

Algorithms for Model Checking 2IW50

Lecture 5: μ -Calculus

Jaco van de Pol

`vdpol@cwi.nl`

`http://www.cwi.nl/~vdpol/amc.html`

Technische Universiteit Eindhoven
Centrum voor Wiskunde en Informatica

O V E R V I E W

Recall: symbolic model checking for CTL was based on **fixed point characterization** of its operators

Idea of **μ -calculus**: add fixed points operators as primitives to the logic

1. Syntax of μ -calculus
2. Semantics of μ -calculus
3. Complexity of evaluating fixed points
4. Analysis: alternation depth
5. Emerson-Lei improved algorithm
6. Embedding CTL in μ -calculus

Motivation for μ -calculus

- μ -calculus is very **expressive**
(includes CTL, LTL, CTL*)
- μ -calculus is very **pure**
“assembly language” for modal logic
(cf: λ -calculus for functional programming)
- μ -calculus is the basic modal calculus for much **scientific research**
- fragments of the μ -calculus are the basis for practical model checkers, such as CADP

Kripke structures & LTS

Mix of Kripke Systems and Labeled Transition Systems: $M = (S, Act, R, L)$ over a set AP of atomic propositions:

- S is a set of states
- Act is a set of action labels
- R is a **labeled** transition relation:
 $R \subseteq S \times Act \times S$
- L is a labeling: $L \in S \rightarrow 2^{AP}$.

Notation: $s \xrightarrow{a} t$ denotes $(s, a, t) \in R$

Special cases:

- **Kripke Structures:** Act is a singleton (only one transition relation)
- **LTS** (process algebra): AP is empty (only propositions: True, False)

In the book, the set of labels Act is not made explicit, but $R \subseteq 2^{S \times S}$ is a set of transitions: for each $a \in R$, $a \subseteq S \times S$.

Syntax of μ -Calculus

Let the following sets be given: AP (atomic propositions); Act (action labels); Var (formal relation variables);

The set of μ -calculus formulas F is inductively defined as:

- if $p \in AP$ then $p \in F$
- if $Q \in Var$ then $Q \in F$
- if $f, g \in F$ then $\neg f, f \wedge g, f \vee g \in F$
- if $f \in F$ and $a \in Act$, then
 - $[a]f \in F$ (for all a -successors)
 - $\langle a \rangle f \in F$ (for some a -successor)
- if $f \in F$ and $Q \in Var$, and **all occurrences of Q in f are under an even number of negations (\neg)** then
 - $\mu Q. f \in F$ (least fixpoint)
 - $\nu Q. f \in F$ (greatest fixpoint)

Some notation/terminology

- “ Q occurs in f only under an even number of \neg -symbols” is called the **syntactic monotonicity** criterion.

This ensures the existence of fixed points.

- An occurrence of Q is **bound** by a surrounding νQ or μQ . Unbound occurrences of Q are called **free**
- A **closed** formula has no free variables. If it has free variables, a formula is called **open**.
- An **environment** e interprets the free relation variables Q :
 - $e : Var \rightarrow 2^S$
 - $e[Q \leftarrow V]$ is a new environment like e , but Q is set to V :

$$e[Q \leftarrow V](X) := \begin{cases} V & \text{if } X = Q \\ e(X) & \text{otherwise} \end{cases}$$

Semantics of μ -calculus

Fix a system: $M = (S, Act, R, L)$.

