6.2 Berechnungsbaumlogik CTL

Ziel: Spezifiziere temporale Eigenschaften von Kripke-Strukturen.

Alternativen: LTL = Linear-time (temporal) logic
interpretiert über Berechnungen (Worten, ohne Verzweigung)

CTL = Computation tree logic
interpretiert über Berechnungsbäumen (mit Verzweigung

CTL* = Verallgemeinerung von beiden.

Definition (Syntax von CTL):

Sei AP eine Menge atomarer Formeln.

Die Menge der CTL-Formeln über AP ist wie folgt definiert

$$\varphi ::= p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid EX\varphi \mid EG\varphi \mid E(\varphi_1 U \varphi_2)$$

mit $p \in AP$.

Außerdem verwenden wir folgende Abkürzungen:

$true := p \vee \neg p$ mit $p \in AP$

$\varphi \to \psi := \neg\varphi \vee \psi$

$EF\varphi := E(true\ U\ \varphi)$

$AX\varphi := \neg EX \neg\varphi$

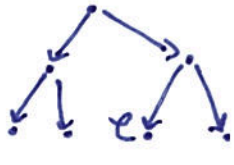
$AF\varphi := \neg EG \neg\varphi$

$AG\varphi := \neg EF \neg\varphi$

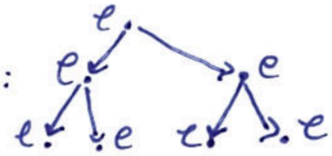
$A(\varphi_1 U \varphi_2) := (\neg EG \neg\varphi_2) \wedge \neg E(\neg\varphi_2\, U (\neg\varphi_1 \wedge \neg\varphi_2))$

Intuitive Bedeutung:

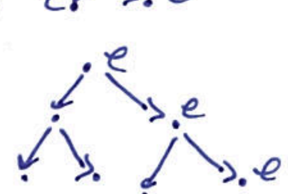
$EF\varphi$ = Es gibt einen Pfad, auf dem schließlich φ gilt:



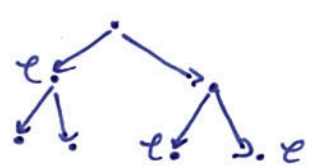
$AG\varphi$ = Auf allen Pfaden gilt immer φ:



$EG\varphi$ = Es gibt einen Pfad, auf dem immer φ gilt:

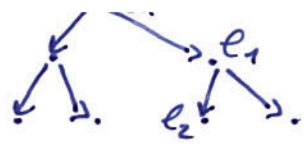


$AF\varphi$ = Auf allen Pfaden gilt schließlich φ:



$E(\varphi_1 U \varphi_2)$ = Es gibt einen Pfad, auf dem φ_1 gilt, bis φ_2 eintritt.

Dabei muss φ_2 definitiv eintreten.



F = Finally $\quad$ X = Next

G = Globally $\quad$ U = Until

Definition (Semantik von CTL):

Sei $K = (AP, S, S_0, \rightarrow, \ell)$ eine Kripke-Struktur.

Die Erfülltheitsrelation $\models$ für CTL ist induktiv über den Aufbau der Formeln und relativ zu einem Zustand $s \in S$ definiert:

$K, s \models p$, falls $p \in \ell(s)$

$K, s \models \neg\varphi$, falls nicht $K, s \models \varphi$ gilt

$K, s \models \varphi_1 \vee \varphi_2$, falls $K, s \models \varphi_1$ oder $K, s \models \varphi_2$

$K, s \models EX\varphi$, falls es einen (unendlichen) Pfad $\pi = s_0 s_1 s_2 \dots$ mit $s_0 = s$ gibt, so dass $K, s_1 \models \varphi$.

$K, s \models EG\varphi$, falls es einen (unendlichen) Pfad $\pi = s_0 s_1 s_2 \dots$ mit $s_0 = s$ gibt, so dass für alle $i \in \mathbb{N}$: $K, s_i \models \varphi$.

$K, s \models E(\varphi_1 U \varphi_2)$, falls es einen (unendlichen) Pfad $\pi = s_0 s_1 s_2 \dots$ mit $s_0 = s$ und auf diesem Pfad ein $j \geq 0$ gibt, so dass $K, s_j \models \varphi_2$ und für alle $0 \leq i < j$ gilt: $K, s_i \models \varphi_1$.

Wir schreiben $K \models \varphi$, falls $K, s_0 \models \varphi$ für alle $s_0 \in S_0$.

6.2.1 Model-Checking nach Emerson & Clarke

Das Model-Checking-Problem für CTL ist wie folgt definiert:

Gegeben: Endliche Kripke-Struktur $K = (AP, S, s_0, \to, \ell)$
und CTL-Formel φ_0 über AP.

Frage: $K \models \varphi_0$?

