

# Reasoning about memory states with automata and tableaux

Stéphane Demri

LSV, ENS Cachan, CNRS, INRIA Saclay

“Worskhop on Tableaux versus Automata  
as Logical Decision Procedures”

Oslo, July 6th, 2009

# Overview

- 1 Memory states
- 2 Separation Logic in a nutshell
- 3 A tableaux calculus for SL
- 4 Automata-based approach (for LTL)
- 5 Temporal Separation Logic  $LTL^{\text{mem}}$
- 6 Automata with Tableaux

# Separation Logic

# Pointer programs

- Pointer: reference to a memory cell (non fixed memory address).
- Dynamic memory allocation/deallocation. (mutable data structures)
- Examples of instructions:
  - $y \rightarrow l := x$ : write  $x$  to the  $l$ -field of the cell pointed to by  $y$ ,
  - `free x`: deallocate the cell pointer to by  $x$ ,
  - $x := \text{malloc}(i)$ : allocate  $i$  memory cells and assign its address to  $x$ .
- Specific properties for pointer programs:
  - No `null` dereference.
  - Memory leak: a memory cell can no longer be reached.
  - Shape analysis: checking the structure of the heap.

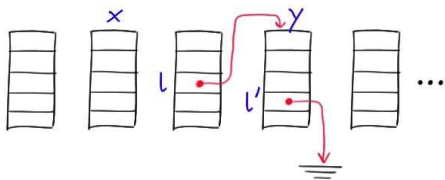
# Reasoning about pointer programs

- Examples of logical specification languages
  - Separation logic [Reynolds, LICS 02]
  - Pointer assertion logic (PAL) [Jensen et al. 97]
  - TVLA [Lev-Ami & Sagiv, SAS 00]: abstract interpretation technique with Kleene's logic (op. semantics in FOL + TC)
  - Evolution Logic [Yahav et al., ESOP 03]: to specify temporal properties of programs with dynamically evolving heaps.

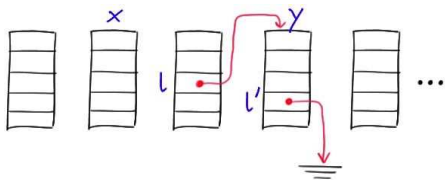
# Reasoning about pointer programs

- Examples of logical specification languages
  - Separation logic [Reynolds, LICS 02]
  - Pointer assertion logic (PAL) [Jensen et al. 97]
  - TVLA [Lev-Ami & Sagiv, SAS 00]: abstract interpretation technique with Kleene's logic (op. semantics in FOL + TC)
  - Evolution Logic [Yahav et al., ESOP 03]: to specify temporal properties of programs with dynamically evolving heaps.
- Model-checking
  - Navigation Temporal Logic [Distefano & Katoen & Rensink, FSTTCS 04]
  - Regular model-checking [Bouajjani et al., TACAS 05]
  - Translation into counter automata [Bouajjani et al, CAV 06; Sangnier, PhD 08]

# Memory states



# Memory states



- Set of variables  $\text{Var}$ .
- Set of selectors/labels  $\text{Sel}$ .
- Set of values  $\text{Val} = \mathbb{N} \uplus \{\text{nil}\}$ .
- Set of stores:  $\mathcal{S} \stackrel{\text{def}}{=} \text{Var} \rightarrow \text{Val}$ .
- Set of heaps:  
 $\mathcal{H} \stackrel{\text{def}}{=} \mathbb{N} \rightarrow_{\text{fin}} (\text{Sel} \rightarrow_{\text{fin}+} \text{Val})$ .
- $h \subseteq_f \mathbb{N} \times \text{Sel} \times \text{Val}$ .

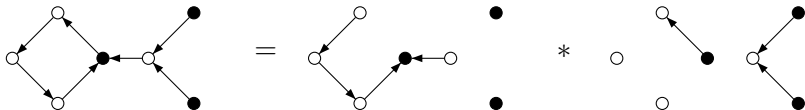
Memory state  $(s, h)$

## Disjoint heaps

- $h_1$  and  $h_2$  are disjoint whenever  $\text{dom}(h_1) \cap \text{dom}(h_2) = \emptyset$ .  
Notation:  $h_1 \perp h_2$ .
- Disjointness does not concern records.
- Disjoint union  $h_1 * h_2$  whenever  $h_1 \perp h_2$ .

# Disjoint heaps

- $h_1$  and  $h_2$  are disjoint whenever  $\text{dom}(h_1) \cap \text{dom}(h_2) = \emptyset$ .  
Notation:  $h_1 \perp h_2$ .
- Disjointness does not concern records.
- Disjoint union  $h_1 * h_2$  whenever  $h_1 \perp h_2$ .
- Disjoint heap graphs (with a unique selector):



## Long-term goals

- To design temporal languages to specify the behaviors of pointer programs.
- To combine an assertion language from separation logic with linear-time/branching-time temporal logics.
- To evaluate the borders for decidability.
- To admit effective procedure with “reasonable” computational complexity for precise analysis.

