### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

chapters...

ntroductio

How to communicate Syntax of CTL Semantics of CTL

Some examples of what we can say

More about semantics

Equivalences Improving our

# Computation Tree Logic

Luis Tertulino & Ronaldo Silveira

October 22, 2015

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

what we can sav

1 In previous chapters...

Introduction

How to communicate

Syntax of CTL

Semantics of CTL

4 Some examples of what we can say

More about semantics

Equivalences

Improving our language

### Computation Tree Logic

Luis Tertulino & Ronaldo Silveira

#### In previous chapters...

Semantics of CTL

what we can sav

Improving our

■ A brief introduction to Propositional Logic, its syntax and its semantics

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

## In previous chapters...

cnapters...

How to communicate Syntax of CTI Semantics of

Semantics of CTL Some examples of what we can say

More about semantics Equivalences

Improving our language

- A brief introduction to Propositional Logic, its syntax and its semantics
- Formal models of time

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

# In previous chapters...

chapters..

How to communicate Syntax of CTL Semantics of

Some examples o what we can say More about semantics

semantics Equivalences Improving our

- A brief introduction to Propositional Logic, its syntax and its semantics
- Formal models of time
  - Frames and Flows of time

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

#### In previous chapters...

- A brief introduction to Propositional Logic, its syntax and its semantics
- Formal models of time
  - Frames and Flows of time
- Temporal Logic extends the Propositional Logic
  - $\blacksquare$  The connectives H and G

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

#### In previous chapters...

- A brief introduction to Propositional Logic, its syntax and its semantics
- Formal models of time
  - Frames and Flows of time
- Temporal Logic extends the Propositional Logic
  - $\blacksquare$  The connectives H and G
- Some practical applications

# Motivation

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

.

#### chapters...

Introduction

Syntax of CTL
Semantics of CTL

Some examples what we can sa More about semantics

semantics Equivalences

Improving our language

■ Needing of uncertainty;

# Motivation

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous

#### Introduction

How to communicate Syntax of CTL Semantics of CTL

What we can sa More about semantics

Equivalences Improving our

- Needing of uncertainty;
- Different paths of the future;

# Intuition

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

la annidana

Introducti

#### Introduction

How to communicate Syntax of CTL Semantics of CTL

what we can sa More about semantics

Equivalences Improving our In Computation Tree Logic (CTL) the model of time is a tree-like structure. This way, we cannot use Linear Temporal Logic (LTL) to express the existence of a certain path of time in which some event occurs.

# History

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Introduction

Improving our

# CTL was defined by:



Figure 1: Mordechai Ben-Ari



Figure 2: Amir Pnueli



Figure 3: Zohar Manna

# History

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Introduction

Semantics of CTL

Improving our

# And, at the same time by:



Figure 4: Ernest Allen Emerson



Figure 5: Edmund Clarke

# Syntax Definition

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Syntax of CTL

Semantics of CTL

what we can sav

The syntax of CTL consists on the syntax of temporal logic plus some path operators. The class of formulas can be defined in Backus-Naur form. If  $\phi$  is a formula:

# Syntax Definition

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Syntax of CTL

what we can sav

The syntax of CTL consists on the syntax of temporal logic plus some path operators. The class of formulas can be defined in Backus-Naur form. If  $\phi$  is a formula:

$$\phi ::= \bot \mid \top \mid p \mid \neg \phi \mid \phi \land \phi \mid \phi \lor \phi \mid \phi \to \phi \mid AX\phi \mid EX\phi \mid$$
$$AF\phi \mid EF\phi \mid AG\phi \mid EG\phi \mid A[\phi U\phi] \mid E[\phi U\phi]$$

# Syntax Definition

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Syntax of CTL

what we can sav

Improving our

The syntax of CTL consists on the syntax of temporal logic plus some path operators. The class of formulas can be defined in Backus-Naur form. If  $\phi$  is a formula:

$$\phi ::= \bot \mid \top \mid p \mid \neg \phi \mid \phi \land \phi \mid \phi \lor \phi \mid \phi \to \phi \mid AX\phi \mid EX\phi \mid$$
$$AF\phi \mid EF\phi \mid AG\phi \mid EG\phi \mid A[\phi U\phi] \mid E[\phi U\phi]$$

With p as a literal (atomic formula), AX, EX, AF, EF, AG e EG unary operators.

