is $\mathrm{Sen}(\phi)(\exists X.\rho) = \exists X^\phi.\mathrm{Sen}(\phi^X)(\rho)$, where $X^\phi = \{x : \phi^{\mathrm{st}}(s) \mid x : s \in X\}$ and $\phi^X \colon (S, F \uplus X, P) \to (S', F' \uplus X^\phi, P')$ extends ϕ canonically.

Satisfaction. The satisfaction relation between models and sentences is the usual Tarskian satisfaction defined inductively on the structure of sentences. For existentially quantified sentences, for example, given a model M of a signature (S, F, P), $M \models \exists X.\rho$ if and only if there exists an expansion M' of M along the signature inclusion $(S, F, P) \hookrightarrow (S, F \uplus X, P)$ such that $M' \models \rho$.

Model Amalgamation. Model amalgamation will prove to be crucial in adding quantifiers over an arbitrary institution. Essentially, it allows us to combine models of different signatures whenever they are compatible with respect to a common sub-signature. Many logical systems of interest for specification theory have model amalgamation, including the examples considered in this paper.

Definition 2. *In any institution, a commuting square of signature morphisms*

$$
\begin{array}{ccc}
\Sigma & \xrightarrow{\varphi_1} & \Sigma_1 \\
\varphi_2 \downarrow & & \downarrow \theta_1 \\
\Sigma_2 & \xrightarrow[\theta_2]{} & \Sigma'
\end{array}
$$

is a weak amalgamation square *if, for each Σ_1-model M_1 and Σ_2-model M_2 such that $\mathrm{Mod}(\varphi_1)(M_1) = \mathrm{Mod}(\varphi_2)(M_2)$, there exists a Σ'-model M', called an* amalgamation *of M_1 and M_2, such that $\mathrm{Mod}(\theta_1)(M') = M_1$ and $\mathrm{Mod}(\theta_2)(M') = M_2$. When M' is unique, the above square is called an* amalgamation square.
  We say that an institution has (weak) model amalgamation *if and only if each pushout square of signature morphisms is a (weak) amalgamation square.*

Therefore, in order to have model amalgamation, the square of signature morphisms must not identify entities of Σ_1 and Σ_2 that do not come from Σ via the signature morphisms φ_1 and φ_2. Moreover, to guarantee the uniqueness of the amalgamation, Σ' must contain only entities that come from Σ_1 or Σ_2.

Presentations. The presentations over an institution represent one of the simplest forms of specifications over that logic being formed merely of a signature and a (usually finite) set of its sentences. We will use presentations in our paper to encode reachability logic into first-order logic.

Definition 3. *The* presentations *of an institution* $\mathbf{I} = (\mathbb{S}ig, \text{Sen}, \text{Mod}, \models)$ *are pairs* (Σ, E) *consisting of a signature* Σ *and a set* E *of* Σ*-sentences. They form a category* $\mathbb{P}res$ *whose arrows* $\phi \colon (\Sigma, E) \to (\Sigma', E')$ *are signature morphisms* $\phi \colon \Sigma \to \Sigma'$ *such that* $E' \models \phi(E)$. *By extending the sentence functor, the model functor and the satisfaction relation from the signatures of* $\mathbf{I}$ *to presentations we obtain an institution* $\mathbf{I}^{\text{pres}} = (\mathbb{P}res, \text{Sen}^{\text{pres}}, \text{Mod}^{\text{pres}}, \models^{\text{pres}})$ *of* $\mathbf{I}$ *presentations.*

Moving Between Institutions. In order to use institutions as formalisations of logical systems in a heterogeneous setting, one needs to define formally a notion of map between institutions. Several concepts have been defined over the years, including semi-morphisms, morphisms, and comorphisms, some of which can be found in [23]. In our work, we focus only on comorphisms [15,24], which reflect the intuition of embedding simpler institutions into more complex ones.

Definition 4. *Given two institutions* $\mathbf{I}$ *and* $\mathbf{I}'$, *a* comorphism $(\Phi, \alpha, \beta) \colon \mathbf{I} \to \mathbf{I}'$ *consists of*

– *a signature functor* $\Phi \colon \mathbb{S}ig \to \mathbb{S}ig'$,
– *a natural transformation* $\alpha \colon \text{Sen} \Rightarrow \Phi \,;\, \text{Sen}'$, *and*
– *a natural transformation* $\beta \colon \Phi^{\text{op}} \,;\, \text{Mod}' \Rightarrow \text{Mod}$

such that the following satisfaction condition *holds for any* $\mathbf{I}$*-signature* Σ, $\Phi(\Sigma)$*-model* M', *and* Σ*-sentence* $\rho \colon M' \models^{\mathbf{I}'}_{\Phi(\Sigma)} \alpha_\Sigma(\rho)$ *iff* $\beta_\Sigma(M') \models^{\mathbf{I}}_{\Sigma} \rho$.

2.2 $\mathbb{K}$ Semantic Framework

The $\mathbb{K}$ framework [20] is an executable semantic framework based on rewriting and used for defining programming languages, computational calculi, type systems and formal analysis tools. It was developed as an alternative to the existing operational-semantics (SOS) frameworks and has been employed to define actual programming languages such as C [9], Python [12], and Java [3].

In defining semantics for programming languages, $\mathbb{K}$ handles cell-like structures named *configurations* and relies on computational structures – *computations* – to model transitions between these configurations by applying local rewriting *rules*. To illustrate how language specifications can be written in the $\mathbb{K}$ semantic framework, we consider the following running example of the (partial) definition of IMP [1], an elementary imperative programming language.

Listing 1.1. The IMP programming language

```
module IMP−SYNTAX
  syntax AExp ::= Int | Id
               | AExp "/" AExp              [left, strict]
               >  AExp "+" AExp             [left, strict]
               | "(" AExp ")"               [bracket]
  syntax BExp ::= Bool
               | AExp "<=" AExp             [seqstrict]
               | "!" BExp                   [strict]
               >  BExp "&&" BExp            [left, strict(1)]
               | "(" BExp ")"               [bracket]
  syntax Block ::= "{" "}" | "{" Stmt "}"
  syntax Stmt ::= Block
               | Id "=" AExp ";"            [strict(2)]
               | "if" "(" BExp ")"
                 Block "else" Block         [strict(1)]
               | "while" "(" BExp ")" Block
               >  Stmt Stmt                 [left]
  syntax Pgm ::= "int" Ids ";" Stmt
  syntax Ids ::= List{Id,","}
endmodule

module IMP
  imports IMP−SYNTAX
  syntax KResult ::= Int | Bool
  configuration ⟨t⟩⟨k⟩ $PGM:Pgm ⟨/k⟩
                ⟨state⟩ .Map ⟨/state⟩⟨/t⟩

  rule ⟨k⟩ X:Id  =>  I ⋯⟨/k⟩ ⟨state⟩⋯ X |−> I ⋯⟨/state⟩
  rule I1:Int / I2:Int   =>  I1 /Int I2   when I2 =/=Int 0
  rule I1:Int + I2:Int   =>  I1 +Int I2
  rule ! T:Bool  =>  notBoolT
  rule ⟨k⟩ int (X:Id,Xs:Ids   =>  Xs);_ ⟨/k⟩
        ⟨state⟩ Rho:Map (.  =>  X |−>0) ⟨/state⟩
        when notBool(X in keys(Rho))
   ⋯
endmodule
```

In the following sections we introduce two logics in order to formalize the
𝕂 framework. *Matching logic* will be used to define the syntactic constructs –
the syntax of the specified programming languages – and the patterns matched
in the semantic rules, and to partially capture the semantics of the program-
ming languages by defining the states of the running programs. Subsequently,
we build *reachability logic* upon matching logic to capture the semantic rules. Its
sentences, defined over the signatures of matching logic, correspond to the rules
in the 𝕂 modules, while the models represent implementations of programming
languages. The language definitions written in 𝕂 will thus be seen, leaving aside
some parsing instructions without logical interpretation, as formal specifications
over reachability logic.

3 Matching Logic

The generality of institutions allows them to accommodate a great variety of logical systems. As a downside however, and as it would be expected for such an abstract notion, certain logics cannot be captured in full detail by institutions; that is, by considering them only as institutions we lose precious information. An example is computation-tree logic, for which we lose the distinction between state and path sentences (which, in fact, do not belong to the sentences of computation-tree logic formalised as an institution).

Matching logic falls into the same category, but this time, we lose the sorts of patterns and the states of models. To palliate this, we extend institutions with notions of classes (for sorts) and stratification of models (for states). The end result – the concept of stratified institution with classes – is obtained as a combination of the institutions with classes described in Definition 5 with the stratified institutions introduced in [2].

3.1 Stratified Institutions with Classes

Definition 5. *An* institution with classes *is a tuple* $(\mathbb{Sig}, \mathrm{Cls}, \mathrm{Sen}, \kappa, \mathrm{Mod}, \models)$, *where*

– $(\mathbb{Sig}, \mathrm{Sen}, \mathrm{Mod}, \models)$ *is an institution,*
– $\mathrm{Cls}\colon \mathbb{Sig} \to \mathbb{Set}$ *is a functor giving for each signature a set whose elements are* called classes *of that signature, and*
– $\kappa\colon \mathrm{Sen} \Rightarrow \mathrm{Cls}$ *is a natural transformation giving a class for each sentence.*

We will use the notation $\mathrm{Sen}(\Sigma)_c$ *for* $\kappa^{-1}(c)$, $c \in \mathrm{Cls}(\Sigma)$ *to denote the set of* Σ-*sentences of class* c.

