The Effects of Adding Reachability Predicates in Propositional Separation Logic

A. Mansutti
Joint work with S. Demri and E. Lozes

INFINI - November 2017

Motivation

- Many tools support separation logic as an assertion language;
- Growing demand to consider more powerful extensions;
- Focus of the community:
  - user-defined inductive predicates;
  - magic wand operator $-\!\!*$;
  - closure under boolean connectives.

Results

We consider propositional separation logic SL($*$,$-\!\!*$)

$+$

list segment predicate `ls`.

We show that its satisfiability problem is undecidable, but removing $-\!\!*$ makes the logic PSPACE-complete.

Separation logic as an assertion language

Verification of imperative programs based on **Hoare triples**:

$$\{P\}\ C\ \{Q\}$$

where C is a program and P, Q are **assertions** in some logical language.

Any (memory) state that satisfies P will satisfy Q after being modified by C.

Hoare calculus: Proof rules manipulating Hoare triples.

Separation logic as an assertion language

The so-called **frame rule**

$$\frac{\{P\}\ C\ \{Q\}}{\{F \wedge P\}\ C\ \{F \wedge Q\}}$$

fails in standard Hoare logic: C can change the satisfaction of F.

Separation logic as an assertion language

The so-called **frame rule**

$$\frac{\{P\}\ C\ \{Q\}}{\{F \wedge P\}\ C\ \{F \wedge Q\}}$$

fails in standard Hoare logic: C can change the satisfaction of F.

Separation logic add the notion of **separation** $(*)$ of a state, so that the frame rule

$$\frac{\{P\}\ C\ \{Q\} \quad \mathsf{modv}(C) \cap \mathsf{fv}(F) = \emptyset}{\{F * P\}\ C\ \{F * Q\}}$$

is valid.

Separation logic

Separation logic is interpreted over **memory states** (s, h) where:

- s is a store, $s : \mathsf{PVAR} \to \mathsf{LOC}$;
- h is a heap, $h : \mathsf{LOC} \to_{\mathsf{fin}} \mathsf{LOC}$.

where LOC and PVAR are countable infinite sets, e.g. $\mathbb{N}$.

Propositional separation logic

Syntax:

$$\phi := \ \neg\phi \ \mid \ \phi_1 \wedge \phi_2 \ \mid \ x = y \ \mid \ \texttt{emp} \ \mid \ x \mapsto y \ \mid \ \phi_1 * \phi_2 \ \mid \ \phi_1 \mathrel{-\!\!*} \phi_2$$

Semantics: standard for $\neg$ and $\wedge$,

$$\begin{array}{lcl}
(s,h) \models x = y & \iff & s(x) = s(y) \\
(s,h) \models \texttt{emp} & \iff & \mathrm{dom}(h) = \emptyset \\
(s,h) \models x \mapsto y & \iff & h(s(x)) = s(y) \text{ and } \mathrm{dom}(h) = \{x\}
\end{array}$$

$$\begin{array}{lcl}
(s,h) \models \phi_1 * \phi_2 & \iff & \exists h_1, h_2 \text{ s.t. } \ h = h_1 + h_2 \ \text{ and} \\
 & & (s,h_1) \models \phi_1 \text{ and } (s,h_2) \models \phi_2
\end{array}$$

$$\begin{array}{lcl}
(s,h) \models \phi_1 \mathrel{-\!\!*} \phi_2 & \iff & \forall h' \text{ if } h, h'\text{are disjoint and } (s,h') \models \phi_1 \\
 & & \ \text{then } (s, h + h') \models \phi_2
\end{array}$$