### Computation Tree Logic

Luis Tertulino & Ronaldo Silveira

Syntax of CTL

Semantics of CTL

Improving our

The propositional operators:  $\neg, \lor, \land, \rightarrow$  have the same meaning of in the propositional logic.

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Syntax of CTL

Semantics of CTL

what we can sav

Improving our

The propositional operators:  $\neg, \lor, \land, \rightarrow$  have the same meaning of in the propositional logic.

The path-specific operators can be read as:

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

Introduct

Syntax of CTL Semantics of

CTL Some examples o what we can say

More about semantics Equivalences The propositional operators:  $\neg$ ,  $\lor$ ,  $\land$ ,  $\rightarrow$  have the same meaning of in the propositional logic.

The path-specific operators can be read as:

■ *A*: is the universal quantifier over paths. Read as: "in all possible paths";

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Syntax of CTL

what we can sav

The propositional operators:  $\neg, \lor, \land, \rightarrow$  have the same meaning of in the propositional logic.

The path-specific operators can be read as:

- A: is the universal quantifier over paths. Read as: "in all possible paths";
- E:is the existential quantifier over paths. Read as: "exists a path in which";

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Syntax of CTL

what we can sav

The propositional operators:  $\neg, \lor, \land, \rightarrow$  have the same meaning of in the propositional logic.

The path-specific operators can be read as:

- A: is the universal quantifier over paths. Read as: "in all possible paths";
- E:is the existential quantifier over paths. Read as: "exists a path in which";

The temporal operators, as in LTL, can be read as:

■ X: "in the next state";

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicat

Syntax of CTL Semantics of CTL

what we can say
More about
semantics
Equivalences
Improving our

The propositional operators:  $\neg$ ,  $\lor$ ,  $\land$ ,  $\rightarrow$  have the same meaning of in the propositional logic.

The path-specific operators can be read as:

- A: is the universal quantifier over paths. Read as: "in all possible paths";
- *E*:is the existential quantifier over paths. Read as: "exists a path in which";

The temporal operators, as in LTL, can be read as:

- X: "in the next state";
- $\blacksquare$  F "There is some state in the future (eventually)";

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Syntax of CTL

what we can sav Improving our

The propositional operators:  $\neg, \lor, \land, \rightarrow$  have the same meaning of in the propositional logic.

The path-specific operators can be read as:

- A: is the universal quantifier over paths. Read as: "in all possible paths";
- E:is the existential quantifier over paths. Read as: "exists a path in which";

The temporal operators, as in LTL, can be read as:

- X: "in the next state";
- F "There is some state in the future (eventually)";
- G "Globally (in all future states)";

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communic

Syntax of CTL Semantics of CTL

Some examples of what we can say More about semantics Equivalences Improving our

The propositional operators:  $\neg$ ,  $\lor$ ,  $\land$ ,  $\rightarrow$  have the same meaning of in the propositional logic.

The path-specific operators can be read as:

- A: is the universal quantifier over paths. Read as: "in all possible paths";
- *E*:is the existential quantifier over paths. Read as: "exists a path in which";

The temporal operators, as in LTL, can be read as:

- X: "in the next state";
- $\blacksquare$  F "There is some state in the future (eventually)";
- *G* "Globally (in all future states)";
- $\blacksquare \varphi U\psi$ :  $\varphi$  is true at least until  $\psi$  becomes true;

# Syntax Notes

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Syntax of CTL

what we can sav

■ Notice that, in CTL, the combination of path specific operators and temporal operators are atomic, e.g., AF is an atomic operator that can be read as "In all paths in the future there is some state where...":

# Syntax Notes

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Syntax of CTL

what we can sav Improving our

- Notice that, in CTL, the combination of path specific operators and temporal operators are atomic, e.g., AF is an atomic operator that can be read as "In all paths in the future there is some state where...":
- Notice as well that the binary operators  $A[\varphi U\psi]$  and  $E[\varphi U\psi]$  can be represented as AU and EU, respectively;

