Computation Tree Logic (CTL)

- LTL formulae ϕ are evaluated on paths .... path formulae
- CTL formulae ψ are evaluated on states .. state formulae

- Syntax of CTL well-formed formulae:

$$
\begin{array}{rcll}
\psi & ::= & p & \text{(Atomic formula } p \in AP\text{)} \\
 & | & \neg\psi & \text{(Negation)} \\
 & | & \psi_1 \wedge \psi_2 & \text{(Conjunction)} \\
 & | & \psi_1 \vee \psi_2 & \text{(Disjunction)} \\
 & | & \psi_1 \Rightarrow \psi_2 & \text{(Implication)} \\
 & | & \mathbf{AX}\psi & \text{(All successors)} \\
 & | & \mathbf{EX}\psi & \text{(Some successors)} \\
 & | & \mathbf{A}[\psi_1\ \mathbf{U}\ \psi_2] & \text{(Until – along all paths)} \\
 & | & \mathbf{E}[\psi_1\ \mathbf{U}\ \psi_2] & \text{(Until – along some path)}
\end{array}
$$

Semantics of CTL

- Assume $M = (S, S_0, R, L)$ and then define:

$$
\begin{array}{lcl}
[\![p]\!]_M(s) & = & p \in L(s) \\
[\![\neg\psi]\!]_M(s) & = & \neg([\![\psi]\!]_M(s)) \\
[\![\psi_1 \wedge \psi_2]\!]_M(s) & = & [\![\psi_1]\!]_M(s) \ \wedge \ [\![\psi_2]\!]_M(s) \\
[\![\psi_1 \vee \psi_2]\!]_M(s) & = & [\![\psi_1]\!]_M(s) \ \vee \ [\![\psi_2]\!]_M(s) \\
[\![\psi_1 \Rightarrow \psi_2]\!]_M(s) & = & [\![\psi_1]\!]_M(s) \ \Rightarrow \ [\![\psi_2]\!]_M(s) \\
[\![\mathbf{AX}\psi]\!]_M(s) & = & \forall s'.\ R\ s\ s' \ \Rightarrow \ [\![\psi]\!]_M(s') \\
[\![\mathbf{EX}\psi]\!]_M(s) & = & \exists s'.\ R\ s\ s' \ \wedge \ [\![\psi]\!]_M(s') \\
[\![\mathbf{A}[\psi_1\ \mathbf{U}\ \psi_2]]\!]_M(s) & = & \forall \pi.\ \mathrm{Path}\ R\ s\ \pi \\
 & & \quad \Rightarrow \exists i.\ [\![\psi_2]\!]_M(\pi(i)) \\
 & & \qquad\quad \wedge \\
 & & \qquad\quad \forall j.\ j{<}i \ \Rightarrow \ [\![\psi_1]\!]_M(\pi(j)) \\
[\![\mathbf{E}[\psi_1\ \mathbf{U}\ \psi_2]]\!]_M(s) & = & \exists \pi.\ \mathrm{Path}\ R\ s\ \pi \\
 & & \quad \wedge \ \exists i.\ [\![\psi_2]\!]_M(\pi(i)) \\
 & & \qquad\quad \wedge \\
 & & \qquad\quad \forall j.\ j{<}i \ \Rightarrow \ [\![\psi_1]\!]_M(\pi(j))
\end{array}
$$

The defined operator **AF**

- Define $\mathbf{AF}\psi = \mathbf{A}[\mathrm{T}\ \mathbf{U}\ \psi]$
- $\mathbf{AF}\psi$ true at s iff ψ true somewhere on every R-path from s

$$
\begin{aligned}
[\![\mathbf{AF}\psi]\!]_M(s) &= [\![\mathbf{A}[\mathrm{T}\ \mathbf{U}\ \psi]]\!]_M(s) \\
&= \forall \pi.\ \mathrm{Path}\ R\ s\ \pi \\
&\qquad \Rightarrow \\
&\qquad \exists i.\ [\![\psi]\!]_M(\pi(i)) \ \wedge\ \forall j.\ j < i \ \Rightarrow\ [\![\mathrm{T}]\!]_M(\pi(j)) \\
&= \forall \pi.\ \mathrm{Path}\ R\ s\ \pi \\
&\qquad \Rightarrow \\
&\qquad \exists i.\ [\![\psi]\!]_M(\pi(i)) \ \wedge\ \forall j.\ j < i \ \Rightarrow\ \mathit{true} \\
&= \forall \pi.\ \mathrm{Path}\ R\ s\ \pi \ \Rightarrow\ \exists i.\ [\![\psi]\!]_M(\pi(i))
\end{aligned}
$$

