Interval temporal logic for visibly pushdown systems

Laura Bozzelli[1] **Angelo Montanari**[2] Adriano Peron[1]

[1]University of Napoli Federico II, Italy
[2]University of Udine, Italy

FSTTCS 2019
Mumbai, India
December 12, 2019

Context: Model Checking

Model checking is a well-established formal method to automatically check the global correctness of reactive systems against behavioral properties. Properties usually specified in point-based Propositional Temporal logics (PTLs).

Two types of PTLs differing in the nature of time: linear-time temporal logics, such as LTL, and branching-time temporal logics, such as CTL and CTL*.

In the last years, interval temporal logic has been proposed as an alternative formalism to model check relevant properties of computation stretches of finite-state systems (finite Kripke structures).

Context: Model Checking

Model checking is a well-established formal method to automatically check the global correctness of reactive systems against behavioral properties. Properties usually specified in point-based Propositional Temporal logics (PTLs).

Two types of PTLs differing in the nature of time: linear-time temporal logics, such as LTL, and branching-time temporal logics, such as CTL and CTL*.

In the last years, interval temporal logic has been proposed as an alternative formalism to model check relevant properties of computation stretches of finite-state systems (finite Kripke structures).

Goal

To check pushdown systems: infinite-state systems modeling the control flow of sequential programs with nested and recursive procedure calls.

To express branching-time, context-free requirements on the behavior of these systems in an interval-based temporal logic framework.

Model checking of Pushdown Automata (PDA)

Checking context-free properties of Pushdown Automata (PDA) is, in general, undecidable (the language inclusion problem for PDA is undecidable).

Model checking of Pushdown Automata (PDA)

Checking context-free properties of Pushdown Automata (PDA) is, in general, undecidable (the language inclusion problem for PDA is undecidable).

Algorithmic solutions for subclasses of PDA: Visibly Pushdown Automata (VPA)

R. Alur et al. - Visibly Pushdown Languages - STOC'04

Robust subclass of PDA: the input symbols over a pushdown alphabet control the admissible stack operations (visibility).

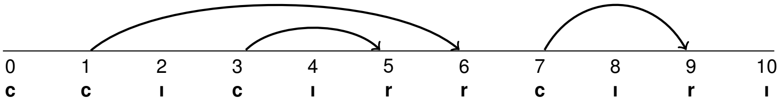
The pushdown alphabet Σ is partitioned into

- a set Σ_{call} of calls forcing push stack-operations,
- a set Σ_{ret} of returns forcing pop stack-operations, and
- a set Σ_{int} of internal actions which do not use the stack.

The class of languages accepted by VPA is closed under Boolean operations and language inclusion is EXPTIME-complete.

Visibly context-free temporal logics

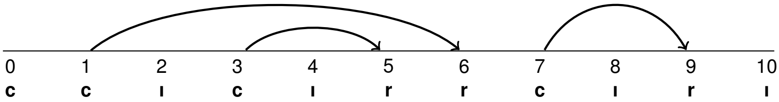
Temporal modalities for navigating over the nested hierarchical structure of a word over a pushdown alphabet: each call is associated with a matching return (if any) in a well-nested manner.



$\Sigma_{\text{call}} = \{c\}$, $\Sigma_{\text{ret}} = \{r\}$, and $\Sigma_{\text{int}} = \{\iota\}$.

Visibly context-free temporal logics

Temporal modalities for navigating over the nested hierarchical structure of a word over a pushdown alphabet: each call is associated with a matching return (if any) in a well-nested manner.



$$\Sigma_{\text{call}} = \{c\}, \Sigma_{\text{ret}} = \{r\}, \text{ and } \Sigma_{\text{int}} = \{\imath\}.$$

A relevant example: CaRet (a linear-time context-free extension of LTL).

