Hybridisation at work

Renato Neves[1], Alexandre Madeira[2], Manuel A. Martins[3], and Luís S. Barbosa[1]

[1] HASLab - INESC TEC & Univ. Minho
{nevrenato@gmail.com, lsb@di.uminho.pt}
[2] HASLab - INESC TEC & Dep. Mathematics, Univ. Aveiro & Critical Software S.A.
madeira@ua.pt
[3] CIDMA - Dep. Mathematics, Univ. Aveiro
martins@ua.pt

Abstract. This paper presents the encoding of the hybridisation method proposed in [MMDB11,DM13] into the HETS platform.

Keywords: Hybrid logics; institutions; reconfigurable systems.

1 Introduction and purpose

Hybrid logics [Bla00] are a brand of modal logics that provides appropriate syntax for the possible worlds semantics through nominals. In particular, it adds to the modal description of transition structures the ability to refer to specific states. This paves the way to an expressive framework for specifying complex software able to evolve through different execution configurations. In a number of papers, starting with [MFMB11], the foundations and methodological aspects of such a framework for *reconfigurability* have been developed, leading to a two-stage method:

- *globally* the system's dynamics is represented by a transition structure described in a hybrid language, whose states correspond to possible configurations;
- *locally* each state is endowed with a structure modeling the respective configuration specification.

The logic used locally depends on the application requirements. Typical candidates are equational, partial algebra or first-order logic ($\mathcal{FOL}$), but one may equally resort to multivalued logics or even to hybrid logic itself equipping, in the last case, each state with another (local) transition system. Instead of fixing a particular hybrid logic, a systematic method to develop on top of each local logic, the characteristic features of hybrid logic was proposed. This process is called *hybridisation* and was characterised in [MMDB11,DM13], framed in the context of the theory of institutions [GB92] to achieve greater generality.

The *hybridisation* process abstracts away the syntactic and semantic details, that are independent of the very essence of the hybrid logic idea. This has a number of benefits, even if paying the price of a heavy notational burden. One is the focus on the essential, the theoretical development not being hindered by irrelevant details. Another one concerns its applicability to a wide number of concrete instances, since all of them

could be regarded as combinations between concrete versions of hybrid logics with other logical systems. In this sense, the hybridisation method can be seen as a source of logics for the specification of *reconfigurable* systems [MFMB11].

Institutions provide a systematic way to relate logics and transport results from one to another, which means that a theorem prover for the latter can be used to reason about specifications written in the former. This is achieved through a special class of maps between institutions, referred to conservative comorphisms.

This paper reports on the implementation of the method introduced in [MMDB11] along two different directions. Firstly the general hybridisation method is incorporated in the HETS platform [MML07] — parsing and static analysis for the hybridisation of any base institution already supported in HETS being provided. Secondly, the comorphism $\mathcal{H}$CASL $\rightarrow$ CASL is implemented, offering effective tool support for proofs on a number of $\mathcal{H}$CASL-sub-institutions, namely $\mathcal{HFOL}$ and hybrid propositional logic[4]. This provides for free the proof support environment of a particularly well established logic. Naturally, the final goal is to have in HETS for free a comorphism from a hybridised logic $\mathcal{HI}$ to $\mathcal{FOL}$ given that a comorphism exists from the base logic $\mathcal{I}$ to $\mathcal{FOL}$.

HETS has been described as a "motherboard" where different "expansion cards" can be plugged. These pieces are individual logics (with their particular analysers and proof tools) as well as logic translations. To make them *compatible*, logics are formalised as institutions and translations as comorphisms. Therefore, the integration of the hybrid specifications on the HETS platform is legitimate, since all formal requirements (e.g., that institutions exist, that a comorphism can be defined, etc.) are already guaranteed by the hybridisation process itself.

The code for this extension to HETS, as well as a set of hybrid specification examples, is available from GITHUB (https://github.com/nevrenato/Hets_Fork). Additionally a ready–to–use HETS system is provided in a virtual machine available at SUGARSYNC (https://www.sugarsync.com/pf/D7620475_67336482_6511440).

