Extending propositional separation logic for robustness properties

Alessio Mansutti

LSV, CNRS, ENS Paris-Saclay

Paris - April 2019
What we will see

An extension of propositional separation logic that

- can express some interesting properties for program verification,
- is \( \text{PSpace} \)-complete,
- has very weak extensions that are \( \text{Tower} \)-hard.

A modal logic on trees that

- is \( \text{Tower} \)-complete,
- it is very easily captured by logics that were independently found to be \( \text{Tower} \)-complete.
Memory states

Separation Logic is interpreted over memory states \((s, h)\) where:

- **store**, \(s : \text{VAR} \rightarrow \text{LOC}\)
- **heap**, \(h : \text{LOC} \rightarrow_{\text{fin}} \text{LOC}\)

where \(\text{VAR} = \{x, y, z, \ldots\}\) set of (program) variables, \(\text{LOC}\) set of locations. \(\text{VAR}\) and \(\text{LOC}\) are countably infinite sets.

Disjoint heaps: \(\text{dom}(h_1) \cap \text{dom}(h_2) = \emptyset\)

Union of disjoint heaps \((h_1 + h_2)\): union of partial functions.

![Diagram](image-url)

Here, \(h(s(x)) = s(y)\)
Propositional Separation Logic $SL(\ast, \ast)$

$\phi := \neg \phi \mid \phi_1 \land \phi_2 \mid emp \mid x = y \mid x \leftrightarrow y \mid \phi_1 \ast \phi_2 \mid \phi_1 \ast \ast \phi_2$

$(s, h) \models \phi \ast \psi$

$(s, h) \models \phi \rightarrow \ast \psi$

Note: the satisfiability problem $SAT(SL(\ast, \ast))$ is PSpace-complete.
From where it started

**Theorem (Demri, Lozes, M. – 2018, Fossacs)**

\( SL(\ast, \rightarrow) \) enriched with \( \text{reach}(x, y) = 2 \) and \( \text{reach}(x, y) = 3 \) is undecidable.

- reduction from \( SL(\forall, \rightarrow) \) \hspace{1cm} (Brochenin et al.'12)
- \( SL(\ast, \rightarrow) + \text{reach}(x, y) = 2 \) is PSpace-complete \hspace{1cm} (Demri et al.'14)
Robustness Properties (Jansen, et al. – ESOP’17)

- \( \varphi \) comply with the **acyclicity** property iff every model of \( \varphi \) is acyclic.

- \( \varphi \) comply with the **garbage freedom** property iff in every model \((s,h) \models \varphi\), for each \( \ell \in \text{dom}(h) \) there is \( x \in v(\varphi) \) s.t. \( s(x) \) reaches \( \ell \).

**Checking for robustness properties** is ExpTime-complete for Symbolic Heaps with Inductive Predicates (IP).

Our Goal
Provide a similar result for **propositional** separation logic.
Checking for robustness properties is ExpTime-complete for Symbolic Heaps with Inductive Predicates (IP).

Our Goal
Provide a similar result for propositional separation logic.
We aim to an extension of propositional separation logic where

- satisfiability/entailment are decidable in PSpace (as $\text{SL}(\ast, \neg\ast)$)
- robustness properties reduce to one of these classical problems

**Known extensions**

$2\text{SL}(\ast, \neg\ast)$ $\leftarrow$ $\text{SL}(\ast, \neg\ast, \text{reach})$ $\leftarrow$ $\text{SL}(\ast, \neg\ast)$

$1\text{SL}(\ast, \neg\ast)$ $\leftarrow$ $\text{SL}(\ast, \neg\ast)$ $\leftarrow$ $\text{BSR}(\text{SL}(\ast, \neg\ast))^{\text{new}}$ $\leftarrow$ $\text{SL}(\ast, \text{reach})$ $\leftarrow$ $\text{SL}(\forall, \ast)$

Tower

PSpace

undecidable
Let’s start with reachability + 1 quantified variable

\[ (s, h) \models \text{reach}^+(x, y) \iff h^L(s(x)) = s(y) \text{ for some } L \geq 1 \]

\[ (s, h) \models \exists u \ \varphi \iff \text{there is } l \in \text{LOC} \text{ s.t. } (s[u \leftarrow l], h) \models \varphi \]

It is only possible to quantify over the variable name \( u \).