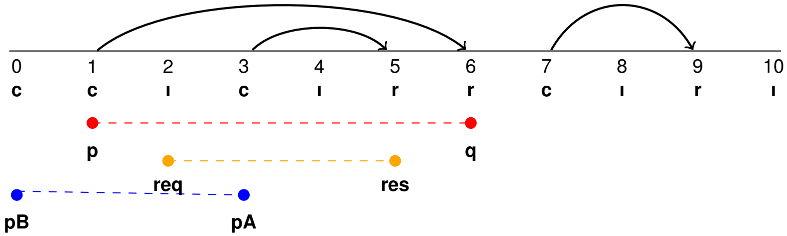
Alur et al. - A temporal logic of nested calls and returns - TACAS'04

CaRet provide versions of LTL modalities evaluated over two kinds of non-regular patterns on input words:

- abstract path: local computation within a procedure removing computation fragments corresponding to nested calls, and
- caller path: call-stack content at a given position of the input.

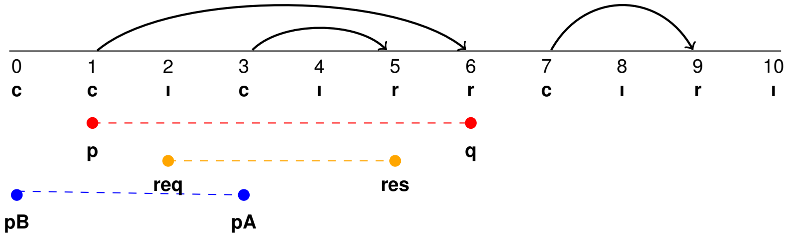
Satisfiability and visibly model-checking for CaRet are EXPTIME-complete.

Examples of context-free properties



Total correctness: if the pre-condition *p* holds when a procedure *A* is invoked, the procedure must return and the post-condition *q* must hold upon return.

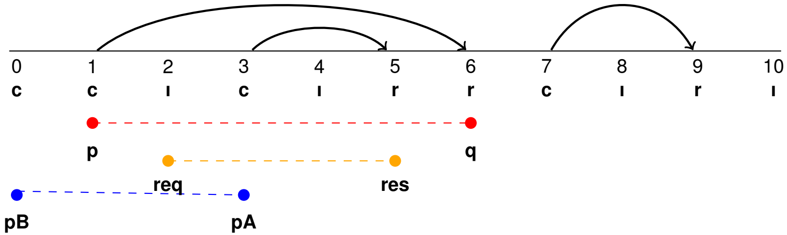
Examples of context-free properties



Total correctness: if the pre-condition *p* holds when a procedure *A* is invoked, the procedure must return and the post-condition *q* must hold upon return.

Local response: in the local computation of a procedure *A*, every request *req* is followed by a response *res*.

Examples of context-free properties



Total correctness: if the pre-condition *p* holds when a procedure *A* is invoked, the procedure must return and the post-condition *q* must hold upon return.

Local response: in the local computation of a procedure *A*, every request *req* is followed by a response *res*.

Call-stack inspection: a procedure *A* is invoked only if procedure *B* belongs to the call stack.

Interval Temporal Logic (ITL)

ITL provides an alternative framework for reasoning about the time. ITL assumes intervals, instead of points, as the primitive temporal entities.

It allows one to specify relevant temporal properties, e.g., durative actions, accomplishments, and temporal aggregations, which cannot be (naturally) expressed by point-based temporal logics.

Many application fields, including reasoning about action and change, planning, verification of programs, temporal and spatio-temporal databases.

Observation: an interval temporal logic (instead of a point-based one) is a natural choice for the specification and verification of context-free requirements.

- The distinctive feature of pushdown systems, namely, the matching of calls and returns, has a natural interval nature: it bounds computation stretches where local properties can be checked.

The landmark in interval temporal logic is Halpern and Shoham's modal logic of time intervals (HS).

- HS satisfiability is undecidable over all relevant classes of linear orders.

Model checking for HS against finite Kripke structures

The model checking (MC) problem for HS against finite Kripke structures has been systematically investigated only very recently.

- Each finite path of a Kripke structure is interpreted as an interval, whose labelling is defined on the basis of the labelling of the component states.
  - Homogeneity assumption: a proposition letter holds over an interval if and only if it holds over each component state.
- We focus on the state-based semantics, where time branches both in the future and in the past.
- MC for HS against finite-state Kripke structures is known to be decidable.

