Formale Spezifikation und Verifikation

Mirco Tribastone

Institut für Informatik
Ludwig-Maximilians-Universität München
tribastone@pst.ifi.lmu.de

Computation Tree Logic

Overview

- Kripke Structures
- Computation Tree Logic (CTL)
- Model Checking CTL
- Fixed-point characterisation of CTL

Kripke Structures

Definition

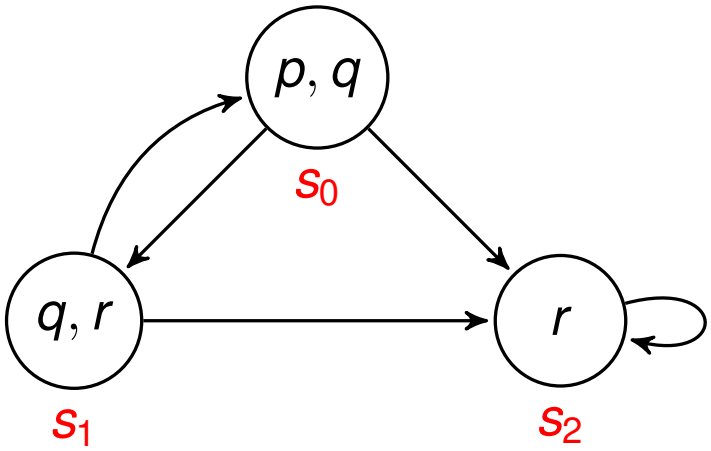
Let Atom be a set of atomic propositions. A model (also called Kripke structure) is a tuple $\mathcal{M} = (S, \rightarrow, L)$ where

- S is a set of states
- $\rightarrow$ is a left-total transition relation, i.e., $\rightarrow \subseteq S \times S$ and for each $s \in S$ there exist $s' \in S$ such that $(s, s') \in \rightarrow$
- L is a labelling function $L : S \rightarrow \mathcal{P}(\text{Atom})$ and $\mathcal{P}(\text{Atom})$ denotes the power-set of Atom.
- As usual, $(s, s') \in \rightarrow$ is often written $s \rightarrow s'$.

Comparing this definition with that of labelled transition systems:

- In LTS transitions are labelled, not states
- An LTS admits deadlocked states, i.e., states without outgoing transitions