2 Hybridisation as a plug-in to HETS

2.1 Hybridisation of CASL

The hybridisation process was first incorporated into HETS through its direct application to what is the platform *lingua franca*: CASL[MHST03]. A comorphism from the outcome $\mathcal{H}$CASL to CASL was also defined. Thus, assisted proof support for $\mathcal{H}$CASL becomes available for free. $\mathcal{H}$CASL specifications add to the usual ones in CASL a declaration of nominals and modalities. Sentences include the typical *hybrid* machinery and quantification over nominals. Thus, the respective grammar is extended as follows:

CFor' = **HFor** | ... ;
HFor = @ *n CFor'* | < *m* > *CFor'* | [*m*] *CFor'* | *Here n* | ! *n CFor'* | ? *n CFor'*;

[4] $\mathcal{H}$CASL consists of the hybridisation of the institution CASL with the models restricted to those with sorts commonly realised in all the states and with common realisation of the quantified variables.

where n is a nominal, m a modality, and the last two alternatives the universal and existential quantifiers over nominals. The grammar above besides extending sentences with the typical *hybrid* machinery also includes the quantification over nominals. The latter makes possible to define complex operators such as the $\downarrow$ binder, or the notion of a rigid designator

Notice also the need to use the keyword *Here*, when having a nominal for a sentence. Such is needed so that the parser for CASL does not take nominals as a common proposition. In the following section a more sophisticated mechanism to prevent this kind of ambiguities is discussed.

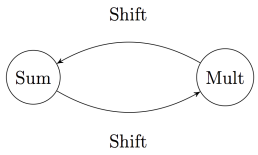

Fig. 1. Kripke frame for the swinging calculator.

Figure 1 depicts a toy example of a calculator which commutes, through a modality $shift$, between two modes of operation: a binary operation is interpreted as arithmetic addition in mode Sum and multiplication in $Mult$. Its $\mathcal{H}$CASL encoding is reproduced below.

logic HYBRID
spec RECONFCALC =
HNAT
then modalities *Shift*
nominals *Sum*,*Mult*
ops $\_\#\_ : Nat \times Nat \to Nat$

%% global axioms
$\forall\, n, m, p : Nat \bullet n \# m = m \# n \wedge (n \# m) \# p = n \# (m \# p)$

%% axioms specific to Sum and Mult
$\forall\, n, m : Nat \bullet @Sum\ n \# 0 = n \wedge @Sum\ n \# suc(m) = suc(n \# m) \wedge @Mult\ n \# 0 = 0$
$\exists\, p, q : Nat \bullet @Mult\ n \# suc(m) = p \wedge @Sum\ n \# q = p \wedge @Mult\ n \# m = q$

%% axioms specific to the Kripke frame
$\bullet\ Here\ Sum \vee Here\ Mult$
$\bullet\ @Sum\ (< Shift > Here\ Mult \wedge [Shift]\ Here\ Mult)$
$\bullet\ @Mult\ (< Shift > Here\ Sum \wedge [Shift]\ Here\ Sum)$

%% lt relation definition, using # op
$\forall\, n, m, r : Nat \bullet n <= m \Rightarrow n \# r <= m \# r$

The encoding from $\mathcal{H}$CASL to CASL provides the expected proof support for this sort of hybrid specifications, *i.e.* the set of proof tools available for CASL is brought to $\mathcal{H}$CASL. Below there are some examples of properties of the swinging calculator specification verified in this way.

Figure 2 registers the corresponding HETS session, showing the proof window, part of the model theory, and the specification graph.