  A. Molinari et al. - Checking interval properties of computations - Acta Informatica 2016
- Intriguing open question: exact complexity of finite-state MC for full HS.
  - It has been proved that it is at least EXPSPACE-hard, but the only known upper bound is non-elementary.

Expressiveness of HS over (finite) Kripke structures

The state-based version of HS is expressively incomparable with LTL and CTL*, and strictly more expressive than finitary CTL* (variant of CTL* with quantification over finite paths).

L. Bozzelli et al. - Interval vs. Point Temporal Logic Model Checking: An Expressiveness Comparison - ACM Trans. Comput. Logic 2019

Expressiveness of HS over (finite) Kripke structures

The state-based version of HS is expressively incomparable with LTL and CTL*, and strictly more expressive than finitary CTL* (variant of CTL* with quantification over finite paths).

L. Bozzelli et al. - Interval vs. Point Temporal Logic Model Checking: An Expressiveness Comparison - ACM Trans. Comput. Logic 2019

Known alternative semantics for HS over (finite) Kripke structures:

Computation-tree semantics: interpreted over computation trees (it allows time to branch only in the future). Expressively equivalent to finitary CTL*.

Linear-time semantics: interpreted over the subpaths (intervals) of a given infinite path. Expressively equivalent to LTL (but much more succinct).

Contributions of the present work

Unification of the linear-time and branching-time semantics of HS in a common framework (binding HS, BHS for short).

- BHS enriches HS with a novel regular binding operator, which restricts the evaluation of a formula to the interval sub-structure induced by the current interval.
- The interval mapping takes into account both finite and infinite paths.

Contributions of the present work

- Unification of the linear-time and branching-time semantics of HS in a common framework (binding HS, BHS for short).
  - BHS enriches HS with a novel regular binding operator, which restricts the evaluation of a formula to the interval sub-structure induced by the current interval.
  - The interval mapping takes into account both finite and infinite paths.
- BHS with the state-based semantics is further extended in order to specify branching-time, context-free requirements of pushdown systems under the homogeneity and visibility assumptions (nested BHS).
  - Despite its simplicity (we just add to BHS a special proposition letter p_{wm} that captures finite intervals with well-matched pairs of calls and returns), such an extension of BHS is quite powerful.
  - Nested BHS differs from known context-free temporal logics where ad hoc modalities are exploited.

Contributions of the present work

- Unification of the linear-time and branching-time semantics of HS in a common framework (binding HS, BHS for short).
  - BHS enriches HS with a novel regular binding operator, which restricts the evaluation of a formula to the interval sub-structure induced by the current interval.
  - The interval mapping takes into account both finite and infinite paths.
- BHS with the state-based semantics is further extended in order to specify branching-time, context-free requirements of pushdown systems under the homogeneity and visibility assumptions (nested BHS).
  - Despite its simplicity (we just add to BHS a special proposition letter p_{wm} that captures finite intervals with well-matched pairs of calls and returns), such an extension of BHS is quite powerful.
  - Nested BHS differs from known context-free temporal logics where ad hoc modalities are exploited.
- We investigate expressiveness and Visibly Pushdown Model Checking (*VPMC*) for nested BHS. We show that *VPMC* for nested BHS is decidable.

The model: Kripke structures

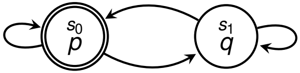
A Kripke structure is $\mathcal{K} = (Prop, S, s_0, R, Lab)$:

- $Prop$ is a finite set of proposition letters,
- S is a (possibly infinite) set of states and $s_0 \in S$ (initial state),
- $R \subseteq S \times S$ is a transition relation,
- $Lab : S \mapsto 2^{Prop}$ assigns to each state s the set of proposition letters that hold over it.

A path is a non-empty finite or infinite sequence of states $\pi = s_1 s_2 \ldots$ such that for all $1 \leq i < |\pi|$, $(s_i, s_{i+1}) \in E$. A path is initial if it starts at the initial state s_0.

For a path π, $\mathrm{first}(\pi)$ is the first state of π, and, in case π is finite, $\mathrm{last}(\pi)$ is the last state of π.

The trace $Lab(\pi)$ of a path $\pi = s_1 s_2 \ldots$ is the word over 2^{Prop} having the same length as $|\pi|$ given by $Lab(s_1)Lab(s_2)\ldots$.