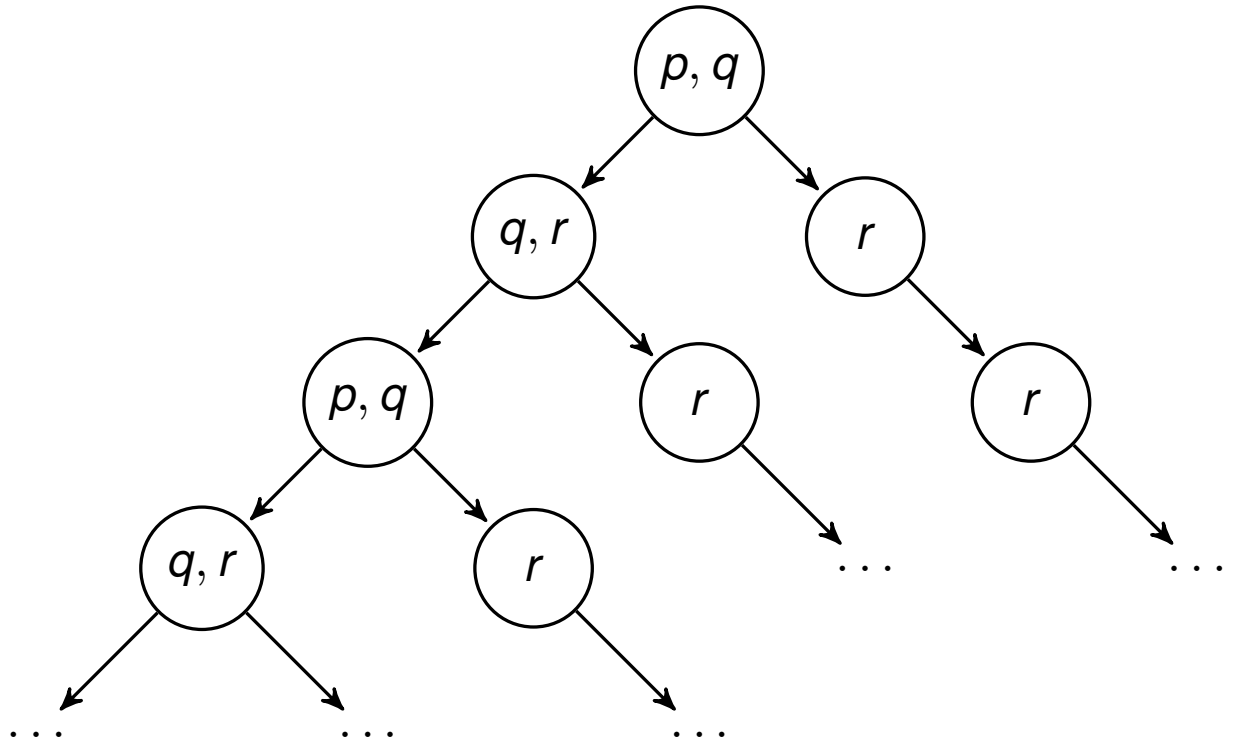
Example



$$S = \{s_0, s_1, s_2\}$$
$$\rightarrow = \{(s_0, s_1), (s_0, s_2), (s_1, s_0), (s_1, s_2), (s_2, s_2)\}$$
$$\text{Atom} = \{p, q, r\}$$
$$L(s_0) = \{p, q\}$$
$$L(s_1) = \{q, r\}$$
$$L(s_2) = \{r\}$$

Computation Tree

Unwinding of the Kripke Structure from s_0



- Nondeterministic behaviour
- Nonterminating computation
- Branching time

Computation Tree Logic

We wish to define a formal method for specifying properties such as

- Statements over states
  - Does state s satisfy the atomic property p?
  - Does state s satisfy the atomic properties p and q?
  - Does state s satisfy the atomic properties p or q?
  - ...
- Statements over paths
  - Is there a path starting from s such that property p always hold?
  - Is there a path starting from s such that property p holds some time in the future?
  - From all paths starting from s does property p always hold?
  - ...

Syntax of CTL – 1

A CTL *formula* is defined by the following BNF grammar:

$$\phi := \bot \mid \top \mid p \mid (\neg\phi) \mid (\phi \wedge \phi) \mid (\phi \vee \phi) \mid (\phi \rightarrow \phi) \mid$$
$$\mathbf{AX}\,\phi \mid \mathbf{EX}\,\phi \mid \mathbf{A}[\,\phi\,\mathbf{U}\,\phi\,] \mid \mathbf{E}[\,\phi\,\mathbf{U}\,\phi\,] \mid \mathbf{AG}\,\phi \mid$$
$$\mathbf{EG}\,\phi \mid \mathbf{AF}\,\phi \mid \mathbf{EF}\,\phi\,, \qquad \text{with } p \in \text{Atom}\,.$$

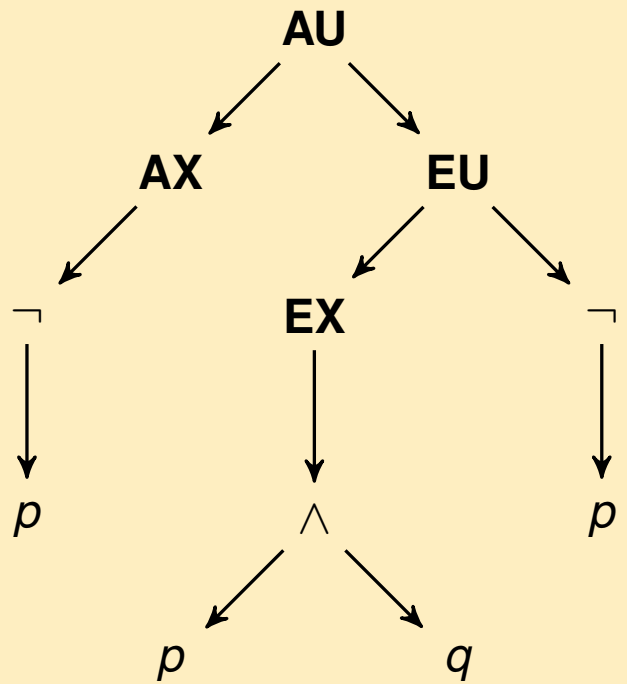
- *A* means *along All paths* (inevitably)
- *E* means *along at least one path* (i.e., there Exists one path) (possibly)
- *X* means *neXt state*
- *F* means *some Future state*
- *G* means *Globally* (all future states)
- *U* means *Until*