# Separation logic

- Introduced by Reynolds, Pym and O'Hearn.
- Reasoning about the heap with a strong form of locality built-in.
- $\mathcal{A} * \mathcal{B}$  is true whenever the heap can be divided into two disjoint parts, one satisfies  $\mathcal{A}$ , the other one  $\mathcal{B}$ .
- $\mathcal{A} - * \mathcal{B}$  is true whenever  $\mathcal{A}$  is true for a (fresh) disjoint heap,  $\mathcal{B}$  is true for the combined heap.

# Hoare triples

- Hoare triple:  $\{\mathcal{A}\} \text{ PROG } \{\mathcal{B}\}$  (total correctness).
- Rule of constancy:

$$\frac{\{\mathcal{A}\} \text{ PROG } \{\mathcal{B}\}}{\{\mathcal{A} \wedge \mathcal{B}'\} \text{ PROG } \{\mathcal{B} \wedge \mathcal{B}'\}}$$

where no variable free in  $\mathcal{B}'$  is modified by PROG.

- Unsoundness of the rule of constancy in separation logic:

$$\frac{\{(\exists z. x \mapsto z)\} [x] := 4 \{x \mapsto 4\}}{\{(\exists z. x \mapsto z) \wedge y \mapsto 3\} [x] := 4 \{x \mapsto 4 \wedge y \mapsto 3\}}$$

(when  $x = y$ )

## When separation logic enters into the play

- Reparation with frame rule:

$$\frac{\{A\} \text{ PROG } \{B\}}{\{A * B'\} \text{ PROG } \{B * B'\}}$$

where no variable free in  $B'$  is modified by PROG.

- Strengthening precedent (SP)

$$\frac{A \Rightarrow B' \quad \{B'\} \text{ PROG } \{B\}}{\{A\} \text{ PROG } \{B\}}$$

- Checking validity/satisfiability in separation logic is a building block of the verification process.

# Separation Logic (SL)

- Expressions

$$e ::= x \mid \text{null}$$

- Atomic formulae

$$\pi ::= e = e' \mid x + i \overset{!}{\hookrightarrow} e \mid \text{emp}$$

- Standard  $e \hookrightarrow e', e''$  can be encoded with  $e \overset{1}{\hookrightarrow} e' \wedge e \overset{2}{\hookrightarrow} e''$ .

- Convention:  $s(\text{null}) = \text{nil}$ .

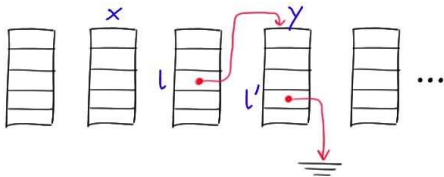
- State formulae

$$\mathcal{A} ::= \pi \mid \mathcal{A} \wedge \mathcal{B} \mid \neg \mathcal{A} \mid \mathcal{A} * \mathcal{B} \mid \mathcal{A} \text{-*} \mathcal{B}$$

# Semantics

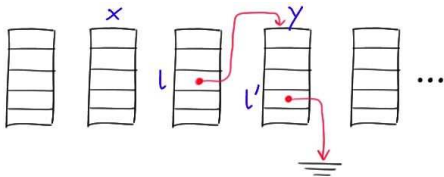
- $(s, h) \models_{\text{SL}} \text{emp}$  iff  $\text{dom}(h) = \emptyset$ .
- $(s, h) \models_{\text{SL}} e = e'$  iff  $s(e) = s(e')$ .
- $(s, h) \models_{\text{SL}} x + i \stackrel{l}{\hookrightarrow} e$  iff  $s(x) \in \mathbb{N}$  and  $s(x) + i \in \text{dom}(h)$  and  $h(s(x) + i)(l) = s(e)$ .
- $(s, h) \models_{\text{SL}} \mathcal{A}_1 * \mathcal{A}_2$  iff  $\exists h_1, h_2$  such that  $h = h_1 * h_2$ ,  
 $(s, h_1) \models_{\text{SL}} \mathcal{A}_1$  and  $(s, h_2) \models_{\text{SL}} \mathcal{A}_2$ .
- $(s, h) \models_{\text{SL}} \mathcal{A}_1 * \mathcal{A}_2$  iff for all  $h'$ , if  $h \perp h'$  and  $(s, h') \models_{\text{SL}} \mathcal{A}_1$   
then  $(s, h * h') \models_{\text{SL}} \mathcal{A}_2$ .

