The μ-Calculus & Temporal Logics

April 2004

S. Arun-Kumar

sak@cse.iitd.ernet.in

Department of Computer Science and Engineering

I. I. T. Delhi, Hauz Khas, New Delhi 110 016.

September 20, 2005

1. The μ-calculus

The Propositional μ-calculus

Background on fixpoints

Let S be any finite set (putatively it is the set of states of some transition system). Then 2^S is the powerset of S. It is clear that 2^S under the subset ordering $\subseteq$ forms a complete lattice. in which

(i) S is the top element

(ii) φ is the bottom element

(iii) For any collection of sets $\mathcal{P} = \{S_i \mid S_i \subseteq S\}$,

$\bigcup_i S_i$ is the least upper bound of $\mathcal{P}$

and $\bigcap_i S_i$ is the greatest lower bound

- If S is the set of states then each element of 2^S can be thought of as a predicate on S.
- Let $f: 2^S \to 2^S$ be any function. Then f is called a predicate transformer since it takes a predicate as an argument to return another predicate as value.
- We identify the predicate true with S and false with the empty set φ.

- Unions of predicates represent disjunctions ($\vee$) and intersections represent conjunctions ($\wedge$)
- Set complement represents negation ($\neg$).
- $f: \mathbb{Z}^S \to \mathbb{Z}^S$ is monotonic provided for all $S_1, S_2 \subseteq S$, $\boxed{S_1 \subseteq S_2 \Rightarrow f(S_1) \subseteq f(S_2)}$
- f is $\cup$-continuous provided for any (infinite) chain $S_1 \subseteq S_2 \subseteq \ldots$

$$f\left(\bigcup_{i \geq 1} S_i\right) = \bigcup_{i \geq 1} f(S_i)$$

- f is $\cap$-continuous provided for any (infinite) chain $S_1 \supseteq S_2 \supseteq \ldots$

$$f\left(\bigcap_{i \geq 1} S_i\right) = \bigcap_{i \geq 1} f(S_i).$$

- The n-fold application of f to any $T \subseteq S$ for $n \geq 0$ is defined inductively as

$$f^0(T) = T$$
$$f^{n+1}(T) = f(f^n(T)) = f^n(f(T))$$

Fixpoints of Monotonic Functions

The Knaster-Tarski theorem: If $f: 2^S \to 2^S$ is a monotonic function then the equation

$$X = f(X)$$

(S not necessarily finite)

has a **least solution** $L = \bigcap \{T \subseteq S \mid f(T) \subseteq T\}$

and a **greatest solution** $G = \bigcup \{T \subseteq S \mid T \subseteq f(T)\}$.

Proof: Consider the two sets

$$\mathcal{T} = \{T \subseteq S \mid f(T) \subseteq T\} \qquad \mathcal{U} = \{T \subseteq S \mid T \subseteq f(T)\}.$$

Claim 1. $\mathcal{T} \neq \varphi \neq \mathcal{U}$

Since $f(S) \subseteq S$ and $\varphi \subseteq f(\varphi)$ resp.

Claim 2. Every solution of the equation $X = f(X)$ is a member of $\mathcal{T}$ and $\mathcal{U}$, since for any solution A, we have $A = f(A) \Rightarrow A \subseteq f(A) \subseteq A$.

Claim 3. $L \subseteq T$ for all $T \in \mathcal{T}$

~~$\Rightarrow f(L) \subseteq f(T)$ for all $T \in \mathcal{T}$.~~

$\Rightarrow f(L) \subseteq f(T) \subseteq T$ for all $T \in \mathcal{T}$.

$\Rightarrow f(L) \subseteq L$ —— ①

$\Rightarrow L \in \mathcal{T}$ and $f(f(L)) \subseteq f(L)$

$\Rightarrow f(L) \in \mathcal{T}$

$\Rightarrow L \subseteq f(L)$ —— ②

From ① and ② we have $L = f(L)$ and L is the least solution. ③

Claim 4.

$G \supseteq U$ for each $U \in \mathcal{U}$

$\Rightarrow f(G) \supseteq f(U) \supseteq U$ for each $U \in \mathcal{U}$

$\Rightarrow f(G) \supseteq G$ — ③

$\Rightarrow G \in \mathcal{U}$ and $f(f(G)) \supseteq f(G)$

$\Rightarrow f(G) \in \mathcal{U}$

$\Rightarrow G \supseteq f(G)$ — ④

From ③ & ④ $G = f(G)$ and G is the greatest fixpoint of f.

Theorem: If f is $\cup$-continuous and $\cap$-continuous, then

(i) f is monotonic

(ii) the least fixpoint (lfp) and the greatest fixpoint (gfp) of f are given by

$$lfp(f) = \bigcup_{i \geq 0} f^i(\phi)$$

$$gfp(f) = \bigcap_{i \geq 0} f^i(S)$$

Proof: If f is $\cup$-continuous then for any chain $S_1 \subseteq S_2 \subseteq \ldots$ we have

$$f\left(\bigcup_{i \geq 0} S_i\right) = \bigcup_{i \geq 0} f(S_i)$$

i.e. $f(S_1 \cup S_2) = f(S_1) \cup f(S_2)$ and $f(S_1 \cap S_2) = f(S_1) \cap f(S_2)$

④

If $S_1 \subseteq S_2$ we have

$S_1 \cup S_2 = S_2$ and $S_1 \cap S_2 = S_1$

$\Rightarrow f(S_1 \cup S_2) = f(S_1) \cup f(S_2) = f(S_2)$ and

$f(S_1 \cap S_2) = f(S_1) \cap f(S_2) = f(S_1)$

$\Rightarrow f(S_1) \subseteq f(S_2)$ and $f(S_1) \subseteq f(S_2)$. ⊣

(ii) Claim $f^i(\varphi)$ is a chain for $i \geq 0$.

Since f is $\cup$-continuous, it is also monotonic

Hence since $\varphi \subseteq f(\varphi)$ we have

$\varphi \subseteq f(\varphi)$ and for each $i \neq 0$ such that

$f^{i-1}(\varphi) \subseteq f^i(\varphi)$ we have

$f^i(\varphi) = f(f^{i-1}(\varphi)) \subseteq f(f^i(\varphi)) = f^{i+1}(\varphi)$.

Since f is $\cup$-continuous we have

$$f\left(\bigcup_{i \geq 0} f^i(\varphi)\right) = \bigcup_{i \geq 0} f^{i+1}(\varphi) = \bigcup_{i \geq 0} f^i(\varphi)$$

Hence $\bigcup_{i \geq 0} f^i(\varphi)$ is a fixpoint of f.

For any other fixpoint F we have

$\varphi \subseteq F$ and hence $f^i(\varphi) \subseteq f^i(F) = F$

for all $i \geq 0$. Hence $\bigcup_{i \geq 0} f^i(\varphi) \subseteq \bigcup_{i \geq 0} f^i(F) = F$

Hence $\bigcup_{i \geq 0} f^i(\varphi)$ is the least fixpoint of f.

A similar argument may be used to prove that $\bigcap_{i \geq 0} f^i(S)$ is the greatest fixpoint of f. if f is $\cap$- continuous.

Firstly of course f is monotonic if it is $\cap$- continuous. Next from

$$S \supseteq f(S)$$

we obtain $f^i(S) \supseteq f^{i+1}(S)$ for all $i \geq 0$.

and further that

$$S \supseteq f(S) \supseteq f^2(S) \supseteq \ldots$$

is a decreasing chain and

$$f\left(\bigcap_{i \geq 0} f^i(S)\right) = \bigcap_{i \geq 0} f^{i+1}(S) = \bigcap_{i \geq 0} f^i(S)$$ which is a fixpoint.

and for any other $F \subseteq S$ we have

$f^i(F) \subseteq f^i(S)$ and $F = \bigcap_{i \geq 0} f^i(F) \subseteq \bigcap_{i \geq 0} f^i(S)$

which shows that $\bigcap_{i \geq 0} f^i(S)$ is the greatest fixpoint.

Theorem: When S is finite $|S| = n$ then for any monotone function $f : 2^S \to 2^S$, $f^n(\phi)$ and $f^n(S)$ are the lfp(f) and gfp(f) respectively.

Proof:

Claim: If S is finite then any monotonic function is both $\cup$-continuous and $\cap$-continuous.

$\vdash$ Consider any chains

$$T_1 \subseteq T_2 \subseteq \ldots \subseteq S$$

and $U_1 \supseteq U_2 \supseteq \ldots$ with $S \supseteq U_1$

Since S is finite it is clear that for some $j \geq 0$, $T_{j+k} = T_j$ for all $k \geq 0$ and $U_{j+k} = U_j$ for all $k \geq 0$.

Hence all increasing and decreasing chains can have at most n-distinct elements in them. For any such chains it is clear from the monotonicity of f that

$$f\left(\bigcup_{i \geq 0} T_i\right) = f(T_j) = \bigcup_{i \geq 1} f(T_i)$$

and $$f\left(\bigcap_{i \geq 1} U_i\right) = f(U_j) = \bigcap_{i \geq 1} f(U_i).$$

Hence the $lfp(f)$ and $gfp(f)$ are given by the least $j \geq 0$ such that $f^j(\varphi) = f^{j+k}(\varphi)$ and $f^j(S) = f^{j+k}(S)$ for all $k \geq 0$. In fact the least such $j \leq n$. in order for the chains

$$\varphi \subseteq f(\varphi) \subseteq \ldots \subseteq f^j(\varphi) = f^{j+1}(\varphi) = \ldots$$

and $$S \supseteq f(S) \supseteq \ldots \supseteq f^j(S) = f^{j+1}(S) = \ldots$$

Clearly for any i, if $f^{i+1}(\varphi) = f^i(\varphi)$ then $f^{i+k}(\varphi) = f^i(\varphi)$ for all $k \geq 0$.

Similarly if $f^{i+1}(S) = f^i(S)$ then $f^{i+k}(S) = f^i(S)$ for all $k \geq 0$.

Hence chains

$$\varphi \subset f(\varphi) \subset \ldots \subset f^j(\varphi) = f^{j+1}(\varphi) = \ldots$$

and $S \supset f(S) \supset \ldots \supset f^j(S) = f^{j+1}(S) = \ldots$

are such that $j \leq n$. Hence $f^n(\varphi)$ and $f^n(S)$ are the respective fixpoints.

Notation: For any equation $X = f(X)$ for functions $f : 2^S \to 2^S$ we denote the least and greatest solutions by

$$\mu[X = f(X)]$$

and

$$\nu[X = f(X)]$$

respectively.

