A glimpse at the μ-calculus

Precise Modeling and Analysis group
University of Oslo
Daniel Fava

March 12, 2019

Roadmap

1. Start with LTL and motivate greater expressivity
2. Give some background: Hennessy Milner Logic (HML)
3. Build a modest foundation for understanding fixed points
4. μ-calculus syntax, semantics, and examples
5. Game theoretic approach to model checking the μ-calculus
6. Bisimulation

Motivation

What do these mean?

$$\Box p$$

$$\Diamond p$$

$$p \mathcal{U} q$$

$$p \mathcal{R} q$$

Motivation

What do these mean?

$$
\begin{aligned}
\Box p &= p \wedge \bigcirc \Box p \\
\Diamond p &= p \vee \bigcirc \Diamond p \\
p \mathcal{U} q &= q \vee \big(p \wedge \bigcirc (p \mathcal{U} q)\big) \\
p \mathcal{R} q &= (p \wedge q) \vee \big(q \wedge \bigcirc (p \mathcal{R} q)\big)
\end{aligned}
$$

Motivation

What do these mean? Notice the recursion

$$\Box p = p \wedge \bigcirc \Box p$$

$$\Diamond p = p \vee \bigcirc \Diamond p$$

$$p \mathcal{U} q = q \vee \big(p \wedge \bigcirc (p \mathcal{U} q)\big)$$

$$p \mathcal{R} q = (p \wedge q) \vee \big(q \wedge \bigcirc (p \mathcal{R} q)\big)$$

Think of $\Box$, $\Diamond$, $\mathcal{U}$, $\mathcal{R}$ as special purpose recursive operators

- What if we could have more powerful (arbitrary) recursions?

Motivation

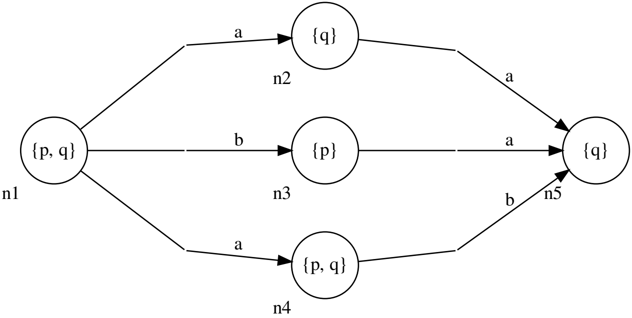
LTL: a trace σ or sets of traces

$$[\![\alpha]\!]^{\sigma} = \{T, F\}$$

μ-calculus: Labeled Transition System (LTS) $\mathcal{M} = (S, \xrightarrow{l}, P_i)$

$$[\![\alpha]\!]^{\mathcal{M}} \subseteq S$$

1. Talk about a node's direct children
2. Talk about a node's descendants



Motivation

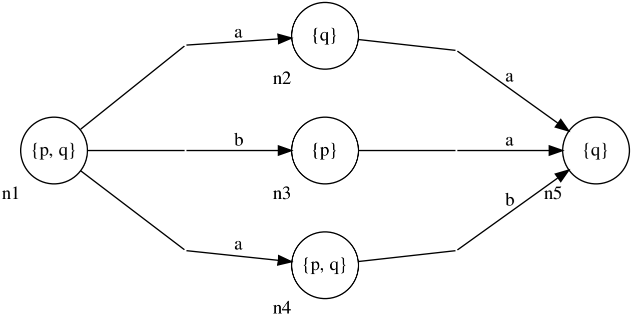
LTL: a trace σ or sets of traces

$$[\![\alpha]\!]^{\sigma} = \{T, F\}$$

μ-calculus: Labeled Transition System (LTS) $\mathcal{M} = (S, \xrightarrow{l}, P_i)$

$$[\![\alpha]\!]^{\mathcal{M}} \subseteq S$$

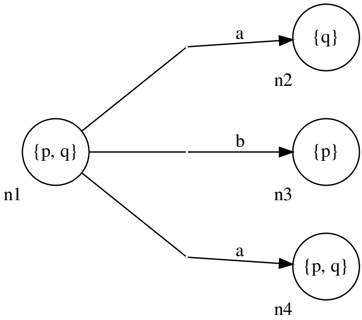
1. Talk about a node's direct children $\Longleftarrow$ Hennessy Milner Logic
2. Talk about a node's descendants $\Longleftarrow$ Fixed points



