FORMAL METHODS

LECTURE V: CTL MODEL CHECKING

Alessandro Artale

Faculty of Computer Science – Free University of Bolzano

Room 2.03

artale@inf.unibz.it   http://www.inf.unibz.it/∼artale/

Some material (text, figures) displayed in these slides is courtesy of:

M. Benerecetti, A. Cimatti, M. Fisher, F. Giunchiglia, M. Pistore, M. Roveri, R.Sebastiani.

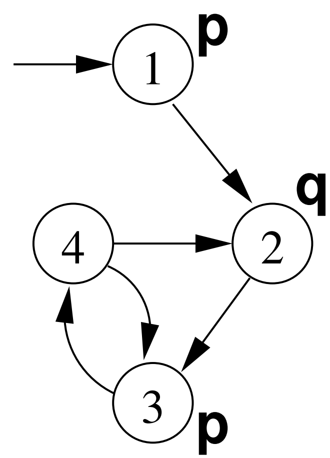
Summary of Lecture V

- CTL Model Checking: General Ideas.
- CTL Model Checking: The Labeling Algorithm.
- Labeling Algorithm in Details.
- CTL Model Checking: Theoretical Issues.

CTL Model Checking

CTL Model Checking is a formal verification technique s.t.

- The system is represented as a Kripke Model $\mathcal{KM}$:



- The property is expressed as a CTL formula φ, e.g.:

$$\boxed{\text{P}}\ \Box(p \Rightarrow \boxed{\text{P}}\ \Diamond q)$$

- The algorithm checks whether **all** the initial states, s_0, of the Kripke model satisfy the formula ($\mathcal{KM}, s_0 \models \varphi$).

CTL M.C. Algorithm: General Ideas

The algorithm proceeds along two macro-steps:

1. Construct the set of states where the formula holds:

$$[\![\varphi]\!] := \{s \in S : \mathcal{KM}, s \models \varphi\}$$

   ($[\![\varphi]\!]$ is called the denotation of φ);

2. Then compare the denotation with the set of initial states:

$$I \subseteq [\![\varphi]\!]?$$

CTL M.C. Algorithm: General Ideas (Cont.)

To compute $[\![\varphi]\!]$ proceed "bottom-up" on the structure of the formula, computing $[\![\varphi_i]\!]$ for each subformula φ_i of φ.

For example, to compute $[\![\boxed{\mathrm{P}}\,\Box(p \Rightarrow \boxed{\mathrm{P}}\,\Diamond q)]\!]$ we need to compute:

- $[\![q]\!]$,
- $[\![\boxed{\mathrm{P}}\,\Diamond q]\!]$,
- $[\![p]\!]$,
- $[\![p \Rightarrow \boxed{\mathrm{P}}\,\Diamond q]\!]$,
- $[\![\boxed{\mathrm{P}}\,\Box(p \Rightarrow \boxed{\mathrm{P}}\,\Diamond q)]\!]$

CTL M.C. Algorithm: General Ideas (Cont.)

To compute each $[\![\varphi_i]\!]$ for generic subformulas:

- Handle boolean operators by standard set operations;
- Handle temporal operators $\boxed{\mathrm{P}}\bigcirc$, $\langle\mathrm{P}\rangle\bigcirc$ by computing pre-images;
- Handle temporal operators $\boxed{\mathrm{P}}\Box$, $\langle\mathrm{P}\rangle\Box$, $\boxed{\mathrm{P}}\Diamond$, $\langle\mathrm{P}\rangle\Diamond$, $\boxed{\mathrm{P}}\,\mathcal{U}$, $\langle\mathrm{P}\rangle\,\mathcal{U}$, by applying fixpoint operators.