# Syntax Notes

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Syntax of CTL

- Notice that, in CTL, the combination of path specific operators and temporal operators are atomic, e.g., AF is an atomic operator that can be read as "In all paths in the future there is some state where...":
- Notice as well that the binary operators  $A[\varphi U\psi]$  and  $E[\varphi U\psi]$  can be represented as AU and EU, respectively;
- We assume that, similarly to the ¬ operator, the "new" unary operators (AX, EX, AF, EF, AG, and EG) have the first precedence. Next comes the  $\wedge$  and  $\vee$  operators. And at last the  $\rightarrow$ , AU and EU;

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

Introduction

communicat

Syntax of CTL Semantics of CTL

Some examples of what we can say More about semantics Equivalences

Improving our language

- Examples of well-formed formulas:
  - $\blacksquare \ \textit{AG}(p \lor \textit{EFq})$

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

chapters...

Introduction

communicate

Syntax of CTL Semantics of CTL

Some examples of what we can say More about semantics

Equivalences Improving our ■ Examples of well-formed formulas:

- $AG(p \lor EFq)$
- $\blacksquare \ AX(q \to E[(p \lor q)Ur])$

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

chapters...

Introduction

Syntax of CTL

#### Syntax of CT Semantics of CTL

what we can say
More about
semantics

Equivalences

■ Examples of well-formed formulas:

- $AG(p \lor EFq)$
- $AX(q \to E[(p \lor q)Ur])$
- $EFEGp \rightarrow AFr$  Note that this is binded as  $(EFEGp) \rightarrow AFr$ , not as  $EFEG(p \rightarrow AFr)$

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Syntax of CTL

- Examples of well-formed formulas:
  - $\blacksquare$   $AG(p \lor EFq)$
  - $\blacksquare$   $AX(q \rightarrow E[(p \lor q)Ur])$
  - $EFEGp \rightarrow AFr$  Note that this is binded as  $(EFEGp) \rightarrow AFr$ , not as  $EFEG(p \rightarrow AFr)$
- Example of formulas that are not well-formed:
  - $\blacksquare A \neg G \neg p$

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Syntax of CTL

- Examples of well-formed formulas:
  - $\blacksquare$   $AG(p \lor EFq)$
  - $\blacksquare$   $AX(q \rightarrow E[(p \lor q)Ur])$
  - $EFEGp \rightarrow AFr$  Note that this is binded as  $(EFEGp) \rightarrow AFr$ , not as  $EFEG(p \rightarrow AFr)$
- Example of formulas that are not well-formed:
  - $\blacksquare A \neg G \neg p$
  - $\blacksquare F[pUs]$

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Syntax of CTL

- Examples of well-formed formulas:
  - $\blacksquare$  AG(p  $\lor$  EFq)
  - $\blacksquare$   $AX(q \rightarrow E[(p \lor q)Ur])$
  - $EFEGp \rightarrow AFr$  Note that this is binded as  $(EFEGp) \rightarrow AFr$ , not as  $EFEG(p \rightarrow AFr)$
- Example of formulas that are not well-formed:
  - $\blacksquare A \neg G \neg p$
  - $\blacksquare F[pUs]$
  - $\blacksquare$   $A[pUs \land qUs]$

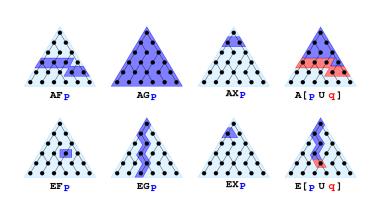
# Semantics Intuition of semantics

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Semantics of CTL

Improving our language



# Semantics Definition of model

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTL Semantics of CTL

Some examples o what we can say

More about semantics Equivalences

Equivalences

Improving language

## Definition

Let Atoms be a set of atomic formulas. A **transition system** or **model**  $\mathcal{M}$  is a triple  $\mathcal{M}=(S,\to,L)$  in which S is a set of states,  $\to$  is a binary relation over S ( $\to\subseteq S\times S$ ) such that for every state  $s\in S$ , exists a s' that  $s\to s'$  and  $L:S\to \mathcal{P}(Atoms)$  (or  $L:S\to (Atoms\to \{0,1\})$ ) is a labelling function.

