Základy algoritmizace

Matematické algoritmy (11MAG)

Jan Přikryl

1. přednáška 11MAG
pondělí 5. října 2014

verze: 2014-11-10 10:35

Obsah

O čem si budeme povídat: algoritmy diskrétní matematiky, slasti a strasti výpočtů v plovoucí řádové čárce, numerická matematika.

O čem budou cvičení: praktické hrátky s algoritmy, Matlab/Python/C/C++/Java ...

Co když neumím programovat? To, že jste postoupili až do prvního magisterského ročníku garantuje, že programovat umíte. V případě nouze se urychleně doučíte. Učebnic základů algoritmizace a programování existuje celá řada, můžete zkusit třeba [4, 1] či informace na stránkách Katedry počítačů ČVUT FEL [2], například ty o předmětu Algoritmizace [3].

Podle čeho se to učí: Literatura je téměř výhradně anglicky, kompletní seznam i s případnými odkazy naleznete na webových stránkách předmětu a také na konci tohoto textu.

1 Algoritmy a algoritmizace

Algoritmus je

- přesný návod či postup, kterým lze vyřešit daný typ úlohy.
- efektivní postup pro výpočet hodnoty nějaké funkce vyjádřený konečným počtem instrukcí.

Definice 1. Algoritmem rozumíme postup, podle kterého se z dat vstupních $x[n]$ vygenerují data výstupní $y[n]$.

Obrázek 1: Formální pohled na algoritmus ze systémového hlediska

- Typy algoritmů
- Co potřebujete znát ?
- Kam až můžeme dojít ?

1.1 Vlastnosti algoritmů

Každý algoritmus musí mít následující vlastnosti:

1. **Konečnost:** výpočet se ukončí v „rozumně“ konečném čase.
2. **Hromadnost:** není sestrojen pouze na jediné $u[n]$, ale na celou řadu možných vstupů.
3. **Jednoznačnost:** přechod do následujícího stavu algoritmu je jednoznačně určen výsledkem stavu předchozího.

Komentář k vlastnostem algoritmů

1. **Konečnost:** předpověď počasí na zítra dosažená výpočtem o den později nemá význam.
2. **Hromadnost:** program pro výpočet odmocniny pracuje nad množinou čísel, není konstruován pro každé číslo zvlášť.
3. **Jednoznačnost:** každý algoritmus je složen z kroků, které na sebe vzájemně navazují. Každý krok je charakterizován jako přechod z jednoho stavu do jiného. Každý stav algoritmu je určen zpracovávanými daty a na tom, jak data v jednotlivých stavech vypadají. Je tedy pevně určeno, který krok bude následovat.

Příklad 2. Numerický výpočet odmocniny Další variantou je numerický výpočet druhé odmocniny čísla x pomocí nelineární diferenční rovnice

$$y[n+1] = \frac{1}{2}\left(y[n] + \frac{u[n]}{y[n]}\right),$$

kde vstupní posloupnost $u[n] = x \cdot \mathbf{1}[n]$ a výstupní posloupnost $y[n]$ postupně konverguje k hodnotě odmocniny. Počáteční podmínku můžeme volit v podstatě libovolnou, například $y[0] = x$ nebo $y[0] = 1$.

Odmocnina z čísla 10 je s přesností na 10 desetinných míst rovna $\sqrt{10} = 3{,}16227766017$.

Pro $u[n] = 10$ dostáváme postupně

$$\begin{array}{ll} y[0] = 3 & y[0]^2 = 9 \\ y[1] = 3{,}165 & y[1]^2 = 10{,}017225 \\ y[2] = 3{,}162278 & y[2]^2 = 10{,}0000021493 \\ y[3] = 3{,}1622776601 & y[3]^2 = 9{,}9999999996 \\ \vdots & \end{array}$$

2 Aplikace algoritmů

Internet umožňuje lidem na celém světě rychle vyhledávat a přistupovat k obrovskému množství informací. Aby to fungovalo, musí poskytovatelé internetu a poskytovatelé internetových služeb používat chytré algoritmy, umožňující zpracovávat a spravovat tak velké množství dat. Příklady úloh, na které v této oblasti narazíme, jsou například *hledání vhodných cest* pro datové pakety, cestující mezi jednotlivými uzly sítě, či *vyhledávání* stránek s určitým obsahem.

Velký objem obchodů je v dnešní době uzavírán elektronicky, a mnoho služeb funguje i na elektronické bázi. Pro celý obor e-komerce je zcela klíčová schopnost uchovat důvěrné údaje (čísla kreditních karet, hesla, či bankovní informace) opravdu v tajnosti. Mezi úlohy z této oblasti, na které narazíme, patří *šifrování veřejným klíčem* či *digitální podpis*.

