Journal of Applied Logic 11 (2013) 272–288

Contents lists available at SciVerse ScienceDirect

Journal of Applied Logic

www.elsevier.com/locate/jal

The expressibility of fragments of Hybrid Graph Logic on finite digraphs

James Gate, Iain A. Stewart *

School of Engineering and Computing Sciences, Durham University, Science Labs, South Road, Durham DH1 3LE, UK

ARTICLE INFO

Article history:
Received 2 January 2013
Accepted 20 May 2013
Available online 24 May 2013

Keywords:
Hybrid graph logic
Modal logic
Finite model theory
Pebble games
Quantifier-rank

ABSTRACT

Hybrid Graph Logic is a logic designed for reasoning about graphs and is built from a basic modal logic, augmented with the use of nominals and a facility to verify the existence of paths in graphs. We study the finite model theory of Hybrid Graph Logic. In particular, we develop pebble games for Hybrid Graph Logic and use these games to exhibit strict infinite hierarchies involving fragments of Hybrid Graph Logic when the logic is used to define problems involving finite digraphs. These fragments are parameterized by the quantifier-rank of formulae along with the numbers of propositional symbols and nominals that are available. We ascertain exactly the relative definability of these parameterized fragments of the logic.

1. Introduction

Graphs of one form or another are ubiquitous in mathematics and computer science, and almost all logics can be used to reason about them. One particular application area of graph-based reasoning is in model checking where we are given a formal model of some system and a property of that system, and we wish to verify whether the given model has the given property. In model checking, the formal model (which may be infinite) is usually supplied as a labelled digraph in the form of a Kripke structure and the property is usually expressed by some formula of a modal or temporal logic (see, for example, [7]; throughout, by 'graph' we mean directed graph, which is a normal Kripke frame). Key to the use of logics for model checking is the decidability and complexity of the related model-checking, satisfiability and validity problems.

Hybrid logics go back to the work of Prior (see, for example, [1]) but have only relatively recently been studied in relation to computer science (see, for example, [1,10,15]). A hybrid logic is an extension of a modal or temporal logic in which symbols are used to name individual points in Kripke structures. Key to many hybrid logics is the use of nominals, n, and nominal operators, $@_n$, which allow us to 'jump' to the point of a Kripke structure named by the nominal n, and the 'binder' $\downarrow$, which allows us to bind variables to points. The study of hybrid logics in theoretical computer science has been in a number of contexts, such as in relation to description logics used in knowledge representation (for example, [5]), to proof theory (for example, [6]) and to model checking (for example, [10]) where the focus has been on the decidability and complexity of the related model-checking, satisfiability and validity problems.

Finite model theory is the study of the model theory of finite structures and, of course, has a strong relationship with fields such as model checking. One thriving aspect of finite model theory is the classification of logics according to their capacity to define problems (that is, isomorphism-closed classes of finite structures), particularly in relation to the computational complexity of the problems; this sub-area of finite model theory is sometimes referred to as descriptive complexity. Up until recently, hybrid logics have not been closely studied in this context; that is, as mechanisms for defining

* Corresponding author.
E-mail address: i.a.stewart@durham.ac.uk (I.A. Stewart).

1570-8683/$ – see front matter
http://dx.doi.org/10.1016/j.jal.2013.05.002

isomorphism-closed classes of (finite) frames (that is, digraphs). However, in [4] Benevides and Schechter defined (amongst other hybrid logics) *Hybrid Graph Logic HGL*, which is a very basic modal logic augmented with the use of nominals and a facility to verify the existence of paths in graphs through the (path-) quantifiers $\Diamond^+$ and $\Box^+$. The intention in [4] was to develop (modal and hybrid) logics for reasoning about graphs (that is, frames, as opposed to Kripke structures) that are expressive enough to define core graph-theoretic problems relating to properties such as connectedness, acyclicity and Hamiltonicity. It should be added that a transitive closure operator has also been added to hybrid logics in the form of Zhen and Seligman's 'community operator' [16] and recently by Lange too [14].

Actually, Hybrid Graph Logic *without* nominals is a well-known and well-studied fragment of both PDL and CTL (see, for example, [1,9]); furthermore, the logic HGL itself (that is, where nominals are allowed) has been independently formulated and studied (from the perspective of tableaux systems) in [13] where it is referred to as basic modal logic extended with nominals and eventualities, and denoted H^*. We continue to refer to the logic as HGL, given that the emphasis of our study follows the tone set in [4].

