Introduction to Software Verification

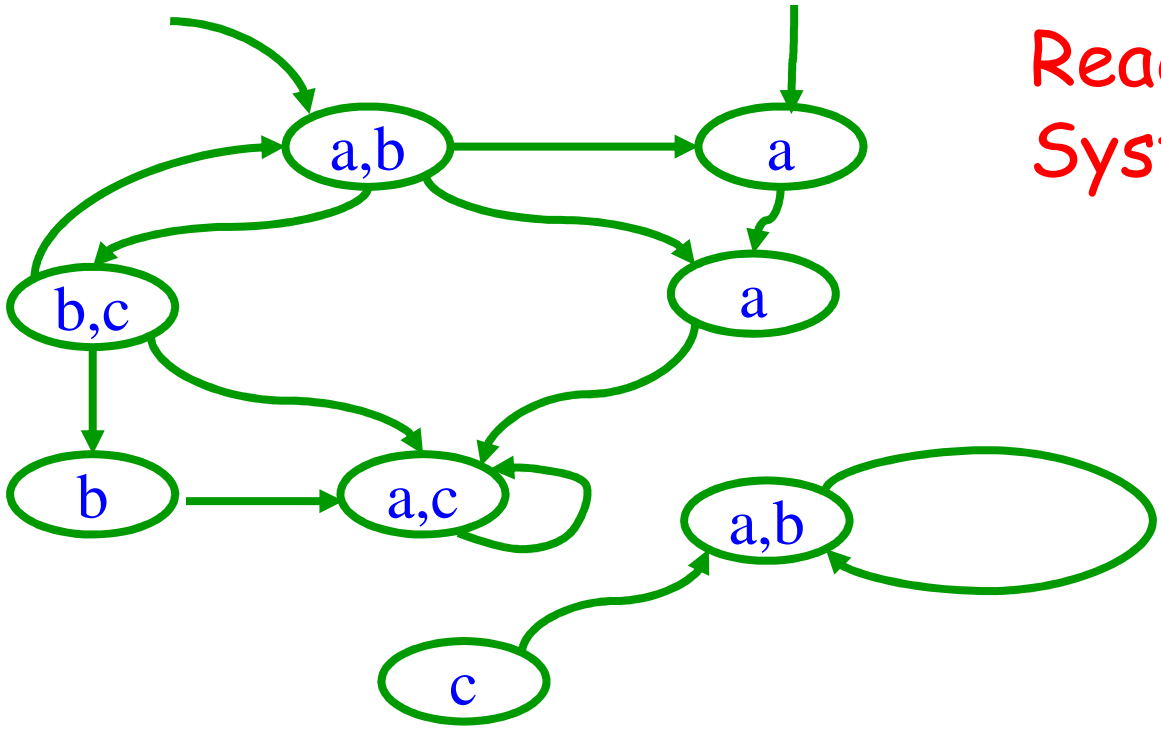
Orna Grumberg

Lectures Material
winter 2017-18

Lecture 5

Model of a system

Kripke structure / transition system



Reactive Systems

Labeled by **atomic propositions AP**
(critical section, variable value…)

Kripke Structure $M=(S,R,L,S_0)$

Given AP – finite set of atomic proposition

- S – (finite) set of states
- $R \subseteq S \times S$ – total transition relation

  For every $s \in S$ there exists $s' \in S$ such that $(s,s') \in R$.

  Totality means that every path is infinite
- $L: S \rightarrow 2^{AP}$ - labeling function that associates every state with the atomic propositions true in that state
- $S_0 \subseteq S$ – set of initial states (optional)

CTL*

State formulas:

- $p \in AP$
- $\neg g_1$, $g_1 \vee g_2$, $g_1 \wedge g_2$ where g_1, g_2 are state formulas
- Ef, Af where f is a path formula

Path formulas:

- Every state formula g is a path formula
- $\neg f_1$, $f_1 \vee f_2$, $f_1 \wedge f_2$, Xf_1, Gf_1, Ff_1, $f_1 U f_2$ where f_1, f_2 are path formulas

CTL* - set of all state formulas

Semantics of CTL*

$\pi = s_0,s_1,\ldots$ is a path in M if $R(s_i,s_{i+1})$ for every i.

π^i – the suffix of π starting at s_i.