The logic: binding HS (BHS)

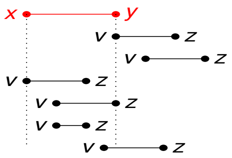
HS features one existential modality for each of the 13 possible ordering relations between pairs of intervals (Allen's relations), apart from equality.
BHS formulas ψ over a set of proposition letters *Prop*:

$$\psi ::= p \mid \neg\psi \mid \psi \wedge \psi \mid \langle X \rangle \psi \mid \mathbb{B}\psi$$

where $p \in Prop$, $\langle X \rangle$ is the existential temporal modality for the Allen relation $X \in \{A, L, B, E, D, O, \overline{A}, \overline{L}, \overline{B}, \overline{E}, \overline{D}, \overline{O}\}$, and $\mathbb{B}$ is the binding operator.

For any temporal modality $\langle X \rangle$, the dual universal modality $[X]$ is defined as $[X]\psi := \neg\langle X \rangle\neg\psi$.

Allen relation	HS	Definition w.r.t. interval structures	Example
MEETS	$\langle A \rangle$	$[x,y]\mathcal{R}_A[v,z] \iff y = v$	
BEFORE	$\langle L \rangle$	$[x,y]\mathcal{R}_L[v,z] \iff y < v$	
STARTED-BY	$\langle B \rangle$	$[x,y]\mathcal{R}_B[v,z] \iff x = v \wedge z < y$	
FINISHED-BY	$\langle E \rangle$	$[x,y]\mathcal{R}_E[v,z] \iff y = z \wedge x < v$	
CONTAINS	$\langle D \rangle$	$[x,y]\mathcal{R}_D[v,z] \iff x < v \wedge z < y$	
OVERLAPS	$\langle O \rangle$	$[x,y]\mathcal{R}_O[v,z] \iff x < v < y < z$	



BHS state-based semantics - 1

Allen's relations are interpreted over paths of a Kripke structure $\mathcal{K}$ in a natural way.
For paths π and π',

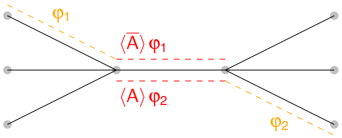
- MEETS: $\pi\, A\, \pi'$ if π is finite and $\text{last}(\pi) = \text{first}(\pi')$;
- BEFORE: $\pi\, L\, \pi'$ if π is finite and $\text{first}(\pi')$ is strictly reachable from $\text{last}(\pi)$;
- STARTED-BY: $\pi\, B\, \pi'$ if π' is a proper prefix of π;
- FINISHED-BY: $\pi\, E\, \pi'$ if π' is a proper suffix of π;
- CONTAINS: $\pi\, D\, \pi'$ if $\pi\, E\, \nu$ and $\nu\, B\, \pi'$ for some path ν (π' internal subpath of π);
- OVERLAPS: $\pi\, O\, \pi'$ if $\pi' = \nu \cdot \nu'$ for some paths ν and ν' such that $\pi\, E\, \nu$;
- Inverse relations: for each $\overline{X} \in \{\overline{A}, \overline{L}, \overline{B}, \overline{E}, \overline{D}, \overline{O}\}$, $\pi\, \overline{X}\, \pi'$ if $\pi'\, X\, \pi$.

BHS state-based semantics - 1

Allen's relations are interpreted over paths of a Kripke structure $\mathcal{K}$ in a natural way.
For paths π and π',

- MEETS: $\pi\ A\ \pi'$ if π is finite and $\text{last}(\pi) = \text{first}(\pi')$;
- BEFORE: $\pi\ L\ \pi'$ if π is finite and $\text{first}(\pi')$ is strictly reachable from $\text{last}(\pi)$;
- STARTED-BY: $\pi\ B\ \pi'$ if π' is a proper prefix of π;
- FINISHED-BY: $\pi\ E\ \pi'$ if π' is a proper suffix of π;
- CONTAINS: $\pi\ D\ \pi'$ if $\pi\ E\ \nu$ and $\nu\ B\ \pi'$ for some path ν (π' internal subpath of π);
- OVERLAPS: $\pi\ O\ \pi'$ if $\pi' = \nu \cdot \nu'$ for some paths ν and ν' such that $\pi\ E\ \nu$;
- Inverse relations: for each $\overline{X} \in \{\overline{A}, \overline{L}, \overline{B}, \overline{E}, \overline{D}, \overline{O}\}$, $\pi\ \overline{X}\ \pi'$ if $\pi'\ X\ \pi$.