In this paper, we focus on Hybrid Graph Logic HGL as a logic for defining problems involving finite directed graphs (that is, finite frames). We develop an Ehrenfeucht–Fraïssé-style pebble game for HGL that allows us to study the expressibility of fragments $\mathrm{HGL}_r(c,d)$ of the logic HGL, parameterized by the quantifier-rank r of formulae, the number c of propositional symbols available and the number d of nominals available. We use our pebble game to show that for any $r, c, d \geqslant 0$, when we equate a logic with the class of (digraph) problems it defines, we have that

$$\mathrm{HGL}_r(c,d) \text{ is a proper subset of } \mathrm{HGL}_{r+1}(c,d);$$

in fact, in addition we show that there are problems definable by formulae of $\mathrm{HGL}_{r+1}(c,d)$ in which the path-quantifiers $\Diamond^+$ and $\Box^+$ are not used. Moreover, we also show that if $r \geqslant 1$, $c, d \geqslant 0$, $c' \geqslant c$, $d' \geqslant d$ and $c' + d' = c + d + 1$ then

$$\mathrm{HGL}_r(c,d) \text{ is a proper subset of } \mathrm{HGL}_r\big(c', d'\big)$$

(and we detail exactly the problems definable in the logics $\mathrm{HGL}_0(c,d)$). Consequently, we obtain a refined view of the structure of Hybrid Graph Logic and ascertain the relative expressive power of the logics formed by restricting the quantifier-rank, the number of propositional symbols and the number of nominals.

In the next section, we give the basic definitions and notation relevant to this paper before developing our pebble games in Section 3. In Section 4, we play our games and obtain our hierarchy results. Our conclusions are given in Section 5.

2. Basic definitions

In this section, we recapitulate the syntax and semantics of Hybrid Graph Logic (as it was defined in [4]). In essence, Hybrid Graph Logic is a hybrid logic augmented with a facility to validate paths in structures, and the structures in which the formulae of Hybrid Graph Logic are interpreted are Kripke structures with a single modality. We explain how we use the semantics of Hybrid Graph Logic so as to work with finite digraphs through the consideration of Kripke structures. Whilst our presentation is self-contained, we refer the reader to [1] and [11] for more information as regards hybrid logics and modal logics, respectively.

2.1. The syntax and semantics of HGL

First, the syntax of Hybrid Graph Logic. Every formula of Hybrid Graph Logic is parameterized by a set of *propositional symbols* P (coming from some set of available propositional symbols) and by a set of *nominals* N (coming from some set of available nominals).

Definition 1. The formulae: p, where p is a propositional symbol; n, where n is a nominal; and $\bot$ are well-formed formulae of Hybrid Graph Logic and are atomic formulae. If ψ and ψ' are well-formed formulae of Hybrid Graph Logic then so are

$$\psi \Rightarrow \psi'; \quad \Diamond\psi; \quad \Diamond^+\psi; \quad \text{and} \quad @_n\psi,$$

where n is a nominal. The language $\mathrm{HGL}(\mathsf{P},\mathsf{N})$ consists of those formulae, built recursively as stated here, for which every propositional symbol used comes from the finite set P and every nominal used comes from the finite set N.

Now for the structures in which we interpret formulae of $\mathrm{HGL}(\mathsf{P},\mathsf{N})$.

Definition 2. We write $\mathcal{G} = (V, E)$ when $\mathcal{G}$ is a digraph with (non-empty finite) vertex set V and edge set E (we allow the possibility of self-loops). When we think of $\mathcal{G}$ as a relational structure $\langle V, E\rangle$, with E a binary relation, we refer to $\mathcal{G}$ as a *frame*, with the vertices of the frame $\mathcal{G}$ referred to as *points*. A *pointed frame* $\langle \mathcal{G}, v\rangle$ is a frame $\mathcal{G} = \langle V, E\rangle$ together with a point $v \in V$. Given a set of propositional symbols P and a set of nominals N, a *Kripke* $\mathsf{P} \cup \mathsf{N}$*-structure* $\mathcal{C} = \langle \mathcal{G}, \mu\rangle$ is a frame $\mathcal{G} = \langle V, E\rangle$ together with a function $\mu : \mathsf{P} \cup \mathsf{N} \to \wp(V)$, called a *valuation function*, for which for every $n \in \mathsf{N}$, $\mu(n)$ is a singleton set (if $\mu(n) = \{v\}$ then we sometimes write $\mu(n) = v$ and say that the nominal sits on the point v). The points