Cvičenie 10 - Riceova veta, redukcie problémov

Ing. Viliam Hromada, PhD.

C-510
Ústav informatiky a matematiky
FEI STU

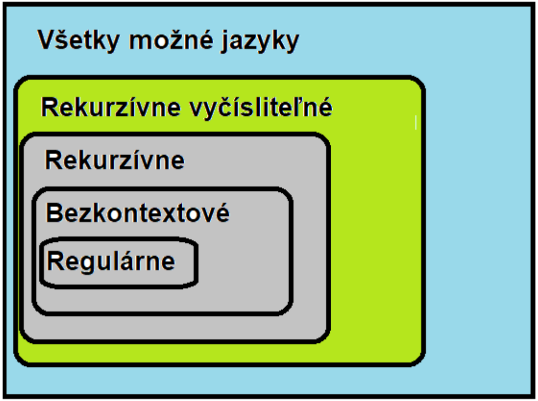
viliam.hromada@stuba.sk

9.12.2020

Úlohy

Pre uvedené jazyky rozhodnite, či sú

1. rekurzívne (sivá oblasť)
2. rekurzívne vyčísliteľné, avšak nie rekurzívne (zelená oblasť)
3. nie rekurzívne vyčísliteľné (modrá oblasť)



Aby sme ukázali, že jazyk je:

1. rekurzívny
   - musíme nájsť TS, ktorý zastaví pre všetky vstupy, ktorý ho akceptuje.
2. rekurzívne vyčísliteľný, ale nie rekurzívny
   - musíme nájsť TS, ktorý zastaví pre tie vstupy, ktoré akceptuje
   - navyše musíme ukázať, že **neexistuje TS, ktorý by zastavil pre všetky vstupy** - alebo Riceovou vetou, alebo redukciou z iného problému, ktorý nie je rozhodnuteľný
3. nie je rekurzívne vyčísliteľný
   - musíme ukázať, že **neexistuje TS**, ktorý by ho dokázal akceptovať - pomocou redukcie z iného problému, ktorý nie je rekurzívne vyčísliteľný

Jazyky

- $L_1 = \{w \in \{0,1\}^* \mid w \text{ kóduje TS } M \text{ s 3 stavmi}\}$
- $L_2 = \{w \in \{0,1\}^* \mid w \text{ kóduje TS } M \text{ a vstup } x \text{ na ktorý zastaví }\}$
- $L_3 = \overline{L_2} = \{w \in \{0,1\}^* \mid w \text{ kóduje TS } M \text{ a vstup } x \text{ na ktorý nezastaví }\}$
- $L_4 = \{L \mid L \text{ je RE jazyk a } |L| \geq 3\}$
- $L_5 = \{L \mid L \text{ je RE jazyk a } |L| \geq 3\}$
- $L_6 = \{L \mid L \text{ je RE jazyk a } L \text{ je konečný }\}$
- $L_7 = \{L \mid L \text{ je RE jazyk a } L \text{ je nekonečný }\}$
- $L_8 = \{w \mid w \text{ kóduje Turingov stroj, ktorý nezastaví pre žiaden vstup}\}$
- $L_9 = \{w \mid w \text{ kóduje Turingov stroj, ktorý zastaví pre aspoň jeden vstup}\}$

Jazyk L_1

Jazyk $L_1 = \{w \in \{0,1\}^* \mid w$ kóduje TS M s 3 stavmi$\}$ predstavujú napríklad reťazce:

- $w = 0101001010011001010001010 0$

  t.j. 3 stavový TS s prechodmi $\delta(q_1, 0) = (q_2, 0, R), \delta(q_2, 0, q_3, 0, R)$
- $w = 0100100101001101010000100 10$

  t.j. 3 stavový TS s prechodmi $\delta(q_1, 1, q_2, 0, R), \delta(q_1, 0, q_4, 1, L)$

Do jazyka nepatrí napríklad:

- $w = 0101001010011$

  t.j. 2 stavový TS s prechodom $\delta(q_1, 0) = (q_2, 0, R)$
- $w = 1010$

  t.j. preddefinovaný TS s 1 stavom bez prechodov

L_1

Tento jazyk hovorí o **štruktúre Turingovych strojov**, nie o ich **jazykoch**, preto Riceova veta nie je aplikovateľná.

Vedeli by sme nájsť Turingov stroj, ktorý spracuje reťazec w, vždy zastaví a zistí, či príslušný TS má 3 stavy?

L_1

Možná idea riešenia cez viacpáskový Turingov stroj:

1. Na prvej páske je na začiatku napísaný vstupný binárny reťazec - predpokladáme, že popisuje nejaký Turingov stroj.
2. Teda, že binárny reťazec musí obsahovať reťazce popisujúce 1 prechod: $0^i 1 0^j 1 0^k 1 0^l 1 0^m$ medzi ktorými sú 11
3. Urobili by sme teda Turingov stroj, ktorý by postupne takýto reťazec spracúval, pričom by sa zameral na 0^i a 0^k, teda skupiny núl popisujúce stavy v danom prechode. Zároveň by kontroloval, či daný reťazec predstavuje korektný kód Turingovho stroja.

L_1

Možná idea riešenia cez viacpáskový Turingov stroj:

4. Počas čítania nejakej skupiny 0^i alebo 0^k by ju ukladal na druhú pásku. Zároveň by na tretej páske držal všetky **rôzne** stavy, ktoré našiel - reprezentované napríklad skupinami núl oddelených jednotkou, pričom jedna skupina n-núl reprezentuje stav q_n.
5. Po prečítaní jednej skupiny 0^i alebo 0^k by sa pozrel, či je taká skupina už na tretej páske. Ak áno, ide ďalej, ak nie, tak ju tam pridá.
6. Po prečítaní celého vstupu w by sa pozrel, či sú na tretej páske práve 3 rôzne stavy. Ak áno - vstup **popisuje** TS s 3 stavmi. Ak nie - vstup **nepopisuje** TS s 3 stavmi.
7. Daný TS by teda určite **skončil**, navyše s korektnou odpoveďou

L_1

- Našli sme teda Turingov stroj (resp. navrhli sme), ktorý dokáže rozpoznať jazyk L_1 a navyše vždy skončí - aj pre tie TS, ktoré majú 3 stavy, aj pre tie, ktoré nemajú 3 stavy.
- Jazyk L_1 je teda **rozpoznateľný**, čiže rekurzívny.

L_2

Jazyk $L_2 = \{w \in \{0,1\}^* \mid w$ kóduje TS M a vstup x na ktorý zastaví $\}$ predstavujú napríklad reťazce:

- $w = 010100101001100101000101001110 0$

  t.j. TS s prechodmi $\delta(q_1, 0) = (q_2, 0, R), \delta(q_2, 0, q_3, 0, R)$ a vstupom 00
- $w = 0100100101001101010000100101111$

  t.j. TS s prechodmi $\delta(q_1, 1, q_2, 0, R), \delta(q_1, 0, q_4, 1, L)$ a vstupom 1

Do jazyka nepatrí napríklad:

- $w = 0101000101001100010101010110100100100 11100$

  t.j.TS s prechodmi
  $\delta(q_1, 0) = (q_3, 0, R), \delta(q_3, 0) = (q_1, 0, L), \delta(q_1, 1) = (q_2, 1, R)$ a vstupom 00.