- The semantics of open formulas is only defined if we know the values of the free variables.
- The semantics of a μ -calculus formula f under environment e is the set of states where it holds:

$$\llbracket p \rrbracket_e = \{s \mid p \in L(s)\}$$

$$\llbracket Q \rrbracket_e = e(Q)$$

$$\llbracket \neg f \rrbracket_e = S \setminus \llbracket f \rrbracket_e$$

$$\llbracket f \wedge g \rrbracket_e = \llbracket f \rrbracket_e \cap \llbracket g \rrbracket_e$$

$$\llbracket f \vee g \rrbracket_e = \llbracket f \rrbracket_e \cup \llbracket g \rrbracket_e$$

$$\llbracket [a]f \rrbracket_e = \{s \mid \forall t. s \xrightarrow{a} t \Rightarrow t \in \llbracket f \rrbracket_e\}$$

$$\llbracket \langle a \rangle f \rrbracket_e = \{s \mid \exists t. s \xrightarrow{a} t \wedge t \in \llbracket f \rrbracket_e\}$$

$$\llbracket \mu Q. f \rrbracket_e = \text{lfp}(Z \mapsto \llbracket f \rrbracket_{e[Q \leftarrow Z]})$$

$$\llbracket \nu Q. f \rrbracket_e = \text{gfp}(Z \mapsto \llbracket f \rrbracket_{e[Q \leftarrow Z]})$$

This is also the **naive algorithm** for model checking μ -calculus (*lfp* and *gfp* by iteration)

Examples

- $\mu Q. \neg Q$ (is **not** a μ -calculus formula)
- Let $A = \{a\}$:
 - $\mathbf{EG} f \dots \nu Q. f \wedge \langle a \rangle Q$
 - $\mathbf{E}[f \mathbf{U} g] \dots \mu Q. g \vee (f \wedge \langle a \rangle Q)$
 - Every p is inevitably followed by a q
 $\nu Q_1. \left((p \Rightarrow (\underline{\mu Q_2. q \vee [a] Q_2})) \wedge [a] Q_1 \right)$
- The last formula can be evaluated “inside-out”:

$$\begin{array}{l}
 Q_2^0 = \textit{False} \\
 Q_2^1 = q \vee [a] Q_2^0 \\
 Q_2^2 = q \vee [a] Q_2^1 \\
 \dots Q_2^\omega = q \vee [a] Q_2^\omega \\
 \hline
 Q_1^0 = \textit{True} \\
 Q_1^1 = p \Rightarrow (Q_2^\omega \wedge [a] Q_1^0) \\
 Q_1^2 = p \Rightarrow (Q_2^\omega \wedge [a] Q_1^1) \\
 \dots Q_1^\omega = p \Rightarrow (Q_2^\omega \wedge [a] Q_1^\omega)
 \end{array}$$

- **Special case:** Q_1 doesn't occur below μQ_2 .

A more difficult case

- On some path, h occurs infinitely often:

$$\nu Q_1. \langle a \rangle \left(\underline{\mu Q_2. (Q_1 \wedge h) \vee \langle a \rangle Q_2} \right)$$

- Problem: the inner fixpoint depends on Q_1 .

$$Q_1^0 = \textit{True}$$

$$Q_2^{00} = \textit{False}$$

$$Q_2^{01} = (Q_1^0 \wedge h) \vee \langle a \rangle Q_2^{00}$$

$$Q_2^{02} = (Q_1^0 \wedge h) \vee \langle a \rangle Q_2^{01}$$

$$\dots Q_2^{0\omega} = (Q_1^0 \wedge h) \vee \langle a \rangle Q_2^{0\omega}$$

$$Q_1^1 = \langle a \rangle Q_2^{0\omega}$$

$$Q_2^{10} = \textit{False}$$

$$Q_2^{11} = (Q_1^1 \wedge h) \vee \langle a \rangle Q_2^{10}$$

$$\dots Q_2^{1\omega} = (Q_1^1 \wedge h) \vee \langle a \rangle Q_2^{1\omega}$$

$$Q_1^2 = \langle a \rangle Q_2^{1\omega}$$

$$\dots Q_1^\omega = \langle a \rangle Q_2^{\omega\omega}$$

Complexity of naive algorithm:

- We check formula f with at most k nested fixed points on the Kripke structure $M = (S, R, Act, L)$.
- In the previous example:
 - The outermost (greatest) fixpoint can decrease at most $|S|$ times.
 - Each time, the innermost (least) fixpoint can increase at most $|S|$ times, so in total this one is evaluated at most $|S|^2$ times.
- In general: the inner fixpoint of formula f is evaluated at most $|S|^k$, where k is the maximum number of nested fixpoints in f .
- Each iteration requires up to $|M| \cdot |f|$ steps.
- **Total time complexity** of naive algorithm:
 $O((|S| + |R|) \cdot |f| \cdot |S|^k)$.