Example. An immediate example of an institution with classes is the atomic fragment of equational first-order logic. In this case, Cls is the forgetful functor that maps every signature (S, F) to its underlying set of sorts S, and $\kappa_{(S,F)}$ is the function that assigns to each atom $t = t'$ the common sort of t and t'.

Definition 6. *A* stratified institution with classes *is a tuple* $\underline{\mathrm{I}} = (\mathbb{Sig}, \mathrm{Cls}, \mathrm{Sen}, \kappa, \mathrm{Mod}, [\![\_]\!], \models)$ *consisting of:*

– *a category* $\mathbb{Sig}$ *of* signatures *and* signature morphisms,
– *a class functor* $\mathrm{Cls}\colon \mathbb{Sig} \to \mathbb{Set}$, *giving for every signature a set of* classes,
– *a sentence functor* $\mathrm{Sen}\colon \mathbb{Sig} \to \mathbb{Set}$, *defining for every signature a set of* sentences,
– *a natural transformation* $\kappa\colon \mathrm{Sen} \Rightarrow \mathrm{Cls}$, *associating a class to each sentence,*
– *a model functor* $\mathrm{Mod}\colon \mathbb{Sig}^{op} \to \mathbb{Cat}$, *defining a category of* models *for every signature,*
– *a stratification* $[\![\_]\!]$ *giving*
  • *for every signature* Σ, *a family of functors* $[\![\_]\!]_{\Sigma,c}\colon \mathrm{Mod}(\Sigma) \to \mathbb{Set}$, *indexed by classes* $c \in \mathrm{Cls}(\Sigma)$, *and*

- *for every signature morphism $\phi\colon \Sigma \to \Sigma'$, a functorial family of natural transformations $[\![\_]\!]_{\phi,c}\colon [\![\_]\!]_{\Sigma',\mathrm{Cls}(\phi)(c)} \Rightarrow \mathrm{Mod}(\phi) ; [\![\_]\!]_{\Sigma,c}$, indexed by classes $c \in \mathrm{Cls}(\Sigma)$, such that $[\![M']\!]_{\phi,c}$ is surjective for every $M' \in |\mathrm{Mod}(\Sigma')|$, and*
- *a satisfaction relation between models and sentences, parameterised by model states and classes: $M \models^m_{\Sigma,c} \rho$, where Σ is a signature, $c \in \mathrm{Cls}(\Sigma)$, $M \in |\mathrm{Mod}(\Sigma)|$, $m \in [\![M]\!]_{\Sigma,c}$, and $\rho \in \mathrm{Sen}(\Sigma)_c$*

such that the following properties are equivalent:

i. $\mathrm{Mod}(\phi)(M') \models^{[\![M']\!]_{\phi,c}(m')}_{\Sigma,c} \rho$

ii. $M' \models^{m'}_{\Sigma',\mathrm{Cls}(\phi)(c)} \mathrm{Sen}(\phi)(\rho),$

for every signature morphism $\phi\colon \Sigma \to \Sigma'$, every class $c \in \mathrm{Cls}(\Sigma)$, every model $M' \in |\mathrm{Mod}(\Sigma')|$, every state $m' \in [\![M']\!]_{\Sigma',\mathrm{Cls}(\phi)(c)}$, and every $\rho \in \mathrm{Sen}(\Sigma)_c$.

The functoriality of $[\![\_]\!]_{\phi,c}\colon [\![\_]\!]_{\Sigma',\mathrm{Cls}(\phi)(c)} \Rightarrow \mathrm{Mod}(\phi) ; [\![\_]\!]_{\Sigma,c}$ means that for every signature morphisms $\phi\colon \Sigma \to \Sigma'$, $\phi'\colon \Sigma' \to \Sigma''$, every Σ''-model M'', and every class $c \in \mathrm{Cls}(\Sigma)$, $[\![M'']\!]_{\phi;\phi',c} = [\![M'']\!]_{\phi',\phi(c)} ; [\![M''{\upharpoonright}_{\phi'}]\!]_{\phi,c}.$

$$[\![M'']\!]_{\Sigma'',\phi'(\phi(c))} \xrightarrow{\;[\![M'']\!]_{\phi',\phi(c)}\;} [\![M''{\upharpoonright}_{\phi'}]\!]_{\Sigma',\phi(c)} \xrightarrow{\;[\![M''{\upharpoonright}_{\phi'}]\!]_{\phi,c}\;} [\![(M''{\upharpoonright}_{\phi'}){\upharpoonright}_{\phi}]\!]_{\Sigma,c}$$
$$\underbrace{}_{[\![M'']\!]_{\phi;\phi',c}}$$

Proposition 1. *Every stratified institution with classes $\underline{\mathrm{I}} = (\mathbb{Sig}, \mathrm{Cls}, \mathrm{Sen}, \kappa, \mathrm{Mod}, [\![\_]\!], \models)$ determines an institution denoted $\flat\mathbf{I}$ whose category of signatures is $\mathbb{Sig}$, sentence functor is Sen, model functor is Mod, and satisfaction relation $\models_\Sigma \subseteq |\mathrm{Mod}(\Sigma)| \times \mathrm{Sen}(\Sigma)$ is defined, for every signature $\Sigma \in |\mathbb{Sig}|$, as follows:*

$$M \models_\Sigma \rho \quad \textit{iff} \quad M \models^m_{\Sigma,c} \rho \textit{ for every } m \in [\![M]\!]_{\Sigma,c}, \textit{ where } c = \kappa_\Sigma(\rho).$$

Computation-Tree Logic (CTL). We formalise computation-tree logic as a first example of a stratified institution with classes. Similarly to other temporal logics, the usual presentation of CTL is based on propositional logic. CTL inherits the signatures of propositional logic, and thus, the category of its *signatures* is $\mathbb{Set}$.

CTL formulae can express properties of a state or a path (i.e. an infinite sequence of states) of a transition system (defined below), being *classified* into state and path formulae: $\mathrm{Cls}(\Sigma) = \{state, path\}$, for every $\Sigma \in |\mathbb{Sig}| = |\mathbb{Set}|$.

We define the functor Sen, and the natural transformation κ simultaneously, describing the *sentences* of a signature and their classes:

- the atomic propositions $a \in \Sigma$ are sentences of class *state*,
- init is a proposition of class *state*,
- $\varphi_1 \wedge \varphi_2$ is a sentence of class *state*, for every φ_1, φ_2 sentences of class *state*,
- $\neg\varphi$ is a sentence of class *state*, for every sentence φ of class *state*,
- $\exists\pi, \forall\pi$ are sentences of class *state*, for every sentence π of class *path*,
- $\bigcirc\varphi$ is a sentence of class *path*, for every sentence φ of class *state*,
- $\varphi_1 \mathsf{U} \varphi_2$ is a sentence of class *path*, for every φ_1, φ_2 sentences of class *state*.

The *models* of a <u>CTL</u> signature Σ are transition systems TS $= (S, \rightarrow, I, L)$, where S is a set of states, $\rightarrow \subseteq S \times S$ is a transition relation, $I \subseteq S$ is a set of initial states, and $L \colon S \to 2^{\Sigma}$ is a labelling function. We define a transition system morphism $h \colon (S, \rightarrow, I, L) \to (S', \rightarrow', I', L')$ as a function $h \colon S \to S'$ such that $h(I) \subseteq I'$, $h(\rightarrow) \subseteq \rightarrow'$, and $L(s) = L'(h(s))$, for every $s \in S$.

The *stratification* of models is defined as follows:

– $[\![\text{TS}]\!]_{\Sigma, state}$ is the set S of states of TS,
– $[\![\text{TS}]\!]_{\Sigma, path}$ is the set of paths of TS, that is sequences $s_0, s_1, \ldots \in S^{\omega}$, such that $s_i \to s_{i+1}$ for $i \in \omega$.

For every signature morphism $\phi \colon \Sigma \to \Sigma'$, the components of the natural transformations $[\![\_]\!]_{\phi, c}$ are identity functions:

$$[\![\text{TS}']\!]_{\phi, state}(s') = s', \quad [\![\text{TS}']\!]_{\phi, path}(p') = p',$$

for every $s' \in [\![\text{TS}']\!]_{\Sigma', state}$, $p' \in [\![\text{TS}']\!]_{\Sigma', path}$, and $\text{TS}' \in |\text{Mod}(\Sigma')|$.

The *satisfaction relation* between models and sentences is given by:

– TS $\models^s_{\Sigma, state} \rho$ iff
  • $\rho \in L(s)$, for $\rho \in \Sigma$
  • $s \in I$, for $\rho = \text{init}$
  • TS $\not\models^s_{\Sigma, state} \varphi$, for $\rho = \neg\varphi$
  • TS $\models^s_{\Sigma, state} \varphi_1$ and TS $\models^s_{\Sigma, state} \varphi_2$, for $\rho = \varphi_1 \wedge \varphi_2$
  – there is $p = s_0, s_1, \ldots \in [\![\text{TS}]\!]_{\Sigma, path}$ with $s_0 = s$ such that TS $\models^p_{\Sigma, path} \pi$, for $\rho = \exists\pi$,
– TS $\models^p_{\Sigma, path} \rho$ iff
  • TS $\models^{s_1}_{\Sigma, state} \varphi$, for $\rho = \bigcirc\varphi$
  • there exists an index j such that TS $\models^{s_j}_{\Sigma, state} \varphi_2$, and for all $i < j$, TS $\models^{s_i}_{\Sigma, state} \varphi_1$, for $\rho = \varphi_1 \mathsf{U} \varphi_2$,

for every signature Σ, every state $s \in S$, every path $p = s_0, s_1, \ldots$, every sentence ρ and every model TS $\in |\text{Mod}(\Sigma)|$.