L_2

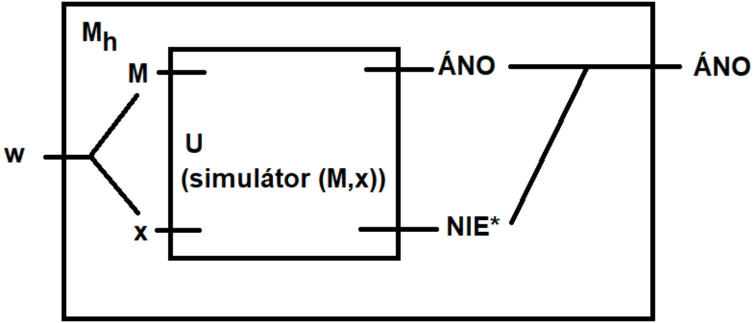
Je tento jazyk rozpoznateľný? Riceovu vetu použiť nevieme, lebo jazyk nehovorí o vlastnostiach RE jazykov.

Vedeli by sme nájsť Turingov stroj, ktorý spracuje reťazec w, vždy zastaví a zistí, či popisuje TS M a vstup x, na ktorý zastaví?

- Teoreticky by sme vedeli spustiť simuláciu TS M so vstupom x. Ak M zastaví na x, tak aj naša simulácia zastaví!
- Ak by však M na x nezastavilo, tak ani naša simulácia nezastaví.

L_2

Tým sme si vlastne naznačili, že tento jazyk je určite minimálne **rekurzívne vyčísliteľný** - teda že vieme nájsť taký TS, ktorý by v konečnom čase vrátil ÁNO pre taký reťazec w reprezentujúci pár (M, x), kde M zastaví na x. Na obrázku sa



volá M_h.

L_2

- Tento TS M_h vezme vstup w, dekóduje z neho Turingov stroj M so vstupom x a vloží ich do univerzálneho TS U, t.j. de facto **simuluje činnosť** M na vstup x.
- Ak M zastaví na x, tak aj simulátor U zastaví. V takom prípade Turingov stroj M_h vráti pre vstupný reťazec w odpoveď ÁNO.
- Ak sa simulátor U nezastaví (pretože M sa na vstup x nezastaví), tak náš Turingov stroj M_h sa rovnako nezastaví a nevráti nič.
- To znamená, že M_h vráti vždy v konečnom čase **len odpoveď ÁNO** - avšak korektne len pre tie inštancie (M, x), ktoré patria do jazyka L_2. Preto $L(M_h) = L_2$ a jazyk L_2 je teda **rekurzívne vyčísliteľný**, pretože existuje TS, ktorý ho akceptuje.

L_2

Podarilo sa nám nájsť taký Turingov stroj, ktorý zastaví s istotou len v prípade, že odpoveď má byť ÁNO.

Existuje Turingov stroj, ktorý by vždy zastavil aj v prípade, že by odpoveď mala byť NIE, t.j. že by M na vstup x nezastavilo? - NIE, takýto Turingov stroj NEEXISTUJE.

Jazyk L_2 **nie je rozpoznateľný (rekurzívny).** Aby sme toto tvrdenie dokázali, urobíme **redukciu** z jazyka L_u, ktorý tiež nie je rozpoznateľný, z ktorej logicky vyplynie, že L_2 nie je rekurzívny (rozpoznateľný).

Redukcia L_u na L_2:

- Musíme nájsť algoritmus, ktorý prevedie každú inštanciu problému L_u na nejakú inštanciu problému L_2 s **rovnakou odpoveďou.**
- Teda, že z dvojice (M, x), kde M akceptuje x, nejakým algoritmom vieme odvodiť dvojicu (M', x') takú, že M' určite zastaví pri spracovaní vstupu x'.
- A naopak, že z dvojice (M, x), kde M neakceptuje x, nejakým algoritmom vieme odvodiť dvojicu (M', x') takú, že M' určite nezastaví pri spracovaní vstupu x'.

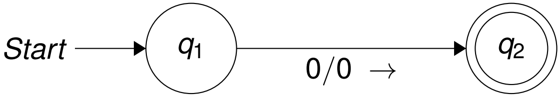
Malá odbočka

Na chvíľu odbočme a uveďme malý trik, ktorý sa pri redukciách môže zísť: Ak máme daný nejaký Turingov stroj M, tak z neho **vždy vieme odvodiť** Turingov stroj M_∞ s rovnakým jazykom, t.j. $L(M) = L(M_\infty)$, pričom M_∞ sa **nikdy nezastaví** pre slová, ktoré neakceptuje.

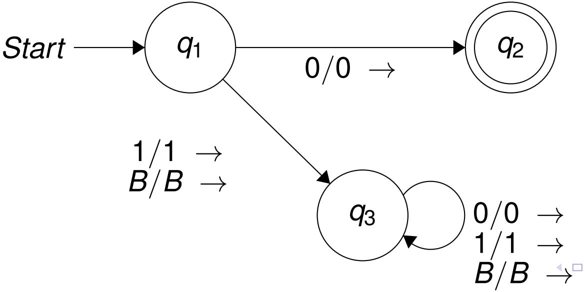
- T.j. existuje jednoduchá úprava, ktorá nezmení akceptovaný jazyk TS, avšak dokáže "vynútiť zacyklenie" pre tie reťazce, pre ktoré by TS M zastavil v neakceptačnom stave.
- V jednoduchosti by sme mohli napríklad vyrobiť nový neakceptačný "cykliaci" stav, v ktorom pridáme slučky na všetky páskové symboly (posun hlavy je irelevantný)
- Rovnako do TS pridáme zo všetkých neakceptačných stavov prechody do tohto nového "cykliaceho" stavu na tie symboly, na ktoré z nich neboli ešte definované prechody.

