Temporal Team Semantics Revisited

Our LICS 2022 paper

Jens Oliver Gutsfeld[1] Arne Meier[2] Christoph Ohrem[1] Jonni Virtema[3]

[1] Universität Münster, Germany
[2] Leibniz Universität Hannover, Germany
[3] University of Sheffield, United Kingdom

21.09.2022 — IFIP Meeting

Core of Team Semantics

- In most studied logics formulae are evaluated in a single state of affairs. E.g.,
  - a first-order assignment in first-order logic,
  - a propositional assignment in propositional logic,
  - a possible world of a Kripke structure in modal logic.

Core of Team Semantics

- In most studied logics formulae are evaluated in a single state of affairs.
  E.g.,
  - a first-order assignment in first-order logic,
  - a propositional assignment in propositional logic,
  - a possible world of a Kripke structure in modal logic.
- In team semantics sets of states of affairs are considered.
  E.g.,
  - a set of first-order assignments in first-order logic,
  - a set of propositional assignments in propositional logic,
  - a set of possible worlds of a Kripke structure in modal logic.
- These sets of things are called teams.

Team semantics for temporal logics

- A trace over AP is an infinite sequence from $(2^{\mathrm{AP}})^{\omega}$.
- Trace can be seen to model an execution of a system over time.
- Important logics for trace properties are, e.g., LTL, CTL, μ-calculus.
  - The system will terminate eventually.
  - Every request is eventually granted.
  - The system will terminate in bounded time.

Team semantics for temporal logics

- A trace over AP is an infinite sequence from $(2^{\mathrm{AP}})^\omega$.
- Trace can be seen to model an execution of a system over time.
- Important logics for trace properties are, e.g., LTL, CTL, μ-calculus.
  - The system will terminate eventually.
  - Every request is eventually granted.
  - The system will terminate in bounded time.
- A trace property is *a property of traces* (the set of satisfying traces) vs. a hyperproperty is *a property of sets of traces* (analogous to a set of teams).
- Logics for hyperproperties: HyperLTL, HyperCTL, TeamLTL, etc.
  - Termination in bounded time is in TeamLTL, but not in HyperLTL.

LTL, HyperLTL, and TeamLTL

- In LTL the satisfying object is a trace.

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \vee \varphi) \mid X\varphi \mid \varphi U \varphi$$

- In HyperLTL the satisfying object is a set of traces and a trace assignment.

$$\varphi ::= \exists\pi\varphi \mid \forall\pi\varphi \mid \psi$$
$$\psi ::= p_\pi \mid \neg\psi \mid (\psi \vee \psi) \mid X\psi \mid \psi U \psi$$

- In TeamLTL the satisfying object is a set of traces. We use team semantics.

$$\varphi ::= p \mid \neg p \mid (\varphi \vee \varphi) \mid (\varphi \wedge \varphi) \mid X\varphi \mid \varphi U \mid \varphi W \varphi$$

$+$ atomic statements of dependence (dependence and inclusion atoms etc.)
$+$ additional connectives (Boolean disjunction, contradictory negation, etc.)

Team Semantics Atoms

- An atomic formula $\mathit{dep}(\varphi_1, \ldots, \varphi_n, \psi)$ expresses that the value of the formulae φ_i functionally determines the value of ψ
- An atomic formula $\varphi_1, \ldots, \varphi_n \subseteq \psi_1 \ldots \psi_n$ states that every truth value combination of the formulae φ_i must also occur as a truth value combination of the formulae ψ_i

Examples: HyperLTL vs. synchronous TeamLTL

- There is a timepoint (common for all traces) after which a does not occur.
  Not expressible in HyperLTL, but in HyperQPTL (with quantification over atomic propositions).
  $$\exists p \forall \pi \, \mathsf{F} p \wedge \mathsf{G}(p \rightarrow \mathsf{G} \neg a_\pi)$$
  Expressible in synchronous TeamLTL: $\mathsf{F}\mathsf{G}\,\neg a$

Examples: HyperLTL vs. synchronous TeamLTL

- There is a timepoint (common for all traces) after which a does not occur.
  Not expressible in HyperLTL, but in HyperQPTL (with quantification over atomic propositions).
  $$\exists p\, \forall \pi\, \mathsf{F} p \land \mathsf{G}(p \to \mathsf{G} \neg a_\pi)$$
  Expressible in synchronous TeamLTL: $\mathsf{FG}\, \neg a$
- Depending on an unknown input, execution traces either agree on a or on b.
  Expressible in HyperLTL with three trace quantifiers:
  $$\exists \pi_1\, \exists \pi_2\, \forall \pi\, \mathsf{G}(a_{\pi_1} \leftrightarrow a_\pi) \lor \mathsf{G}(b_{\pi_2} \leftrightarrow b_\pi).$$
  Expressible in synchronous TeamLTL: $\mathsf{G}(a \varovee \neg a) \lor \mathsf{G}(b \varovee \neg b)$.