A more careful analysis will yield a more optimal treatment for nested fixpoints **of the same type**.

Positive normal form

- A μ -calculus formula is in **positive normal form** if negations occur only before propositions.
- to transform a formula to positive normal form, push negations inside using logical **dualities**:

$\neg\neg f$	\mapsto	f
$\neg(f \vee g)$	\mapsto	$(\neg f) \wedge (\neg g)$
$\neg(f \wedge g)$	\mapsto	$(\neg f) \vee (\neg g)$
$\neg([a]f)$	\mapsto	$\langle a \rangle(\neg f)$
$\neg(\langle a \rangle f)$	\mapsto	$[a](\neg f)$
$\neg(\mu Q. f(Q))$	\mapsto	$\nu Q. \neg f(\neg Q)$
$\neg(\nu Q. f(Q))$	\mapsto	$\mu Q. \neg f(\neg Q)$

- Note: due to syntactic monotonicity, single negations before relation variables cannot arise.
- Hence, the result is a positive normal form.
- **Check: the result is logically equivalent.**

How complex is a μ -formula? (1)

- **Nesting Depth:**

maximal number of nested fixed points in a positive normal form

- $ND(p) := 0$
- $ND(Q) := 0$
- $ND(\neg f) := ND(f)$
- $ND([a]f) := ND(f)$
- $ND(\langle a \rangle f) := ND(f)$
- $ND(f \wedge g) := \max(ND(f), ND(g))$
- $ND(f \vee g) := \max(ND(f), ND(g))$
- $ND(\mu Q. f) := 1 + ND(f)$
- $ND(\nu Q. f) := 1 + ND(f)$

- **Example:**

$$ND\left(\left(\mu Q_1. \nu Q_2. Q_1 \vee Q_2\right) \wedge \left(\underline{\mu Q_3}. \underline{\mu Q_4}. \left(Q_3 \wedge \underline{\mu Q_5}. p \vee Q_5\right)\right)\right) = 3$$

How complex is a μ -formula? (2)

- **Alternation Depth:** number of alternating fixed points of a formula in positive normal form.

$$- AD(p) := AD(\neg p) := AD(Q) := 0$$

$$- AD([a]f) := AD(\langle a \rangle f) := AD(f)$$

$$- AD(f \wedge g) := \max(AD(f), AD(g))$$

$$- AD(f \vee g) := \max(AD(f), AD(g))$$

$$- AD(\mu Q. f) :=$$

$$1 + \max\{AD(g) \mid g \text{ is a } \nu\text{-subformula of } f\}$$

$$- AD(\nu Q. f) :=$$

$$1 + \max\{AD(g) \mid g \text{ is a } \mu\text{-subformula of } f\}$$

(where by agreement, $\max(\emptyset) = 0$)

- **Examples:**

$$AD\left(\left(\underline{\mu Q_1} \cdot \underline{\nu Q_2} \cdot Q_1 \vee Q_2\right) \wedge \left(\mu Q_3 \cdot \mu Q_4 \cdot \left(Q_3 \wedge \mu Q_5 \cdot p \vee Q_5\right)\right)\right) = 2$$

$$AD\left(\left(\mu Q_1 \cdot \nu Q_2 \cdot Q_1 \vee Q_2\right) \wedge \left(\underline{\mu Q_3} \cdot \underline{\nu Q_4} \cdot \left(Q_3 \wedge \underline{\mu Q_5} \cdot p \vee Q_5\right)\right)\right) = 3$$

How complex is a μ -formula? (3)

