Universität Stuttgart **Sommersemester**
Formale Methoden der Informatik **Übungsblatt 4**
Prof. J. Esparza/ M. Luttenberger

Übungen zu Model Checking

Besprechung am 13.07.06

Aufgabe 4.1

Gegeben sei folgendes paralleles Programm:

```
while(true) {    ||  while(true) {
    x = x + 1;   ||      y = y + 1;
    x = x - 1;   ||      y = y - 1;
}                ||  }
```

wobei im Startzustand $x = 0$ und $y = 0$ gelte. Weiterhin sei die **LTL**-Formel $\phi = (x = 1)\,\mathbf{R}\,\mathbf{X}(y = 0)$ gegeben. Sie

(a) Zeichnen Sie die Kripkestruktur $\mathcal{K}$ für obiges Programm. $\mathcal{K}$ sollte aus 4 Zuständen bestehen.

(b) Geben Sie nun einen Büchiautomaten $\mathcal{B}_{\neg\phi}$ zu $\neg\phi$ an. Es ist nicht verlangt, die Konstruktion aus der Vorlesung zu verwenden.

(c) Bilden Sie das *vollständige* Kreuzprodukt $\mathcal{B}_{\mathcal{K}} \times \mathcal{B}_{\neg\phi}$, d.h. nicht nur den vom Startzustand erreichbaren Teilgraphen.

Hinweis: Nach Definition besitzt $\mathcal{B}_{\mathcal{K}}$ ausschließlich akzeptierende Zustände. Überlegen Sie sich, warum damit die Konstruktion des Kreuzproduktautomaton entsprechend dem Fall endlicher Automaton verlaufen darf.

(d) Überprüfen Sie, ob $\mathcal{K} \models \phi$ gilt, indem Sie die verschachtelte Tiefensuche auf $\mathcal{B}_{\mathcal{K}} \times \mathcal{B}_{\neg\phi}$ ausführen. Geben Sie ggf. das gefundene Gegenbeispiel an.

(e) Die letzte Teilaufgabe hat das lokale Modelchecking für die **LTL**-Formel ϕ betrachtet. D.h. man ist nur daran interessiert gewesen, ob ϕ im Startzustand von $\mathcal{K}$ gilt.

Überlegen Sie sich anhand von $\mathcal{B}_{\mathcal{K}} \times \mathcal{B}_{\neg\phi}$ einen *globalen* **LTL**-Modelcheckingalgorithmus. D.h. der Algorithmus soll *alle* Zustände einer Kripkestruktur bestimmen, welche die gegebene **LTL**-Formel (nicht) erfüllen. Welche Komplexität hat dieser Algorithmus?

Führen Sie Ihren Algorithmus auf $\mathcal{B}_{\mathcal{K}} \times \mathcal{B}_{\neg\phi}$ aus.

(f) *Falls Sie die Grundlagenvorlesung gehört haben bzw.* ***CTL*** *kennen*: In den Folien finden Sie die Definition der Logik **CTL***, welche **LTL** und **CTL** vereint. Überlegen Sie sich, wie Sie durch Kombination der Algorithmen für das CTL-Model Checking und dem Algorithmus aus der letzten Teilaufgabe algorithmisch überprüfen können, ob eine gegebene Kripkestruktur ein Modell für eine **CTL***-Formel ist.

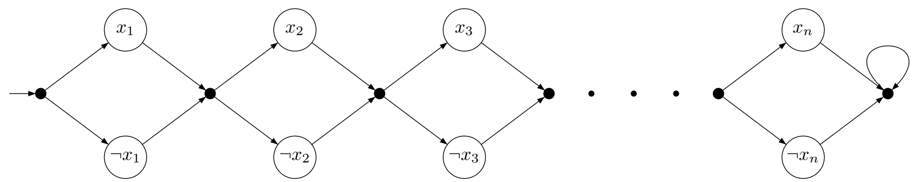
Aufgabe 4.2

Aus der Vorlesung wissen Sie, dass Modelchecking für **LTL PSPACE**-vollständig ist. In dieser Aufgabe sollen Sie zeigen, dass dieses Problem bereits **NP**-schwer ist[1], falls man nur das Fragment $\mathbf{LTL_F}$ von **LTL** betrachtet, in welchem einzig der Temporaloperator $\mathbf{F}$ (und die Negation $\neg\mathbf{F} \equiv \mathbf{G}$) erlaubt ist. D.h. $a \in \mathbf{AP}$ ist eine $\mathbf{LTL_F}$-Formel, und sind ϕ_1, ϕ_2 $\mathbf{LTL_F}$-Formeln, so auch

$$\phi_1 \vee \phi_2, \neg\phi_1, F\phi_1.$$

Ansonsten enthält $\mathbf{LTL_F}$ keine weiteren Formeln.

Reduzieren Sie 3-KNF-SAT auf das Entscheidungsproblem $\mathcal{K} \models \phi$, wobei $\mathcal{K}$ und $\phi \in \mathbf{LTL_F}$ die Eingabeparameter sind. Konstruieren Sie hierfür zu gegebener 3-KNF-Formel $\psi = \bigwedge_{i=1}^{k} K_i$ mit Klauseln K_i eine Kripkestruktur $\mathcal{K}$ der Form



und eine $\mathbf{LTL_F}$-Formel ϕ, so dass $\mathcal{K} \models \phi$ genau dann gilt, wenn ψ erfüllbar ist. Hierbei seien $x_1, \ldots, x_n$ genau die Variablen, welche in ψ auftreten. Wählen Sie $\mathbf{AP} = \{K_1, \ldots, K_k\}$.

[1] Erinnerung: D.h., jedes Problem aus **NP** lässt sich in Polynomialzeit darauf reduzieren