Example: MEETS and MET-BY.



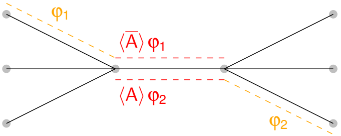
φ_1
$\langle\overline{A}\rangle\ \varphi_1$
$\langle A\rangle\ \varphi_2$
φ_2

BHS state-based semantics - 1

Allen's relations are interpreted over paths of a Kripke structure $\mathcal{K}$ in a natural way. For paths π and π',

> MEETS: $\pi\ A\ \pi'$ if π is finite and $\mathrm{last}(\pi) = \mathrm{first}(\pi')$;
>
> BEFORE: $\pi\ L\ \pi'$ if π is finite and $\mathrm{first}(\pi')$ is strictly reachable from $\mathrm{last}(\pi)$;
>
> STARTED-BY: $\pi\ B\ \pi'$ if π' is a proper prefix of π;
>
> FINISHED-BY: $\pi\ E\ \pi'$ if π' is a proper suffix of π;
>
> CONTAINS: $\pi\ D\ \pi'$ if $\pi\ E\ \nu$ and $\nu\ B\ \pi'$ for some path ν (π' internal subpath of π);
>
> OVERLAPS: $\pi\ O\ \pi'$ if $\pi' = \nu \cdot \nu'$ for some paths ν and ν' such that $\pi\ E\ \nu$;
>
> Inverse relations: for each $\overline{X} \in \{\overline{A}, \overline{L}, \overline{B}, \overline{E}, \overline{D}, \overline{O}\}$, $\pi\ \overline{X}\ \pi'$ if $\pi'\ X\ \pi$.

Example: MEETS and MET-BY.


φ_1
$\langle\overline{A}\rangle\,\varphi_1$
$\langle A\rangle\,\varphi_2$
φ_2

A binding context C is either ε (the empty context) or a path of $\mathcal{K}$.

> A path π belongs to the binding context C if either $C = \varepsilon$ or π is a subpath of C.

BHS state-based semantics - 2

For a Kripke structure $\mathcal{K}$, a binding context C, a path π in C, and an HS formula ψ, the satisfaction relation $\mathcal{K}, \pi, C \models \psi$ is defined as ($X \in \{A, L, B, E, D, O, \overline{A}, \overline{L}, \overline{B}, \overline{E}, \overline{D}, \overline{O}\}$):

- $\mathcal{K}, \pi, C \models p$ for $p \in Prop$ if $p \in Lab(\pi(i))$ for all $1 \leq i \leq |\pi|$ (homogeneity);
- semantics of Boolean connectives is as usual;
- $\mathcal{K}, \pi, C \models \langle X \rangle \psi$ if $\mathcal{K}, \pi', C \models \psi$ for some path π' in C such that $\pi \; X \; \pi'$;
- $\mathcal{K}, \pi, C \models \mathbb{B}\psi$ if $\mathcal{K}, \pi, \pi \models \psi$.

BHS state-based semantics - 2

For a Kripke structure $\mathcal{K}$, a binding context C, a path π in C, and an HS formula ψ, the satisfaction relation $\mathcal{K}, \pi, C \models \psi$ is defined as ($X \in \{A, L, B, E, D, O, \overline{A}, \overline{L}, \overline{B}, \overline{E}, \overline{D}, \overline{O}\}$):

- $\mathcal{K}, \pi, C \models p$ for $p \in Prop$ if $p \in Lab(\pi(i))$ for all $1 \leq i \leq |\pi|$ (homogeneity);
- semantics of Boolean connectives is as usual;
- $\mathcal{K}, \pi, C \models \langle X \rangle \psi$ if $\mathcal{K}, \pi', C \models \psi$ for some path π' in C such that $\pi \; X \; \pi'$;
- $\mathcal{K}, \pi, C \models \mathbb{B}\psi$ if $\mathcal{K}, \pi, \pi \models \psi$.

