Fakulta informačních technologií
Vysoké učení technické v Brně

Teoretická informatika (TIN) – 2019/2020
Úkol 3

Jakub Svoboda (xsvobo0z)

20. 12. 2019

1. Příklad číslo 1

sqrt : ℕ → ℕ, sqrt(x) = z takové, že $z^2 \leq x \wedge (z+1)^2 > x$.

Funkci můžeme vyjádřit například takto:

$$sqrt(x,0) = \xi\,()$$

$$sqrt(x,y+1) = plus(\pi_3^3, \quad eq(\xi, \quad monus(mult(plus(\pi_3^3, \sigma\, o\, \xi), \quad plus(\pi_3^3, \quad \sigma\, o\, \xi)), plus(\pi_2^3, \sigma\, o\, \xi)))) \,(x, y, sqrt(x,y))$$

Neboli formálně zapsáno jako:

$$sqrt(x,0) = \xi\,()$$

$$sqrt(x,y+1) = plus\ o\ (\pi_3^3 \times (eq\ o\ (\xi \times (monus\ o\ (mult\ o\ ((plus\ o\ (\pi_3^3 \times (\sigma\ o\ \xi))) \times (plus\ o\ (\pi_3^3 \times (\sigma\ o\ \xi)))) \times (plus\ o\ (\pi_2^3 \times (\sigma\ o\ \xi)))))))) \big(x, y, sqrt(x,y)\big)$$

Pro otestování lze využít následující kód v jazyce python:

```
def monus(a,b):
	if a >= b:
		return a-b
	else:
		return  0

def eq(a,b):
	if a == b:
		return 1
	else:
		return 0

def mult(a,b):
	return a*b

def plus(a,b):
	return a+b

def E():
	return 0

def s(x):
	return x+1

def sqrt(x):
	if x == 0:
		return 0
	else:
		i = x-1
		return plus(sqrt(i), eq(E(), monus( mult(plus(sqrt(i),1), plus(sqrt(i),1)), plus(i,1))))


for x in range(0, 20):
	print(x, sqrt(x))
```

2. Příklad číslo 2

$f(n) = \sqrt{2}n^3$

$g(n) = 10000n^2 + 500n + 211$

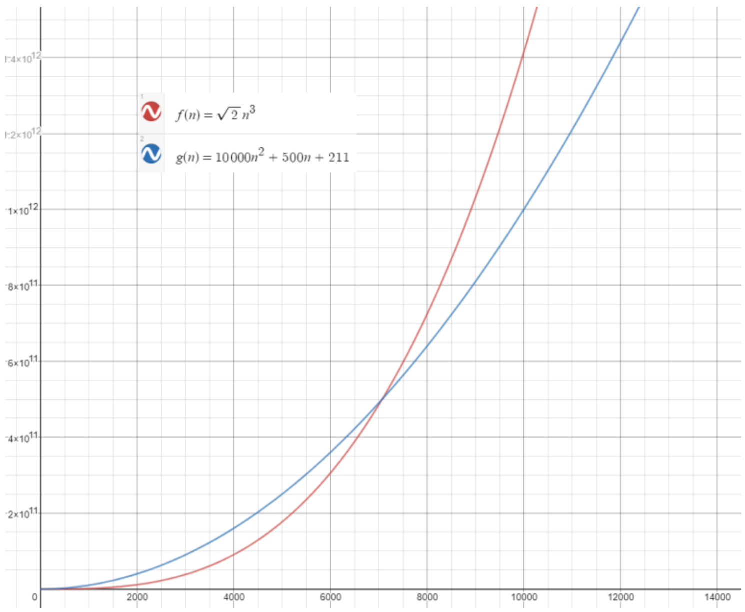
Dokažte, že O(g(n)) ⊂ O(f(n)).

Asymptotické horní omezení funkce f(n) je množina O(f(n)) = {g(n) ∈ F | ∃c ∈ R^+, ∃n_0 ∈ ℕ ∀n ∈ ℕ : n ≥ n_0 ⇒ 0 ≤ g(n) ≤ c*f(n)} (definice ze skript TIN).

