Algoritmus, správnost algoritmu, složitost algoritmu, složitost úlohy, třída P, třída NP

Algoritmus = dobře definovaný proces (posloupnost výpočetních kroků), který zpracuje vstup a vydá výstup

Problém, úloha = obecná specifikace vztahu zadání/řešení
- instancí úlohy U je konkrétní zadání všech parametrů, které daná úloha obsahuje
- algoritmus A řeší úlohu U, jestliže pro každý vstup vydá správné řešení
- alg., který řeší úlohu se vždy zastaví

1. Složitost algoritmu

- viz otázka 01-Fronty a haldy
- f(n), g(n) jsou nezáporné funkce
- pokud $f(n) \in O(g(n))$, pak $g(n) \in \Omega(f(n))$
- pokud $f(n) \in \Theta(g(n))$, pak $f(n) \in O(g(n))$ a zároveň $f(n) \in \Omega(g(n))$

- pro relace O, Θ, Ω platí **tranzitivita**:
  - -jestliže $f(n) \in O(g(n))$ a $g(n) \in O(h(n))$, pak $f(n) \in O(h(n))$
  - -jestliže $f(n) \in \Omega(g(n))$ a $g(n) \in \Omega(h(n))$, pak $f(n) \in \Omega(h(n))$
  - -jestliže $f(n) \in \Theta(g(n))$ a $g(n) \in \Theta(h(n))$, pak $f(n) \in \Theta(h(n))$

- pro relace O, Θ, Ω platí **reflexivita**:
  $f(n) \in O(f(n))$, $f(n) \in \Omega(f(n))$, $f(n) \in \Theta(f(n))$

2. Nezařazené pojmy

- rekurzivní vztahy, $T(n) = aT(\frac{n}{b}) + cn;$, Master Theorem, trojúhelníková nerovnost
-algoritmy bubble sort, Eukleidův. alg., minimální kostra, nejkratší cesty, matice vzdáleností, ohodnocení hran, Floydův algoritmus

3. Správnost algoritmu

1. alg. se musí zastavit
2. alg. po zastavení vydá správný výstup

Variant = přirozené číslo, které se během práce alg. snižuje, až nabude nejmenší možnou hodnotu
- zaručuje ukončení alg. po konečně mnoha krocích
- v bubble sortu je to např. proměnná i ve vnějším for-cyklu

Invariant
= tvrzení, které:
- platí před vykonáním prvního cyklu algoritmu, nebo po prvním vykonání cyklu,
- platí-li před vykonáním cyklu, platí i po jeho vykonání,

- při ukončení práce algoritmu zaručuje správnost řešení.

- pro bubble sort je invariantem tvrzení, že po i-tém proběhnutí vnějšího cyklu jsou a[n−i+1], a[n−i+2], . . . , a[n] největší z čísel a jsou seřazené podle velikosti

Bellmanův princip optimality

- jestliže v grafu G neexistuje cyklus záporné délky, pak pro každé tři vrcholy x, y, z platí
`u(x, y) = min(u(x,z) + a(z, y))`

4. Modely výpočtu

Počítač s libovolným přístupem – RAM se skládá z programové jednotky, aritmetické jednotky, paměti a vstupní a výstupní jednotky

Programová jednotka obsahuje programový registr a vlastní program (programový registr ukazuje na instrukci, která má být provedena)

Aritmetická jednotka provádí aritmetické operace sčítání, odčítání, násobení a celočíselné dělení

Paměť je rozdělena na paměťové buňky, každá buňka může obsahovat celé číslo. Pořadové číslo paměťové buňky.

Vstupní jednotka = vstupní páska a hlava. Vstupní páska je rozdělena na pole (v každém poli může být celé číslo). Hlava snímá v každém okamžiku jedno pole. Po přečtení pole se hlava posune o jedno pole doprava.