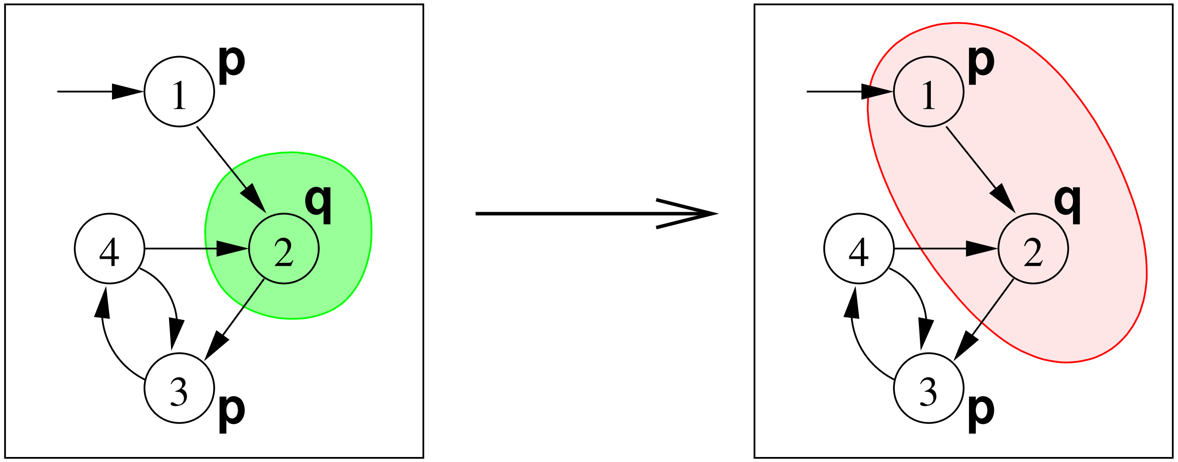
Summary

- CTL Model Checking: General Ideas.
- CTL Model Checking: The Labeling Algorithm.
- Labeling Algorithm in Details.
- CTL Model Checking: Theoretical Issues.

The Labeling Algorithm: General Idea

- The Labeling Algorithm:
  - Input: Kripke Model and a CTL formula;
  - Output: set of states satisfying the formula.
- Main Idea: Label the states of the Kripke Model with the subformulas of φ satisfied there.