# Semantics Definition of model

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTL Semantics of CTL

More about semantics

Fauivalences

Equivalences Improving our

## Definition

Let Atoms be a set of atomic formulas. A **transition system** or **model**  $\mathcal{M}$  is a triple  $\mathcal{M}=(S,\to,L)$  in which S is a set of states,  $\to$  is a binary relation over S ( $\to\subseteq S\times S$ ) such that for every state  $s\in S$ , exists a s' that  $s\to s'$  and  $L:S\to \mathcal{P}(Atoms)$  (or  $L:S\to (Atoms\to \{0,1\})$ ) is a labelling function.

CTL formulas are satisfied by a transition system and a specific state.

**Notation:** we will use  $\mathcal{M}, s \vDash \varphi$  to denote that the model  $\mathcal{M}, s$  satisfies the formula  $\varphi$ 

# Semantics Satisfaction

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Semantics of CTL

what we can sav

Improving our

# Definition

The **satisfaction** of a formula in CTL is recursive over the structure of the formula. It can be done as follows:

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTL Semantics of CTL

Some examples of what we can say More about semantics

More about semantics Equivalences Improving our Take an arbitrary model  $\mathcal{M}$ . Let  $s, s_1, s_2, s_3$  be states in S. Let  $\varphi, \varphi_1, \varphi_2$  be well-formed formulas of CTL. And let p be an atom. The satisfaction of CTL formulas can be defined as follows:

■  $\mathcal{M}$ ,  $s \vDash \top$  and  $\mathcal{M}$ ,  $s \not\vDash \bot$  for all  $s \in S$ 

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTL Semantics of CTL

What we can say
More about
semantics
Equivalences

Equivalences Improving our

- $\mathcal{M}, s \vDash \top$  and  $\mathcal{M}, s \not\vDash \bot$  for all  $s \in S$
- $\mathcal{M}$ ,  $s \models p$  iff  $p \in L(S)$

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Semantics of

what we can sav

- $\mathcal{M}, s \vDash \top$  and  $\mathcal{M}, s \not\vDash \bot$  for all  $s \in S$
- $\blacksquare \mathcal{M}, s \models p \text{ iff } p \in L(S)$
- $\blacksquare \mathcal{M}, s \vDash \neg \varphi \text{ iff } \mathcal{M}, s \not\vDash \varphi$

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Semantics of

what we can sav

- $\blacksquare \mathcal{M}, s \vDash \top \text{ and } \mathcal{M}, s \not\vDash \bot \text{ for all } s \in S$
- $\blacksquare \mathcal{M}, s \models p \text{ iff } p \in L(S)$
- $\blacksquare \mathcal{M}, s \vDash \neg \varphi \text{ iff } \mathcal{M}, s \nvDash \varphi$
- $\blacksquare \mathcal{M}, s \vDash \varphi_1 \land \varphi_2 \text{ iff } \mathcal{M}, s \vDash \varphi_1 \text{ AND } \mathcal{M}, s \vDash \varphi_2$

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Semantics of

what we can sav

- $\blacksquare \mathcal{M}, s \vDash \top \text{ and } \mathcal{M}, s \not\vDash \bot \text{ for all } s \in S$
- $\blacksquare \mathcal{M}, s \models p \text{ iff } p \in L(S)$
- $\blacksquare \mathcal{M}, s \vDash \neg \varphi \text{ iff } \mathcal{M}, s \nvDash \varphi$
- $\blacksquare \mathcal{M}, s \vDash \varphi_1 \land \varphi_2 \text{ iff } \mathcal{M}, s \vDash \varphi_1 \text{ AND } \mathcal{M}, s \vDash \varphi_2$
- $\blacksquare \mathcal{M}, s \models \varphi_1 \lor \varphi_2 \text{ iff } \mathcal{M}, s \models \varphi_1 \text{ OR } \mathcal{M}, s \models \varphi_2$