# Memory states with arithmetic and records



$x+1 \xrightarrow{l} y$   
 $y \xrightarrow{l'} \text{null}$

# Memory states with arithmetic and records



$$\begin{array}{l} x+1 \xrightarrow{l} y \quad h(s(x) + 1)(l) = s(y) \\ y \xrightarrow{l'} \text{null} \quad h(s(y))(l') = \text{nil} \end{array}$$

## Simple properties on memory states

- The memory heap has at least two cells ( $\text{size} \geq 2$ ):

$$\neg \text{emp} * \neg \text{emp}$$

- The memory heap has exactly one cell at address  $x$  with value  $e$  at selector  $l$ :

$$x \overset{l}{\hookrightarrow} e \wedge \neg(\text{size} \geq 2)$$

- The variable  $x$  is allocated in the heap ( $\text{alloc}(x)$ ):

$$(x \overset{l}{\hookrightarrow} \text{null})-*\perp$$

## On the complexity of SL

- SL with FO quantification is undecidable:
  - even without  $*$  and  $\neg$  [Calcagno & Yang & O'Hearn, FSTTCS 01].
  - even with a unique selector [Brochenin & Demri & Lozes, CSL 08]
- Model-checking, satisfiability and validity for SL are PSPACE-complete problems.
  - PSPACE-hardness from [CYOH, FSTTCS 01].
  - Upper bound without arithmetics can be also obtained by translation into a “separation-free” version [Lozes, PhD 04].

## Small store property

- Standard property:  $\mathcal{A}$  is satisfiable iff there is a store  $s$  such that  $(s, \emptyset) \models_{\text{SL}} \neg(\mathcal{A} * \perp)$ .
- Refinement:  $\mathcal{A}$  is satisfiable iff there is a store  $s$  such that
  - $(s, \emptyset) \models_{\text{SL}} \neg(\mathcal{A} * \perp)$ ,
  - for each variable  $x \in Y$ ,  
 $s(x) \leq (\text{card}(Y) + 1) \times \max(\text{offsets})$ ,

where

- $Y$  is the set of variables occurring in  $\mathcal{A}$ ,
- $\text{offsets}$  is the set of indices  $i$  such that  $x + i$  occurs in  $\mathcal{A}$  for some variable  $x$ .

# **A tableaux calculus for Separation Logic**

[Galmiche & Méry, JLC 08]

## A fragment of propositional SL

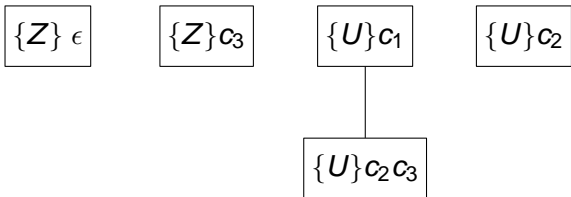
- $\mathcal{A} ::= loc \mapsto a, b \mid \text{emp} \mid \mathcal{A} * \mathcal{A} \mid \mathcal{A} \multimap \mathcal{A} \mid \mathcal{A} \wedge \mathcal{A} \mid \neg \mathcal{A}$
- $loc \mapsto a, b \equiv loc \xrightarrow{1} a \wedge loc \xrightarrow{2} b \wedge \neg(\neg \text{emp} * \neg \text{emp})$ .
- Labelled tableau calculus for SL using “resource graphs”  
[Galmiche & Méry, JLC 08].

# Labelling algebra

- Label:  $x = c_{i_1}, \dots, c_{i_n}$  with  $i_1 < \dots < i_n$ .
- Each  $c_j$  is interpreted as a heap.
- Composition  $x \circ y$ .  
 $c_1 c_4 \circ c_2 c_3 = c_1 c_2 c_3 c_4$  whereas  $c_1 c_4 \circ c_1 c_3$  undefined.
- Labelling algebra  $\mathcal{L}_A = (\mathcal{L}, \circ, \epsilon)$ .
- Constraints  $C$ :
  - $Tx$  with  $T \in \{\emptyset, \{Z\}, \{U\}, \{Z, U\}\}$  (size constraint).
  - $x \diamond y$  (equality).

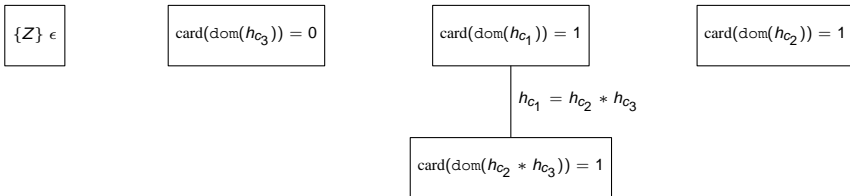
# Resource graphs

- Introduction of  $\vdash$  with  $X \vdash C$  decidable in P.
- $\{\{Z\}c_1, c_2c_3 \diamond c_1\} \vdash \{Z\}c_2$ .
- A resource graph is a graphical representation of constraints derivable from a set of constraints.



# Resource graphs

- Introduction of  $\vdash$  with  $X \vdash C$  decidable in P.
- $\{\{Z\}c_1, c_2c_3 \diamond c_1\} \vdash \{Z\}c_2$ .
- A resource graph is a graphical representation of constraints derivable from a set of constraints.



# Rules

- Labelled formula  $\mathbf{S}\mathcal{A} : x$  with  $\mathbf{S} \in \{\mathbf{T}, \mathbf{F}\}$  and  $x \in \mathcal{L}$ .