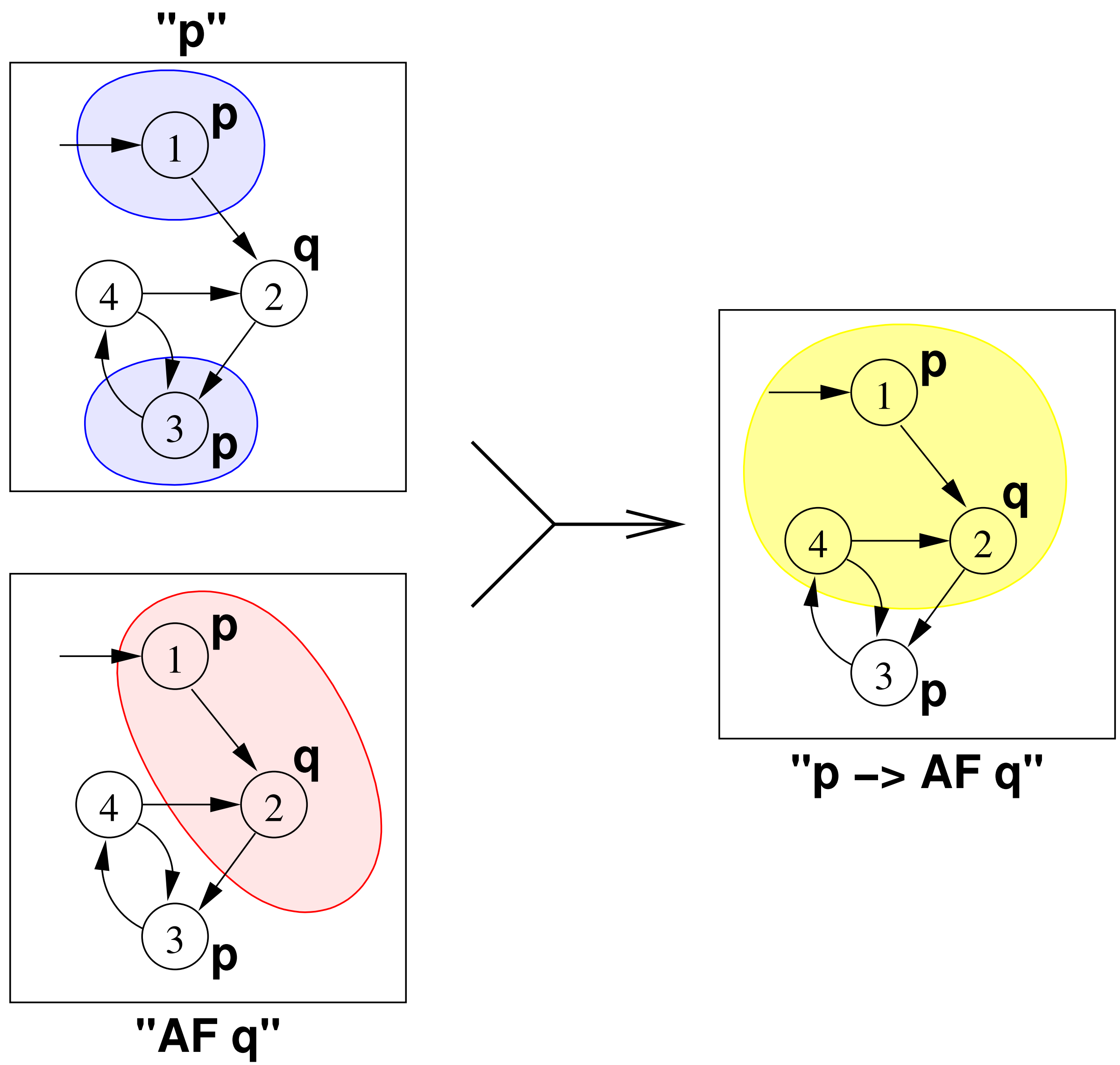
The Labeling Algorithm: An Example



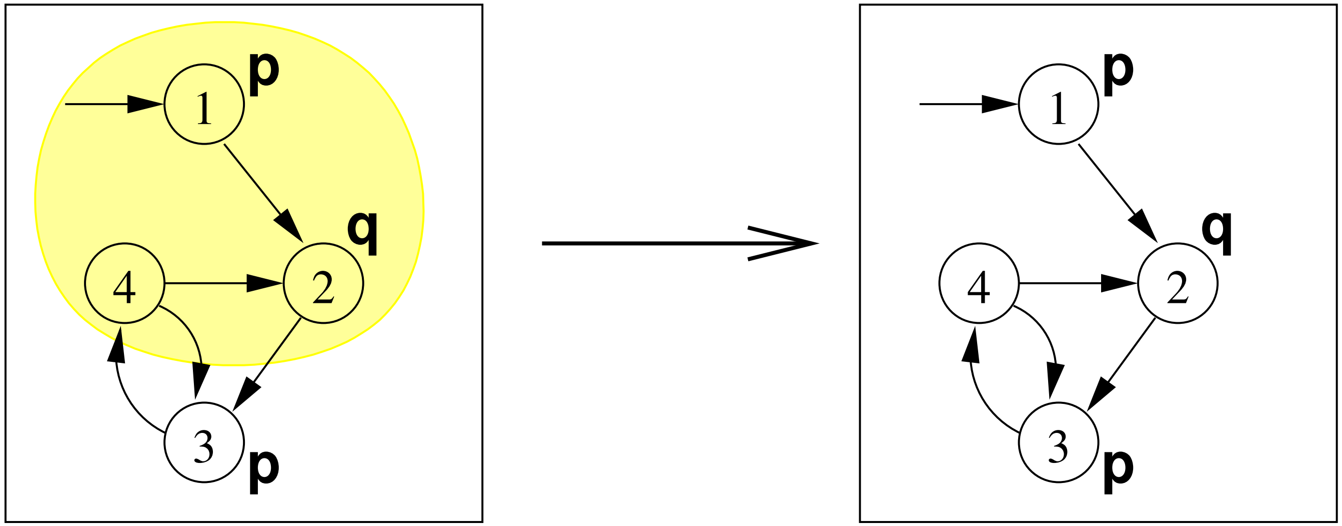
"q" "AF q"

