Moses Gedächtnisprotokoll SS22

Aufgabe 1 (x Punkte)

Formale Modellierung mit Symbolen, Mengen und Funktionen
Mengen:

- $PERSON$,
- $FIRMA$,
- $AUSBILDUNG$
- $ANGESTELLT \subseteq PERSON \times FIRMA$
- $ausbildungen-von : PERSON \to P(AUSBILDUNG)$
- $arbeiten-bei : FIRMA \to P(AUSBILDUNG)$ gibt die Menge aller Berufe zurück, die bei einer Firma ausgeübt werden.

a) Menge definieren: alle Personen, die in Firma a angestellt sind

b) Relation definieren: Schnittmenge der gemeinsamen Berufe zweier Firmen
$gemeinsame-jobs(f, f') =$

c) Funktion definieren $personal-filter : PERSON^* \times FIRMA \to P(PERSON)$
Funktion soll die Menge der Personen zurückgeben, die bei Firma f arbeiten und in der Folge enthalten sind

d) Bedeutung einer Relation in natürlicher Sprache erklären
$R := \{(f, f') \in FIRMA \times FIRMA | f = f' \vee |arbeiten-bei(f)| < |arbeiten-bei(f')|\}$

Aufgabe 2 (x Punkte)

Formale Modellierung von Anforderungen:
4 Anforderungen an Relation $ANGESTELLT \subseteq PERSON \times FIRMA$

- Zwei Personen sind bei maximal einer Firma gemeinsam angestellt

Aufgabe 3 (x Punkte)

// Diese Aufgabe war wie eine Art Lückentext gestellt, bei dem einzelne Teile der Herleitung bereits vorgegeben, andere ergänzt werden mussten.

Syntax und Semantik:
a) $< x := (x \oplus 4); skip >$ Herleitung in IMP angeben mit $\sigma(x) = 3$

b) und Ergänzung von Kalkülregeln bei einer IMP ähnlichen Sprache
$b \in BExp$ und $a_1, a_2 \in AExp$ und $X \in Var$
$X := b \ ? \ a_1 : a_2$
Zwei Fälle:

1. Wenn b zu true ausgewertet wird, wird X der Wert von a_1 zugewiesen
2. Wenn b zu false ausgewertet wird, wird X der Wert von a_2 zugewiesen

es gibt eine Lösung mit zwei zusätzlichen Kalkülregeln

if und while nicht in Sprache enthalten

Aufgabe 4 (x Punkte)

Determinismusbeweis mit RASM: do c_1 after c_2, add x v (Lückentexte)

Aufgabe 5 (x Punkte)

a) aus Transistionsystem ein Gleichungssystem machen b) einen Prozessausdruck mit
(Q ||| R) || S
zu einer gegebenen Menge von Traces konstruieren

Aufgabe 6 (20 Punkte)

a)
// Leider bin ich mir bei den TS und Übergängen nicht mehr 100% sicher...

- Zwei Transitionssysteme + entsprechende Automaten gegeben.
- asynchrone shared-memory-Komposition
- Bei resultierendem Transitionssystem S_0 und → ergänzen.

$$TS_1 = (S_1, S_0^1, E_1, \rightarrow_1)$$
$$S_1 = \{(21, a), (22, a), (21, b), (22, b)\}$$
$$S_0^1 = \{(21, a)\}$$
$$E_1 =$$
$$\rightarrow_1 = \{\}$$

$$TS_2 = (S_2, S_0^2, E_2, \rightarrow_2)$$
$$S_2 = \{(11, a), (11, b)\}$$
$$S_0^2 = \{(11, a)\}$$
$$E_2 = \{y, z\}$$
$$\rightarrow_2 = \{(21a, y, 21b), (21b, z, 21a)\}$$

$$TS = (S, S_0, E, \rightarrow)$$
$$S = \{(11, 21, a), (11, 22, a), (11, 21, b), (11, 22, b)\}$$
$$S_0 = L\ddot{o}\text{sung hier anzugeben}$$
$$E = \{x, y, z\}$$
$$\rightarrow = L\ddot{o}\text{sung hier anzugeben}$$

b)
Mit Gegenbeispiel (zu konstruierendes TS) zeigen, dass ¬ P1(TS) ⇒ P2(TS)

$$P1 = \forall s \in S_0 : (s, e, s')$$
$$P2 = \forall s, s' \in S :$$

insgesamt 100 punkte