- Boolean rules: 
$$\frac{\mathbf{T}(\mathcal{A} \wedge \mathcal{A}') : x}{\mathbf{T}\mathcal{A} : x \quad \mathbf{T}\mathcal{A}' : x} \quad \frac{\mathbf{T}(\mathcal{A} \vee \mathcal{A}') : x}{\mathbf{T}\mathcal{A} : x \mid \mathbf{T}\mathcal{A}' : x}$$

- Atomic rules: 
$$\frac{\mathbf{T}(l \mapsto a, b) : x}{\{U\}x} \quad \frac{\mathbf{T}_{\text{emp}} : x}{\{Z\}x}$$

- Separation rules: 
$$\frac{\mathbf{T}(\mathcal{A} * \mathcal{A}') : x}{\mathbf{T}\mathcal{A} : c_i \quad \mathbf{T}\mathcal{A}' : c_j \quad (c_i \circ c_j) \diamond x} \quad \frac{\mathbf{F}(\mathcal{A} * \mathcal{A}') : x}{\mathbf{F}\mathcal{A} : y \mid \mathbf{F}\mathcal{A}' : z}$$
  - $c_i, c_j$  are fresh,
  - $(y \circ z) \diamond x$  derivable from constraints on the branch i.e.,  $\text{Branch} \vdash (y \circ z) \diamond x$

# Consistency

- The resource graph induced by a branch is structurally consistent when it can represent a memory state.
- Concepts of measure  $\mu$  and interpretation  $\lambda$  are considered to characterize structural consistency,  
See details in [Galmiche & Méry, JLC 08].
- $\mu$  determines the sizes and  $\lambda$  the memory cells.
- Logical consistency: none of the situations below.
  - $\mathbf{T}(I \mapsto a, b) : x, \mathbf{F}(I \mapsto a, b) : y$  and  $\text{Branch} \vdash x \sim_{\mu, \lambda} y$ .
  - $\mathbf{F}_{\text{emp}} : x$  and  $\mu(x) = 0$ .

## Completeness and termination

- A branch is closed  $\stackrel{\text{def}}{\iff}$  there are no measure  $\mu$  and interpretation  $\lambda$  such that its induced resource graph is both structurally and logically consistent with respect to  $\mu, \lambda$ .
- A tableau is closed  $\stackrel{\text{def}}{\iff}$  all the branches are closed.
- A tableau for  $\mathcal{A}$  has root  $\mathbf{FT} \rightarrow^* \mathcal{A} : \epsilon$ .

## Completeness and termination

- A branch is closed  $\stackrel{\text{def}}{\iff}$  there are no measure  $\mu$  and interpretation  $\lambda$  such that its induced resource graph is both structurally and logically consistent with respect to  $\mu, \lambda$ .
- A tableau is closed  $\stackrel{\text{def}}{\iff}$  all the branches are closed.
- A tableau for  $\mathcal{A}$  has root  $\mathbf{FT} \rightarrow * \mathcal{A} : \epsilon$ .
- **Theorem:** [Galmiche & Méry, JLC 08]  
 $\mathcal{A}$  is valid iff  $\mathcal{A}$  has a closed tableau.
- Termination is shown by using  $\lambda\mu$ -equivalences.

## Other results in [Galmiche & Méry, JLC 08]

- Complete labelled tableau calculus for first-order SL.
- Addition of equality.
- Decidable FO fragments by proof-theoretical analysis.

## Other results in [Galmiche & Méry, JLC 08]

- Complete labelled tableau calculus for first-order SL.
- Addition of equality.
- Decidable FO fragments by proof-theoretical analysis.

Automated-based decision procedure for SL ?

## **Automata-based approach for logical decision problems**

# From logic to automata

- Automata-based approach consists in reducing logical problems into automata-based decision problems.
- Examples of target problems:
  - $L(A) = \emptyset$  ?
  - $L(A) \subseteq L(B)$  ?
  - Is  $L(A)$  the universal language ?
- Pioneering work by Büchi [Büchi, 62].
  - MSO over  $(\mathbb{N}, <)$ .
  - Models of a formula over  $P_1, \dots, P_N$  are  $\omega$ -sequences over the alphabet  $\mathcal{P}(\{P_1, \dots, P_N\})$ .
  - Büchi automata are equivalent to MSO formulae.

## Desirable properties

- Reduction is **simple**.  
ex: LTL formula  $\mapsto$  alternating automaton
- **Complexity** of the automata-based target problem is **well-characterized**.  
ex: PDL formula  $\mapsto$  nondeterministic Büchi tree automaton.
- The reduction is **semantically faithful**.  
ex: First-order model-checking for automatic structures.
- Reduction allows to obtain the **optimal** complexity of the **source** logical problem.  
ex: CTL model-checking is in P by reduction into hesitant alternating automata (HAA).

# Temporal logics

- Automated-based approach for LTL-like logics.  
See e.g. [Vardi & Wolper, IC 94]
- Alternating automata for temporal logics.  
See e.g. [Muller & Saoudi & Schupp, LICS'88]
- Branching-time model-checking.  
[Kupferman & Vardi & Wolper, JACM 00]
- See also the works for description logics, modal logics, etc.