Robustness properties reduce to entailment

- **Acyclicity**: \( \varphi \models \neg \exists u \ \text{reach}^+(u, u) \)
- **Garbage freedom**: \( \varphi \models \forall u \ (\text{alloc}(u) \Rightarrow \bigvee_{x \in \text{fv}(\varphi)} \text{reach}(x, u)) \)

where \( u \not\in \text{fv}(\varphi) \) and

\[ \text{alloc}(x) \overset{\text{def}}{=} (x \leftrightarrow x) \rightarrow \bot \]

\[ \text{reach}(x, y) \overset{\text{def}}{=} x = y \lor \text{reach}^+(x, y) \]
Undecidability and Restrictions

Theorem (Demri, Lozes, M. – 2018, Fossacs)

$\text{SL}(\ast, \rightarrow) \text{ enriched with } \text{reach}(x, y) = 2 \text{ and } \text{reach}(x, y) = 3 \text{ is undecidable.}$

$\implies \text{SAT}(1\text{SL}(\ast, \rightarrow, \text{reach}^+)) \text{ is undecidable.}$

We syntactically restrict the logic so that $\text{reach}^+(x, y)$ is s.t.

**R1:** it does not appear on the right side of its first $\rightarrow \ast$ ancestor (seeing the formula as a tree)

- $\varphi \rightarrow (\psi \ast \text{reach}^+(u, u))$ violates R1

**R2:** if $x = u$ then $y = u$ (syntactically)

- $\text{reach}^+(u, x)$ violates R2

**Note:** robustness properties are still expressible (formulae as before)!
Results

1. SAT($1SL_{R1}^{R2}(\star, \neg\star, \text{reach}^+)$) is PSpace-complete
   - strictly subsumes $1SL(\star, \neg\star)$ and $SL(\star, \text{reach}^+)$. 

2. SAT($1SL_{R1}(\star, \neg\star, \text{reach}^+)$) is Tower-hard.

Proof Techniques

(1) extend the core formulae technique used for $SL(\star, \neg\star)$.

(2) reduction from “an auxiliary logic on trees”.
Core formulae technique

(and a bit of $1SL^{R_2}_{R_1}(\ast, \neg\ast, \text{reach}^+)$)
First order theories: Gaifman Locality Theorem

Theorem (Gaifman – 1982, Herbrand Symposium)

Every FO sentence is logically equivalent to a Boolean combination of local formulae.

- application of Ehrenfeucht-Fraïssé games

Relation between models

\[ M \leftrightarrow_n M' \] (partial iso. up to n)

\[ M \leftrightarrow \text{EF-games} \]

Duplicator has a winning strategy

(n round game)

\[ M \approx_n M' \] (n nested quantifiers)

Semantics of logic
First order theories: Gaifman Locality Theorem

Theorem (Gaifman – 1982, Herbrand Symposium)

Every FO sentence is logically equivalent to a Boolean combination of local formulae.

- application of Ehrenfeucht-Fraïssé games
“Locality theorem” for $\text{SL}(\ast, \neg \ast)$

**Theorem (Lozes, 2004 – Space)**

*Every formula of $\text{SL}(\ast, \neg \ast)$ is logically equivalent to a Boolean combination of core formulae.*

From this theorem we can get:

- expressive power results
- complexity result (small model property)
- axiomatisation

When considering extensions of the logic, we need to derive new core formulae and reprove the theorem.

