Úloha č. 1

Pro sestrojení M' je v daném stroji M potřeba zbavit se pouze přechodů, které využívají všechny tři akce současně. K tomuto účelu si pro každý stav $q \in Q(M)$ pořídíme dva nové stavy, r_{qL} a r_{qR}. Jelikož stavů M je pouze konečně mnoho, i těchto nově vytvořených stavů bude konečně mnoho.

Máme-li pak v M přechod do (q, a, L), $q \in Q$, $a \in \Sigma$ používající všechny tři akce, nahradíme ho přechodem do (r_{qL}, a, N). Ze stavu r_{qL} pak budou existovat právě přechody $(r_{qL}, a) \to (q, a, L) \forall a \in \Sigma$. Protože i Σ je konečná, těchto pravidel bude také pouze konečně mnoho. Obdobnou transformaci provedeme pro přechody posouvající hlavu vpravo.

Snadno vidíme, že tato dvojice přechodů je ekvivalentní s původním přechodem, avšak nepoužívá nikdy všechny tři akce zároveň. Sestrojili jsme tak ekvivalentní Turingův stroj M', který nikdy nevykonává všechny akce najednou, čímž je úloha splněna.

Úloha č. 2

Jak víme ze cvičení, každý Turingův stroj lze převést na stroj s páskou, která je potenciálně nekonečná pouze doprava. Zároveň každý Turingův stroj dokážeme převést na variantu, ve které žádný přechod nenechává čtecí hlavu na místě. Jelikož vždy můžeme začít těmito dvěma převody, budeme dále BÚNO předpokládat, že převádíme stroj s poloviční páskou, který v každém přechodu posouvá hlavu doleva nebo doprava.

Jediný rozdíl mezi daným strojem a námi hledanou podobou pak je to, že původní stroj dokáže posouvat čtecí hlavu vlevo, zatímco cílový stroj umí pohyb RESET. Abychom simulovali pohyb doleva, rozšíříme nejprve abecedu stroje. Pro každý znak $a \in \Sigma$ zavedeme tři nové znaky $a^?$, a^X a $a^\#$. Pro každý stav $q \in Q$ navíc zavedeme čtyři nové stavy, c_q^1, c_q^2, n_q, y_q.

Máme-li v původním stroji přechod do (q, b, L), nahradíme ho přechodem do $(n_q, b^\#, RESET)$. Nově vytvořené stavy budou obsahovat pro všechna $a \in \Sigma$ následující pravidla:

$$
\begin{array}{lll}
(n_q, a^X) \to (n_q, a^X, R) & (n_q, a^?) \to (n_q, a^X, R) & (n_q, a) \to (c_q^1, a^?, R) \\
(c_q^1, a) \to (c_q^2, a, R) & (c_q^1, a^\#) \to (c_q^1, a, RESET) & (c_q^1, a^?) \to (q, a, RESET) \\
(c_q^2, a) \to (n_q, a, RESET) & (c_q^2, a^\#) \to (y_q, a, RESET) & \\
(y_q, a^X) \to (y_q, a, R) & (y_q, a^?) \to (q, a, R) &
\end{array}
$$

Myšlenka tohoto postupu je následující. Nejprve najdeme pole, které je dvě políčka vlevo od označeného $b^\#$. Znakem a^X označujeme pole, která hledaná určitě nejsou. Znak $a^?$ je ten, o němž se toto snažíme dozvědět. Stav n prochází zleva hledaný řetězec, přechází všechna a^X, pole $a^?$ přepíše na a^X (není to hledané) a příští, ještě neobjevené pole přepíše na $a^?$, načež přejde do kontrolního stavu c^1. Stav c^1 se pouze posune doprava a přejde do stavu c^2 (hledáme pole o dvě vlevo).

Stav c^2 zkontroluje, zda se nacházíme v označeném poli $b^\#$. Pokud ne, vrátí se do stavu n, hlavu posune zpět na začátek, a celý proces začíná znovu s dalším posunutým písmenem. V průběhu tohoto výpočtu bude tedy na začátku pásky několik znaků typu a^X, hned za posledním

z nich znak typu $a^?$, za ním může být několik běžných znaků a za nimi znak $b^\#$, za kterým jsou opět jen běžné znaky.

Pokud hlava ve stavu c^2 naopak přečte znak typu $a^\#$, resetuje hlavu, ale přepíše tento znak zpět do jeho původní podoby a přejde do „úspěšného“ stavu y. Stav y přepisuje všechny znaky a^X do jejich původní podoby, dokud se nedostane ke znaku $a^?$. V tuto chvíli ví, že je dvě políčka vlevo od počátečního znaku, tudíž tento znak přepíše do původní podoby, posune hlavu doprava a přejde do „cílového“ stavu q.

Tím jsme dostali stroj do situace, kde písmeno na počáteční pozici jsme přepsali na b, všechna ostatní písmena pásky zůstala zachována a stroj se nachází ve stavu q vlevo od počátečního stavu. Převedli jsme tak úspěšně přechod (q, b, L) do instrukcí našeho stroje.

Můžeme si všimnout, že tento přístup nefunguje, pokud jsme začínali na prvním nebo druhém poli pásky. V takovém případě by se c^2 přeskočilo $a^\#$, čímž bychom porušili invariant algoritmu, stav n by přečetl $a^\#$ a výpočet by selhal, protože pro tuto situaci neexistuje přechod. Tyto dva krajní případy je tedy potřeba ošetřit.

Nejprve si můžeme uvědomit, že počáteční pole nemůže být první pole pásky. Simulujeme totiž posun hlavy doleva, který z prvního pole automatu s poloviční páskou nelze provést. Zbývá tedy situace, kdy začínáme na druhém poli pásky. V takovém případě znak $a^\#$ přečteme ve stavu c^1, kterému jsme pro tuto situaci přidali speciální přechodové pravidlo. Stroj přepíše $a^\#$ na a a stále ve stavu c^1 resetuje hlavu stroje, což je ekvivalentní posunu doleva. Na prvním poli pásky v této situaci musí být znak $a^?$. Takový znak c^1 v „běžném“ případě nemůže přečíst, tudíž víme, že jsme na prvním poli pásky, a můžeme přejít do stavu q, přepsat $a^?$ na a a resetovat hlavu, což odpovídá setrvání na místě. Tím jsme opět simulovali přechod (q, b, L).

Jak jsme ukázali výše, jsem schopni pomocí našeho stroje generovat posloupnost instrukcí, na jejímž konci je výpočet ve stejném stavu jako po přechodu posouvajícím čtecí hlavu doleva. Rozšířili jsme přitom stroj jen o konečně mnoho znaků abecedy, konečně mnoho stavů a konečně mnoho přechodových pravidel. Jsme tudíž schopni převést libovolný Turingův stroj na cílenou variantu stroje, čímž je úloha splněna.