The defined operator **EF**

- Define $\mathbf{EF}\psi = \mathbf{E}[\mathrm{T}\ \mathbf{U}\ \psi]$

- $\mathbf{EF}\psi$ true at s iff ψ true somewhere on some R-path from s

$$
\begin{aligned}
[\![\mathbf{EF}\psi]\!]_M(s) &= [\![\mathbf{E}[\mathrm{T}\ \mathbf{U}\ \psi]]\!]_M(s) \\
&= \exists \pi.\ \mathrm{Path}\ R\ s\ \pi \\
&\qquad \wedge \\
&\qquad \exists i.\ [\![\psi]\!]_M(\pi(i)) \ \wedge\ \forall j.\ j < i \ \Rightarrow\ [\![\mathrm{T}]\!]_M(\pi(j)) \\
&= \exists \pi.\ \mathrm{Path}\ R\ s\ \pi \\
&\qquad \wedge \\
&\qquad \exists i.\ [\![\psi]\!]_M(\pi(i)) \ \wedge\ \forall j.\ j < i \ \Rightarrow\ \mathit{true} \\
&= \exists \pi.\ \mathrm{Path}\ R\ s\ \pi \ \wedge\ \exists i.\ [\![\psi]\!]_M(\pi(i))
\end{aligned}
$$

- “can reach a state satisfying $p \in AP$” is $\mathbf{EF}\ p$

The defined operator **AG**

- Define $\mathbf{AG}\psi = \neg\mathbf{EF}(\neg\psi)$

- $\mathbf{AG}\psi$ true at s iff ψ true everywhere on every R-path from s

$$
\begin{aligned}
[\![\mathbf{AG}\psi]\!]_M(s) &= [\![\neg\mathbf{EF}(\neg\psi)]\!]_M(s) \\
&= \neg([\![\mathbf{EF}(\neg\psi)]\!]_M(s)) \\
&= \neg(\exists\pi.\ \mathsf{Path}\ R\ s\ \pi \wedge \exists i.\ [\![\neg\psi]\!]_M(\pi(i))) \\
&= \neg(\exists\pi.\ \mathsf{Path}\ R\ s\ \pi \wedge \exists i.\ \neg[\![\psi]\!]_M(\pi(i))) \\
&= \forall\pi.\ \neg(\mathsf{Path}\ R\ s\ \pi \wedge \exists i.\ \neg[\![\psi]\!]_M(\pi(i))) \\
&= \forall\pi.\ \neg\mathsf{Path}\ R\ s\ \pi \vee \neg(\exists i.\ \neg[\![\psi]\!]_M(\pi(i))) \\
&= \forall\pi.\ \neg\mathsf{Path}\ R\ s\ \pi \vee \forall i.\ \neg\neg[\![\psi]\!]_M(\pi(i)) \\
&= \forall\pi.\ \neg\mathsf{Path}\ R\ s\ \pi \vee \forall i.\ [\![\psi]\!]_M(\pi(i)) \\
&= \forall\pi.\ \mathsf{Path}\ R\ s\ \pi \Rightarrow \forall i.\ [\![\psi]\!]_M(\pi(i))
\end{aligned}
$$

- $\mathbf{AG}\psi$ means ψ true at all reachable states

- $[\![\mathbf{AG}(p)]\!]_M(s) \ \equiv\ \forall s'.\ R^*\ s\ s' \ \Rightarrow\ p \in L(s')$

- "can always reach a state satisfying $p \in AP$" is $\mathbf{AG}(\mathbf{EF}\ p)$