Syntax of CTL – 2

Binding Priority

1. Unary connectives $\neg$, **AG**, **EG**, **AF**, **EF**, and **AX**
2. $\wedge$ and $\vee$
3. $\rightarrow$, $\mathbf{A}[\,\cdot\,\mathbf{U}\,\cdot\,]$, and $\mathbf{E}[\,\cdot\,\mathbf{U}\,\cdot\,]$

Parse Tree of $\mathbf{A}[\,\mathbf{AX}\,\neg p\,\mathbf{U}\,\mathbf{E}[\,\mathbf{EX}(p \wedge q)\,\mathbf{U}\,\neg p\,]\,]$



Semantics of CTL – 1

Let $\mathcal{M} = (S, \rightarrow, L)$ be a model for CTL. Given $s \in S$ the relation

$$\mathcal{M}, s \models \phi \qquad \text{(often abbreviated } s \models \phi\text{)}$$

defines whether the CTL formula ϕ holds in s.

The satisfaction relation $\models$ is defined by structural induction thus:

- $\mathcal{M}, s \models \top$ and $\mathcal{M}, s \not\models \bot$ for all $s \in S$.
- $\mathcal{M}, s \models p$ iff $p \in L(s)$.
- $\mathcal{M}, s \models \neg\phi$ iff $\mathcal{M}, s \not\models \phi$.
- $\mathcal{M}, s \models \phi_1 \wedge \phi_2$ iff $\mathcal{M}, s \models \phi_1$ and $\mathcal{M}, s \models \phi_2$.
- $\mathcal{M}, s \models \phi_1 \vee \phi_2$ iff $\mathcal{M}, s \models \phi_1$ or $\mathcal{M}, s \models \phi_2$.
- $\mathcal{M}, s \models \phi_1 \rightarrow \phi_2$ iff $\mathcal{M}, s \not\models \phi_1$ or $\mathcal{M}, s \models \phi_2$.

Semantics of $p \rightarrow q$

The formula $p \rightarrow q$ is interpreted as checking whether *truth is preserved*.

ϕ	ψ	$\phi \rightarrow \psi$
T	T	T
T	F	F
F	T	T
F	F	T

ϕ	ψ	$\neg\phi \vee \psi$
T	T	T
T	F	F
F	T	T
F	F	T

Therefore,

$$\mathcal{M}, s \models \phi_1 \rightarrow \phi_2 \text{ iff } \mathcal{M}, s \not\models \phi_1 \text{ or } \mathcal{M}, s \models \phi_2 \ .$$

Semantics of CTL – 2

Let $\mathcal{M} = (S, \rightarrow, L)$ be a model for CTL. Given $s \in S$ the relation

$$\mathcal{M}, s \models \phi \qquad \text{(often abbreviated } s \models \phi\text{)}$$

defines whether the CTL formula ϕ holds in s.

