Abstract

Hintikka’s World is a graphical and pedagogical tool that shows how artificial agents can reason about higher-order knowledge. In this demonstration paper, we present the implementation of symbolic models in Hintikka’s World. They enable the tool to scale, by helping it to face the state explosion, which makes it possible to provide examples featuring real card games, such as Hanabi.

1 Introduction

Constructing intelligent programs that play games with imperfect information is challenging — two notorious examples that have recently stirred some interest being Hanabi [Bard et al., 2019] and Starcraft 2 [Hu et al., 2018]. As far as we know, an important ingredient that misses from most approaches is the ability to reason about higher-order knowledge (an agent knows that another agent knows that...). In these systems, epistemic logic and its dynamic extension, dynamic epistemic logic (DEL) [Baltag et al., 1998], [van Ditmarsch et al., 2008] may offer formal tools for providing explanations in such AI programs.

The only pedagogical tool explaining these models that we are aware of is Hintikka’s world, which was presented at ECAI-IJCAI 2018 [Schwarzentruber, 2018]. Hintikka’s world is a proof of concept of a graphical user interface that represents Kripke models by comic strips, as shown in Figure 1. It enables the user to explore the mental states of agents. The tool is available at the following address: http://hintikkasworld.irisa.fr/ and the source code is available here: https://gitlab.inria.fr/fschwarz/hintikkasworld

Kripke models are graphs, represented explicitly in memory in the first version of the tool. Explicit models are useful to learn how dynamic epistemic logic works, by means of toy examples: “Muddy Children”, “Sally and Anne” [Wimmer and Perner, 1983], etc. However, the first version of Hintikka’s world does not scale. For instance, in real card games, such as Hanabi, there is an exponential number of possible configurations of cards. In its standard version, Hanabi has 50 cards total, each player’s hand contains 4 cards, and the order of the cards is important. Therefore, with 4 players, the initial Kripke model features $50 \times 49 \times 48 \times \cdots \times 35$ configurations, that is $1.03 \times 10^{39}$. Thus, it is impossible to explicitly represent the full graph in memory.

That is why, in this demonstration, we propose to represent Kripke models symbolically by using the approach of Charrier and Schwarzentruber [2017; 2018]. The implementation relies on Binary Decision Diagrams (BDDs) [Bryant, 1986]. There is another implementation of symbolic epistemic models, called DEMO [van Benthem et al., 2015], but their implementation is difficult to use in a web application and has no graphical user interface.

2 Demonstration Outline

In the demonstration we run through a game of (a variant of) Hanabi. In Hanabi, each agent has cards with a color and a number, but cannot see his/her own hand. At each turn, in Hintikka’s World, the user can play the role of one of the agents: he/she can either give the information to some other agent about a number or a color, or play a card. The goal is to play the cards in increasing order for each color. During the process, the system keeps track on the knowledge of the agents. More precisely, the system shows the real world (the real distribution of the cards). When the user clicks on an agent, the system displays a sampling of some possible worlds for that agent (i.e., some distributions of cards he/she still considers as possible at this stage of the game). The agents also reason about knowledge of other agents, as shown in Figure 2 (two levels of knowledge are shown).

Note that in this demonstration, in order to explain models of DEL, the tool still presents examples that rely on explicit models, such as “Sally and Anne”, “Muddy Children”, “Consecutive Numbers”, etc.
3 Symbolic Models

We emphasize the use of model checking over theorem proving, as advocated by Halpern and Vardi [1991]. We use the same ideas as in symbolic model checking, as defined for temporal logics [Burch et al., 1990], adapted to DEL, as explained in papers by a subset of the contributors of this demo [Charrier and Schwarzenbruter, 2017; Charrier and Schwarzenbruter, 2018]. Our model checking procedure relies now on symbolic Kripke models, aimed at representing succinctly Kripke models. A Kripke model is a graph whose nodes are possible worlds; edges are labeled by agents and an edge \( w \rightarrow^{a} u \) means that agent \( a \) considers world \( u \) as possible in world \( w \). Each world \( w \) is equipped with a valuation telling the true atomic propositions in \( w \). The tool shows that graph on the right of the screen (in Figure 1, the Kripke model has two possible worlds, \( w \) and \( u \); \( p \) is true in \( w \) but not in \( u \); \( \rightarrow_{a} \) is given in red and \( \rightarrow_{b} \) in blue).