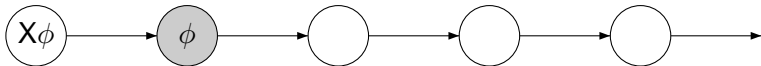
## Current trends

- Algorithmic automata theory
  - Design of efficient decision procedures for testing nonemptiness, complementing etc.
  - Comparable to the design of efficient strategies in tableaux-based calculi.
- New source problems.  
ex: axiom pinpointing [Baader & Peñaloza, IJCAR 08].
- Deciding Presburger arithmetic with automata (see the works of Boigelot, Leroux, Wolper etc.).
- Reasoning about heap manipulating programs:
  - Shape analysis using regular model-checking [Bouajjani et al., TACAS 05]
  - Counter automaton abstraction [Bouajjani et al., CAV 06]

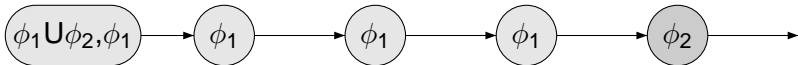
## **Automata-based approach for LTL**

# LTL operators in a nutshell

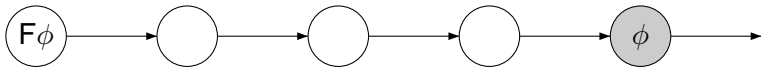
$X\phi$ : next-time  $\phi$



$\phi_1 U \phi_2$ :  $\phi_1$  until  $\phi_2$



$F\phi$ : sometimes  $\phi$

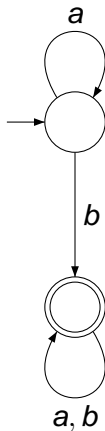


# About plain LTL

- Formulae:  $\phi ::= p \mid \neg\phi \mid \phi \wedge \psi \mid \phi \mathbf{U} \psi \mid \mathbf{X}\phi$ .
- Models:  $\sigma: \mathbb{N} \rightarrow \mathcal{P}(\text{PROP})$  and  $\sigma, i \models p$  iff  $p \in \sigma(i)$ .
- $\text{Models}(\phi) = \{\sigma \in (\mathcal{P}(\text{PROP}))^\omega : \sigma, 0 \models \phi\}$ .
- Model-checking and satisfiability are PSPACE-complete.  
[Sistla & Clarke, JACM 85].

# Basics on Büchi automata

# Büchi automaton for $a^* \cdot b \cdot \{a, b\}^\omega$



# Generalized Büchi automaton

- A generalized Büchi automaton is a structure

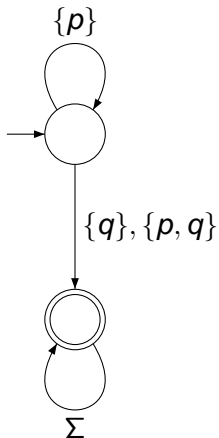
$$A = (\Sigma, S, S_0, \rho, \{F_1, \dots, F_k\})$$

such that  $F_1, \dots, F_k \subseteq S$  and accepting condition:  
 $\text{inf}(c) \cap F_i \neq \emptyset$  for every  $F_i$ .

- $\text{GBA} \mapsto \text{BA}$  in LOGSPACE.
- Nonemptiness problem is NLOGSPACE-complete  
[Vardi & Wolper, IC 94].
- By contrast, the universality problem for Büchi automata is PSPACE-complete.

## **From LTL formulae to automata**

## Automaton for $p \cup q$ models



$$\Sigma = \{\{\}, \{p\}, \{q\}, \{p, q\}\}$$

# Optimal automata-based approach

[Vardi & Wolper, IC 94] For every LTL formula  $\phi$ , there is a Büchi automaton  $A_\phi$  such that

- $L(A_\phi) = \text{Models}(\phi)$ ,
- $|A_\phi|$  is in  $2^{\mathcal{O}(|\phi|)}$  and,
- $A_\phi$  can be effectively computed in polynomial space in  $|\phi|$ .

# Closure set

- $cl(\phi)$ : set of formulae “useful” to verify the satisfiability status of  $\phi$ .
- $cl(\phi)$  is the smallest set
  - containing the subformulae of  $\phi$ ,
  - closed under negation (we identify  $\neg\neg\psi$  with  $\psi$ ),
  - if  $\varphi_1 \cup \varphi_2 \in cl(\phi)$ , then  $X(\varphi_1 \cup \varphi_2) \in cl(\phi)$ .
- The cardinal of  $cl(\phi)$  is linear in the size of  $\phi$ .
- Example:  $cl(p_1 \cup Xp_2) = \{p_1, p_2, Xp_2, p_1 \cup Xp_2, X(p_1 \cup Xp_2)\}$ .

# Hintikka sequence

An Hintikka sequence for  $\phi$ :  $\alpha_0\alpha_1 \dots \in \mathcal{P}(cl(\phi))^\omega$  s.t.