3.2   Matching Logic

The original notion of matching logic developed in [18] can be described as a stratified institution with classes <u>ML</u> as follows.

Signatures. The signatures of <u>ML</u> are algebraic signatures.

Example. Let us consider the specification of the IMP programming language exemplified in Listing 1.1. The signature of the IMP-SYNTAX module is obtained by adding to the built-in syntactic categories and their corresponding semantic operations new sorts and operation symbols introduced by the **syntax** keyword. For example, in the fragment of the syntax module below, the AExp sort is introduced as a supersort of the *Int* and *Id* sorts.[1] Addition and division

---

[1] For simplicity, the formalism we used in this paper does not take into account the subsorting relation. We could further include subsorts following ideas developed for order-sorted equational logic [11].

are defined as binary operations with arguments and results of sort AExp, while bracketing is defined as a unary operation of the same sort. We note that only the **bracket** attribute has an effect on the signature of the specification as it determines the removal of its corresponding symbol of operation from the signature. The **left** and **strict** attributes are only used in parsing programs and in refining the evaluation strategy (by sequencing computational tasks), and thus, they do not play a role in defining the signature.

Listing 1.2. The IMP programming language – AExp syntax

```
syntax AExp ::= Int | Id
              | AExp "/" AExp              [left, strict]
              > AExp "+" AExp              [left, strict]
              | "(" AExp ")"               [bracket]
```

The signature of the fragment above is $\text{AExp}^{\text{Sig}} = (S \cup S_{\text{BUILT-IN}}, F \cup F_{\text{BUILT-IN}})$, where $S_{\text{BUILT-IN}}$ and $F_{\text{BUILT-IN}}$ are the built-in sorts and operations, $S = \{\text{AExp}\}$ and $F_{AExp\,AExp\rightarrow AExp} = \{\_+\_, \_/\_\}$. Similarly, the signature of the IMP module is obtained from the signature of the imported module IMP-SYNTAX, extending its signature through the addition of the sorts T, K, State and KResult and the operations $\langle k \rangle_-\langle/k\rangle \in F_{\text{Pgm}\rightarrow\text{K}}$,[2] $\langle\text{state}\rangle_-\langle/\text{state}\rangle \in F_{\text{Map}\rightarrow\text{State}}$ and $\langle\text{t}\rangle_-\langle/\text{t}\rangle \in F_{\text{K State}\rightarrow\text{T}}$ introduced by the keyword **configuration**.

Classes of a Signature. Every algebraic signature (S, F) determines (through the functor Cls) the set of classes S, that is the set of its sorts. Similarly, every morphism $\phi\colon (S, F) \rightarrow (S', F')$ determines a translation of classes $\phi^{\text{st}}\colon S \rightarrow S'$.

Sentences. The sentences (or *patterns*) in <u>ML</u> of given sorts are defined as follows: for every signature Σ, $\text{Sen}(\Sigma)$ is the least set that contains *basic patterns* (terms over Σ, see [18]) and that is closed under the Boolean connectives $\neg, \wedge$, and the existential quantifier $\exists$.

– For each $s \in S$, the basic patterns of sort s are first-order Σ-terms of sort s.
– For every pattern π of sort s, $\neg\pi$ is a pattern of sort s.
– For every two patterns π_1, π_2 of sort s, $\pi_1 \wedge \pi_2$ is a pattern of sort s.
– For every variable x of sort s (defined formally as a tuple $\langle x, s, \Sigma \rangle$, where x is the name of the variable, and s is its sort), and every pattern $\pi \in \text{Sen}(S, F \uplus \{x : s\})$, $\exists x : s.\pi$ is a pattern in $\text{Sen}(\Sigma)$.

The *sentence translation* along a signature morphism $\phi\colon \Sigma \rightarrow \Sigma'$ is defined similarly to the translation of first-order sentences. For instance, for basic patterns π of sort s, $\text{Sen}(\phi)(\pi) = \phi^{\text{tm}}_s(\pi)$, where ϕ^{tm} is the extension of ϕ to terms that maps $\sigma(t_1, \ldots, t_n) : s$ to $\phi^{\text{op}}(\sigma)(\phi^{\text{tm}}(t_1), \ldots, \phi^{\text{tm}}(t_n)) : \phi^{\text{st}}(s)$, for every $\sigma \in F_{s_1 \ldots s_n \rightarrow s}$, and term t_i of sort s_i.

Example. We can give as examples of sentences of an <u>ML</u>-signature, the patterns matched in the $\mathbb{K}$ rules corresponding to the IMP programming language specification presented in Listing 1.1: I1:Int + I2:Int, I1:Int/I2:Int $\wedge$ I2=/=0.

---

[2] Pgm is the sort defined in Listing 1.1 for capturing the syntax of an IMP program.

Classes of Sentences. The class of a pattern is given by its sort through the natural transformation $\kappa \colon \mathrm{Sen} \Rightarrow \mathrm{Cls}$ that is defined inductively on the structure of sentences:

- $\kappa_{(S,F)}(\pi) = s$, for every basic pattern $\pi \in (T_\Sigma)_s$,
- $\kappa_{(S,F)}(\neg \pi) = \kappa_{(S,F)}(\pi)$, for every pattern π,
- $\kappa_{(S,F)}(\pi_1 \wedge \pi_2) = \kappa_{(S,F)}(\pi_1) = \kappa_{(S,F)}(\pi_2)$, for every two patterns π_1, π_2,
- $\kappa_{(S,F)}(\exists x \colon s.\pi) = \kappa_{(S,F \uplus \{x \colon s\})}(\pi)$, for every pattern π.

Models. The models of $\underline{\mathrm{ML}}$ are *multialgebras* [13]. These are generalisations of algebras having non-deterministic operations that return sets of possible values; that is, multialgebras interpret operation symbols from the carrier set of their arity to the powerset of the carrier set of their sort. For a signature $\Sigma = (S, F)$, a *multialgebra homomorphism* $h \colon M \to N$ is a family of functions indexed by the signature's sorts $\{h_s \colon M_s \to N_s \mid s \in S\}$, such that $h_s(M_\sigma(m_1, \ldots, m_n)) \subseteq N_\sigma(h_{s_1}(m_1), \ldots, h_{s_n}(m_n))$, for every $\sigma \in F_{s_1 \ldots s_n \to s}$ and every $m_i \in M_{s_i}$.

Stratification. The stratification of models is given, for every signature Σ and class s of Σ, by $[\![M]\!]_{\Sigma,s} = M_s$, and for every signature morphism $\phi \colon \Sigma \to \Sigma'$, class s of Σ and model M' of Σ', by $[\![M']\!]_{\phi,s}(m') = m'$, where $m' \in M'_{\phi^{\mathrm{st}}(s)}$.

Satisfaction Relation. The satisfaction relation is based on the interpretation of patterns in models. For any multialgebra M, we define M_π, the interpretation of a pattern π in M, inductively, as follows:

- for every pattern $\pi \in F_{\lambda \to s}$, M_π is the interpretation of the constant π,
- $M_\pi = \bigcup \{M_\sigma(m_1, \ldots, m_n) \mid m_i \in M_{t_i}\}$, for every basic pattern $\sigma(t_1, \ldots, t_n)$,
- $M_\pi = M_s \setminus M_{\pi_1}$, for every pattern $\pi = \neg \pi_1$, where π_1 is a pattern of sort s,
- $M_\pi = M_{\pi_1} \cap M_{\pi_2}$, for every pattern $\pi = \pi_1 \wedge \pi_2$,
- $M_\pi = \bigcup \{(M, X)_{\pi_1} \mid X \subseteq M_t\}$, for every pattern $\pi = \exists x \colon t.\pi_1$, where π_1 is a pattern of sort s, and (M, X) is the expansion of M along the inclusion $(S, F) \subseteq (S, F \uplus \{x \colon t\})$ given by $(M, X)_x = X$.

We now have all the necessary concepts for defining the *satisfaction relation*:

$$M \models_{\Sigma,s}^m \pi \quad \text{iff} \quad m \in M_\pi.$$

Proposition 2. *For every signature morphism* $\phi \colon \Sigma \to \Sigma'$, *sort* $s \in \mathrm{Cls}(\Sigma)$, *multialgebra* $M' \in \mathrm{Mod}(\Sigma')$, *state* $m' \in [\![M']\!]_{\Sigma',\mathrm{Cls}(\phi)(s)}$, *and pattern* π *of sort* s

$$\mathrm{Mod}(\phi)(M') \models_{\Sigma,s}^{[\![M']\!]_{\phi,s}(m')} \pi \quad \text{iff} \quad M' \models_{\Sigma',\mathrm{Cls}(\phi)(s)}^{m'} \mathrm{Sen}(\phi)(\pi).$$

In order to formalise the $\mathbb{K}$ framework we should interpret the variables in a deterministic manner. For example, in the specification of the IMP language, the variables in the patterns matched by the semantic rules have a deterministic interpretation: the variables $\mathrm{I1}, \mathrm{I2}$ and T of the patterns $\mathrm{I1} : Int + \mathrm{I2} : Int$, $!\,\mathrm{T} : Bool$ are interpreted as sole elements of sort Int or $Bool$ respectively, as opposed to the interpretation of variables in $\underline{\mathrm{ML}}$ as sets of elements.