Malá odbočka

T.j. z TS M, ktorý akceptuje $L = 0^*$ ale môže zastaviť pre reťazce, ktoré neakceptuje, napr $\varepsilon, 1, 101, \ldots$



vyrobíme TS M_∞, ktorý znovu akceptuje $L = 0^*$, ale pre ostatné reťazce nikdy nezastaví:



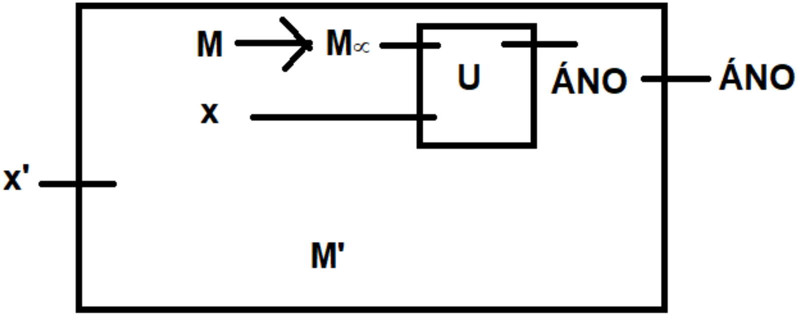
Redukcia L_u na L_2

Tento trik použijeme pri redukcii L_u na L_2:

1. Nech (M, x) je inštancia problému L_u.
2. Uvažujme TS M', so svojím vstupom x', ktorý pracuje nasledovne: Vezme inštanciu (M, x) problému L_u a z M vyrobí TS M_∞, ktorý nikdy nezastaví pre neakceptované reťazce.
3. Následne M' urobí simuláciu spracovania slova x pomocou Turingovho stroja M_∞, t.j. v Univerzálnom Turingovom stroji UTS spustí simuláciu toho, čo by urobil M_∞ pre vstup x.
4. Ak sa UTS stroj zastaví, tak aj M' sa zastaví - a napríklad povie, že svoj vstup x' akceptuje. Zaujímavosťou tohto konkrétneho riešenia je, že vstup x' sa v podstate nikde nepoužíva.
5. Ak sa UTS nezastaví, tak ani M' sa nezastaví.

Redukcia L_u na L_2

Na obrázku je vznik inštancie (M', x') pre problém L_2. TS M' v sebe obsahuje nejakú inštanciu (M, x) problému L_u, pričom najprv prevedie M na M_∞ a následne simuluje činnosť M_∞ na vstupe x v UTS. Ak UTS zastaví s odpoveďou ÁNO, aj M' vráti odpoveď ÁNO.



Redukcia L_u na L_2

Prečo je to správne riešenie?

- Predstavme si, že (M, x) má tú vlastnosť, že $x \in L(M)$. Potom v Turingovom stroji M' dôjde k tomu, že UTS vráti odpoveď ÁNO a celý M' sa tým pádom zastaví - to, že s akou odpoveďou, je momentálne úplne irelevantné - ale nech je to ÁNO.
- Predstavme si, že (M, x) má tú vlastnosť, že $x \notin L(M)$. Potom v Turingovom stroji M' dôjde k tomu, že UTS sa nikdy nezastaví a tým pádom sa nikdy nezastaví ani M'.

Redukcia L_u na L_2

- Ak $(M, x) \in L_u$, potom $(M', x') \in L_2$.
- Ak $(M, x) \notin L_u$, potom $(M', x') \notin L_2$.
- Teda L_u sa redukuje na L_2.
- Tým pádom platí implikácia: Ak L_u je nerozhodnuteľný problém, aj L_2 je nerozhodnuteľný problém.
- A keďže **vieme**, že L_u je nerozhodnuteľný problém, tak je tým pádom **dokázané**, že aj L_2 je nerozhodnuteľný problém.

L_2

Na konci minulého cvičenia sme pre presne tento istý jazyk L_2 (označený ako L_h) dokázali, že nie je rozhodnuteľný trochu iným spôsobom - predpokladali sme, že rozhodnuteľný je a logicky sme dospeli k záveru, že potom je aj L_u rozhodnuteľný, čo je SPOR, keďže vieme, že L_u nie je rozhodnuteľný.

Samozrejme, oba prístupy - aj redukcia, aj takýto dôkaz sporom - sú správne a v podstate ide o to isté, len z dvoch rôznych uhlov pohľadu.

L_3

Jazyk $L_3 = \overline{L_2} = \{w \in \{0,1\}^* \mid w$ kóduje TS M a vstup x na ktorý nezastaví $\}$ predstavuje napríklad reťazec

- $w = 0101000101001100010101010110100100100100011100$

  t.j.TS s prechodmi
  $\delta(q_1, 0) = (q_3, 0, R), \delta(q_3, 0) = (q_1, 0, L), \delta(q_1, 1) = (q_2, 1, R)$ a vstupom 00.

Do jazyka nepatrí napríklad:

- $w = 01010010100110010100010100111 00$

  t.j. TS s prechodmi $\delta(q_1, 0) = (q_2, 0, R), \delta(q_2, 0, q_3, 0, R)$ a vstupom 00

L_3

- Jazyk L_3 je **doplnkom jazyka** L_2.
- Ako už vieme, L_2 je rekurzívne vyčísliteľný jazyk, ktorý nie je rekurzívny.
- Podľa znalostí z prednášky to logicky môže znamenať jediné - jazyk L_3 **nemôže byť** rekurzívne vyčísliteľný.
- Teda nemôže existovať taký Turingov stroj, ktorý by dokázal s istotou povedať, že (M, x) je taký pár, že M je Turingov stroj, ktorý sa nikdy nezastaví pri spracovaní vstupu x.

L_3

- Tento výsledok si môžeme potvrdiť aj tým, že nájdeme redukciu nejakého problému, o ktorom vieme, že **nie je rekurzívne vyčísliteľný** na problém L_3.
- Z prednášky sú také problémy napríklad
  - diagonalizačný jazyk
    $L_d = \{w \mid w$ je kódovanie TS M , ktorý neakceptuje sám seba$w \notin L(M)\}$
  - jazyk prázdnych jazykov $L_e = \{w \mid w$ je kódovanie TS $M, L(M) = \emptyset\}$

Redukcia L_d na L_3

Ukážeme si redukciu problému L_d na problém L_3. T.j. potrebujeme nájsť spôsob, ako:

- Každú inštanciu problému L_d s odpoveďou áno previesť na inštanciu problému L_3 s odpoveďou áno.
- Každú inštanciu problému L_d s odpoveďou nie previesť na inštanciu problému L_3 s odpoveďou nie.

Redukcia L_d na L_3

Je dobré si uvedomiť:

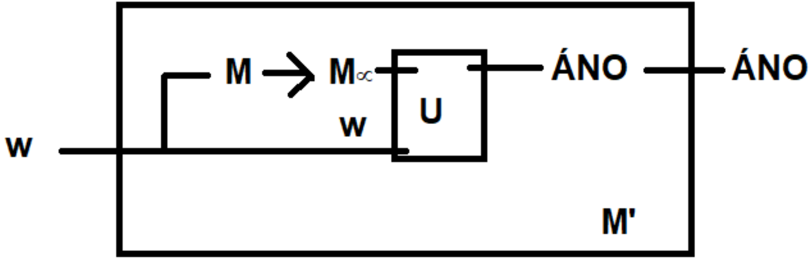
- Inštancia problému L_d s odpoveďou áno = reťazec w, ktorý kóduje taký TS M, že $w \notin L(M)$, teda w kóduje Turingov stroj, ktorý neakceptuje sám seba ako binárny reťazec.
- Inštancia problému L_3 s odpoveďou áno = pár (M', x') taký, že M' kóduje TS, ktorý sa pri spracovaní vstupu x' nezastaví
- Inštancia problému L_d s odpoveďou nie = reťazec w, ktorý kóduje taký TS M, že $w \in L(M)$, teda w kóduje Turingov stroj, ktorý akceptuje sám seba ako binárny reťazec.
- Inštancia problému L_3 s odpoveďou nie = pár (M', x') taký, že M' kóduje TS, ktorý sa pri spracovaní vstupu x' zastaví.