The satisfaction relation $\models$ is defined by structural induction thus:

- $\mathcal{M}, s \models \mathbf{AX}\,\phi$ iff *for all* s' such that $s \rightarrow s'$ we have $\mathcal{M}, s' \models \phi$.
- $\mathcal{M}, s \models \mathbf{EX}\,\phi$ iff *for some* s' such that $s \rightarrow s'$ we have $\mathcal{M}, s' \models \phi$.
- $\mathcal{M}, s \models \mathbf{AG}\,\phi$ iff *for all* paths $s_1 \rightarrow s_2 \rightarrow \ldots$, with $s = s_1$, we have $\mathcal{M}, s_i \models \phi$, with $i = 1, 2, \ldots$
- $\mathcal{M}, s \models \mathbf{EG}\,\phi$ iff *there exists* a path $s_1 \rightarrow s_2 \rightarrow \ldots$, with $s = s_1$, such that we have $\mathcal{M}, s_i \models \phi$, with $i = 1, 2, \ldots$

Semantics of CTL – 3

Let $\mathcal{M} = (S, \rightarrow, L)$ be a model for CTL. Given $s \in S$ the relation

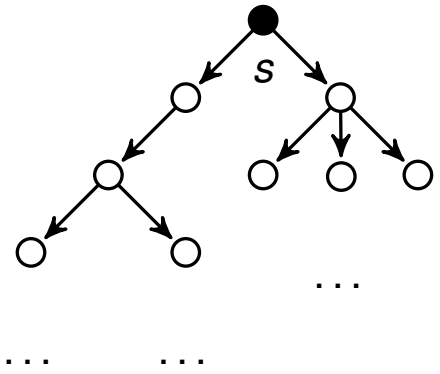
$$\mathcal{M}, s \models \phi \qquad \text{(often abbreviated } s \models \phi\text{)}$$

defines whether the CTL formula ϕ holds in s.

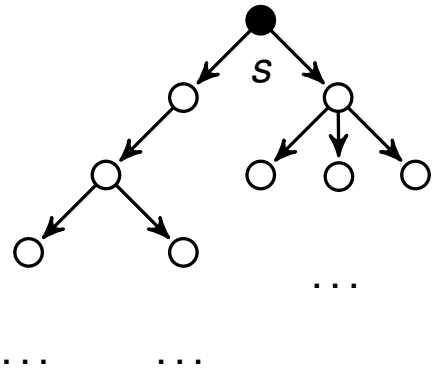
The satisfaction relation $\models$ is defined by structural induction thus:

- $\mathcal{M}, s \models \mathbf{AF}\,\phi$ iff *for all* paths $s_1 \rightarrow s_2 \rightarrow \ldots$, with $s = s_1$, there exists s_i such that $\mathcal{M}, s_i \models \phi$.
- $\mathcal{M}, s \models \mathbf{EF}\,\phi$ iff *for some* path $s_1 \rightarrow s_2 \rightarrow \ldots$, with $s = s_1$, there exists s_i such that $\mathcal{M}, s_i \models \phi$.
- $\mathcal{M}, s \models \mathbf{A}[\,\phi_1\,\mathbf{U}\,\phi_2\,]$ iff *for all* paths $s_1 \rightarrow s_2 \rightarrow \ldots$, with $s = s_1$, there is some s_i such that $\mathcal{M}, s_i \models \phi_2$ and, for each $j < i$ we have $\mathcal{M}, s_j \models \phi_1$.
- $\mathcal{M}, s \models \mathbf{E}[\,\phi_1\,\mathbf{U}\,\phi_2\,]$ iff *for some* path $s_1 \rightarrow s_2 \rightarrow \ldots$, with $s = s_1$, there is some s_i such that $\mathcal{M}, s_i \models \phi_2$ and, for each $j < i$, we have $\mathcal{M}, s_j \models \phi_1$.