$\implies$ It does not work (at all) for $\text{1SL}_{R1}^{R2}(\ast, \neg \ast, \text{reach}^+)$. 
Core formulae for $SL(\ast, \neg\ast)$

Fix $X \subseteq \text{VAR}$ and $\alpha \in \mathbb{N}^+$

$$\text{Core}(X, \alpha) \overset{\text{def}}{=} \left\{ \begin{array}{ll} x = y, & x \hookrightarrow y, \quad & \beta \in [0, \alpha], \\ \text{alloc}(x), & \text{size} \geq \beta \\ \end{array} \right\}$$

where $(s, h) \models \text{size} \geq \beta$ iff $\text{card}(\text{dom}(h)) \geq \beta$.

- indistinguishability relation:

  $(s, h) \leftrightarrow^X_\alpha (s', h')$ iff $\forall \varphi \in \text{Core}(X, \alpha), (s, h) \models \varphi$ iff $(s', h') \models \varphi$

- Both EF-game and winning strategy for Duplicator are hidden inside two (technical) elimination lemmas.
### Core formulae: $\ast$ elimination lemma

<table>
<thead>
<tr>
<th>Lemma</th>
</tr>
</thead>
<tbody>
<tr>
<td>Suppose $(s, h) \leftrightarrow_{\alpha}^{X} (s', h')$. Then,</td>
</tr>
<tr>
<td>for every $\alpha_1 + \alpha_2 = \alpha$ ($\alpha_1, \alpha_2 \in \mathbb{N}^+$), and every $h_1 + h_2 = h$, (Spoiler)</td>
</tr>
<tr>
<td>there are $h'_1 + h'_2 = h'$ such that (Duplicator)</td>
</tr>
<tr>
<td>$(s, h_1) \leftrightarrow_{\alpha_1}^{X} (s', h'<em>1)$ and $(s, h_2) \leftrightarrow</em>{\alpha_2}^{X} (s', h'_2)$.</td>
</tr>
</tbody>
</table>

- necessary to obtain a winning strategy for Duplicator
Core formulae: \(*\) elimination lemma

**Lemma**

Suppose \((s, h) \leftrightarrow^X_\alpha (s', h')\). Then,

for every \(\alpha_1 + \alpha_2 = \alpha\) (\(\alpha_1, \alpha_2 \in \mathbb{N}^+\)), and every \(h_1 + h_2 = h\), (Spoiler)

there are \(h'_1 + h'_2 = h'\) such that

\((s, h_1) \leftrightarrow^X_{\alpha_1} (s', h'_1)\) and \((s, h_2) \leftrightarrow^X_{\alpha_2} (s', h'_2)\). (Duplicator)

necessary to obtain a winning strategy for Duplicator

By \(\text{Relation} \iff \text{EF-games} \iff \text{Semantics}\) it leads to:

For every \(\varphi \in \text{Bool(Core}(X, \alpha_1))\) and \(\psi \in \text{Bool(Core}(X, \alpha_2))\)
there is \(\chi \in \text{Bool(Core}(X, \alpha_1 + \alpha_2))\) such that

\[ \varphi \ast \psi \iff \chi \]

**Note:** similar elimination lemma for \(\lnot \ast\).
Core formulae: after $\ast$ and $\ast\ast$ elimination

**Theorem**

For every $\varphi$ in $\text{SL}(\ast, \ast\ast)$:

1. there is an equivalent Boolean combination of core formulae.
2. for every $\alpha \geq |\varphi|$, $X \supseteq v(\varphi)$ and $(s, h) \leftrightarrow_{\alpha}^{X} (s', h')$,

$$
(s, h) \models \varphi \text{ iff } (s', h') \models \varphi.
$$

[2] allows to derive a small-model property which leads to a proof that $\text{SAT}(\text{SL}(\ast, \ast\ast))$ is in PSpace.
1SL_{R1}^{R2}(\ast, \neg\ast, \text{reach}^+) is in PSpace: Not so easy...

\[ \pi := x = y \ | \ x \leftrightarrow y \ | \ \text{emp} \ | \ \mathcal{A} \rightarrow \mathcal{C} \ (R1) \]
\[ \mathcal{C} := \pi \ | \ \mathcal{C} \land \mathcal{C} \ | \ \neg \mathcal{C} \ | \ \exists u \ \mathcal{C} \ | \ \mathcal{C} \ast \mathcal{C} \]
\[ \mathcal{A} := \pi \ | \ \text{reach}^+(v_1, v_2) \ | \ \mathcal{A} \land \mathcal{A} \ | \ \neg \mathcal{A} \ | \ \exists u \ \mathcal{A} \ | \ \mathcal{A} \ast \mathcal{A} \]

where if \( v_1 = u \) then \( v_2 = u \) (R2).