Statt des Model-Checking-Problems lösen wir folgende, allgemeinere Aufgabe:

Bestimme für jede Teilformel φ von φ_0 die Menge $S(\varphi)$ der Zustände, in denen φ gilt:

$$S(\varphi) := \{ s \in S \mid K, s \models \varphi \}$$

Damit lässt sich das Model-Checking-Problem durch Prüfen von

$$s_0 \in S(\varphi_0)$$ lösen.

Die Mengen $S(\varphi)$ werden iterativ berechnet, wobei in der i-ten Iteration die Teilformeln der Tiefe $i \in \mathbb{N}$ behandelt werden.

Formal ist die Tiefe wie folgt definiert:

$d(p) := 0$ für alle $p \in AP$

$d(\neg\varphi) := 1 + d(\varphi)$

$d(\varphi_1 \vee \varphi_2) := 1 + \max\{d(\varphi_1), d(\varphi_2)\}$

$d(EX\varphi) := 1 + d(\varphi)$

$d(EG\varphi) := 1 + d(\varphi)$

$d(E(\varphi_1 U \varphi_2)) := 1 + \max\{d(\varphi_1), d(\varphi_2)\}$.

Außerdem bezeichnen wir mit $cl(\varphi)$ die Menge aller Teilformeln von φ.

Der Algorithmus von Emerson & Clarke (1981, Turing Award 2008) ist wie folgt:

Input: $K = (AP, S, s_0, \to, \ell)$ und φ_0 über AP

Output: Mengen $S(\varphi)$ für alle $\varphi \in cl(\varphi_0)$.

begin
  for $i = 0$ **to** $d(\varphi_0)$ **do**
    for all $\varphi \in cl(\varphi_0)$ with $d(\varphi) = i$ **do**
      check(φ);
    od
  od
end

Der Unteralgorithmus check(φ) ist abhängig von der Form von φ.

check(p) mit $p \in AP$:
  $S(p) := \{ s \in S \mid p \in \ell(s) \}$

check($\neg\varphi$):
  $S(\neg\varphi) := S \setminus S(\varphi)$

check($\varphi_1 \vee \varphi_2$):
  $S(\varphi_1 \vee \varphi_2) := S(\varphi_1) \cup S(\varphi_2)$

check($EX\varphi$):
  $S(EX\varphi) := \{ s \in S \mid \exists t \in S(\varphi) : s \to t \}$

check($E(\varphi_1 U \varphi_2)$):
- Nutze die Äquivalenz
  $E(\varphi_1 U \varphi_2) \models\!\mid \varphi_2 \vee (\varphi_1 \wedge EX(E(\varphi_1 U \varphi_2)))$
- Zur Implementierung, starte in $S(\varphi_2)$.
  Füge in einer Breitensuche rückwärts* weitere Zustände hinzu, (entlang der Transitionsrelation) die φ_1 erfüllen:

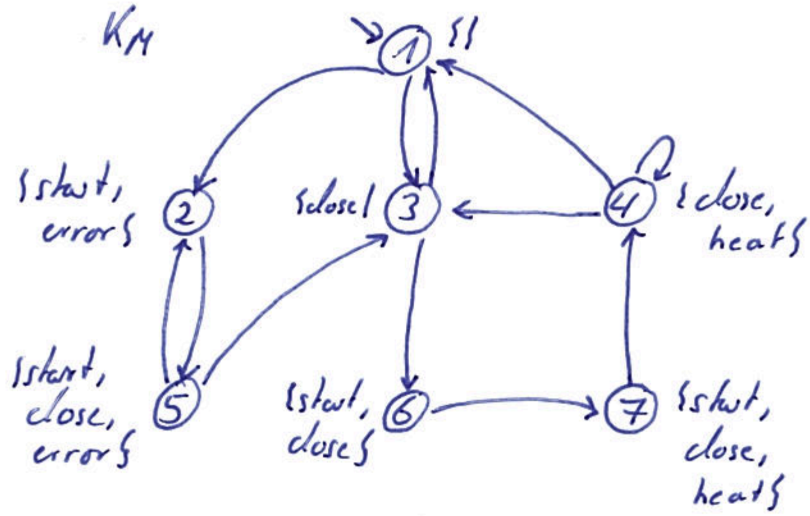
```
Q := ∅;
Q' := S(φ2);
while Q ≠ Q' do
    Q := Q';
    Q' := Q ∪ { s ∈ S | K,s ⊨ φ1 ∧ ∃t ∈ Q : s → t }
od
S(E(φ1 U φ2)) := Q;
```

check($EG\varphi$):
  Hausaufgabe.