SL + Reachability predicates

$$
\begin{array}{ll}
\multicolumn{2}{l}{(s, h) \models \mathtt{ls}(\mathtt{x}, \mathtt{y})} \\
\iff & \text{if } s(\mathtt{x}) = s(\mathtt{y}) \text{ then } h \text{ is empty, otherwise} \\
& h = \{\ell_0 \mapsto \ell_1, \ell_1 \mapsto \ell_2, \ldots, \ell_{n-1} \mapsto \ell_n\} \text{ with } n \geq 1, \\
& \ell_0 = s(\mathtt{x}),\ \ell_n = s(\mathtt{y}) \text{ and for all } i \neq j \in [0, n],\ \ell_i \neq \ell_j
\end{array}
$$

$$
\begin{array}{ll}
\multicolumn{2}{l}{(s, h) \models \mathtt{reach}(\mathtt{x}, \mathtt{y})} \\
\iff & h \sqsupseteq \{s(\mathtt{x}) \mapsto \ell_1, \ell_1 \mapsto \ell_2, \ldots, \ell_{n-1} \mapsto s(\mathtt{y})\}
\end{array}
$$

$$
\begin{array}{ll}
\multicolumn{2}{l}{(s, h) \models \mathtt{reach}^{+}(\mathtt{x}, \mathtt{y})} \\
\iff & h \sqsupseteq \{s(\mathtt{x}) \mapsto \ell_1, \ell_1 \mapsto \ell_2, \ldots, \ell_{n-1} \mapsto s(\mathtt{y})\} \text{ with } n \geq 1
\end{array}
$$

Reachability predicates

- SL($*$,$-\!\!*$,$\texttt{ls}$) and SL($*$,$-\!\!*$,$\texttt{reach}$) are interdefinable;
- both logics can be seen as fragments of SL($*$,$-\!\!*$,$\texttt{reach}^+$).

Main contribution:

- We show the undecidability of SL($*$,$-\!\!*$,$\texttt{ls}$)
- and the PSPACE-completeness of SL($*$,$\texttt{reach}^+$).

Undecidability: Reduction of SL($\forall$,$-\!\!*$) to SL($*$,$-\!\!*$,$\mathtt{ls}$)

We consider the first-order extension of SL($-\!\!*$) obtained by adding the universal quantifier $\forall$.

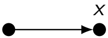
$(s, h) \models \forall x.\phi$ if and only if for all $\ell \in \mathsf{LOC}$, $(s[x \leftarrow \ell], h) \models \phi$

The satisfiability problem for SL($\forall$,$-\!\!*$) is undecidable. *(IAC 2012)*

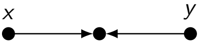
Undecidability: Reduction of SL($\forall$,$-\!\!*$) to SL($*$,$-\!\!*$,$\mathtt{ls}$)

Suppose we can express the following properties in SL($*$,$-\!\!*$,$\mathtt{ls}$)

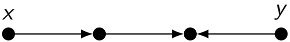
$\mathtt{alloc}^{-1}(x)$:



$n(x) = n(y)$:



$n(x) \hookrightarrow n(y)$:



Then we can encode formulae of SL($\forall$,$-\!\!*$) in SL($*$,$-\!\!*$,$\mathtt{ls}$) by using part of the heap to mimic the store's updates.

Translation from SL($\forall$,$\mathrel{-\!\!*}$) to SL($*$,$\mathrel{-\!\!*}$,$\mathtt{ls}$)

Formula ψ of SL($\forall$,$\mathrel{-\!\!*}$) with variables $x_1, \dots, x_q$.
For the translation we use $X \supseteq \{x_1, \dots x_q\}$ variables.

$$
\begin{aligned}
\mathrm{T}(\psi_1 \wedge \psi_2, X) &\overset{\text{def}}{=} \mathrm{T}(\psi_1, X) \wedge \mathrm{T}(\psi_2, X) \\
\mathrm{T}(\neg\psi, X) &\overset{\text{def}}{=} \neg\mathrm{T}(\psi, X) \\
\mathrm{T}(\mathtt{x}_i = \mathtt{x}_j, X) &\overset{\text{def}}{=} n(\mathtt{x}_i) = n(\mathtt{x}_j) \\
\mathrm{T}(\mathtt{x}_i \hookrightarrow \mathtt{x}_j, X) &\overset{\text{def}}{=} n(\mathtt{x}_i) \hookrightarrow n(\mathtt{x}_j) \\
\mathrm{T}(\forall\mathtt{x}_i\ \psi, X) &\overset{\text{def}}{=} (\mathtt{alloc}(\mathtt{x}_i) \wedge \mathtt{size} = 1) \mathrel{-\!\!*} (\mathrm{OK}(X) \Rightarrow \mathrm{T}(\psi, X))
\end{aligned}
$$