$\mathcal{K}$ is a model of ψ, written $\mathcal{K} \models \psi$, if $\mathcal{K}, \pi, \varepsilon \models \psi$ for all initial paths π.

- Modalities for relations in $\{B, E, D\}$ have linear-time semantics: select only subpaths of the current path.
- Modalities for relations in $\{A, L, O, \overline{A}, \overline{L}, \overline{B}, \overline{E}, \overline{D}, \overline{O}\}$:
  - branching-time semantics if the current context is empty: non-deterministically extend the current path in the future or in the past;
  - linear-time semantics if the current context is not empty: select only subpaths of the current binding context.

Checking pushdown systems

Let $Prop \supseteq \{\text{call}, \text{ret}, \text{int}\}$.
It induces the pushdown alphabet $\Sigma_{Prop} = \Sigma_{\text{call}} \cup \Sigma_{\text{ret}} \cup \Sigma_{\text{int}}$:

$$\Sigma_t = \{P \subseteq Prop \mid P \cap \{\text{call}, \text{ret}, \text{int}\} = \{t\}\} \text{ for all } t \in \{\text{call}, \text{ret}, \text{int}\}$$

Checking pushdown systems

Let *Prop* $\supseteq \{\text{call}, \text{ret}, \text{int}\}$.
It induces the pushdown alphabet $\Sigma_{Prop} = \Sigma_{\text{call}} \cup \Sigma_{\text{ret}} \cup \Sigma_{\text{int}}$:

$$\Sigma_t = \{P \subseteq Prop \mid P \cap \{\text{call}, \text{ret}, \text{int}\} = \{t\}\} \text{ for all } t \in \{\text{call}, \text{ret}, \text{int}\}$$

We show how to turn a *Visibly Pushdown System* (VPS) $\mathcal{PS}$ over *Prop* into an infinite-state Kripke structure $\mathcal{K}_{\mathcal{PS}}$.

Basic correspondence: the set of states of $\mathcal{K}_{\mathcal{PS}}$ is the set of configurations of $\mathcal{PS}$.

Checking pushdown systems

Let $\mathit{Prop} \supseteq \{\text{call}, \text{ret}, \text{int}\}$.
It induces the pushdown alphabet $\Sigma_{\mathit{Prop}} = \Sigma_{\text{call}} \cup \Sigma_{\text{ret}} \cup \Sigma_{\text{int}}$:

$$\Sigma_t = \{P \subseteq \mathit{Prop} \mid P \cap \{\text{call}, \text{ret}, \text{int}\} = \{t\}\} \text{ for all } t \in \{\text{call}, \text{ret}, \text{int}\}$$

We show how to turn a *Visibly Pushdown System* (VPS) $\mathcal{PS}$ over *Prop* into an infinite-state Kripke structure $\mathcal{K}_{\mathcal{PS}}$.

Basic correspondence: the set of states of $\mathcal{K}_{\mathcal{PS}}$ is the set of configurations of $\mathcal{PS}$.

Traces of $\mathcal{K}_{\mathcal{PS}}$ are words over the pushdown alphabet Σ_{Prop}.

An (initial) computation of $\mathcal{PS}$ is an (initial) path in $\mathcal{K}_{\mathcal{PS}}$.

The extended logic: nested BHS

Let $Prop \supseteq \{\text{call}, \text{ret}, \text{int}\}$.

A word over Σ_{Prop} is well-matched if each call has a matching return, and vice versa.

Nested BHS is just BHS + the special well-matched proposition letter p_{wm}

$\mathcal{K}, \pi, C \models p_{wm}$ if $Lab(\pi)$ is a finite well-matched word over Σ_{Prop}.

The extended logic: nested BHS

Let $Prop \supseteq \{\text{call}, \text{ret}, \text{int}\}$.

A word over Σ_{Prop} is well-matched if each call has a matching return, and vice versa.

Nested BHS is just BHS + the special well-matched proposition letter p_{wm}

$\mathcal{K}, \pi, C \models p_{wm}$ if $Lab(\pi)$ is a finite well-matched word over Σ_{Prop}.