- Asymmetric \( \mathcal{A} \rightarrow \mathcal{C} \): design two sets of core formulae against
  - two \( \ast \) and two \( \exists \) elimination lemmas;
  - one \( \rightarrow \ast \) elimination lemma that glues the two set of core formulae.

- instead of “size \( \geq \beta \) s.t. \( \beta \in [1, \alpha] \)”, the \( \beta \)s of new core formulae are bounded by functions on \( \alpha \), e.g.

\[ \#\text{loop}(\beta) \geq \gamma \quad \gamma \in [1, \frac{1}{2} \alpha(\alpha + 3) - 1] \]

bounds are found by solving a set of recurrence equations.
Core formulae: Example on a toy logic

\[ \varphi := \neg \varphi \mid \varphi_1 \land \varphi_2 \mid \varphi_1 \ast \varphi_2 \mid \exists u \varphi \mid \text{alloc}(u) \mid \text{reach}^+(u, u) \]

Some formulae expressible in this logic:

- \( \text{size} \geq 0 \overset{\text{def}}{=} \top \)
- \( \text{size} \geq \beta + 1 \overset{\text{def}}{=} \exists u \left( \text{alloc}(u) \ast \text{size} \geq \beta \right) \)
- \( \text{reach}^+(u, u) = \beta \) iff there is a loop of size exactly \( \beta \) involving \( s(u) \).
- \( \#\text{loops}(\beta) \geq \gamma \overset{\text{def}}{=} \exists u \text{reach}^+(u, u) = \beta \ast \ldots \ast \exists u \text{reach}^+(u, u) = \beta \)
- \( \text{rem} \geq \beta \) iff there are at least \( \beta \) memory cells not in a loop.
Designing Core Formulae

- Fix \( \alpha \in \mathbb{N}^+ \)

- Let \( \text{Core}(\alpha) \) be the finite set of predicates:

\[
\begin{align*}
\text{rem} \geq \beta, \\
\#\text{loops}(\beta) \geq \gamma, \\
\#\text{loops} > \mathcal{R}(\alpha) \geq \gamma,
\end{align*}
\]

\[
\beta \in [1, \mathcal{R}(\alpha)], \\
gamma \in [1, \mathcal{L}(\alpha)]
\]

for some functions \( \mathcal{L} \) and \( \mathcal{R} \) in \([\mathbb{N} \rightarrow \mathbb{N}]\).

\[
\#\text{loops}_{> \beta} \geq \gamma = \exists u \text{ reach}^+(u, u) \geq \beta + 1 \ast \ldots \ast \exists u \text{ reach}^+(u, u) \geq \beta + 1
\]
Designing Core Formulae

- Fix $\alpha \in \mathbb{N}^+$
- Let $\text{Core}(\alpha)$ be the **finite** set of predicates:
  \[
  \begin{cases}
  \text{rem} \geq \beta, \\
  \#\text{loops}(\beta) \geq \gamma, \\
  \#\text{loops}_{>\mathcal{R}(\alpha)} \geq \gamma,
  \end{cases}
  \]
  \[
  \beta \in [1, \mathcal{R}(\alpha)], \\
  \gamma \in [1, \mathcal{L}(\alpha)]
  \]
  
  for some functions $\mathcal{L}$ and $\mathcal{R}$ in $[\mathbb{N} \rightarrow \mathbb{N}]$.