**(HIN1)**  $\phi \in \alpha_0$ ,

**(HIN2)** for every  $\psi \wedge \psi' \in cl(\phi)$ ,  $\psi \wedge \psi' \in \alpha_i$  iff  $\psi \in \alpha_i$  and  $\psi' \in \alpha_i$ ,

**(HIN3)** for every  $\neg\psi \in cl(\phi)$ ,  $\neg\psi \in \alpha_i$  iff  $\psi \notin \alpha_i$ ,

**(HIN4)** for every  $X\psi \in cl(\phi)$ ,  $X\psi \in \alpha_i$  iff  $\psi \in \alpha_{i+1}$ ,

**(HIN5)** for every  $\psi \mathbf{U} \psi' \in cl(\phi)$ ,  $\psi \mathbf{U} \psi' \in \alpha_i$  iff either  $\psi' \in \alpha_i$  or  $(\psi \in \alpha_i$  and  $X(\psi \mathbf{U} \psi') \in \alpha_i)$ ,

**(HIN6)** for every  $\psi \mathbf{U} \psi' \in cl(\phi)$ , if  $\psi \mathbf{U} \psi' \in \alpha_i$ , then there is  $j \geq i$  such that  $\psi' \in \alpha_j$ .

$\phi$  is satisfiable iff  $\phi$  has at least one Hintikka sequence.

# Maximally consistent sets

A set  $X \subseteq cl(\phi)$  is maximally consistent iff:

- for every  $\psi \in cl(\phi)$ , either  $\psi \in X$  or  $\neg\psi \in X$ ,
- for every  $\neg\psi \in cl(\phi)$ ,  $\neg\psi \in X$  iff  $\psi \notin X$ ,
- for every  $\phi_1 \wedge \phi_2 \in cl(\phi)$ ,  $\phi_1 \wedge \phi_2 \in X$  iff  $\{\phi_1, \phi_2\} \subseteq X$ ,
- for every  $\phi_1 \mathbf{U} \phi_2 \in cl(\phi)$ ,  $\phi_1 \mathbf{U} \phi_2 \in X$  iff either  $\phi_2 \in X$  or  $\{\phi_1, X(\phi_1 \mathbf{U} \phi_2)\} \subseteq X$ .

$$A_\phi = (\Sigma, \mathbf{S}, \mathbf{S}_0, \rho, F_1, \dots, F_k)$$

- $\Sigma = \mathbf{S}$  is the set of maximally consistent sets wrt  $\phi$ ,
- $\mathbf{S}_0 = \{X \in \mathbf{S} : \phi \in X\}$ ,
- $Y \in \rho(X, \mathbf{a})$  iff
  - $X = \mathbf{a}$ ,
  - for  $X\psi \in cl(\phi)$ ,  $X\psi \in X$  iff  $\psi \in Y$ ,
- If  $\psi_1 \mathbf{U} \psi'_1, \dots, \psi_k \mathbf{U} \psi'_k$  occurs in  $\phi$ , then

$$F_i \stackrel{\text{def}}{=} \{X \in \mathbf{S} : \text{either } \psi_i \mathbf{U} \psi'_i \notin X \text{ or } \psi'_i \in X\}$$

- If  $\mathbf{U}$  does not occur in  $\phi$ , then  $k = 1$  and  $F_1 = \mathbf{S}$ .

## $A_\phi$ recognizes Hintikka sequences for $\phi$

- $L(A_\phi)$  is the set of Hintikka sequences for  $\phi$ .
- $\phi$  is satisfiable iff  $L(A_\phi) \neq \emptyset$ .
- From a generalized BA it is possible to build an equivalent standard BA in LOGSPACE.
- Using other complexity considerations (such as Savitch Theorem), one can test whether  $\phi$  is LTL satisfiable in polynomial space in  $|\phi|$ .

# Temporal separation logic $LTL^{mem}$

[Brochenin & Demri & Lozes, APAL 09]

# The logic $LTL^{mem}$

- Syntax

$e ::= X^u x \mid \text{null}$  (expressions)

$\pi ::= e = e' \mid X^u x + i \overset{!}{\hookrightarrow} e$  (atomic formulae)

$\mathcal{A} ::= \pi \mid \mathcal{A} \wedge \mathcal{B} \mid \neg \mathcal{A}$  (classical fragment)

$\mid \mathcal{A} * \mathcal{B} \mid \mathcal{A} \text{-} * \mathcal{B} \mid \text{emp}$  (spatial fragment)

$\phi ::= \mathcal{A} \mid X\phi \mid \phi \mathbf{U} \phi' \mid \phi \wedge \phi' \mid \neg \phi$  (temporal formulae)

- “ $X^u x$ ”: value of  $x$  at the  $u$ th next position.