The defined operator EG

- Define $\mathbf{EG}\psi = \neg\mathbf{AF}(\neg\psi)$
- $\mathbf{EG}\psi$ true at s iff ψ true everywhere on some R-path from s

$$
\begin{aligned}
[\![\mathbf{EG}\psi]\!]_M(s) &= [\![\neg\mathbf{AF}(\neg\psi)]\!]_M(s)\\
&= \neg([\![\mathbf{AF}(\neg\psi)]\!]_M(s))\\
&= \neg(\forall\pi.\ \mathsf{Path}\ R\ s\ \pi \Rightarrow \exists i.\ [\![\neg\psi]\!]_M(\pi(i)))\\
&= \neg(\forall\pi.\ \mathsf{Path}\ R\ s\ \pi \Rightarrow \exists i.\ \neg[\![\psi]\!]_M(\pi(i)))\\
&= \exists\pi.\ \neg(\mathsf{Path}\ R\ s\ \pi \Rightarrow \exists i.\ \neg[\![\psi]\!]_M(\pi(i)))\\
&= \exists\pi.\ \mathsf{Path}\ R\ s\ \pi \wedge \neg(\exists i.\ \neg[\![\psi]\!]_M(\pi(i)))\\
&= \exists\pi.\ \mathsf{Path}\ R\ s\ \pi \wedge \forall i.\ \neg\neg[\![\psi]\!]_M(\pi(i))\\
&= \exists\pi.\ \mathsf{Path}\ R\ s\ \pi \wedge \forall i.\ [\![\psi]\!]_M(\pi(i))
\end{aligned}
$$

The defined operator $\mathbf{A}[\psi_1 \ \mathbf{W} \ \psi_2]$

- $\mathbf{A}[\psi_1 \ \mathbf{W} \ \psi_2]$ is a 'partial correctness' version of $\mathbf{A}[\psi_1 \ \mathbf{U} \ \psi_2]$
- It is true at s if along all R-paths from s:
  - ψ_1 always holds on the path, or
  - ψ_2 holds sometime on the path, and until it does ψ_1 holds
- Define

$$
\begin{aligned}
&[\![\mathbf{A}[\psi_1 \ \mathbf{W} \ \psi_2]]\!]_M(s) \\
&\quad = [\![\neg\mathbf{E}[(\psi_1 \wedge \neg\psi_2) \ \mathbf{U} \ (\neg\psi_1 \wedge \neg\psi_2)]]\!]_M(s) \\
&\quad = \neg[\![\mathbf{E}[(\psi_1 \wedge \neg\psi_2) \ \mathbf{U} \ (\neg\psi_1 \wedge \neg\psi_2)]]\!]_M(s) \\
&\quad = \neg(\exists \pi.\ \mathrm{Path}\ R\ s\ \pi \\
&\qquad\qquad \wedge \\
&\qquad\qquad \exists i.\ [\![\neg\psi_1 \wedge \neg\psi_2]\!]_M(\pi(i)) \\
&\qquad\qquad\quad \wedge \\
&\qquad\qquad\quad \forall j.\ j<i \ \Rightarrow\ [\![\psi_1 \wedge \neg\psi_2]\!]_M(\pi(j)))
\end{aligned}
$$

- Exercise: understand the next two slides!

$\mathbf{A}[\psi_1\ \mathbf{W}\ \psi_2]$ continued (1)

- Continuing:

$$
\begin{array}{ll}
& \neg(\exists\pi.\ \mathsf{Path}\ R\ s\ \pi \\
& \qquad \wedge \\
& \qquad \exists i.\ [\![\neg\psi_1\wedge\neg\psi_2]\!]_M(\pi(i))\ \wedge\ \forall j.\ j{<}i\ \Rightarrow\ [\![\psi_1\wedge\neg\psi_2]\!]_M(\pi(j))) \\
= & \forall\pi.\ \neg(\mathsf{Path}\ R\ s\ \pi \\
& \qquad \wedge \\
& \qquad \exists i.\ [\![\neg\psi_1\wedge\neg\psi_2]\!]_M(\pi(i))\ \wedge\ \forall j.\ j{<}i\ \Rightarrow\ [\![\psi_1\wedge\neg\psi_2]\!]_M(\pi(j))) \\
= & \forall\pi.\ \mathsf{Path}\ R\ s\ \pi \\
& \qquad \Rightarrow \\
& \qquad \neg(\exists i.\ [\![\neg\psi_1\wedge\neg\psi_2]\!]_M(\pi(i))\ \wedge\ \forall j.\ j{<}i\ \Rightarrow\ [\![\psi_1\wedge\neg\psi_2]\!]_M(\pi(j))) \\
= & \forall\pi.\ \mathsf{Path}\ R\ s\ \pi \\
& \qquad \Rightarrow \\
& \qquad \forall i.\ \neg[\![\neg\psi_1\wedge\neg\psi_2]\!]_M(\pi(i)) \vee \neg(\forall j.\ j{<}i\ \Rightarrow\ [\![\psi_1\wedge\neg\psi_2]\!]_M(\pi(j)))
\end{array}
$$