Background: Hennessy Milner Logic (1/3)

- Syntax $\Phi ::= tt \mid ff \mid p_i \mid \neg p_i \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid [a]\Phi \mid \langle a \rangle \Phi$
- Semantics

$$\llbracket tt \rrbracket^{\mathcal{M}} = S \qquad \llbracket ff \rrbracket^{\mathcal{M}} = \emptyset$$
$$\llbracket p_i \rrbracket^{\mathcal{M}} = P_i \qquad \llbracket \neg p_i \rrbracket^{\mathcal{M}} = S - P_i$$



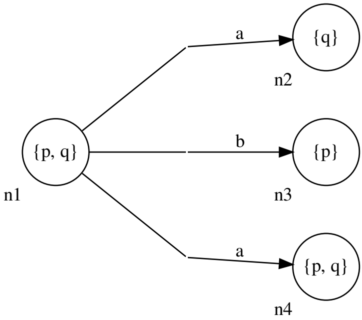
Examples:

1. $\llbracket tt \rrbracket^{\mathcal{M}} = \{n_1, n_2, n_3, n_4, n_5\}$
2. $\llbracket p \rrbracket^{\mathcal{M}} = \{n_1, n_3, n_4\}$

Background: Hennessy Milner Logic (2/3)

- Syntax $\Phi ::= tt \mid ff \mid p_i \mid \neg p_i \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid [a]\Phi \mid \langle a \rangle \Phi$
- Semantics

$$[\![\alpha \vee \beta]\!]^{\mathcal{M}} = [\![\alpha]\!]^{\mathcal{M}} \cup [\![\beta]\!]^{\mathcal{M}}$$
$$[\![\alpha \wedge \beta]\!]^{\mathcal{M}} = [\![\alpha]\!]^{\mathcal{M}} \cap [\![\beta]\!]^{\mathcal{M}}$$



Example:

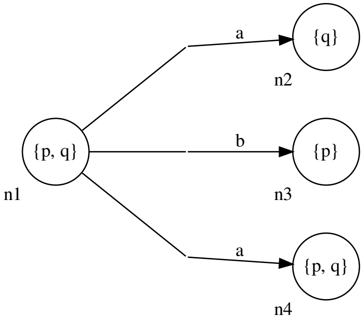
$$[\![p \wedge q]\!]^{\mathcal{M}} = \{n_1, n_4\}$$

Background: Hennessy Milner Logic (3/3)

- Syntax $\Phi ::= tt \mid ff \mid p_i \mid \neg p_i \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid [a]\Phi \mid \langle a \rangle \Phi$
- Semantics

  $[a]$ All children accessible via an a-transition
  $[\![[a]\alpha]\!]^{\mathcal{M}} = \{ s \in S \mid \forall t.\ \ s \xrightarrow{a} t \quad \rightarrow t \in [\![\alpha]\!]^{\mathcal{M}} \}$

  $\langle a \rangle$ At least one child accessible via an
  $[\![\langle a \rangle \alpha]\!]^{\mathcal{M}} = \{ s \in S \mid \exists t.\ \ s \xrightarrow{a} t \quad \wedge \quad t \in [\![\alpha]\!]^{\mathcal{M}} \}$



Examples:

1. $n_1 \in [\![[a]q]\!]^{\mathcal{M}}$
2. $n_1 \notin [\![[a]p]\!]^{\mathcal{M}}$
3. $n_1 \in [\![\langle a \rangle p]\!]^{\mathcal{M}}$

Background: Fixed-points (1/3)

- Fixed point
- Monotonic function
- Partial order relation $\sqsubseteq$
- Upper bound
- Least Upper Bound (lub) $\quad \bigsqcup$
- Lower bound
- Greatest Lower Bound (glb) $\quad \bigsqcap$
- Complete lattice
- Boundedness of complete lattices

Tarski-Knaster theorem

- A monotonic function $f : L \rightarrow L$ on a complete lattice L has a *greatest fixed point* (gfp) and a *least fixed point* (lfp).