Kripke structures and traces

A rooted Kripke structure is 4-tuple (W, R, V, r), where

- W is a (finite) set of states of the structure.
- the element $r \in W$ is the root of the structure.
- R is a right-total binary relation on W (i.e, $\forall x \in W \, \exists y \in W$ s.t. xRy).
- $V\colon W \to 2^{\mathsf{AP}}$ is an evaluation function.

A trace t over K is an infinite sequence s.t $t[0] = r$ and $t[i]Rt[i+1]$, for $i \in \mathbb{N}$.
($t[i]$ is the ith element of the sequence t.)

Time evaluation functions

Definition

Given a (possibly infinite) set of traces T over some common Kripke structure, a time evaluation function (tef for short) for T is a function

$$\tau\colon \mathbb{N} \times T \to \mathbb{N}$$

that given a trace $t \in T$ and a value of a the global clock $i \in \mathbb{N}$ outputs the value $\tau(i, t)$ of the local clock of trace t at global time i.

If τ is a tef and $k \in \mathbb{N}$ a natural number, then $\tau[k, \infty]$ is the k-shifted tef defined by putting $\tau[k, \infty](i, t) := \tau(i + k, t)$, for everty $t \in T$ and $i \in \mathbb{N}$.

Temporal teams

Definition

A temporal team is a tuple (T, τ), where T is a set of traces over some common Kripke structure and τ is a time evaluation function for T.

Temporal Semantics of TeamLTL

Definition

Let (T,τ) be a temporal team over a Kripke structure (W,R,V,r).

$$
\begin{array}{llll}
(T,\tau) \models p & \text{iff} & \forall t \in T : p \in t|[\tau(0,t)]) & (T,\tau) \models \neg p \quad \text{iff} \quad \forall t \in T : p \notin t|[\tau(0,t)] \\
(T,\tau) \models \phi \wedge \psi & \text{iff} & (T,\tau) \models \phi \text{ and } (T,\tau) \models \psi & (T,i) \models \mathsf{X}\varphi \quad \text{iff} \quad (T,\tau[1,\infty]) \models \varphi \\
(T,\tau) \models \phi \vee \psi & \text{iff} & \multicolumn{2}{l}{(T_1,\tau) \models \phi \text{ and } (T_2,\tau) \models \psi, \text{ for some } T_1, T_2 \text{ s.t. } T_1 \cup T_2 = T} \\
(T,\tau) \models \phi \mathsf{U} \psi & \text{iff} & \multicolumn{2}{l}{\exists k \in \mathbb{N} \text{ s.t. } (T,\tau[k,\infty]) \models \psi \text{ and } \forall m : 0 \leq m < k \Rightarrow (T,\tau[m,\infty]) \models \phi} \\
(T,\tau) \models \phi \mathsf{W} \psi & \text{iff} & \multicolumn{2}{l}{\forall k \in \mathbb{N} : (T,\tau[k,\infty]) \models \phi \text{ or } \exists m \text{ s.t. } m \leq k \text{ and } (T,\tau[m,\infty]) \models \psi}
\end{array}
$$

Note: If τ is the synchronous time evaluation function (i.e., $\forall t \forall i : \tau(t,i) = i$), then the above is exactly the semantics for synchronous TeamLTL as defined in [KMVZ18].

Properties of tefs

* marks optional properties

Strict Monotonicity: $\forall i : \tau(i) < \tau(i+1)$ (wrt. canonical order of tuples)

Stepwise: $\forall i \, \forall t : \tau(i+1, t) \in \{\tau(i,t), \tau(i,t)+1\}$.
Whenever a local clock ticks it ticks exactly one step.
Important to differentiate neXt operator from Future.

*Fairness: $\forall i \, \forall t \exists j : \tau(j,t) \geq i$.

*Non-Parallelism: $\forall i : i = \sum_t \tau(i,t)$

*Synchronousity: $\tau(i,t) = \tau(i,t')$ for all i, t, t'.

Quantification of tefs

Definition

Fix a set AP of atomic propositions. The set of formulae of TeamLTL (over AP) is generated by the following grammar:

$$\varphi ::= p \mid \neg p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathsf{X}\varphi \mid \varphi \mathsf{U} \varphi \mid \varphi \mathsf{W} \varphi$$

where $p \in \mathrm{AP}$.
The logical constants $\top, \bot$ and connectives $\to, \leftrightarrow$ are defined as usual (e.g., $\bot := p \wedge \neg p$), and $\mathsf{F}\phi := \top \mathsf{U} \phi$ and $\mathsf{G}\phi := \phi \mathsf{W} \bot$.