$\mathbf{A}[\psi_1\ \mathbf{W}\ \psi_2]$ continued (2)

- Continuing:

$$= \forall \pi.\ \mathrm{Path}\ R\ s\ \pi \Rightarrow \forall i.\ \neg [\![\neg\psi_1 \wedge \neg\psi_2]\!]_M(\pi(i)) \vee \neg(\forall j.\ j<i \Rightarrow [\![\psi_1 \wedge \neg\psi_2]\!]_M(\pi(j)))$$

$$= \forall \pi.\ \mathrm{Path}\ R\ s\ \pi \Rightarrow \forall i.\ \neg(\forall j.\ j<i \Rightarrow [\![\psi_1 \wedge \neg\psi_2]\!]_M(\pi(j))) \vee \neg [\![\neg\psi_1 \wedge \neg\psi_2]\!]_M(\pi(i))$$

$$= \forall \pi.\ \mathrm{Path}\ R\ s\ \pi \Rightarrow \forall i.\ (\forall j.\ j<i \Rightarrow [\![\psi_1 \wedge \neg\psi_2]\!]_M(\pi(j))) \Rightarrow [\![\psi_1 \vee \psi_2]\!]_M(\pi(i))$$

- Exercise: explain why this is $[\![\mathbf{A}[\psi_1\ \mathbf{W}\ \psi_2]]\!]_M(s)$?
  - this exercise illustrates the subtlety of writing CTL!

Sanity check: $\mathbf{A}[\psi\ \mathbf{W}\ \mathsf{F}] \ = \ \mathbf{AG}\ \psi$

- From last slide:
  $[\![\mathbf{A}[\psi_1\ \mathbf{W}\ \psi_2]]\!]_M(s)$
  $= \ \forall\pi.\ \mathsf{Path}\ R\ s\ \pi$
  $\Rightarrow\ \forall i.\ (\forall j.\ j{<}i \Rightarrow [\![\psi_1{\wedge}\neg\psi_2]\!]_M(\pi(j))) \Rightarrow [\![\psi_1{\vee}\psi_2]\!]_M(\pi(i))$
- Set ψ_1 to ψ and ψ_2 to F:
  $[\![\mathbf{A}[\psi\ \mathbf{W}\ \mathsf{F}]]\!]_M(s)$
  $= \ \forall\pi.\ \mathsf{Path}\ R\ s\ \pi$
  $\Rightarrow\ \forall i.\ (\forall j.\ j{<}i\ \Rightarrow\ [\![\psi{\wedge}\neg\mathsf{F}]\!]_M(\pi(j)))\ \Rightarrow\ [\![\psi{\vee}\mathsf{F}]\!]_M(\pi(i))$
- Simplify:
  $[\![\mathbf{A}[\psi\ \mathbf{W}\ \mathsf{F}]]\!]_M(s)$
  $= \ \forall\pi.\ \mathsf{Path}\ R\ s\ \pi \Rightarrow \forall i.\ (\forall j.\ j{<}i \Rightarrow [\![\psi]\!]_M(\pi(j))) \Rightarrow [\![\psi]\!]_M(\pi(i))$
- By induction on i:
  $[\![\mathbf{A}[\psi\ \mathbf{W}\ \mathsf{F}]]\!]_M(s) \ = \ \forall\pi.\ \mathsf{Path}\ R\ s\ \pi\ \Rightarrow\ \forall i.\ [\![\psi]\!]_M(\pi(i))$