- $@Sum\ n\ \#\ m >= n$ %(lemma2)%
- $@Mult\ (m = 0 \vee m = suc(0) \Rightarrow n\ \#\ m <= n)$ %(lemma6)%
- $@Sum\ [Shift]\ [Shift]\ Here\ Sum$ %(Cyclicity1)%

$\forall\ n, m, r : Nat$

- $n\ \#\ 0 = 0 \Rightarrow < Shift > n\ \#\ 0 = n$ %(StateExclusion)%
- $\exists\ p : Nat \bullet @Sum\ n\ \#\ n = p \Rightarrow @Mult\ n\ \#\ suc(suc(0)) = p$ %(DoubleDef)%
- $\exists\ p, q : Nat \bullet m <= suc(0) \Rightarrow @Sum\ n\ \#\ m = p \wedge @Mult\ n\ \#\ m = q \Rightarrow p >= q$ %(CasesSumBiggerMult)%

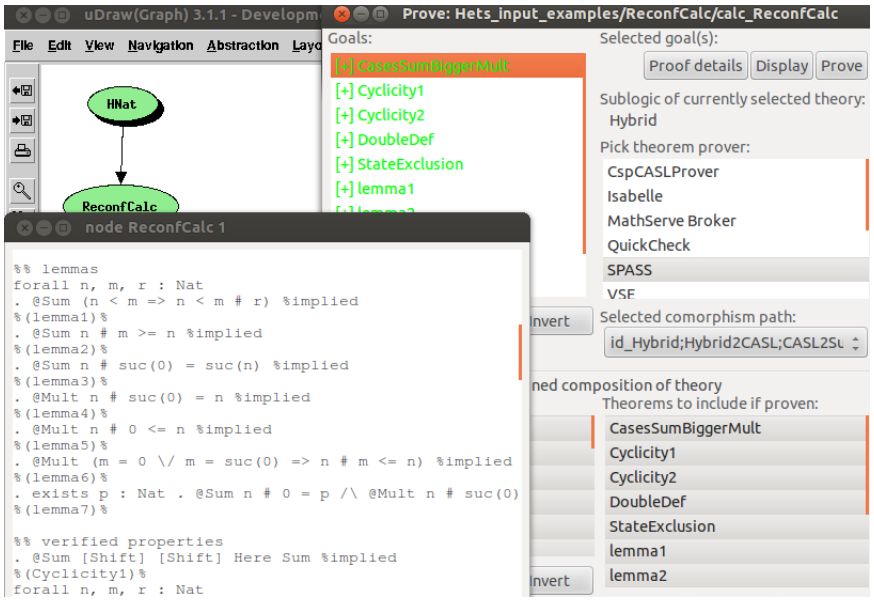

Fig. 2. A HETS session for the swinging calculator.

2.2 Generic hybridisation

The current integration of the hybridisation method into HETS offers the user hybridised versions of logics already "plugged in" HETS . Examples include propositional logic,

CASL, COCASL, and any other logic previously hibridised. The implementation of the hybridisation framework can also be used to expand this list: given an identifier, a sentences' parser and an analyser, a new logic can be taken into the picture.

In each case the resulting grammar for writing *hybridised* specifications is the composition of a specification in the base logic, the declaration of nominals and modalities, and the sentences enriched with hybrid properties.

Note, finally, that *hybridisation* as described here may introduce ambiguities. In the example below, for instance, it is not clear which nominals belong to the base or the hybridised layer.

To clear out possible ambiguities resulting from double or multiple applications of the hybridisation method, formulas associated to the base layer are wrapped into curly brackets (i.e. they are transported to the hybridised level through an injection).
Follows an example specified in HETS using the *double hybridisation* of propositional logic:

logic HYBRIDIZE
spec GEOGRAPHY =
  baselogic *Hybridize*
  Basic_Spec{ **baselogic** *Propositional*
    Basic_Spec { **props** *p* }
    Nominals *Portugal*, *England*, *Canada*
    Modalities *Car*
    • @ *Portugal* <*Car*> *England* }
  Nominals *Europe*, *America*
  Modalities *Plane*
  • *America* => { **not (** *Portugal* ∨ *England* **)** } ∧ *Europe* => { **not** *Canada* };
  • @ *Europe* <*Plane*> **(** *America* ∧ { *Canada* } **)**

The specification above exemplifies a double hybridisation. It describes routes in a map linked by some means of transport (the modalities) between different places (identified by nominals). One level of hybridisation corresponds to countries; the second one to continents. Clearly, in this case nominals can be ordered respecting the order used to build an hierarchy of countries and continents. Note how this hierarchy is brought back into sentences. For instance the last one in the above specification states : *from Europe one can travel by plane to America; and, in particular, to Canada*.