Redukcia L_d na L_3

1. Nech M' je Turingov stroj, ktorého vstupom bude reťazec w.
2. Turingov stroj M' dekóduje reťazec w na nejaký Turingov stroj M, z ktorého vyrobí Turingov stroj M_∞, ktorý v prípade, že vstup neakceptuje, nikdy nezastaví (viď trik z minulého príkladu)
3. TS M' následne simuluje činnosť M_∞ na reťazci w pomocou Univerzálneho Turingovho stroja UTS U. Ak sa UTS U zastaví, aj M' zastaví svoju činnosť (a napríklad vstup w akceptuje).
4. Ak sa U nezastaví, tak ani M' nezastaví.

Redukcia L_d na L_3

Turingov stroj M' je znázornený na obrázku:



Redukcia L_d na L_3

Uvedený postup je algoritmizovateľný. Navyše platí:

- Ak $w \in L_d$, t.j. w je inštancia problému L_d s odpoveďou ÁNO, teda w predstavuje Turingov stroj, ktorý neakceptuje sám seba, tak potom keď sa v univerzálnom stroji simuluje činnosť M_∞ na w, w nie je akceptované M_∞ a UTS nikdy neskončí. Teda ani M' so vstupom w nikdy neskončí a teda predstavuje inštanciu problému L_3 s odpoveďou ÁNO: $(M', w) \in L_3$.
- Ak $w \notin L_d$, t.j. w je inštancia problému L_d s odpoveďou NIE, teda w predstavuje Turingov stroj, ktorý akceptuje sám seba, tak potom keď sa v univerzálnom stroji simuluje činnosť M_∞ na w, w je akceptované M_∞ a UTS určite skončí. Teda aj M' so vstupom w určite skončí a teda predstavuje inštanciu problému L_3 s odpoveďou NIE: $(M', w) \notin L_3$.

Redukcia L_d na L_3

- Našli sme redukciu L_d na L_3.
- A keďže L_d je jazyk, ktorý **nie je ani rozhodnuteľný, ani rekurzívne vyčísliteľný**,
- tak potom jazyk L_3 nie je **ani rozhodnuteľný, ani rekurzívne vyčísliteľný**.

L_4

Jazyk $L_4 = \{L \mid L \text{ je RE jazyk a } |L| \geq 3\}$. Tento jazyk tvoria také RE jazyky, ktoré obsahujú aspoň 3 reťazce.

L_4

- Jazyk L_4 je príkladom jazyka, ktorý **predstavuje vlastnosť rekurzívne vyčísliteľných jazykov**, keďže príslušnosť do L_4 priamo závisí od nejakej vlastnosti tohto jazyka - v tomto prípade koľko reťazcov obsahuje.
- Preto môžeme použiť Riceovu vetu - avšak v prípade, že daná vlastnosť **nie je triviálna**.
- Zistíme to tak, že sa nám podarí ukázať, že:
  - **existuje RE jazyk**, ktorý vlastnosť spĺňa - takým je napríklad $L = a^n b^n c^n$, ktorý obsahuje nekonečne veľa reťazcov, nekonečno je určite viac alebo rovné ako 3
  - **existuje RE jazyk**, ktorý vlastnosť nespĺňa - takým je napríklad $L = \emptyset$, ktorý obsahuje 0 reťazcov, 0 je menej ako 3
- Vlastnosť L_4 je teda **netriviálna** - môžeme použiť Riceovu vetu!

L_4

- Riceova veta nám v tomto prípade jednoducho hovorí, že jazyk L_4 musí byť nerozhodnuteľný!
- Teda o tom, že jazyk L_4 nie je rekurzívny už nemusíme rozmýšľať - priamo to vyplýva z Riceovej vety.
- Druhou otázkou však zostáva:
  - Je / nie je jazyk L_4 rekurzívne vyčísliteľný?
  - Ak je - musíme nájsť TS, ktorý túto vlastnosť vie v prípade kladnej odpovede zistiť v konečnom čase.
  - Ak nie je - musíme naň zredukovať nejaký iný problém, ktorý nie je RE (napríklad L_d)

L_4

- Vieme nájsť Turingov stroj, ktorý by na vstup dostal nejaký rekurzívne vyčísliteľný jazyk v konečnej forme - napríklad vo forme Turingovho stroja M, ktorý ho akceptuje - a vedel by v konečnom čase vrátiť odpoveď áno, ak zadaný jazyk obsahuje aspoň 3 reťazce?
- Ak máte teda zadaný nejaký Turingov stroj, ako by ste zisťovali, či akceptuje aspoň 3 reťazce?
- Vieme, že stačí ak nejaké 3 nájdeme a máme vystarané!

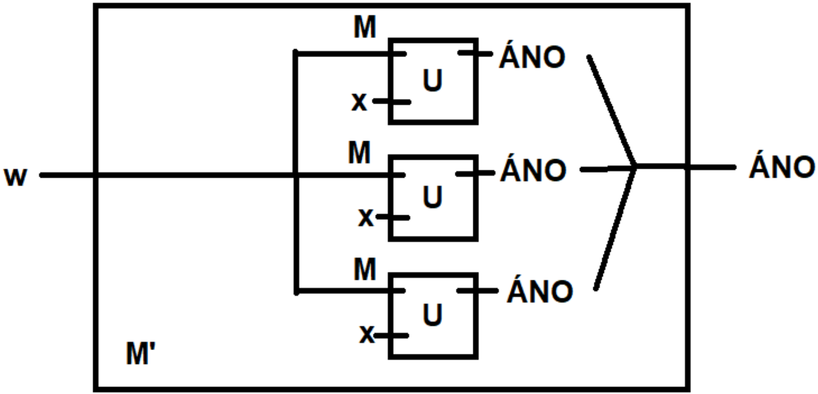
L_4

Uvažujme nasledovný Turingov stroj M':

- Na vstup dostane reťazec w, ktorý kóduje nejaký Turingov stroj M, ktorý akceptuje nejaký RE jazyk.
- Reťazec w dekóduje na príslušný Turingov stroj M.
- Následne bude **nedeterministicky** generovať reťazce a paralelne testovať, či ich M akceptuje alebo nie.
- Keďže celý proces prebieha **nedeterministicky**, môžeme sa spoľahnúť, že **ak** M akceptuje aspoň 3 reťazce, tak tieto 3 reťazce budú nedeterministicky vygenerované a keď ich M akceptuje, tak činnosť M' ukončíme a vrátime správnu odpoveď ÁNO.
- V prípade, že by M neakceptoval aspoň 3 reťazce, tak uvedený proces nikdy neskončí.