Background: Fixed-points (1/3)

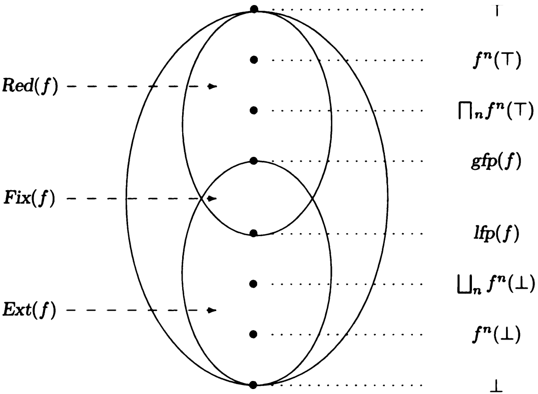
- Fixed point $f(x) = x^2 + x - 4$
- Monotonic function $x \leqslant x' \rightarrow f(x) \leqslant f(x')$
- Partial order relation $\sqsubseteq$
- Upper bound $Y \subseteq S$, $u \in S$, if $\forall s \in S.\ s \sqsubseteq u$
- Least Upper Bound (lub) $\bigsqcup$
- Lower bound $Y \subseteq S$, $l \in S$, if $\forall s \in S.\ l \sqsubseteq s$
- Greatest Lower Bound (glb) $\bigsqcap$
- Complete lattice $(S, \sqsubseteq, \bigsqcup, \bigsqcap)$
- Boundedness of complete lattices $\bigsqcup \varnothing = \bot, \quad \bigsqcap \varnothing = \top$

Tarski-Knaster theorem

- A monotonic function $f : L \rightarrow L$ on a complete lattice L has a *greatest fixed point* (gfp) and a *least fixed point* (lfp).

Background: Fixed-points (2/3)

- Reductive $f(x) \sqsubseteq x$
- Extensive $x \sqsubseteq f(x)$



Tarski-Knaster theorem

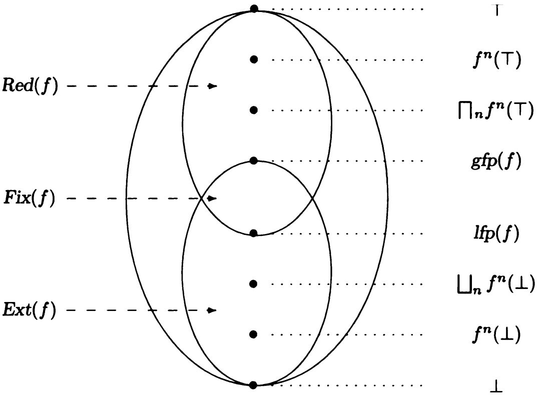
- A monotonic function $f : L \to L$ on a complete lattice L has a *greatest fixed point* (gfp) and a *least fixed point* (lfp).

$$\mathrm{gfp}(f) = \bigsqcup\{x \in L \mid x \sqsubseteq f(x)\} \;=\; \bigsqcup\{Ext(f)\} \in \mathrm{Fix}(f)$$

$$\mathrm{lfp}(f) = \bigsqcap\{x \in L \mid f(x) \sqsubseteq x\} \;=\; \bigsqcap\{Red(f)\} \in \mathrm{Fix}(f)$$

Background: Fixed-points (3/3)