where $\mathrm{OK}(X)$ is the formula $(\bigwedge_{i \neq j} \mathtt{x}_i \neq \mathtt{x}_j) \wedge (\bigwedge_i \neg\mathtt{alloc}^{-1}(\mathtt{x}_i))$

Translation from SL($\forall$,$-\!\!*$) to SL($*$,$-\!\!*$,$\mathtt{ls}$)

To correctly translate $\mathrm{T}(\psi_1 -\!\!* \psi_2, X)$ we need one copy $\bar{x}_i$ of each variable x_i.
The translation

$$
\begin{gathered}
(\mathtt{ALLOC\_ONLY}(\overline{\mathit{fv}(\psi_1)}) \wedge \mathrm{T}(\psi_1, X)[\mathtt{x} \leftarrow \bar{\mathtt{x}}]) -\!\!* \\
(((\bigwedge_{\mathtt{z} \in \mathit{fv}(\psi_1)} n(\mathtt{z}) = n(\bar{\mathtt{z}})) \wedge \mathrm{OK}(X)) \Rightarrow \\
(\mathtt{DEALLOC\_ONLY}(\overline{\mathit{fv}(\psi_1)}) * \mathrm{T}(\psi_2, X)))
\end{gathered}
$$

Memory states

Separation logic is interpreted over **memory states** (s, h) where:

- s is a store, $s : \mathrm{PVAR} \to \mathrm{LOC}$;
- h is a heap, $h : \mathrm{LOC} \to_{\mathrm{fin}} \mathrm{LOC}$.

where LOC and PVAR are countable infinite sets.

Generalized memory states

Separation logic interpreted over **generalized memory states** (L, s, h) where:

- s is a store, $s : \mathrm{PVAR} \rightarrow L$;
- h is a heap, $h : L \rightarrow_{\mathrm{fin}} L$.

where L and PVAR are countable infinite sets.

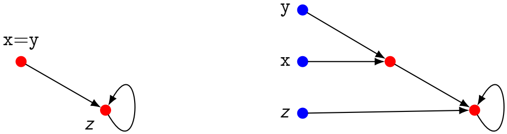
Generalized memory states: Encoding relation

$X = \{\mathtt{x}_1, \overline{\mathtt{x}_1}, \ldots, \mathtt{x}_q, \overline{\mathtt{x}_q}\}$, $Y \subseteq \{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$.

$(\mathsf{LOC}_1, s_1, h_1) \rhd_q^Y (\mathsf{LOC}_2, s_2, h_2)$ if it holds that:

- $\mathsf{LOC}_1 = \mathsf{LOC}_2 \setminus \{s_2(\mathtt{x}) \mid \mathtt{x} \in X\}$,
- for all $\mathtt{x}, \mathtt{y} \in X$, $s_2(\mathtt{x}) \neq s_2(\mathtt{y})$,
- $h_2 = h_1 + \{s_2(\mathtt{x}) \mapsto s_1(\mathtt{x}) \mid \mathtt{x} \in Y\}$.

Example: $Y = \{x, y, z\}$, ● $\in \mathsf{LOC}_1$, ● $\in \mathsf{LOC}_2$



Undecidability result

Lemma

$X = \{\mathtt{x}_1, \overline{\mathtt{x}_1}, \ldots, \mathtt{x}_q, \overline{\mathtt{x}_q}\}$, $Y \subseteq \{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$, ψ be a formula in $\mathrm{SL}(\forall, -\!\!*)$ with free variables among Y that does not contain any bound variable of ψ and $(\mathsf{LOC}_1, s_1, h_1) \rhd_q^Y (\mathsf{LOC}_2, s_2, h_2)$.