Exercise:

1. If f does not contain any free occurrences of X, then what would $\mu[X = f]$ and $\nu[X = f]$ signify?

2. Can you give non-trivial examples of recursion equations $X = f(X)$ such that

$$\mu[X = f(X)] = \nu[X = f(X)] \; ?$$

Simultaneous Equations in two variables

Given the equations

$$X_1 = f_1(X_1, X_2)$$
$$\text{and } X_2 = f_2(X_1, X_2)$$

We may think of this System as a Single equation

$$\vec{X} = \vec{f}(\vec{X})$$

where $\vec{X} = \langle X_1, X_2 \rangle$ and

$$\vec{f}(\vec{X}) = \langle f_1(\vec{X}), f_2(\vec{X}) \rangle$$

$$\vec{f}: 2^S \times 2^S \to 2^S \times 2^S$$

Fact: Given complete lattices $(L_1, \sqsubseteq_1)$ and $(L_2, \sqsubseteq_2)$, $(L_1 \times L_2, \sqsubseteq_{12})$ is also a complete lattice where

$$\langle x_1, x_2 \rangle \sqsubseteq_{12} \langle y_1, y_2 \rangle \text{ iff } x_1 \sqsubseteq_1 y_1 \text{ and } x_2 \sqsubseteq_2 y_2$$

⊢ The bottom element is $\langle \bot_1, \bot_2 \rangle$ and the top element is $\langle \top_1, \top_2 \rangle$. The join and meet operations are defined pointwise. i.e.

$$\langle x_1, x_2 \rangle \sqcup_{12} \langle y_1, y_2 \rangle = \langle x_1 \sqcup_1 y_1, x_2 \sqcup_2 y_2 \rangle$$
$$\langle x_1, x_2 \rangle \sqcap_{12} \langle y_1, y_2 \rangle = \langle x_1 \sqcap_1 y_1, x_2 \sqcap_2 y_2 \rangle$$

Consider any infinite chain in $L_1 \times L_2$

$$\langle x_1^1, x_2^1 \rangle \sqsubseteq_{12} \langle x_1^2, x_2^2 \rangle \sqsubseteq_{12} \cdots \sqsubseteq_{12} \langle x_1^i, x_2^i \rangle \sqsubseteq_{12} \langle x_1^{i+1}, x_2^{i+1} \rangle \cdot$$

$$\sqsubseteq_{12} \cdots$$

Then clearly

$$x_1^1 \sqsubseteq_1 x_1^2 \sqsubseteq_1 \cdots \sqsubseteq_1 x_1^i \sqsubseteq_1 x_1^{i+1} \sqsubseteq_1 \cdots$$

and $x_2^1 \sqsubseteq_2 x_2^2 \sqsubseteq_2 \cdots \sqsubseteq_2 x_2^i \sqsubseteq_2 x_2^{i+1} \sqsubseteq_2 \cdots$

Define

$$\bigsqcup_{i>0}{}_{12} \langle x_1^i, x_2^i \rangle = \langle \bigsqcup_{i>0}{}_1 x_1^i, \bigsqcup_{i>0}{}_2 x_2^i \rangle$$

which is clearly the least upper bound of the chain

Similarly we may define for decreasing chains

$$\bigsqcap_{i>0}{}_{12} \langle x_1^i, x_2^i \rangle = \langle \bigsqcap_{i>0}{}_1 x_1^i, \bigsqcap_{i>0}{}_2 x_2^i \rangle \quad \dashv$$

Def. If $f(X,Y)$ is monotonic in each of its arguments then f is a monotonic function

Lemma. $\vec{f}$ is monotonic if each component of f is monotonic. $\dashv$

Given monotonic functions $f_1(X_1, X_2)$ and $f_2(X_1, X_2)$ clearly $\vec{f}(\vec{X})$ is also a monotonic function on the complete lattice $\mathbb{Z}^S \times \mathbb{Z}^S$. Hence $\vec{f}$ has a least and greatest fixpoint given by

$$\mu\vec{X}[\vec{f}(\vec{X})] = \bigcap \{\vec{T} \mid \vec{f}(\vec{T}) \subseteq \vec{T}\}$$

$$\nu\vec{X}[\vec{f}(\vec{X})] = \bigcup \{\vec{U} \mid \vec{U} \subseteq \vec{f}(\vec{U})\}$$

where $\subseteq$ is the pointwise extension of the subset ordering $\subseteq$ on $\mathbb{Z}^S$ to $\mathbb{Z}^S \times \mathbb{Z}^S$.

Equations with free variables

Consider the equation

$$X = f(X,Y).$$

where $f: 2^S \times 2^S \to 2^S$ is monotone over X

i.e. for any X_1, X_2 and Y

$$X_1 \subseteq X_2 \Rightarrow f(X_1,Y) \subseteq f(X_2,Y)$$

For each value $Y \subseteq S$ define a function f_Y s.t.

$$f_Y(X) = f(X,Y)$$

Clearly for each $Y \subseteq S$, f_Y is monotone and hence it is possible to compute the least and greatest solutions viz. $\mu X[f_Y(X)]$ and $\nu X[f_Y(X)]$.

Then we have the following lemma

Lemma. If $f(X,Y)$ is monotone in X and $Y_1, Y_2 \subseteq S$

then $\mu X[f_{Y_1}(X)] \subseteq \mu X[f_{Y_2}(X)]$

and $\nu X[f_{Y_1}(X)] \subseteq \nu X[f_{Y_2}(X)]$

provided $f(X,Y_1) \subseteq f(X,Y_2)$ for all $X \subseteq S$.

Y_1 and Y_2 need not be related!

Proof: By definition we have

$$\mu X[f_{Y_1}(X)] = \bigcap \{T_1 \subseteq S \mid f_{Y_1}(T_1) \subseteq T_1\} = \bigcap \mathcal{T}_1$$

$$\mu X[f_{Y_2}(X)] = \bigcap \{T_2 \subseteq S \mid f_{Y_2}(T_2) \subseteq T_2\} = \bigcap \mathcal{T}_2.$$

$$\nu X[f_{Y_1}(X)] = \bigcup \{U_1 \subseteq S \mid f_{Y_1}(U_1) \supseteq U_1\} = \bigcup \mathcal{U}_1$$

$$\nu X[f_{Y_2}(X)] = \bigcup \{U_2 \subseteq S \mid f_{Y_2}(U_2) \supseteq U_2\} = \bigcup \mathcal{U}_2$$

Hence for any $T_2 \in \mathcal{T}_2$ we have

$$f_{Y_2}(T_2) \subseteq T_2 \Rightarrow f(T_2, Y_2) \subseteq T_2$$

Since $f(X, Y_1) \subseteq f(X, Y_2)$ for all $X \subseteq S$ we have

$$f(T_2, Y_1) \subseteq f(T_2, Y_2) \subseteq T_2$$

$\Rightarrow T_2 \in \mathcal{T}_1$.

$\therefore\ \mathcal{T}_2 \subseteq \mathcal{T}_1 \Rightarrow \bigcap \mathcal{T}_2 \supseteq \bigcap \mathcal{T}_1$

Hence $\boxed{\mu X[f_{Y_1}(X)] \subseteq \mu X[f_{Y_2}(X)]}$

Similarly for any $T_1 \in \mathcal{T}_1$ we have

$$f_{Y_1}(T_1) \supseteq T_1 \Rightarrow f(T_1, Y_1) \supseteq T_1$$

i.e. $T_1 \subseteq f(T_1, Y_1) \subseteq f(T_1, Y_2)$

$\Rightarrow T_1 \in \mathcal{T}_2$

$\therefore\ \mathcal{T}_1 \subseteq \mathcal{T}_2 \Rightarrow \bigcup \mathcal{T}_1 \subseteq \bigcup \mathcal{T}_2$

Hence $\boxed{\nu X[f_{Y_1}(X)] \subseteq \nu X[f_{Y_2}(X)]}$

(5)

The Propositional μ-calculus

Syntax

Let $VAR = \{X, Y, \ldots\}$ be an infinite collection of relational variables

$AP = \{p, q, \ldots\}$ a set of propositional atoms

$Act = \{a, b, c, \ldots\}$ a set of actions

Then

$$\varphi ::= p \mid X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \langle a \rangle \varphi \mid [a]\varphi \mid \mu[X=\varphi] \mid \nu[X=\varphi]$$

Syntactically monotone

The semantics of μ-calculus formulas is defined over a labelled transition system $\langle S, Act, \rightarrow, \ell \rangle$

where S is the set of states

$\rightarrow \subseteq S \times Act \times S$ is the labelled transition relation

$\ell : S \rightarrow 2^{AP}$ is the labelling on states

Syntactically monotone

Restriction to monotone operators. Since we are calculating fixpoints over predicate transformers. Given the usual notions of free and bound relational variables we restrict

$\mu X[\varphi]$ and $\nu X[\varphi]$ to those formulae φ in which all free occurrences of X occur under an even number of $\neg$

Semantics

Let $\mathcal{L} = \langle S, Act, \rightarrow, \ell \rangle$ be a LTS with ℓ being a labelling function on states. Given an environment

$$e : VAR \longrightarrow 2^S$$

let $e_X : VAR - \{X\} \longrightarrow 2^S$ where $e_X(X)$ is undefined and $e(Y) = e_X(Y)$ for all $Y \neq X$

Then

$$[\![p]\!]e = \{s \in S \mid p \in \ell(s)\}.$$

$$[\![X]\!]e = e(X)$$

$$[\![\neg\varphi]\!]e = S - [\![\varphi]\!]e$$

$$[\![\varphi_1 \wedge \varphi_2]\!]e = [\![\varphi_1]\!]e \cap [\![\varphi_2]\!]e$$

$$[\![\varphi_1 \vee \varphi_2]\!]e = [\![\varphi_1]\!]e \cup [\![\varphi_2]\!]e$$

$$[\![\langle a \rangle \varphi]\!]e = \{s \in S \mid \exists t : s \xrightarrow{a} t \wedge t \in [\![\varphi]\!]e\}$$

$$[\![[a]\varphi]\!]e = \{s \in S \mid \forall t : s \xrightarrow{a} t \Rightarrow t \in [\![\varphi]\!]e\}.$$

$[\![\mu[X = \varphi]]\!]e$ = the least solution of the equation $X = [\![\varphi]\!]e_X$

$[\![\nu[X = \varphi]]\!]e$ = the greatest solution of $X = [\![\varphi]\!]e_X$

provided $[\![\varphi]\!]e_X : 2^S \rightarrow 2^S$ is a monotonic function of X.