L_4

Činnosť M' je uvedená na obrázku:



L_4

Vidíme, že Turingov stroj M':

- Vezme vstup w, dekóduje ho na 3 kópie Turingovho stroja M.
- Paralelne 3-krát generuje nejaký reťazec x
- Paralelne 3-krát simuluje TS M na reťazcoch x (viď 3 univerzálne TS U na obrázku)
- Ak sa podarí nájsť 3 také reťazce x, že ich M bude akceptovať, potom M spĺňa vlastnosť L_4
- A teda hlavný Turingov stroj M' bude akceptovať vstup w, pretože vstup w práve **kóduje taký Turingov stroj**, ktorý predstavuje jazyk spĺňajúci vlastnosť L_4.

L_4

Jazyk L_4 je teda:

- Rekurzívne vyčísliteľný

avšak nie je:

- rekurzívny (rozhodnuteľný).

Logicky teda vieme predvídať, že jeho doplnok, jazyk $\overline{L_4} = \{L \mid L \text{ je RE jazyk a } |L| < 3\}$, teda množina RE jazykov, ktoré obsahujú nanajvýš 2 reťazce, **nemôže byť** rekurzívne vyčísliteľná.

L_5

Jazyk $L_5 = \{L \mid L \text{ je RE jazyk a } |L| < 3\}$. Tento jazyk tvoria také RE jazyky, ktoré obsahujú nanajvýš 2 reťazce.

L_5

- Jazyk L_5 je **práve doplnkom** jazyka L_4 z predchádzajúcej úlohy.
- Automaticky by sme teda vedeli, že nemôže byť ani rekurzívny, ani rekurzívne vyčísliteľný.
- Vieme to však dokázať aj inak.

L_5

- Znovu platí, že jazyk L_5 popisuje nejakú vlastnosť RE jazykov, teda má zmysel uvažovať, či nevieme použiť Riceovu vetu, aby sme dokázali, že jazyk nie je rekurzívny.
- Do L_5 patria také jazyky, ktoré obsahujú najviac 2 reťazce. Teda napríklad:
  - Jazyk $L = \emptyset$ obsahujúci 0 reťazcov do L_5 určite patrí.
  - Jazyk $L = a^n b^n c^n$ obsahujúci nekonečne veľa reťazcov do L_5 určite nepatrí.
- Vidíme, že teda existujú RE jazyky, ktoré do L_5 patria. Rovnako existujú aj jazyky, ktoré do L_5 nepatria. Preto L_5 predstavuje **netriviálnu vlastnosť RE jazykov**, a teda podľa Riceovej vety je L_5 **nerozhodnuteľný**, resp. **nerekurzívny**.

L_5

- Ďalším krokom bude dokázať, že L_5 nie je ani rekurzívne vyčísliteľný.
- Na to potrebujeme ukázať, že sa na L_5 redukuje nejaký problém, ktorý nie je rekurzívne vyčísliteľný.
- Znovu nám ako vhodný problém môže poslúžiť L_d a trik, kde z nejakého Turingoveho stroja M vyrobíme ekvivalentný Turingov stroj M_∞, ktorý nikdy nezastaví pre tie vstupy, ktoré neakceptuje.

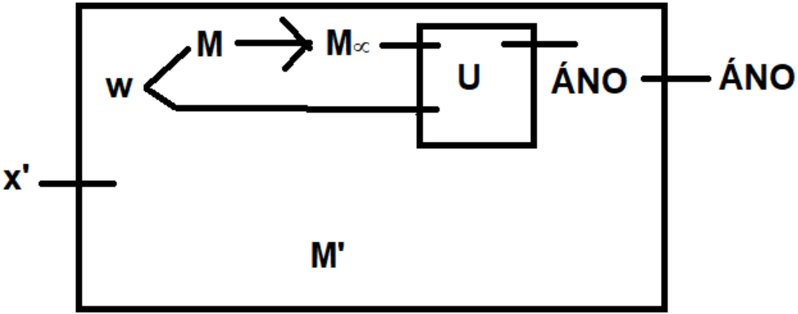
Redukcia L_d na L_5

Nech w je inštancia problému L_d. Zostrojme nasledovný Turingov stroj M':

1. Jeho vstupom bude nejaká postupnosť x'
2. Turingov stroj vezme inštanciu w problému L_d a z w odvodí v ňom zakódovaný TS M, z neho odvodí M_∞, ktorý nikdy neskončí pre vstupy, ktoré neakceptuje a následne pomocou UTS U začne simuláciu M_∞ na reťazci w.
3. Ak simulácia skončí (len v prípade, že $w \in L(M)$), potom Turingov stroj akceptuje svoj vstup x', bez ohľadu na to, čo ním je.
4. Ak simulácia neskončí, tak neskončí ani M', a teda vstup neakceptuje, bez ohľadu na to, čo ním je.

Redukcia L_d na L_5

Obrázok činnosti M' vzhľadom na svoj vstup x' a na inštanciu w problému L_d.



Redukcia L_d na L_5

O správnosti redukcie sa musíme presvedčiť:

- Ak $w \in L_d$, tak w kóduje TS M, ktorý neakceptuje sám seba. Teda $w \notin L(M), w \notin L(M_\infty)$, U sa teda určite nezastaví a celý Turingov stroj M' sa nezastaví - **bez ohľadu na svoj vstup** x'. Teda žiaden vstup x' akceptovať nebude, a teda v tomto prípade $L(M') = \emptyset$, čiže $|L(M')| < 3$.
- Ak $w \notin L_d$, tak w kóduje TS M, ktorý akceptuje sám seba. Teda $w \in L(M), w \in L(M_\infty)$, U sa teda určite zastaví a celý Turingov stroj M' akceptuje svoj vstup x'. Teda bude akceptovať **každý vstup** x' a v tomto prípade $|L(M')| = \infty \geq 3$.

Redukcia L_d na L_5

Teda existuje algoritmus, ktorý vezme inštanciu w problému L_d a na jej základe vyrobí inštanciu problému L_5 s rovnakou odpoveďou.

Takže problém L_d sa redukuje na problém L_5 a teda L_5 predstavuje jazyk, ktorý **nie je rekurzívne vyčísliteľný**.

L_6

Jazyk $L_6 = \{L \mid L$ je RE jazyk a L je konečný $\}$. Tento jazyk tvoria také RE jazyky, ktoré obsahujú konečný počet reťazcov.

L_6

- Jazyk L_6 popisuje nejakú vlastnosť RE jazykov.
- Existuje RE jazyk, napríklad $L = \emptyset$, ktorý patrí do L_6, teda je konečný.
- Existuje RE jazyk, napríklad $L = a^n b^n c^n$, ktorý konečný nie je, a teda do L_6 nepatrí.
- Môžeme teda uplatniť Riceovu vetu a teda jazyk L_6 určite nie je rozhodnuteľný, resp. rekurzívny.

