



## Logic

Luca Campa   Philipp Dablander   Aaron Groß   Aart Middeldorp  
Alexander Montag   Johannes Niederhauser   Vera Schmitt

### Definitions

- ▶ **atomic formula**:  $P \mid P(t, \dots, t)$
- ▶ **literal** is atomic formula or negation of atomic formula
- ▶ **clause** is set of literals  $\{l_1, \dots, l_n\}$
- ▶ **clausal form** is set of clauses  $\{C_1, \dots, C_m\}$ , representing  $\forall (C_1 \wedge \dots \wedge C_m)$
- ▶ clauses  $C_1$  and  $C_2$  **without common variables clash** on literals  $l_1 \in C_1$  and  $l_2 \in C_2$  if  $l_1$  and  $l_2^c$  are unifiable
- ▶ **resolvent** of clauses  $C_1$  and  $C_2$  clashing on literals  $l_1 \in C_1$  and  $l_2 \in C_2$  is clause

$$((C_1 \setminus \{l_1\}) \cup (C_2 \setminus \{l_2\}))\theta$$

where  $\theta$  is mgu of  $l_1$  and  $l_2^c$

- ▶  $C\sigma$  is **factor** of  $C$  if two or more literals in  $C$  have mgu  $\sigma$

## Outline

1. Summary of Previous Lecture
2. Post's Adequacy Theorem
3. Intermezzo
4. Model Checking
5. Branching-Time Temporal Logic (CTL)
6. CTL Model Checking Algorithm
7. Further Reading

### Resolution with Factoring

input: clausal form  $S$

output: yes if  $S$  is satisfiable  
no if  $S$  is unsatisfiable  
 $\infty$  if  $S$  is satisfiable

- ① repeatedly add resolvents (renaming clauses if necessary) and factors
- ② return no as soon as empty clause  $\square$  is derived
- ③ return yes if all clashing clauses have been resolved and factoring produces no new clauses (modulo renaming)

### Theorem

resolution with factoring is sound and complete:  
clausal form  $S$  is unsatisfiable if and only if  $S$  admits refutation

### Decision Problem (Church's Theorem)

instance: set of formulas  $\Gamma$ , first-order formula  $\psi$

question:  $\Gamma \models \psi$  ?

is **undecidable** even when  $\Gamma = \emptyset$

### Definition

set  $X$  of boolean functions is called **adequate** or **functionally complete** if every boolean function can be expressed using functions from  $X$

### Theorem (Algebraic Normal Form)

every boolean function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  can be uniquely written as

$$f(x_1, \dots, x_n) = \bigoplus_{A \subseteq \{1, \dots, n\}} c_A \cdot \prod_{i \in A} x_i$$

with  $c_A \in \{0, 1\}$  for all  $A \subseteq \{1, \dots, n\}$

### Part I: Propositional Logic

algebraic normal forms, binary decision diagrams, conjunctive normal forms, DPLL, Horn formulas, natural deduction, **Post's adequacy theorem**, resolution, SAT, semantics, sorting networks, soundness and completeness, syntax, Tseitin's transformation

### Part II: Predicate Logic

natural deduction, quantifier equivalences, resolution, semantics, Skolemization, syntax, undecidability, unification

### Part III: Model Checking

adequacy, **branching-time temporal logic**, CTL\*, fairness, linear-time temporal logic, **model checking algorithms**, symbolic model checking

### Example

boolean function  $f(x_1, x_2, x_3, x_4) = x_1x_2 + \bar{x}_3 + x_2\bar{x}_4$

determine  $c_A$  for  $A \subseteq \{x_1, \dots, x_4\}$

A	$c_A$	A	$c_A$	A	$c_A$	A	$c_A$
$\emptyset$	1	$\{x_4\}$	1	$\{x_2, x_3\}$	1	$\{x_1, x_2, x_4\}$	1
$\{x_1\}$	1	$\{x_1, x_3\}$	1	$\{x_2, x_4\}$	1	$\{x_1, x_3, x_4\}$	0
$\{x_2\}$	1	$\{x_1, x_3\}$	0	$\{x_3, x_4\}$	0	$\{x_1, x_3, x_4\}$	0
$\{x_3\}$	0	$\{x_1, x_4\}$	1	$\{x_1, x_2, x_3\}$	1	$\{x_1, x_2, x_3, x_4\}$	1