- Exercises
  1. Describe the property: $\mathbf{A}[\mathsf{T}\ \mathbf{W}\ \psi]$.
  2. Describe the property: $\neg\mathbf{E}[\neg\psi_2\ \mathbf{U}\ \neg(\psi_1{\vee}\psi_2)]$.
  3. Define $\mathbf{E}[\psi_1\ \mathbf{W}\ \psi_2] = \mathbf{E}[\psi_1\ \mathbf{U}\ \psi_2] \vee \mathbf{EG}\psi_1$.
     Describe the property: $\mathbf{E}[\psi_1\ \mathbf{W}\ \psi_2]$?

Summary of CTL operators (primitive + defined)

- CTL formulae:

p	(Atomic formula - $p \in AP$)
$\neg\psi$	(Negation)
$\psi_1 \wedge \psi_2$	(Conjunction)
$\psi_1 \vee \psi_2$	(Disjunction)
$\psi_1 \Rightarrow \psi_2$	(Implication)
$\mathbf{AX}\psi$	(All successors)
$\mathbf{EX}\psi$	(Some successors)
$\mathbf{AF}\psi$	(Somewhere – along all paths)
$\mathbf{EF}\psi$	(Somewhere – along some path)
$\mathbf{AG}\psi$	(Everywhere – along all paths)
$\mathbf{EG}\psi$	(Everywhere – along some path)
$\mathbf{A}[\psi_1\ \mathbf{U}\ \psi_2]$	(Until – along all paths)
$\mathbf{E}[\psi_1\ \mathbf{U}\ \psi_2]$	(Until – along some path)
$\mathbf{A}[\psi_1\ \mathbf{W}\ \psi_2]$	(Unless – along all paths)
$\mathbf{E}[\psi_1\ \mathbf{W}\ \psi_2]$	(Unless – along some path)

Example CTL formulae

- $\mathbf{EF}(\mathit{Started} \land \neg\mathit{Ready})$

  It is possible to get to a state where Started holds but Ready does not hold

- $\mathbf{AG}(\mathit{Req} \Rightarrow \mathbf{AF}\mathit{Ack})$

  If a request Req occurs, then it will eventually be acknowledged by Ack

- $\mathbf{AG}(\mathbf{AF}\mathit{DeviceEnabled})$

  DeviceEnabled is always true somewhere along every path starting anywhere: i.e. DeviceEnabled holds infinitely often along every path

- $\mathbf{AG}(\mathbf{EF}\mathit{Restart})$

  From any state it is possible to get to a state for which Restart holds

More CTL examples (1)

- $\mathbf{AG}(Req \Rightarrow \mathbf{A}[Req\ \mathbf{U}\ Ack])$

  If a request Req occurs, then it continues to hold, until it is eventually acknowledged

- $\mathbf{AG}(Req \Rightarrow \mathbf{AX}(\mathbf{A}[\neg Req\ \mathbf{U}\ Ack]))$

  Whenever Req is true either it must become false on the next cycle and remains false until Ack, or Ack must become true on the next cycle

  Exercise: is the **AX** necessary?

- $\mathbf{AG}(Req \Rightarrow (\neg Ack \Rightarrow \mathbf{AX}(\mathbf{A}[Req\ \mathbf{U}\ Ack])))$

  Whenever Req is true and Ack is false then Ack will eventually become true and until it does Req will remain true

  Exercise: is the **AX** necessary?

More CTL examples (2)

- $\mathbf{AG}(Enabled \Rightarrow \mathbf{AG}(Start \Rightarrow \mathbf{A}[\neg Waiting\ \mathbf{U}\ Ack]))$

  If Enabled is ever true then if Start is true in any subsequent state then Ack will eventually become true, and until it does Waiting will be false

- $\mathbf{AG}(\neg Req_1 \wedge \neg Req_2 \Rightarrow \mathbf{A}[\neg Req_1 \wedge \neg Req_2\ \mathbf{U}\ (Start \wedge \neg Req_2)])$

  Whenever Req_1 and Req_2 are false, they remain false until Start becomes true with Req_2 still false