- Reductive $f(x) \sqsubseteq x$
- Extensive $x \sqsubseteq f(x)$



Kleene fixed-point theorem

$$\mathrm{gfp} = f^{\infty}(\top) = \bigsqcap_{n \geqslant 0} f^n(\top) \qquad \mathrm{lfp} = f^{\infty}(\bot) = \bigsqcup_{n \geqslant 0} f^n(\bot)$$

μ-calculus (1/2)

- Extends HML by adding variables $X, Y, Z, \ldots$
- Syntax
  - Add variables and fixed point operators on top of HML

$$\Phi ::= tt \mid ff \mid p_i \mid \neg p_i \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid [a]\Phi \mid \langle a\rangle\Phi \mid$$
$$X \mid \mu X.\Phi \mid \nu X.\Phi$$

  - Variable occurrences can be free, or
  - bounded by the fixed-point operators

- Note the absence of "first class" negation from the syntax

μ-calculus (2/2)

- Semantics
  - Adds function from variables to sets of states called *valuation*

$$\mathcal{V} : Var \rightarrow 2^S$$

  - A variable occurring free is interpreted by the valuation

$$[\![X]\!]^{\mathcal{M}}_{\mathcal{V}} = \mathcal{V}(X)$$

  - Fixed-points are defined according to Tarski-Knaster theorem

$$\begin{aligned}
[\![\mu X.\alpha]\!]^{\mathcal{M}}_{\mathcal{V}} &= \bigsqcap\{S' \subseteq S \mid [\![\alpha]\!]^{\mathcal{M}}_{\mathcal{V}[S'/X]} \subseteq S'\} && \text{(lfp)} \\
&= \bigsqcap\{S' \subseteq S \mid f(S') \subseteq S'\} \\
[\![\nu X.\alpha]\!]^{\mathcal{M}}_{\mathcal{V}} &= \bigsqcup\{S' \subseteq S \mid S' \subseteq [\![\alpha]\!]^{\mathcal{M}}_{\mathcal{V}[S'/X]}\} && \text{(gfp)} \\
&= \bigsqcup\{S' \subseteq S \mid S' \subseteq f(S')\}
\end{aligned}$$

where $f(S') = [\![\alpha]\!]^{\mathcal{M}}_{\mathcal{V}[S'/X]}$

- Tarski-Knaster doesn't help us compute FPs
  It only guarantees their existence
- We will use Kleene's FP theorem for computing FPs

μ-calculus: Example (1/3)

$\mu X.[a]X$ represent states with no infinite sequences of a-transitions

$\mu^0 X.[a]X = \emptyset$ false

$\mu^1 X.[a]X = [a]\emptyset$

$= \{s \in S \mid \forall t.\ s \xrightarrow{a} t \rightarrow t \models \emptyset\}$

since no t satisfies $\emptyset$, the right hand side (RHS) of $\rightarrow$ is false;

thus the left hand side (LHS) of $\rightarrow$ cannot be true.

This represents states with no outgoing a-transitions

$\mu^2 X.[a]X = [a]T$

where $T = \mu^1 X.[a]X$ are states with no outgoing a-transitions

Thus μ^2 means states with no aa-paths

μ-calculus: Example (2/3)

$\nu X.p \wedge [a]X$ is informally analogous to LTL $\Box p$

$\nu^0 X.p \wedge [a]X = S$ true

$\nu^1 X.p \wedge [a]X = p \wedge [a]S$

Intersection between all nodes satisfying p (LHS of $\wedge$)

and all nodes (RHS of $\wedge$)

$\nu^2 X.p \wedge [a]X = p \wedge [a]T$

Where $T = \nu^1 X.p \wedge [a]X$ are all nodes that satisfy p

Thus μ^2 is the intersection between all nodes that satisfy p

and all nodes that have an outgoing edge labeled a

to a node that satisfies p

All nodes that satisfy p and whose descendants that are reachable through a-transitions also satisfy p.