Matching Logic with Deterministic Variables ($\underline{\text{ML}}^+$). We refine the above definition of matching logic $\underline{\text{ML}} = (\text{Sig}, \text{Cls}, \text{Sen}, \kappa, \text{Mod}, [\![\_]\!], \models)$, by interpreting the variables in a deterministic way, as presented in [18].

$\underline{\text{ML}}^+$ is defined as a stratified institution with classes, whose *category of signatures* is denoted by Sig^+. Its objects are tuples (S, F, D), where (S, F) and (S, D) are algebraic signatures of $\underline{\text{ML}}$, such that $F_{w \to s} \cap D_{w \to s} = \emptyset$ for every $w \in S^*$, and $s \in S$. For signatures $\Sigma = (S, F, D)$ and $\Sigma' = (S', F', D')$, a signature morphism $\phi \colon \Sigma \to \Sigma'$ is a tuple $(\phi^{\text{st}}, \phi^{\text{op}}, \phi^{\text{det}})$, where the pairs $(\phi^{\text{st}}, \phi^{\text{op}})$ and $(\phi^{\text{st}}, \phi^{\text{det}})$ are signature morphisms in Sig.

We define the functor $\text{U} \colon \text{Sig}^+ \to \text{Sig}$ by $\text{U}(S, F, D) = (S, F \cup D)$ for signatures, and by $\text{U}(\phi) = (\phi^{\text{st}}, \phi^{\text{op}} \cup \phi^{\text{det}})$ for signature morphisms. The *classes* and the *sentences of a signature* are given by the functor compositions $\text{Cls}^+ = \text{U} ; \text{Cls}$, and $\text{Sen}^+ = \text{U} ; \text{Sen}$[3] respectively. The *classes of sentences* are determined by the composition $\text{U} \cdot \kappa \colon \text{Sen}^+ \to \text{Cls}^+$ of the functor U with the natural transformation κ, that is, $(\text{U} \cdot \kappa)_\Sigma = \kappa_{\text{U}(\Sigma)}$, for every signature Σ.

The *models* of $\underline{\text{ML}}^+$ are determined by the functor $\text{Mod}^+ \colon (\text{Sig}^+)^{\text{op}} \to \mathbb{C}\text{at}$, that assigns to each signature $\Sigma = (S, F, D)$ the full subcategory of $\text{Mod}(\text{U}(\Sigma))$ consisting of the models M in which every operation symbol in D is interpreted in a deterministic way. For every signature morphism $\phi \colon \Sigma \to \Sigma'$ and every model $M' \in |\text{Mod}^+(\Sigma')|$, we define $\text{Mod}^+(\phi)(M')$ as $\text{Mod}(\text{U}(\phi))(M')$. Notice that the functor Mod^+ is well-defined, as $|\text{Mod}(\text{U}(\phi))(M')_\sigma(m_1, \ldots, m_n)| = |M'_{\phi^{\text{det}}(\sigma)}(m_1, \ldots, m_n)| = 1$ for every operation symbol σ of D.

The *stratification of models* is defined just as in the case of $\underline{\text{ML}}$:

– $[\![\_]\!]^+_{\Sigma, c} \colon \text{Mod}^+(\Sigma) \to \text{Set}$ maps every model M to $[\![M]\!]_{\text{U}(\Sigma), c}$
– $[\![M]\!]^+_{\phi, c} \colon [\![M]\!]^+_{\Sigma', \text{Cls}^+(\phi)(c)} \to \text{Mod}^+(\phi) ; [\![M]\!]^+_{\Sigma, c}$ maps every state m to the state $[\![M]\!]_{\text{U}(\phi), c}(m)$, for every morphism $\phi \colon \Sigma \to \Sigma'$ and class c of Σ.

Finally, the *satisfaction relation* between models and sentences is defined analogously to the satisfaction relation of $\underline{\text{ML}}$. As a result, it holds for example, that for any basic pattern π, any signature $\Sigma \in \text{Sig}^+$, any class $c \in \text{Cls}^+(\Sigma)$, and any model $M \in |\text{Mod}^+(\Sigma)|$, $M(\models^{\underline{\text{ML}}^+})^m_{\Sigma, c} \pi$ iff $M(\models^{\underline{\text{ML}}})^m_{\text{U}(\Sigma), c} \pi$. We note, however, that the satisfaction relation of $\underline{\text{ML}}^+$ is not a restriction of the satisfaction relation of $\underline{\text{ML}}$. For example, if π were an existentially quantified pattern $\exists x \colon t . \pi_1$, then only the converse implication of the above equivalence would be ensured to hold. This follows because in $\underline{\text{ML}}$ every expansion of M may interpret in a non-deterministic manner the variable $x \colon t$; in order words, there is no guarantee that there exists an expansion of M in $\underline{\text{ML}}$ that satisfies π and is also a model of $\underline{\text{ML}}^+$.

---

[3] Technically, the quantification in $\underline{\text{ML}}^+$ is done only over variables that are interpreted in a deterministic manner. This means that every extension with variables over signature $\text{U}(\Sigma)$ (in $\underline{\text{ML}}$) corresponds to a deterministic extension of Σ in $\underline{\text{ML}}^+$.

3.3   Encoding Matching Logic into First-Order Logic

There exists a comorphism of institutions between $\flat\mathbf{ML}^+$, the institution obtained from $\underline{\mathrm{ML}}^+$ following Proposition 1, and $\mathbf{FOL}$, the institution of first-order logic. In short, the deterministic operations of any given $\flat\mathbf{ML}^+$-signature are preserved by the signature-translation component of the comorphism, while each non-deterministic operation is transformed into a new predicate. In this manner, the interpretation of first-order predicates corresponds to the interpretation of non-deterministic operations in multialgebras. Furthermore, the underlying sentence-translation map of the comorphism encodes each matching pattern to a corresponding universally quantified sentence over states having the same class as the pattern. We define $(\Phi, \alpha, \beta)\colon \flat\mathbf{ML}^+ \to \mathbf{FOL}$ as follows:

For Signatures: The underlying signature functor $\Phi\colon \mathbb{S}\mathrm{ig}^{\flat\mathbf{ML}^+} \to \mathbb{S}\mathrm{ig}^{\mathbf{FOL}}$ maps

- every $\flat\mathbf{ML}^+$ signature $\Sigma = (S, F, D)$ to the $\mathbf{FOL}$ signature $\Sigma' = (S', F', P')$ where $S' = S$, $F'_{w\to s} = D_{w\to s}$, $P'_\lambda = \emptyset$, and $P'_{ws} = F_{w\to s}$ for $ws \neq \lambda$.
- every $\flat\mathbf{ML}^+$-signature morphism $\phi\colon \Sigma_1 \to \Sigma_2$ to the $\mathbf{FOL}$-signature morphism $\phi' = (\phi'^{\mathrm{st}}, \phi'^{\mathrm{op}}, \phi'^{\mathrm{rel}})$, where $\phi'^{\mathrm{st}} = \phi^{\mathrm{st}}$, $\phi'^{\mathrm{op}} = \phi^{\mathrm{det}}$, and $\phi'^{\mathrm{rel}}_{ws} = \phi^{\mathrm{op}}_{w\to s}$, for $ws \neq \lambda$.

For Models: The model functor $\beta_\Sigma\colon \mathrm{Mod}^{\mathbf{FOL}}(\Phi(\Sigma)) \to \mathrm{Mod}^{\flat\mathbf{ML}^+}(\Sigma)$ given by a signature $\Sigma = (S, F, D)$ maps

- every first-order structure M' for $\Phi(\Sigma)$ to the multialgebra M whose carrier sets M_s are defined as M'_s for every sort $s \in S$, whose interpretations $M_\sigma\colon M_{s_1} \times \ldots \times M_{s_n} \to 2^{M_s}$ of function symbols $\sigma \in F_{s_1\ldots s_n\to s}$ are defined as $M_\sigma(m_1, \ldots, m_n) = \{m \in M_s \mid (m_1, \ldots, m_n, m) \in M'_\sigma\}$, and whose interpretations M_σ of function symbols $\sigma \in D_{w\to s}$ are given by the composition of M'_σ with the singleton-forming map $\{\_\}\colon M_s \to 2^{M_s}$, and
- every morphism of first-order structures $h'\colon M' \to N'$ in $\mathrm{Mod}^{\mathbf{FOL}}(\Phi(\Sigma))$ to a multialgebra morphism $h\colon \beta_\Sigma(M') \to \beta_\Sigma(N')$ given by $h_s = h'_s$, for every $s \in S$. We note that the fact that h' commutes with the interpretation of operation symbols suits the deterministic nature of the morphism h for the interpretation of operations in D, while its compatibility with the interpretation of predicate symbols guarantees the satisfaction of the morphism condition for multialgebras.