- ▷ $\boxed{\text{P}}\Diamond q \equiv (q \vee \boxed{\text{P}}\bigcirc(\boxed{\text{P}}\Diamond q))$
- ▷ $[\![\boxed{\text{P}}\Diamond \text{q}]\!]$ can be computed as the union of:
  - $[\![q]\!] = \{2\}$
  - $[\![q \vee \boxed{\text{P}}\bigcirc q]\!] = \{2\} \cup \{1\} = \{1,2\}$
  - $[\![q \vee \boxed{\text{P}}\bigcirc(q \vee \boxed{\text{P}}\bigcirc q)]\!] = \{2\} \cup \{1\} = \{1,2\}$ (fixpoint).

The Labeling Algorithm: An Example (Cont.)



The Labeling Algorithm: An Example (Cont.)



"p –> AF q" "AG(p –> AF q)"

- ▷ $\boxed{\mathrm{P}}\ \Box\varphi \equiv (\varphi \wedge \boxed{\mathrm{P}} \bigcirc (\boxed{\mathrm{P}}\ \Box\varphi))$
- ▷ $[\![\boxed{\mathrm{P}}\ \Box\varphi]\!]$ can be computed as the intersection of:
  - $[\![\varphi]\!] = \{1,2,4\}$
  - $[\![\varphi \wedge \boxed{\mathrm{P}} \bigcirc \varphi]\!] = \{1,2,4\} \cap \{1,3\} = \{1\}$
  - $[\![\varphi \wedge \boxed{\mathrm{P}} \bigcirc (\varphi \wedge \boxed{\mathrm{P}} \bigcirc \varphi)]\!] = \{1,2,4\} \cap \{\} = \{\}$ (fixpoint)

The Labeling Algorithm: An Example (Cont.)

- The set of states where the formula holds is empty, thus:
  - The initial state does not satisfy the property;
  - $\mathcal{KM} \not\models \boxed{\text{P}}\,\Box(p \Rightarrow \boxed{\text{P}}\,\Diamond q)$.

- Counterexample: A lazo-shaped path: $1,2,\{3,4\}^{\omega}$ (satisfying $\Diamond_{\text{P}}\Diamond(p \wedge \Diamond_{\text{P}}\Box\neg q)$)

Summary

- CTL Model Checking: General Ideas.
- CTL Model Checking: The Labeling Algorithm.
- Labeling Algorithm in Details.
- CTL Model Checking: Theoretical Issues.

The Labeling Algorithm: General Schema

- Assume φ written in terms of $\neg$, $\wedge$, $\Diamond\bigcirc$, $\Diamond\,\mathcal{U}$, $\Diamond\Box$ – minimal set of CTL operators
- The Labeling algorithm takes a CTL formula and a Kripke Model as input and returns the set of states satisfying the formula (i.e., the *denotation* of φ):
  1. For every $\varphi_i \in Sub(\varphi)$, find $[\![\varphi_i]\!]$;
  2. Compute $[\![\varphi]\!]$ starting from $[\![\varphi_i]\!]$;
  3. Check if $I \subseteq [\![\varphi]\!]$.
- Subformulas $Sub(\varphi)$ of φ are checked bottom-up
- To compute each $[\![\varphi_i]\!]$: if the main operator of φ_i is a
  - *Boolean Operator*: apply standard set operations;
  - *Temporal Operator*: apply recursive rules until a fixpoint is reached.