Suppose $[\![\varphi]\!]e_X : 2^S \to 2^S$ is monotonic in X.

Then

$$[\![\mu[X=\varphi]]\!]e = \bigcap \{T \subseteq S \mid [\![\varphi]\!]e_X(T) \subseteq T\}$$

$$[\![\nu[X=\varphi]]\!]e = \bigcup \{U \subseteq S \mid [\![\varphi]\!]e_X(U) \supseteq U\}.$$

In particular if S is finite and the LTS is finite then we have

$$[\![\mu[X=\varphi]]\!]e = \bigcup_{i \geq 0} ([\![\varphi]\!]e_X)^i(\{\})$$

$$[\![\nu[X=\varphi]]\!]e = \bigcap_{i \geq 0} ([\![\varphi]\!]e_X)^i(S)$$

provided the LTS is finite

Facts (Exercise).

1.a) Show that $\neg$ is antimonotone i.e. show that for any predicate transformer $f: \mathbb{Z}^S \to \mathbb{Z}^S$ which is monotone, $\bar{f}(X) = S - f(X)$ is antimonotone i.e.

$$T \subseteq U \Rightarrow \bar{f}(T) \supseteq \bar{f}(U)$$

b) Show that $\tilde{f}(X) = S - f(S-X)$ is monotone.

2. Show that $\wedge, \vee$, $\langle a \rangle$ and $[a]$ are monotone operators.

3. Show that if φ is any formula constructed from the language

$$\varphi ::= p \mid X \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi \mid \langle a \rangle \varphi \mid [a]\varphi$$

then φ is monotone provided every variable occurs within an even number of nestings of $\neg$.

4. Show that the usual duality laws hold. i.e.

$$\neg[a]\varphi \Leftrightarrow \langle a \rangle \neg\varphi$$
$$\neg\langle a \rangle \varphi \Leftrightarrow [a]\neg\varphi$$
$$\neg \mu X[\varphi] \Leftrightarrow \nu X[\varphi\{\neg X/X\}]$$
$$\neg \nu X[\varphi] \Leftrightarrow \mu X[\neg\varphi\{\neg X/X\}].$$

provided syntactic monotonicity is ensured in the last two cases.

Examples of fixpoint equations over finite structures.

Example 1 $X = \langle a \rangle X$

Assuming a finite set S of states we may use iterative methods to compute the least and greatest fixpoints.

Consider the empty set then the monotonic function $f: 2^S \to 2^S$ defined by $f = [\![\langle a \rangle]\!] e_X$. We have

$$f\{\} = \{s \in S \mid \exists t : s \xrightarrow{a} t \wedge t \in \{\}\} = \{\}$$

Hence $\{\}$ is a fixpoint of f and must be the least.

Hence $\boxed{[\![\mu[X = \langle a \rangle X]]\!] e = \{\}}$

Now consider

$$f(S) = \{s \in S \mid \exists t : s \xrightarrow{a} t \wedge t \in S\} = \{s \in S \mid s \xrightarrow{a}\} = S_1$$

Clearly $f(S) \subseteq S$.

$$f^2(S) = \{s \in S \mid \exists t : s \xrightarrow{a} t \wedge t \in f(S)\}$$
$$= \{s \in S \mid s \xrightarrow{a} \xrightarrow{a}\} = S_2.$$

Since any state from which there are two consecutive a-transitions possible is also a state from which there is one a-transition possible

It is clear that $S \supseteq S_1 \supseteq S_2$

Generalizing we get

$$f^i(S) = \{s \in S \mid s (\xrightarrow{a})^i\} = S_i \supseteq S_{i+1}$$

and $\bigcap S_i = \{s \in S \mid s (\xrightarrow{a})^\omega\}$. Hence

$$\boxed{[\![\nu[X = \langle a \rangle X]]\!] e = \{s \in S \mid s (\xrightarrow{a})^\omega\}.}$$

Example 2. $X = [a]X$

$f = [\![[a]]\!] e_X$ and

$$f\{\} = \{ s \in S \mid \forall t : s \xrightarrow{a} t \Rightarrow t \in \{\} \} = \{ s \in S \mid s \not\xrightarrow{a} \} = S_1$$

We have $\{\} \subseteq S_1$

$$f(S_1) = \{ s \in S \mid \forall t : s \xrightarrow{a} t \Rightarrow t \in S_1 \} = S_2$$

$$= S_1 \cup \{ s \in S \mid \forall t : s \xrightarrow{a} t \Rightarrow t \not\xrightarrow{a} \}.$$

Generalizing we get

$$f^i(S) = S_{i-1} \cup \{ s \in S \mid \forall t : s \xrightarrow{a}{}^{i} t \Rightarrow t \not\xrightarrow{a} \}.$$

Hence $\bigcup_{i \geq 0} S_i$

$$= \{ s \in S \mid \not\exists \text{ an infinite sequence of } a\text{-moves from } s \}$$

$$= [\![\mu [X = [a]X]]\!] e$$

Similarly

$$f(S) = \{ s \in S \mid \forall t : s \xrightarrow{a} t \Rightarrow t \in S \}.$$

$$= \{ s \in S \mid s \not\xrightarrow{a} \} \cup \{ s \in S \mid s \xrightarrow{a} \} = S$$

It is clear that S is a fixpoint of f and hence must also be the greatest

$$[\![\nu [X = [a]X]]\!] e = S$$

Example 3 $\quad X = \langle a \rangle X \wedge \langle b \rangle \mathit{true}$

In Set theoretic terms the right-hand side expression represents the function

$$f(T) = \{ s \in S \mid \exists t : s \xrightarrow{a} t \wedge t \in T \} \cap \{ s \in S \mid \exists u : s \xrightarrow{b} u \wedge u \in S \}$$

$$= \{ s \in S \mid s \xrightarrow{b} \wedge \exists t : s \xrightarrow{a} t \wedge t \in T \}.$$

Consider

$$f(\{\}) = \{ s \in S \mid s \xrightarrow{b} \wedge \exists t : s \xrightarrow{a} t \wedge t \in \{\} \} = \{\}.$$

Hence $\{\}$ is a fixpoint of f and is the least.

Intuitively this is reasonable because $\{\}$ is also the least fixpoint of $X = \langle a \rangle X$ and clearly $\mu[X = \langle a \rangle X \wedge \langle b \rangle \mathit{true}] \Rightarrow \mu[X = \langle a \rangle X]$

Consider

$$f(S) = \{ s \in S \mid s \xrightarrow{b} \wedge \exists t : s \xrightarrow{a} t \wedge t \in S \} = S_1 = \{ s \in S \mid s \xrightarrow{a} \wedge s \xrightarrow{b} \}$$

$$f(S_1) = \{ s \in S \mid s \xrightarrow{b} \wedge \exists t : s \xrightarrow{a} t \wedge t \in S_1 \}$$

$$= \{ s \in S \mid s \xrightarrow{b} \wedge \exists t : s \xrightarrow{a} t \wedge t \xrightarrow{a} \wedge t \xrightarrow{b} \} = S_2$$

Clearly $\quad S \supseteq S_1 \supseteq S_2$

Generalizing we have

$$f(S_i) = \{ s \in S \mid s \xrightarrow{b} \wedge \exists t : s \xrightarrow{a} t \wedge t \in S_i \}$$

and $\bigcap_{i \geq 0} S_i$ is the set of states such that

(i) every state in the set has a b-transition from it and

(ii) from every state there is also an a-transition to another state in the set.

The set $[\![\nu [X = \langle a \rangle X \wedge \langle b \rangle true]\!] e$ is the set of states that

(i) form a strongly connected component with respect to $\xrightarrow{a}$ (an $\xrightarrow{a}$ SCC)

(ii) every state in the $\xrightarrow{a}$ SCC also has a b-transition which may or may not lead out of the $\xrightarrow{a}$ SCC.

Example 4. Consider

$$X = \langle a \rangle X \vee \langle b \rangle \underline{true}$$

Consider the expression

$$\langle a \rangle \{\} \vee \langle b \rangle S = \{ s \in S \mid \exists t: s \xrightarrow{a} t \wedge t \in \{\} \}$$
$$\cup \{ s \in S \mid \exists t: s \xrightarrow{b} t \wedge t \in S \}$$
$$= \varphi \cup \{ s \in S \mid s \xrightarrow{b} \}. = \{ s \in S \mid s \xrightarrow{b} \}.$$

$$\langle a \rangle (\langle a \rangle \{\} \vee \langle b \rangle S) \vee \langle b \rangle S$$
$$= \{ s \in S \mid \exists t: s \xrightarrow{a} t \wedge t \in \langle a \rangle \{\} \vee \langle b \rangle S \}$$
$$\cup \{ s \in S \mid s \xrightarrow{b} \}$$
$$= \{ s \in S \mid s \xrightarrow{b} \} \cup \{ s \in S \mid s \xrightarrow{a} \xrightarrow{b} \}.$$

In the i-th iteration we get

$$f^i(\{\}) = \{ s \in S \mid \exists t: s \xrightarrow{a}{}^{j} \xrightarrow{b}, \; j \leq i \}.$$

and $[\![\mu [X = \langle a \rangle X \vee \langle b \rangle \underline{true}]]\!]$ is the set of states from which after a finite number of a-transitions a b-transition is possible.

To compute the greatest fixpoint we proceed as follows:

$$\langle a\rangle S \vee \langle b\rangle S = \{ s \in S \mid s \xrightarrow{a} \vee\, s \xrightarrow{b} \}.$$

$$\langle a\rangle(\langle a\rangle S \vee \langle b\rangle S) \vee \langle b\rangle S = \{ s \in S \mid s \xrightarrow{a} \xrightarrow{a} \vee\, s \xrightarrow{a} \xrightarrow{b} \} \cup \{ s \in S \mid s \xrightarrow{b} \}.$$

After the i-th iteration we get

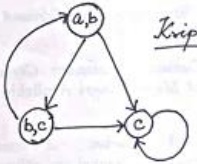
$$\{ s \in S \mid s (\xrightarrow{a})^{i} \} \cup \{ s \in S \mid s (\longrightarrow)^{j} \xrightarrow{b}, \; j < i \}.$$

Hence $[\![\nu [X = \langle a\rangle X \vee \langle b\rangle \underline{true}\,]]\!]$ is the set of states from which a finite or infinite number of a-transitions are possible and those states from which after a finite number of a-transitions a b-transition is possible.