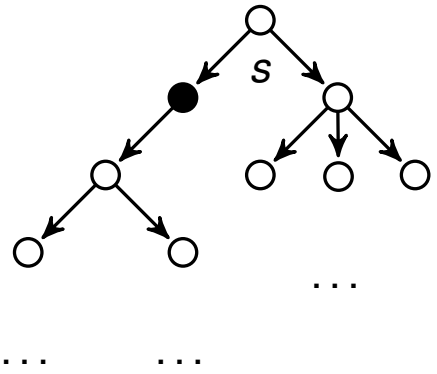
Illustrations – 1



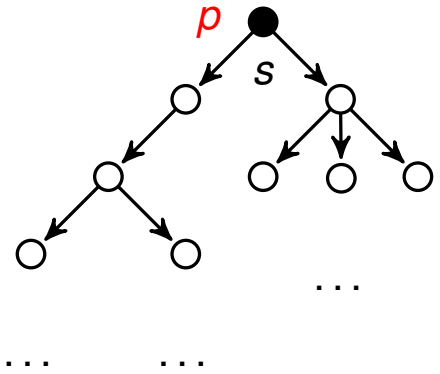
$\mathcal{M}, s \models \text{black}$



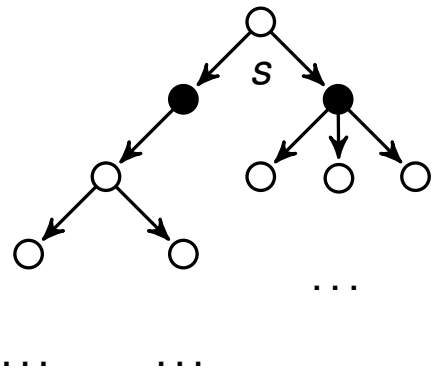
$\mathcal{M}, s \models \text{black} \vee \text{grey}$



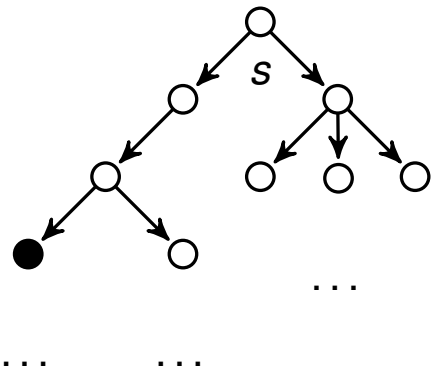
$\mathcal{M}, s \models \mathbf{EX}\,\text{black}$