We have $(s_1, h_1) \models \psi$ iff $(s_2, h_2) \models \mathrm{T}(\psi, X)$.

Undecidability result

Lemma

$X = \{\mathtt{x}_1, \overline{\mathtt{x}_1}, \ldots, \mathtt{x}_q, \overline{\mathtt{x}_q}\}$, $Y \subseteq \{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$, ψ be a formula in $\mathrm{SL}(\forall, -\!\!*)$ with free variables among Y that does not contain any bound variable of ψ and $(\mathsf{LOC}_1, s_1, h_1) \rhd_q^Y (\mathsf{LOC}_2, s_2, h_2)$.

We have $(s_1, h_1) \models \psi$ iff $(s_2, h_2) \models \mathrm{T}(\psi, X)$.

Theorem

A closed formula ψ of SL($\forall$,$-\!\!*$) with variables in $\{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$ is satisfiable whenever

$$\bigwedge_{i \in [1,q]} (\neg \mathtt{alloc}(\mathtt{x}_i) \wedge \neg \mathtt{alloc}(\overline{\mathtt{x}_i})) \wedge \mathrm{OK}(X) \wedge \mathrm{T}(\psi, X)$$

is satisfiable.

Expressing the auxiliary atomic predicates

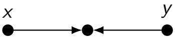
$n(x) = n(y)$, $n(x) \hookrightarrow n(y)$, $\texttt{alloc}^{-1}(x)$ definable in $\mathsf{SL}(*, \mathrel{-\!\!*}, \texttt{ls})$.

Idea: I can express that there exists a subheap of size n that satisfies a formula ϕ with $[\phi]_n \triangleq (\phi \wedge \texttt{size} = n) * \top$.

Example: $n(x) = n(y)$ expressed with

$$[\texttt{alloc}(x) \wedge \texttt{alloc}(y) \wedge \psi]_2$$

where ψ exactly characterize all the heaps of size 2 where it holds



Results

The following fragments have undecidable satisfiability problem:

- $\mathsf{SL}(*, \mathrel{-\!\!*}) + n(x) = n(y)$, $n(x) \hookrightarrow n(y)$ and $\mathtt{alloc}^{-1}(x)$;
- $\mathsf{SL}(*, \mathrel{-\!\!*}, \mathtt{ls})$;
- $\mathsf{SL}(*, \mathrel{-\!\!*}) + \mathtt{reach}(x, y) = 2$ and $\mathtt{reach}(x, y) = 3$;

We consider now SL($*$,$\mathtt{reach}^{+}$)

To show decidability:

- Find properties that can be expressed using $*$ and $\mathtt{reach}^{+}$ and make atomic (test) formulae for these properties;
- $*$ elimination: show that boolean combinations of these fomulae are sufficiently expressive to capture SL($*$,$\mathtt{reach}^{+}$);
- show a small-model property for the logic of test formulae. Apply it to SL($*$,$\mathtt{reach}^{+}$).

Actually, we study SL($*$,$\mathtt{reach}^{+}$,$\mathtt{alloc}$). This logic is at least as expressive as SL($*$,$-\!\!*$).

Example: SL($*$,$-\!*$)

In (standard) separation logic we can express:

- $\texttt{size} \geq \beta$, i.e. that the heap has size at least β:

$$\neg\texttt{emp} * \neg\texttt{emp} * \ldots * \neg\texttt{emp} \qquad \beta \text{ times}$$

- $\texttt{alloc}(x)$, i.e. $s(x)$ is in the domain of definition of h:

$$(x \mapsto x) -\!* \perp$$

- $x \hookrightarrow y$, i.e. $h(s(x)) = s(y)$:

$$x \mapsto y * \top$$

where $\top \equiv \texttt{emp} \vee \neg\texttt{emp}$.