For Sentences: The sentence translation $\alpha_\Sigma\colon \mathrm{Sen}^{\flat\mathbf{ML}^+}(\Sigma) \to \mathrm{Sen}^{\mathbf{FOL}}(\Phi(\Sigma))$ given by a signature $\Sigma = (S, F, D)$ maps every Σ-pattern π to the sentence $\alpha_\Sigma(\pi) = \forall m\colon s.\mathrm{FOL}_\Sigma^{m\,:\,s}(\pi)$, where $s = \kappa_\Sigma(\pi)$, m is a first-order variable of sort s for the signature $\Phi(\Sigma)$, and $\mathrm{FOL}_\Sigma^{m\,:\,s}\colon \kappa_\Sigma^{-1}(s) \to \mathrm{Sen}^{\mathbf{FOL}}(\Phi(\Sigma) \uplus \{m\colon s\})$ is the sorted translation of sentences defined as follows:

We begin with a notation: for every operation symbol $\sigma \in (F \cup D)_{s_1, \ldots, s_n\to s}$, and every variables $m_i\colon s_i$ and $m\colon s$, we denote by $\sigma^=(m_1, \ldots, m_n, m)$ either the relational atom $\sigma(m_1, \ldots, m_n, m)$ if $\sigma \in F$, or the equational atom $\sigma(m_1, \ldots, m_n) = m$ if $\sigma \in D$.

– for every basic pattern $\pi \in (F \cup D)_{\lambda \to s}$, $\mathrm{FOL}_{\Sigma}^{m:s}(\pi) = \pi^{=}(m)$,
– for every basic pattern $\pi = \sigma(t_1, \ldots, t_n)$, with $\sigma \in (F \cup D)_{s_1, \ldots, s_n \to s}$,

$$\mathrm{FOL}_{\Sigma}^{m:s}(\pi) = \exists m_1 : s_1 \ldots \exists m_n : s_n . \mathrm{FOL}_{\Sigma}^{m_1:s_1}(t_1) \wedge \ldots \wedge \mathrm{FOL}_{\Sigma}^{m_n:s_n}(t_n)$$
$$\wedge \, \sigma^{=}(m_1, \ldots, m_n, m),$$

– for every pattern $\pi = \neg\pi_1$, $\mathrm{FOL}_{\Sigma}^{m:s}(\neg\pi_1) = \neg\mathrm{FOL}_{\Sigma}^{m:s}(\pi_1)$,
– for every pattern $\pi = \pi_1 \wedge \pi_2$, $\mathrm{FOL}_{\Sigma}^{m:s}(\pi_1 \wedge \pi_2) = \mathrm{FOL}_{\Sigma}^{m:s}(\pi_1) \wedge \mathrm{FOL}_{\Sigma}^{m:s}(\pi_2)$,
– for every pattern $\pi = \exists x : t.\pi_1$, where $\pi_1 \in \mathrm{Sen}^{\flat\mathbf{ML}^+}(\Sigma \uplus \{x : t\})$, we have $\mathrm{FOL}_{\Sigma}^{m:s}(\exists x : t.\pi_1) = \exists x : t.\xi_{\Sigma}(\mathrm{FOL}_{\Sigma \uplus \{x:t\}}^{m:s}(\pi_1))$, where ξ_{Σ} is a first-order signature morphism from $\Phi(\Sigma \uplus \{x : t\}) \uplus \{m : s\}$ to $\Phi(\Sigma) \uplus \{m : s\} \uplus \{x : t\}$ defined as the extension of $1_{\Phi(\Sigma)}$ that maps the matching-logic variable $x : t$ for the signature Σ to the first-order variable $x : t$ for the signature $\Phi(\Sigma) \uplus \{m : s\}$, and the first-order variable $m : s$ for the signature $\Phi(\Sigma \uplus \{x : t\})$ to the first-order variable $m : s$ but for the signature $\Phi(\Sigma)$.[4]

The naturality of α results from an analogous property for $\mathrm{FOL}_{\Sigma}^{m:s}$.

Proposition 3. *For every two $\flat\mathbf{ML}^+$ signatures Σ_1, Σ_2, signature morphism $\phi : \Sigma_1 \to \Sigma_2$, and variable $m : s$ for Σ_1, the following diagram commutes.*

$$
\begin{array}{ccc}
\kappa_{\Sigma_1}^{-1}(s) & \xrightarrow{\ \mathrm{FOL}_{\Sigma_1}^{m:s}\ } & \mathrm{Sen}^{\mathbf{FOL}}(\Phi(\Sigma_1) \uplus \{m : s\}) \\[4pt]
{\scriptstyle \mathrm{Sen}^{\flat\mathbf{ML}^+}(\phi)(\_)} \Big\downarrow & & \Big\downarrow {\scriptstyle \mathrm{Sen}^{\mathbf{FOL}}(\Phi(\phi)^m)} \\[4pt]
\kappa_{\Sigma_2}^{-1}(\phi^{\mathrm{st}}(s)) & \xrightarrow[\ \mathrm{FOL}_{\Sigma_2}^{m:\phi^{\mathrm{st}}(s)}\ ]{} & \mathrm{Sen}^{\mathbf{FOL}}(\Phi(\Sigma_2) \uplus \{m : \phi^{\mathrm{st}}(s)\})
\end{array}
$$

Satisfaction Condition. In order to show that the definitions of the components of the comorphism given above guarantee that the satisfaction condition holds, it suffices to know that Proposition 4 holds.

Proposition 4. *For every $\flat\mathbf{ML}^+$ signature Σ, every first-order structure M for $\Phi(\Sigma)$, and every Σ-pattern π of sort s, $M_{\mathrm{FOL}_{\Sigma}^{m:s}(\pi)} = \beta_{\Sigma}(M)_{\pi}$.*

This can be easily shown by induction on the structure of π, starting with the base case of patterns $\pi \in F_{\lambda \to s}$, for which $M_{\mathrm{FOL}_{\Sigma}^{m:s}(\pi)}$ is the set of states $m \in M_s$ such that $(M, m) \models \pi(m)$, that is M_{π}, and, by definition, $\beta_{\Sigma}(M)_{\pi} = M_{\pi}$.

4   Reachability Logic

In order to capture reachability logic [21] as an institution, we first define an abstract, parameterised institution over an arbitrary stratified institution with classes, which necessarily has to enjoy properties such as the existence of a

---

[4] We recall from the definitions of the institutions of matching and first-order logic that from a technical point of view, variables are triples, consisting of name, sort, and signature over which they are defined. Consequently, the signature morphism ξ_{Σ} maps $\langle x, t, \Sigma \rangle$ to $\langle x, t, \Phi(\Sigma) \uplus \langle m, s, \Phi(\Sigma) \rangle \rangle$, and $\langle m, s, \Phi(\Sigma \uplus x) \rangle$ to $\langle m, s, \Phi(\Sigma) \rangle$.

quantification space, model amalgamation, and preservation of pushouts by the class functor. We then obtain the concrete version of reachability logic that underlies the $\mathbb{K}$ framework by instantiating the parameter of the abstract version with $\underline{\mathrm{ML}}^+$, the stratified institution with classes of matching logic, which we show to satisfy the desired properties.

4.1   Abstract Reachability Logic

We formalise reachability logic in two steps: we begin by describing a sub-institution of reachability logic whose sentences are all atomic (reachability atoms), and we subsequently extend it by adding logical connectives and quantifiers through a general universal-quantification construction.

   To define atomic abstract reachability logic we first describe it as a pre-institution [22] whose construction is based upon a stratified institution with classes. This amounts to defining the same elements as those comprised by an institution but without imposing the requirement of the satisfaction condition.

   Throughout this section we assume an arbitrary, but fixed stratified institution with classes $\underline{\mathrm{M}} = (\mathbb{S}\mathrm{ig}^{\underline{\mathrm{M}}}, \mathrm{Cls}^{\underline{\mathrm{M}}}, \mathrm{Sen}^{\underline{\mathrm{M}}}, \mathrm{Mod}^{\underline{\mathrm{M}}}, [\![\_]\!]^{\underline{\mathrm{M}}}, \models^{\underline{\mathrm{M}}})$. This serves as a parameter for all the constructions below.

Signatures. The category of signatures of atomic abstract reachability logic, denoted by $\mathbb{S}\mathrm{ig}^{\mathbf{ARL}(\underline{\mathrm{M}})}$, is the same as the category of signatures of $\underline{\mathrm{M}}$.

Sentences. For every signature Σ, $\mathrm{Sen}^{\mathbf{ARL}(\underline{\mathrm{M}})}(\Sigma)$ is the set of pairs of sentences of the stratified institution with classes, denoted by $\pi_1 \Rightarrow \pi_2$, where $\pi_1, \pi_2 \in \mathrm{Sen}^{\underline{\mathrm{M}}}(\Sigma)$. The translation of such a sentence $\pi_1 \Rightarrow \pi_2$ along a signature morphism $\phi\colon \Sigma \to \Sigma'$ is defined as the pair of its translated components according to $\mathrm{Sen}^{\underline{\mathrm{M}}}(\phi)$: $\mathrm{Sen}^{\mathbf{ARL}(\underline{\mathrm{M}})}(\phi)(\pi_1 \Rightarrow \pi_2) = \mathrm{Sen}^{\underline{\mathrm{M}}}(\phi)(\pi_1) \Rightarrow \mathrm{Sen}^{\underline{\mathrm{M}}}(\phi)(\pi_2)$.