- Examples

$G(\text{alloc}(x) \Rightarrow F \text{alloc}(y))$

$GF(\text{size} \geq 2) \quad (Xx = x) \mathbf{U} (y \overset{!}{\hookrightarrow} z)$

# Semantics

Models: elements of  $(\mathcal{S} \times \mathcal{H})^\omega$  of the form  $\rho = (s_i, h_i)_{i \geq 0}$ .

$$\rho, t \models e = e' \quad \text{iff } \llbracket e \rrbracket_{\rho, t} = \llbracket e' \rrbracket_{\rho, t} \quad \text{with } \llbracket Xe \rrbracket_{\rho, t} = \llbracket e \rrbracket_{\rho, t}, \\ \llbracket x \rrbracket_{\rho, t} = s_t(x) \text{ and } \llbracket \text{null} \rrbracket_{\rho, t} = \text{nil}.$$

$$\rho, t \models X^u x + i \overset{!}{\hookrightarrow} e' \quad \text{iff } h_t(\llbracket X^u x \rrbracket_{\rho, t} + i) = \llbracket e' \rrbracket_{\rho, t}$$

$$\rho, t \models \mathcal{A}_1 * \mathcal{A}_2 \quad \text{iff } \exists h_1, h_2 \text{ s.t. } h_t = h_1 * h_2, \\ \rho[h_t \leftarrow h_1], t \models \mathcal{A}_1, \\ \text{and } \rho[h_t \leftarrow h_2], t \models \mathcal{A}_2.$$

$$\rho, t \models \mathcal{A}_1 * \mathcal{A}_2 \quad \text{iff } \forall h', \text{ if } h_t \perp h' \text{ and } \rho[h_t \leftarrow h'], t \models \mathcal{A}_1 \\ \text{then } \rho[h_t \leftarrow h * h'], t \models \mathcal{A}_2.$$

$$\rho, t \models \phi \mathbf{U} \phi' \quad \text{iff } \exists t' \geq t \text{ such that } \rho, t' \models \phi', \\ \text{and } \forall t'', t \leq t'' < t', \rho, t'' \models \phi.$$

# Satisfiability problems

- Satisfiability problem  $\text{SAT}(\text{Frag})$  with underlying fragment  $\text{Frag} \subseteq \text{SL}$ .
- Problem  $\text{SAT}^{ct}(\text{Frag})$  with constant heap  
→ temporal language allows us to explore the heap.
- Problem  $\text{SAT}_{init}(\text{Frag})$  with a fixed initial heap.
- Model-checking problems have been also considered in  
[BDL, APAL 09].

# Fragments with decidable temporal reasoning

Classical fragment (CL)

$$\mathcal{A} ::= e = e' \mid x + i \xrightarrow{l} e \\ \mid \mathcal{A} \wedge \mathcal{A} \mid \neg \mathcal{A}$$

Record fragment (RF)

$$\mathcal{A} ::= e = e' \mid x \xrightarrow{l} e \\ \mid \mathcal{A} * \mathcal{A} \mid \mathcal{A} \rightarrow * \mathcal{A} \mid \text{emp} \\ \mid \mathcal{A} \wedge \mathcal{A} \mid \neg \mathcal{A}$$

**Theorem:** [BDL, APAL 09] The satisfiability problems for  $\text{LTL}^{\text{mem}}(\text{CL})$  and  $\text{LTL}^{\text{mem}}(\text{RF})$  are PSPACE-complete.

## Deciding $LTL^{\text{mem}}$ with tableaux and automata

# Bounding the syntactic resources

- Test formulae

$$\begin{aligned} e ::= x \mid \text{null} & & f ::= x + i \\ \psi ::= f \stackrel{l}{\hookrightarrow} e \mid \text{alloc}(f) \mid e = e' \mid \text{size} \geq k \end{aligned}$$

- $i, k \in \mathbb{N}$ ; variable  $x$ ; label  $l$ .
- **Measure**  $\mu = (\text{offsets}, w, L, V)$  restricts the test formulae.
- $i \in \text{offsets}, k \in [0, w[, l \in L, x \in V$ .
- $\mathcal{T}_\mu$  : set of test formulae restricted to the resources from the measure.
- Every formula is equivalent to a Boolean combination of test formulae.

# Symbolic models and abstraction

- $\phi \mapsto \mu$  with  $(X^u \mathbf{x} \in \phi \text{ implies } \langle \mathbf{x}, u \rangle \in V_\mu)$ .
- Symbolic model:  $\sigma : \mathbb{N} \rightarrow \mathcal{P}(\mathcal{T}_\mu)$ .
- Abstraction:  $\rho \in (\mathcal{S} \times \mathcal{H})^\omega \mapsto \mathit{Abs}_\mu(\rho) \in \mathcal{P}(\mathcal{T}_\mu)^\omega$ .

$$\mathit{Abs}_\mu(\rho)(i) \stackrel{\text{def}}{=} \{\mathcal{A} \in \mathcal{T}_\mu : \rho, i \models \mathcal{A}\}.$$

- Symbolic satisfaction relation:  $\sigma, i \models_\mu \phi$  defined by induction on  $\phi$  with the base case:  $\sigma, i \models_\mu \mathcal{A} \stackrel{\text{def}}{\Leftrightarrow}$

$$\models_{\text{SL}} \mathit{TF}[\sigma(i)] \Rightarrow \mathcal{A}[X^u \mathbf{x} \leftarrow \langle \mathbf{x}, u \rangle]$$

where

$$\mathit{TF}[\sigma(i)] = \left( \bigwedge_{\mathcal{B} \in \sigma(i)} \mathcal{B} \wedge \bigwedge_{\mathcal{B} \in (\mathcal{T}_\mu \setminus \sigma(i))} \neg \mathcal{B} \right)$$