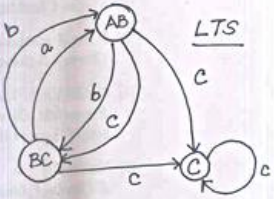
2. CTL & LTL

The Logic CTL*

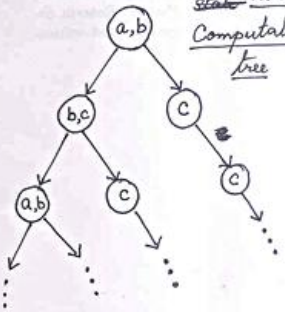
- CTL* formulas describe properties of computation trees. Given a Kripke structure and an initial state the unwinding of the structure into a tree (infinite) with the initial node as the root is the computation tree defined by the initial state



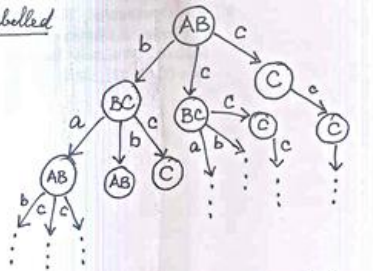
Kripke Structure



LTS



~~state~~ node-labelled Computation tree



edge-labelled computation tree

- Primarily concerned with node-labelled computation trees, which have two kinds of modalities. A formula of CTL* may be either
  - a path formula describing properties of states in a particular path, or
  - a state formula with quantifies over all paths emanating from that state

  using temporal operators X, U, R

  using path quantifiers A, E

- Syntax of CTL*

  Assume a collection AP of atomic propositions. Then the language of CTL* is defined as

  State formulas $\psi ::= p \mid \neg\psi \mid \psi \wedge \psi \mid E\varphi$

  Path formulas $\varphi ::= \psi \mid \neg\varphi \mid \varphi \wedge \varphi \mid X\varphi \mid \varphi \, U \, \varphi$

  Every state formula is also a path formula

  path quantified

  temporal operators

  (2)

- **Derived operators** (duals too)

1) $\chi_1 \vee \chi_2 \stackrel{df}{=} \neg(\neg\chi_1 \wedge \neg\chi_2)$ where χ_1, χ_2 are either state or path formulas.

2) $F\varphi \stackrel{df}{=} true\ U\ \varphi$

3) $G\varphi \stackrel{df}{=} \neg F \neg\varphi$

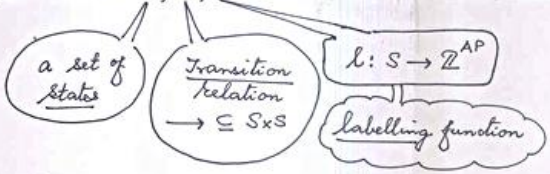
4) $A\varphi \stackrel{df}{=} \neg E \neg\varphi$

5) $\varphi_1 R \varphi_2 \stackrel{df}{=} \neg(\neg\varphi_1\ U\ \neg\varphi_2)$

6) X is its own dual (self-dual)
(Check it out!)
i.e. $X\varphi \Leftrightarrow \neg X \neg\varphi$

- **Semantics of CTL***

Let $\mathcal{K} = \langle S, \rightarrow, \ell \rangle$ be a Kripke structure.



Note: 1) Since $\mathcal{K}$ is a Kripke structure $\rightarrow$ must total. Hence since S is finite, there must be cycles in the graph.

Note 2. Since $\rightarrow$ is total every node has an outgoing edge. Hence every computation tree has only infinite maximal paths — there are no finite length maximal paths.

- **Semantics of CTL*** (Contd.)

Let $\pi = s_0, s_1, \ldots$ be a path $(\forall i \geq 0 : s_i \in S)$

$$\pi^i = s_i, s_{i+1}, \ldots = \pi(s_i)$$

is the suffix of π starting from $s_i \in S$.

$$\pi^0 = \pi = \pi(s_0)$$

State formulas. Given any $s \in S$ in $\mathcal{K}$

$\mathcal{K}, s \models p$ iff ~~$L(p)$~~ $p \in l(s)$

$\mathcal{K}, s \models \neg\psi$ iff $\mathcal{K}, s \not\models \psi$

$\mathcal{K}, s \models \psi_1 \wedge \psi_2$ iff $\mathcal{K}, s \models \psi_1$ and $\mathcal{K}, s \models \psi_2$

$\mathcal{K}, s \models E\varphi$ iff $\exists$ a path $\pi(s) : \mathcal{K}, \pi(s) \models \varphi$

φ is a path formula

Path formulas. Given any path π in $\mathcal{K}$, $\pi = s_0, s_1, s_2, \ldots$

$\mathcal{K}, \pi \models \psi$ iff $\mathcal{K}, s_0 \models \psi$

(ψ is a state formula!)

$\mathcal{K}, \pi \models \neg\varphi$ iff $\mathcal{K}, \pi \not\models \varphi$

$\mathcal{K}, \pi \models \varphi_1 \wedge \varphi_2$ iff $\mathcal{K}, \pi \models \varphi_1$ and $\mathcal{K}, \pi \models \varphi_2$

$\mathcal{K}, \pi \models X\varphi$ iff $\mathcal{K}, \pi^1 \models \varphi$

$\mathcal{K}, \pi \models \varphi_1 \cup \varphi_2$ iff $\exists k \geq 0 [\ \mathcal{K}, \pi^k \models \varphi_2$ and $\forall j < k: \mathcal{K}, \pi^j \models \varphi_1\ ]$

- **CTL: a sublogic of CTL***

CTL is the sublogic in which every temporal operator X, U, F, G, R must be immediately preceded by a path quantifier. (to make it a state formula)

Hence the language of CTL is the 2-level one defined as

$\psi ::= p \mid \neg\psi \mid \psi \wedge \psi \mid EX\varphi \mid E[\varphi \cup \varphi] \mid EG\varphi$ ($p \in AP$)

$\varphi ::= \psi \mid \neg\varphi \mid \varphi \wedge \varphi$

Note the presence of $EG\,\varphi$ (!) in the language.

This operator EG cannot be expressed in pure CTL

since $EG\,\varphi \stackrel{df}{=} E(\neg F \neg \varphi)$ in CTL*

The presence of this negation between E and F violates the condition of F being immediately preceded by a path quantifier

The semantics may be taken from that of CTL*.

The derived operators of CTL.

$AX\,\varphi \stackrel{df}{=} \neg EX \neg \varphi$

$AF\,\varphi \stackrel{df}{=} \neg EG \neg \varphi$

$AG\,\varphi \stackrel{df}{=} \neg EF \neg \varphi$

$EF\,\varphi \stackrel{df}{=} E[true\ U\ \varphi]$

lookit!

$$A[\varphi_1 U \varphi_2] \stackrel{df}{=} \neg E[\neg\varphi_2\ U\ (\neg\varphi_1 \wedge \neg\varphi_2)] \wedge \neg EG \neg\varphi_2$$

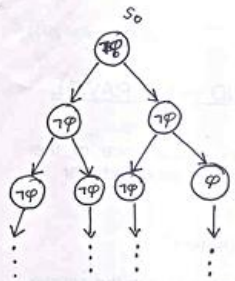
$$A[\varphi_1 R \varphi_2] \stackrel{df}{=} \neg E[\neg\varphi_1 U \neg\varphi_2]$$

$$E[\varphi_1 R \varphi_2] \stackrel{df}{=} \neg A[\neg\varphi_1 U \neg\varphi_2]$$

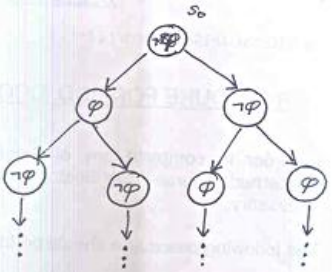
The four most widely used operators

EG , EF

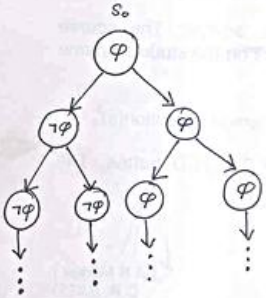
AG , AF



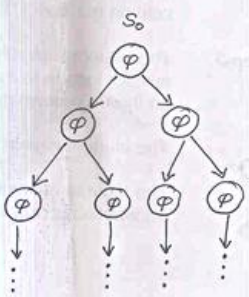
$\mathcal{K}, s_0 \models EF\,\varphi$



$\mathcal{K}, s_0 \models AF\,\varphi$



$\mathcal{K}, s_0 \models EG\,\varphi$



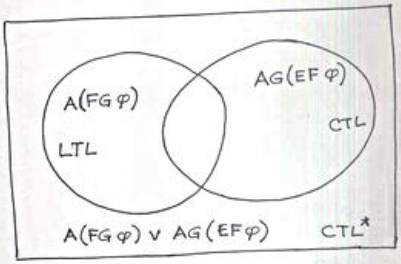
$\mathcal{K}, s_0 \models AG\,\varphi$.

(7)

LTL: a sublogic of CTL*

LTL is the sublogic consisting of only formulas of the form $A\varphi$ where φ is a path formula. In other words LTL is defined ~~the~~ as ~~the logic defined~~

$$\psi ::= A\varphi$$
$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid X\varphi \mid \varphi U \varphi$$



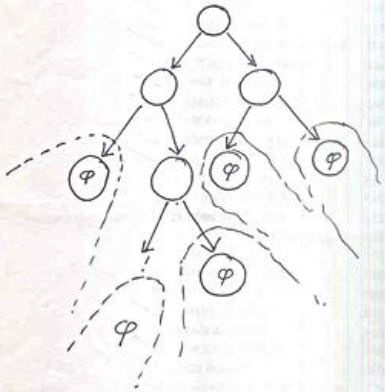
Differences between LTL & CTL

- There is no CTL formula that can express the LTS property

$$A(FG\varphi)$$

i.e. along all paths π
eventually φ holds in all states of the path

i.e. $\forall \pi : \exists k \geq 0 : \forall m \geq k : \quad \mathcal{K}, \pi(s_m) \models \varphi$

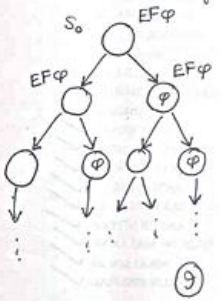


A(FG φ)

- There is no LTL formula that can express the CTL formula AG(EF φ)

  i.e. along every path in every state EFφ holds

  i.e. along every path, ~~in~~ starting from every state, ~~starting~~ there exists a path such that eventually φ holds.