Beispiel:

Betrachte folgendes Modell einer Mikrowelle

K_M



Prüfe, ob

$$K_M \models AG(start \to AF\, heat)$$

Zunächst wird die Formel in CTL-Syntax ohne Abkürzungen überführt:

$$\varphi_0 := \neg E(true\ U(start \wedge EG \neg heat))$$

Die Menge der Teilformeln von φ_0 ist:

$cl(\varphi_0) := \{ start,\ heat,$	....	Tiefe 0
$\neg heat,$		Tiefe 1
$EG \neg heat,\ true$ (für $heat \vee \neg heat$)		Tiefe 2
$start \wedge EG \neg heat,$		Tiefe 3
$E(true\ U(start \wedge EG \neg heat))$		Tiefe 4
$\neg E(true\ U(start \wedge EG \neg heat)) \}$		Tiefe 5

Berechne die zugehörigen Zustandsmengen:

$S(start) = \{2, 5, 6, 7\}$ $\quad S(heat) = \{4, 7\}$

$S(\neg heat) = \{1, 2, 3, 5, 6\}$

$S(EG \neg heat) = \{1, 2, 3, 5\}$ $\quad S(true) = S$

$S(start \wedge EG \neg heat) = \{2, 5\}$

$S(E(true\ U(start \wedge EG \neg heat))) = S$

$S(\varphi_0) = \emptyset$, es gilt also nicht $K_M \models \varphi_0$.

6.2.2 Satz von Hennessy und Milner

Satz (Hennessy & Milner '85):

$K \approx K'$ gdw. $\forall \varphi \in CTL: K \models \varphi$ gdw. $K' \models \varphi$.

Der Satz wird in Form von zwei Lemmas gezeigt.

Für die Richtung von links nach rechts benötigen wir den Begriff der entsprechenden Pfade.

Definition:

Seien $K = (AP, S, s_0, \rightarrow, l)$ und $K' = (AP, S', s_0', \rightarrow', l')$ Kripke-Strukturen.

Sei $R \subseteq S \times S'$ eine Bisimulation zwischen K und K'.

Zwei Pfade

$\pi = s_0 s_1 \ldots$ mit $s_i \in S$ für alle $i \in \mathbb{N}$

und

$\pi' = s_0' s_1' \ldots$ mit $s_i' \in S'$ für alle $i \in \mathbb{N}$

heißen entsprechend, falls $(s_i, s_i') \in R$ für alle $i \in \mathbb{N}$.

Lemma:

Seien $K = (AP, S, s_0, \rightarrow, l)$ und $K' = (AP, S', s_0', \rightarrow', l')$ Kripke-Strukturen und $R \subseteq S \times S'$ eine Bisimulation.

Sei ferner $(s, s') \in R$ ein Paar von Zuständen, die über R verbunden sind.

- Dann gibt es für jeden Pfad

  $\pi = s.s_1.s_2 \ldots$, der in s beginnt,

  einen entsprechenden Pfad

  $\pi' = s'.s_1'.s_2' \ldots$, der in s' beginnt.

- Umgekehrt gibt es für jeden Pfad π', der in s' beginnt, einen entsprechenden Pfad π von s aus.

Lemma:

Falls $K \approx K'$, dann gilt $\forall \varphi \in CTL$: $K \models \varphi$ gdw. $K' \models \varphi$.

Beweis:

Seien $K = (AP, S, S_0, \rightarrow, L)$ und $K' = (AP, S', S_0', \rightarrow', L')$ mit $K \approx K'$ mittels R.

Zeige für alle Zustandspaare $(s, s') \in R$ und alle CTL-Formeln φ:

$$K, s \models \varphi \quad \text{gdw.} \quad K', s' \models \varphi.$$

Der Beweis wird per Induktion über die Struktur von CTL-Formeln geführt.

IA: $p \in AP$
Da $(s, s') \in R$, folgt $L(s) = L'(s')$.
Also $p \in L(s)$ gdw. $p \in L'(s')$.
Das heißt

$$K, s \models p \quad \text{gdw.} \quad K', s' \models p.$$

IS: Angenommen die Äquivalenz gilt für φ_1 und φ_2.
Wir betrachten nur einen der fünf Fälle.

Fall $EG\,\varphi_1$:

Gelte $K, s \models EG\varphi_1$,
das heißt, es gibt einen Pfad

$$\pi = s_0 s_1 s_2 \ldots \quad \text{mit } s_0 = s,$$

so dass

$$K, s_i \models \varphi_1 \quad \text{für alle } i \in \mathbb{N}.$$

Mit dem Lemma über entsprechende Pfade gibt es einen Pfad

$$\pi' = s_0' s_1' s_2' \ldots \quad \text{mit } s_0' = s',$$

so dass $(s_i, s_i') \in R$ für alle $i \in \mathbb{N}$.

Da aber für φ_1 die Äquivalenz

$$K, s_i \models \varphi_1 \quad \text{gdw.} \quad K', s_i' \models \varphi_1$$

per Induktionsvoraussetzung gilt,
folgt

$$K', s_i' \models \varphi_1 \quad \text{für alle } i \in \mathbb{N}.$$

Mit der Semantik von CTL heißt das

$$K', s' \models EG\, \varphi_1.$$

$\square$