Theoretische Informatik: Blatt 6

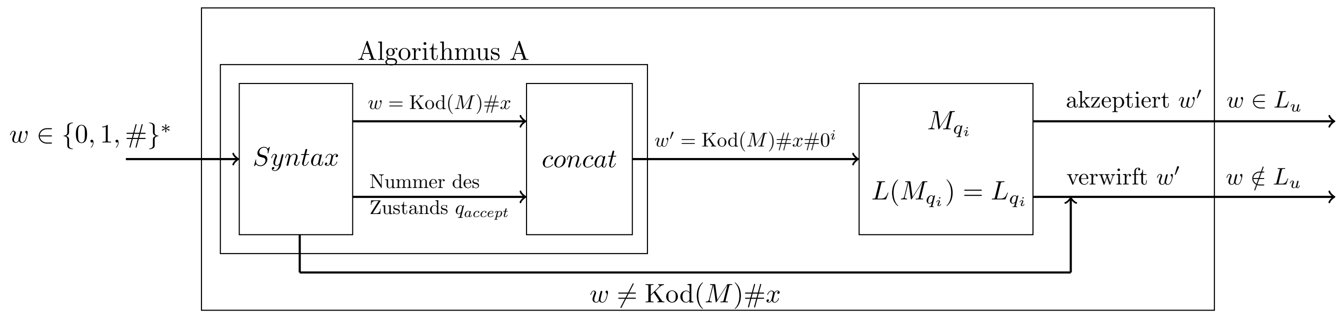
Abgabe bis 9. Oktober 2015
Assistent: Sacha Krug, CHN D 42

Linus Fessler, Markus Hauptner, Philipp Schimmelfennig

Aufgabe 16

Wir wollen zeigen, dass $L_{q_i} \notin \mathcal{L}_R$, also nicht rekursiv ist. Dazu machen wir einen Widerspruchsbeweis.
Annahme: L_{qi} sei rekursiv. Wir zeigen $L_u \leq_R L_{q_i}$.

Algorithums B für L_U



Für ein Wort w entscheiden wir zuerst, ob die Syntax einem Wort in L_u entspricht. Falls nein, ist $w \notin L_u$. Falls ja, wählen wir als i die Nummer des Zustands q_{accept} in der Kodierung von M und erzeugen daraus w'. Falls die Anzahl Zustände der TM M nicht $\geq i+1$ ist, verwerfen wir w (diese Arbeit führt der Algorithmus A aus). Ansonsten fahren wir wie folgt fort: Da eine TM aus q_{accept} nicht mehr herausgeht, ist $w \in L_u$, falls M_{q_i} w' akzeptiert, also M den i-ten Zustand erreicht. Falls M_{q_i} w' verwirft, akzeptiert M also w nicht.
Da A immer hält und nach Annahme M_{q_i} immer hält (da rekursiv), hält auch B immer. Also gilt $L_u \leq_R L_{q_i}$. Aus $L_{q_i} \in \mathcal{L}_R$ folgt also $L_u \in \mathcal{L}_R$. Aber wir wissen $L_u \notin \mathcal{L}_R$. Das ist ein Widerspruch, womit $L_{qi} \notin \mathcal{L}_R$ gilt.

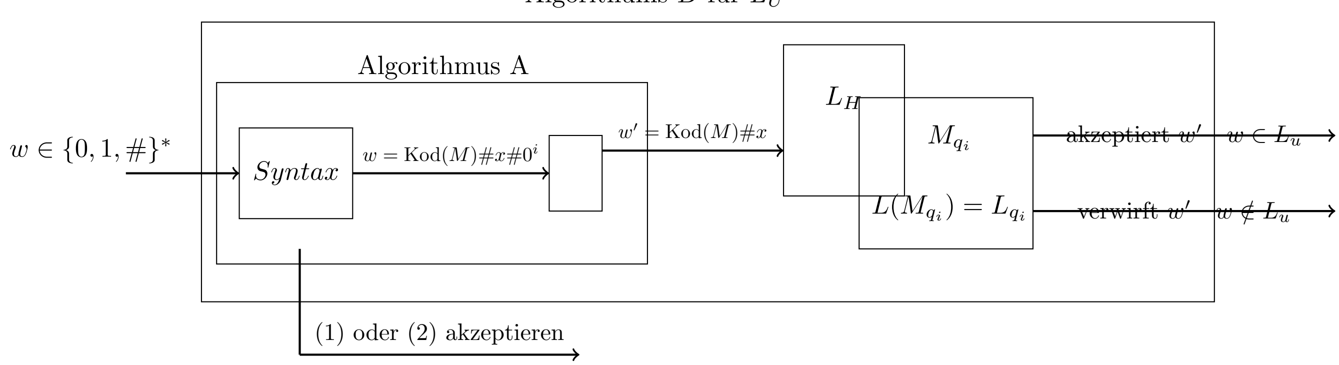
Aufgabe 17

Wir wollen zeigen, dass $L_{q_i}{}'$ nicht in $\mathcal{L}_R$ ist. Dazu genügt es nach *Lemma 5.4* zu zeigen, dass $(L_{q_i}{}')^C \notin \mathcal{L}_R$.