L_6

- Na zistenie toho, či je jazyk L_6 aspoň rekurzívne vyčísliteľný, alebo nie, sa skúsme zamyslieť, ako by sme zisťovali, či nejaký jazyk popisuje konečný počet reťazcov.
- Respektíve, keďže každý RE jazyk je popísateľný nejakým TS, či príslušný TS akceptuje konečný počet reťazcov.
- Najväčší problém je s tým, ako ohraničiť konečný počet reťazcov? Museli by sme vedieť stanoviť nejakú číselnú hranicu a ukázať, že do jazyka už väčší počet reťazcov nepatrí. To by mohlo byť dosť náročné, až nemožné...
- Pokúsme sa teda dokázať, že L_6 **nie je rekurzívne vyčísliteľný**

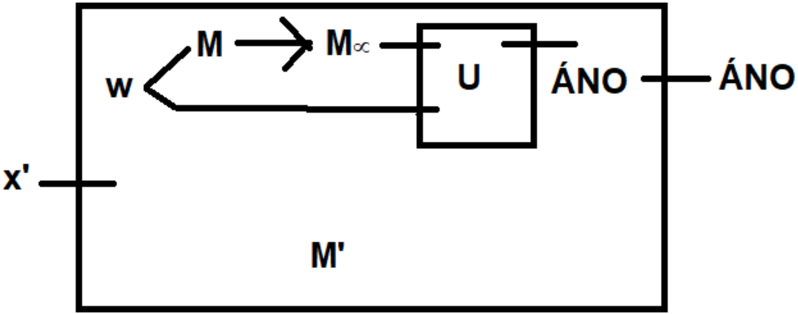
Redukcia L_d na L_6

V podstate tu vznikne identická situácia, ako pri jazyku L_5. Nech w je inštancia problému L_d. Zostrojme nasledovný Turingov stroj M':

1. Jeho vstupom bude nejaká postupnosť x'
2. Turingov stroj vezme inštanciu w problému L_d a z w odvodí v ňom zakódovaný TS M, z neho odvodí M_∞, ktorý nikdy neskončí pre vstupy, ktoré neakceptuje a následne pomocou UTS U začne simuláciu M_∞ na reťazci w.
3. Ak simulácia neskončí, teda $w \notin L(M)$, tak neskončí ani M', a teda vstup neakceptuje, bez ohľadu na to, čo ním je.
4. Ak simulácia skončí (teda $w \in L(M)$), potom Turingov stroj akceptuje svoj vstup x', bez ohľadu na to, čo ním je.

Redukcia L_d na L_6

Obrázok činnosti M' vzhľadom na svoj vstup x' a na inštanciu w problému L_d.



Redukcia L_d na L_6

O správnosti redukcie sa musíme presvedčiť:

- Ak $w \in L_d$, tak w kóduje TS M, ktorý neakceptuje sám seba. Teda $w \notin L(M), w \notin L(M_\infty)$, U sa teda určite nezastaví a celý Turingov stroj M' sa nezastaví - **bez ohľadu na svoj vstup** x'. Teda žiaden vstup x' akceptovať nebude, a teda v tomto prípade $L(M') = \emptyset$, čiže $L(M)$ je **konečný.**
- Ak $w \notin L_d$, tak w kóduje TS M, ktorý akceptuje sám seba. Teda $w \in L(M), w \in L(M_\infty)$, U sa teda určite zastaví a celý Turingov stroj M' akceptuje svoj vstup x'. Teda bude akceptovať **každý vstup** x' a v tomto prípade $L(M)$ je nekonečný, lebo akceptuje neobmedzený počet reťazcov.

Redukcia L_d na L_6

Teda existuje algoritmus, ktorý vezme inštanciu w problému L_d a na jej základe vyrobí inštanciu problému L_6 s rovnakou odpoveďou.

Takže problém L_d sa redukuje na problém L_6 a teda L_6 predstavuje jazyk, ktorý **nie je rekurzívne vyčísliteľný**.

L_7

Jazyk $L_7 = \{L \mid L$ je RE jazyk a L je nekonečný $\}$. Tento jazyk tvoria také RE jazyky, ktoré obsahujú nekonečný počet reťazcov.

L_7

- Keď sa nad tým zamyslíme, tak jazyk L_7 je **doplnkom** jazyka L_6.
- Keďže vieme, že L_6 nie je RE jazyk, tak pre L_7 sú možné 2 možnosti:
  - L_7 tiež nie je RE jazyk
  - L_7 je síce RE jazyk, ale nie je rekurzívny
  - Zatiaľ teda s istotou vieme len to, že L_7 nie je rozhodnuteľný (rekurzívny). Túto vlastnosť si vieme rovnako potvrdiť aj Riceovou vetou.

L_7

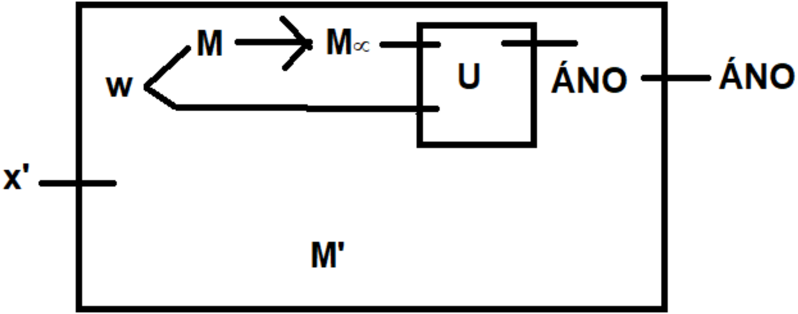
- Jazyk L_7 popisuje nejakú vlastnosť RE jazykov.
- Existuje RE jazyk, napríklad $L = a^n b^n c^n$, ktorý patrí do L_7, keďže je nekonečný.
- Existuje RE jazyk, napríklad $L = \emptyset$, ktorý je konečný, a teda do L_7 nepatrí.
- Môžeme teda uplatniť Riceovu vetu a teda jazyk L_7 určite nie je rozhodnuteľný, resp. rekurzívny.