$$f(0, 0, 0, 0) = 1$$

$$f(x_1, x_2, x_3, x_4) = 1 \oplus x_1 \oplus x_2 \oplus x_4 \oplus x_1x_2 \oplus x_1x_4 \oplus x_2x_3 \oplus x_2x_4 \oplus x_1x_2x_3 \oplus x_1x_2x_4 \oplus x_1x_2x_3x_4$$

### Outline

1. Summary of Previous Lecture
2. Post's Adequacy Theorem
3. Intermezzo
4. Model Checking
5. Branching-Time Temporal Logic (CTL)
6. CTL Model Checking Algorithm
7. Further Reading

### Theorem (Post's Adequacy Theorem)

set  $X$  of boolean functions is adequate if and only if following conditions hold:

- 1 there exists  $f \in X$  such that  $f(0, \dots, 0) \neq 0$
- 2 there exists  $f \in X$  such that  $f(1, \dots, 1) \neq 1$
- 3 there exists  $f \in X$  which is not **monotone**
- 4 there exists  $f \in X$  which is not **self-dual**
- 5 there exists  $f \in X$  which is not **affine**

### Definitions

boolean function  $f$  is

- ▶ **monotone** if  $f(x_1, \dots, x_n) \leq f(y_1, \dots, y_n)$  for all  $x_1 \leq y_1, \dots, x_n \leq y_n$
- ▶ **self-dual** if  $f(x_1, \dots, x_n) = \overline{f(\bar{x}_1, \dots, \bar{x}_n)}$
- ▶ **affine** if  $f(x_1, \dots, x_n) = c_0 \oplus c_1x_1 \oplus \dots \oplus c_nx_n$  for some  $c_0, \dots, c_n \in \{0, 1\}$

### Lemma

boolean function  $f$  is **not monotone** if and only if

$$f(b_1, \dots, b_{i-1}, x, b_{i+1}, \dots, b_n) = \bar{x} \quad \text{for all } x \in \{0, 1\}$$

for some  $i$  and  $b_1, \dots, b_{i-1}, b_{i+1}, \dots, b_n \in \{0, 1\}$

### Lemma

boolean function  $f$  is **not self-dual** if and only if

$$f(b_1, \dots, b_n) = f(\bar{b}_1, \dots, \bar{b}_n)$$

for some  $b_1, \dots, b_n \in \{0, 1\}$

### Remark

boolean function  $f$  is affine if and only if algebraic normal form of  $f$  is linear

### Examples

	-	·	+	=	⊕		0	1
$f(0, \dots, 0) \neq 0$	✓	×	×	✓	×	✓	×	✓
$f(1, \dots, 1) \neq 1$	✓	×	×	×	✓	✓	✓	×
not monotone	✓	×	×	✓	✓	✓	×	×
not self-dual	×	✓	✓	✓	✓	✓	✓	✓
not affine	×	✓	✓	×	×	✓	×	×

### Definitions

boolean function  $f$  is

- ▶ monotone if  $f(x_1, \dots, x_n) \leq f(y_1, \dots, y_n)$  for all  $x_1 \leq y_1, \dots, x_n \leq y_n$
- ▶ self-dual if  $f(x_1, \dots, x_n) = \overline{f(\bar{x}_1, \dots, \bar{x}_n)}$
- ▶ affine if  $f(x_1, \dots, x_n) = c_0 \oplus c_1x_1 \oplus \dots \oplus c_nx_n$  for some  $c_0, \dots, c_n \in \{0, 1\}$

### Theorem (Post's Adequacy Theorem)

set  $X$  of boolean functions is adequate if and only if following conditions hold:

- 1  $\exists f_1 \in X$  such that  $f_1(0, \dots, 0) \neq 0$
- 2  $\exists f_2 \in X$  such that  $f_2(1, \dots, 1) \neq 1$
- 3  $\exists f_3 \in X$  which is not monotone
- 4  $\exists f_4 \in X$  which is not self-dual
- 5  $\exists f_5 \in X$  which is not affine