Provedeme výpočet rovnic:

$$\sqrt{2}n^3 = 10000n^2 + 500n + 211$$

Rovnost platí pro n ≅ 7071.1. Jelikož je však obor hodnot ℕ, můžeme uvažovat, že *f(n) > g(n)* pro n ≥ 7072 (n_0 = 7072).



1) Dokažte, že O(g(n)) ⊆ O(f(n)).
   Z grafu obou funkcí je patrné, že od n ≥ 7072 je g(n) < f(n) (a obě funkce se protínají pouze v jednom bodě). Obě funkce mají stejný definiční obor (ℕ) a můžeme tedy tvrdit, že vztah O(g(n)) ⊆ O(f(n)) je platný.
   Alternativně můžeme toto ověřit pomocí výpočtu limity. Pro $\forall n > n_0$: $\frac{1000n^2+500n+211}{\sqrt{2}n^3} \leq 1$:

$$\lim_{n\to\infty} \frac{1000n^2 + 500n + 211}{\sqrt{2}n^3}$$

   využijeme L'Hospitalova pravidla:

$$\lim_{n\to\infty} \frac{2000n + 500}{\sqrt{2} * 2n^2} = \lim_{n\to\infty} \frac{2000}{\sqrt{2} * 4n} = 0$$

2) Dokažte, že O(g(n)) ≠ O(f(n)).

   Funkce f(n) a g(n) jsou si rovny pouze v n=7072 (jediný kořen). Navíc je zřejmé, že od n>7072 je f(n) > g(n) a tudíž O(g(n)) ≠ O(f(n)).

Jelikož bylo dokázáno O(g(n)) ⊆ O(f(n)) a O(g(n)) ≠ O(f(n)), je pak patrné, že vztah $O(g(n)) \subset O(f(n))$ je platný.

3. Příklad číslo 3

Zadaný problém lze charakterizovat jako jazyk K, kde K obsahuje řetězce, kde $\sum_{i=0}^{n} c_i * s_i + B * s_i \geq C \wedge \sum_{i=0}^{n} o_i * s_i \leq O \; pro \; s_i \in \{0,1\}$, kde c_i značí obsah vitamínu C v daném druhu zeleniny a o_i značí cenu za kilo daného druhu zeleniny, B je obsah vitamínu C v kile brokolice, C je množství vitamínu C nutné pro uzdravení a O je obnos v peněžence, n je celkové hmotnost zeleniny v obchodě a s_i udává, zda je daná zelenina nakoupena. V tomto úkole uvažujeme, že teta zeleninu nakupuje po celých kilogramech a každý kilogram zeleniny je nová položka.

I. **Ukážeme, že K je v NP (ukazujeme horní mez složitosti)**
   Ukážeme, že existuje nedeterministický vícepáskový TS, který přijímá jazyk K v polynomiálním čase. Označme ho M: M může pracovat následovně:
   1. M ověří, zda má na vstupu správně zformovanou instanci problému K (jinak odmítne). M kontroluje správné použití oddělovačů – lze realizovat se složitostí O(n).
   2. M označí náhodně *k* zelenin tak, že přepíše s_i na 1 (O(n)).
   3. M následně prochází jednotlivé zeleniny (O(n)) a pro každou označenou provede vynásobení váhy s počtem vitamínu na kilogram, sumu zapisuje na pomocnou pásku. Ke každému kilogramu pak přičte vitamíny z brokolice (B*0.1) Na pomocné pásce také uchovává celkovou cenu, kdy pro každou označenou zeleninu (O(n)) násobí její váhu cenou za kilogram. Na pomocnou pásku si zapisuje sumu vitamínu C.
   4. TS M přijme, pokud suma vitamínu C nepřekročila konstantu C a celková cena nepřekročila obnos O.