L_7

- Na zistenie toho, či je jazyk L_7 aspoň rekurzívne vyčísliteľný, alebo nie, sa skúsme zamyslieť, ako by sme zisťovali, či nejaký jazyk popisuje nekonečný počet reťazcov.
- Respektíve, keďže každý RE jazyk je popísateľný nejakým TS, či príslušný TS akceptuje nekonečný počet reťazcov.
- Podobne, ako pri L_6 - ako vieme zistiť, či nejaký Turingov stroj akceptuje nekonečne veľa reťazcov? Počítať ich nemôžeme!
- Pokúsme sa teda dokázať, že ani L_7 **nie je rekurzívne vyčísliteľný**

Redukcia L_d na L_7

Táto redukcia je **podstatne náročnejšia**, než v predchádzajúcom príklade. **Nemôžeme totižto zrecyklovať riešenie z minulej úlohy a redukovať L_d na L_7 nasledovným spôsobom:**



Redukcia L_d na L_7

Zdôvodnenie:

1. Keby sme zobrali inštanciu $w \in L_d$, tak uvedený TS M' v danom prípade nič neakceptuje. To však **nie je dobre**, pretože pre inštanciu $w \in L_d$ (t.j. s odpoveďou ÁNO), potrebujeme dostať inštanciu L_7 s odpoveďou ÁNO - to v žiadnom prípade nie je prázdny jazyk!
2. Rovnako naopak, ak by sme zobrali inštanciu $w \notin L_d$, tak uvedený TS M' v danom prípade akceptoval všetko. To však **tiež nie je dobre**, pretože pre inštanciu $w \notin L_d$ (t.j. s odpoveďou NIE), potrebujeme dostať inštanciu L_7 s odpoveďou NIE - teda konečný jazyk - čo rozhodne nie je prípad, že TS akceptuje každý reťazec.
3. Preto takáto redukcia **je zlá!**

Redukcia L_d na L_7

Riešenie spočíva v nasledovnej myšlienke:

- Redukcia musí zabezpečiť, aby sme z inštancie pôvodného problému s odpoveďou ÁNO dostali taký RE jazyk (alebo Turingov stroj), ktorý má (akceptuje) určite **nekonečne veľa reťazcov.**
- A naopak, aby sme z inštancie pôvodného problému s odpoveďou NIE dostali taký RE jazyk (alebo Turingov stroj), ktorý má (akceptuje) určite **konečný počet reťazcov.**

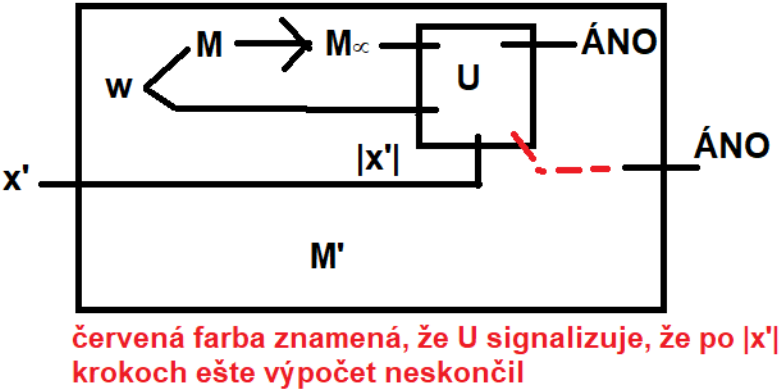
Redukcia L_d na L_7

Uvažujme redukciu z problému L_d, pričom zostrojíme Turingov stroj M':

- Nech vstup TS M' je nejaký reťazec x'.
- Nech w je inštancia problému L_d, ktorú má TS M' k dispozícii.
- TS M' znovu dekóduje w na príslušný TS M, z neho odvodí TS M_∞, ktorý zastaví len pre tie reťazce, ktoré akceptuje a pomocou UTS U simuluje činnosť M_∞ na reťazci w.
- Avšak tentokrát je simulácia **riadená** tak, aby sa z nej vykonalo len toľko krokov výpočtu, koľko má vstup x' znakov, t.j. $|x'|$.
- Turingov stroj M' bude teraz akceptovať **len také reťazce** x', pre ktoré platí, že simulácia M na vstupe w, ktorú vykonáva U, sa po vykonaní $|x'|$ krokov výpočtu ešte **nezastavila**!

Redukcia L_d na L_7

Situáciu sa pokúša priblížiť obrázok:



Redukcia L_d na L_7

O správnosti redukcie sa musíme presvedčiť:

- Ak $w \in L_d$, tak w kóduje TS M, ktorý neakceptuje sám seba. Teda $w \notin L(M), w \notin L(M_\infty)$, U sa teda určite nezastaví. Pre všetky možné reťazce na vstupe x' teda platí, že keď vezmeme počet ich znakov $|x'|$ a spustíme simuláciu U, tak po $|x'|$ bude signalizované, že ešte neskončila (lebo M_∞ pre w nikdy nezastaví). Preto bez ohľadu na to, akú veľkosť má vstup x', bude akceptovaný. Turingov stroj M' teda akceptuje **nekonečný jazyk** a $L(M') \in L_7$.
- Ak $w \notin L_d$, tak w kóduje TS M, ktorý akceptuje sám seba. Teda $w \in L(M), w \in L(M_\infty)$, U sa teda určite zastaví **po nejakom počte krokov**. Tento počet krokov je **konečné číslo** a Turingov stroj M' bude akceptovať **len tie reťazce** x', ktorých dĺžka $|x'|$ je **kratšia**. To však znamená, že Turingov stroj M' akceptuje len **konečnú množinu reťazcov**, a teda a $L(M') \notin L_7$.

Redukcia L_d na L_7

Teda existuje algoritmus, ktorý vezme inštanciu w problému L_d a na jej základe vyrobí inštanciu problému L_7 s rovnakou odpoveďou.

Takže problém L_d sa redukuje na problém L_7 a teda L_7 predstavuje jazyk, ktorý **nie je rekurzívne vyčísliteľný**.

Našli sme teda zaujímavý pár problémov L_6 a L_7, ktoré sú vzájomne k sebe doplnkové, a **ani jeden** nie je RE jazyk.

L_8

Jazyk $L_8 = \{w \mid w \text{ kóduje Turingov stroj, ktorý nezastaví pre žiaden vstup}\}$.

L_8

- Keďže tvrdenie L_8 nie je tvrdením o vlastnostiach RE jazykov, nie je možné použiť Riceovu vetu.
- Preto, aby sme zistili, do akej kategórie patrí L_8, musíme alebo nájsť príslušný TS, alebo nájsť príslušnú redukciu.

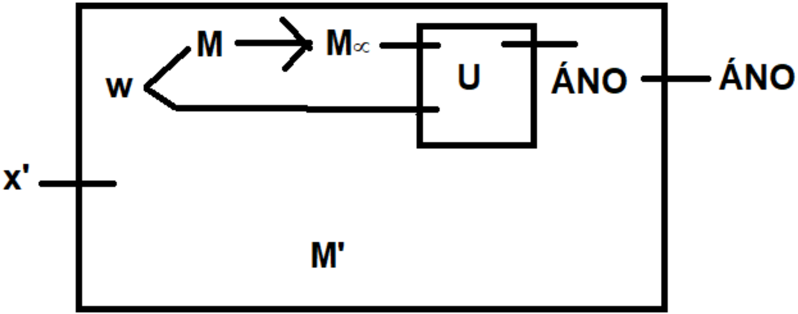
Redukcia L_d na L_8

Dá sa ukázať, že jazyk L_8 nie je rekurzívne vyčísliteľný, redukciou z jazyka L_d. Nech w je inštancia problému L_d. Zostrojme nasledovný Turingov stroj M':

1. Jeho vstupom bude nejaká postupnosť x'
2. Turingov stroj vezme inštanciu w problému L_d a z w odvodí v ňom zakódovaný TS M, z neho odvodí M_∞, ktorý nikdy neskončí pre vstupy, ktoré neakceptuje a následne pomocou UTS U začne simuláciu M_∞ na reťazci w.
3. Ak simulácia neskončí (teda $w \notin L(M)$), tak neskončí ani M', **bez ohľadu na to, aký reťazec** x' má na vstupe.
4. Ak simulácia skončí (teda $w \in L(M)$), potom Turingov stroj akceptuje svoj vstup x', bez ohľadu na to, čo ním je, a zastaví.