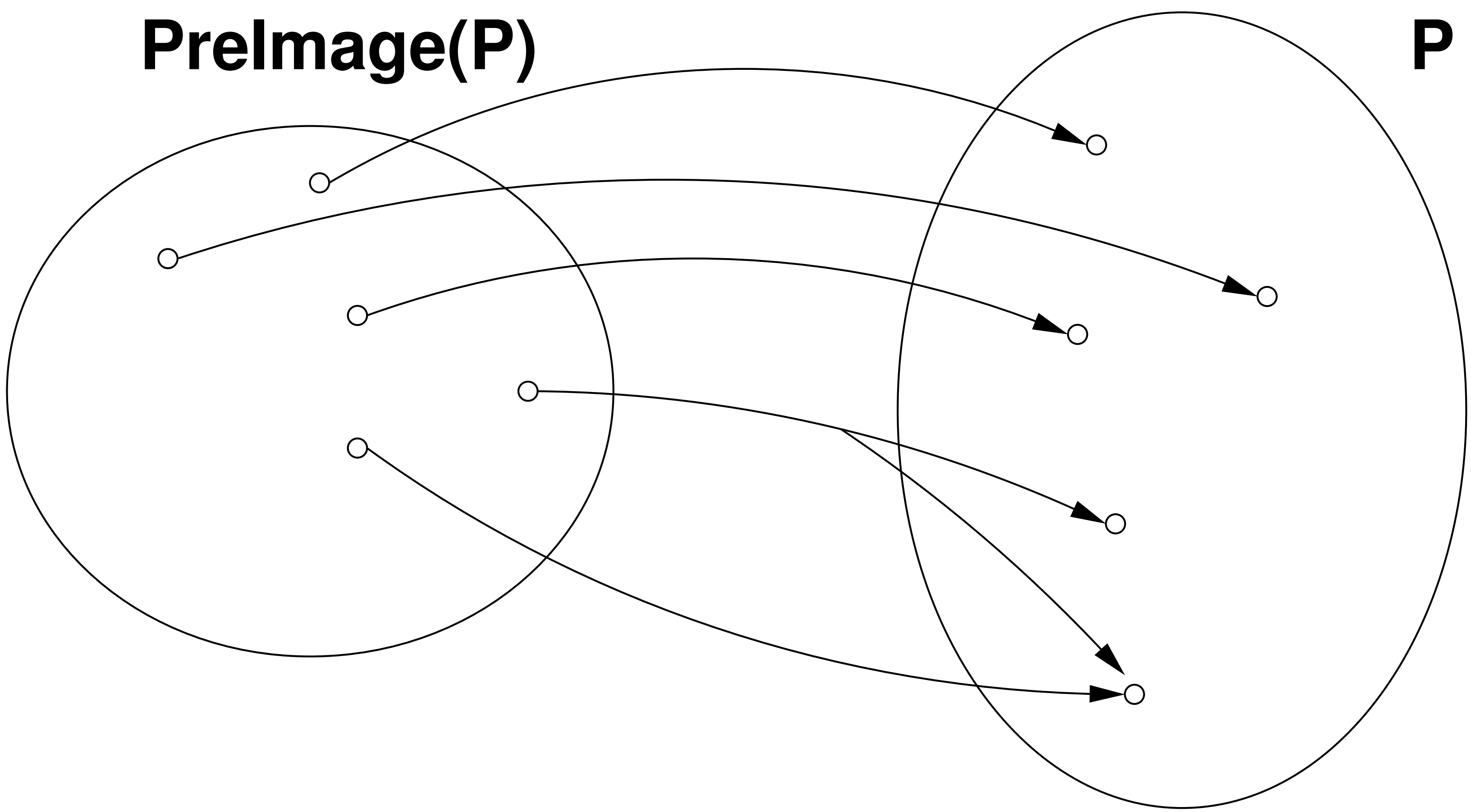
Denotation of Formulas: The Boolean Case