Example. If we instantiate the parameter $\underline{\mathrm{M}}$ with the stratified institution with classes $\underline{\mathrm{ML}}^+$, the sentences of $\mathbf{ARL}(\underline{\mathrm{ML}}^+)$ will only capture atomic $\mathbb{K}$ semantic rules, i.e. without quantification and side conditions. This means we could only express atomic rules in the specification of the simple imperative programming language IMP, like **rule** ! true => $\mathrm{not}_{\mathrm{Bool}}$ true.

Models. The reachability models of a signature Σ, given by the $\mathrm{Mod}^{\mathbf{ARL}(\underline{\mathrm{M}})}$ functor, are pairs $(M, \rightsquigarrow)$ of Σ-models M of the underlying stratified institution with classes, and families of preorders $\rightsquigarrow_c \subseteq [\![M]\!]_{\Sigma,c} \times [\![M]\!]_{\Sigma,c}$ indexed by the classes of the signature. The model homomorphisms $h\colon (M_1, \rightsquigarrow_1) \to (M_2, \rightsquigarrow_2)$ are defined as the morphisms between the $\underline{\mathrm{M}}$-models M_1 and M_2 that preserve the preorders: for every $c \in \mathrm{Cls}(\Sigma)$, the function $[\![h]\!]_{\Sigma,c}$ from $([\![M_1]\!]_{\Sigma,c}, \rightsquigarrow_1)$ to $([\![M_2]\!]_{\Sigma,c}, \rightsquigarrow_2)$ is monotone. This allows $\mathrm{Mod}^{\mathbf{ARL}(\underline{\mathrm{M}})}(\Sigma)$ to inherit the identities and the composition of model homomorphisms of $\mathrm{Mod}^{\underline{\mathrm{M}}}(\Sigma)$.

   The model reduct $\mathrm{Mod}^{\mathbf{ARL}(\underline{\mathrm{M}})}(\phi)\colon \mathrm{Mod}^{\mathbf{ARL}(\underline{\mathrm{M}})}(\Sigma') \to \mathrm{Mod}^{\mathbf{ARL}(\underline{\mathrm{M}})}(\Sigma)$ given by a signature morphism $\phi\colon \Sigma \to \Sigma'$ is defined as

- $\mathrm{Mod}^{\mathbf{ARL}(\underline{\mathrm{M}})}(\phi)(M', \rightsquigarrow') = (\mathrm{Mod}^{\underline{\mathrm{M}}}(\phi)(M'), \rightsquigarrow)$ for every Σ'-model $(M', \rightsquigarrow')$, where $\rightsquigarrow_c \subseteq [\![M'\!\restriction_\phi]\!]_{\Sigma,c} \times [\![M'\!\restriction_\phi]\!]_{\Sigma,c}$ is the reflexive and transitive closure of $[\![M']\!]_{\phi,c}(\rightsquigarrow'_{\phi(c)})$, which will be further denoted by $\to_c$,

– $\mathrm{Mod}^{\mathbf{ARL}(\underline{\mathrm{M}})}(\phi)(h')$ is simply $\mathrm{Mod}^{\underline{\mathrm{M}}}(\phi)(h')$ for every two Σ'-models M_1', M_2', and every model homomorphism $h' : (M_1', \leadsto_1') \to (M_2', \leadsto_2')$.

Satisfaction Relation. The satisfaction relation between any model $(M, \leadsto)$ and any sentence $\pi_1 \Rightarrow \pi_2$ is defined as follows: $(M, \leadsto) \models_{\Sigma}^{\mathbf{ARL}(\underline{\mathrm{M}})} \pi_1 \Rightarrow \pi_2$ if and only if for every $m \in [\![M]\!]_{\Sigma,c}$ such that $M(\models^{\underline{\mathrm{M}}})_{\Sigma,c}^m \pi_1$, there exists $n \in [\![M]\!]_{\Sigma,c}$ such that $M(\models^{\underline{\mathrm{M}}})_{\Sigma,c}^n \pi_2$, and $m \leadsto_c n$.

Corollary 1. $\mathbf{ARL}(\underline{\mathrm{M}}) = (\mathbb{S}\mathrm{ig}^{\mathbf{ARL}(\underline{\mathrm{M}})}, \mathrm{Sen}^{\mathbf{ARL}(\underline{\mathrm{M}})}, \mathrm{Mod}^{\mathbf{ARL}(\underline{\mathrm{M}})}, \models^{\mathbf{ARL}(\underline{\mathrm{M}})})$ *is a pre-institution.*

The direct implication of the satisfaction condition holds unconditionally.

Proposition 5. *For every signature morphism $\phi : \Sigma \to \Sigma'$, every class $c \in \mathrm{Cls}(\Sigma)$, every model $(M', \leadsto') \in |\mathrm{Mod}^{\mathbf{ARL}(\underline{\mathrm{M}})}(\Sigma')|$, and every sentence $\pi_1 \Rightarrow \pi_2$,*

$$(M', \leadsto') \models_{\Sigma'}^{\mathbf{ARL}(\underline{\mathrm{M}})} \phi(\pi_1 \Rightarrow \pi_2) \ \text{implies} \ (M', \leadsto')\restriction_\phi \models_{\Sigma}^{\mathbf{ARL}(\underline{\mathrm{M}})} \pi_1 \Rightarrow \pi_2.$$

The converse of Proposition 5 holds if the stratification of the underlying institution of $\mathbf{ARL}(\underline{\mathrm{M}})$ satisfies a property similar to that of lifting relations from [7, Chapter 9].

Proposition 6. *If in $\mathbf{ARL}(\underline{\mathrm{M}})$, for every signature morphism $\phi : \Sigma \to \Sigma'$, every class $c \in \mathrm{Cls}(\Sigma)$, every Σ'-model $(M', \leadsto')$, and every states $m' \in [\![M']\!]_{\Sigma', \phi(c)}$ and $n \in [\![M'\restriction_\phi]\!]_{\Sigma,c}$ such that $[\![M']\!]_{\phi,c}(m') \leadsto_c n$, there exists $n' \in [\![M']\!]_{\Sigma', \phi(c)}$ such that $m' \leadsto'_{\phi(c)} n'$ and $[\![M']\!]_{\phi,c}(n') = n$, then*

$$(M', \leadsto')\restriction_\phi \models_{\Sigma}^{\mathbf{ARL}(\underline{\mathrm{M}})} \pi_1 \Rightarrow \pi_2 \ \text{implies} \ (M', \leadsto') \models_{\Sigma'}^{\mathbf{ARL}(\underline{\mathrm{M}})} \phi(\pi_1 \Rightarrow \pi_2),$$

for every sentence $\pi_1 \Rightarrow \pi_2$.

Corollary 2. *If the stratified institution with classes $\underline{\mathrm{M}}$ satisfies the hypothesis of Proposition 6, then $\mathbf{ARL}(\underline{\mathrm{M}})$ is an institution.*

In most concrete examples of stratified institutions with classes the natural transformations $[\![M']\!]_{\phi,c}$ of the stratification are bijective, or even identities (see for example the definitions of $\underline{\mathrm{ML}}^+$ and $\underline{\mathrm{CTL}}$). Therefore, the hypothesis of Proposition 6 is usually satisfied, entailing that $\mathbf{ARL}(\underline{\mathrm{M}})$ is an institution.

We have hitherto defined only an atomic fragment of the desired institution of abstract reachability logic. To describe the construction of the institution with universally quantified Horn-clause sentences over the atomic sentences of $\mathbf{ARL}(\underline{\mathrm{M}})$, we use the notion of quantification space originating from [8].

Definition 7. *For any category $\mathbb{S}\mathrm{ig}$ a class of arrows $\mathcal{D} \subseteq \mathbb{S}\mathrm{ig}$ is called a* quantification space *if, for any $\chi : \Sigma \to \Sigma' \in \mathcal{D}$ and $\varphi : \Sigma \to \Sigma_1$ there exists a designated pushout*

$$\Sigma \xrightarrow{\varphi} \Sigma_1$$
$$\chi \downarrow \qquad \downarrow \chi(\varphi)$$
$$\Sigma' \xrightarrow{\varphi[\chi]} \Sigma_1'$$

with $\chi(\varphi) \in \mathcal{D}$ and such that the horizontal composition of these designated pushouts is also a designated pushout, i.e. for the pushouts in the diagram below

$$\Sigma \xrightarrow{\varphi} \Sigma_1 \xrightarrow{\theta} \Sigma_2$$
$$\chi \downarrow \qquad \downarrow \chi(\varphi) \qquad \downarrow \chi(\varphi)(\theta)$$
$$\Sigma' \xrightarrow{\varphi[\chi]} \Sigma_1' \xrightarrow{\theta[\chi(\varphi)]} \Sigma_2'$$

$\varphi[\chi] \, ; \theta[\chi(\varphi)] = (\varphi \, ; \theta)[\chi]$ *and* $\chi(\varphi)(\theta) = \chi(\varphi \, ; \theta)$, *and such that* $\chi(1_\Sigma) = \chi$ *and* $1_\Sigma[\chi] = 1_{\Sigma'}$. *A quantification space $\mathcal{D}$ for* $\mathbb{S}$ig *is* adequate *for a functor* Mod: $\mathbb{S}$ig$^{\mathrm{op}} \to \mathbb{C}$at *when the aforementioned designated pushouts are weak amalgamation squares for* Mod. *A quantification space $\mathcal{D}$ for* $\mathbb{S}$ig *is called* adequate *for an institution if it is adequate for its model functor.*