### Proof ( $\Leftarrow$ )

- ▶ first task: define  $0, 1, \bar{x}$
- ▶ define  $g(x) = f_1(x, \dots, x)$  and  $h(x) = f_2(x, \dots, x)$
- ▶  $g(x) = 1$  or  $g(x) = \bar{x}$  and  $h(x) = 0$  or  $h(x) = \bar{x}$
- ▶ we distinguish four cases:
 

① $g(x) = 1$ and $h(x) = \bar{x}$	③ $g(x) = 1$ and $h(x) = 0$
② $g(x) = \bar{x}$ and $h(x) = 0$	④ $g(x) = \bar{x}$ and $h(x) = \bar{x}$

### Proof ( $\Leftarrow$ )

▶ first task: define  $0, 1, \bar{x}$

①  $g(x) = 1$  and  $h(x) = \bar{x}$      $h(g(x)) = 0$

②  $g(x) = \bar{x}$  and  $h(x) = 0$      $g(h(x)) = 1$

③  $g(x) = 1$  and  $h(x) = 0$

there exist  $i \in \{1, \dots, m\}$  and  $b_1, \dots, b_{i-1}, b_{i+1}, \dots, b_m \in \{0, 1\}$  such that

$$f_3(b_1, \dots, b_{i-1}, x, b_{i+1}, \dots, b_m) = \bar{x}$$

$b_j = g(x)$  or  $b_j = h(x)$  for  $j \neq i$

so  $\bar{x}$  is defined using  $f_3, g, h$

③ there exists  $f_3 \in X$  which is not monotone



### Proof ( $\Leftarrow$ )

▶ first task: define  $0, 1, \bar{x}$

④  $g(x) = \bar{x}$  and  $h(x) = \bar{x}$

there exists  $b_1, \dots, b_k \in \{0, 1\}$  such that  $f_4(\bar{b}_1, \dots, \bar{b}_k) = f_4(b_1, \dots, b_k)$

define  $i(x) = f_4(x \oplus b_1, \dots, x \oplus b_k)$

$x \oplus b_j = x$  or  $x \oplus b_j = \bar{x} = g(x)$ , so  $i(x)$  is defined using  $f_4$  and  $g$

$i(x) = 0$  or  $i(x) = 1$

$g(i(x)) = 1$  or  $g(i(x)) = 0$

④ there exists  $f_4 \in X$  which is not self-dual



### Proof ( $\Leftarrow$ )

▶ second task: define  $xy$

there exist  $g_1, g_2, g_3, g_4$  such that (wlog)

$$f_5(x_1, \dots, x_l) = x_1 x_2 g_1(x_3, \dots, x_l) \oplus x_1 g_2(x_3, \dots, x_l) \oplus x_2 g_3(x_3, \dots, x_l) \oplus g_4(x_3, \dots, x_l)$$

with  $g_1(x_3, \dots, x_l) \neq 0$

there exist  $c_3, \dots, c_l \in \{0, 1\}$  such that  $g_1(c_3, \dots, c_l) = 1$

define  $c = g_2(c_3, \dots, c_l)$ ,  $d = g_3(c_3, \dots, c_l)$ ,  $e = g_4(c_3, \dots, c_l)$

$$f_5(x_1, x_2, c_3, \dots, c_l) = x_1 x_2 \oplus x_1 c \oplus x_2 d \oplus e$$

define  $h(x, y) = f_5(x \oplus d, y \oplus c, c_3, \dots, c_l) \oplus cd \oplus e$

$$h(x, y) = (x \oplus d)(y \oplus c) \oplus (x \oplus d)c \oplus (y \oplus c)d \oplus e \oplus cd \oplus e = xy$$

⑤ there exists  $f_5 \in X$  which is not affine



### Remark

proof of "if direction" is **constructive**

### Demo

#### BoolTool

by Patrick Muxel (2004), Philipp Ruff (2006), Caroline Terzer (2006), Markus Plattner (2007), Elias Zischg (2012)