We also consider the linear-time fragment of nested BHS (nested BHS_{lin}):
each temporal modality $\langle X \rangle \notin \{\langle B \rangle, \langle E \rangle, \langle D \rangle\}$ occurs in the scope of the binding operator $\mathbb{B}$.

The extended logic: nested BHS

Let $\mathit{Prop} \supseteq \{\mathrm{call}, \mathrm{ret}, \mathrm{int}\}$.

A word over Σ_{Prop} is well-matched if each call has a matching return, and vice versa.

Nested BHS is just BHS + the special well-matched proposition letter p_{wm}

$\mathcal{K}, \pi, C \models p_{wm}$ if $\mathit{Lab}(\pi)$ is a finite well-matched word over Σ_{Prop}.

We also consider the linear-time fragment of nested BHS (nested BHS$_{\mathrm{lin}}$): each temporal modality $\langle X \rangle \notin \{\langle B \rangle, \langle E \rangle, \langle D \rangle\}$ occurs in the scope of the binding operator $\mathbb{B}$.

Visibly pushdown model checking (*VPMC*) for nested BHS :
given a VPS $\mathcal{PS}$ over *Prop* and a nested BHS formula ψ over $\mathit{Prop} \cup \{p_{wm}\}$, check whether $\mathcal{K}_{\mathcal{PS}} \models \psi$ or not.

Specification of requirements in nested BHS

Some basic formulas to be used as building blocks (macros).

Left and right endpoints. Given a nested BHS formula ψ, formulas $\mathit{left}(\psi)$ and $\mathit{right}(\psi)$ constrain the current interval (computation) to be finite and ψ to hold at its left and right endpoints (point intervals), respectively:

$$\mathit{left}(\psi) := \langle\overline{\mathrm{A}}\rangle(\psi \wedge [B]\mathtt{false}) \text{ and } \mathit{right}(\psi) := \langle\mathrm{A}\rangle(\psi \wedge [B]\mathtt{false})$$

Minimal well-matched intervals. The formula θ_{mwm} characterizes those finite intervals whose left endpoint is a matched call and whose right endpoint is the matching return:

$$\theta_{mwm} := \mathit{left}(\mathrm{call}) \wedge \mathit{right}(\mathrm{ret}) \wedge p_{wm} \wedge [B]\neg p_{wm}$$

Procedural context interval. The formula θ_{pc} captures those computations π starting at a configuration s that precede the end (if any) of the procedural context associated with s, i.e., such that each non-first return position has a matched-call:

$$\theta_{pc} := \xi_{\mathrm{ret}} \wedge [B]\xi_{\mathrm{ret}} \qquad \xi_{\mathrm{ret}} := \mathit{right}(\mathrm{ret}) \rightarrow (\theta_{mwm} \vee \langle\mathrm{E}\rangle\,\theta_{mwm} \vee [E]\mathtt{false})$$

Expressive power of nested BHS: an example

In nested BHS, we can naturally express procedural-context/abstract/caller versions of CTL and CTL* requirements, which cannot be formulated in linear-time context-free temporal logics.

Example: a procedural context version of the CTL formula $\mathbf{E}(p_1 \mathrm{U} p_2)$.

Consider the condition: "*there is a computation* π *from the current configuration s such that* $p_1 \mathrm{U} p_2$ *holds along a prefix of* π *which precedes the end (if any) of the procedural context associated with s*".

Such a condition is captured by the formula:

$$\langle \mathrm{A} \rangle (\theta_{pc} \wedge [B] p_1 \wedge right(p_2))$$

Notice that $\langle \mathrm{A} \rangle$ plays the role of the existential path quantifier **E** of CTL* (state-based semantics of nested BHS).

Various other meaningful examples can be found in the paper.

Expressiveness of nested BHS

We compared nested $\mathsf{BHS}_{\mathsf{lin}}$ with known linear-time context-free extensions of LTL

R. Alur et al. - First-Order and Temporal Logics for Nested Words - LICS 2007

We considered the first-order logic for nested words FO_μ and the temporal logics CaRet, CaRet + within modality W, and the Nested-Word Temporal Logic NWTL.