Ve oblastech výroby a přepravy řeší firmy často problém optimální alokace zdrojů tak, aby byly na jednu stranu minimalizovány výrobní či režijní náklady, na druhou stranu aby byl co možná nejvyšší užitek. Letecký dopravce se bude snažit přiřazovat posádky na jednotlivé lety způsobem, jenž generuje co nejmenší dodatečné náklady – zároveň je jeho manévrovací prostor ovšem omezen nařízeními z oblasti bezpečnosti provozu, všeobecnými právními předpisy a podobně. Poskytovatel internetu při investicích do infrastruktury potřebuje investovat své zdroje cíleně tak, aby výsledek co nejefektivněji sloužil zákazníkům. Obě tyto úlohy řeší algoritmy matematické optimalizační techniky zvané *lineární programování*.

Ve oblastech výroby a přepravy řeší firmy často problém optimální alokace zdrojů: minimalizace nákladů vs. co možná nejvyšší užitek.

Letecký dopravce: přiřazení posádek na lety s co nejmenšími náklady, optimální využití strojů. **Poskytovatel internetu:** cílené investice do infrastruktury. **Svoz odpadu:** Minimum najetých kilometrů.

Příklady algoritmů:

- *lineární* či *dynamické programování* – optimalizace,
- *grafové algoritmy* – komponenty grafu, kostra, nejkratší cesta.

Numerická matematika

Numerické řešení soustav algebraických rovnic, diferenciálních rovnic a speciálních funkcí:

Metoda konečných prvků: řešení složitých parciálních diferenciálních rovnic s praktickými aplikacemi

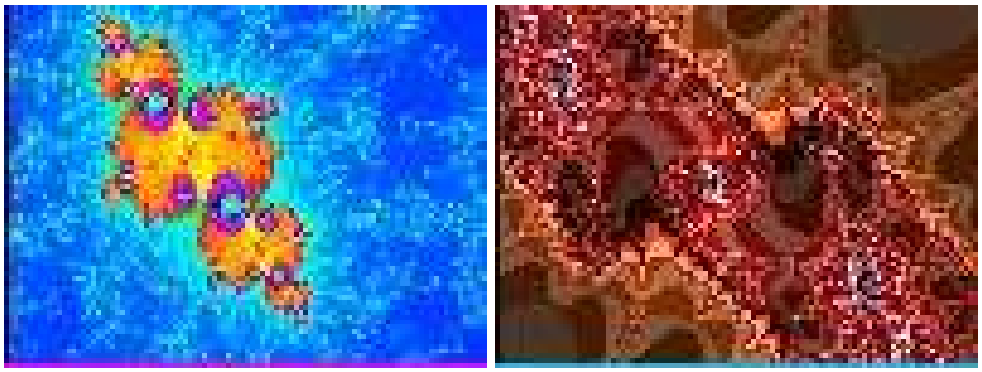
Jinak těžko řešitelné úlohy: nelineární parciální diferenciální rovnice, například Navierovy-Stokesovy rovnice. Tyto rovnice popisují proudění a počítáme je například při studiu obtékání vzduchu okolo křídel, viz. Obrázek 4.

Navierovy-Stokesovy rovnice:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\nabla)\mathbf{u} = \mathbf{f} - \nabla p + \nu \triangle \mathbf{u}$$

kde $\mathbf{u}$ a $\mathbf{f}$ jsou vektorové funkce rychlosti a síly, p je tlak a ν je úměrná viskozitě kapaliny.

$$\frac{\partial u_x}{\partial t} + u_x \frac{\partial u_x}{\partial x} + u_y \frac{\partial u_y}{\partial y} + u_z \frac{\partial u_z}{\partial z} =$$



Obrázek 2: Fraktály (vlevo tzv. Juliina množina, vpravo).

Obrázek 3: Metoda konečných prvků.

Obrázek 4: I tento jev, způsobený prouděním okolo křídla letounu, lze popsat Navierovými-Stoeksovými rovnicemi.

$$= f_x(x, y, z, t) - \frac{\partial p}{\partial x} + \nu \left[\frac{\partial^2 u_x}{\partial x^2} + \frac{\partial^2 u_x}{\partial y^2} + \frac{\partial^2 u_x}{\partial z^2} \right]$$

Výše uvedené seznamy nejsou zdaleka vyčerpávající, můžeme z nich ale odvodit dvě základní charakterisitky, společné mnoha zajímavým algoritmům:

1. Pro daný problém existuje většinou velké množství možných řešení, z nichž většina z různých důvodů není to, co potřebujeme.
2. Algoritmy mají praktický užitek. Metoda hledání nejkratší cesty v grafu umožní přepravní společnosti snížit přepravní náklady, neboť sníží náklady na palivo a na práci personálu. Ten samý algoritmus může ve směrovači počítačové sítě hledat způsob, jak co nejrychleji doručit vaši zprávu adresátovi.

Algoritmy jsou technologie

Výpočetní čas je omezený zdroj, stejně tak, jak je omezená velikost operační paměti počítače. Oba tyto zdroje je třeba využívat rozumně, – algoritmy, jež jsou efektivní z hlediska doby výpočtu či nároků na operační paměť, jsou nástrojem k takovému rozumnému využití.