#### BoolTool Reloaded

by Martin Neuner (2023)

### Proof sketch ( $\Rightarrow$ )

▶ suppose  $X$  has no functions that satisfy condition ①

▶ claim: all functions constructed from  $X$  violate condition ①

▶  $X$  cannot be adequate because  $x|y$  cannot be expressed



## Outline

1. Summary of Previous Lecture
2. Post's Adequacy Theorem
- 3. Intermezzo**
4. Model Checking
5. Branching-Time Temporal Logic (CTL)
6. CTL Model Checking Algorithm
7. Further Reading

## Outline

1. Summary of Previous Lecture
2. Post's Adequacy Theorem
3. Intermezzo
- 4. Model Checking**
5. Branching-Time Temporal Logic (CTL)
6. CTL Model Checking Algorithm
7. Further Reading

### Question

Which of the following statements are true ?

- A** If  $f(x)$  and  $g(x_1, \dots, x_n)$  are self-dual then  $f(g(x_1, \dots, x_n))$  is self-dual.
- B** The function  $f(x, y) = \bar{x} + y$  is monotone.
- C** For any number of input variables  $n$ , there are  $2^{n+1}$  distinct affine boolean functions.
- D** The set of all monotone boolean functions is adequate.



### Formal Verification comprises

- ▶ **framework for modeling systems** (description language)
- ▶ **specification language** for describing properties to be verified
- ▶ **verification method** to establish whether description of system satisfies specification

### Model Checking

**automatic** formal verification approach for concurrent systems based on **temporal logic**

### Temporal Logic

- ▶ formulas are not statically true or false in model
- ▶ models of temporal logic contain several states and truth is **dynamic**
- ▶ formula can be true in some states and false in others

## Model Checking

- ▶ models are transition systems  $\mathcal{M}$
- ▶ properties are formulas  $\varphi$  in temporal logic
- ▶ model checker determines whether  $\mathcal{M} \models \varphi$  is true or not

## Two Temporal Logics

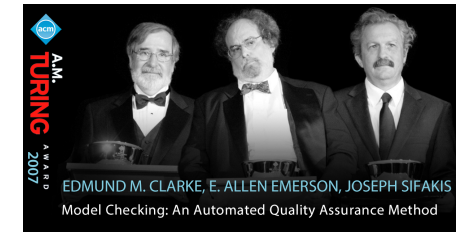
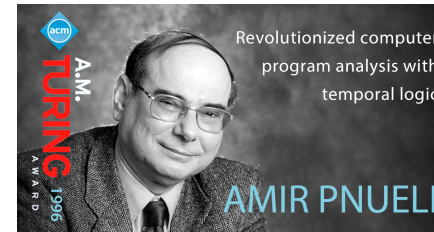
- ▶ computation tree logic (CTL) lectures 9 and 10
- ▶ linear-time temporal logic (LTL) lectures 10 and 11

## Impact

both logics have been proven to be **extremely fruitful** in verifying hardware and communication protocols, and are increasingly applied to software verification

## Outline

1. Summary of Previous Lecture
2. Post's Adequacy Theorem
3. Intermezzo
4. Model Checking
5. Branching-Time Temporal Logic (CTL)  
Syntax Semantics
6. CTL Model Checking Algorithm
7. Further Reading



## ACM Turing Awards

1996 Amir Pnueli

2007 Edmund M. Clarke, E. Allen Emerson, Joseph Sifakis

## Definition

- ▶ **CTL (computation tree logic)** formulas are built from
  - ▶ atoms  $p, q, r, p_1, p_2, \dots$
  - ▶ logical connectives  $\perp, \top, \neg, \wedge, \vee, \rightarrow$
  - ▶ **temporal connectives**  $AX, EX, AF, EF, AG, EG, AU, EU$

according to following BNF grammar:

$$\varphi ::= \perp \mid \top \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi) \mid (AX\varphi) \mid (EX\varphi) \mid (AF\varphi) \mid (EF\varphi) \mid (AG\varphi) \mid (EG\varphi) \mid A[\varphi U \varphi] \mid E[\varphi U \varphi]$$

