4 – CTL/LTL MODEL CHECKING

Sommersemester 2009

Dr. Carsten Sinz, Universität Karlsruhe

CTL Model Checking

- **Model Checking Problem:** Gegeben eine Kripke-Struktur M und eine CTL-Formel f. Bestimme alle Zustände $s \in S$, in denen f gilt, d.h.

$$\{s \in S \mid M, s \models f\}$$

- Algorithmus:
  - Berechne "Labels"

    $L^*(s) = \{ g \mid g \text{ Subformel von } f \text{ mit } M,s \models g \}$

    für alle Zustände s, die angeben, welche Subformeln von f (einschließlich f selbst) in einem Zustand gelten.
  - Damit gilt: $M, s \models f \text{ gdw. } f \in L^*(s)$

CTL Model Checking

- Berechnung der Labels L*(s):
  - Beginne mit atomaren Formeln p: $p \in L^*(s)$ gdw. $p \in L(s)$
  - Fahre mit zunehmend größeren Subformeln g von f fort
  - **Annahme**: Subformeln enthalten nur die Operatoren $\neg$, $\vee$, **EX**, **EG** und **EU** (alle anderen lassen sich eliminieren)
    - Für $g = \neg h$: $g \in L^*(s)$ gdw. $h \notin L^*(s)$
    - Für $g = h_1 \vee h_2$: $g \in L^*(s)$ gdw. $h_1 \in L^*(s)$ oder $h_2 \in L^*(s)$
    - Für $g =$ **EX** h: $g \in L^*(s)$ gdw. $h \in L^*(t)$ für einen Nachfolger t von s (d.h. $(s,t) \in T$)
    - Die Fälle $g =$ **EG** h und $g =$ **EU** h sind komplizierter, siehe nächste Folien

CTL MC: Operator **EU** (I)

- **Annahme:** Bereits berechnet, in welchen Zuständen die Subformeln h_1 und h_2 gelten.
- **Frage:** In welchen Zuständen gilt **E**[h_1 **U** h_2] ?

1. Auf jeden Fall in allen Zuständen, in denen h_2 gilt. Diese können direkt markiert werden.
2. Außerdem in allen Zuständen, die zu einem Zustand aus 1. führen, und auf dem Pfad dorthin immer h_1 gilt.
   - **Idee:** Gehe in Transitionsrelation, ausgehend von den Zuständen aus 1. rückwärts

CTL MC: Operator **EU** (II)


```
checkEU(h1, h2)
{
   N := { s | h2 ∈ L*(s) }
   for all s∈N:
      L*(s) := L*(s) ∪ { E(h1 U h2) }
   while(N ≠ ∅) {
      choose s ∈ N, remove s from N
      for all t with T(t,s): // predecessors
         if(E(h1 U h2) ∉ L*(t) and h1 ∈ L*(t)) {
            L*(t) := L*(t) ∪ { E(h1 U h2) }
            add t to N
      }
   }
}
```

CTL MC: Operator **EG** (I)

- Zustände, in denen h gilt, bereits berechnet. *In welchen Zuständen gilt **EG** h ?*
- Idee:
  - Schränke Kripke-Modell M ein auf Zustände, in denen h gilt:
    $M' = (S', T', L')$ mit $S' = \{s \in S \mid M, s \models h\}, T' = T|_{S' \times S'}, L' = L|_{S'}$
  - Berechne nicht-triviale starke Zusammenhangskomponenten in M' (nicht-trivial: entweder mehr als ein Knoten oder Knoten mit "self-loop")
  - Markiere alle Zustände in M', die zu einer nicht-trivialen starken Zusammenhangskomponente führen