$\mathcal{M}, s \models p \wedge \text{black}$

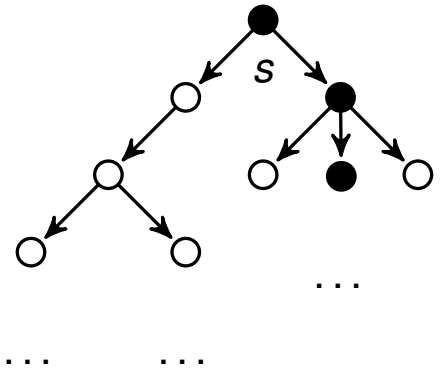


$\mathcal{M}, s \models \mathbf{AX}\,\text{black}$

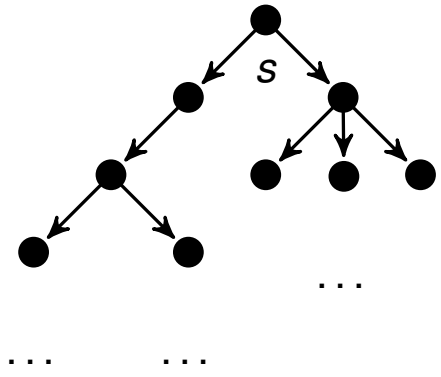


$\mathcal{M}, s \models \mathbf{EF}\,\text{black}$

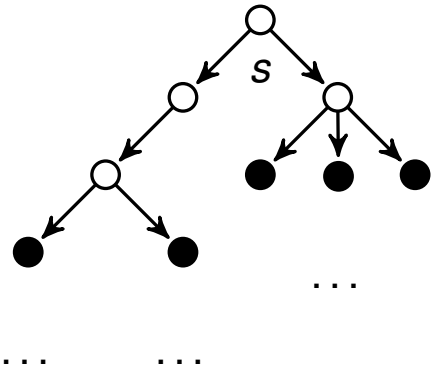
Illustrations – 2



$\mathcal{M}, s \models \mathbf{EG}\,\text{black}$

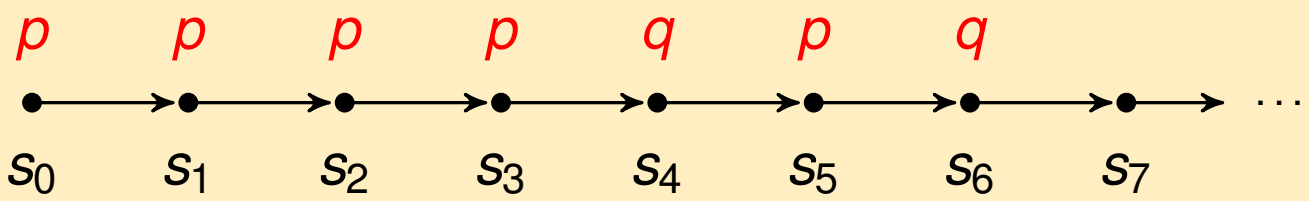


$\mathcal{M}, s \models \mathbf{AG}\,\text{black}$



$\mathcal{M}, s \models \mathbf{AF}\,\text{black}$

Until Semantics



$$\mathcal{M}, s_i \models \mathbf{E}[\, p\,\mathbf{U}\,q\,], \quad 0 \le i \le 3\,.$$

$$\mathcal{M}, s_4 \models \mathbf{E}[\, p\,\mathbf{U}\,q\,]\,?$$

$$\mathcal{M}, s_5 \models \mathbf{E}[\, p\,\mathbf{U}\,q\,]\,?$$

$$\mathcal{M}, s_6 \models \mathbf{E}[\, p\,\mathbf{U}\,q\,]\,?$$

Practical Patterns of Specification

- It is possible to get to a state where the machine has been started, but it is not ready yet:

$$\mathbf{EF}(\text{started} \wedge \neg \text{ready})$$

- For any state, if a message arrives, then it will eventually be acknowledged:

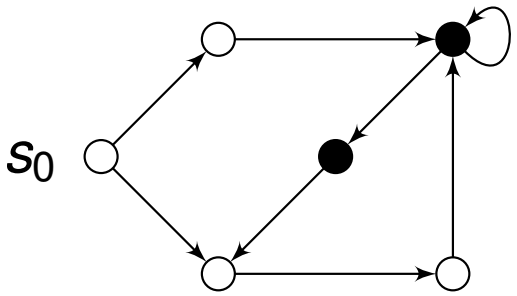
$$\mathbf{AG}(\text{arrived} \rightarrow \mathbf{AF}\,\text{ack})$$

- An upwards travelling elevator at the second floor does not change its direction when it has passengers wishing to travel to the fifth floor:

$$\mathbf{AG}(\text{floor=2} \wedge \text{direction=up} \wedge \text{button pressed=5} \rightarrow \mathbf{A}[\, \mathit{direction} = \mathit{up}\, \mathbf{U}\, \mathit{floor} = 5\,])$$

An Example

Given the Kripke structure $\mathcal{M}$



$\mathcal{M}, s_0 \models \mathbf{AG}\,\text{black}$?

$\mathcal{M}, s_0 \models \mathbf{EF}\,\text{black}$?

$\mathcal{M}, s_0 \models \mathbf{AF}\,\text{black}$?

$\mathcal{M}, s_0 \models \mathbf{EF}\,\mathbf{AG}\,\text{black}$?

$\mathcal{M}, s_0 \models \mathbf{EF}\,\mathbf{EG}\,\text{black}$?

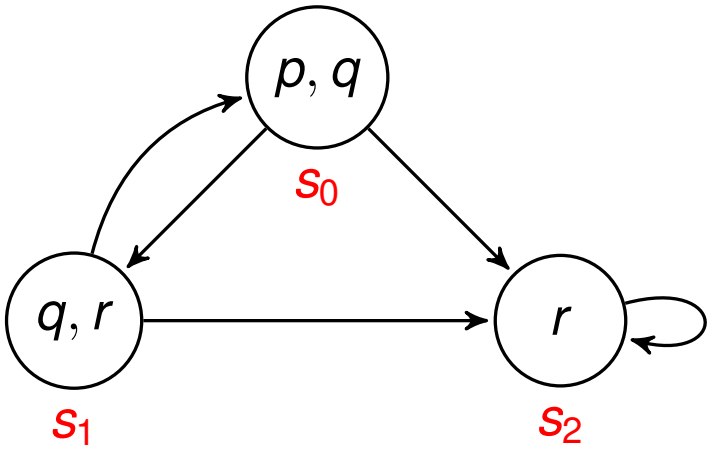
$\mathcal{M}, s_0 \models \mathbf{AG}\,\mathbf{EF}\,\mathbf{EG}\,\text{black}$?

