VSL Demonstrační cvičení 4
ZS 2005

Ing. Zbyněk Křivka

krivka@fit.vutbr.cz

Konzultace: Po 11-12h, místnost C48

Poslední možnost odevzdat

krivka@fit.vutbr.cz

Konzultace: Po 11-12h, místnost C48

Poslední možnost
odevzdat vypracovanou
domácí úlohu
na tento týden !!!

Příklad 4.1 – Zadání

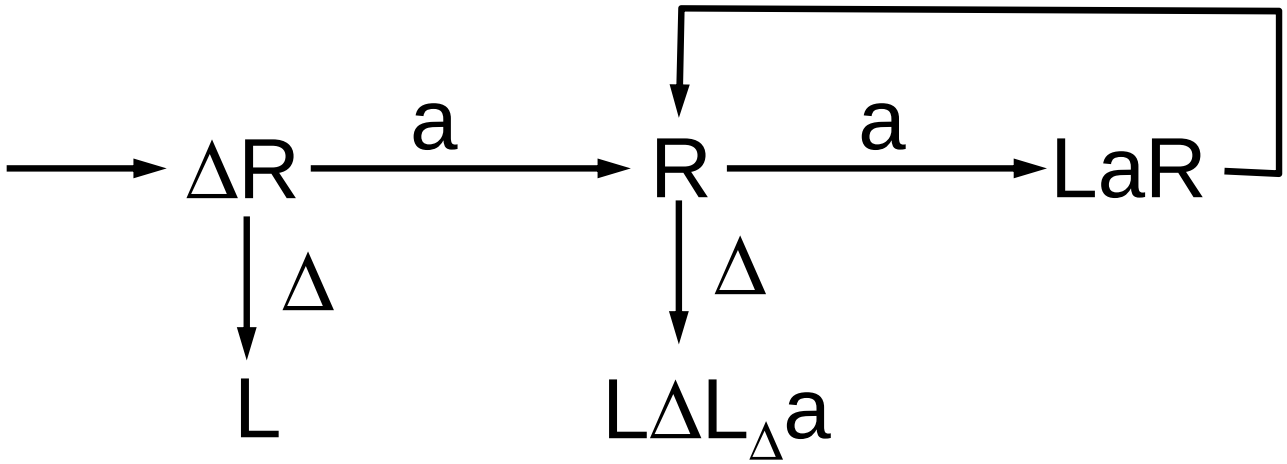
- Určete přesnou časovou i prostorovou složitost modulu $\boldsymbol{S_L}$.
- Uveďte obecný vztah mezi časovou a prostorovou složitostí a zdůvodněte jej.

Označme ***n*** délku řetězce od pozice hlavy (bez samotné pozice) až po nejbližší pravý blank (bez blanku).

Vstupní abeceda je $\Sigma = \{a\}$.

Příklad aplikace S_L: $\underline{a}aaa\Delta \rightarrow \underline{a}a\Delta\Delta$ (n = 2)

Příklad 4.1a – Řešení



- $\underline{\Delta}\Delta$, $n = 0$ … $T(n) = 3$; $M(n) = 1 + n + 1$
- $\underline{\Delta}a\Delta$, $n = 1$ … $T(n) = 2+1+2+n+1$
- $\underline{\Delta}aa\Delta$, $n \geq 2$ … $T(n) = 2 + (n-1)(1+3) +$
  $1+2+n+1 =$ **$5n + 2$ pro $n \geq 1$**

Příklad 4.1b – Řešení

- Paměťová náročnost výpočtu je vždy nižší než časová složitost:
  - Memory(w) ≤ Time(w)+1
- Proč?
  - Minimálně na posun na další pole a kontrolu jeho obsahu potřebujeme 1 časovou jednotku. Pokud navíc provedeme ještě nějaký zápis na pásku nebo opakované čtení některých částí pásky, tak bude Time(w) >> Memory(w).

Příklad 4.2 - Zadání

a) Ukažte, že jazyk L nad abecedou $\Sigma = \{x,y\}$ tvořený řetězci tvaru ww^R, kde $w \in \Sigma^*$ a w^R je řetězec w zapsaný v obráceném pořadí, je přijímán nedeterministickým vícepáskovým Turingovým strojem s časovou složitostí $\Theta(n)$.

b) Ukažte, že časová složitost přijímání jazyka L (viz a)) na deterministickém jednopáskovém Turingově stroji je $\Theta(n^2)$.

c) Patří jazyk L do třídy **P**?

Nebudu konstruovat detailně, pouze rámcově!

L – jazyk palindromů sudé délky

Příklad 4.2a – Řešení

Princip:

1. Páska: vstup (ww^R) např.: $\underline{\Delta}xyxxyx\Delta$
2. Páska: přesun w^R z první pásky na druhou – nedeterministicky např.: $\underline{\Delta}xyx\Delta$

Postup:

1) Inicializace 2. pásky - nedeterministicky přesune w^R z první pásky na druhou … $\Theta(c_1*n) = \Theta(n)$,
2) První pásku čtu od začátku, druhou pásku od konce a porovnávám/kontroluji shodu symbolů i shodu délek w a w^R … $\Theta(c_2*n) = \Theta(n)$.

Příklad 4.2bc – Řešení

Řešení: $|w| = n$

Inspirace: konstrukce kompozitního TS pro S_L nebo S_R.

Využiji 2 zarážky (např. blanky), abych si poznačil, kde se nacházím při čtení přední části (w) a zadní části (w^R).

Pro zapamatování použiji proměnnou ω.