- ▶ notational conventions:
  - ▶ binding precedence  $\neg, AX, EX, AF, EF, AG, EG > \wedge, \vee > \rightarrow, AU, EU$
  - ▶ omit outer parentheses
  - ▶  $\rightarrow, \wedge, \vee$  are right-associative

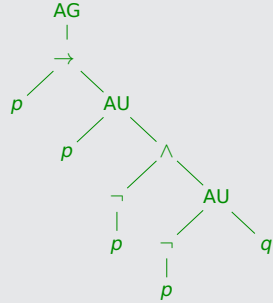
## Example

formula  $\neg A[EX p U \neg q]$

parse tree



formula  $AG(p \rightarrow A[p U \neg p \wedge A[\neg p U q]])$



A	$\forall$ paths	G	$\forall$ states globally	X	next state
E	$\exists$ path	F	$\exists$ state future	U	until

## Outline

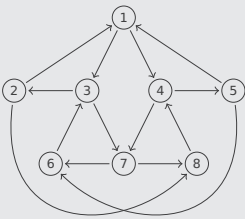
1. Summary of Previous Lecture
2. Post's Adequacy Theorem
3. Intermezzo
4. Model Checking
5. Branching-Time Temporal Logic (CTL)
  - Syntax
  - Semantics
6. CTL Model Checking Algorithm
7. Further Reading

## Definition

transition system (model) is triple  $\mathcal{M} = (S, \rightarrow, L)$  with

- ① set of states  $S$
- ② transition relation  $\rightarrow \subseteq S \times S$  such that  $\forall s \in S \exists t \in S$  with  $s \rightarrow t$  ("no deadlock")
- ③ labelling function  $L: S \rightarrow \mathcal{P}(\text{atoms})$

## Example



model  $\mathcal{M} = (S, \rightarrow, L)$

$S = \{1, 2, 3, 4, 5, 6, 7, 8\}$

$L(1) = \{I_A, I_B\}$	$L(5) = \{I_A, P_B\}$
$L(2) = \{P_A, I_B\}$	$L(6) = \{R_A, P_B\}$
$L(3) = \{R_A, I_B\}$	$L(7) = \{R_A, R_B\}$
$L(4) = \{I_A, R_B\}$	$L(8) = \{P_A, R_B\}$

## Definition

satisfaction of CTL formula  $\varphi$  in state  $s \in S$  of model  $\mathcal{M} = (S, \rightarrow, L)$

$$\mathcal{M}, s \models \varphi$$

is defined by induction on  $\varphi$ :

$\mathcal{M}, s \models \top$	$\mathcal{M}, s \not\models \perp$	$\mathcal{M}, s \models \varphi \wedge \psi \iff \mathcal{M}, s \models \varphi$ and $\mathcal{M}, s \models \psi$
$\mathcal{M}, s \models p \iff p \in L(s)$	$\mathcal{M}, s \models \varphi \vee \psi \iff \mathcal{M}, s \models \varphi$ or $\mathcal{M}, s \models \psi$	
$\mathcal{M}, s \models \neg \varphi \iff \mathcal{M}, s \not\models \varphi$	$\mathcal{M}, s \models \varphi \rightarrow \psi \iff \mathcal{M}, s \not\models \varphi$ or $\mathcal{M}, s \models \psi$	
$\mathcal{M}, s \models AX \varphi \iff \forall \text{ paths } s = s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \dots \mathcal{M}, s_2 \models \varphi$		
$\mathcal{M}, s \models EX \varphi \iff \exists \text{ path } s = s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \dots \mathcal{M}, s_2 \models \varphi$		
$\mathcal{M}, s \models AF \varphi \iff \forall \text{ paths } s = s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \dots \exists i \geq 1 \mathcal{M}, s_i \models \varphi$		
$\mathcal{M}, s \models EF \varphi \iff \exists \text{ path } s = s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \dots \exists i \geq 1 \mathcal{M}, s_i \models \varphi$		