In other words any subtree of the computation starting at s_0 contains a branch in which at some point φ is true

- The CTL* formula $A(FG\varphi) \vee AG(EF\varphi)$ is expressible neither in LTL nor in CTL.

Consider $\neg A(FG\varphi)$. This formula is satisfied by any computation tree that satisfies

$\neg A(FG\varphi) \Leftrightarrow E(GF\neg\varphi)$.

i.e. any computation tree in which there exists a path in which between there are only finite maximal sequences of states in which φ is true.

Similarly $\neg AG(EF\varphi)$

$\Leftrightarrow EF(AE\neg\varphi)$

is satisfied by a computation tree in which there exists a path such that somewhere along the path there exists a computation tree in which φ does not hold anywhere.

i.e. there exists a computation subtree in which φ never holds.

Questions 1. Is $EF\varphi$ expressible in LTL in some form? i.e. Does it require something as complicated as $AG(EF\varphi)$ to show that CTL is not contained in LTL?

2. Can the above intuition be used to show that CTL* is more powerful (10) than either LTL or CTL?

Model Checking in CTL

- Any CTL formula ψ can be expressed using the operators

$$\neg, \vee, EX, EU, EG$$

- We consider only formulas ψ which employ these operators.
- Assume given a Kripke structure

$$\mathcal{K} = \langle S, \rightarrow, \ell \rangle$$

an arbitrary state $s \in S$, and a CTL formula ψ. We need to determine whether

$$\mathcal{K}, s \models \psi.$$

The model-checking problem uses induction on the structure of ψ. With each $s \in S$ associate a set labels(s) such that

$$\text{labels}(s) = \ell(s) \cup \{ \psi' \mid \psi' \text{ is a subformula of } \psi, \mathcal{K}, s \models \psi' \}.$$

The algorithm systematically constructs labels(s), so that

$$\mathcal{K}, s \models \psi \text{ iff } \psi \in \text{labels}(s)$$

Initially: $labels(s) = \ell(s)$ for each $s \in S$.

Algo: Assuming all subformulas of ψ have been checked for each state consider the root operator of ψ

Case $\psi \equiv \neg \psi_1$:

Check $\neg(\psi_1)$:: foreach $s \in S$

{ if $\psi_1 \notin labels(s)$

then $labels(s) := labels(s) \cup \{\psi\}$

}

Case $\psi \equiv \psi_1 \vee \psi_2$

Check $V(\psi_1, \psi_2)$ foreach $s \in S$

{

if $\psi_1 \in labels(s)$ or $\psi_2 \in labels(s)$

then $labels(s) := labels(s) \cup \{\psi_1 \vee \psi_2\}$.

}

Case $\psi \equiv EX\,\psi_1$.

Check $EX(\psi_1)$ foreach $s \in S$

{ if $\exists t \in S : s \rightarrow t \wedge \psi_1 \in labels(t)$

then $labels(s) = labels(s) \cup \{\psi\}$

}.

Case $\psi \equiv E[\psi_1 \cup \psi_2]$

CheckEU(ψ_1, ψ_2)

```
T := {t∈S | ψ2 ∈ labels(t)};
foreach t∈T { labels(t) := labels(t) ∪ {ψ} }
while T ≠ φ
{ choose t∈T; T := T − {t};
  foreach s∈S: s→t
  { if ψ ∉ labels(s) and ψ1 ∈ labels(s)
    then { labels(s) := labels(s) ∪ {ψ};
           T := T ∪ {s}
         }
  }
}
```

Note: In each of these cases the algorithm takes no more than $O(|S| + |\rightarrow|)$ time.

In the sequel we show that the algorithm for the case $\psi \equiv EG\psi_1$ also takes no more than that. The algorithm works by considering all subformulas of a given ψ. The total complexity of the algorithm therefore is

$$O(|\psi| . (|S| + |\rightarrow|))$$

- Checking for $\psi \equiv EG\,\psi_1$

— based on decomposition of graph into strongly connected components (SCC).

Def: Given a graph $\mathcal{K} = \langle S, \rightarrow, \ell \rangle$, a SCC is a maximal subgraph $\langle C, \rightarrow_C, \ell_C \rangle$ with $C \subseteq S$ such that every node in C is R (reachable) from every other node in C through a path passing only through nodes in C.

$\rightarrow_C = \rightarrow \cap (C \times C)$

$\ell_C = \ell|_C$

$\langle C, \rightarrow, \ell \rangle$ is nontrivial iff

either C has only a single node with a self loop.

or C has more than one node.

— Consider the subgraph of the Kripke structure $\mathcal{K}$, $\mathcal{K}' = \langle S', \rightarrow', \ell' \rangle$ where

$S' = \{ s \in S \mid \mathcal{K}, s \models \psi_1 \}$

with $\rightarrow'$, ℓ' appropriate restrictions.

$\mathcal{K}'$ need not be a Kripke structure

$\rightarrow'$ may not be total

But may be made one by eliminating the states with no outgoing transitions

$\mathcal{K}'' = \langle S'', \rightarrow'', \ell'' \rangle$

with $S'' = \{ s \in S' \mid s \rightarrow \neq \varphi \}$ would be a Kripke structure provided $S'' \neq \varphi$.

Lemma. $\mathcal{K}, s \models EG\,\psi_1$ iff the following conditions are satisfied

(i) $s \in S''$

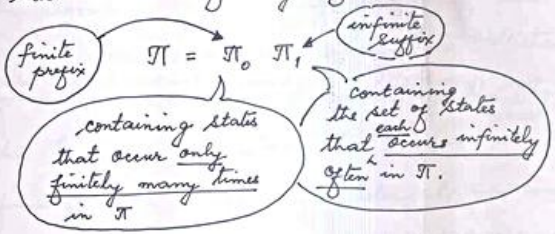
(ii) There exists a path in $\mathcal{K}''$: $s \xrightarrow{+} t$ for some t in a nontrivial SCC.

Proof: ($\Rightarrow$) Assume $\mathcal{K}, s \models EG\,\psi_1$ $\Rightarrow$ $s \in S'$ and there exists a path π in $\mathcal{K}$

$\pi : s = s_0 \to s_1 \to s_2 \to \ldots$

such that for every s_i, $\psi_1 \in labels(s_i)$.

Hence $\{s_i \mid i \geq 0\} \subseteq S'$. Since S is finite there must be a finite subset of states from S' that occur infinitely often in π. Hence



Then clearly $C = inf(\pi_1)$ is contained in some SCC. of $\mathcal{K}$. Further $C \subseteq S''$ and there exists a path $s \to^{+} t$ where $t \in C$.

($\Leftarrow$) Assume (i) & (ii) are satisfied. Then there exists a path $\pi_0 : s \to^+ t$ to some t in a nontrivial SCC of K'. Since t is in a non-trivial SCC there exists a path $\pi_1 : t \to^+ t$. Since $s \in S'$ and $t \in S'$ and π_0 is contained in K' and π_1 is also contained in S', we have that every state in π_0 and π_1 satisfies ψ_1. i.e. $\psi_1 \in$ labels (u) for every state u in π_0 and π_1. Now consider the infinite path $\pi = s \to^+ (t \to^+ t)^\omega$. Clearly then $K, s \models EG\psi_1$. $\dashv$

The algorithm for the case $\psi \equiv EG\psi_1$ then

1) requires partitioning S' into a collection of SCCs. which may be done by an algorithm due to Tarjan in time $O(|S'| + |\to'|)$.
2) finding those states that belong to non-trivial SCCs.
3) working backwards from those states in to find paths ~~that~~ in which each state satisfies ψ_1. This entire computation may be done in $O(|S| + |\to|$

Case $\psi \equiv EG\,\psi_1$.

CheckEG(ψ_1) $S' := \{ s \in S \mid \psi_1 \in labels(s) \}$;

$SCC := \{ C \subseteq S' \mid C \text{ is a nontrivial SCC} \}$;

$T := \bigcup SCC$;

foreach $t \in T$ { $labels(t) := labels(t) \cup \{\psi\}$ };

while $T \neq \varphi$

{ choose $t \in T$; $T := T - \{t\}$;

foreach $s \in S'$: $s \longrightarrow t$

{ if $\psi \notin labels(s)$

then { $labels(s) := labels(s) \cup \{\psi\}$;

$T := T \cup \{s\}$

}

}

}

Theorem: $\mathcal{K}, s \models \psi$ may be checked in time

$$O(|\psi| \cdot (|S| + |\rightarrow|))$$

⊣.

Fairness Constraints

- We are often interested in fair computations only.
- Each fairness constraint $\overset{F}{\ell}$ defines a subset of the states, $F \subseteq S$.
- A fair computation is one in which there are an infinite number of occurrences of some state in each fairness constraint. This is the generalized Büchi acceptance condition.
- A fair Kripke structure $\mathcal{K}_F = \langle S, \to, \ell, \mathcal{F} \rangle$ is just a Kripke structure $\mathcal{K} = \langle S, \to, \ell \rangle$ with $\mathcal{F} \subseteq 2^S$ being the fairness constraints.
- The effect of a fairness constraint is to eliminate all those paths in $\mathcal{K}$ which may be unfair in some way for the purpose of satisfaction of state formulas.
- Satisfaction under fairness. For any path formula φ and state formula ψ we define

  $\mathcal{K}_F, s \models_F \psi$ means ψ is true of s in a fair Kripke structure

  $\mathcal{K}_F, \pi \models_F \varphi$ means $\mathcal{K}, \pi \models \varphi$

- CTL* semantics under fairness.

$\mathcal{K}_F, s \models_F p$ iff there does exist a fair computation starting from s, and $p \in \ell(s)$.

$\mathcal{K}_F, s \models_F E\varphi$ iff there exists a fair path π_F starting from s such that $\mathcal{K}_F, \pi_F \models_F \varphi$.

$\mathcal{K}_F, s \models_F A\varphi$ iff for all fair paths π_F starting from s, $\mathcal{K}_F, \pi_F \models_F \varphi$.

All other clauses have their semantics "textually" unchanged.

- Fair SCCs

Let $\mathcal{F} = \{F_1, \ldots, F_k\} \subseteq 2^S$ and let $C \subseteq S$ be a SCC of a fair Kripke structure $\mathcal{K}_F$. Then C is fair iff $C \cap F_i \neq \varphi$ for each $1 \leq i \leq k$.

- Checking $EG\psi_1$ under fairness constraints

– As before construct $\mathcal{K}_F'$ with

$$S' = \{ s \in S \mid \mathcal{K}_F, s \models_F \psi_1 \}$$

S' consists of only those states fairly satisfying ψ_1

and $\mathcal{F}' = \{ F_i \cap S' \mid 1 \leq i \leq k \} = \{F_i' \mid 1 \leq i \leq k\}$.