These formulae induce a partition on the heap:

- $\text{rem} \geq \beta$ speaks about memory cells not in a loop
- $\#\text{loops}(\beta) \geq \gamma$ speaks about locations in loops of size $\beta \in [1, \mathcal{R}(\alpha)]$
- $\#\text{loops}_{>\mathcal{R}(\alpha)} \geq \gamma$ speaks about locations in loops of size $> \mathcal{R}(\alpha)$.

$\#\text{loops}_{>\beta} \geq \gamma = \exists u \text{ reach}^+(u, u) \geq \beta + 1 \ast \ldots \ast \exists u \text{ reach}^+(u, u) \geq \beta + 1$
Lemma

Suppose $(s, h) \leftrightarrow^x_{\alpha} (s', h')$. Then,

for every $\alpha_1 + \alpha_2 = \alpha$ ($\alpha_1, \alpha_2 \in \mathbb{N}^+$), and every $h_1 + h_2 = h$, (Spoiler)

- Test the core formulae against the ∗ elimination lemma.
- standard-ish way of doing things in EF-games.
Find $\mathcal{R}$ and $\mathcal{L}$

**Lemma**

Suppose $(s, h) \leftrightarrow^x_{\alpha} (s', h')$. Then,

for every $\alpha_1 + \alpha_2 = \alpha$ ($\alpha_1, \alpha_2 \in \mathbb{N}^+$), and every $h_1 + h_2 = h$,  \hspace{1cm} (Spoiler)

Test the core formulae against the $\ast$ elimination lemma.

- standard-ish way of doing things in EF-games.

What happens to the locations corresponding to $\text{rem} \geq \beta$, when we split a heap?
Find $\mathcal{R}$ and $\mathcal{L}$

**Lemma**

Suppose $(s, h) \leftrightarrow_{\alpha}^X (s', h')$. Then,

for every $\alpha_1 + \alpha_2 = \alpha$ ($\alpha_1, \alpha_2 \in \mathbb{N}^+$), and every $h_1 + h_2 = h$, \((\text{Spoiler})\)

... 

- Test the core formulae against the $\ast$ elimination lemma.
- Standard-ish way of doing things in EF-games.

What happens to the locations corresponding to $\text{rem} \geq \beta$, when we split a heap?

They correspond to $\text{rem} \geq \beta$, also in the subheaps.
Find $\mathcal{R}$ and $\mathcal{L}$

Lemma

Suppose $(s, h) \leftrightarrow^X_\alpha (s', h')$. Then,

for every $\alpha_1 + \alpha_2 = \alpha$ ($\alpha_1, \alpha_2 \in \mathbb{N}^+$), and every $h_1 + h_2 = h$.

(Spoiler)

Test the core formulae against the $\ast$ elimination lemma.

Standard-ish way of doing things in EF-games.

What happens to the locations corresponding to $\text{rem} \geq \beta$, when we split a heap?

They correspond to $\text{rem} \geq \beta$, also in the subheaps.
Find $\mathcal{R}$ and $\mathcal{L}$

For $\mathcal{L}$, roughly speaking...

\[
\text{#loops...} \sim \begin{array}{c}
\text{h}_1 \\
\text{rem} \geq \beta
\end{array} + \begin{array}{c}
\text{h}_2 \\
\text{rem} \geq \beta
\end{array}
\]
Find $\mathcal{R}$ and $\mathcal{L}$

For $\mathcal{L}$, roughly speaking...

\[ \mathcal{L}(\alpha) \geq \max_{\alpha_1, \alpha_2 \in \mathbb{N}^+} (\mathcal{L}(\alpha_1) + \mathcal{L}(\alpha_2) + \mathcal{R}(\max(\alpha_1, \alpha_2))) \]
Find $R$ and $L$

We have the inequalities

$$
R(1) \geq 1 \quad R(\alpha) \geq \max_{\alpha_1, \alpha_2 \in \mathbb{N}^+} (R(\alpha_1) + R(\alpha_2))
$$

$$
L(1) \geq 1 \quad L(\alpha) \geq \max_{\alpha_1, \alpha_2 \in \mathbb{N}^+} (L(\alpha_1) + L(\alpha_2) + R(\alpha_1) + R(\alpha_2))
$$

Which admit $R(\alpha) = \alpha$ and $L(\alpha) = \frac{1}{2} \alpha (\alpha + 1)$ as a solution.