## Definition (cont'd)

**satisfaction** of CTL formula  $\varphi$  in state  $s \in S$  of model  $\mathcal{M} = (S, \rightarrow, L)$

$$\mathcal{M}, s \models \varphi$$

is defined by induction on  $\varphi$ :

$$\mathcal{M}, s \models \text{AG } \varphi \iff \forall \text{ paths } s = s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \dots \quad \forall i \geq 1 \quad \mathcal{M}, s_i \models \varphi$$

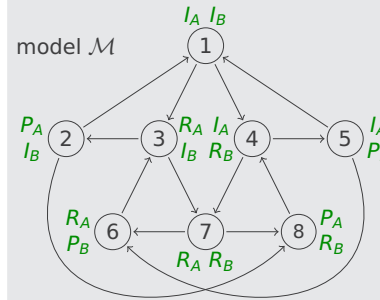
$$\mathcal{M}, s \models \text{EG } \varphi \iff \exists \text{ path } s = s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \dots \quad \forall i \geq 1 \quad \mathcal{M}, s_i \models \varphi$$

$$\mathcal{M}, s \models \text{A}[\varphi \text{U } \psi] \iff \forall \text{ paths } s = s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \dots \\ \exists i \geq 1 \quad \mathcal{M}, s_i \models \psi \quad \text{and} \quad \forall j < i \quad \mathcal{M}, s_j \models \varphi$$

$$\mathcal{M}, s \models \text{E}[\varphi \text{U } \psi] \iff \exists \text{ path } s = s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \dots \\ \exists i \geq 1 \quad \mathcal{M}, s_i \models \psi \quad \text{and} \quad \forall j < i \quad \mathcal{M}, s_j \models \varphi$$

## Example

model  $\mathcal{M}$



$$\mathcal{M}, 1 \not\models I_A \wedge R_B$$

$$\mathcal{M}, 1 \not\models I_B \rightarrow P_A \vee R_B$$

$$\mathcal{M}, 4 \models I_A \wedge R_B$$

$$\mathcal{M}, 2 \models I_B \rightarrow P_A \vee R_B$$

$$\mathcal{M}, 1 \models \text{AX}(R_A \vee R_B)$$

$$\mathcal{M}, 1 \not\models \text{EX } P_B$$

$$\mathcal{M}, 3 \not\models \text{AX } P_A$$

$$\mathcal{M}, 3 \models \text{EX } P_A$$

$$\mathcal{M}, 1 \models \text{AF}(R_A \vee R_B)$$

$$\mathcal{M}, 1 \models \text{EF}(R_A \wedge R_B)$$

$$\mathcal{M}, 5 \not\models \text{AF } R_B$$

$$\mathcal{M}, 5 \not\models \text{EF}(P_A \wedge P_B)$$

$$\mathcal{M}, 1 \models \text{AG}(R_A \rightarrow \text{EF } P_A)$$

$$\mathcal{M}, 2 \models \text{EG}(\neg P_A \rightarrow R_B)$$

$$\mathcal{M}, 1 \not\models \text{AG}(R_A \rightarrow \text{AF } P_A)$$

$$\mathcal{M}, 2 \not\models \text{EG } P_A$$

$$\mathcal{M}, 1 \models \neg \text{A}[P_A \text{U } P_A]$$

$$\mathcal{M}, 1 \models \text{EXE}[R_A \text{U } P_A]$$

$$\mathcal{M}, 7 \models \text{A}[P_A \text{U } R_A]$$

$$\mathcal{M}, 7 \not\models \text{E}[P_A \wedge P_B \text{U } I_A \vee I_B]$$

## Theorem

satisfaction of CTL formulas in finite models is **decidable**

## Definition

CTL formulas  $\varphi$  and  $\psi$  are **semantically equivalent** ( $\varphi \equiv \psi$ ) if

$$\mathcal{M}, s \models \varphi \iff \mathcal{M}, s \models \psi$$

for all models  $\mathcal{M} = (S, \rightarrow, L)$  and states  $s \in S$

## Theorem

$$\neg \text{AF } \varphi \equiv \text{EG } \neg \varphi$$

$$\text{AF } \varphi \equiv \text{A}[\text{T } \text{U } \varphi]$$

$$\neg \text{EF } \varphi \equiv \text{AG } \neg \varphi$$

$$\text{EF } \varphi \equiv \text{E}[\text{T } \text{U } \varphi]$$

$$\neg \text{AX } \varphi \equiv \text{EX } \neg \varphi$$

$$\text{A}[\varphi \text{U } \psi] \equiv \neg(\text{E}[\neg \psi \text{U } (\neg \varphi \wedge \neg \psi)]) \vee \text{EG } \neg \psi$$