- **Dependent Alternation Depth (dAD):** number of alternating fixed points, such that the innermost fixpoint **depends** on the outermost.
- The definition of $dAD()$ is similar to $AD()$, but:
 - $dAD(\mu Q. f) := \max(dAD(f), 1 + \max\{dAD(g) \mid g \text{ is a } \nu\text{-subformula of } f \text{ and } Q \text{ occurs in } g\})$
 - $dAD(\nu Q. f) := \max(dAD(f), 1 + \max\{dAD(g) \mid g \text{ is a } \mu\text{-subformula of } f \text{ and } Q \text{ occurs in } g\})$

(where by agreement, $\max(\emptyset) = 0$)

- **Examples:**

$$dAD\left(\left(\underline{\mu Q_1} \cdot \underline{\nu Q_2} \cdot \underline{Q_1} \vee Q_2\right) \wedge \left(\mu Q_3 \cdot \mu Q_4 \cdot \left(Q_3 \wedge \mu Q_5 \cdot p \vee Q_5\right)\right)\right) = 2$$

$$dAD\left(\left(\underline{\mu Q_1} \cdot \underline{\nu Q_2} \cdot \underline{Q_1} \vee Q_2\right) \wedge \left(\underline{\mu Q_3} \cdot \underline{\nu Q_4} \cdot \left(\underline{Q_3} \wedge \mu Q_5 \cdot p \vee Q_5\right)\right)\right) = \mathbf{2}$$

A bit more fixpoint theory

- Given a **finite** set S and a **monotonic** $\tau : 2^S \rightarrow 2^S$ in the partial order $(2^S, \subseteq)$.
- We used to compute the least fixpoint from \emptyset :

$$\emptyset \subseteq \tau(\emptyset) \subseteq \tau^2(\emptyset) \subseteq \dots \subseteq \tau^i(\emptyset) = \tau^{i+1}(\emptyset)$$

then $\mu X. \tau(X) = \tau^i(\emptyset)$

- **Actually, instead of \emptyset we can start in any set known to be smaller than the fixpoint:**
- Next, assume $W \subseteq \mu X. \tau(X)$, so we have:

$$\emptyset \subseteq W \subseteq \tau^i(\emptyset)$$

- By monotonicity and the definition of fixpoint:

$$\tau^i(\emptyset) \subseteq \tau^i(W) \subseteq \tau^{i+i}(\emptyset) = \tau^i(\emptyset)$$

- So if $W \subseteq \mu X. \tau(X)$ we compute the lfp as:

$$W, \tau(W), \tau^2(W), \dots, \tau^j(W) = \tau^{j+1}(W)$$

This converges at some $j \leq i$ (may be $<$)

Nested fixpoints of same type

- We now consider two nested μ -fixpoints:

$$\mu Q_1. \sigma(Q_1, \underline{\mu Q_2. \tau(Q_1, Q_2)})$$

- Start computation with $Q_1 = Q_2 = \text{False}$:

$$\begin{aligned} Q_1^0 &= \text{False} & Q_2^{00} &= \text{False} \\ & & Q_2^{01} &= \tau(Q_1^0, Q_2^{00}) \\ & & \dots Q_2^{0\omega} &= \tau(Q_1^0, Q_2^{0\omega}) \\ Q_1^1 &= \sigma(Q_1^0, Q_2^{0\omega}) \end{aligned}$$

- Clearly, $Q_1^0 \subseteq Q_1^1$, so also

$$Q_2^{0\omega} = \mu Q_2. \tau(Q_1^0, Q_2) \subseteq \mu Q_2. \tau(Q_1^1, Q_2) = Q_2^{1\omega}$$

- So we can continue with $Q_2^{0\omega}$ instead of restarting at *False*!

$$\begin{aligned} Q_2^{10} &= Q_2^{0\omega} \\ Q_2^{11} &= \tau(Q_1^1, Q_2^{10}) \\ \dots Q_2^{1\omega} &= \tau(Q_1^1, Q_2^{1\omega}) \\ Q_1^2 &= \sigma(Q_1^1, Q_2^{1\omega}) \end{aligned}$$