Example: SL($*$,$-\!\!*$)

In (standard) separation logic we can express:

- s
- a n:
- $x \hookrightarrow y$, i.e. $h(s(x)) = s(y)$:

$$x \mapsto y * \top$$

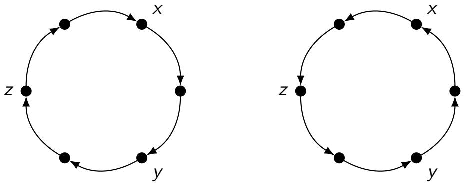
where $\top \equiv \texttt{emp} \vee \neg\texttt{emp}$.

Each Separation Logic formula is equivalent to a boolean combinations of formulae of the form

$$x = y,\ \texttt{alloc}(x),\ x \hookrightarrow y,\ \texttt{size} \geq \beta.$$

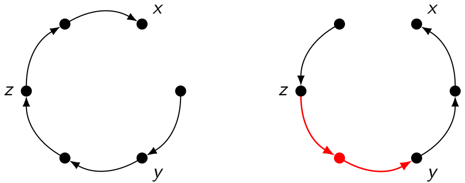
This leads to PSPACE-completeness for the satisfiability problem of SL formulae.

SL($*$,reach$^+$,alloc): What can be distinguished?



- Same $\texttt{reach}^+$ formulae are satisfied;
- $(\texttt{alloc}(x) \wedge \texttt{size} = 1) * \texttt{reach}^+(z, y)$ satisfied only by the second memory state.

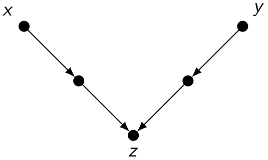
SL($*$,$\texttt{reach}^+$,$\texttt{alloc}$): What can be distinguished?

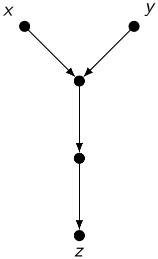


- Same $\texttt{reach}^+$ formulae are satisfied;
- $(\texttt{alloc}(x) \land \texttt{size} = 1) * \texttt{reach}^+(z, y)$ satisfied only by the second memory state.

The order in which variables are reached from a variable is important!

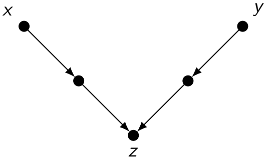
SL($*$,$\texttt{reach}^+$,$\texttt{alloc}$): What can be distinguished?

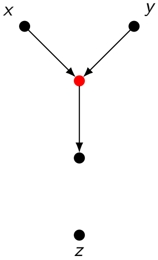




- Same $\texttt{reach}^+$ formulae are satisfied;
- $\texttt{size} = 1 * (\neg\texttt{reach}^+(x,z) \wedge \neg\texttt{reach}^+(y,z))$ satisfied only by the second memory state.

SL($*$,reach$^+$,alloc): What can be distinguished?

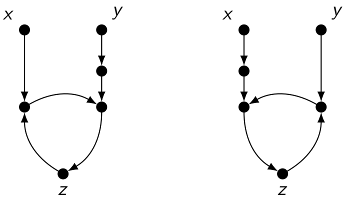




- Same reach$^+$ formulae are satisfied;
- $\mathtt{size} = 1 * (\neg\mathtt{reach}^+(x,z) \wedge \neg\mathtt{reach}^+(y,z))$ satisfied only by the second memory state.

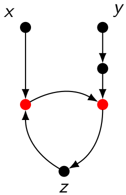
The existence of “shared paths” between variables is important!

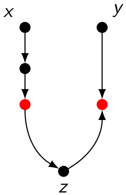
SL($*$,$\mathtt{reach}^+$,$\mathtt{alloc}$): What can be distinguished?