Lemma 2. $\mathcal{K}_F, s \models_F EG\ \psi_1$ iff

(i) $s \in S'$ and

(ii) $\exists \pi : s \longrightarrow^+ t$ where

a) π consists of states only from S'

b) t belongs to a non-trivial fair SCC of $\mathcal{K}'_F$. ⊣

- Model Checking CTL under fairness

– CheckFair $EG(\psi_1)$ same as CheckEG (ψ_1)

with SCC changed to FairSCC.

To determine which SCCs are fair it is necessary to check membership $C \cap F_i$ for each $C \in SCC$ and each $1 \le i \le k$. The complexity of the resulting algorithm is $O(|\mathcal{F}| \cdot (|S| + |\rightarrow|))$.

– Checking other cases requires that there be fair computations ~~em~~. starting from the given state.

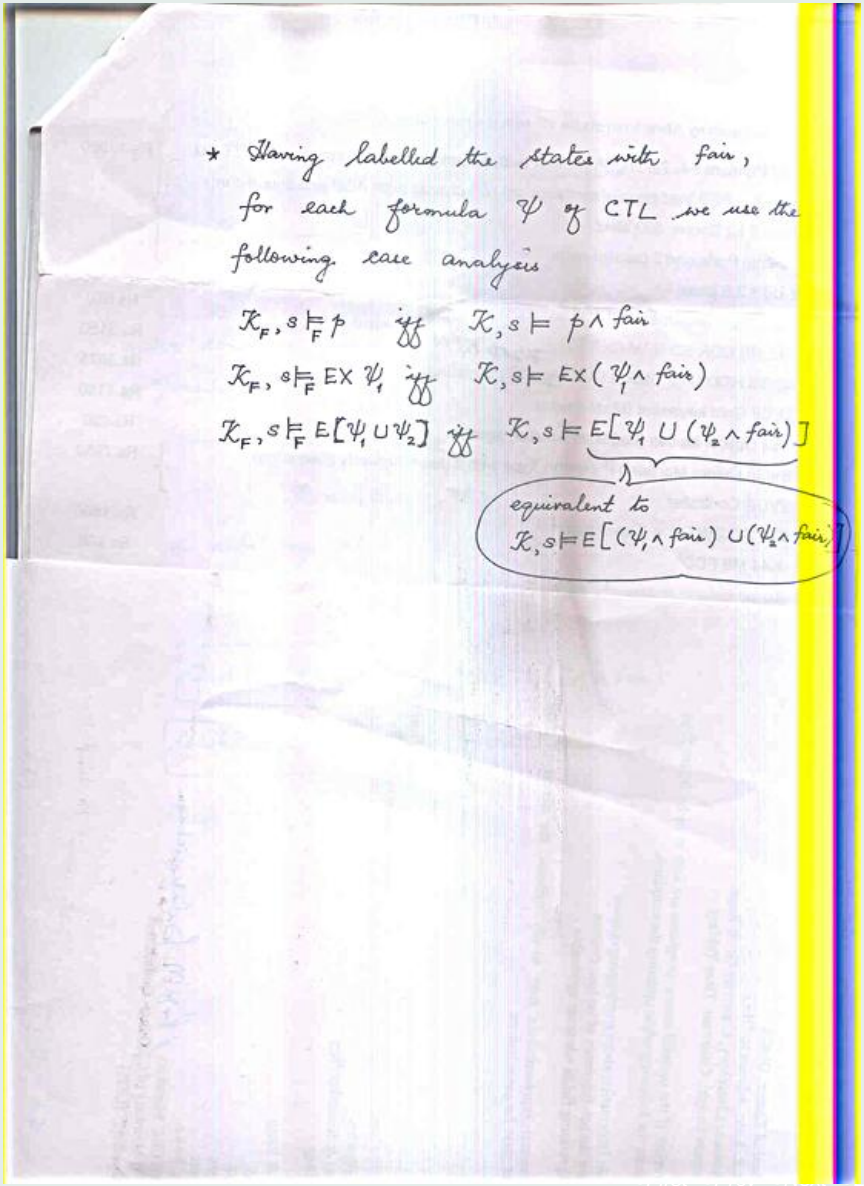
* Introduce a new atomic proposition

fair

* The formula $EG\ true$ may be used

i.e. call procedure CheckFair $EG(true)$

to label states with the proposition fair.


★ Having labelled the states with fair,
for each formula ψ of CTL we use the
following case analysis
$\mathcal{K}_F, s \models_F p$ iff $\mathcal{K}, s \models p \wedge fair$
$\mathcal{K}_F, s \models_F EX\,\psi_1$ iff $\mathcal{K}, s \models EX(\psi_1 \wedge fair)$
$\mathcal{K}_F, s \models_F E[\psi_1 \cup \psi_2]$ iff $\mathcal{K}, s \models E[\psi_1 \cup (\psi_2 \wedge fair)]$
equivalent to
$\mathcal{K}, s \models E[(\psi_1 \wedge fair) \cup (\psi_2 \wedge fair)]$

CTL* Model Checking

The LTL model-checking problem is to determine

$$\mathcal{K}, s \models A\varphi \quad \Leftrightarrow \quad \mathcal{K}, s \models \neg E \neg \varphi$$

for any path formula φ in the language

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid X\varphi \mid \varphi U \varphi.$$

This is done by checking for emptiness of the language in $\mathcal{K} \otimes \neg\varphi$.

Hence in general, LTL model checking may be used for determining whether

$$\mathcal{K}, s \models E\varphi.$$

where φ is the form given above. where

- the only state subformulae in φ are atomic propositions.

This algorithm may be extended to CTL* model checking as follows:

1) Define new atomic propositions corresponding to each state subformula. ψ.
2) Replace the state subformulas by their atomic proposition counterparts. Let the new formula be $E\varphi'$.

If $E\varphi'$ is a CTL formula then perform CTL model-checking to determine $\mathcal{K}, s \models E\varphi'$
Otherwise φ' is a pure path formula from LTL and the algorithm for LTL model-checking is used. Note that from the definition of CTL*

$$\psi ::= p \mid \psi \vee \psi \mid \neg\psi \mid E\varphi$$
$$\varphi ::= \psi \mid \varphi \vee \varphi \mid \neg\varphi \mid X\varphi \mid \varphi U \varphi$$

- every state formula is also a path formula but not vice-versa.
- The quantifiers E and A are applied to a path formula φ which implies that either

(i) the root of φ is X or U and hence we have the CTL operators EX and EU

or (ii) the root operator is $\vee$ or $\neg$ and the subformula under it is a path formula

Recall that LTL is defined by the BNF

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid X\varphi \mid \varphi U \varphi$$
$$\psi ::= p \mid A\varphi$$

CTL* model-checking algorithm

For any formula of CTL*, the state subformulas at each level are defined inductively as follows:

- level 0. contains all atomic propositions
- level $i+1$. contains all state subformulas ψ such that all state subformulas of ψ are of level $\leq i$. and ψ is not contained in any lower level.

Example. $AG[(\neg p \wedge q) \rightarrow A(G\neg r \vee F\neg s)]$
$\Leftrightarrow \neg EF[\neg p \wedge q \wedge E(Fr \wedge Gs)]$

level 0: p, q, r, s.

level 1: $E(Fr \wedge Gs), \neg p$

level 2. $EF[\neg p \wedge q \wedge E(Fr \wedge Gs)]$

level 3. $\neg EF[\neg p \wedge q \wedge E(Fr \wedge Gs)]$

Def: For any f in CTL* a subformula $E\varphi$ is maximal iff $E\varphi$ is not a proper subformula of any other subformula of f.

Let f be any CTL* formula. Let ψ be a state subformula of f of level i. Assume S has already been labelled with all subformulas of level $< i$.

At stage i. ψ is added to all those states which satisfy ψ.

Case $\psi \equiv p$. Then $\psi \in labels(s)$ iff $\psi \in \ell(s)$

Case $\psi \equiv \neg\psi_1$. Then $\psi \in labels(s)$ iff $\psi_1 \notin labels(s)$

Case $\psi \equiv \psi_1 \vee \psi_2$. Then $\psi \in labels(s)$ iff $\{\psi_1, \psi_2\} \cap labels(s) \neq \{\}$.

Case $\psi \equiv E\psi_1$. Execute procedure Check (ψ')

procedure Check (ψ) ($E\psi_1$)

if ψ is a CTL formula

then apply CTL model-checking for ψ.;

$\psi' = \psi[a_1/E\psi_1, \ldots, a_k/E\psi_k]$ (new atomic propositions; maximal subformulas of ψ)

foreach $s \in S$.

{ foreach $i : 1..k$

{ if $E\psi_i \in labels(s)$ then $labels(s) \cup = \{a_i\}$ };

apply LTL modelchecking for ψ'; ($E\psi_1'$, ψ_1' is a pure LTL path formula)

foreach $s \in S$

{ if $\psi' \in labels(s)$ then $labels(s) \cup = \{\psi\}$.

It is not clear why it is a pure LTL path formula.
It is also not clear why CTL model-checking cannot be applied to it if ψ' is a CTL-formula

(4)

Example:

level 0. The atomic propositions p, q, r, s are handled

level 1. The formula $\neg p$ is handled. Then

level 2. $E(Fr \wedge Gs)$ is handled as a pure LTL formula and added to the labels of each state that it satisfies.

level 2 $EF[\neg p \wedge q \wedge a]$. Again an LTL modelchecking procedure is adopted and the appropriate states are labelled with this.

level 3. $\neg EF[\neg p \wedge q \wedge a]$ is handled by the negation rule.

Presumably because it says so in the algorithm. However why can't CTL model-checking be used here?

3. The μ-calculus & CTL

Home Page
Title Page
◀◀ ▶▶
◀ ▶
Page 54 of 73
Go Back
Full Screen
Close
Quit

μ-calculus and CTL

For the purpose of modelling Kripke structures we assume that the set Act is a singleton and that the transition relation is total with respect to this single action (Hence it is not necessary to label the transitions).

Now consider the semantics of CTL formulas expressed as sets of states satisfying the formula.
i.e. $[\![\psi]\!] = \{s \in S \mid \mathcal{K}, s \models \psi\}$.