Algoritmy, stejně jako počítačový hardware, lze v dnešní době považovat za technologii. Celkový výkon systémů závisí na volbě efektivních algoritmů do stejné míry, jako závisí na volbě dostatečně výkonného hardware. A s tím, jak se vyvíjí počítačové technologie, se ruku v ruce vyvíjí i algoritmy.

3 Prerekvizity předmětu

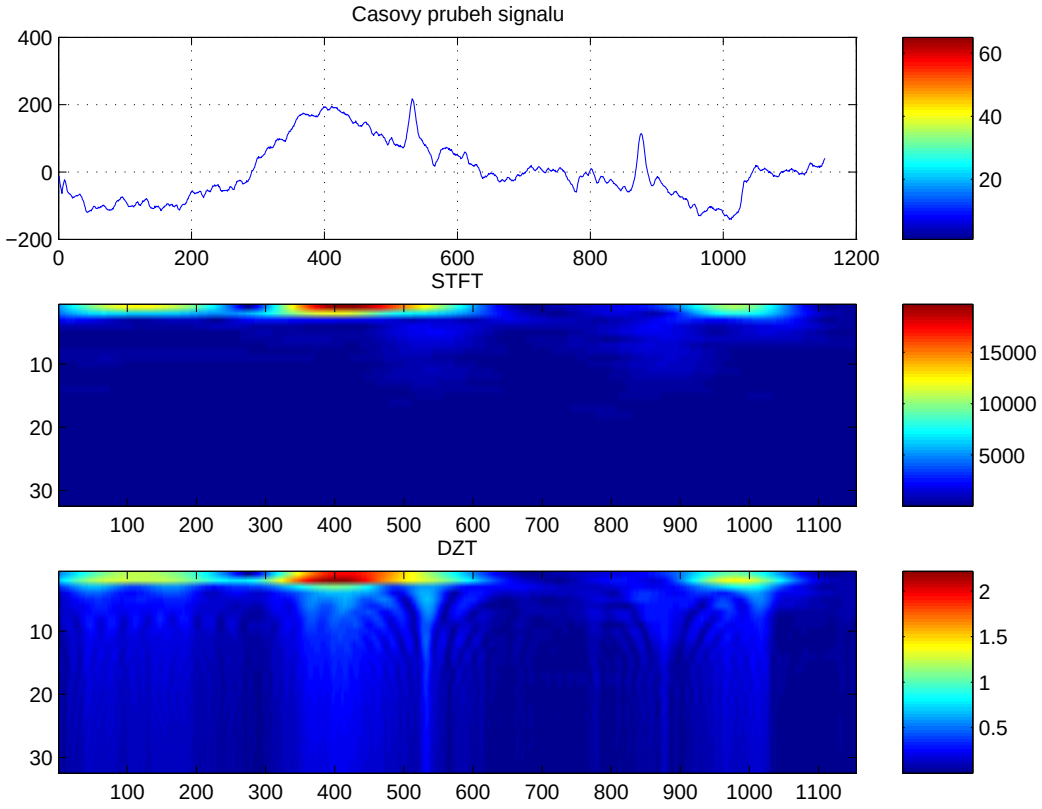
Předpokládáme:

- základy algebry a matematické analýzy
- základy numerické matematiky
- diferenční rovnice a jejich řešení
- základy strukturovaného programování
- aktivní znalost alespoň jednoho programovacího jazyka (C, C++, Python, Java, Basic) nebo alespoň prostředí MATLAB

Kam až můžeme dojít?

- Objevit krásu některých algoritmů.
- Pochopit třeba numerické základy kryptologie.
- Nebát se inženýrských úloh, které vyžadují algoritmizaci.
- Pochopit rychlé algoritmy s aplikacemi v reálném světě

Rychlá Fourierova transformace – analýza EEG signálu



Obrázek 5: Rychlá Fourierova transformace – analýza EEG signálu

Kurs pokrývá standardní algoritmy, jež nabízí pro daný problém a pro daná vstupní data optimální výkon.

Dvě nejčastější chyby při výběru algoritmu pro danou úlohu:

- **ignorujeme výkon algoritmu** – rychlejší algoritmy jsou současně složitější na implementaci
- **příliš zkoumáme výkon algoritmu** – nepatrně rychlejší algoritmus může být výrazně složitější na implementaci

Reference

[1] Bohuslav Hudec. *Programovací techniky.* Skripta. Česká technika – nakladatelství ČVUT, 2001.

[2] Katedra počítačů ČVUT FEL.

[3] Jiří Zdeněk Miroslav Balík, Božena Mannová. Algoritmizace (x36alg) – podklady k přednáškám.

[4] Miroslav Virius. *Základy algoritmizace.* Skripta. Česká technika – nakladatelství ČVUT, 2008.