Redukcia L_d na L_8

Obrázok činnosti M' vzhľadom na svoj vstup x' a na inštanciu w problému L_d.



Redukcia L_d na L_8

O správnosti redukcie sa musíme presvedčiť:

- Ak $w \in L_d$, tak w kóduje TS M, ktorý neakceptuje sám seba. Teda $w \notin L(M), w \notin L(M_\infty)$, U sa teda určite nezastaví a celý Turingov stroj M' sa nezastaví - **bez ohľadu na svoj vstup** x'. Teda TS M' sa **nikdy nezastaví** bez ohľadu na vstup, čiže $M' \in L_8$.
- Ak $w \notin L_d$, tak w kóduje TS M, ktorý akceptuje sám seba. Teda $w \in L(M), w \in L(M_\infty)$, U sa teda určite zastaví a celý Turingov stroj M' akceptuje svoj vstup x'. Teda Turingov stroj sa zastaví vždy, pre **každý možný vstup** x' (hoci samotný vstup x' na jeho činnosť nemal žiaden vplyv). Nie je teda pravda, že by sa M' zastavil na všetky vstupy a $M' \notin L_8$.

Redukcia L_d na L_8

Teda existuje algoritmus, ktorý vezme inštanciu w problému L_d a na jej základe vyrobí inštanciu problému L_8 s rovnakou odpoveďou.

Takže problém L_d sa redukuje na problém L_8 a teda L_8 predstavuje jazyk, ktorý **nie je rekurzívne vyčísliteľný**.

L_9

Jazyk $L_9 = \{w \mid w \text{ kóduje Turingov stroj, ktorý zastaví pre aspoň jeden vstup}\}$.

L_9

- Keďže tvrdenie L_8 nie je tvrdením o vlastnostiach RE jazykov, nie je možné použiť Riceovu vetu.
- Preto, aby sme zistili, do akej kategórie patrí L_8, musíme alebo nájsť príslušný TS, alebo nájsť príslušnú redukciu.

L_9

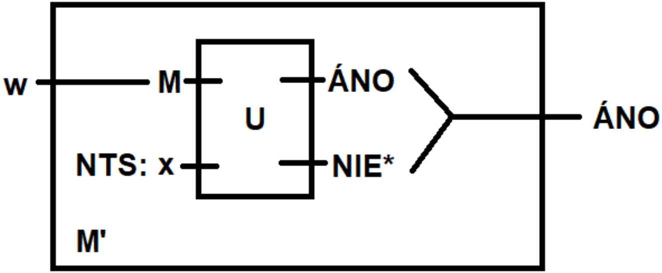
- Vieme nájsť Turingov stroj M', ktorý by na vstup dostal nejaký Turingov stroj v zakódovanej podobe ako reťazec w a zistí, či TS M zastaví pre aspoň jeden vstup.
- V podstate nám stačí nájsť aspoň 1 reťazec x pre ktorý Turingov stroj M zastaví.
- Ak taký reťazec existuje, budeme ho hľadať nedeterministicky.

L_9

Uvažujme nasledovný Turingov stroj M':

- Na vstup dostane reťazec w, ktorý kóduje nejaký Turingov stroj M, ktorý akceptuje nejaký RE jazyk.
- Reťazec w dekóduje na príslušný Turingov stroj M.
- Následne bude **nedeterministicky** generovať reťazce x a testovať, či sa na ne M zastaví alebo nie, pomocou univerzálneho Turingovho stroja U. Ak sa totižto U pri simulovaní M na vstup x zastaví - či už s odpoveďou ÁNO, alebo NIE - tak to znamená, že vstupný reťazec w naozaj reprezentuje taký Turingov stroj, ktorý vie zastaviť pre nejaký vstup, a teda $w \in L_9$
- V prípade, že taký reťazec x neexistuje, tak navrhnutý postup **nikdy neskončí.**
- Teda náš M' patrí do kategórie TS, ktoré s istotou skončia **len v prípade**, že odpoveď je ÁNO.

Činnosť M' je uvedená na obrázku:



NIE* znamená, že U zastavil s odpoveďou NIE (to však môže a nemusí vždy nastať)

L_9

Vidíme, že Turingov stroj M':

- Vezme vstup w, dekóduje ho na Turingov stroj M.
- Nedeterministicky vygeneruje náhodný reťazec x a spustí simuláciu správania M na vstupe x.
- Ak má vygenerovaný reťazec x tú vlastnosť, že naň M zastaví - či už tak, že ho akceptuje, alebo tak, že ho neakceptuje, avšak zastaví, tak potom reťazec w, resp. M spĺňa vlastnosť L_4
- A teda hlavný Turingov stroj M' bude akceptovať vstup w, pretože vstup w práve **kóduje taký Turingov stroj**, ktorý predstavuje jazyk spĺňajúci vlastnosť L_9.

Možno si kladiete otázku:

- Čo ak sa nedeterministicky vygeneruje taký reťazec x, pre ktorý sa U pri simulácii M zacyklí?
- Je potrebné si uvedomiť, že x generujeme nedeterministicky, t.j. pri praktickom deterministickom Turingovom stroji by dochádzalo k deterministickému prechádzaniu všetkých možností.
- Teda v podstate sa pri generovaní reťazca x výpočet rozdelí na viacero vetiev a každá vetva spúšťa U na potenciálne inom reťazci x.
- Ako sme spomínali, tak deterministické modelovanie prebieha "do šírky", práve kvôli tomu, aby sa zabránilo tomu, aby sa TS zacyklil v jednej vetve.
- Čiže v prípade, že by jedna vetva výpočtu narazila na situáciu, že M zastaví na reťazec x, je možnosť o tom ihneď informovať.

L_9

Jazyk L_9 je teda:

- Rekurzívne vyčísliteľný

avšak nevieme, či náhodou nie je aj

- rekurzívny (rozhodnuteľný).

L_9

- Jednoduchou úvahou vieme, že L_9 **nemôže byť rekurzívny**.
- Stačí si uvedomiť, že jazyk L_9 je **doplnok** jazyka L_8.
- A keďže jazyk L_8 nebol rekurzívny, ani jeho doplnok nemôže byť rekurzívny.
- Alternatívne je možné nájsť redukciu nejakého nerozhodnuteľného problému na L_9.

Použitá literatúra

1 Hopcroft, Motwani, Ullman - Introduction to Automata Theory, Languages and Computations, 3rd Ed.