## Outline

1. Summary of Previous Lecture
2. Post's Adequacy Theorem
3. Intermezzo
4. Model Checking
5. Branching-Time Temporal Logic (CTL)
6. CTL Model Checking Algorithm
7. Further Reading

### CTL Model Checking Algorithm 1

input: • model  $\mathcal{M} = (S, \rightarrow, L)$  and CTL formula  $\varphi$   
 output: •  $\{s \in S \mid \mathcal{M}, s \models \varphi\}$

label each state  $s \in S$  by those subformulas of  $\varphi$  that are satisfied in  $s$

- $\top$  label every state
- $\perp$  label no state
- $p$  label  $s \iff p \in L(s)$
- $\neg\varphi$  label  $s \iff s$  is not labelled with  $\varphi$
- $\varphi \wedge \psi$  label  $s \iff s$  is labelled with both  $\varphi$  and  $\psi$
- $\varphi \vee \psi$  label  $s \iff s$  is labelled with  $\varphi$  or  $\psi$
- $\varphi \rightarrow \psi$  label  $s \iff s$  is not labelled with  $\varphi$  or  $s$  is labelled with  $\psi$
- $AX\varphi$  label  $s \iff t$  is labelled with  $\varphi$  for all  $t$  with  $s \rightarrow t$

### CTL Model Checking Algorithm 2

- $EX\varphi$  label  $s \iff t$  is labelled with  $\varphi$  for some  $t$  with  $s \rightarrow t$
- $AF\varphi$  label  $s \iff$ 
  - ①  $s$  is labelled with  $\varphi$
  - ②  $t$  is labelled with  $AF\varphi$  for all  $t$  with  $s \rightarrow t$
  - ③ repeat ② until no change
- $EF\varphi$  label  $s \iff$ 
  - ①  $s$  is labelled with  $\varphi$
  - ②  $t$  is labelled with  $EF\varphi$  for some  $t$  with  $s \rightarrow t$
  - ③ repeat ② until no change
- $AG\varphi$ 
  - ① label every  $s$  that is labelled with  $\varphi$
  - ② remove label from  $s \iff t$  is not labelled with  $AG\varphi$  for some  $t$  with  $s \rightarrow t$
  - ③ repeat ② until no change

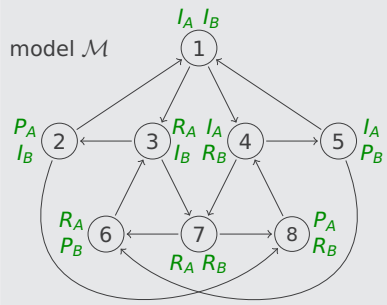
### CTL Model Checking Algorithm 3

- $EG\varphi$ 
  - ① label every  $s$  that is labelled with  $\varphi$
  - ② remove label from  $s \iff t$  is not labelled with  $EG\varphi$  for all  $t$  with  $s \rightarrow t$
  - ③ repeat ② until no change
- $A[\varphi U \psi]$  label  $s \iff$ 
  - ①  $s$  is labelled with  $\psi$
  - ②  $s$  is labelled with  $\varphi$  and  $t$  with  $A[\varphi U \psi]$  for all  $t$  with  $s \rightarrow t$
  - ③ repeat ② until no change
- $E[\varphi U \psi]$  label  $s \iff$ 
  - ①  $s$  is labelled with  $\psi$
  - ②  $s$  is labelled with  $\varphi$  and  $t$  with  $E[\varphi U \psi]$  for some  $t$  with  $s \rightarrow t$
  - ③ repeat ② until no change