Proposition 7. *For any institution $\mathbf{I}$ with an adequate quantification space $\mathcal{D}$, the following data defines an institution, called the* institution of universally $\mathcal{D}$-quantified Horn clauses over $\mathbf{I}$, *and denoted* $\mathbf{HCL(I)}$:

– $\mathbb{S}$ig$^{\mathbf{HCL(I)}} = \mathbb{S}ig^{\mathbf{I}}$,
– Mod$^{\mathbf{HCL(I)}} = Mod^{\mathbf{I}}$,
– Sen$^{\mathbf{HCL(I)}}(\Sigma)$

$$= \{\forall \chi . \rho_1' \wedge \ldots \wedge \rho_n' \to \rho' \mid (\chi \colon \Sigma \to \Sigma') \in \mathcal{D} \text{ and } \rho_i', \rho' \in \mathrm{Sen}^{\mathbf{I}}(\Sigma')\},$$

for every signature Σ
– Sen$^{\mathbf{HCL(I)}}(\varphi)(\forall \chi . \rho_1' \wedge \ldots \wedge \rho_n' \to \rho')$

$$= \forall \chi(\varphi) . \mathrm{Sen}^{\mathbf{I}}(\varphi[\chi])(\rho_1') \wedge \ldots \wedge \mathrm{Sen}^{\mathbf{I}}(\varphi[\chi])(\rho_n') \to \mathrm{Sen}^{\mathbf{I}}(\varphi[\chi])(\rho'),$$

for every signature morphism $\varphi \colon \Sigma \to \Sigma_1$
– $M \models_\Sigma^{\mathbf{HCL(I)}} \forall \chi . \rho_1' \wedge \ldots \wedge \rho_n' \to \rho'$

iff for all χ-expansions M' of M, $M' \models_{\Sigma'}^{\mathbf{I}} \rho'$ if $M' \models_{\Sigma'}^{\mathbf{I}} \rho_i'$ for $i = \overline{1, n}$.

To build an institution with universally quantified sentences over $\mathbf{ARL}(\underline{M})$ as described in Proposition 7, we need to ensure that $\mathbf{ARL}(\underline{M})$ satisfies its hypothesis. This cannot be guaranteed in general, because $\underline{M}$ is abstract. Nevertheless, we can obtain an appropriate set of hypotheses for the underlying stratified institution $\underline{M}$ that allow us to apply Proposition 7:

– the existence of a quantification space for $\mathbf{ARL}(\underline{M})$ is guaranteed by the existence of a quantification space for $\underline{M}$, as the categories $\mathbb{S}$ig$^{\mathbf{ARL}(\underline{M})}$ and $\mathbb{S}$ig$^{\underline{M}}$ are equal,

– the fact that **ARL**($\underline{\text{M}}$) has weak model amalgamation follows from the weak model amalgamation property of $\underline{\text{M}}$ (see Definition 8 below) and the preservation of pushouts by the class functor of $\underline{\text{M}}$ (see Proposition 8 below).

Definition 8. *A stratified institution with classes $\underline{\text{M}}$ has* (weak) model amalgamation *whenever its corresponding institution $\flat\mathbf{M}$ has this property.*

Proposition 8. *For every stratified institution with classes $\underline{\text{M}}$ having (weak) model amalgamation such that its class functor* Cls *preserves pushouts, **ARL**($\underline{\text{M}}$) has (weak) model amalgamation.*

Corollary 3. *If $\underline{\text{M}}$ has an adequate quantification space and a pushout preserving class functor, then **HCL**(**ARL**($\underline{\text{M}}$)) is an institution.*

4.2   Defining Reachability over Matching Logic

In order to capture reachability logic in its original, concrete form, we must instantiate the parameter of the institution **HCL**(**ARL**($\underline{\text{M}}$)) defined above, with the stratified institution $\underline{\text{ML}}^+$. To this end, we first point out that by adding variables as deterministic constants to the signatures of $\underline{\text{ML}}^+$ we obtain a quantification space. Furthermore, to show that the quantification space is adequate, we use the property of model amalgamation of the comorphism (Φ, α, β) between $\flat\mathbf{ML}^+$ and **FOL** defined in Sect. 3.3.

Proposition 9. $\underline{\text{ML}}^+$ *has pushouts of signatures. Moreover, its class functor preserves pushouts.*

Example. Let us consider the $\mathbb{K}$ definition of the IMP programming language. By splitting the syntax module into three modules, AEXP, BEXP and IMP-SYNTAX importing the two expressions modules, we have an immediate and natural example of a pushout of signatures: as both the AExp and BExp modules import the BUILT-IN module containing the built-in sorts and corresponding operations of $\mathbb{K}$, we need to construct the pushout of their signatures in order to obtain the signature of the module IMP-SYNTAX.

$$
\begin{array}{ccc}
\text{BUILT-IN}^{\text{Sig}} & \xrightarrow{\;\subseteq\;} & \text{AEXP}^{\text{Sig}} \\
{\scriptstyle\subseteq}\downarrow & & \downarrow{\scriptstyle\subseteq} \\
\text{BEXP}^{\text{Sig}} & \xrightarrow[\;\subseteq\;]{} & \text{IMP-SYNTAX}^{\text{Sig}}
\end{array}
$$

Proposition 10. *In $\underline{\text{ML}}^+$, the family of extensions with deterministic constants forms a quantification space.*

The following definition originates from [4].

Definition 9. *An institution comorphism $(\Phi, \alpha, \beta)\colon \mathbf{I} \to \mathbf{I}'$ has* weak model amalgamation *if for every $\mathbf{I}$-signature morphism $\varphi\colon \Sigma \to \Sigma'$, every Σ'-model M', and every $\Phi(\Sigma)$-model N such that $\beta_\Sigma(N) = M'\!\restriction_\varphi$, there exists a $\Phi(\Sigma')$-model N' such that $\beta_{\Sigma'}(N') = M'$ and $N'\!\restriction_{\Phi(\varphi)} = N$. We say that $(\Phi, \alpha, \beta)\colon \mathbf{I} \to \mathbf{I}'$ has* model amalgamation *when N' is required to be unique.*

Remark 1. $\underline{\mathrm{ML}}^+$ *has model amalgamation. Let us first note that the comorphism (Φ, α, β) between the institution $\flat\mathbf{ML}^+$ and $\mathbf{FOL}$ defined in the previous section has model amalgamation. This property holds trivially since the model reduction functors β_Σ are isomorphisms of categories, for every signature Σ. As the institution of $\mathbf{FOL}$ also has model amalgamation, we can use a general result of institution theory to deduce that $\flat\mathbf{ML}^+$ has model amalgamation.*

Corollary 4. $\mathbf{HCL}(\mathbf{ARL}(\underline{\mathrm{ML}}^+))$ *is an institution.*

4.3   Encoding Reachability Logic into First-Order Logic

For any institution $\mathbf{ARL}(\underline{M})$ defined over a stratified institution with classes $\underline{M}$, there exists a comorphism of institutions between $\mathbf{ARL}(\underline{M})$ and $\mathbf{FOL}^{\mathrm{pres}}$, the institution of presentations over first-order logic, whenever there exists a comorphism of institutions (Φ, α, β) between $\flat\mathbf{M}$ and $\mathbf{FOL}$ such that:

- the classes of a signature in $\mathbb{S}\mathrm{ig}^{\flat\mathbf{M}}$ are given by the sorts of its translation to $\mathbf{FOL}$: $\mathrm{Cls} = \Phi\,;\mathrm{St}$, where St is the forgetful functor $\mathrm{St}\colon \mathbb{S}\mathrm{ig}^{\mathbf{FOL}} \to \mathbb{S}\mathrm{et}$,
- for every signature Σ, $\alpha_\Sigma(\pi) = \forall m\colon s.\mathrm{FOL}_\Sigma^{m\,:\,s}(\pi)$, for every π of class s,[5]
- for every $N \in |\mathrm{Mod}^{\mathbf{FOL}}(\Phi(\Sigma))|$, and every $s \in \mathrm{Cls}(\Sigma)$, $N_s = [\![\beta_\Sigma(N)]\!]_{\Sigma,s}$.

The signature-translation component of the comorphism encodes the reachability relation through the addition of new preorder predicates and corresponding axioms for each class of the signature. The new predicates determine relations on reachable states that define the preorder-family component of a reachability model. The sentence component of the comorphism translates each reachability statement between two patterns to a sentence that expresses the existence of a reachable state for the target pattern for every state of the source pattern. We define the comorphism $(\Phi^R, \alpha^R, \beta^R)\colon \mathbf{ARL}(\underline{M}) \to \mathbf{FOL}^{\mathrm{pres}}$ as follows:

For Signatures: The signature functor $\Phi^R\colon \mathbb{S}\mathrm{ig}^{\mathbf{ARL}(\underline{M})} \to \mathbb{S}\mathrm{ig}^{\mathbf{FOL}^{\mathrm{pres}}}$ maps every signature Σ of $\mathbf{ARL}(\underline{M})$ to $\Phi^R(\Sigma) = (\mathrm{Reach}(\Sigma), E)$, where

- $\mathrm{Reach}(\Sigma)$ denotes the first-order signature obtained by adding to $\Phi(\Sigma) = (S', F', P')$ a predicate *reach* of arity $s\,s$ for every sort $s \in S'$, and
- E is a set of axioms that define the predicates *reach* as preorders:
  $\{\forall x\colon s.reach(x,x), \forall x, y, z\colon s.reach(x,y) \wedge reach(y,z) \to reach(x,z) \mid s \in S'\}$.