# Checking satisfiability with symbolic models

- **Lemma:**  $\phi$  in  $LTL^{\text{mem}}(\text{RF})$  is satisfiable iff there is a symbolic model  $\sigma : \mathbb{N} \rightarrow \mathcal{P}(\mathcal{T}_{\mu_\phi})$  such that
  - $\sigma$  symbolically satisfies  $\phi$  ( $\sigma, 0 \models_{\mu_\phi} \phi$ )
  - there is a model  $\rho$  of  $LTL^{\text{mem}}$  such that  $\text{Abs}_{\mu_\phi}(\rho) = \sigma$ .
- For instance,  $\{Xx = X^2x, \dots\}, \{x \neq Xx, \dots\} \dots$  has no concrete models.

# The generalized Büchi automaton $A_{\phi}^{\mu}$

- $Q$  is the set of atoms of  $\phi$  (sets of subformulae).
- $I = \{X \in Q : \phi \in X\}$ .
- $\Sigma = \mathcal{P}(\mathcal{I}_{\mu})$ .
- $X \xrightarrow{a} Y$  iff
  - for every atomic formula  $\mathcal{A}$  of  $X$ ,  
 $\models_{\text{SL}} TF[a] \Rightarrow \mathcal{A}[X^u_{\mathbf{x}} \leftarrow \langle \mathbf{x}, u \rangle]$ .
  - for every  $X\phi' \in cl(\phi)$ ,  $X\phi' \in X$  iff  $\phi' \in Y$ .
- Let  $\{\phi_1 \mathbf{U} \phi'_1, \dots, \phi_n \mathbf{U} \phi'_n\}$  be the set of until formulae in  $cl(\phi)$ . We pose  $\mathcal{F} = \{F_1, \dots, F_n\}$  where  $F_i = \{X \in Q : \phi_i \mathbf{U} \phi'_i \notin X \text{ or } \phi'_i \in X\}$  for  $i \in \{1, \dots, n\}$ .
- **Lemma:** Let  $\phi$  in  $\text{LTL}^{\text{mem}}(\text{RF})$  and  $\mu \geq \mu_{\phi}$ . Then,  $L(A_{\phi}^{\mu})$  is the set of symbolic models satisfying  $\phi$ .

# The automaton $A_{sat}^\mu$ for consistency

- $\Sigma = \mathcal{P}(\mathcal{T}_\mu)$ ,  $Q = I = F = \Sigma$ ,
- $a \xrightarrow{a'} a''$  iff:
  - $TF[a]$ ,  $TF[a'']$  are satisfiable, and  $a = a'$ ,
  - for  $\langle x, u \rangle = \langle x', u' \rangle \in \mathcal{T}_\mu$  with  $u, u' \geq 1$ ,  
 $\langle x, u \rangle \in a$  iff  $\langle x', u' - 1 \rangle \in a''$ .
  - for  $\langle x, u \rangle = \text{null} \in \mathcal{T}_\mu$  with  $u \geq 1$ ,  
 $\langle x, u \rangle \in a$  iff  $\langle x, u - 1 \rangle \in a''$ .
- **Lemma:** Let  $\phi$  in  $LTL^{\text{mem}}(\text{RF})$  and  $\mu = \mu_\phi$ . Then  $L(A_{sat}^\mu)$  is the set of symbolic models being the abstraction of some concrete model.

# When tableaux enter into the play

- Reasoning tasks:
  - Validity checking in  $A_{\phi}^{\mu}$ : “for every atomic formula  $\mathcal{A}$  of  $X$ ,  $\models_{\text{SL}} TF[a] \Rightarrow \mathcal{A}[X^u_{\mathbf{x}} \leftarrow \langle \mathbf{x}, u \rangle]$ ”.
  - Satisfiability checking in  $A_{\text{sat}}^{\mu}$ : “ $TF[a], TF[a']$  are satisfiable”.
- Decision procedures:
  - Tableaux calculus with termination  
[Galmiche & Méry, JLC 08].
  - Small store property and model-checking.

## Other satisfiability problems

- $SAT_{init}^{ct}(\text{Frag})$ : satisfiability problem of the fragment Frag with fixed initial memory state and constant heap models.
- $SAT_{init}^{ct}(\text{RF})$  and  $SAT_{init}^{ct}(\text{SL} \setminus \neg*)$  are PSPACE-complete.
- The PSPACE upper bound is preserved with extensions of LTL.
- $SAT(\text{SL} \setminus \neg*)$  are  $\Sigma_1^1$ -complete.

# Conclusion

- Introduction of a logic mixing temporal operators and assertions from separation logic.
- Automata and tableaux can work together.

# Conclusion

- Introduction of a logic mixing temporal operators and assertions from separation logic.
- Automata and tableaux can work together.
- A selection of perspectives:
  - New decision procedures for fragments of first-order SL  
See e.g., [Galmiche & Méry, JLC 08]
  - Tableaux calculus for  $SL(*)$  with one selector?  
(non-elementary complexity).
  - Which classes of constraints on successive heaps restore decidability?