### Example 1

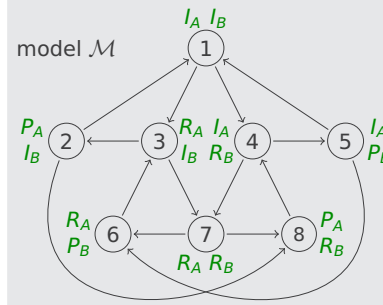
model $\mathcal{M}$	$R_A$	$P_A$	$AF P_A$	$R_A \rightarrow AF P_A$	$AG(R_A \rightarrow AF P_A)$
1				✓	(1 → 3)
2		✓	✓	✓	(2 → 1)
3	✓				
4				✓	(4 → 7)
5				✓	(5 → 6)
6	✓				
7	✓				
8		✓	✓	✓	(8 → 4)

### Example 2



	$R_A$	$P_A$	$EF P_A$	$R_A \rightarrow EF P_A$	$AG(R_A \rightarrow EF P_A)$
1			✓	✓	✓
2		✓	✓	✓	✓
3	✓		✓	✓	✓
4			✓	✓	✓
5			✓	✓	✓
6	✓		✓	✓	✓
7	✓		✓	✓	✓
8		✓	✓	✓	✓

### Example 3



	$R_B$	$\neg R_B$	$P_B$	$E[\neg R_B \cup P_B]$	$\neg E[\neg R_B \cup P_B]$
1		✓			✓
2		✓			✓
3		✓			✓
4	✓				✓
5			✓	✓	
6			✓	✓	✓
7	✓				✓
8	✓				✓

### More Efficient Algorithm for EG

EG  $\varphi$  ① restrict graph to states satisfying  $\varphi$ :

$$S' = \{s \in S \mid \mathcal{M}, s \models \varphi\}$$

$$\rightarrow' = \{(s, t) \mid s \rightarrow t \text{ and } s, t \in S'\}$$

- ② compute **non-trivial strongly connected components** of  $(S', \rightarrow')$
- ③ label all states in such **SCCs**
- ④ compute and label all states that in  $(S', \rightarrow')$  can reach labelled state

### Complexity

$f$ : # connectives

$\mathcal{O}(f \cdot (V + E))$  with  $V$ : # states instead of  $\mathcal{O}(f \cdot V \cdot (V + E))$

$E$ : # transitions

### State Explosion Problem

size of model is more often than not exponential in number of variables and number of components which execute in parallel

- ▶ OBDDs to represent sets of states
- ▶ abstraction
- ▶ partial order reduction
- ▶ induction
- ▶ composition

lecture 11

### Demo

CMCV

by Matthias Perktold (2014)

# Outline

1. Summary of Previous Lecture
2. Post's Adequacy Theorem
3. Intermezzo
4. Model Checking
5. Branching-Time Temporal Logic (CTL)
6. CTL Model Checking Algorithm
7. Further Reading

## Huth and Ryan

- ▶ Section 3.4.1
- ▶ Section 3.4.2
- ▶ Section 3.6.1

## Post Adequacy Theorem

- ▶ Post's Functional Completeness Theorem  
Francis Jeffrey Pelletier and Norman M. Martin  
Notre Dame Journal of Formal Logic 31(2), pp. 462–475, 1990  
doi: [10.1305/ndjfl/1093635508](https://doi.org/10.1305/ndjfl/1093635508)
- ▶ Boolean Function and Computation Models  
Peter Clote and Evangelos Kranakis  
Texts in Theoretical Computer Science, Springer, 2012  
doi: [10.1007/978-3-662-04943-3](https://doi.org/10.1007/978-3-662-04943-3)

## Important Concepts

- |            |                          |         |                           |
|------------|--------------------------|---------|---------------------------|
| ▶ AF       | ▶ AX                     | ▶ EG    | ▶ monotonicity            |
| ▶ affinity | ▶ computation tree logic | ▶ EU    | ▶ Post's adequacy theorem |
| ▶ AG       | ▶ CTL                    | ▶ EX    | ▶ self-duality            |
| ▶ AU       | ▶ EF                     | ▶ model | ▶ temporal connective     |

homework for May 21