Asynchronous Hyperlogics

Ali Nosherwan Hamed

ANU

2026

Outline

Temporal logic – what and why

A modal logic to reason about systems over time.

We can use it to model computational processes.

- ~~possible worlds~~ states, ~~accessibility relation~~ state transitions.
- A trace (sequence of states) models a system execution
- A set of infinite traces models a system

What temporal logics are studied in the literature?

LTL

Linear-time Temporal Logic (LTL)

$$\varphi ::= a \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \mathsf{X}\varphi \mid \varphi_1 \,\mathsf{U}\, \varphi_2$$

We define $\Diamond\varphi = \mathit{true}\mathsf{U}\varphi$ and $\Box\varphi = \neg\Diamond\neg\varphi$

E.g. $\Box(a \rightarrow \Diamond b)$ expresses "event a is always followed by event b"

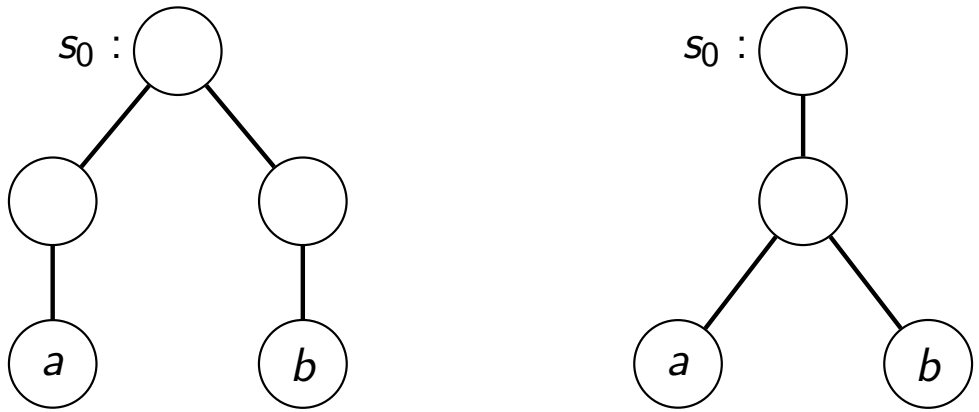
- LTL expresses properties of individual traces
- So, LTL is not expressive enough to capture branching behaviour

Kripke structures

We use Kripke structures to model systems with branching paths.

$$\mathcal{K} = \langle S, s_{init}, \delta, L \rangle$$

Two Kripke structures can be trace equivalent but differ in their branching structure.

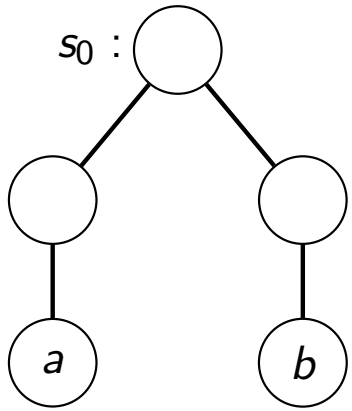


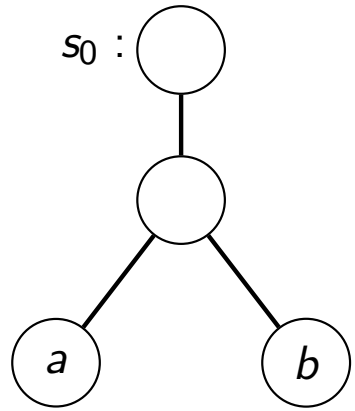
CTL*

Computation Tree Logic (CTL*) adds existential (E) and universal quantification (A) over paths.[1]

CTL* can describe branching-time properties, e.g. "there is a path where event a occurs"

AX(EX$a \wedge$ EXb) distinguishes the following:





[1] with some restrictions.

HyperCTL*

CTL* quantifies over the paths in a system but can only refer to a single path at a time.

To reason about relationships between multiple paths, we extend CTL* with path variables.

This gives us HyperCTL*

$$\varphi ::= a_\pi \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \mathsf{X}\varphi \mid \varphi_1 \,\mathsf{U}\, \varphi_2 \mid \exists \pi.\varphi$$

We require that temporal operators only occur inside the scope of path quantifiers.

Hyperproperties

HyperCTL*can express hyperproperties since it can compare multiple paths.

A hyperproperty is a set of sets of traces.

e.g. observational determinism: $\forall\pi.\forall\pi'.\Box(I_\pi = I_{\pi'}) \rightarrow \Box(O_\pi = O_{\pi'})$
where

- I = public inputs
- O = public observations

If two executions receive the same inputs, they must produce the same outputs.

Why asynchronous?

We want to reason about execution traces according to the relative order of the sequences of actions in each trace.

Also, HyperCTL*is not stuttering invariant. Adding or removing repeated states can change whether or not a property holds.

And, removing X is not enough, e.g., $\forall \pi. \forall \pi'. (\Box p_\pi = p_{\pi'})$

Solution: We use the notion of a *trajectory* that controls the relative speed at which traces progress.

We choose at each instant which traces move and which traces stutter.

Preliminaries

A *pointed path* is a pair (σ, p), where $p \in \mathbb{N}_0$ is a natural number (called the *pointer*).

Pointed paths allow to traverse a path by moving the pointer.

Given a pointed path (σ, p) and $n > 0$, we use $(\sigma, p) + n$ to denote the resulting path $(\sigma, p + n)$.

Given a Kripke structure $\mathcal{K}$, we denote the set of all pointed paths by $\mathsf{PP}_{\mathcal{K}} = \{(\sigma, p) \mid \sigma \in \mathsf{Paths}(\mathcal{K}) \text{ and } p \in \mathbb{N}_0\}$.