3 Discussion

The *hybridisation* process and its implementation on HETS proved an effective and flexible way to prove properties of *hybrid* specifications and thus to support the design method in [MFMB11]. The implementation compares well with respect to dedicated provers for (specific) hybrid logics, although a systematic comparison is still being done. Typically, such tools, such as HTAB[HA09] or HYLORES[AH02], are faster to prove a formula with low complexity, but HETS achieves similar or even better performance in more complex ones. In some cases, formulas hard to deal with in HTAB are straightforward in HETS. A typical example is $A(\downarrow x\langle m\rangle\neg x)$.

Moreover, the genericity of the approach reported in this paper seems highly attractive in practice.

The results of [MMDB11,DM13] have yet a great potential to be explored on top of their integration in the HETS integration. The first reference shows that a comorphism from an arbitrary institution $\mathcal{I}$ to $\mathcal{FOL}$ gives rise to another comorphism from its hybridisation $\mathcal{HI}$ to $\mathcal{FOL}$. Reference [DM13] refines this by characterising the conservativeness of such maps. Conservativeness is sometimes achieved not for the "free hybridisation" but for a restrict semantics of the hybridised institution satisfying a set of properties. Those restrictions are axiomatised on the $\mathcal{FOL}$ "side" as suitable presentations. The HETS rich support for $\mathcal{FOL}$ justifies the pertinence of the "hybridisation of comorphisms" method, since it extends tool support for a wide class of hybridised logics. Those includes not only hybrid equational logic and hybrid first-order logic (already supported by the comorphism presented above) but also hybridised modal logic and even hybridised hybrid propositional logic, among others.

Acknowledgements: This work is funded by ERDF - European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and by National Funds through the FCT, Portuguese Foundation for Science and Technology within project FCOMP-01-0124-FEDER-028923, and by *CIDMA-Centro de Investigação e Desenvolvimento em Matemática e Aplicações* of Universidade de Aveiro within the project FCOMP-01-0124-FEDER-022690.

References

[AH02] C. Areces and J. Heguiabehere. Hylores: A hybrid logic prover based on direct resolution. In *CADE-18*, volume 2392 of *LNCS*. Springer, 2002.

[Bla00] Patrick Blackburn. Representation, reasoning, and relational structures: A hybrid logic manifesto. *Logic Journal of IGPL*, 8(3):339–365, 2000.

[DM13] R. Diaconescu and A. Madeira. Encoding hybridized institutions into first order logic. Submitted to a journal, 2013.

[GB92] Joseph A. Goguen and Rod M. Burstall. Institutions: abstract model theory for specification and programming. *J. ACM*, 39:95–146, 1992.

[HA09] G. Hoffmann and C. Areces. Htab: a terminating tableaux system for hybrid logic. *Electr. Notes Theor. Comput. Sci.*, 231:3–19, 2009.

[MFMB11] A. Madeira, J. M. Faria, M. A. Martins, and L. S. Barbosa. Hybrid specification of reactive systems: An institutional approach. In G. Barthe, A. Pardo, and G. Schneider, editors, *SEFM 2011: Software Engineering and Formal Methods*, volume 7041 of *LNCS*, pages 269–285. Springer, 2011.

[MHST03] Till Mossakowski, Anne Haxthausen, Donald Sannella, and Andrzej Tarlecki. CASL: The common algebraic specification language: Semantics and proof theory. *Computing and Informatics*, 22:285–321, 2003.

[MMDB11] M. A. Martins, A. Madeira, R. Diaconescu, and L. S. Barbosa. Hybridization of institutions. In A. Corradini, B. Klin, and C. Cîrstea, editors, *CALCO 2011: Algebra and Coalgebra in Computer Science*, volume 6859 of *LNCS*. Springer, 2011.

[MML07] Till Mossakowski, Christian Maeder, and K. Lüttich. The heterogeneous tool set, hets. In *TACAS'07: Tools and Algorithms for Construction and Analysis of Systems*, volume 4424 of *LNCS*, pages 519–522. Springer, 2007.