Übung 8 – Hausaufgaben + Temporale Logik

Formale Techniken in der Software-Entwicklung

Christian Kroiß

UPPAAL textuell

BTW:

Spezifikation des Textuellen Formats für UPPAAL:

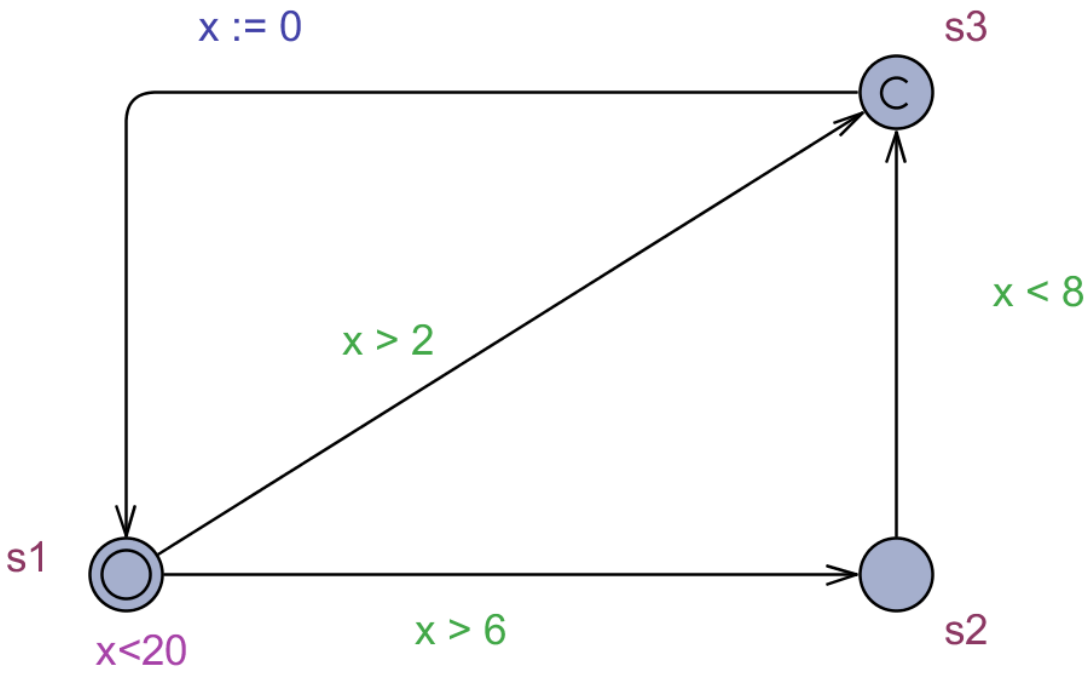
http://www.cs.aau.dk/~behrmann/utap/syntax.html

Beispiele:

http://www.brics.dk/~omoeller/v00/

Aufgabe 7-1

Gegeben: Automat mit Uhr x



Aufgabe 7-1

a) Erklären Sie das Verhalten von P im Zustand s1.

- S1 muss nach weniger als 20 ZE verlassen werden
- Ab 2 ZE ist die mittlere Trans. nutzbar
- Ab 6 ZE sind beide Trans. Nutzbar

Aufgabe 7-1

b) Hat P einen Deadlock?

Ja:

- s2 muss nicht verlassen werden
- ab 8 ZE kann s2 nicht verlassen werden

Aufgabe 7-1

c) Welchen Wert kann x minimal und maximal annehmen, wenn s3 erreicht wird?

In s3 gilt: $2 \leq x \leq 20$

Aufgabe 7-1

c) Welchen Wert kann x minimal und maximal annehmen, wenn s3 erreicht wird?

In s3 gilt: $2 \leq x \leq 20$

d) Spezifizieren Sie die Anforderung, dass der Wert von x immer kleiner als 30 ist, in Timed CTL.