To satisfy the $\ast$ elimination lemma, build $\leftrightarrow_x^\alpha$ w.r.t.

$$
\begin{align*}
\text{rem} & \geq \beta, & \beta & \in [1, \alpha], \\
\#\text{loops}(\beta) & \geq \gamma, & \gamma & \in [1, \frac{1}{2} \alpha (\alpha + 1)], \\
\#\text{loops}_{>\alpha} & \geq \gamma,
\end{align*}
$$

(it is not a solution for the toy logic, we forgot the variable $u$!)
First recap

- SL(\(*, -, reach\))
  - undecidable
- 1SL_{R1}(\(*, -, reach^+\))
  - unknown
- 1SL_{R2}^{R2}(\(*, -, reach^+\))
  - PSpace-complete
- SL(\(*, reach\))
- 1SL(\(*, -*\))
  - PSpace-complete

- 1SL_{R1}^{R2}(\(*, -, reach^+\)) strictly generalise other PSpace-complete extensions of propositional separation logic.

- It can be used to check for robustness properties.
ALT: An auxiliary logic on trees
(or, what happens if we allow $\text{reach}^+(u, x)$)
Auxiliary logic on trees (ALT)

$$\varphi := \varphi_1 \land \varphi_2 \mid \neg \varphi \mid \langle U \rangle \varphi \mid \Diamond \varphi \mid \Diamond^* \varphi \mid \bigtriangledown \mid \approx$$

- interpreted on acyclic heaps (finite forests, encoding parent relation)
- one current node $n \in \text{LOC}$, one fixed target node $r \in \text{LOC}$
- $h, n \models_r \langle U \rangle \varphi$ iff there is $n' \in \text{LOC}$ s.t. $h, n' \models_r \varphi$
- $h, n \models_r \bigtriangledown$ iff $n \in \text{dom}(h)$ and $n$ reaches $r$ in at least one step
- $h, n \models_r \approx$ iff $n \in \text{dom}(h)$ and $n$ does not reach $r$ in at least one step
- $\Diamond \varphi \equiv (\text{size} = 1) \ast \varphi$, \hspace{1cm} $\Diamond^* \varphi \equiv \top \ast \varphi$

We prove that SAT(ALT) is a Tower-complete problem.
Auxiliary logic on trees (ALT)

We prove that SAT(ALT) is a Tower-complete problem.
What can **ALT** do?

Given a pointed model \((h, n)\) and a target node \(r\):

If we consider a portion of \(h\) with domain in \(\{n' \in \text{LOC} \mid h, n' \models \emptyset\}\), **ALT can only express** size bounds.

- Proof done with EF-games for **ALT**.

\[
\begin{align*}
\text{size}(\emptyset) & \geq 0 \quad \text{def} = \top \\
\text{size}(\emptyset) & \geq \beta + 1 \quad \text{def} = \langle U \rangle (\emptyset \land \Diamond (\neg \text{alloc} \land \text{size}(\emptyset) \geq \beta)) \\
\text{where } \text{alloc} & \text{ def} = \emptyset \lor \Delta.
\end{align*}
\]
What can ALT do?

- If $h, n \models_\mathbb{r} \Delta$, ALT can check bounds on the number of descendants and children of $n$:

  \[
  \#\text{desc} \geq \beta \overset{\text{def}}{=} \Diamond^* ([U] \neg \otimes \land \Delta \land \Diamond (\neg \text{alloc} \land \text{size}(\otimes) \geq \beta))
  \]

  \[
  \#\text{child} \geq 0 \overset{\text{def}}{=} \top
  \]

  \[
  \#\text{child} \geq \beta + 1 \overset{\text{def}}{=} \#\text{desc} \geq \beta + 1 \land \neg \Diamond^\beta (\Delta \land \neg \#\text{desc} \geq 1)
  \]