State formulas:

- $M,s \models p \Leftrightarrow p \in L(s)$
- $M,s \models Ef \Leftrightarrow$ there is a path π from s s.t. $M, \pi \models f$
- $M,s \models Af \Leftrightarrow$ for every path π from s, $M, \pi \models f$

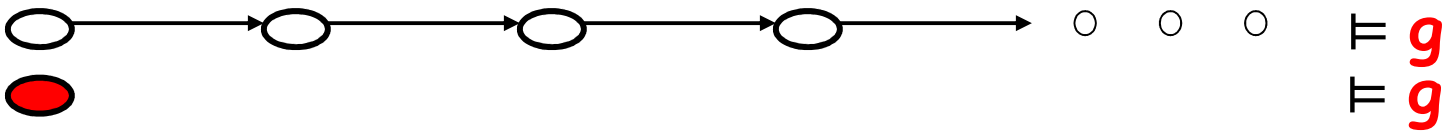
Semantics of CTL*

$\pi = s_0, s_1, \ldots$ is a path in M if $R(s_i, s_{i+1})$ for every i.

π^i – the suffix of π starting at s_i.

Path formulas:

- M, $\pi \models g$, where g is a state formula $\Leftrightarrow$ M, $s_0 \models g$



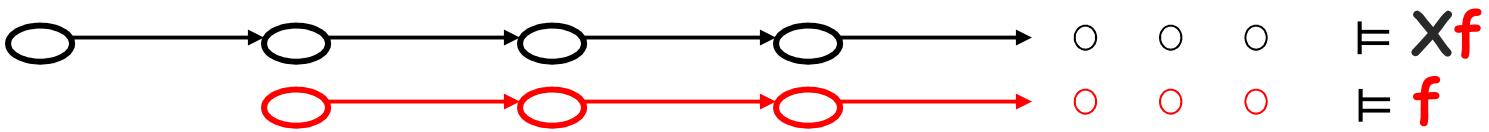
Semantics of CTL*

$\pi = s_0, s_1, \ldots$ is a path in M if $R(s_i, s_{i+1})$ for every i.

π^i – the suffix of π starting at s_i.

Path formulas:

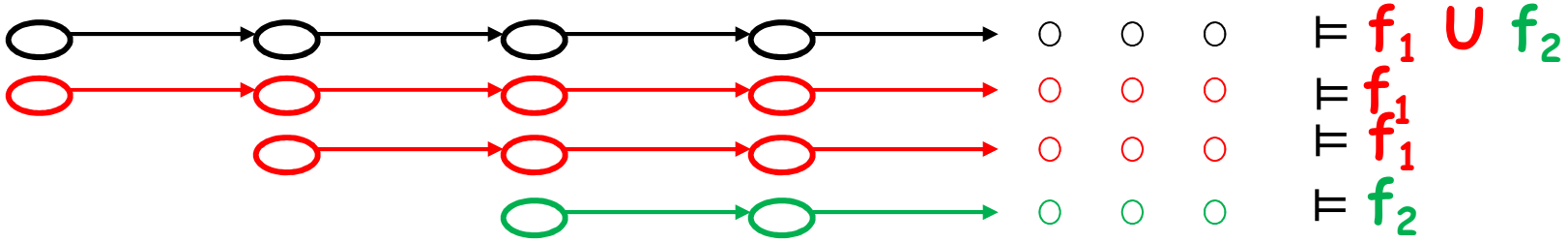
- $M, \pi \models g$, where g is a state formula $\Leftrightarrow M, s_0 \models g$
- $M, \pi \models Xf \Leftrightarrow M, \pi^1 \models f$



Semantics of CTL*

$\pi = s_0, s_1, \ldots$ is a path in M if $R(s_i, s_{i+1})$ for every i.

π^i – the suffix of π starting at s_i.



- $M, \pi \models Gf \Leftrightarrow$ for every $k \geq 0$, $M, \pi^k \models f$
- $M, \pi \models Ff \Leftrightarrow$ there exists $k \geq 0$, s.t. $M, \pi^k \models f$
- $M, \pi \models f_1 \, U \, f_2 \Leftrightarrow$ there exists $k \geq 0$, s.t. $M, \pi^k \models f_2$ and for every $0 \leq j < k$, $M, \pi^j \models f_1$

Semantics of CTL*

$\pi = s_0, s_1, \ldots$ is a path in M if $R(s_i, s_{i+1})$ for every i.

