Towards relating hyper- and epistemic-temporal logics

Matvey Soloviev

Remote talk @ PLAS 2024, 2024-10-14

Joint work with **Vineet Rajani** and **Musard Balliu**

October 13, 2024

Overview

- KCTL* and HyperCTL* represent two major classes of logics for security properties. Which one is more expressive?
- Bozzelli et al. (2014): incomparable, but HyperCTL* plus a linear past operator subsumes KCTL*.
- Rabe (2016): HyperLTL plus quantification over props also subsumes KLTL.
- **This work:** KCTL* plus quantification over props subsumes HyperCTL*.

Background: Formal specification of security properties

We want to formally specify and verify security properties, which usually compare behaviour of programs to counterfactual, alternative runs.

Common specification logics such as LTL and CTL* are inadequate, as they describe single executions.

Background: Hyperproperties and hyperlogics

Hyperproperties (Clarkson-Schneider 2008) were introduced as a formalism to describe and stratify security properties in terms of how many runs need to be considered simultaneously to describe them.

As companion logics to this framework, Clarkson et al. (2014) introduce HyperLTL and HyperCTL*, which extend the respective temporal logics by adding quantification over runs.

Background: Epistemic-temporal logics

Another lineage of logics derives from formal models of knowledge (Hintikka 1962, Halpern 1986...). Security properties are represented in terms of the knowledge of agents involved in the system.

Such logics don't constrain the number of alternative runs to be considered, and don't let you refer to individual counterfactual runs. Instead, they package quantifications such as "in no run that A considers possible, ...".

When added into LTL and CTL*, we get the logics KLTL and KCTL*.

Comparing expressivity

The two families of logics have strengths and weaknesses in terms of tractability and intuition. Common security properties, however, can be expressed in either (S.-B.-Guanciale 2023). This raises the question: Which one is more expressive?

- Bozzelli et al. (2014) show that KCTL* and HyperCTL* are incomparable – examples in each that are inexpressible in the other.
- Also in Bozzelli et al. (2014): adding past modalities X^- and U^- to HyperCTL* produces a logic that is stronger than either.
- Rabe (2016): adding quantification over propositions $\exists a.\, \varphi(a)$, which binds an arbitrary proposition to the variable a, to HyperLTL(!) also produces a logic QPTL that subsumes both HyperLTL and KLTL.

Our work

These results paint a picture that suggests that hyperlogics may be more powerful – you just need to add a little expressivity to them to subsume epistemic-temporal ones. But is this actually true?

Our work

These results paint a picture that suggests that hyperlogics may be more powerful – you just need to add a little expressivity to them to subsume epistemic-temporal ones. But is this actually true?

We show that by adding quantification over propositions to KCTL*, we can likewise obtain a logic KPCTL* that subsumes HyperCTL*.

Details

We construct a polynomial-time reduction that

- given a HyperCTL* model H, constructs a KPCTL* model $\Lambda(H)$,
- and, given any HyperCTL* formula φ, constructs a KPCTL* formula $[\![\varphi]\!]$

such that $\models_{\Lambda(H)} [\![\varphi]\!]$ iff $\models_H \varphi$.

Excerpt of the logics

HyperCTL*:

$$\begin{array}{lcl}
\varphi & ::= & \top \mid p[x] \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{X}\varphi \mid \varphi \mathsf{U} \varphi \mid \exists x.\varphi \\
\Pi, y, i \models_K p[x] & \Leftrightarrow & p \in V(\Pi(x)(i)) \\
\Pi, y, i \models_K \exists x.\varphi & \Leftrightarrow & \Pi[x \mapsto \pi'], x, i \models_K \varphi \\
 & & \quad \text{for some initial path } \pi' \text{ of } K \text{ s.t. } \pi'[0,i] = \Pi(y)[0,i]
\end{array}$$

KCTL*:

$$\begin{array}{lcl}
\varphi & ::= & \top \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{X}\varphi \mid \varphi \mathsf{U} \varphi \mid \exists \varphi \mid \mathsf{K}_a\varphi \\
\pi, i \models_\Lambda p & \Leftrightarrow & p \in V(\pi(i)) \\
\pi, i \models_\Lambda \mathsf{K}_a\varphi & \Leftrightarrow & \text{for all initial paths } \pi' \text{ of } K \text{ s.t.} \\
 & & V(\pi[0,i]) \text{ and } V(\pi'[0,i]) \text{ are Obs}_a\text{-equivalent, } \pi', i \models_\Lambda \varphi.
\end{array}$$

Augmenting KCTL*

To obtain KPCTL*, we add proposition bindings to the context of KCTL*, and use them to interpret quantification over propositions:

$$\Phi, \pi, i \models_\Lambda P \quad \Leftrightarrow \quad \pi, i \models_\Lambda \Phi(P)$$

$$\Phi, \pi, i \models_\Lambda \exists P.\, \varphi \quad \Leftrightarrow \quad \exists \psi \in \text{KCTL*}\colon\ \Phi[P \mapsto \psi], \pi, i \models_\Lambda \varphi$$

Characteristic formulae

Our proof depends on the existence of characteristic formulae for runs: formulae which are true everywhere in a particular run and nowhere else (or in some run suffix, for branching-time logics).

The same assumption is baked into QPTL's quantification over propositions, which is implemented as a quantification over truth tables. We just need to make it explicit as we use an intensional variant, where quantification is over KCTL* formulae.

Knowledge modalities

The model $\Lambda(H)$ is just H with two special knowledge modalities K^+ and K^-.

K^+ represents "knowing everything", and relates a particular point in a run only to corresponding points where the history is the same.

K^- represents "knowing nothing", and relates a particular point to corresponding points in all runs (we assume the setting is synchronous). This lets us access all runs.

Detecting characteristic formulae

Given a KPCTL* formula ψ, we can encode the property that $\mathsf{FG}\psi$ is the characteristic formula of the current run as follows:

$$\mathrm{CHAR}(\psi) \triangleq \mathsf{FG}\psi \land \forall\varphi.\, (\mathsf{FG}\varphi) \Rightarrow \mathsf{K}^{-}((\mathsf{FG}\psi) \Rightarrow \mathsf{FG}\varphi)$$

Here, F and G are standard derived “eventually” and “forever” modalities; $\mathsf{FG}\varphi$ thus means that φ holds over some suffix of the current run.

This allows us to quantify over just the characteristic formulae. We will use this as a gadget to emulate quantification over paths.

Converting formulae

To define the reduction of formulae $[\![\cdot]\!]$, we convert existential HyperCTL* quantification over branches x into existential quantification over propositions P_x which are characteristic formulae of some path:

$$[\![\exists x.\varphi]\!] \triangleq \exists P_x. \neg \mathsf{K}^+ \neg(\mathrm{CHAR}(P_x) \wedge [\![\varphi]\!]).$$

In HyperCTL*, atomic propositions p have to be evaluated with respect to a path variable x. We convert this by setting

$$[\![p[x]]\!] \triangleq \mathsf{K}^-(\mathsf{FG} P_x \Rightarrow p),$$

using K^- to pick out, from among all runs, the one where the characteristic formula P_x holds, and evaluate p there.