- Easy to encode words as acyclic memory states

```
abaa
```

```
\[ \begin{array}{c}
  a & b & a & a \\
  \downarrow & \downarrow & \downarrow & \downarrow \\
  r
\end{array} \]
```
PITL (Moszkowski’83)

\[ \varphi := pt \mid a \mid \varphi_1 \mid \varphi_2 \mid \neg \varphi \mid \varphi_1 \land \varphi_2 \]

- interpreted on finite non-empty words over a finite alphabet \( \Sigma \)
- \( w \models pt \iff |w| = 1 \)
- \( w \models a \iff \text{first letter of } w \text{ is } a \in \Sigma \) (locality principle)
- \( w \models \varphi_1 \mid \varphi_2 \iff w[1:j] \models \varphi_1 \text{ and } w[j:|w|] \models \varphi_2 \)
  for some \( j \in [1,|w|] \)

Note: SAT(PITL) is Tower-complete.
Reducing PITL to ALT

- Set of models encoding words can be characterised in ALT

- However, difficult to translate $\varphi_1 | \varphi_2$!

After the cut, left side does not reach $r$ anymore.

$\Rightarrow$ nodes on the left side satisfy $\otimes$

$\Rightarrow$ We cannot express the satisfaction of $\varphi_1$. 
PITL to ALT: alternative semantics for PITL

- A marked representation of $a \in \Sigma$

$$w_1 \ldots w_{j-1} w_j w_{j+1} \ldots [w|w|]$$

- $\varphi \mid \psi$ on standard semantics:

$$\begin{array}{c}
\varphi_1 \\
\underbrace{w_1 \ldots w_{j-1} w_j}_{\varphi_1} \\
\varphi_2 \\
\underbrace{w_j w_{j+1} \ldots [w|w|]}_{\varphi_2}
\end{array}$$

- $\varphi \mid \psi$ on marked semantics

$$\begin{array}{c}
\varphi_1 \\
\underbrace{w_1 \ldots w_{j-1} [w_j w_{j+1} \ldots [w|w|]]}_{\varphi_1} \\
\varphi_2 \\
\underbrace{w_j w_{j+1} \ldots [w|w|]}_{\varphi_2}
\end{array}$$

- alternative semantics is equivalent to the original one.
**ALT, marking an element**

- Given an alphabet $\Sigma = \{a_1, \ldots, a_n\}$, $a_i$ and $[a_i]$ are encoded as

- $\Rightarrow$ marking a character $\sim$ removing a single child.

- SAT(PITL) can be reduced to SAT(ALT),
  (translated formula is in 2ExpSpace if $\Sigma$ is coded in binary)

- $\Rightarrow$ ALT is Tower-complete (upper-bound from MSO).
Some logics that are Tower-hard

- It is easy to see that ALT is a fragment of $1SL_{R1}(*, -, reach^+)$:
  
  fix $x \in \text{VAR}$ to play the role of the target node $r$,
  
  $$
  \langle U \rangle \varphi \equiv \exists u \varphi \quad \triangle \equiv \text{reach}^+(u, x) \quad \ominus \equiv \text{alloc}(u) \land \neg \triangle
  $$

  + impose acyclic heaps: $\neg \exists u \text{reach}^+(u, u)$.

- ALT is a fragment of $MSL(*, \Diamond, \langle U \rangle)$

- $\text{ALT} \preceq_{\text{SAT}} \text{MLH}(*, \Diamond, \langle U \rangle)$ with modal depth 2.
  
  (then $\ast, \exists u, \text{alloc}(u), \text{alloc}^2(u)$ is Tower-c.)

- $\text{ALT} \preceq_{\text{SAT}} \text{QCTL}(U)$ without imbricated until operators $U$
  
  (or $\text{QCTL}(\text{EF})$ with 2 imbrication of $\text{EF}$)

  **Note:** in these results $\ast$ can always be replaced with $\Diamond$ and $\Diamond^\ast$. 
- ALT improves the understanding of some Tower-complete logics.

- It seems to be an interesting tool to prove Tower-hardness.