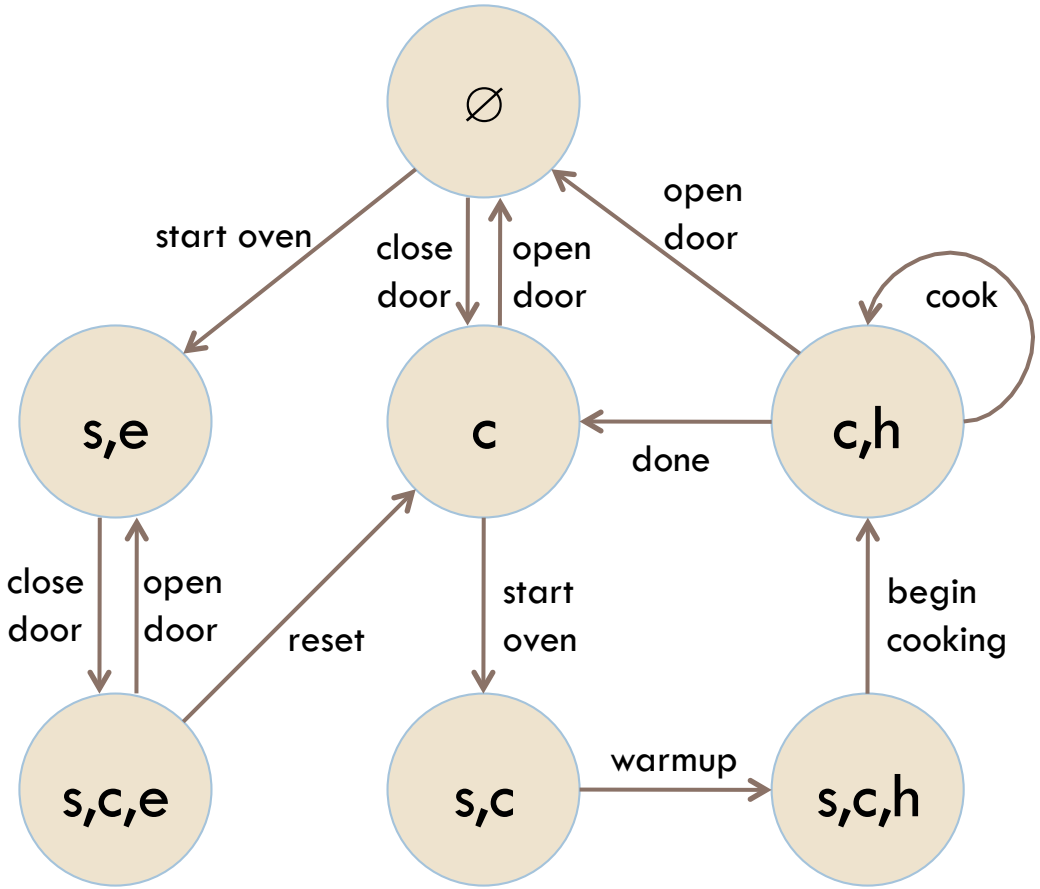
CTL MC: Operator **EG** (II)


```
checkEG(h)
{
   S' := { s | h ∈ L*(s) }
   CS:= { C | C is nontrivial SCC of S' }
   N := union of all nodes in an SCC of CS
   for all s∈N:
      L*(s) := L*(s) ∪ { EG h }
   while(N ≠ Ø) {
      choose s ∈ N, remove s from N
      for all t with t∈S' and T(t,s):
         if(EG h ∉ L*(t)) {
            L*(t) := L*(t) ∪ { EG h }
            add t to N
      }
   }
}
```

CTL MC: Komplexität

- Wie aufwendig ist die Berechnung von $\{s \in S \mid M, s \models f\}$?
  - Bearbeitung jeder Subformel: maximal O(|S|+|T|)
  - Maximal |f| Subformeln zu bearbeiten
  - Insgesamt: O(|f|•(|S|+|T|))
- **Satz:** Das CTL Model-Checking-Problem lässt sich in polynomieller Zeit lösen, genauer in O(|f|•(|S|+|T|)).
- **Anmerkung:** Kripke-Struktur kann aber bereits sehr groß sein (z.B. exponentiell in der Größe einer logischen Beschreibung in einer Hardware-Entwurfssprache)

CTL MC: Beispiel "Mikrowellenherd"



Zustand Mikrowellenherd:

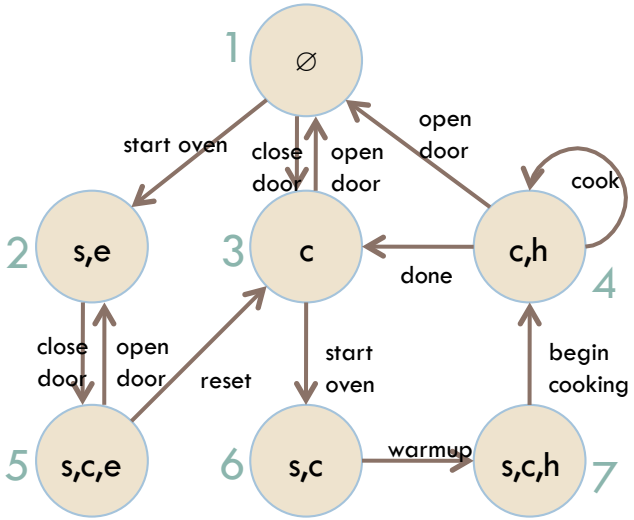
s: started
c: door closed
e: error
h: heating on