μ-calculus: Example (3/3)

$\mu X.p \vee (\langle a \rangle True \wedge [a]X)$ is informally analogous to LTL $\Diamond p$

$\mu^0 X.p \vee (\langle a \rangle True \wedge [a]X) = \emptyset$

$\mu^1 X.p \vee (\langle a \rangle True \wedge [a]\emptyset) = p \vee (\langle a \rangle True \wedge [a]\emptyset)$

$\langle a \rangle True$ is the set of states with an outer a-transition

$[a]\emptyset$ is the set of states with no outgoing a-transition

Therefore, intersection $\wedge$ is empty

and the formula boils down to the set of states satisfying p

$\mu^2 X.p \vee (\langle a \rangle True \wedge [a]T) = p \vee (\langle a \rangle True \wedge [a]T)$

where $T = \mu^1$ which means nodes satisfying p

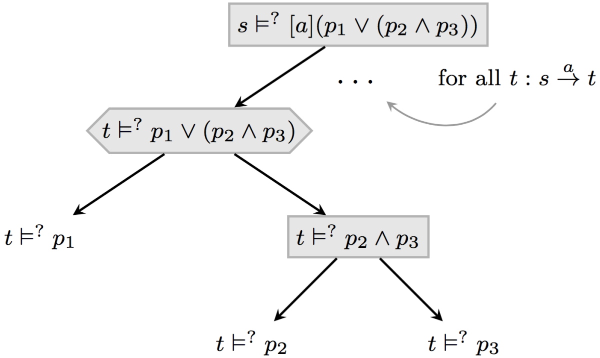
$[a]T$ are nodes whose children reachable via a-transitions satisfy p

Thus either p is satisfied, or it is satisfied via a node reachable through an a-transitions, or via an aa-transition, or via an a^n-transition.

Note

- Increasing complexity with alternation of fixed point types
  - With one fix-point we talk about termination properties
  - With two fix-points we can write fairness formulas

Model checking via parity games (1/5)



Adam pick t from $s \xrightarrow{a} t$ such that $t \not\models (p_1 \vee (p_2 \wedge p3)$

Eve reply by showing that either $t \models p_1$ or that $t \models p_2$ and $t \models p_3$.

Model checking via parity games (2/5)

Definition (Game)

A game is a triple $G = (V, T, Acc)$ where

1. V are *nodes* partitioned between two players, Adam and Eve, $V = V_A \cup V_E$ and $V_A \cap V_E = \emptyset$,
2. $T \subseteq V \times V$ is a *transition relation* determining the possible successors of each node, and
3. $Acc \subseteq V^\omega$ is a set defining the *winning condition*

- It is Adam's turn if $v \in V_A$, otherwise $v \in V_E$ and it is Eve's
- The player who cannot make a move loses
- If a play is infinite, $v_0 v_1 \ldots$, then Eve wins if $v_0 v_1 \ldots \in Acc$

Model checking via parity games (3/5)

Theorem (Reducing model-checking to parity games)

Let $\mathcal{G}(\mathcal{M}, \alpha)$ denote a game constructed from the labeled transition system $\mathcal{M}$ and the μ-calculus formula α.
For every sentence α, transition system $\mathcal{M}$, and initial state s, then $\mathcal{M}, s \models \alpha$ iff Eve has a winning strategy for the position (s, α) in $\mathcal{G}(\mathcal{M}, \alpha)$.

Model checking via parity games (4/5)

Define $\mathcal{G}(\mathcal{M}, \alpha)$ inductively on the syntax of α

- Create node (s, β) for every state s of $\mathcal{M}$ and every formula β in the closure of α (similar to the automata based LTL model checking construction we have seen)
- Recall that Eve's goal is to show that a formula holds, and that the player who can't make a move loses

(s, p) Eve wins if p holds in s, that is $s \models p$
Thus assign (s, p) to Adam and we put no transitions from it

$(s, \neg p)$ Same as (s, p) but reversing Adam and Eve's roles

$(s, \langle a\rangle\beta)$ Connect to (t, β) for all t such that $s \xrightarrow{a} t$ and
$(s, [a]\beta)$ assign $(s, [a]\beta)$ to Adam and $(s, \langle a\rangle\beta)$ to Eve