We know that $EF\ \psi_1 \Leftrightarrow \psi_1 \vee EX(EF\ \psi_1)$ and

$AF\ \psi_1 \Leftrightarrow \psi_1 \vee AX(AF\ \psi_1)$

which are solutions of as fixpoints

$$Z = \psi_1 \vee AX\ Z$$

Because of F we require solutions in which ψ_1 becomes true after a finite number of transitions.
i.e. it cannot happen that there exists an infinite computation in which ψ_1 never becomes true.

Hence

$$\boxed{AF\ \psi_1 \Leftrightarrow \mu[Z = \psi_1 \vee AX\ Z]} \text{ and}$$
$$\boxed{EF\ \psi_1 \Leftrightarrow \mu[Z = \psi_1 \vee EX\ Z]}$$

Similarly we may reason with the other CTL operators to obtain the following translations into the μ-calculus.

$$AG\,\psi_1 \Leftrightarrow \nu[Z = \psi_1 \wedge AX\,Z]$$
$$EG\,\psi_1 \Leftrightarrow \nu[Z = \psi_1 \wedge EX\,Z]$$

For the until and release operators we have the following translations

$$A[\psi_1 \,U\, \psi_2] \Leftrightarrow \mu[Z = \psi_2 \vee (\psi_1 \wedge AX\,Z)]$$
$$E[\psi_1 \,U\, \psi_2] \Leftrightarrow \mu[Z = \psi_2 \vee (\psi_1 \wedge EX\,Z)]$$

$$A[\psi_1 \,R\, \psi_2] \Leftrightarrow \nu[Z = \psi_2 \wedge (\psi_1 \vee AX\,Z)]$$
$$E[\psi_1 \,R\, \psi_2] \Leftrightarrow \nu[Z = \psi_2 \wedge (\psi_1 \vee EX\,Z)]$$

Intuition:

1. Least fixpoints denote eventuality properties
2. Greatest fixpoints denote properties that should always hold (forever).

Exercises.

1. Prove that AX and EX are monotonic operators.
2. Explain what sets of states are denoted by

$$\nu[Z = \psi_1 \vee AX Z]$$

and $\nu[Z = \psi_1 \vee EX Z]$

Signify? Why aren't they interesting?

3. Similarly find out the significance of

$$\mu[Z = \psi_1 \wedge AX Z]$$

and $\mu[Z = \psi_1 \wedge EX Z]$

4. What are the equations governing the Weak-until operator

Theorem. $EG\,\psi_1$ is the greatest fixpoint of the equation

$$Z = \psi_1 \wedge EX\,Z$$

Proof: **Claim 1.** $f(Z) = \psi_1 \wedge EX\,Z$ is monotonic in Z.

⊢ Consider $T_1 \subseteq T_2 \subseteq S$. and let $s_1 \in f(T_1)$.

$$f(T_1) = \{s \in S \mid s \models \psi_1 \text{ and } \exists t : s \to t \wedge t \in T_1\}.$$

$$f(T_2) = \{s \in S \mid s \models \psi_1 \text{ and } \exists t : s \to t \wedge t \in T_2\}.$$

Since $s_1 \in f(T_1)$, $s_1 \models \psi_1$ and $\exists t_1 : s_1 \to t_1 \wedge t_1 \in T_1 \subseteq T_2$ and hence $t_1 \in T_2$ and $s_1 \in f(T_2)$. ⊣.

Claim 2: f is $\cap$-continuous.

Claim 3. $EG\,\psi_1 = \bigcap_{i \geq 0} f^i(S)$

$$f(S) = \{s \in S \mid s \models \psi_1 \wedge \exists t : s \to t \wedge t \in S\}.$$

Since the Kripke structure is total, we have

$$f(S) = \{s \in S \mid s \models \psi_1\}. \subseteq S$$

$$f^2(S) = \{s \in S \mid s \models \psi_1 \wedge \exists t : s \to t \wedge t \models \psi_1\}.$$

Generalizing we get that

$$f^i(S) = \{s \in S \mid s \models \psi_1 \wedge \exists t_1, \ldots, t_{i-1} : s \to t_1 \to \ldots \to t_{i-1} \wedge t_1, t_2, \ldots, t_{i-1} \models \psi_1\}.$$

$$\bigcap_{i \geq 0} f^i(S) = \{s \in S \mid s \models \psi_1 \wedge \exists \text{ an infinite path from } s, \text{ all of whose states satisfy } \psi_1\}$$

$$= EG\,\psi_1$$ ⊣

Theorem. $E[\psi_1 \cup \psi_2]$ is the least fixpoint of the equation

$$Z = \psi_2 \vee (\psi_1 \wedge EX Z)$$

Proof: It is obvious that

$$f(Z) = \psi_2 \vee (\psi_1 \wedge EX Z)$$

is monotonic and is hence $\cup$-continuous.

Claim 1. $E[\psi_1 \cup \psi_2]$ is a fixpoint of f.

$$\vdash [\![E[\psi_1 \cup \psi_2]]\!] = \{s \in S \mid s \models \psi_2 \vee (s \models \psi_1 \wedge \exists t : s \to t \wedge t \models E[\psi_1 \cup \psi_2])\}.$$

$$= \{s \in S \mid s \models \psi_2 \vee (s \models \psi_1 \wedge \text{there is a finite path } s \to s_1 \to \dots \to s_n \text{ with } \forall i : s_i \models \psi_1 \text{ and } s_n \models \psi_2)\}.$$

$$= f([\![E[\psi_1 \cup \psi_2]]\!]) \quad \dashv.$$

Claim: $[\![E[\psi_1 \cup \psi_2]]\!] = \bigcup_{i \geq 0} f^i(\{\})$

Consider $f(\{\}) = \{s \in S \mid s \models \psi_2 \vee (s \models \psi_1 \wedge \exists t : s \to t \wedge t \in \{\})\}$

$$= \{s \in S \mid s \models \psi_2\} = T_1$$

and $f(T_1) = \{s \in S \mid s \models \psi_2 \vee (s \models \psi_1 \wedge \exists t \in T_1 : s \to t)\}$

$$= \{s \in S \mid s \models \psi_2 \vee (s \models \psi_1 \wedge \exists t : s \to t \wedge t \models \psi_2)\}$$

$$= T_2$$

It is easy to see that since $\{\} \subseteq T_1 \subseteq T_2$,

$$\{\} \subseteq f(\{\}) \subseteq f^2(\{\})$$

In general

$f^{i+1}(\{\}) = \{ s \in S \mid \exists$ a path of length $j \leq i$ to a state $\models \psi_2$ in which all the intermediate states $s_k \models \psi_1, k < j \}$.

Hence $\bigcup_{i \geq 0} f^i(\{\}) = \mu[Z = \psi_2 \vee (\psi_1 \wedge EX Z]$

is the set of all states $s \in S$ such that there exists a finite path $s \to^* t$ in which $t \models \psi_2$ and all states before t in the path satisfy ψ_1. ⊣

The above theorem tells us how to iteratively compute least fixpoints.

$T := \{s \in S \mid s \models \psi_2\}$; $\bigcup := \{s \in S \mid s \models \psi_1\}$;

repeat

foreach $s \in \bigcup$: $s \to t \in T$

{ if $s \notin T$ then $T := T \cup \{s\}$. }

until T is stable

i.e. no more new elements get added

Model Checking in the μ-calculus

Let f be a predicate transformer and let S be the finite set of states with $f: 2^S \to 2^S$.

```
function LFP (f);
{  T := {};  T' := f(T);
   while T ≠ T' do
   { T := T';  T' := f(T) };
   return T }
```

The complexity of each of these algorithms is $O(n)$ evaluations of f, where $n = |S|$

```
function GFP (f);
{  U := S;  U' := f(S);
   while (U ≠ U') do
   { U := U';  U' := f(S) };
   return U
}
```

We generalize these procedures for evaluating arbitrary μ-calculus formulas keeping in mind the following.

(i) the algorithm proceeds by induction on the structure of the formula and

(ii) Because of the nesting of fixpoint operators and the presence of free variables we need an environment.

function eval (f, e) — f: μ-calculus formula; e: an environment

{ Case $f \equiv p$ → return $\{s \mid p \in \ell(s)\}$.

Case $f \equiv Z$ → return $e(Z)$

Case $f \equiv g_1 \wedge g_2$ → return eval $(g_1, e) \cap$ eval (g_2, e)

Case $f \equiv g_1 \vee g_2$ → return eval $(g_1, e) \cup$ eval (g_2, e)

Case $f \equiv \langle a \rangle g$ → return $\{s \in S \mid \exists t [s \xrightarrow{a} t \wedge t \in \text{eval}(g, e)]\}$

Case $f \equiv [a] g$ → return $\{s \in S \mid \forall t [s \xrightarrow{a} t \Rightarrow t \in \text{eval}(g, e)]\}$.

Case $f \equiv \mu[Z = g(Z)]$ →

{ $T := \{\}$;
repeat
{ $T_{old} := T$; $T :=$ eval $(g, e\{T/Z\})$ }
until $T_{old} = T$;
return T
}

Case $f \equiv \nu[Z = g(Z)]$ →

{ $U := S$;
repeat
{ $U_{old} := U$; $U :=$ eval $(g, e\{U/Z\})$ }
until $U = U_{old}$;
return U
}.

}

g could contain fixpoint operators too!

Alternation depth.

For any μ-calculus formula f, $ad(f)$ is defined inductively as follows:

$ad(p) = ad(Z) = 0$

$ad(f \wedge g) = ad(f \vee g) = \max(ad(f), ad(g))$

$ad(\langle a \rangle f) = ad([a]f) = ad(f)$

$ad(\mu[Z = f(Z)])$ is defined inductively on the structure of f

i.e. $ad(\mu[Z = f(Z)])$

$= \max(1, ad(f), 1 + \max(ad(g)))$

where g is any top level ν-subformula of f.

Similarly

$ad(\nu[Z = f(Z)])$

$= \max(1, ad(f), 1 + \max(ad(g)))$

where g is any top-level μ-subformula of f.

The Complexity of model-checking μ-calculus formulae

Since each fixpoint evaluation can be $O(n)$ iterations where $n = |S|$. For any labelled transition system $L = \langle S, Act, \rightarrow, l \rangle$ it follows that the entire algorithm may take upto $\boxed{O((|S| + |\rightarrow|) \cdot |f| \cdot n^k)}$ where k is the number of nestings of fixpoint operators. This is because of the following.

(i) the loop for each fixpoint operator executes at most $(n+1)$ times.

(ii) if there are nested fixpoint operators then the evaluations of $e\{T/Z\}$ and $e\{U/Z\}$ require the computations of those fixpoints and hence if f has k-nested fixpoint operators ~~then~~ then n^k iterations may be required.