Zu prüfende Eigenschaft:

AG(s ⇒ **AF** h)

CTL MC: Beispiel (II)

- **AG**(s ⇒ **AF** h) ist äquivalent zu
  (1) ¬**EF**(s ∧ **EG** ¬h) und
  (2) ¬**E**[true **U** (s ∧ **EG** ¬h)]
- Wir prüfen nun (2), beginnend mit zunehmend wachsenden Subformeln:
  - S(s) = { 2, 5, 6, 7 }
  - S(¬h) = { 1, 2, 3, 5, 6 }
  - S(**EG** ¬h) = ?
  - …

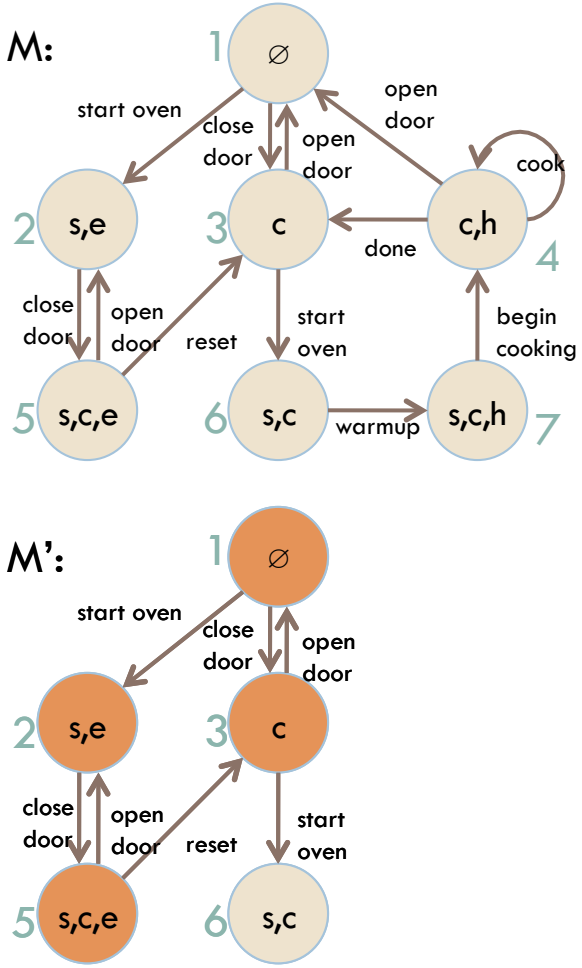


Def.: S(f) bezeichnet die Menge der Zustände, in deren Markierung f enthalten ist.

CTL MC: Beispiel (III)

- S(**EG** ¬h) = ?
  1. Berechne M' = $M|_{\neg h}$
  2. Berechne nicht-triviale SCCs von M'
  3. Markiere Zustände, die zu einem SCC führen