π^i – the suffix of π starting at s_i.

Path formulas:

- $M, \pi \models g$, where g is a state formula $\Leftrightarrow M, s_0 \models g$
- $M, \pi \models Xf \Leftrightarrow M, \pi^1 \models f$
- $M, \pi \models Gf \Leftrightarrow$ for every $k \geq 0$, $M, \pi^k \models f$
- $M, \pi \models Ff \Leftrightarrow$ there exists $k \geq 0$, s.t. $M, \pi^k \models f$
- $M, \pi \models f_1 \, U \, f_2 \Leftrightarrow$ there exists $k \geq 0$, s.t. $M, \pi^k \models f_2$ and for every $0 \leq j < k$, $M, \pi^j \models f_1$

Semantics of CTL*

$M \models g \Leftrightarrow$ for every initial state s: $M,s \models g$

LTL/CTL/CTL*

LTL – state formulas of the form $A\psi$

ψ - path formula, contains **no** path **quantifiers**

- interpreted over infinite computation paths

CTL – state formulas where path quantifiers and temporal operators appear in pairs:

AG, AU, AF, AX, EG, EU, EF, EX

- interpreted over infinite computation trees

CTL* - Allows any combination of temporal operators and path quantifiers. Includes both LTL and CTL

LTL

State formulas:

- Af where f is a path formula

Path formulas:

- $p \in AP$
- $\neg f_1$, $f_1 \vee f_2$, $f_1 \wedge f_2$, Xf_1, Gf_1, Ff_1, $f_1 U f_2$ where f_1, f_2 are path formulas

LTL - set of all state formulas

CTL

CTL - set of all state formulas

- $p \in AP$
- $\neg g_1$, $g_1 \vee g_2$, $g_1 \wedge g_2$
- **AX** g_1, **AG** g_1, **AF** g_1, **A** g_1 **U** g_2,
- **EX** g_1, **EG** g_1, **EF** g_1, **E** g_1 **U** g_2,

  where g_1, g_2 are state formulas

Semantics of CTL

Recall: path $\pi = s_0, s_1, \ldots$

- $M,s \models p \iff p \in L(s)$ for $p \in AP$

- $M,s \models \varphi_1 \vee \varphi_2 \iff M,s \models \varphi_1$ or $M,s \models \varphi_2$
- $M,s \models EX\varphi \iff$ there is s' s.t. $R(s,s')$ and $M,s' \models \varphi$
- $M,s \models EG\varphi \iff$ there is a path π from s, s.t. for every $i \geq 0$, $M,s_i \models \varphi$

Semantics of CTL

- $M,s \models E[\varphi_1 U \varphi_2] \Leftrightarrow$ there is a path π from s
  and there is $k \geq 0$ s.t. $M,s_k \models \varphi_2$
  and for every $k > i \geq 0$, $M,s_i \models \varphi_1$

- $M,s \models AG\varphi \Leftrightarrow$ for every path π from s
  and for every $i \geq 0$, $M,s_i \models \varphi$

- $M,s \models AF\varphi \Leftrightarrow$ for every path π from s
  there exists $i \geq 0$ s.t. $M,s_i \models \varphi$

Examples (LTL)

1. $AG\ \neg(start \wedge \neg ready)$
2. $AG\ (req \rightarrow F\ ack)$
3. $A\ GF\ enbled$
4. $A\ FG\ deadlock$
5. $A\ (GF\ enbled \rightarrow GF\ running)$

Cannot express existential properties: "from any state the system can…"

Examples (CTL)

1. EF (start$\wedge\neg$ready)
2. AG (req $\rightarrow$ AF ack)
3. AG (AF enbled)
4. AF (AG deadlock)
5. AG (EF restart)
6. AG (non_critical $\rightarrow$ EX tryring)
7. AG (try $\rightarrow$ A[try U succeed])

Equivalence

- Path formulas ψ_1, ψ_2 are equivalent if:
  For every M and path π
  $$M, \pi \models \psi_1 \text{ iff } M, \pi \models \psi_2$$

- State formulas φ_1, φ_2 are equivalent if:
  For every M and state s
  $$M, s \models \varphi_1 \text{ iff } M, s \models \varphi_2$$

Expressiveness

$\neg$, $\vee$, X, U, E suffice to express all CTL*:

- $Ff \equiv$ true U f
- $Gf \equiv \neg F(\neg f)$
- $Af \equiv \neg E(\neg f)$

In CTL: EX, EG, EU are sufficient

- $A[pUq] \equiv (\neg EG \neg q) \wedge \neg E[\neg q \; U \; (\neg p \wedge \neg q)]$

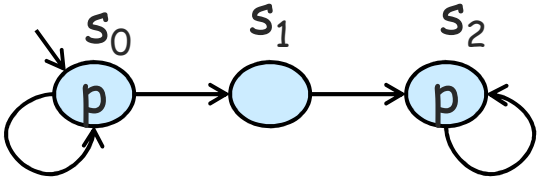
LTL vs. CTL

- A (FG p) has no equivalent in CTL

  "in all paths, p globally holds from some point on"

- Failed attempts:

  AFAGP : "in every path there is a point from which all reachable states satisfy p."



All paths satisfy FGp

- $s_0,s_0,s_0,\ldots$
- $s_0,s_0,\ldots s_0,s_1,s_2,s_2,s_2\ldots$

But first one does not sat FAGp

LTL and CTL vs. CTL*

- E (GFp) has no equivalent in LTL or CTL

Theorem:

- The expressive powers of LTL and CTL are incomparable. That is,
  - There is an LTL formula that has no equivalent CTL formula
  - There is a CTL formula that has no equivalent LTL formula
- CTL* is more expressive than either of them

Explicit Model Checking for CTL

Model Checking [CE81,QS82]

An efficient procedure that receives:

- A finite-state model describing a system
- A temporal logic formula describing a property

It returns

yes, if the system has the property

no + Counterexample, otherwise

CTL Model Checking M |= f

- For each s, computes **label(s)**:
  set of subformulas of f which are true in s

- The Model Checking algorithm works **iteratively** on subformulas of **f**, from **simpler** subformulas to more **complex** ones

- For checking **AG(request ⇒ AF grant)**
  - Check **grant**, **request**
  - Then check **AF grant**
  - Next check **request ⇒ AF grant**
  - Finally check **AG(request ⇒ AF grant)**

Model Checking M |= f (cont.)

- We check subformula **g** of **f** only after all subformulas of **g** have already been checked
- For subformula **g**, the algorithm adds g to label(s) for every state that satisfies g
  - $g \in \text{label}(s) \Leftrightarrow M,s \models g$
- The algorithm has time complexity: **O(|M| × |f|)**

Model Checking M |= f (cont.)

- $M \models f$ if and only if $f \in labels(s)$ for **all initial** states s of M
- Denote $S_f = \{ s \mid M, s \models f \}$
- $M \models f$ if and only if $S_0 \subseteq S_f$

Model Checking Atomic Propositions

- For atomic propsition **p∈AP**:

  $p \in label(s) \Leftrightarrow p \in L(s)$

  Held by alg — Defined by M

How do we handle **more complex** formulas?

Observation:

- Sufficient to handle $\neg$, $\vee$, EX, EU, EG

Model Checking $\neg$, $\vee$ formulas

$\neg f_1$: add to label(s) if and only if $f_1 \notin$ labels(s)

$f_1 \vee f_2$: add to label(s) if and only if
$f_1 \in$ labels(s) or $f_2 \in$ labels(s)

Model Checking $g = EX\ f_1$

add g to label(s) if and only if s has a successor t such that $f_1 \in$ labels(t)

procedure CheckEX (f_1)

T := { t | $f_1 \in$ label(t) }

while $T \neq \emptyset$ do

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

for all s s.t. R(s,t) do

if EX $f_1 \notin$ label(s) then

label(s) : = label(s) $\cup$ { EX f_1};

end for all

end while

Model Checking $g = E(f_1\ U\ f_2)$

procedure CheckEU (f_1, f_2)

T := { s | $f_2 \in$ label(s) }

For all s$\in$T do label(s) := label(s)$\cup$ { $E(f_1\ U\ f_2)$ }

while T $\neq \emptyset$ do

choose s $\in$T ; T := T \ {s} ;

for all t s.t. R(t,s) do

if $E(f_1\ U\ f_2) \notin$ label(t) and $f_1 \in$ label(t) then

label(t) : = label(t) $\cup$ { $E(f_1\ U\ f_2)$ };

T : = T $\cup$ {t}

end for all

end while

Do not add a state to T more than once

Example $g = E(f_1 \ U \ f_2)$