- Same $\mathtt{reach}^+$ formulae are satisfied;
- Same “order”, same “shared path”;
- $\mathtt{size} = 1 * (\neg\mathtt{reach}^+(z,z) \wedge \mathtt{alloc}(z) \wedge \mathtt{reach}^+(x,z))$ satisfied only by the second memory state.

SL($*$,$\texttt{reach}^+$,$\texttt{alloc}$): What can be distinguished?



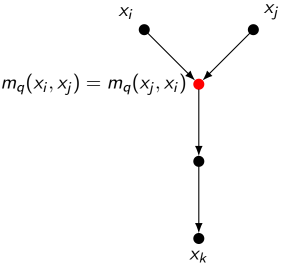


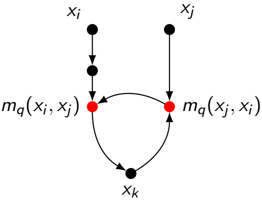
- Same $\texttt{reach}^+$ formulae are satisfied;
- Same “order”, same “shared path”;
- $\texttt{size} = 1 * (\neg\texttt{reach}^+(z,z) \wedge \texttt{alloc}(z) \wedge \texttt{reach}^+(x,z))$ satisfied only by the second memory state.

The existence of “meet points” is important!

Meet points

Memory state (s, h). Set of variables $\{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$.
We define meet-point $[\![m_q(\mathtt{x}_i, \mathtt{x}_j)]\!]_{s,h}$.





Test formulae

Given $\{x_1, \ldots, x_q\}$ and $\alpha \in \mathbb{N}$, we define $\mathsf{Test}(q, \alpha)$ as the set of following test formulae:

$$v = v' \quad v \hookrightarrow v' \quad \mathtt{alloc}(v) \quad \mathtt{sees}_q(v, v') \geq \beta + 1 \quad \mathtt{sizeR}_q \geq \beta,$$

where $\beta \in [1, \alpha]$ and v, v' are variables x_i or meet points $m_q(x_i, x_j)$, for $i, j \in [1, q]$.

Theorem (that we want to prove)

Let ψ be in SL($*$, $\mathtt{reach}^+$, $\mathtt{alloc}$) built over the variables in $x_1, \ldots, x_q$. Then ψ is logically equivalent to a boolean combination of test formulae from $\mathsf{Test}(q, |\psi|)$.

Test formulae: $\mathtt{sees}_q$

$$(s, h) \models \mathtt{sees}_q(v, v') \geq \beta + 1$$

if and only if

- $[\![v']\!]^q_{s,h}$ is the first location correspondant to program variables x_i or meet points $m_q(x_i, x_j)$ reached from $[\![v]\!]^q_{s,h}$;
- the path from $[\![v]\!]^q_{s,h}$ to $[\![v']\!]^q_{s,h}$ is at least of length $\beta + 1$.

Recall: The order in which variables are reached from a variable is important!

Test formulae: $\mathtt{sizeR}_q$

$$(s,h) \models \mathtt{sizeR}_q \geq \beta$$

if and only if the number of locations in $\mathrm{dom}(h)$ that are not corresponding to program variables x_i or in the path between two program variables x_i, x_j is greater or equal than β, where $\beta \in [1,\alpha]$, $i,j \in [1,q]$.

Rationale:

$$\varphi_{x,y} \;=\; \mathtt{reach}(x,y) = 3 \wedge \mathtt{alloc}(y) \wedge \neg\mathtt{reach}(y,x)$$

$$\varphi_{x,y} \wedge (\varphi_{x,y} * \mathtt{size} \geq 4)$$

Atomic formulae are combinations of test formulae

Lemma

Given α, $q \geq 1$, $i, j \in [1, q]$, for any atomic formula among $\mathtt{reach}^{+}(x_i, x_j)$, $\mathtt{ls}(x_i, x_j)$, $\mathtt{reach}(x_i, x_j)$ and $\mathtt{size} \geq \beta$ with $\beta \leq \alpha$, there is a Boolean combination of test formulae from $\mathsf{Test}(q, \alpha)$ *logically equivalent to it.*