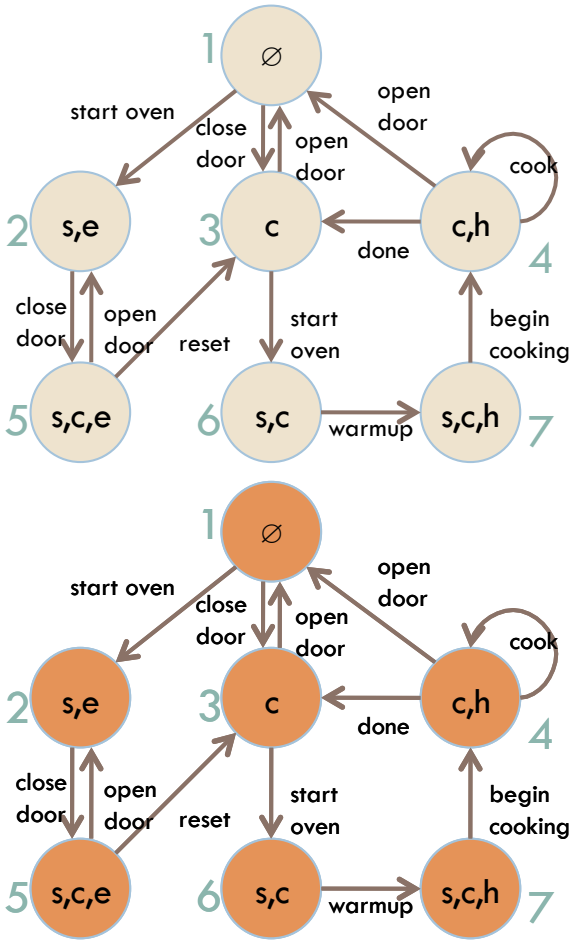
➠ S(**EG** ¬h) = { 1, 2, 3, 5 }



CTL MC: Beispiel (IV)

- S(**E**[true **U** (s ∧ **EG** ¬h)]) = ?
  1. Markiere alle Zustände, in denen s ∧ **EG** ¬h gilt (S(**EG** ¬h) = { 1, 2, 3, 5 })
  2. Markiere Zustände, die einen Zustand aus 1. erreichen (und auf dem Pfad dorthin gilt immer *true*)

➠ S(**E**[true **U** (s ∧ **EG** ¬h)]) = { 1, 2, 3, 4, 5, 6, 7 }



CTL MC: Beispiel (V)

- S(**E**[true **U** (s ∧ **EG** ¬h)]) = { 1, 2, 3, 4, 5, 6, 7 }, also S(¬**E**[true **U** (s ∧ **EG** ¬h)]) = ∅
- Die Eigenschaft **AG**(s ⇒ **AF** h) gilt also in keinem Zustand von M.

LTL Model Checking

- **Gegeben:** Kripke-Struktur *M=(S,T,L)*, LTL-Pfadformel *g*
- **Fragestellung:** Für welche Zustände s gilt $M, s \models \mathbf{A}\, g$?
- **Vorbemerkung:**
  - Wir betrachten Pfadformeln über den LTL-Operatoren **X, U** und den Booleschen Konnektiven ¬, ∨.
  - **F, G,** und **R** können ersetzt werden:

$$\mathbf{F}\, f = true\, \mathbf{U}\, f$$

$$\mathbf{G}\, f = \neg\, \mathbf{F}\, \neg f$$

$$f_1\, \mathbf{R}\, f_2 = \neg[\neg f_1\, \mathbf{U}\, \neg f_2]$$

  - $M, s \models \mathbf{A}\, g$ gilt genau dann, wenn

LTL Model Checking

- Nach Lichtenstein, Pnueli (1985/2000)
- **Hilfskonstruktion:** Abschluss CL(f) einer LTL-Pfadformel
  - $f \in CL(f)$
  - $\neg f_1 \in CL(f)$ gdw. $f_1 \in CL(f)$
  - Falls $f_1 \vee f_2 \in CL(f)$, dann auch f_1 und f_2
  - Falls $\mathbf{X} f_1 \in CL(f)$, dann auch f_1
  - Falls $\neg \mathbf{X} f_1 \in CL(f)$, dann auch $\mathbf{X} \neg f_1$
  - Falls $f_1 \, \mathbf{U} \, f_2 \in CL(f)$, dann auch f_1, f_2 und $\mathbf{X}[f_1 \, \mathbf{U} \, f_2]$
- CL(f) enthält alle Subformeln s von f sowie deren Negate ¬s, außerdem **X**s für alle **U**-Subformeln s.
- **Idee:** CL(f) enthält Formeln, die den Wahrheitswert von f beeinflussen können.

LTL Model Checking: Atome

- **Definition:** Sei M=(S,T,L) eine Kripke-Struktur über der Menge von Aussagenvariablen P. Ein Atom für M und eine LTL-Pfadformel f ist ein Paar $A = (s_A, K_A)$ mit $s_a \in S$ und $K_A \subseteq CL(f) \cup P$, wobei

  - Für jede Aussagenvariable $p \in P$: $p \in K_A$ gdw. $p \in L(s_A)$.
  - Für alle $g \in CL(f)$: $g \in K_A$ gdw. $\neg g \notin K_A$.
  - Für alle $f_1 \vee f_2 \in CL(f)$: $f_1 \vee f_2 \in K_A$ gdw. $f_1 \in K_A$ oder $f_2 \in K_A$.
  - Für alle $\neg \mathbf{X}\, g \in CL(f)$: $\neg \mathbf{X}\, g \in K_A$ gdw. $\mathbf{X} \neg g \in K_A$.
  - Für alle $f_1 \,\mathbf{U}\, f_2 \in CL(f)$:
    $f_1 \,\mathbf{U}\, f_2 \in K_A$ gdw. $f_2 \in K_A$ oder $f_1, \mathbf{X}[f_1 \,\mathbf{U}\, f_2] \in K_A$.