II. **Ukážeme, že K je NP těžká**
   Ukážeme pomocí polynomiální redukce z Subset Sum Problem, což je NP úplný problém. SSP se dá popsat takto: Je dána určitá neprázdná množina obsahující čísla z ℕ. Dále je hledána podmnožina této množiny, kde součet prvků této podmnožiny je roven nějakému číslu X. Formálně lze SSP zapsat takto: $\sum_{i=0}^{n} v_i * y_i = X, pro \; y_i \in \{0,1\}$. Ukážeme, že problém SSP lze redukovat na problém K, a to následovně:
   - $v_i = o_i = (c_i + B)$
   - $X = O = C$

   Tuto jednoduchou konverzi provede úplný DTS se složitostí spadající do O(n^k). Odpovědi na SSP pak budou přesně korespondovat s odpovědí na problém K. Tedy pokud bude nalezena kombinace zelenin vyhovující výše uvedeným podmínkám, pak stejná kombinace bude i řešením problému SSP.

Jelikož jsme dokázali složitost pro horní i spodní mez, je tedy zřejmé, že problém K je NP-complete.

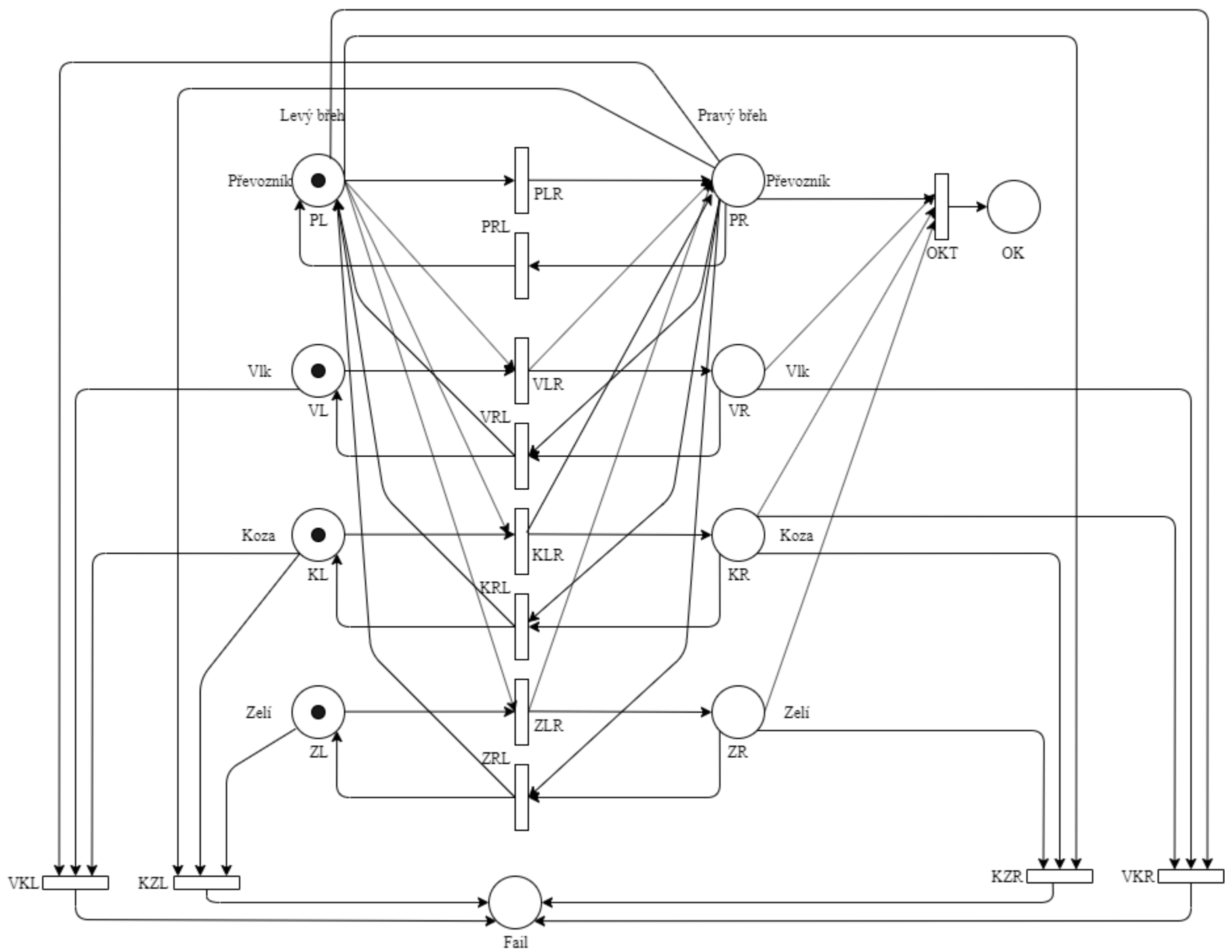
Synovec tety Květy Alan tvrdí, že je schopen tento problém řešit v polynomiálním čase pomocí psacího stroje s nekonečnou páskou. Předpokládáme, že tento jeho stroj je deterministický. Pokud by jeho tvrzení skutečně bylo pravdivé, Alan by tímto dokázal, že NP=P. Pro lidstvo to znamená, že řada problému bude řešitelná v čase nižším, než se dosud předpokládalo. Přestalo by fungovat šifrování atd. Pro Alana samotného by to znamenalo, že by zcela jistě dostal cenu Turingovu cenu za významný přínos v oboru informatiky. Dále by dostal jeden milion dolarů za vyřešení jednoho z *Millenium Prize Problems*. Možná by dostal i zápočet z TINu.