Emerson-Lei algorithm (idea)

Idea:

- function $eval(f, e)$ will return the set of states that satisfy formula f , in environment e .
(recall that $e : Var \rightarrow 2^S$)
- Follow the **recursive** definition of the semantics of μ -calculus formulas.
- **Save** the value of the current approximation of the fixpoint for Q_i in array $A[i]$, in order to reuse it in later iterations.
- **Reset** $A[i]$ only if:
 - an upper Q_j of different type changes, and
 - **variable** Q_j occurs below the μQ_i or νQ_i .
- The latter test makes it **essentially** more optimal than the variant of the algorithm in the book.

Emerson-Lei algorithm (code)

Initialization:

forall fixed point i **do**:

$$A[i] := \begin{cases} \textit{False} & \text{if } Q_i \text{ has a } \mu \\ \textit{True} & \text{if } Q_i \text{ has a } \nu \end{cases}$$

function $eval(f, e)$

if $f = Q_i$ **then return** $e(Q_i)$;

if $f = g_1 \vee g_2$ **then return** $eval(g_1) \cup eval(g_2)$;

...

if $f = \mu Q_i. g(Q_i)$ **then**

forall top-level formulas $\nu Q_j. g'(Q_j)$ within g **do**

if Q_i occurs in $g'(Q_j)$

then $A[j] := \textit{True}$; /* **Reset** */

repeat

$Q_{old} := A[i]$; /* **continue from prev. value** */

$A[i] := eval(g, e[Q_i \leftarrow A[i]])$;

until $A[i] = Q_{old}$

return $A[i]$

...

Complexity of Emerson-Lei

- Let formula f be given, with dependent alternation depth $dAD(f) = d$.
- Let the Kripke structure be (S, Act, R, L) .
- Take a block of fixed points of the same type:
 - its length is at most $|f|$.
 - the value of each fixed point in it can grow/shrink at most $|S|$ times.
- In total, the innermost block will have no more than $(|f| \cdot |S|)^d$ iterations of the repeat-loop.
- Each iteration requires time at most $O(|f| \cdot (|S| + |R|))$
- Hence: the overall complexity of Emerson-Lei is

$$O(|f| \cdot (|S| + |R|) \cdot (|f| \cdot |S|)^d)$$

Embedding CTL-formulas

Again assume $A = \{a\}$.

Given the fixed-point characterization of CTL, there is a straightforward translation of CTL to μ -calculus:

- $Tr(p) = p$
- $Tr(\neg f) = \neg Tr(f)$
- $Tr(f \wedge g) = Tr(f) \wedge Tr(g)$
- $Tr(\mathbf{EX} f) = \langle a \rangle Tr(f)$
- $Tr(\mathbf{EG} f) = \nu Y. (Tr(f) \wedge \langle a \rangle Y)$
- $Tr(\mathbf{E}[f \mathbf{U} g]) = \mu Y. (Tr(g) \vee (Tr(f) \wedge \langle a \rangle Y))$

Note:

- $Tr(f)$ is syntactically monotone
- $Tr(f)$ is a closed μ -calculus formula
- $dAD(f) \leq 1$ (although $AD(f)$ is not limited!)
which is called the **alternation free** fragment.

Conclusion

- μ -calculus incorporates **least + greatest fixed points** directly in the logic
- the **naive** algorithm is **exponential** in the **nesting depth** of fixpoints
- a careful analysis leads to an algorithm which is **exponential** in the **(dependent) alternation depth** only
- Hence: alternation free μ -calculus is **linear** in the Kripke structure and **polynomial** in the formula.
- **CTL** translates into the **alternation free** fragment of μ -calculus
- for the latter we essentially needed the **dependent** alternation depth.
- fairness constraints typically lead to one extra alternation ($dAD() = 2$)