LTL Model Checking: Atome

- In einem Atom $A = (s_A, K_A)$ ist K_A eine maximal konsistente Formelmenge aus $CL(f) \cup P$, die mit dem Label von s_A übereinstimmt (*Hintikka-Menge* [Jaakko Hintikka, *1929]).
- **Beispiel:** f = [a **U** (**X** b)]
  - CL(f) = { f, ¬f, a, **X**b, **X**[a **U** (**X** b)], ¬a, b, ¬b, **X**¬b, **X**¬[a **U** (**X** b)], ¬**X**b, ¬**X**¬b, ¬**X**[a **U** (**X** b)], ¬**X**¬[a **U** (**X** b)] }
  - **Annahme:** $L(s_A)$= { a, b }. **Möglichkeiten für K_A:**
    - { a, b, **X**b, ¬**X**¬b, [a **U** (**X** b)], **X**[a **U** (**X** b)], ¬**X**¬[a **U** (**X** b)] }
    - { a, b, ¬**X**b, **X**¬b, [a **U** (**X** b)], **X**[a **U** (**X** b)], ¬**X**¬[a **U** (**X** b)] }
    - ...
- **Anmerkung:** Die Menge der Atome für eine Formel f und einen Zustand s kann exponentiell in |f| sein!

LTL MC: Atom-Graph, E-Sequenz

- Konstruiere zu gegebener Pfadformel f und Kripke-Struktur M=(S,T,L) einen Graph G:
  - **Knoten:** Atome $A = (s_A, K_A)$
  - **Kanten:** (A,B) ist Kante in G gdw.:
    - $(s_A,s_B) \in T$ und
    - für jede Formel $\mathbf{X}g \in CL(f)$: $\mathbf{X}g \in K_A$ gdw. $g \in K_B$
- **Definition:** Eine Eventualitäten-Sequenz (E-Sequenz) ist ein unendlicher Pfad π in G, so dass für jedes $f_1 \mathbf{U} f_2 \in K_A$ (für ein Atom A auf π) ein Atom b existiert mit $f_2 \in K_B$, das von A aus erreichbar ist.

LTL Model Checking

- **Satz:** $M, s \models \mathbf{E}\, f$ gilt genau dann, wenn es eine E-Sequenz gibt, die mit einem Atom (s,K) mit $f \in K$ startet.
- **Definition:** Eine nichttriviale SCC C eines Graphen G heißt *selbsterfüllend*, wenn es für jedes Atom A in C und für jedes f_1 **U** $f_2 \in K_A$ ein Atom B in C gibt mit $f_2 \in K_B$.
- **Lemma:** Eine E-Sequenz ausgehend von A=(s,K) existiert genau dann, wenn es einen Pfad in G ausgehend von A gibt, der zu einer selbsterfüllenden SCC führt.

LTL Model Checking: Algorithmus

- **Satz:** $M, s \models \mathbf{E}\, f$ gilt genau dann, wenn es ein Atom A=(s,K) in G gibt, so dass
  - f∈K und
  - es existiert ein Pfad in G ausgehend von A, der zu einer selbsterfüllenden SCC führt.
- **LTL Model Checking Algorithmus:**
  1. Berechne Atom-Graph G und SCCs in diesem Graph
  2. Prüfe für jedes Atom $A=(s,K_A)$ in G mit $f \in K_A$: Gibt es einen Pfad in G (von A ausgehend), der zu einer selbsterfüllenden SCC führt.

LTL Model Checking: Komplexität

- **Satz:** Der Algorithmus für LTL Model Checking besitzt die Komplexität $O((|S|+|T|)\cdot 2^{|f|})$
- **Satz:** Das LTL Model Checking Problem ist PSPACE-vollständig