$$(L_{q_i}{}')^C = \begin{cases} (1) & w \neq \text{Kod}(M)\#x\#0^i \\ (2) & w = \text{Kod}(M)\#x\#0^i \text{ und M hat weniger als } i+1 \text{ Zustände} \\ (3) & w = \text{Kod}(M)\#x\#0^i\text{, M hat mehr als } i \text{ Zustände, M erreicht den } i\text{-ten Zustand nicht} \end{cases}$$

Um zu zeigen, dass $(L_{q_i}{}')^C \notin \mathcal{L}_R$ benutzen wir einen Widerspruchsbeweis.
Annahme: L_H ist rekursiv. Wir zeigen $L_H \leq_R (L_{q_i}{}')^C$.

Algorithums B für L_U

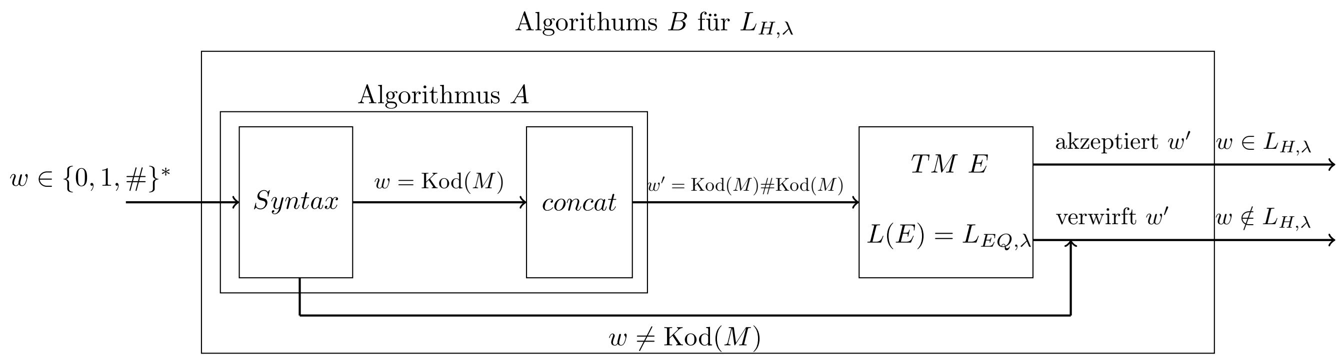


Für ein Wort w bestimmt Algorithmus A zunächst seine Syntax. Ist diese nicht passend für ein Wort in L_u (1) wird das Wort akzeptiert. Ist diese passend aber hat M weniger als $i+1$ Zustände (2) wird das Wort akzeptiert.
Nach der Annahme ist L_H rekursiv, daher kann der Algorithmus A eine TM simulieren, die prüft, ob $w' = \text{Kod}(M)\#x \in L_H$ und damit ob M auf x hält. Hält M nicht, ist (3) erfüllt und w wird akzeptiert. Hält M, so kann M von einer TM $M_M$in endlicher Zeit simuliert werden. Dabei wird festgehalten, ob M den i-ten Zustand mindestens einmal erreicht. Falls ja, so wird w verworfen, falls nein, ist (3) erfüllt und w wird akzeptiert.

B hält also immer.

Aufgabe 18

Wir wollen zeigen, dass $L_{Eq,\lambda} \notin \mathcal{L}_R$. Dazu nehmen wir an, dass $L_{Eq,\lambda}$ rekursiv ist, und zeigen für $L_{H,\lambda} = \{\text{Kod}(M) \mid x \in \{0,1\}^* \text{ und } M \text{ hält auf } \lambda\}$, dass $L_{H,\lambda} \leq_R L_{Eq,\lambda}$.



Für ein Wort w testet der Algorithmus A zunächst die Syntax, ob $w = \text{Kod}(M)$ für eine TM M, sonst wird w von B verworfen und $w \notin L_{H,\lambda}$.
Ist hingegen $w = \text{Kod}(M)$, konstruiert der Algorithmus A $w' = \text{Kod}(M)\#\text{Kod}(M)$ als Spezialfall von $\text{Kod}(M)\#\text{Kod}(\overline{M})$ und gibt es als Eingabe für die TM E weiter. Verwirft E w', hat M nicht auf λ gehalten. Also ist $w \notin L_{H,\lambda}$. Akzeptiert E, dann gilt die Tautologie $\lambda \in L(M) \leftrightarrow \lambda \in L(M)$. Um zu sagen, dass $\lambda \in L(M)$ oder $\lambda \notin L(M)$, muss M auf λ gehalten haben.
Nach der Annahme hält E immer, und damit auch B. Also gilt $L_{H,\lambda} \leq_R L_{Eq,\lambda}$.
Aus $L_{Eq,\lambda} \in \mathcal{L}_R$ folgt also $L_{H,\lambda} \in \mathcal{L}_R$. Aber wir wissen $L_{H,\lambda} \notin \mathcal{L}_R$. Damit haben wir unseren Widerpsruch und die Annahme war falsch $\Rightarrow L_{Eq,\lambda} \notin \mathcal{L}_R$.