A symbolic model gives a Boolean formula \( \chi(\vec{x}) \) that succinctly describes the set of possible worlds: a world is a valuation over Boolean variables \( \vec{x} \) satisfying \( \chi(\vec{x}) \). It also gives, for each agent \( a \), a Boolean formula \( \pi_{a}(\vec{x}, \vec{x}') \) that tells whether there is an edge labeled by agent \( a \) from a world described by a valuation over \( \vec{x} \) and a world described by a valuation over \( \vec{x}' \). All these Boolean formulas are then classically converted in BDDs. Typically, for Hanabi, \( \chi(\vec{x}) \) tells that \( \vec{x} \) describes an initial possible configuration. Formula \( \pi_{a}(\vec{x}, \vec{x}') \) tells that agents other than \( a \) have the same cards in \( \vec{x} \) and \( \vec{x}' \) (this models the fact that agent \( a \) only sees his/her own cards).

Dynamic epistemic logic also provides so-called event models for describing actions (public announcements, public actions, private announcements/actions, etc.). The reader may refer to the textbook on DEL [van Ditmarsch et al., 2008] and to Charrier and Schwarzenbruter [2017] for symbolic event models, that we do not detail here.

4 System Description

Whereas the first version was written in JavaScript, in order to ease the development, the new version is written in TypeScript and relies on the Angular 7 framework.

4.1 Binary Decision Diagrams

As shown by Charrier and Schwarzenbruter [2017], the symbolic model checking of DEL is PSPACE-complete, thus is critical. We manipulate sets of worlds as well as relations by means of Binary Decision Diagrams. To this aim, we wrote a JavaScript wrapper of the C library CUDD (Colorado University Decision Diagram Package) [Somenzi, 2001]: we wrote a thin wrapper in C, then compiled into Web Assembly using Emscripten, in order to be usable from our JavaScript module.

In order to show possible worlds for a given agent \( a \) in some world \( w \), we first construct the BDD of \( \pi_{a}(\text{descr}(w), \vec{x}') \) where \( \text{descr}(w) \) are the Boolean values of \( \vec{x} \) corresponding to world \( w \). We then count the number of possible valuations \( \vec{x}' \) that make \( \pi_{a}(\text{descr}(w), \vec{x}') \) true (BDDs are an efficient representation for counting valuations satisfying a Boolean formula). If the number of such valuations is small, we show all possible worlds, otherwise we randomly generate valuations for \( \vec{x}' \) that makes \( \pi_{a}(\text{descr}(w), \vec{x}') \) true (we randomly select a branch that leads to the “true” leaf in the BDD of \( \pi_{a}(\text{descr}(w), \vec{x}') \)).

4.2 Class Architecture

Figure 3 shows the new architecture of *Hintikka’s world*. EpistemicModel is an abstract class, used by the graphical user interface (GUI), that is independent from the concrete example (“Muddy Children”, “Sally and Anne”, “Hanabi”, etc.) but also, more interestingly, independent from the representation of the epistemic model itself. In particular, an epistemic model can be an ExplicitEpistemicModel (a graph) or a SymbolicEpistemicModel that relies on BDDs. To obtain a comic strips for a given example, it suffices to implement the method draw of a class that inherits from class World, that tells how a possible world is drawn.

4.3 Adding New Examples

Providing new examples is easy. Explicit epistemic models are directly described (sets of nodes and of edges). Symbolic epistemic models are described by a Boolean formula \( \chi \), and Boolean formulas for \( \pi_{a} \). The system provides a way to easily describe how worlds are displayed in the comic strips.

5 Future Work

We plan to extend our implementation to Algebraic decision diagrams (ADD) [Bahar et al., 1997] in order to add examples in robotics that need numerical values, not only Boolean values. We also aim to study counting and sampling specific techniques (see for instance Meel et al. [2016]) for generating possible worlds, in order to tackle even bigger examples.

Acknowledgments

We thank Hai Trung Pham for his work during his internship and Arthur Queffelec for his valuable comments on the architecture.
References