Let $\mathcal{KM} = \langle S, I, R, L, \Sigma \rangle$ be a Kripke Model.

$$
\begin{array}{lcl}
[\![false]\!] & = & \{\} \\
[\![true]\!] & = & S \\
[\![p]\!] & = & \{s \mid p \in L(s)\} \\
[\![\neg \varphi_1]\!] & = & S \setminus [\![\varphi_1]\!] \\
[\![\varphi_1 \wedge \varphi_2]\!] & = & [\![\varphi_1]\!] \cap [\![\varphi_2]\!]
\end{array}
$$

Denotation of Formulas: The $\Diamond_{P} \bigcirc$ Case

- $[\![\Diamond_{P} \bigcirc \varphi]\!] = \{s \in S \mid \exists s'.\langle s, s'\rangle \in R \text{ and } s' \in [\![\varphi]\!]\}$
- $[\![\Diamond_{P} \bigcirc \varphi]\!]$ is said to be the Pre-image of $[\![\varphi]\!]$ ($\textsc{Pre}([\![\varphi]\!])$).
- Key step of every CTL M.C. operation.



Denotation of Formulas: The $\Diamond_P \Box$ Case

- From the semantics of the $\Box$ temporal operator:

$$\Box\varphi \equiv \varphi \wedge \bigcirc(\Box\varphi)$$

- Then, the following equivalence holds:

$$\Diamond_P \Box\varphi \equiv \varphi \wedge \Diamond_P \bigcirc(\Diamond_P \Box\varphi)$$

- To compute $[\![\Diamond_P \Box\varphi]\!]$ we can apply the following recursive definition:

$$[\![\Diamond_P \Box\varphi]\!] = [\![\varphi]\!] \cap \textsc{Pre}([\![\Diamond_P \Box\varphi]\!])$$

Denotation of Formulas: The $\Diamond_P \Box$ Case (Cont

- We can compute $X := [\![\Diamond_P \Box \varphi]\!]$ inductively as follows:

$$\begin{array}{lcl} X_1 & := & [\![\varphi]\!] \\ X_2 & := & X_1 \cap \textsc{Pre}(X_1) \\ \ldots & & \\ X_{j+1} & := & X_j \cap \textsc{Pre}(X_j) \end{array}$$

- When $X_n = X_{n+1}$ we reach a fixpoint and we stop.
- **Termination.** Since $X_{j+1} \subseteq X_j$ for every $j \geq 0$, thus a fixed point always exists (Knaster-Tarski's theorem).

Denotation of Formulas: The $\Diamond_P\ \mathcal{U}$ Case

- From the semantics of the $\mathcal{U}$ temporal operator:

  $$\varphi\,\mathcal{U}\,\psi \equiv \psi \vee (\varphi \wedge \bigcirc(\varphi\,\mathcal{U}\,\psi))$$

- Then, the following equivalence holds:

  $$\Diamond_P(\varphi\,\mathcal{U}\,\psi) \equiv \psi \vee (\varphi \wedge \Diamond_P \bigcirc \Diamond_P(\varphi\,\mathcal{U}\,\psi))$$

- To compute $[\![\Diamond_P(\varphi\,\mathcal{U}\,\psi)]\!]$ we can apply the following recursive definition:

  $$[\![\Diamond_P(\varphi\,\mathcal{U}\,\psi)]\!] = [\![\psi]\!] \cup ([\![\varphi]\!] \cap \textsc{Pre}([\![\Diamond_P(\varphi\,\mathcal{U}\,\psi)]\!]))$$

- We can compute $X := [\![\Diamond_P (\varphi\, \mathcal{U}\, \psi)]\!]$ inductively as follows:

$$
\begin{aligned}
X_1 &:= [\![\psi]\!] \\
X_2 &:= X_1 \cup ([\![\varphi]\!] \cap \textsc{Pre}(X_1)) \\
&\dots \\
X_{j+1} &:= X_j \cup ([\![\varphi]\!] \cap \textsc{Pre}(X_j))
\end{aligned}
$$

- When $X_n = X_{n+1}$ we reach a fixpoint and we stop.
- **Termination.** Since $X_{j+1} \supseteq X_j$ for every $j \geq 0$, thus a fixed point always exists (Knaster-Tarski's theorem).