$(s, \mu X.\beta(X))$ Connect to $(s, \beta(\mu X.\beta(X)))$ and to $(s, \beta(\nu X.\beta(X)))$
$(s, \nu X.\beta(X))$ This corresponds to the intuition that a fixed-point is equivalent to its unfolding.

$$[\![\mu X.\alpha]\!]^{\mathcal{M}}_{\mathcal{V}} = [\![\alpha[\mu X.\alpha/X]]\!]^{\mathcal{M}}_{\mathcal{V}}$$
$$[\![\nu X.\alpha]\!]^{\mathcal{M}}_{\mathcal{V}} = [\![\alpha[\nu X.\alpha/X]]\!]^{\mathcal{M}}_{\mathcal{V}}$$

Model checking via parity games (5/5)

- How to define *Acc* and the *parity winning condition*

  See [Bradfield and Walukiewicz, 2015]

- Model checking $\mathcal{M} \models \alpha$

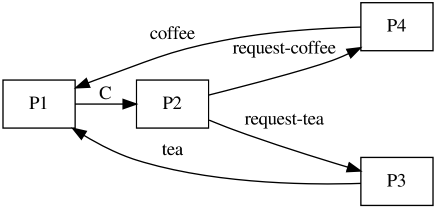
  Use algorithm for determining winner of parity game once $\mathcal{G}(\mathcal{M}, \alpha)$ has been created

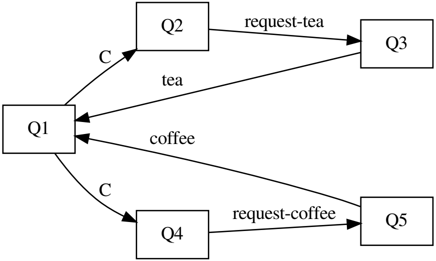
Bisimulation (1/3)

- Equivalence between systems
  - Preserves compositionality
    - Programs as functions (denotational semantics)

$$x := 2 \quad \text{and} \quad x := 1;\ x := x + 1$$
$$x := 2 \mid\mid x := 2 \quad \text{versus} \quad x := 2 \mid\mid x := 1;\ x := x + 1$$

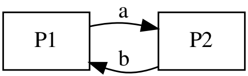
    - Language acceptance (trace equivalence)

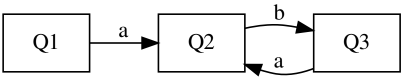




Bisimulation (2/3)

- Equivalence between systems
  - Not overly strong as graph isomorphism





Bisimulation (3/3)

Definition (Bisimulation)

Bisimulation is a symmetric relation $\mathcal{R}$ on the states of an LTS such that whenever $P \mathcal{R} Q$, for all t we have:

- for all P' which $P \xrightarrow{\mathrm{t}} P'$, there is Q' such that $Q \xrightarrow{\mathrm{t}} Q'$ and $P' \mathcal{R} Q'$

Definition (Logic equivalence)

Two statements are logically equivalent if they have the same truth value in every model

logic	logic equivalence
LTL	trace equivalence
HML, μ-calculus, CTL	bisimilarity

References

- Lattice and fixed points

  - Nielson, F., Nielson, H. R., and Hankin, C. (2015). *Principles of program analysis.*
    Springer
  - Davey, B. A. and Priestley, H. A. (2002). *Introduction to lattices and order.*
    Cambridge university press

- μ-calculus and model checking

  - Bradfield, J. and Walukiewicz, I. (2015). The mu-calculus and model-checking.
    Handbook of Model Checking. Springer-Verlag, pages 35–45
  - Cleaveland, R. (1990). Tableau-based model checking in the propositional mu-calculus.
    Acta Informatica, 27(8):725–747

- Bisimulation

  - Sangiorgi, D. (2012). *Introduction to bisimulation and coinduction.*
    Cambridge University Press