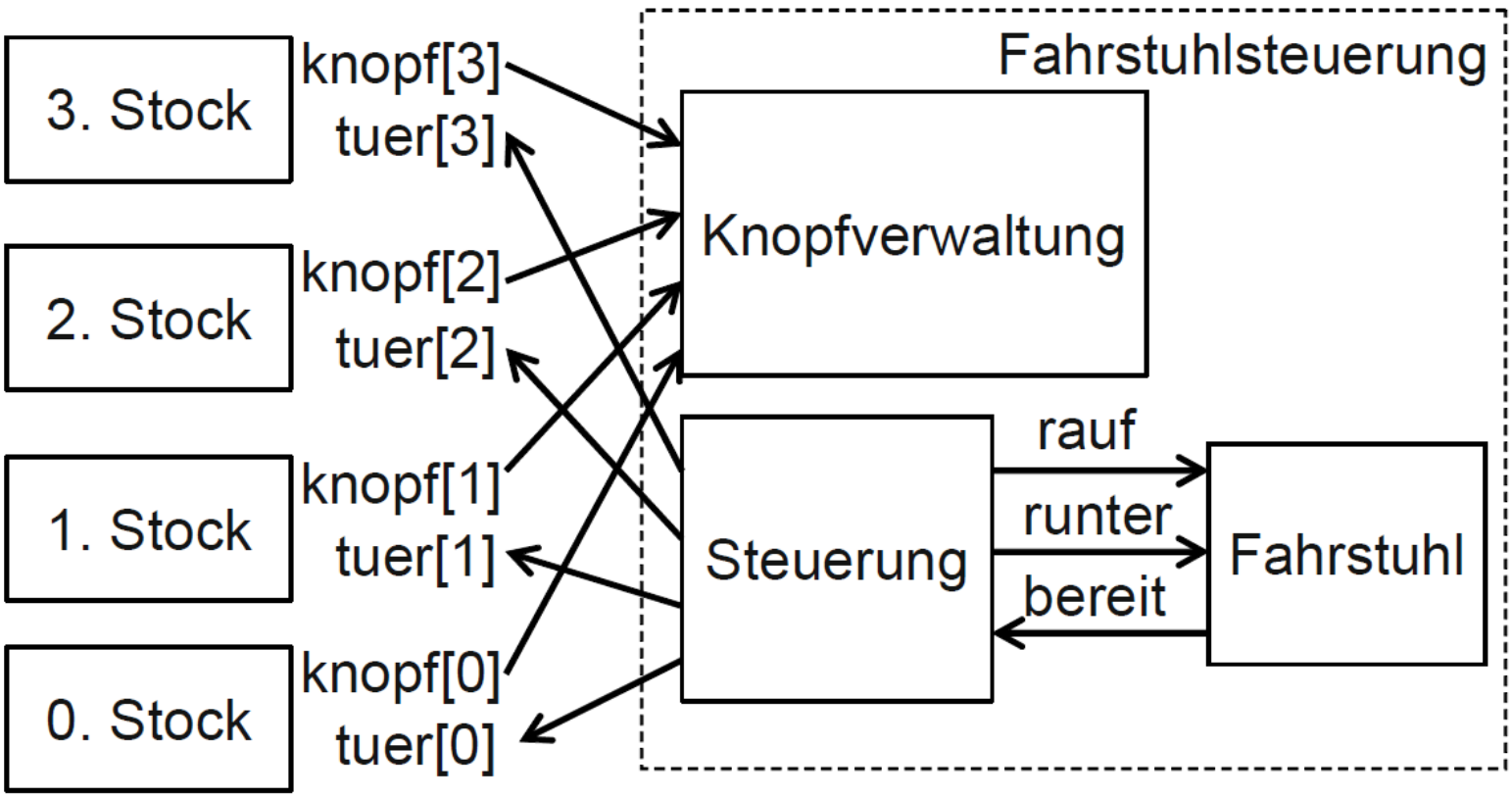
```
A[] P.x < 30
```

Aufgabe 7-1

e) Spezifizieren Sie die Anforderung, dass es die Möglichkeit gibt, dass nie s2 durchlaufen wird, in Timed CTL.

```
E[] not P.s2
```

Aufgabe 7-2


3. Stock
2. Stock
1. Stock
0. Stock
knopf[3]
tuer[3]
knopf[2]
tuer[2]
knopf[1]
tuer[1]
knopf[0]
tuer[0]
Fahrstuhlsteuerung
Knopfverwaltung
Steuerung
rauf
runter
bereit
Fahrstuhl

Aufgabe 7-2

Aufgabe:

1. Spezifikation entwickeln
2. Feststellen: wie lange dauert es maximal vom Drücken eines Knopfs bis zum Öffnen der Tür

Aufgabe 7-2

Notation:

knopf[i]: Knopf auf Stockwerk i wird gedrückt

tuer[i]: Tür in Stockwerk i wird geöffnet

Eigenschaften:

- Knopf bleibt gedrückt bis Fahrstuhl in angekommen und Tür offen
- Einsteigen dauert minimal 4 und maximal 10 ZE
- Die Fahrzeit von einem bis zum nächsten Stockwerk beträgt 4 ZE (nicht in Angabe)

Aufgabe 7-2

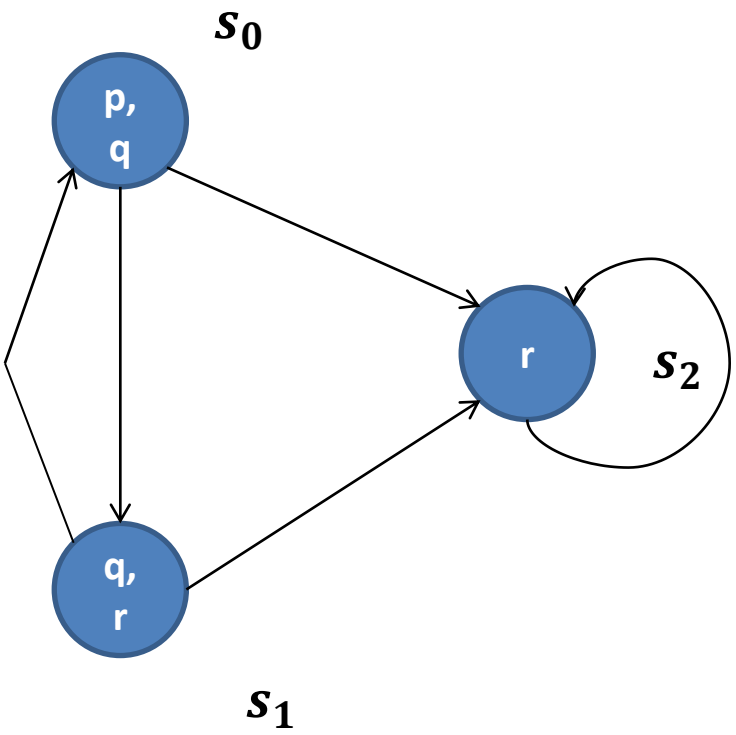
- Modell: Siehe aufzug.xml
- Frage: wie Lange dauert es maximal vom Drücken eines Knopfs bis zum Öffnen der Tür?
  - im Beispiel modellierbar durch

$$\mathrm{d_{max}} = \max(d \in \mathbb{N} \mid !\ (AG\ (Stock0.Warten \rightarrow Stock0.warten \leq d))\ )$$

Ausprobieren in UPPAAL liefert

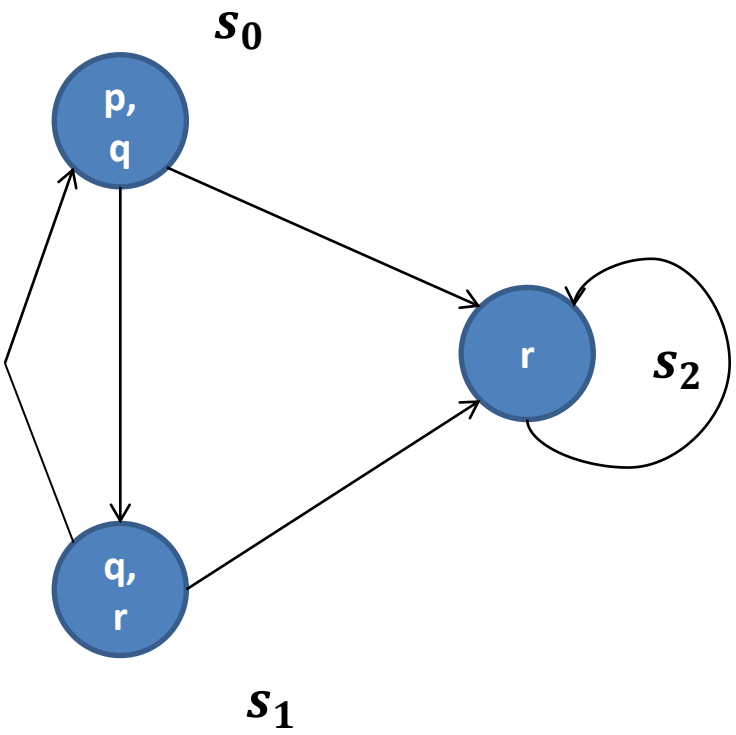
Ergebnis: d_{max}= 32 bei Trace an dem zunächst alle Anforderungen in der Reihenfolge `Stock3, Stock1, Stock2, Stock0` gestellt werden