Asynchronous HyperCTL*

$$\varphi ::= a_{\pi,\tau} \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \mathsf{X}\varphi \mid \varphi_1 \,\mathsf{U}\, \varphi_2 \mid \exists\pi.\varphi \mid \forall\pi.\varphi \mid \mathsf{E}\tau.\varphi \mid \mathsf{A}\tau.\varphi$$

$\mathrm{Pvar}(\varphi)$ is the set of path variables quantified in φ.
$\mathrm{Tvar}(\varphi)$ is the set of trajectory variables quantified in φ.

A formula φ is *well-formed* if for all atoms $a_{\pi,\tau}$ in φ, π and τ are quantified in φ.

Trajectories

A *trajectory* $t = t_0 t_1 t_2 \cdots$ for a formula φ is an infinite sequence of subsets of $\text{Pvar}(\varphi)$.

In each step of the trajectory one or more of the paths make progress or all may stutter.

E.g. $t = \{\pi_1, \pi_2\}\{\pi_1\}\{\pi_1\}\{\pi_2\}\{\pi_1, \pi_2\}...$

A trajectory is *fair* for a path variable $\pi \in \text{Pvar}(\varphi)$ if there are infinitely many j such that $\pi \in t_j$.

A trajectory is fair if it is fair for all path variables in $\text{Pvar}(\varphi)$.

Asynchronous path assignments

For the semantics of A-HyperCTL*, we need:

- a *trajectory assignment* Γ
- an asynchronous path assignment Π
  - maps each pair (π, τ) to a pointed path.

We use $(\Pi, \Gamma) + 1$ for the successor of (Π, Γ), defined as (Π', Γ'), where $\Gamma'(\tau) = \Gamma(\tau)^1$, and:

- if $\pi \in \Gamma(\tau)(0)$: $\Pi'(\pi, \tau) = \Pi(\pi, \tau) + 1$
- if $\pi \notin \Gamma(\tau)(0)$: $\Pi'(\pi, \tau) = \Pi(\pi, \tau)$

Note: Π can assign the same π to different pointed paths depending on the trajectory.

Semantics of A-HyperCTL*

The satisfaction of an A-HyperCTL* formula φ over a path assignment Π, a trajectory assignment Γ, and a Kripke structure $\mathcal{K}$ is defined as follows:

$$
\begin{array}{llll}
(\Pi, \Gamma) & \models_{\mathcal{K}} & a_{\pi,\tau} & \text{iff} \quad a \in L(\sigma(p)) \text{ where } (\sigma, p) = \Pi(\pi, \tau) \\
(\Pi, \Gamma) & \models_{\mathcal{K}} & \mathsf{X}\varphi & \text{iff} \quad (\Pi, \Gamma) + 1 \models_{\mathcal{K}} \varphi \\
(\Pi, \Gamma) & \models_{\mathcal{K}} & \exists \pi. \varphi & \text{iff} \quad \text{for some } \sigma \in \mathsf{Paths}(\mathcal{K}) : \\
 & & & \quad (\Pi[(\pi, \tau) \mapsto (\sigma, 0)], \Gamma) \models_{\mathcal{K}} \varphi \text{ for all } \tau \\
(\Pi, \Gamma) & \models_{\mathcal{K}} & \mathsf{E}\tau. \varphi & \text{iff} \quad \text{for some } t \in \mathsf{TRJ}_{\mathsf{Pvar}(\varphi)} : \\
 & & & \quad (\Pi, \Gamma[\tau \mapsto t]) \models_{\mathcal{K}} \varphi
\end{array}
$$

Example

$$\varphi_{\mathsf{NI}} = \forall\pi.\exists\pi'.\mathsf{E}\tau.(\mathsf{h}_{\pi,\tau} \neq \mathsf{h}_{\pi',\tau}) \wedge \Box(\mathsf{obs}_{\pi,\tau} = \mathsf{obs}_{\pi',\tau}),$$

where obs denotes the output observations.

For all paths π, there should exist another path π' and a trajectory τ, such that:

π and π' start from different values of $\mathtt{h}$ and τ can align all the observations along π and π'

Future work

- Translate HyperCTL* models that stutter into "condensed" A-HyperCTL* models, with stuttering happening in the trajectory.
  - Is such a condensed model interesting? What does it say about the family of stuttering models?
- Bisimulation and any weaker equivalences.
- Define a bounded semantics (bound on trajectory length)
  - Relation with Bounded Next-preserving Branching Bisimulation.

Other avenues for research

Model checking for HyperLTL is decidable, but for A-HyperLTL is undecidable. BUT there are decidable fragments.

Similarly, model checking for HyperCTL* is decidable. Any interesting fragments of A-HyperCTL* that are decidable?

Key References

J. Baumeister, N. Coenen, B. Bonakdarpour, B. Finkbeiner, and C. Sánchez.
A temporal logic for asynchronous hyperproperties.
In A. Silva and K. R. M. Leino, editors, *Computer Aided Verification*, pages 694–717, Cham, 2021. Springer International Publishing.

L. Bozzelli, A. Peron, and C. Sánchez.
Expressiveness and decidability of temporal logics for asynchronous hyperproperties.
In *33rd International Conference on Concurrency Theory*, 2022.

M. R. Clarkson, B. Finkbeiner, M. Koleini, K. K. Micinski, M. N. Rabe, and C. Sánchez.
Temporal logics for hyperproperties.
In M. Abadi and S. Kremer, editors, *Principles of Security and Trust*, pages 265–284, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

T.-H. Hsu, B. Bonakdarpour, B. Finkbeiner, and C. Sánchez.
Bounded model checking for asynchronous hyperproperties.
In S. Sankaranarayanan and N. Sharygina, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, pages 29–46, Cham, 2023. Springer Nature Switzerland.