For Sentences: For every signature Σ of $\mathbf{ARL}(\underline{M})$, the sentence translation function $\alpha_\Sigma^R\colon \mathrm{Sen}^{\mathbf{ARL}(\underline{M})}(\Sigma) \to \mathrm{Sen}^{\mathbf{FOL}}(\mathrm{Reach}(\Sigma))$ maps every Σ-sentence $\pi_1 \Rightarrow \pi_2$ to $\alpha_\Sigma^R(\pi_1 \Rightarrow \pi_2) = \forall m\colon s.\mathrm{FOL}_\Sigma^{m\,:\,s}(\pi_1) \to \exists n\colon s.\mathrm{FOL}_\Sigma^{n\,:\,s}(\pi_2) \wedge reach(m,n)$.

For Models: For every signature Σ of $\mathbf{ARL}(\underline{M})$, $\beta_\Sigma^R\colon \mathrm{Mod}^{\mathbf{FOL}}(\mathrm{Reach}(\Sigma), E) \to \mathrm{Mod}^{\mathbf{ARL}(M)}(\Sigma)$ is the model functor that maps every first-order structure $N \in$

---

[5] Note that, in this case, $\mathrm{FOL}_\Sigma^{m\,:\,s}(\pi)$ is just a notation, and it should not be confused with the first-order sentence described in the previous section, for which we would need to instantiate $\underline{M}$ with $\underline{\mathrm{ML}}^+$.

$|\text{Mod}^{\textbf{FOL}}(\text{Reach}(\Sigma), E)|$ to the model $(M, \rightsquigarrow) \in |\text{Mod}^{\textbf{ARL}(M)}(\Sigma)|$, given by $M = \beta_\Sigma(N) \in |\text{Mod}^{\underline{M}}(\Sigma)|$ and $\rightsquigarrow_s = \{(m, n) \mid (N, m, n) \models \text{reach}(x, y)\}$, for every sort s. Note that $\rightsquigarrow_s$ is well-defined as $\text{Cls} = \Phi \, ; \text{St}$ and $N_s = [\![M]\!]_{\Sigma, s}$.

To encode the Horn-clause reachability logic of Corollary 4 (defined over $\underline{\text{ML}}^+$) into first-order logic, it suffices to notice that the comorphism considered in Sect. 3.3, $(\Phi, \alpha, \beta) \colon \flat \textbf{ML}^+ \to \textbf{FOL}$, satisfies all of the above requirements, and thus can be extended to a comorphism $(\Phi^R, \alpha^R, \beta^R) \colon \textbf{ARL}(\underline{\text{ML}}^+) \to \textbf{FOL}^{\text{pres}}$. This can be further extended to an encoding of $\textbf{HCL}(\textbf{ARL}(\underline{\text{ML}}^+))$ into $\textbf{FOL}^{\text{pres}}$ through the use of a general result about Horn-clause institutions.

Proposition 11. *Let $\textbf{I}$ and $\textbf{I}'$ be institutions equipped with quantification spaces. Every comorphism of institutions $(\Phi, \alpha, \beta) \colon \textbf{I} \to \textbf{I}'$ that has weak model amalgamation, and for which Φ preserves the quantification space of $\textbf{I}$, can be extended to a comorphism of institutions between $\textbf{HCL}(\textbf{I})$ and $\textbf{HCL}(\textbf{I}')$.*

5 Conclusions and Future Research

In this work, we proposed an institutional formalisation of the logical systems that underlie the $\mathbb{K}$ semantic framework. These logical systems account for the structural properties of program configurations (through matching logic), and changes of these configurations (through reachability logic).

Our work sets the foundation for integrating the $\mathbb{K}$ semantic framework into heterogeneous institution-based toolsets, allowing us to exploit the combined potential of the $\mathbb{K}$ tool and of other software tools such as the MiniSat solver, the SPASS automated prover or the Isabelle interactive proof assistant. Having both matching and reachability logic defined as institutions allows us to integrate them into the logic graphs of institution-based heterogeneous specification languages such as HETCASL [16]. As an immediate result, the $\mathbb{K}$ framework can inherit the powerful module systems developed for specifications built over arbitrary institutions, with dedicated operators for aggregating, renaming, extending, hiding and parameterising modules. In addition, this will enable us to combine reachability logic and the tool support provided by $\mathbb{K}$ with other logical systems and tools. Towards that end, as a preliminary effort to integrate the $\mathbb{K}$ framework into HETS [17], we described comorphisms from matching and reachability logic to the institution of first-order logic.

Another line of research concerns the development of $\mathbb{K}$ from a purely formal-specification perspective, including for example, studies on modularisation and initial semantics. Within this context, verification can be performed based on the proof systems that have already been defined for $\mathbb{K}$.

References

1. The IMP language. http://www.kframework.org/imgs/releases/k/tutorial/1_k/2_imp/lesson_5/imp.pdf
2. Aiguier, M., Diaconescu, R.: Stratified institutions and elementary homomorphisms. Inf. Process. Lett. **103**(1), 5–13 (2007)

3. Bogdănaş, D., Roşu, G.: K-Java: a complete semantics of Java. In: Proceedings of the 42nd Symposium on Principles of Programming Languages, POPL 2015. ACM (2015)
4. Borzyszkowski, T.: Logical systems for structured specifications. Theor. Comput. Sci. **286**(2), 197–245 (2002)
5. Chiriţă, C.E.: An institutional foundation for the K semantic framework. Master's thesis, University of Bucharest (2014)
6. Şerbănuţă, T.F., Arusoaie, A., Lazar, D., Ellison, C., Lucanu, D., Roşu, G.: The K primer (version 3.3). Electron. Notes Theor. Comput. Sci. **304**, 57–80 (2014)
7. Diaconescu, R.: Institution-independent Model Theory. Studies in Universal Logic. Springer, London (2008). http://books.google.ro/books?id=aEpn60-EDXwC
8. Diaconescu, R.: Quasi-boolean encodings and conditionals in algebraic specification. J. Logic Algebraic Program. **79**(2), 174–188 (2010)
9. Ellison, C., Roşu, G.: An executable formal semantics of C with applications. In: Proceedings of the 39th Symposium on Principles of Programming Languages (POPL 2012), pp. 533–544. ACM (2012)
10. Goguen, J.A., Burstall, R.M.: Institutions: abstract model theory for specification and programming. J. ACM **39**(1), 95–146 (1992)
11. Goguen, J.A., Diaconescu, R.: An Oxford survey of order sorted algebra. Math. Struct. Comput. Sci. **4**(3), 363–392 (1994)
12. Guth, D.: A formal semantics of python 3.3. Master's thesis, University of Illinois at Urbana-Champaign, July 2013
13. Lamo, Y.: The Institution of Multialgebras-a general framework for algebraic software development. Ph.D. thesis, University of Bergen (2003)
14. Lane, S.M.: Categories for the Working Mathematician. Springer, New York (1998). http://books.google.ro/books?id=eBvhyc4z8HQC
15. Meseguer, J.: General logics. In: Ebbinghaus, H.D., Fernandez-Prida, J., Garrido, M., Lascar, D., Artalejo, M.R. (eds.) Logic Colloquium 1987 Proceedings of the Colloquium held in Granada, Studies in Logic and the Foundations of Mathematics, vol. 129, pp. 275–329. Elsevier (1989)
16. Mossakowski, T.: HetCasl-heterogeneous specification. Language summary (2004)
17. Mossakowski, T., Maeder, C., Lüttich, K.: The heterogeneous tool set, Hets. In: Grumberg, O., Huth, M. (eds.) TACAS 2007. LNCS, vol. 4424, pp. 519–522. Springer, Heidelberg (2007)
18. Roşu, G.: Matching logic: a logic for structural reasoning. Technical report, University of Illinois, January 2014. http://hdl.handle.net/2142/47004,
19. Roşu, G., Şerbănuţă, T.F.: An overview of the K semantic framework. J. Log. Algebraic Program. **79**(6), 397–434 (2010)
20. Roşu, G., Şerbănuţă, T.F.: K overview and SIMPLE case study. Electron. Notes Theoret. Comput. Sci. **304**, 3–56 (2014)
21. Roşu, G., Ştefănescu, A., Ciobâcă, Ş., Moore, B.M.: One-path reachability logic. In: Proceedings of the 28th Symposium on Logic in Computer Science (LICS 2013), pp. 358–367. IEEE, June 2013
22. Salibra, A., Scollo, G.: Interpolation and compactness in categories of preinstitutions. Math. Struct. Comput. Sci. **6**(3), 261–286 (1996)
23. Sannella, D., Tarlecki, A.: Foundations of Algebraic Specification and Formal Software Development. Monographs in Theoretical Computer Science. Springer, Heidelberg (2012)
24. Tarlecki, A.: Moving between logical systems. In: Haveraaen, M., Dahl, O.-J., Owe, O. (eds.) Abstract Data Types 1995 and COMPASS 1995. LNCS, vol. 1130, pp. 478–502. Springer, Heidelberg (1996)

Recent Trends in Algebraic Development Techniques
22nd International Workshop, WADT 2014, Sinaia,
Romania, September 4-7, 2014, Revised Selected
Papers
Codescu, M.; Diaconescu, R.; Țuțu, I. (Eds.)
2015, XI, 171 p. 35 illus. in color., Softcover
ISBN: 978-3-319-28113-1