It is known that NWTL and CaRet + W are expressively complete for FO_μ, while it is an open question whether the same holds for CaRet.

Expressiveness of nested BHS

We compared nested BHS$_{lin}$ with known linear-time context-free extensions of LTL

R. Alur et al. - First-Order and Temporal Logics for Nested Words - LICS 2007

We considered the first-order logic for nested words FO$_\mu$ and the temporal logics CaRet, CaRet + within modality W, and the Nested-Word Temporal Logic NWTL.

It is known that NWTL and CaRet + W are expressively complete for FO$_\mu$, while it is an open question whether the same holds for CaRet.

Theorem (Expressiveness of nested BHS)

We proved that:

- nested BHS$_{lin}$ is as expressive as FO$_\mu$;

- NWTL (resp., CaRet $+$ W) formulas can be translated in polynomial time into equivalent nested BHS$_{lin}$ formulas ψ, where, for CaRet formulas, ψ is of the form $\mathbb{B}\psi'$ for some nested HS formula ψ';

- nested BHS is strictly more expressive than FO$_\mu$;

- HS (and thus nested BHS) is strictly more expressive than standard CTL*.

VPMC for nested BHS: decision procedures

Let $\mathcal{PS}$ be a VPS and ψ be a nested BHS formula. ψ can be translated into a nondeterministic VPA (NVPA) accepting encodings of the computations of $\mathcal{PS}$ satisfying ψ.

VPMC for nested BHS: decision procedures

Let $\mathcal{PS}$ be a VPS and ψ be a nested BHS formula. ψ can be translated into a nondeterministic VPA (NVPA) accepting encodings of the computations of $\mathcal{PS}$ satisfying ψ.

Theorem

Given a VPS $\mathcal{PS}$ and a nested BHS formula ψ, one can construct an NVPA accepting the words encoding the computations π of $\mathcal{PS}$ such that $\mathcal{K}_{\mathcal{PS}}, \pi \models \psi$. Moreover, the VPMC problem for nested BHS is decidable with a non-elementary complexity.

VPMC for nested BHS: decision procedures

Let $\mathcal{PS}$ be a VPS and ψ be a nested BHS formula. ψ can be translated into a nondeterministic VPA (NVPA) accepting encodings of the computations of $\mathcal{PS}$ satisfying ψ.

Theorem

Given a VPS $\mathcal{PS}$ and a nested BHS formula ψ, one can construct an NVPA accepting the words encoding the computations π of $\mathcal{PS}$ such that $\mathcal{K}_{\mathcal{PS}}, \pi \models \psi$. Moreover, the VPMC problem for nested BHS is decidable with a non-elementary complexity.

Proof.

The first statement holds for nested $\mathrm{BHS}_{\mathrm{lin}}$ since it can be translated into FO_{μ} and the latter can be translated into NVPA with a non-elementary blow-up.
As for nested BHS, we exploit the above encoding, the closure of NVPA under Boolean operators and the operators $\langle X \rangle_{\mathcal{PS}}$, and the fact that a nested BHS formula can be seen as a nested HS formula with atomic formulas in nested $\mathrm{BHS}_{\mathrm{lin}}$.
Non-elementary hardness already holds for finite MC against $\mathrm{BHS}_{\mathrm{lin}}$: polynomial-time reduction from the universality problem for star-free regular expressions built from union, concatenation, and negation. $\square$

Conclusions and future work

Summary of the results

We devised a novel branching-time context-free logical framework to model check visibly pushdown systems (*VPMC*):

- it is based on an extension of HS with the state-based semantics over Kripke structures (under the homogeneity assumption);
- it strictly subsumes well-known linear-time context-free extensions of LTL;
- it is decidable although with a non-elementary complexity.

Future work

Is (nested) BHS strictly more expressive than (nested) HS?

Exact complexity of the *VPMC* problem for nested HS and its relevant fragments, and for nested BHS in terms of nesting depth of binding modality.

Expressiveness comparison of nested BHS and visibly pushdown μ-calculus, a known extension of the modal μ-calculus with future context-free modalities.

To relaxe the homogeneity assumption.