ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

TIK Institut für Technische Informatik und Kommunikationsnetze

HS 2009

Dr. Kai Lampka
Jasmin Smula
Raphael Eidenbenz

Discrete Event Systems
Exercise 10

1 Computation Tree Logic (CTL) Model Checking

Sei folgende Kripke-Struktur $\mathcal{K}$ gegeben:

$$\mathcal{K} := \left\{ \begin{array}{l} \mathbb{S} := \{1,2,3,4\}; \\ \mathbb{S}_0 := \{1\}; \\ \mathbb{E} := \{(1,3),(3,2),(2,1),(2,4),(4,2)\}; \\ \mathcal{AP} := \{green, yellow, red, black\}; \\ \mathcal{L} := \{1 \mapsto red, 2 \mapsto yellow, 3 \mapsto green, 4 \mapsto black\}; \end{array} \right\}.$$

a) Bitte geben Sie den Graphen der Kripke-Struktur $\mathcal{K}$ an.

b) Entwickeln Sie nun den Computation-Tree für den Zustand $\mathbb{S}_0$ bis zur Tiefe 7.
Anmerkung: Der Wurzelknoten hat Tiefe 1.

Seien folgende CTL-Formeln gegeben:

$$\begin{array}{lll} \Omega_1 = \exists\Box green & \Omega_2 = \exists\Box\neg green & \Omega_3 = \exists(yellow \Rightarrow (\forall \bigcirc black)) \\ \Omega_4 = \forall\Box yellow & \Omega_5 = \exists \bigcirc (true \bigcup black) & \Omega_6 = \forall(black \bigcup black) \\ \Omega_7 = \exists(black \bigcup black) & \Omega_8 = \forall(\neg black \bigcup black) & \Omega_9 = \forall(\neg yellow \bigcup (\exists \bigcirc black)) \\ \Omega_{10} = \exists\Diamond black & \Omega_{11} = \forall\Diamond black & \Omega_{12} = \forall(green \bigcup (\forall(yellow \bigcup red))) \end{array}$$

c) Welche dieser Formeln sind syntaktisch inkorrekt? Begründen Sie Ihre Entscheidung.

d) Überführen Sie die syntaktisch korrekten CTL-Formeln in ihre existenzielle Normalform (ENF).

e) Konstruieren Sie die Syntaxbäume *(parse trees)* für die syntaktisch korrekten CTL-Formeln in ENF. Annotieren Sie die Knoten der Syntaxbäume mit den dazugehörigen Erfüllbarkeitsmengen $Satisfy_{\mathcal{K}}(\Omega)$ bzgl. der Kripke-Struktur $\mathcal{K}$.

f) Entscheiden Sie nun, welche dieser Formeln von $\mathcal{K}$ erfüllt werden und welche nicht.

g) Geben Sie für alle unerfüllbaren Formeln, die mit einem All-Quantor ($\forall$) beginnen, einen Pfad an, der die Formel falsifiziert.

2 Euklidscher Algorithmus

Folgender Algorithmus, bekannt als Euklids Algorithmus, berechnet den grössten gemeinsamen Teiler (ggT) zweier Zahlen $x, y \in \mathbb{N}$ mit $x \geq y$.

```
0:  int ggT(int x, int y) {
1:    if(y == 0) return x;
2:    int z = 0;
3:    while(x > 0){
4:      x--; z++;
5:      if(z == y)  z = 0;
6:    }
7:    return ggT(y, z);}
```

a) Führen Sie obigen Code mit $x = 225, y = 60$ "manuell" aus. Notieren Sie dabei die jeweiligen Aufrufe der Funktion `ggt` sowie das Endresultat. Beschreiben Sie in Worten wie der Algorithmus funktioniert.

b) Geben Sie analog zu Euklids Algorithmus ein PN an, das den ggT berechnet. Das PN soll zwei Eingabestellen P_x, P_y haben, die im Initialzustand x, bzw. y Tokens enthalten, sowie eine Ausgabestelle P_{ggT} die nach Ausführen des Token Games `ggT(`x, y`)` Tokens enthält.

Hinweis: Verwenden Sie Inhibitorkanten.