- $\mathbf{AG}(Req \Rightarrow \mathbf{AX}(Ack \Rightarrow \mathbf{AF}\ \neg Req))$

  If Req is true and Ack becomes true one cycle later, then eventually Req will become false

Some abbreviations

- $\mathbf{AX}_i\ \psi \ \equiv\ \underbrace{\mathbf{AX}(\mathbf{AX}(\cdots(\mathbf{AX}\ \psi)\cdots))}_{i \text{ instances of } \mathbf{AX}}$

  ψ is true on all paths i units of time later

- $\mathbf{ABF}_{i..j}\ \psi \ \equiv\ \mathbf{AX}_i\ \underbrace{(\psi \lor \mathbf{AX}(\psi \lor\ \cdots\ \mathbf{AX}(\psi \lor \mathbf{AX}\ \psi)\cdots))}_{j-i \text{ instances of } \mathbf{AX}}$

  ψ is true on all paths sometime between i units of time later and j units of time later

- $\mathbf{AG}(Req \Rightarrow \mathbf{AX}(Ack_1 \land \mathbf{ABF}_{1..6}(Ack_2 \land \mathbf{A}[Wait\ \mathbf{U}\ Reply])))$

  One cycle after Req, Ack_1 should become true,
  and then Ack_2 becomes true 1 to 6 cycles later
  and then eventually $Reply$ becomes true, but until
  it does $Wait$ holds from the time of Ack_2

- More abbreviations in 'Industry Standard' language PSL

CTL model checking

- For LTL path formulae ϕ recall that $M \models \phi$ is defined by:

$$M \models \phi \;\Leftrightarrow\; \forall \pi\, s.\; s \in S_0 \wedge \mathsf{Path}\; R\; s\; \pi \Rightarrow [\![\phi]\!]_M(\pi)$$

- For CTL state formulae ψ the definition of $M \models \psi$ is:

$$M \models \psi \;\Leftrightarrow\; \forall s.\; s \in S_0 \Rightarrow [\![\psi]\!]_M(s)$$

- M common; LTL, CTL formulae and semantics $[\![\;]\!]_M$ differ

- CTL model checking algorithm:
  - compute $\{s \mid [\![\psi]\!]_M(s) = \mathit{true}\}$ bottom up
  - check $S_0 \subseteq \{s \mid [\![\psi]\!]_M(s) = \mathit{true}\}$
  - symbolic model checking represents these sets as BDDs

CTL model checking: p, $\mathbf{AX}\psi$, $\mathbf{EX}\psi$

- For CTL formula ψ let $\{\!|\psi|\!\}_M = \{s \mid [\![\psi]\!]_M(s) = \mathit{true}\}$
- When unambiguous will write $\{\!|\psi|\!\}$ instead of $\{\!|\psi|\!\}_M$
- $\{\!|p|\!\} = \{s \mid p \in L(s)\}$
  - scan through set of states S marking states labelled with p
  - $\{\!|p|\!\}$ is set of marked states
- To compute $\{\!|\mathbf{AX}\psi|\!\}$
  - recursively compute $\{\!|\psi|\!\}$
  - marks those states all of whose successors are in $\{\!|\psi|\!\}$
  - $\{\!|\mathbf{AX}\psi|\!\}$ is the set of marked states
- To compute $\{\!|\mathbf{EX}\psi|\!\}$
  - recursively compute $\{\!|\psi|\!\}$
  - marks those states with at least one successor in $\{\!|\psi|\!\}$
  - $\{\!|\mathbf{EX}\psi|\!\}$ is the set of marked states

CTL model checking: $[\![\mathbf{E}[\psi_1 \ \mathbf{U} \ \psi_2]]\!]$, $[\![\mathbf{A}[\psi_1 \ \mathbf{U} \ \psi_2]]\!]$

- To compute $[\![\mathbf{E}[\psi_1 \ \mathbf{U} \ \psi_2]]\!]$
  - recursively compute $[\![\psi_1]\!]$ and $[\![\psi_2]\!]$
  - mark all states in $[\![\psi_2]\!]$
  - mark all states in $[\![\psi_1]\!]$ with a successor state that is marked
  - repeat previous line until no change
  - $[\![\mathbf{E}[\psi_1 \ \mathbf{U} \ \psi_2]]\!]$ is set of marked states