$\mathcal{M}, s_0 \models \mathbf{AG}(\text{black} \rightarrow \mathbf{AF}\,\neg\text{black})$?

$\mathcal{M}, s_0 \models \mathbf{AG}(\text{black} \rightarrow \mathbf{EF}\,\neg\text{black})$?

$\mathcal{M}, s_0 \models \mathbf{AG}(\mathbf{A}[\,\neg\text{black}\,\mathbf{U}\,\text{black}\,])$?

Another Example



$\mathcal{M}, s_0 \models p \wedge q$

$\mathcal{M}, s_0 \models \neg r$

$\mathcal{M}, s_0 \models \top$

$\mathcal{M}, s_0 \models \mathbf{EX}(q \wedge r)$

$\mathcal{M}, s_0 \models \neg\, \mathbf{AX}(q \wedge r)$

$\mathcal{M}, s_0 \models \neg\, \mathbf{EF}(p \wedge r)$

$\mathcal{M}, s_2 \models \mathbf{EG}\, r$

$\mathcal{M}, s_2 \models \mathbf{AG}\, r$

$\mathcal{M}, s_0 \models \mathbf{AF}\, r$

$\mathcal{M}, s_0 \models \mathbf{E}[\, (p \wedge q)\, \mathbf{U}\, r\,]$

$\mathcal{M}, s_0 \models \mathbf{A}[\, p\, \mathbf{U}\, r\,]$

Equivalences Between Formulas

Definition

Two CTL formulas ϕ and ψ are said to be *semantically equivalent*, written $\phi \equiv \psi$, if $\mathcal{M}, s \models \phi \iff \mathcal{M}, s \models \psi$, for any state s in any model $\mathcal{M}$.

$$\top \equiv \psi \vee \neg\psi$$

$$\phi \wedge \psi \equiv \neg(\neg\phi \vee \neg\psi)$$

$$\phi \rightarrow \psi \equiv \neg\phi \vee \psi$$

$$\neg\, \mathbf{AF}\, \phi \equiv \mathbf{EG}\, \neg\phi$$

$$\neg\, \mathbf{EF}\, \phi \equiv \mathbf{AG}\, \neg\phi$$

$$\neg\, \mathbf{AX}\, \phi \equiv \mathbf{EX}\, \neg\phi$$

$$\mathbf{AF}\, \phi \equiv \mathbf{A}[\, \top\, \mathbf{U}\, \phi\,]$$

$$\mathbf{EF}\, \phi \equiv \mathbf{E}[\, \top\, \mathbf{U}\, \phi\,]$$

$$\mathbf{A}[\, \phi\, \mathbf{U}\, \psi\,] \equiv \neg\big(\, \mathbf{E}[\, \neg\psi\, \mathbf{U}(\neg\phi \wedge \neg\psi)\,] \vee \mathbf{EG}\, \neg\psi\big)$$

Other Crucial Equivalences

$$\mathbf{AG}\,\phi \equiv \phi \wedge \mathbf{AX}\,\mathbf{AG}\,\phi$$
$$\mathbf{EG}\,\phi \equiv \phi \wedge \mathbf{EX}\,\mathbf{EG}\,\phi$$
$$\mathbf{AF}\,\phi \equiv \phi \vee \mathbf{AX}\,\mathbf{AF}\,\phi$$
$$\mathbf{EF}\,\phi \equiv \phi \vee \mathbf{EX}\,\mathbf{EF}\,\phi$$
$$\mathbf{A}[\,\phi\,\mathbf{U}\,\psi\,] \equiv \psi \vee \left(\phi \wedge \mathbf{AX}\,\mathbf{A}[\,\phi\,\mathbf{U}\,\psi\,]\right)$$
$$\mathbf{E}[\,\phi\,\mathbf{U}\,\psi\,] \equiv \psi \vee \left(\phi \wedge \mathbf{EX}\,\mathbf{E}[\,\phi\,\mathbf{U}\,\psi\,]\right)$$

For instance, $\mathbf{EG}\,\phi$ means that ϕ must be always true along some path. But this is equivalent to stating that it must be true now and that there exist a successor such that ϕ holds along at least one of its paths.