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...
Introduction

How to communicate Syntax of CTI Semantics of

Some examples of what we can say More about semantics

semantics Equivalences

- $\mathcal{M}, s \vDash \top$  and  $\mathcal{M}, s \not\vDash \bot$  for all  $s \in S$
- $\mathcal{M}$ ,  $s \models p$  iff  $p \in L(S)$
- $\blacksquare \mathcal{M}, s \vDash \neg \varphi \text{ iff } \mathcal{M}, s \nvDash \varphi$
- $\mathcal{M}, s \vDash \varphi_1 \land \varphi_2$  iff  $\mathcal{M}, s \vDash \varphi_1$  AND  $\mathcal{M}, s \vDash \varphi_2$
- $\mathcal{M}$ ,  $s \vDash \varphi_1 \lor \varphi_2$  iff  $\mathcal{M}$ ,  $s \vDash \varphi_1$  OR  $\mathcal{M}$ ,  $s \vDash \varphi_2$
- $\blacksquare \mathcal{M}, s \vDash \varphi_1 \rightarrow \varphi_2 \text{ iff } \mathcal{M}, s \not\vDash \varphi_1 \text{ OR } \mathcal{M}, s \vDash \varphi_2$

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...
Introductio

How to communicate Syntax of CTL Semantics of CTL

Some examples of what we can say More about semantics

semantics Equivalences Improving our Take an arbitrary model  $\mathcal{M}$ . Let  $s, s_1, s_2, s_3$  be states in S. Let  $\varphi, \varphi_1, \varphi_2$  be well-formed formulas of CTL. And let p be an atom. The satisfaction of CTL formulas can be defined as follows:

■  $\mathcal{M}, s \vDash AX\varphi$  iff for all  $s_1$  that  $s \to s_1$  and  $\mathcal{M}, s_1 \vDash \varphi$ . Thus, AX says: "in every next state..."

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTI Semantics of CTL

what we can say
More about
semantics
Equivalences
Improving our

- $\mathcal{M}, s \vDash AX\varphi$  iff for all  $s_1$  that  $s \to s_1$  and  $\mathcal{M}, s_1 \vDash \varphi$ . Thus, AX says: "in every next state..."
- $\mathcal{M}, s \vDash EX\varphi$  iff exists  $s_1$  that  $s \to s_1$  and  $\mathcal{M}, s_1 \vDash \varphi$ . Thus, EX says: "in some next state…"

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTI Semantics of CTI

what we can say
More about
semantics
Equivalences
Improving our

- $\mathcal{M}, s \vDash AX\varphi$  iff for all  $s_1$  that  $s \to s_1$  and  $\mathcal{M}, s_1 \vDash \varphi$ . Thus, AX says: "in every next state..."
- $\mathcal{M}, s \vDash EX\varphi$  iff exists  $s_1$  that  $s \to s_1$  and  $\mathcal{M}, s_1 \vDash \varphi$ . Thus, EX says: "in some next state…"
- $\mathcal{M}$ , s,  $\vDash AG\varphi$  iff for all paths  $s_1 \to s_2 \to s_3 \to ...$  in which  $s = s_1$ , for all  $s_i$ ,  $\mathcal{M}$ ,  $s_i \vDash \varphi$ . Thus, AG says: "In all possible paths from now on in all next states..."

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTI Semantics of CTL

what we can say
More about
semantics
Equivalences
Improving our

- $\mathcal{M}, s \vDash AX\varphi$  iff for all  $s_1$  that  $s \to s_1$  and  $\mathcal{M}, s_1 \vDash \varphi$ . Thus, AX says: "in every next state..."
- $\mathcal{M}, s \vDash EX\varphi$  iff exists  $s_1$  that  $s \to s_1$  and  $\mathcal{M}, s_1 \vDash \varphi$ . Thus, EX says: "in some next state…"
- $\mathcal{M}, s, \models AG\varphi$  iff for all paths  $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow ...$  in which  $s = s_1$ , for all  $s_i$ ,  $\mathcal{M}, s_i \models \varphi$ . Thus, AG says: "In all possible paths from now on in all next states..."
- $\mathcal{M}$ , s,  $\vDash AG\varphi$  iff exists some path  $s_1 \to s_2 \to s_3 \to ...$  in which  $s = s_1$ , for all  $s_i$ ,  $\mathcal{M}$ ,  $s_i \vDash \varphi$  Thus, EG says: "Exists a path from now on in all next states..."