- More formally: $[\![\mathbf{E}[\psi_1 \ \mathbf{U} \ \psi_2]]\!] = \bigcup_{n=0}^{\infty} [\![\mathbf{E}[\psi_1 \ \mathbf{U} \ \psi_2]]\!]_n$ where:

$$
\begin{array}{lcl}
[\![\mathbf{E}[\psi_1 \ \mathbf{U} \ \psi_2]]\!]_0 & = & [\![\psi_2]\!] \\
[\![\mathbf{E}[\psi_1 \ \mathbf{U} \ \psi_2]]\!]_{n+1} & = & [\![\mathbf{E}[\psi_1 \ \mathbf{U} \ \psi_2]]\!]_n \\
 & & \cup \\
 & & \{s \in [\![\psi_1]\!] \mid \exists s' \in [\![\mathbf{E}[\psi_1 \ \mathbf{U} \ \psi_2]]\!]_n. \ R\ s\ s'\}
\end{array}
$$

- $[\![\mathbf{A}[\psi_1 \ \mathbf{U} \ \psi_2]]\!]$ similar, but with a more complicated iteration
  - details omitted

Example: checking **EF** p

- $\mathbf{EF}p = \mathbf{E}[\mathsf{T}\ \mathbf{U}\ p]$
  - holds if ψ holds along some path
- Note $[\![\mathsf{T}]\!] = S$
- Let $\mathcal{S}_n = [\![\mathbf{E}[\mathsf{T}\ \mathbf{U}\ p]]\!]_n$ then:

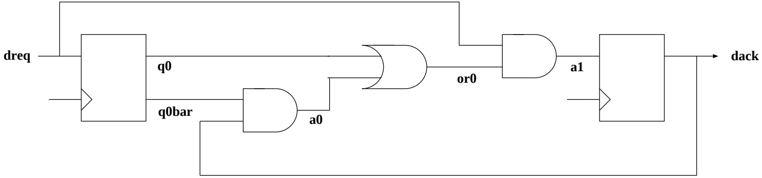
$$
\begin{aligned}
\mathcal{S}_0 &= [\![\mathbf{E}[\mathsf{T}\ \mathbf{U}\ p]]\!]_0 \\
&= [\![p]\!] \\
&= \{s \mid p \in L(s)\}
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{S}_{n+1} &= \mathcal{S}_n \cup \{s \in [\![\mathsf{T}]\!] \mid \exists s' \in [\![\mathbf{E}[\mathsf{T}\ \mathbf{U}\ p]]\!]_n.\ R\ s\ s'\} \\
&= \mathcal{S}_n \cup \{s \mid \exists s' \in \mathcal{S}_n.\ R\ s\ s'\}
\end{aligned}
$$

  - mark all the states labelled with p
  - mark all with at least one marked successor
  - repeat until no change
  - $[\![\mathbf{EF}\ p]\!]$ is set of marked states

Example: RCV

- Recall the handshake circuit:



- State represented by a triple of Booleans $(dreq, q0, dack)$
- A model of RCV is M_{RCV} where:

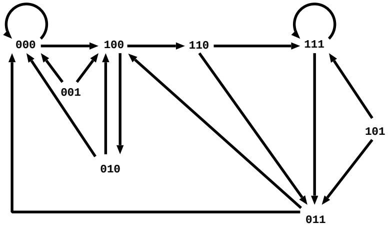
  $M = (S_{\mathrm{RCV}}, S_{0\,\mathrm{RCV}}, R_{\mathrm{RCV}}, L_{\mathrm{RCV}})$

  and

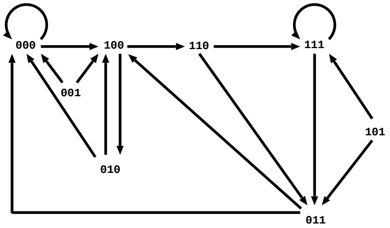
  $$R_{\mathrm{RCV}}\ (dreq, q0, dack)\ (dreq', q0', dack') = (q0' = dreq) \wedge (dack' = (dreq \wedge (q0 \vee dack)))$$