Quantification of tefs

Definition

Fix a set AP of atomic propositions. The set of formulae of TeamCTL* (over AP) is generated by the following grammar:

$$\varphi ::= p \mid \neg p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathsf{X}\varphi \mid \varphi \mathsf{U} \varphi \mid \varphi \mathsf{W} \varphi \mid \exists \phi \mid \forall \phi$$

where $p \in \mathrm{AP}$ and $\exists, \forall$ are tef quantifiers.

The logical constants $\top, \bot$ and connectives $\rightarrow, \leftrightarrow$ are defined as usual (e.g., $\bot := p \wedge \neg p$), and $\mathsf{F}\phi := \top \mathsf{U} \phi$ and $\mathsf{G}\phi := \phi \mathsf{W} \bot$.

Quantification of tefs

Definition

Fix a set AP of atomic propositions. The set of formulae of TeamCTL (over AP) is generated by the following grammar:

$$\varphi ::= p \mid \neg p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists \mathsf{X}\varphi \mid \exists \varphi \mathsf{U} \varphi \mid \exists \varphi \mathsf{W} \varphi \mid \forall \mathsf{X}\varphi \mid \forall \varphi \mathsf{U} \varphi \mid \forall \varphi \mathsf{W} \varphi$$

where $p \in \mathrm{AP}$.
The logical constants $\top, \bot$ and connectives $\rightarrow, \leftrightarrow$ are defined as usual (e.g., $\bot := p \wedge \neg p$), and $\mathsf{F}\phi := \top \mathsf{U} \phi$ and $\mathsf{G}\phi := \phi \mathsf{W} \bot$.

TeamCTL($\varovee$) is highly undecidable

Theorem

Model checking for TeamCTL($\varovee$) *is* Σ^1_1*-hard.*

Proof Idea: reduce existence of *b*-recurring computation of given 2-counter machine *M* and instruction label *b* to model checking problem of TeamCTL($\varovee$).

Deciding the Logic Using Alternating Asynchronous Büchi Automata (AABA)

- AABA are like standard ABA, but operate over multiple input words which can be read asynchronously
- In detail: AABA read tuples ω-words over an alphabet Σ, each single step advances only one of these words and the automaton can use disjunctive ($\vee$) and conjunctive ($\wedge$) alternation, with a Büchi acceptance condition
- Restricted sets of tefs are used to consider restricted sets of runs of AABA since all problems of interest are highly undecidable

Deciding the Logic Using Alternating Asynchronous Büchi Automata (AABA)

- The emptiness problem of AABA is decidable for some sets of tefs, e.g k-synchronous and k-context-bounded tefs
- For k-synchronous tefs, the problem is EXPSPACE-complete
- For k-context-bounded tefs, it is $(k-2)$-EXPSPACE-complete
- Path checking and fixed size satisfiability of our logic can be reduced to the emptiness problem of AABA

Deciding the Logic Using Alternating Asynchronous Büchi Automata (AABA)

- Path checking is to decide whether a formula φ holds for a finite multiset T of ultimately periodic traces
- The finite satisfiability problem is to decide whether there is a multiset T of size n such that φ holds for an input formula φ and natural number n
- The translation of formulae to AABA is based on the classical Fischer-Ladner construction for LTL
- Asynchronicity is handled using alternation

Summary

- General framework for temporal team semantics
- We can combine asynchronous and synchronous tefs
- We can embed synchronous TeamLTL
- Highly undecidable model-checking problem
- For certain sets of tefs, the path checking and fixed satisfiability problems become decidable by reduction to AABA

Summary

- General framework for temporal team semantics
- We can combine asynchronous and synchronous tefs
- We can embed synchronous TeamLTL
- Highly undecidable model-checking problem
- For certain sets of tefs, the path checking and fixed satisfiability problems become decidable by reduction to AABA

Current and future directions

- Identification of decidable fragments and variants
- Consider tefs also as inputs given in some finite way
- Lift decision algorithms for our logics to new models (Pushdown, VASS, etc.)

Summary

- General framework for temporal team semantics
- We can combine asynchronous and synchronous tefs
- We can embed synchronous TeamLTL
- Highly undecidable model-checking problem
- For certain sets of tefs, the path checking and fixed satisfiability problems become decidable by reduction to AABA

Current and future directions

- Identification of decidable fragments and variants
- Consider tefs also as inputs given in some finite way
- Lift decision algorithms for our logics to new models (Pushdown, VASS, etc.)

Thank you!