For example, $\mathtt{reach}^{+}(\mathtt{x}_i, \mathtt{x}_j)$ can be shown equivalent to

$$\bigvee_{\substack{v_1, \ldots, v_n \in \mathrm{Terms}_q,\\ \mathtt{x}_i = v_1, \mathtt{x}_j = v_n}} \bigwedge_{1 \leq \delta \leq n-1} \mathtt{sees}_q(v_\delta, v_{\delta+1}) \geq 1.$$

where Terms_q is the set of program varibles x_i and meet points $m_q(x_i, x_j)$, $i, j \in [1, q]$.

Indistinguishability of two memory states

Lemma

Let $q, \alpha, \alpha_1, \alpha_2 \geq 1$ with $\alpha = \alpha_1 + \alpha_2$ and (s, h), (s', h') be such that $(s, h) \approx_\alpha^q (s', h')$. For all heaps h_1, h_2 such that $h = h_1 + h_2$ there are heaps h'_1, h'_2 such that

- $h' = h'_1 + h'_2$
- $(s, h_1) \approx_{\alpha_1}^q (s, h'_1)$
- $(s, h_2) \approx_{\alpha_2}^q (s, h'_2)$.

where $(s, h) \approx_\alpha^q (s', h')$ whenever (s, h) and (s', h') satisfy the same test formulae of $\mathrm{Test}(q, \alpha)$.

Test formulae capture SL($*$,$\mathtt{reach}^+$,$\mathtt{alloc}$)

Theorem

Let φ *be in* $\mathrm{SL}(*, \mathtt{reach}^+, \mathtt{alloc})$ *with variables* $\mathtt{x}_1, \ldots, \mathtt{x}_q$.

- *For all* $\alpha \geq |\varphi|$ *and all memory states* (s, h), (s', h') *such that* $(s, h) \approx_\alpha^q (s', h')$, *we have* $(s, h) \models \varphi$ *iff* $(s', h') \models \varphi$.
- φ *is logically equivalent to a Boolean combination of test formulae from* $\mathsf{Test}(q, |\varphi|)$.

Results

Theorem

Let φ be a satisfiable $\mathrm{SL}(*, \mathtt{reach}^+)$ formula built over $\mathtt{x}_1, \ldots, \mathtt{x}_q$. There is (s, h) such that $(s, h) \models \varphi$ and

$$\mathrm{card}(\mathrm{dom}(h)) \leq (q^2 + q) \cdot (|\varphi| + 1) + |\varphi|$$

- The satisfiability problem for SL($*$, $\mathtt{reach}^+$, $\mathtt{alloc}$) is PSPACE-complete.
- The satisfiability problem for SL($*$, $-\!\!*$, $\mathtt{reach}^+$) in which $\mathtt{reach}^+$ is not in the scope of $-\!\!*$ is in EXPSPACE.

Concluding Remarks

Main results:

- $\mathsf{SL}(*, \mathrel{-\!\!*}, \texttt{ls})$ admits an undecidable satisfiability problem, but
- if $\texttt{ls}$ is not in the scope of $\mathrel{-\!\!*}$ then the problem is decidable.

What's next? Satisfiability problem of fragments with $\texttt{ls}$ in the scope of $\mathrel{-\!\!*}$.

- Little to no result in the litterature.
- $\mathsf{SL}(\mathrel{-\!\!*}) + n(x) = n(y)$, $n(x) \hookrightarrow n(y)$ and $\texttt{alloc}^{-1}(x)$;
- $\mathsf{SL}(\mathrel{-\!\!*}, \texttt{ls})$ and $\mathsf{SL}(\mathrel{-\!\!*}, \texttt{reach})$;
- $\mathsf{SL}(*, \mathrel{-\!\!\circledast}, \texttt{ls})$ with negation only on atomic proposition.