4. Příklad číslo 4

Sít modelující tento systém lze popsat jako (obrázek níže):
$N = (P, T, F, W, K, M_0)$

- P = {PL, PR, VL, VR, KL, KR, ZL, ZR, OK, FAIL}
- T = {PLR, PRL, VLR, VRL, KLR, KRL, ZLR, ZRL, OKT, VKL, KZL, KZR, VRK}
- F = {<PL, PLR>, <PLR, PR>, <PR, PRL>, <PRL, PL>, <VL, VLR>, <VLR, VR>, <VR, VRL>, <VRL, VR>, <KL, KLR>, <KLR, KR>, <KR, KRL>, <KRL, KL>, <ZL, ZLR>, <ZLR, ZR>, <ZR, ZRL>, <ZRL, ZL>, <PL, VLR>, <VLR, PR>, <PL, KLR>, <KLR, PR>, <PL, ZLR>, <ZLR, PR>, <PR, VRL>, <VRL, PL>, <PR, KRL>, <KRL, PL>, <PR, ZRL>, <ZRL, PL>, <PR, VKL>, <VL, VKL>, <KL, VKL>, <VKL, FAIL>, <PR, KZL>, <KL, KZL>, <ZL, KZL>, <KZL, FAIL>, <PL, KZR>, <ZR, KZR>, <KR, KZR>, <KZR, FAIL>,<PL, VKR>, <KR, VKR>, <VR, VKR>, <VKR, FAIL>,<PR, OKT>, <VR, OKT>, <KR, OKT>, <ZR, OKT>, <OKT, OK>}
- $\forall f \in F : W(f) = 1$ (váhy všech hran jsou rovny jedné).
- K = {<PL,1>, <PR,1>, <VL,1>, <VR,1>, <KL,1> <KR,1>, <ZL,1>, <ZR,1>, <OK,1>, <FAIL,1>}
- M_0 = {<PL,1>, <PR,0>, <VL,1>, <VR,0>, <KL,1> <KR,0>, <ZL,1>, <ZR,0>, <OK,0>, <FAIL,0>}

Přítomnost tokenu v místě OK značí, že přechod řeky dopadl úspěšně. Přítomnost tokenu v místě FAIL značí, že došlo ke snězení zelí nebo kozy.



Levý břeh
Pravý břeh
Převozník
PL
PLR
PRL
Převozník
PR
OKT
OK
Vlk
VL
VLR
VRL
Vlk
VR
Koza
KL
KLR
KRL
Koza
KR
Zelí
ZL
ZLR
ZRL
Zelí
ZR
VKL
KZL
KZR
VKR
Fail