Složitost: Memory(n) = $|w| + |w^R| = 2*n$

Přibližně: Time(n) = $2*(n - 1)*n / 2 + 2*n \approx \Theta(n^2) \Rightarrow L \in \mathbf{P}$

Δ	a_1	a_2	…	a_m	a_m	…	a_2	a_1	Δ

ω

Opakuji pro $m = n$, $m = n$-1, $m = n$-2, …, m = 1.

Obě strany ohraničují blanky.

Příklad 4.3 - Zadání

a) Uveďte příklady algoritmů, které mají při výpočtu na jednopáskovém deterministickém Turingově stroji složitosti $\Theta(n)$, $\Theta(n^2)$ a $\Theta(c^n)$.

b) Uveďte příklady funkcí $g_1(n)$, $g_2(n)$ a $g_3(n)$, aby patřily do množin funkcí následovně:
$g_1(n) \in O(f(n))$, $g_2(n) \in \Theta(f(n))$, $g_3(n) \in \Omega(f(n))$,
kde $f(n) = n^2 + 2n + 3$.

Příklad 4.3a – Řešení

Princip:

Z hlediska mocnosti (tříd přijímaných jazyků) jsou modely det. TS = n-pásk. det. TS = nedet. TS = n-pásk. nedet. TS ekvivalentní.

Při požadavku na zachování třídy složitosti (časové či prostorové) již tyto rovnosti nemusí platit.

Příklady algoritmů TS: $\Theta(n)$, $\Theta(n^2)$ a $\Theta(c^n)$.

1. Kontrola seřazenosti posloupnosti symbolů.
2. Seřazení posloupnosti symbolů metodou BubbleSort nebo porovnání dvou řetězců
3. Rozhodnutí, zda je daná vstupní věta reprezentací splnitelné logické formule.

Příklad 4.3b – Řešení

Příklady $g_1(n) \in O(f(n))$, $g_2(n) \in \Theta(f(n))$, $g_3(n) \in \Omega(f(n))$, kde $f(n) = n^2 + 2n + 3$.

- $g_1(n) = c$
  - $g'_1(n) = 6n + 2$
  - $g''_1(n) = f(n)$ $\in O(f(n))$,
- $g_2(n) = 14n^2 + n - 5$
  - $g'_2(n) = 0{,}01n^2 + 6$
  - $g''_2(n) = f(n)$ $\in \Theta(f(n))$,
- $g_3(n) = n^3$
  - $g'_3(n) = 2^n$
  - $g''_3(n) = f(n)$ $\in \Omega(f(n))$.

Příklad 4.4 – Zadání

a) Ve skeletonové jazyce napište algoritmus pro výpočet mocniny dvou přirozených čísel $\boldsymbol{exp(n, m)} = \boldsymbol{n^m}$, kde $n,m \in \mathbf{N} \cup \{0\}$.

Povolené základní příkazy:

- **`incr var, decr var, while var <> 0 do … end.`**

Pomocné zápisy:

- **`clear var, var1 ← var2.`**

b) Přesná čas. složitost makra **`var1 ← var2`** (operace inkrementace, dekrementace, kontrola na nulu trvá 1 časovou jednotku).

c) Pro a) odhadněte třídu časové a prostorové složitosti.

Příklad 4.4a – Řešení 1/2

Vstup: *x*, *y* Výstup: *z*

x = mocněnec,
y = mocnitel

```
Exp(x, y, z):
  z ← 1;
  while(y<>0) do
      tmpz ← z;
      mult(x, tmpz, z);
      decr y;
  end;
End;
```

```
Mult(x, y, z):
  clear z;
  auxx ← y;
  while(auxx<>0) do
      plus(z, x, z);
      decr auxx;
  end;
End;
```

```
Plus(x, y, z):
  z ← x;
  auxy ← y;
  while(auxy<>0) do
      incr z;
      decr auxy;
  end;
End;
```

Příklad 4.4a – Řešení 2/2

```
Exp(x, y, z):   z ← 1; tmpy ← y;
                while(tmpy<>0) do
                    tmpz ← z;
                    clear z;
                    auxy ← tmpz;
                    while(auxy<>0) do
                          z ← z;
                          auxy2 ← x
                          while(auxy2<>0) do
                                incr z;
                                decr auxy2;
                          end;
                          decr auxy;
                    end;
                    decr tmpy;
                end;
              End;
```

Příklad 4.4b – Řešení

```
name1 ← name2:
  while (aux<>0) do
      decr aux; end;
  while (name1<>0) do
      decr name1; end;

  while (name2<>0) do
      incr aux;
      decr name2;
  end;
  while(aux<>0) do
      incr name1;
      incr name2;
      decr aux;
  end;
End;
```

T(name1, name2, aux) =

+ 2 * aux + 1

+ 2 * name1 + 1

+ 3 * name2 + 1

+ 4 * name2 + 1

= 4 + 2*aux + 2*name1 + 7*name2

Třída časové složitosti je lineární.

SPACE(name1, name2, aux) = 3

Příklad 4.4c – Řešení

```
z ← 1; tmpy ← y;
while(tmpy<>0) do
    tmpz ← z;
    clear z;
    auxy ← tmpz;
    while(auxy<>0) do
          z ← z;
          auxy2 ← x
          while(auxy2<>0) do
                incr z;
                decr auxy2;
          end;
          decr auxy;
    end;
    decr tmpy;
end;
```

```
T(O(f)):
f=c0*(x+c1)^(c3*y+c2)
```

Dotazy?

Děkuji za pozornost!

Nashledanou!