Improving the algorithm.

If the alternation depth of $f < k$, then there exist nestings of fixpoint operators which don't alternate (i.e. $\mu[Z = \mu[Y = \ldots]]$ or $\nu[Z = \nu[Y = \ldots]]$ are such nestings which don't alternate). In such cases the algorithm may be improved using simultaneous solutions. The net complexity would then be $O((n+1+|\rightarrow|) \cdot |f| \cdot n^d)$ where d is the alternation depth.

Bisimulation Checking

Given a lts $\langle S, Act, \rightarrow, \ell \rangle$ we generalize the notion of bisimulation as follows:

Def. $R \subseteq S \times S$ is a bisimulation if for all sRt

(i) $\ell(s) = \ell(t)$

(ii) $\forall s \xrightarrow{a} s' \Rightarrow \exists t' : t \xrightarrow{a} t' \wedge s'Rt'$

(iii) $\forall t \xrightarrow{a} t' \Rightarrow \exists s' : s \xrightarrow{a} s' \wedge s'Rt'$

We also have the following facts.

1. If R is a bisimulation then so is R^{-1}
2. If R_1 and R_2 are bisimulations then so is $R_1 \circ R_2$.
3. If $\{R_i \mid i \in I\}$ is a family of bisimulations then $\bigcup_{i \in I} R_i$ is a bisimulation

Now consider the function

$$\mathcal{B} : 2^{S \times S} \longrightarrow 2^{S \times S}$$

such that for any $R \subseteq S \times S$

$$\mathcal{B}(R) = \{(s,t) \mid (s,t) \models (i), (ii), (iii) \text{ of the def}\}.$$

Then it is easy to prove the following facts.

4. $R \subseteq S \times S$ is a bisimulation iff $R \subseteq \mathcal{B}(R)$.
5. If R is a bisimulation then so is $\mathcal{B}(R)$.

From fact 4. it is clear that every bisimulation is a pre-fixpoint of the function $\mathcal{B}$. Hence there exists a greatest fixpoint which is also a bisimulation. (By the Knaster-Tarski theorem). This is provided $\mathcal{B}$ is monotone, which may be easily shown. Hence we have

6. $\mathcal{B}$ is monotone, i.e. $R_1 \subseteq R_2 \Rightarrow \mathcal{B}(R_1) \subseteq \mathcal{B}(R_2)$.
7. There exists a greatest fixpoint of $\mathcal{B}$ given by
$$\sim = \bigcup \{R \subseteq S \times S \mid R \subseteq \mathcal{B}(R)\}.$$
which is itself a bisimulation.
8. $s \sim t$ iff there exists a bisimulation R with sRt.

Finite State LTSs

Let S and $\longrightarrow$ be finite, then again we have

9. $\mathcal{B}$ is both $\cup$-continuous and $\cap$-continuous
10. Then $\sim = \mathcal{B}^{nm}(S)$ where $n = |S|$ and $m = |T|$.

Let $R_0 = \{(s,t) \mid \ell(s) = \ell(t)\}$. Then clearly we have $\mathcal{B}(R_0) \subseteq R_0$. and since $\mathcal{B}$ is monotone we obtain the decreasing chain

$$R_0 \supseteq R_1 \supseteq \ldots \supseteq R_i \supseteq R_{i+1} \supseteq \ldots$$

where $R_{i+1} = \mathcal{B}(R_i)$.

Def: Let $\mathcal{C}: 2^{S \times S} \to 2^{S \times S}$ be the same as $\mathcal{B}$ except that for any relation R,

$$\mathcal{C}(R) = \{(s,t) \in R \mid (i), (ii), (iii) \text{ of } \mathcal{B} \text{ hold}\}.$$

Then it is clear that $\mathcal{C}(R) \subseteq R$ for any relation R. and it is no longer necessary to consider the full generality of $\mathcal{B}$.

Then it is clear that $\boxed{R_i = \mathcal{B}^i(R_0) = \mathcal{C}^i(R_0).}$

Hence for finite-state LTSs with a finite number of transitions, there exists a (trivial!) $O(mn)$ algorithm to compute $\sim$.

Partition Refinement

11. If $R \subseteq S \times S$ is an equivalence relation then $\mathcal{C}(R)$ is also an equivalence relation.

It is very easy to prove the above fact.

Def: Let $R \subseteq S \times S$ be an equivalence relation and let $B, B' \in S/R$ (such that B and B' are not necessarily distinct). Then for any $a \in Act$ define $\ominus(R, B, B', a) = R'$

$$= \{(s,t) \mid (s,t \in S-B \wedge sRt) \vee (s,t \in B \wedge (s \xrightarrow{a} \cap B' \neq \{\} \Leftrightarrow t \xrightarrow{a} \cap B' \neq \{\}))\}$$

read "split B wrt B', a"

Lemma. Let $R \subseteq S \times S$ be an equivalence relation. For every $B' \in S/R$ and every $a \in Act$, if $s \, \mathcal{C}(R) \, t$. then

$$s \xrightarrow{a} \cap B' \neq \{\} \Leftrightarrow t \xrightarrow{a} \cap B' \neq \{\}$$

Pf: Consider any $s\,\mathcal{B}(R)\,t$ and let

$$s \xrightarrow{a} \cap B' \neq \{\}$$

then $s \xrightarrow{a} s' \in B'$. By the def of $\mathcal{B}$ it follows that $t \xrightarrow{a} t' \in B'$. Hence $t \xrightarrow{a} \cap B' \neq \{\}$.
The converse follows by Symmetry $\rightarrow$.

Lemma. Let $R' = \Theta(R, B, B', a)$. as in its def. Then.

(i) R' is an equivalence relation
(ii) R' is at least as fine as R.
(iii) $\mathcal{C}(R)$ is at least as fine as R'.
(iv) If $\mathcal{C}(R)$ is strictly finer than R then there exist $B, B' \in S/R$ and $a \in Act$ such that $R' = \Theta(R, B, B', a)$ is strictly finer than R.

Proof (i) It is clear that R' is reflexive since R is reflexive. For Symmetry is obvious from the definition of Θ. Let $sR'tR'u$. It is clear that $s \xrightarrow{a} \cap B' \neq \{\} \Leftrightarrow t \xrightarrow{a} \cap B' \neq \{\} \Leftrightarrow u \xrightarrow{a} \cap B' \neq \{\}$.
Hence $sR'u$.
(ii) That R' is at least as fine as R. is clear from the fact that only B is split if at all.

(iii) Since $\mathcal{C}(R) \subseteq R$, consider any $(s,t) \in \mathcal{C}(R)$. Clearly $(s,t) \in R$. Hence either $\nexists s,t \in S-B$ or $s,t \in B$. If $s,t \in S-B$ then clearly $sR't$. Otherwise by the definition of Θ if $\neg(sR't)$ then $s \xrightarrow{a} \cap B' \neq \{\} \not\Leftrightarrow t \xrightarrow{a} \cap B' \neq \{\}$

Without loss of generality assume

$$s \xrightarrow{a} \cap B' = \{\}$$

and $t \xrightarrow{a} \cap B' \neq \{\}$.

Then it follows that

$$t \xrightarrow{a} t' \not\Rightarrow \exists s' : s \xrightarrow{a} s' \wedge s' R t'$$

and hence $\neg(s\ \mathcal{C}(R)\ t)$ which contradicts the assumption $s\ \mathcal{C}(R)\ t$.

(iv) Assume sRt and $\neg(s\ \mathcal{C}(R)\ t)$. Then for some a, s' we have

$$s \xrightarrow{a} s' \wedge \forall t' : t \xrightarrow{a} t' \Rightarrow \neg s' R t'$$

If $(s,t) \in B \in S/R$. and $s' \in B' \in S/R$, then clearly

$$\neg(s\ \Theta(R,B,B',a)\ t).$$

Hence $\Theta(R,B,B',a)$ is strictly finer than R.

CTL*, Kripke Structures & Bisimulation

Let $\mathcal{K}_1$ and $\mathcal{K}_2$ be two Kripke Structures then the notion of a bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$ is obvious.

$$\mathcal{K}_1 = (S_1, \rightarrow_1, L_1)$$

$$\mathcal{K}_2 = (S_2, \rightarrow_2, L_2)$$

Two paths $\pi_1 = s_1^0 \rightarrow s_1^1 \rightarrow \cdots \rightarrow$ and in $\mathcal{K}_1$ and

$\pi_2 = s_2^0 \rightarrow s_2^1 \rightarrow \cdots \rightarrow$ in $\mathcal{K}_2$

correspond if there exists a bisimulation $B \subseteq S_1 \times S_2$ such that $s_1^i B s_2^i$. for $i \geq 0$.

Lemma. Let B be a bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$. Then for any $s_1 B s_2$, for every path π_1 from s_1 there exists a corresponding path π_2 starting from s_2.

Lemma. Let s_1 and s_2 be bisimilar states and π_1 and π_2 be corresponding paths. Then for any state formula ψ and path formula φ we have

$\mathcal{K}_1, s_1 \models \psi$ iff $\mathcal{K}_2, s_2 \models \psi$

and $\mathcal{K}_1, \pi_1 \models \varphi$ iff $\mathcal{K}_2, \pi_2 \models \varphi$.

Proof: By induction on the structures of φ and ψ +.

Theorem: If $\mathcal{K}_1 \sim \mathcal{K}_2$ then for every CTL* formula f, $s_1 \models f$ iff $s_2 \models f$.

Claims without proof.

1. If two structures satisfy the same set of CTL* formulas then they are bisimulation equivalent.
2. In fact if two structures satisfy the same set of CTL formulas they are bisimilar.
3. It follows that if two structures can be distinguished by a CTL* formula, they can also be distinguished by a CTL formula.

Fair Bisimulations for Kripke Structures with fairness constraints

$B \subseteq S_1 \times S_2$ is a fair bisimulation iff for all $s_1 B s_2$

(i) $l(s_1) = l(s_2)$

(ii) For each fair path π_1 in $\mathcal{K}_1$ there is a B-corresponding fair path π_2 in $\mathcal{K}_2$

(iii) For each fair path π_2 in $\mathcal{K}_2$, there is a B-corresponding fair path π_1 in $\mathcal{K}_1$.

Theorem. If $\mathcal{K}_1 \sim_F \mathcal{K}_2$ then for every CTL* formula f interpreted over fair paths, $\mathcal{K}_1 \models f$ iff $\mathcal{K}_2 \models f$.