Other Crucial Equivalences

$$\mathbf{AG}\,\phi \equiv \phi \wedge \mathbf{AX}\,\mathbf{AG}\,\phi$$
$$\mathbf{EG}\,\phi \equiv \phi \wedge \mathbf{EX}\,\mathbf{EG}\,\phi$$
$$\mathbf{AF}\,\phi \equiv \phi \vee \mathbf{AX}\,\mathbf{AF}\,\phi$$
$$\mathbf{EF}\,\phi \equiv \phi \vee \mathbf{EX}\,\mathbf{EF}\,\phi$$
$$\mathbf{A}[\,\phi\,\mathbf{U}\,\psi\,] \equiv \psi \vee \left(\phi \wedge \mathbf{AX}\,\mathbf{A}[\,\phi\,\mathbf{U}\,\psi\,]\right)$$
$$\mathbf{E}[\,\phi\,\mathbf{U}\,\psi\,] \equiv \psi \vee \left(\phi \wedge \mathbf{EX}\,\mathbf{E}[\,\phi\,\mathbf{U}\,\psi\,]\right)$$

Similarly, **AF** ϕ means that ϕ must be true at some point along each path. But this is equivalent to stating that either it must be true now, or all the successors must be such that ϕ holds at some point along each of their paths.

Preprocessing of CTL Formulas

Theorem

Any CTL formula can be transformed into a semantically equivalent CTL formula which uses only the logical connectives $\bot$, $\neg$, $\wedge$, **AF**, **EU**, and **EX**. We say that these connectives are an *adequate set* for CTL.

function TRANS(ϕ)
if ϕ is $\top$ **then**
  return $\neg\bot$
else if ϕ is $\bot$ or $\phi \in$ Atom **then**
  return ϕ
else if ϕ is $\neg\phi_1$ **then**
  return $\neg$TRANS(ϕ_1)
else if ϕ is $\phi_1 \wedge \phi_2$ **then**
  return TRANS(ϕ_1) $\wedge$ TRANS(ϕ_2)
else if ϕ is $\phi_1 \vee \phi_2$ **then**
  return
  $\neg\big(\neg$TRANS(ϕ_1) $\wedge \neg$TRANS(ϕ_2)$\big)$
else if ϕ is $\phi_1 \rightarrow \phi_2$ **then**
  return TRANS($\neg\phi_1 \vee \phi_2$)
else if ϕ is **AX** ϕ_1 **then**
  return TRANS($\neg$ **EX** $\neg\phi_1$)
end if...

...
if ϕ is **EX** ϕ_1 **then**
  return **EX** TRANS(ϕ_1)
else if ϕ is **AG** ϕ_1 **then**
  return TRANS($\neg$ **EF** $\neg\phi_1$)
else if ϕ is **EG** ϕ_1 **then**
  TRANS($\neg$ **AF** $\neg\phi_1$)
else if ϕ is **AF** ϕ_1 **then**
  AF TRANS(ϕ_1)
else if ϕ is **EF** ϕ_1 **then**
  return TRANS(**E**[$\top$ **U** ϕ_1])
else if ϕ is **A**[ϕ_1 **U** ϕ_2] **then**
  return
  TRANS($\neg\big($ **E**[$\neg\phi_2$ **U**($\neg\phi_1 \wedge \neg\phi_2$)] $\vee$ **EG** $\neg\phi_2\big)$)
else if ϕ is **E**[ϕ_1 **U** ϕ_2] **then**
  return **E**[TRANS(ϕ_1) **U** TRANS(ϕ_2)]
end if