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Semantics of

what we can sav

Take an arbitrary model  $\mathcal{M}$ . Let  $s, s_1, s_2, s_3$  be states in S. Let  $\varphi, \varphi_1, \varphi_2$  be well-formed formulas of CTL. And let p be an atom. The satisfaction of CTL formulas can be defined as follows:

 $\blacksquare \mathcal{M}, s, \models AF\varphi$  iff for all paths  $s_1 \to s_2 \to s_3 \to \dots$  in which  $s = s_1$ , exists  $s_i$ ,  $\mathcal{M}, s_i \models \varphi$ . Thus, AF says: "In all possible paths from now on, in some next state..."

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTI Semantics of CTL

Some examples of what we can say More about semantics

semantics
Equivalences
Improving our

- $\mathcal{M}$ , s,  $\vDash AF\varphi$  iff for all paths  $s_1 \to s_2 \to s_3 \to ...$  in which  $s = s_1$ , exists  $s_i$ ,  $\mathcal{M}$ ,  $s_i \vDash \varphi$ . Thus, AF says: "In all possible paths from now on, in some next state..."
- $\mathcal{M}$ , s,  $\vDash EF\varphi$  iff exists some path  $s_1 \to s_2 \to s_3 \to ...$  in which  $s = s_1$ , that exists  $s_i$ ,  $\mathcal{M}$ ,  $s_i \vDash \varphi$ . Thus, EF says: "In some path from now on, in some next state..."

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CT Semantics of

Some examples of what we can say More about semantics

semantics Equivalences Improving our language

- $\mathcal{M}$ , s,  $\vDash AF\varphi$  iff for all paths  $s_1 \to s_2 \to s_3 \to ...$  in which  $s = s_1$ , exists  $s_i$ ,  $\mathcal{M}$ ,  $s_i \vDash \varphi$ . Thus, AF says: "In all possible paths from now on, in some next state..."
- $\mathcal{M}$ , s,  $\vDash EF\varphi$  iff exists some path  $s_1 \to s_2 \to s_3 \to ...$  in which  $s = s_1$ , that exists  $s_i$ ,  $\mathcal{M}$ ,  $s_i \vDash \varphi$ . Thus, EF says: "In some path from now on, in some next state..."
- $\mathcal{M}, s, \models A[\varphi_1 U \varphi_2]$  iff for all paths  $s_1 \to s_2 \to s_3 \to ...$  in which  $s = s_1$ , this path satisfies  $\varphi_1 U \varphi_2$ , i.e., exists  $s_i$  in the path such that  $\mathcal{M}, s_i \models \varphi_2$  and, for all j < i,  $\mathcal{M}, s_j \models \varphi_1$ . Thus, AU says: "For all paths from now on, until some state..."

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Semantics of CTL

what we can sav

Take an arbitrary model  $\mathcal{M}$ . Let  $s, s_1, s_2, s_3$  be states in S. Let  $\varphi, \varphi_1, \varphi_2$  be well-formed formulas of CTL. And let p be an atom. The satisfaction of CTL formulas can be defined as follows:

■  $\mathcal{M}$ , s,  $\models E[\varphi_1 U \varphi_2]$  iff exists some path  $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow ...$ in which  $s = s_1$ , this path satisfies  $\varphi_1 U \varphi_2$ . Thus, EU says: "In some path from now on, until some state..."