The Pseudo-Code

We assume the Kripke Model to be a global variable:

FUNCTION Label(φ) {

case φ **of**

$true$:	**return** S;
$false$:	**return** $\{\}$;
an atom p:	**return** $\{s \in S \mid p \in L(s)\}$;
$\neg\varphi_1$:	**return** $S \setminus$ Label(φ_1);
$\varphi_1 \wedge \varphi_2$:	**return** Label(φ_1)$\cap$Label(φ_2);
$\Diamond_P \bigcirc \varphi_1$:	**return** PRE(Label(φ_1));
$\Diamond_P (\varphi_1 \, \mathcal{U} \, \varphi_2)$:	**return** Label_EU(Label(φ_1),Label(φ_2));
$\Diamond_P \Box \varphi_1$:	**return** Label_EG(Label(φ_1));

end case

}

PreImage

$[\![\langle P \rangle \bigcirc \varphi]\!] = \textsc{Pre}([\![\varphi]\!]) = \{s \in S \mid \exists s'.\langle s, s' \rangle \in R \text{ and } s' \in [\![\varphi]\!]\}$

$\textsc{Function Pre}([\![\varphi]\!])\{$

var X;

$X := \{\}$;

for each $s' \in [\![\varphi]\!]$ **do**

for each $s \in S$ **do**

if $\langle s, s' \rangle \in R$ **then**

$X := X \cup \{s\}$;

return X

$\}$

Label_EG

$[\![\langle P \rangle \square \varphi]\!] = [\![\varphi]\!] \cap \textsc{Pre}([\![\langle P \rangle \square \varphi]\!])$

$\textsc{Function Label\_EG}([\![\varphi]\!])\{$

var $X, OLD\text{-}X;$

$X := [\![\varphi]\!];$

$OLD\text{-}X := \emptyset;$

while $X \neq OLD\text{-}X$

begin

$OLD\text{-}X := X;$

$X := X \cap \textsc{Pre}(X)$

end

return X

$\}$

Label_EU

$$[\![\langle\mathrm{P}\rangle(\varphi\,\mathcal{U}\,\psi)]\!] = [\![\psi]\!] \cup ([\![\varphi]\!] \cap \textsc{Pre}([\![\langle\mathrm{P}\rangle(\varphi\,\mathcal{U}\,\psi)]\!]))$$

FUNCTION LABEL_EU($[\![\varphi]\!]$,$[\![\psi]\!]$){

  var $X, \mathit{OLD}\text{-}X$;

  $X := [\![\psi]\!]$;

  $\mathit{OLD}\text{-}X := S$;

  while $X \neq \mathit{OLD}\text{-}X$

  begin

    $\mathit{OLD}\text{-}X := X$;

    $X := X \cup ([\![\varphi]\!] \cap \textsc{Pre}(X))$

  end

  return X

}

Summary

- CTL Model Checking: General Ideas.
- CTL Model Checking: The Labeling Algorithm.
- Labeling Algorithm in Details.
- CTL Model Checking: Theoretical Issues.

Correctness and Termination

- The Labeling algorithm works recursively on the structure φ.
- For most of the logical constructors the algorithm does the correct things according to the semantics of CTL.
- To prove that the algorithm is *Correct* and *Terminating* we need to prove the correctness and termination of both $\langle P \rangle \Box$ and $\langle P \rangle \mathcal{U}$ operators.