RCV state transition diagram

- Possible states for RCV:
  $\{000, 001, 010, 011, 100, 101, 110, 111\}$
  where $b_2 b_1 b_0$ denotes state
  $\texttt{dreq} = b_2 \ \wedge \ \texttt{q0} = b_1 \ \wedge \ \texttt{dack} = b_0$
- Graph of the transition relation:



Computing $\{\!\![\mathbf{EF}\ \texttt{At111}]\!\!\}$ where $\texttt{At111} \in L_{\mathrm{RCV}}(s) \Leftrightarrow s = 111$

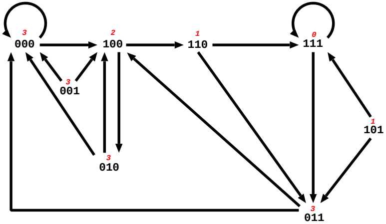


- Define:

$$
\begin{aligned}
\mathcal{S}_0 &= \{s \mid \texttt{At111} \in L_{\mathrm{RCV}}(s)\} \\
&= \{s \mid s = 111\} \\
&= \{111\}
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{S}_{n+1} &= \mathcal{S}_n \ \cup \ \{s \mid \exists s' \in \mathcal{S}_n.\ \mathcal{R}(s, s')\} \\
&= \mathcal{S}_n \ \cup \ \{b_2 b_1 b_0 \mid \\
&\qquad \exists b_2' b_1' b_0' \in \mathcal{S}_n.\ (b_1' = b_2) \ \wedge \ (b_0' = b_2 \wedge (b_1 \vee b_0))\}
\end{aligned}
$$

Computing {**EF** `At111`} (continued)



- Compute:

$$
\begin{aligned}
\mathcal{S}_0 &= \{111\} \\
\mathcal{S}_1 &= \{111\} \cup \{101, 110\} \\
&= \{111, 101, 110\} \\
\mathcal{S}_2 &= \{111, 101, 110\} \cup \{100\} \\
&= \{111, 101, 110, 100\} \\
\mathcal{S}_3 &= \{111, 101, 110, 100\} \cup \{000, 001, 010, 011\} \\
&= \{111, 101, 110, 100, 000, 001, 010, 011\} \\
\mathcal{S}_n &= \mathcal{S}_3 \quad (n > 3)
\end{aligned}
$$

- $\{\textbf{EF}\ \mathtt{At111}\} = \mathbb{B}^3 = S_{\mathrm{RCV}}$
- $M_{\mathrm{RCV}} \models \textbf{EF}\ \mathtt{At111} \Leftrightarrow S_{0\mathrm{RCV}} \subseteq S$

Symbolic model checking

- Represent sets of states with BDDs
- Represent Transition relation with a BDD
- If BDDs of $\{\psi\}$, $\{\psi_1\}$, $\{\psi_2\}$ are known, then:
  - BDDs of $\{\neg\psi\}$, $\{\psi_1 \wedge \psi_2\}$, $\{\psi_1 \vee \psi_2\}$, $\{\psi_1 \Rightarrow \psi_2\}$ computed using standard BDD algorithms
  - BDDs of $\{\mathbf{AX}\psi\}$, $\{\mathbf{EX}\psi\}$, $\{\mathbf{A}[\psi_1\ \mathbf{U}\ \psi_2]\}$, $\{\mathbf{E}[\psi_1\ \mathbf{U}\ \psi_2]]\}$ computed using straightforward algorithms (see textbooks)
- Model checking CTL generalises reachable states Iteration

History of Model checking

- CTL model checking due to Emerson, Clarke & Sifakis
- Symbolic model checking due to several people:
  - Clarke & McMillan (idea usually credited to McMillan's PhD)
  - Coudert, Berthet & Madre
  - Pixley
- SMV (McMillan) is a popular symbolic model checker:

`http://www.cs.cmu.edu/~modelcheck/smv.html`	(original)
`http://www.kenmcmil.com/smv.html`	(Cadence extension by McMillan)
`http://nusmv.irst.itc.it/`	(new implementation)

- Other temporal logics
  - CTL*: combines CTL and LTL
  - Engineer friendly industrial languages: PSL, SVA