Beispiele zu LTL



1. $M, s_0 \models p \wedge q$
2. $M, s_0 \models \top$
3. $M, s_0 \models \neg r$
4. $M, s_0 \models \mathrm{X}\, r$
5. $M, s_0 \models \mathrm{X}(q \wedge r)$ **DOESN'T HOLD**
6. $M, s_0 \models \mathrm{G}\neg(p \wedge r)$
7. $M, s_2 \models \mathrm{G}\, r$
8. For any s:
   $M, s \models \mathrm{F}(\neg q \wedge r) \rightarrow \mathrm{FG}\, \mathrm{r}$

Beispiele zu LTL



9. $\mathrm{GF}\, p$

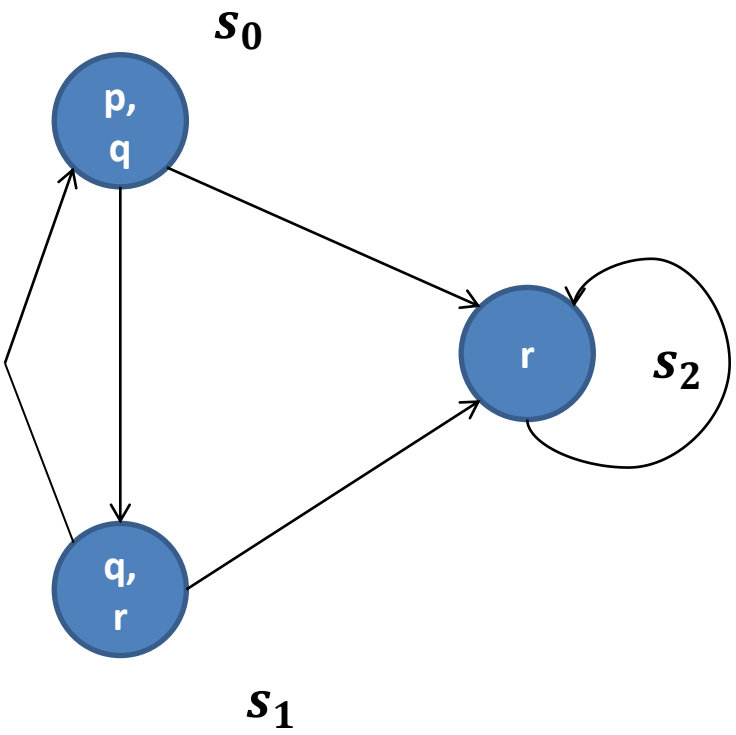
is satisfied by

$$s_0 \to s_1 \to s_0 \to s_1 \to \ \ldots$$

but not by

$$s_0 \to s_2 \to s_2 \to \cdots$$

Beispiele zu CTL



1. $M, s_0 \models p \wedge q$
2. $M, s_0 \models \neg r$
3. $M, s_0 \models \top$
4. $M, s_0 \models \mathrm{EX}(q \wedge r)$
5. $M, s_0 \models \neg\mathrm{AX}(q \wedge r)$ **DOESN'T HOLD**
6. $M, s_0 \models \neg\mathrm{EF}(p \wedge r)$
7. $M, s_2 \models \mathrm{EG}\, r$
8. $M, s_0 \models \mathrm{AF}\ \mathrm{r}$
9. $M, s_0 \models E[(p \wedge q)\ \mathrm{U}\ r]$
10. $M, s_0 \models A[p\ \mathrm{U}\ r]$
11. $M, s_0 \models \mathrm{AG}(p \vee q \vee r \rightarrow EF\ EG\ r)$

trifft zu, da in allen Zuständen, in denen $p \vee q \vee r$ gilt (alle), ein Zustand erreicht werden kann, in dem $EG\ r$ gilt (in diesem Fall s_2).

Literaturhinweis:

“Logic in computer science: modelling and reasoning about systems”, Michael Huth and Mark Ryan, 2nd edition, Cambridge University Press, 2004