Monotone Functions and Fixpoints

Definition. Let S be a set and F a function, $F : 2^S \rightarrow 2^S$, then:

1. F is monotone iff $X \subseteq Y$ then $F(X) \subseteq F(Y)$;
2. A subset X of S is called a fixpoint of F iff $F(X) = X$;
3. X is a least fixpoint (LFP) of F, written $\mu X.F(X)$, iff, for every other fixpoint Y of F, $X \subseteq Y$
4. X is a greatest fixpoint (GFP) of F, written $\nu X.F(X)$, iff, for every other fixpoint Y of F, $Y \subseteq X$

Example. Let $S = \{s_0, s_1\}$ and $F(X) = X \cup \{s_0\}$.

Knaster-Tarski Theorem

Notation: $F^i(X)$ means applying F i-times, i.e., $F(F(\ldots F(X)\ldots))$.

Theorem[Knaster-Tarski]. Let S be a finite set with $n+1$ elements. If $F : 2^S \to 2^S$ is a monotone function then:

1. $\mu X.F(X) \equiv F^{n+1}(\emptyset)$;
2. $\nu X.F(X) \equiv F^{n+1}(S)$.

Proof. (See the textbook “Logic in CS” pg.241)

Correctness and Termination: $\Diamond \Box$ Case

The function LABEL_EG computes:

$$[\![\Diamond \Box \varphi]\!] = [\![\varphi]\!] \cap \textsc{Pre}([\![\Diamond \Box \varphi]\!])$$

applying the semantic equivalence:

$$\Diamond \Box \varphi \equiv \varphi \wedge \Diamond \bigcirc (\Diamond \Box \varphi)$$

Thus, $[\![\Diamond \Box \varphi]\!]$ is the fixpoint of the function:

$$F(X) = [\![\varphi]\!] \cap \textsc{Pre}(X)$$

Correctness and Termination: $\Diamond_{P} \Box$ Case (Con

Theorem. Let $F(X) = [\![\varphi]\!] \cap \textsc{Pre}(X)$, and let S have $n+1$ elements. Then:

1. F is monotone;
2. $[\![\Diamond_{P} \Box \varphi]\!]$ is the greatest fixpoint of F.

Proof. (See the textbook “Logic in CS” pg.242)

Correctness and Termination: $\Diamond_{P}\ \mathcal{U}$ Case

The function LABEL_EU computes:

$$[\![\Diamond_{P}(\varphi\,\mathcal{U}\,\psi)]\!] = [\![\psi]\!] \cup ([\![\varphi]\!] \cap \textsc{Pre}([\![\Diamond_{P}(\varphi\,\mathcal{U}\,\psi)]\!]))$$

applying the semantic equivalence:

$$\Diamond_{P}(\varphi\,\mathcal{U}\,\psi) \equiv \psi \vee (\varphi \wedge \Diamond_{P} \bigcirc \Diamond_{P}(\varphi\,\mathcal{U}\,\psi))$$

Thus, $[\![\Diamond_{P}(\varphi\,\mathcal{U}\,\psi)]\!]$ is the fixpoint of the function:

$$F(X) = [\![\psi]\!] \cup ([\![\varphi]\!] \cap \textsc{Pre}(X))$$

Correctness and Termination: $\Diamond_P\ \mathcal{U}$ Case (Con

Theorem. Let $F(X) = [\![\psi]\!] \cup ([\![\varphi]\!] \cap \textsc{Pre}(X))$, and let S have $n+1$ elements. Then:

1. F is monotone;
2. $[\![\Diamond_P(\varphi\, \mathcal{U}\, \psi)]\!]$ is the least fixpoint of F.

Proof. (See the textbook “Logic in CS” pg.243)

Summary of Lecture V

- CTL Model Checking: General Ideas.
- CTL Model Checking: The Labeling Algorithm.
- Labeling Algorithm in Details.
- CTL Model Checking: Theoretical Issues.