# Examples

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTL Semantics of CTL

# Some examples of what we can say More about

semantics Equivalences Improving our ■ "It's possible to get to a state where something has started but it's not ready":  $EF(started \land \neg ready)$ 

# **Examples**

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

## Some examples of what we can say

- "It's possible to get to a state where something has started but it's not ready":  $EF(started \land \neg ready)$
- "A certain process is enabled infinitely often on every computation path": AG(AFenabled)

# **Examples**

#### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

## Some examples of what we can sav

- "It's possible to get to a state where something has started but it's not ready":  $EF(started \land \neg ready)$
- "A certain process is enabled infinitely often on every computation path": AG(AFenabled)
- "An upwards travelling lift at the second floor does not change its direction when it has passengers wishing to go to the fifth floor":

 $AG(floor2 \land directionup \land button5 \rightarrow A[directionup Ufloor5])$ 

# Equivalences

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters... Introduction

How to communicate Syntax of CTI Semantics of CTL

Some examples o what we can say More about

## Equivalences

Improving our language

# Definition

Two CTL formulas  $\varphi$  and  $\psi$  are said to be **semantically equivalent** if any state in any model which satisfies one of them also satisfies the other;

# Equivalences

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...
Introduction

How to communicate Syntax of CTL Semantics of

Some examples o what we can say More about

## Equivalences

Improving our language

## Definition

Two CTL formulas  $\varphi$  and  $\psi$  are said to be **semantically equivalent** if any state in any model which satisfies one of them also satisfies the other;

Notation: we denote the semantic equivalence of  $\varphi$  and  $\psi$  by  $\varphi \equiv \psi$ 

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...
Introduction

How to communicate Syntax of CTL Semantics of CTL

Some examples what we can sa More about

#### Equivalences

Improving our language

$$\blacksquare \neg AF\varphi \equiv EG\neg \varphi$$

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...
Introduction

How to communicate Syntax of CTL Semantics of CTL

Some examples of what we can say More about

#### Equivalences

Improving our language

- $\blacksquare \neg AF\varphi \equiv EG\neg \varphi$
- $\blacksquare \neg \textit{EF}\varphi \equiv \textit{AG}\neg \varphi$

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTL Semantics of CTL

Some examples of what we can say More about

#### Equivalences

Improving our language

$$\blacksquare \ \neg \mathit{AF}\varphi \equiv \mathit{EG} \neg \varphi$$

$$\blacksquare \neg \mathit{EF}\varphi \equiv \mathit{AG}\neg\varphi$$

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTL Semantics of CTL

Some examples of what we can say More about

#### Equivalences

Improving our language

$$\blacksquare \ \neg \mathit{AF}\varphi \equiv \mathit{EG} \neg \varphi$$

$$\blacksquare \neg EF\varphi \equiv AG\neg \varphi$$

$$\blacksquare \neg AX\varphi \equiv EX\neg \varphi$$

$$AF\varphi \equiv A[\top U\varphi]$$

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTL Semantics of CTL

Some examples of what we can say More about

#### Equivalences

Improving our language

$$\blacksquare \neg AF\varphi \equiv EG\neg \varphi$$

$$\blacksquare \neg \mathit{EF}\varphi \equiv \mathit{AG}\neg \varphi$$

$$AF\varphi \equiv A[\top U\varphi]$$

$$\blacksquare \ \textit{EF}\varphi \equiv \textit{E}[\top \textit{U}\varphi]$$

# Minimum set of CTL connectives

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...
Introduction

How to communicate Syntax of CTL Semantics of CTI

Some examples o what we can say More about semantics

#### semantics Equivalences

Improving our

Because of the equivalences shown and the ones in propositional logic, we can have some minimum sets of conectives for the CTL syntax. One of them is defined in Backus-Naur formalism below:

$$\phi ::= \bot \mid p \mid \neg \phi \mid \phi \land \phi \mid EX\phi \mid AF\phi \mid E[\phi U\phi]$$

# That's all we need?

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTL Semantics of CTL

Some examples o what we can say More about semantics

semantics Equivalences Improving our language Even if CTL allow explicit quantification over paths, it cannot allow some expressions to be formed. For example, we cannot say, as in LTL: "All paths in which have p on them, also have q on them".