Konfigurace počítače = stav vstupní a výstupní pásky, paměťových buněk a registru
-výpočet je posloupnost konfigurací
Příkazy: LOAD, STORE, ADD, SUBTRACT, MULTIPLY, DIVIDE, READ, WRITE, JUMP, STOP, ACCEPT, REJECT

Časová složitost programu pro RAM je počet kroků počítače

Paměťová složitost = max adresa použité buňky

Věta: Ke každému programu P pro RAM, který pracuje v čase O(f(n)), existuje program P′, který má časovou složitost O([f(n)]2) a paměťovou složitost O(f(n))

Turingův stroj

- skládá se z řídící jednotky, nekonečné pásky, čtecí hlavy
= je sedmice **(Q,Σ, Γ, δ, q0, B, F)**, kde **Q** je konečná mn. stavů, **Σ** je konečná mn. vstupních symbolů, **Γ** je konečná mn. páskových symbolů, přitom $\Sigma \subset \Gamma$, **B** je prázdný symbol (blank, není vstupním symbolem), **δ** je přechodová funkce, tj. $Q\times\Gamma \to Q \times \Gamma \times \{L,R\}$
- L znamená pohyb hlavy o jedno pole doleva, R doprava
-**q0** $\in$ Q je počáteční stav a **F** $\subset$ Q je mn. koncových stavů

Zastavení TS
- pokud se TS dostane do jednoho z koncových stavů $q \in F$, stroj se úspěšně zastaví
- výstupem je obsah pásky v okamžiku zastavení

Počáteční situace: Na začátku se TS nachází ve stavu q_0, na pásce má na prvních n polích vstupní slovo $a_1....a_n$ ($a_i \in \Sigma$), ostatní pole obsahují blank B a hlava čte první pole pásky

Krok Turingova stroje
$\delta(q, X_i) = (p, Y, R)$..... stroj se po přijetí symbolu X_i na vstupu přesune ze stavu q do stavu p, na pásku zapíše symbol Y a hlavu posune o jedno pole doprava
- dáno přechodovou funkcí

Jazyk přijímaný TS
- TS může akceptovat slova $w \in \Sigma^*$ tím, že se zastaví
- mn. přijímaných slov se nazývá jazyk L(M)

Neúspěšné zastavení - TS nemá definován následující krok a není v koncovém stavu

Tvrzení: Ke každému Tur. stroji M existuje program P pro RAM takový, že oba mají stejné chování. Jestliže M potřeboval n kroků, P má časovou složitost $O(n^2)$

5. Třídy P a NP

Rozhodovací úlohy

- jejich řešením je buď ano nebo ne, ale často v sobě obsahují konstrukční úlohu (pokud chci zjistit, zda existuje max. tok, musím ho nejdřív najít)
- např. SAT, TSP
- každou úlohu lze **zakódovat** pomocí vhodné abecedy do slov, která buď TS přijímá nebo nepřijímá, instance rozhodovací úlohy můžeme tedy chápat jako slovo nějaké abecedy

Jazyk L_U úlohy U = mn. všech ANO instancí úlohy. Ano instance je instance úlohy, kterou TS přijme.

Třída P

- rozhodovací úloha U leží ve třídě P, jestliže existuje TS, který rozhodne jazyk L_U a pracuje v **polynomiálním čase**
- např.:
  - existuje minimální kostra v grafu s cenou menší než k?
  - existuje nejkratší cesta v DAG s cenou menší než k?

Třída NP

- rozhodovací úloha U leží ve třídě NP, jestliže existuje nedeterministický TS (nebo program P), který rozhodne jazyk L_U a pracuje v **polynomiálním čase**
- nedeterministický algoritmus vyplivne řetězec s na výstup a deterministický alg. dá odpověď ANO nebo NEVIM
- např. problém batohu, barevnost

6. Zdroje

[1] Demlová, Marie: Přednášky 1-5 z TAL.
`http://math.feld.cvut.cz/demlova/teaching/tal`