This expression can be translated in LTL as follows:

$$Fp \rightarrow Fq$$

# That's all we need?

#### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTL Semantics of CTL

What we can say
More about
semantics

Equivalences Improving our language We can try expressing it as  $AFp \rightarrow AFq$  but it does not have the same meaning. This one statement means "If all paths have a p along them, then all paths have a q along then"

# That's all we need?

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTL Semantics of CTI

what we can say
More about
semantics
Equivalences

Improving our language

We can try expressing it as  $AFp \to AFq$  but it does not have the same meaning. This one statement means "If all paths have a p along them, then all paths have a q along then" We can try to translate it as  $AG(p \to AFq)$  which is closer, but not exactly the same. This one means "for all paths, in all states on the future, if they hold p then, all paths will eventually hold q"

# Presenting CTL\*

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters... Introduction

How to communicate Syntax of CTL Semantics of CTL

what we can say
More about
semantics

Improving our language

For this, we can extend the CTL by dropping the constraint that every temporal operator (X, U, F, G) has to be associated with an unique path quantifier (A, E).

# Presenting CTL\*

### Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...
Introductio

How to communicate Syntax of CTL Semantics of CTL

what we can say
More about
semantics

Improving our language

For this, we can extend the CTL by dropping the constraint that every temporal operator  $(X,\ U,\ F,\ G)$  has to be associated with an unique path quantifier  $(A,\ E)$ .

This allows us to generate some statements:

# Presenting CTL\* Statements only possible with CTL\*

## Computation Tree Logic

Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTL Semantics of CTL

Some examples o what we can say More about semantics

More about semantics Equivalences

Equivalences Improving our language ■ "In all possible paths, q is true until r is true or p is true until r is true":  $A[qUr \lor pUr]$ 

# Presenting CTL\* Statements only possible with CTL\*

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...
Introduction

How to communicate Syntax of CTL Semantics of

More about semantics

Improving our language

- "In all possible paths, q is true until r is true or p is true until r is true":  $A[qUr \lor pUr]$
- "There is a path in which p eventually occurring will occur in all states": E[GFp]

# Presenting CTL\* Statements only possible with CTL\*

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters... Introduction

How to communicate Syntax of CTL Semantics of CTI

What we can say
More about
semantics
Equivalences

Improving our language

- "In all possible paths, q is true until r is true or p is true until r is true":  $A[qUr \lor pUr]$
- "There is a path in which p eventually occurring will occur in all states": E[GFp]
- "In all paths, p will occur in the next state or in the next of the next":  $A[Xp \lor XXp]$

# Presenting CTL\* CTL\* syntax

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTL Semantics of CTL

CTL Some examples o what we can say

More about semantics Equivalences

Improving our language

The syntax of CTL\* can be defined with the BNF bellow:

$$\phi ::= \bot \mid \top \mid p \mid \neg \phi \mid \phi \land \phi \mid \phi \lor \phi \mid \phi \rightarrow \phi \mid A[\alpha] \mid E[\alpha] \mid$$

$$\alpha ::= \phi | \ \neg \alpha \ | \ \alpha \wedge \alpha \ | \ \alpha \vee \alpha \ | \ \alpha \rightarrow \alpha \ | \ \alpha \textit{U}\alpha \ | \ \textit{G}\alpha \ | \ \textit{F}\alpha \ | \ \textit{X}\alpha |$$

With the same meanings of each operator.

# Presenting CTL\* LTL $\subset$ CTL\* and CTL $\subset$ CTL\*

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

In previous chapters...

How to communicate Syntax of CTL Semantics of CTL

Some examples of what we can say More about semantics

Equivalences
Improving our language

Although we don't define path operators to LTL we can assume that it consider in all paths. Therefore, we can say that a formula  $\phi$  in LTL is a formula  $A[\phi]$  in CTL\*;

# Presenting CTL\* LTL ⊂ CTL\* and CTL ⊂ CTL\*

## Computation Tree Logic

#### Luis Tertulino & Ronaldo Silveira

Semantics of CTL

what we can sav

Improving our language

Although we don't define path operators to LTL we can assume that it consider in all paths. Therefore, we can say that a formula  $\phi$  in LTL is a formula  $A[\phi]$  in CTL\*; For CTL. it is trivial: