Různé algoritmické problémy

Obsah

O přednášce

- směsice algoritmických problémů z různých oblastí
- klasické problémy („co by měl informatik znát“)
- mnoho aplikací
- zajímavé algoritmy, pěkné myšlenky

Teorie čísel a kombinatorika

- problémy „nad čísly“
- vstupem, výstup: většinou čísla
  - jen několik
  - zato mohou být „dlouhá“

Největší společný dělitel

- vstup: přirozená čísla a, b
- výstup: největší společný dělitel a, b
- příklad: 180, 252

Jak na to?

Naivní algoritmus I

- projít všechny čísla od 1 do $min(a, b)$
- pro každé vyzkoušet, zda dělí a i b
- vzít největší

Naivní algoritmus II

- „školní“ algoritmus
- najít všechny dělitele čísel a, b
- projít dělitele, vybrat společné, vynásobit
- příklad:
  - $180 = 2^2 \cdot 3^2 \cdot 5$
  - $504 = 2^3 \cdot 3^2 \cdot 7$
  - $NSD = 2^2 \cdot 3^2 = 36$

Euclidův algoritmus: základ

základní myšlenka: pokud $a > b$, pak:

$$NSD(a, b) = NSD(a - b, b)$$

příklad:

krok	a	b
1	504	180
2	324	180
3	180	144
4	144	36
5	108	36
6	72	36
7	36	36
8	36	0

Operace modulo

- modulo = zbytek po dělení
- příklady:
  - $13 \bmod 5 = 3$
  - $28 \bmod 4 = 0$
  - $14 \bmod 3 = 2$
  - $18 \bmod 7 = ??$
  - $29 \bmod 13 = ??$

Euclidův algoritmus: vylepšení

vylepšená základní myšlenka: pokud $a > b$, pak:

$$NSD(a, b) = NSD(a \ mod \ b, b)$$

krok	*a*	*b*
1	504	180
2	180	144
3	144	36
5	36	36
6	36	0

Euclidův algoritmus: pseudokód

odčítací varianta, bez rekurze

```
function gcd(a, b)
    if a = 0
       return b
    while b != 0
        if a > b
           a := a - b
        else
           b := b - a
    return a
```

Euclidův algoritmus: pseudokód

modulo varianta, rekurzivně

```
function gcd(a, b)
    if b = 0
       return a
    else
       return gcd(b, a mod b)
```

Příklady

- 160, 75
- 57, 33

Prvočísla

- dělitelné jen 1 a sebou samým
- předmět zájmu matematiků od pradávna, cca od 70. let (moderní kryptologie) i důležité aplikace
- problémy s prvočísly:
  - výpis (počet) prvočísel v intervalu
  - test prvočíselnosti
  - rozklad na prvočísla (hledání dělitelů)

Eratosthenovo síto

- problém: výpis prvočísel od 2 do n
- algoritmus: opakuj:
  - označ další neškrknuté číslo na seznamu jako prvočíslo
  - všechny násobky tohoto čísla vyškrkni

Test prvočíselnosti

Problém

Vstup: číslo n
Výstup: ANO/NE (je/není prvočíslo)

Test prvočíselnosti: naivní algoritmus

- zkoušíme všechny možné dělitele od 2 do $n - 1$
- vylepšení:
  - dělíme pouze lichými čísly
  - dělíme pouze čísly tvaru $6k \pm 1$
  - dělíme pouze do $\sqrt{n}$

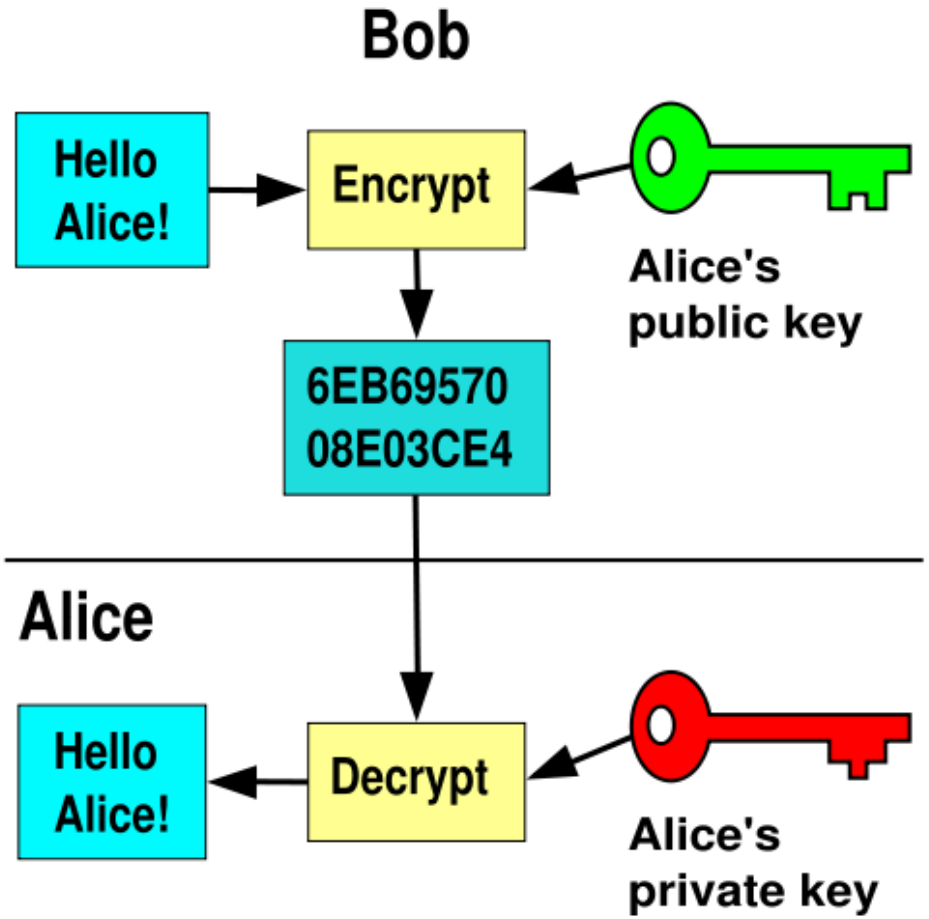
Test prvočíselnosti: chytřejší algoritmy

- náhodnostní algoritmy
- polynomiální deterministický algoritmus (objeven 2002)
- (vysoce) nad rámec tohoto kurzu
- umíme to dělat rychle

Rozklad na prvočísla

- rozklad na prvočísla = faktorizace
- naivní algoritmy:
  - průchod všech možných dělitelů
  - zlepšení podobně jako u testů prvočíslenosti
- chytřejší algoritmy:
  - složitá matematika
  - aktivní výzkumná oblast
  - přesto to neumíme dělat rychle
  - max cca 200 ciferná čísla

Aplikace: asymetrická kryptologie



Asymetrická kryptologie: realizace

- jednosměrné funkce
  - jednoduché vypočítat jedním směrem
  - obtížné druhým (inverze)
  - ilustrace: míchání barev
- RSA (Rivest, Shamir, Adleman) algoritmus
  - jednosměrná funkce: násobení prvočísel (inverze = faktorizace)
  - veřejný klíč: součin velkých prvočísel
  - bezpečnost $\sim$ nikdo neumí provádět efektivně faktorizaci
  - využití modulární aritmetiky, Eulerovy věty, ...

Náhodná čísla

- vstup: „semínko“
- výstup: posloupnost „náhodných“ čísel
- „pseudo-náhodná“ čísla – snaha, aby posloupnost měla charakteristiky co nejpodobnější reálným náhodným číslům
- hlavní žádoucí vlastnosti: uniformní distribuce, vzájemná nezávislost
- aplikace: numerické výpočty, simulace, kryptologie, ...

Generování pseudo-náhodných čísel

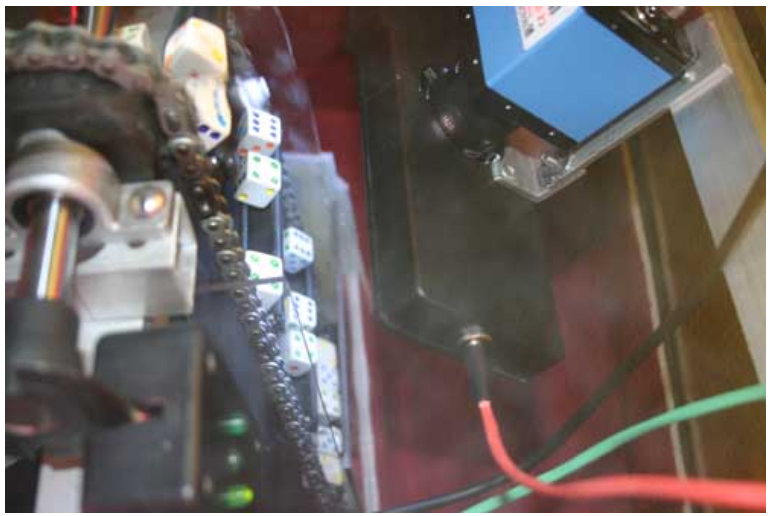
základní přístup:

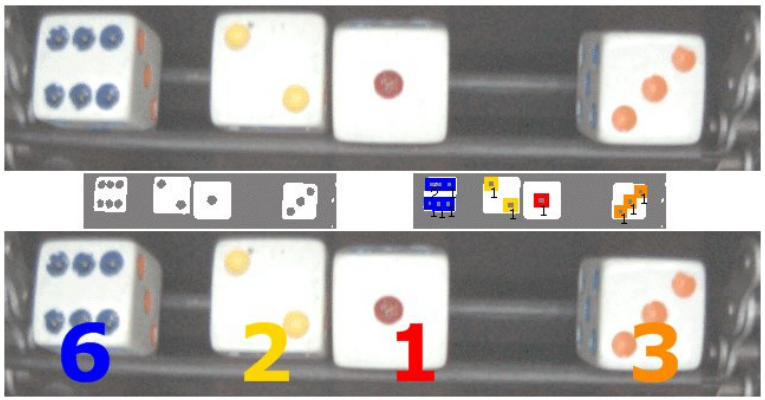
- lineární kongruentní rovnice
- iterování rovnice typu:

$$X_{n+1} = (aX_n + b) \ mod \ m$$

Opravdu náhodná čísla

- `random.org` – atmosférický šum
- `http://gamesbyemail.com/News/DiceOMatic`





Matice

- násobení matic
- výpočet determinantu
- řešení lineárních rovnic (Gaussova eliminační metoda)

Lineární programování

- hledání přiřazení proměnným:
  - splňujících systém lineárních nerovnic
  - maximalizující lineární optimalizační funkci
- obecný problém, má mnoho konkrétních aplikací (grafy, optimalizace, ...)
- základní řešení: simplexový algoritmus

Kombinatorika

generování (např. pro množinu $\{A, B, C, D\}$:

- permutací ($ABCD, ABDC, ACBD, ...$)
- kombinací, podmnožin ($\emptyset, A, B, C, D, AB, AC, ..$)
- rozdělení (partitions) ($(ABCD), (A, BCD), (A, B, CD), ..$)

Řetězce

- vstup: textový řetězec
- výstup: určitá informace o textu, zpracovaný text
- příklady aplikací:
  - vyhledávání v „běžných“ textech (editory, prohlížeče)
  - komprese souborů
  - bioinformatika

Nejdelší rostoucí úsek, podposloupnost

Pozn. nikoliv text, ale posloupnost čísel, typem algoritmu se sem ale hodí

Nejdelší rostoucí úsek

vstup: posloupnost čísel
výstup: nejdelší rostoucí úsek (po sobě jdoucí čísla)

Nejdelší rostoucí podposloupnost

vstup: posloupnost čísel
výstup: nejdelší rostoucí podposloupnost (čísla nemusí jít po sobě)

Nejdelší rostoucí úsek, podposloupnost

příklad: 4, 1, 8, 6, 5, 11, 2, 4, 9, 12, 5, 13, 7, 8, 12, 10, 11

Nejdelší rostoucí úsek, podposloupnost

příklad: 4, 1, 8, 6, 5, 11, 2, 4, 9, 12, 5, 13, 7, 8, 12, 10, 11

nejdelší rostoucí úsek: 2, 4, 9, 12

nejdelší rostoucí podposloupnost: 1, 2, 4, 5, 7, 8, 10, 11

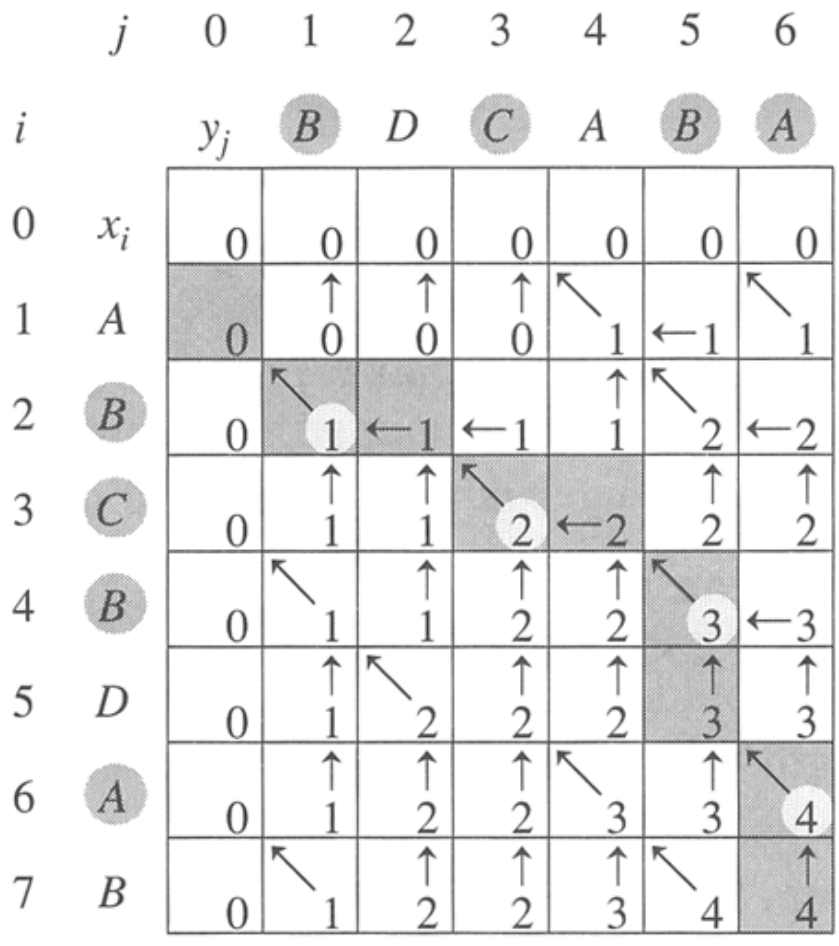
Algoritmy

- nejdelší rostoucí úsek:
  - jeden průchod pole, lineární složitost $O(n)$
  - pamatuji si aktuální délku rostoucí podposloupnosti + dosavadní maximum
- nejdelší rostoucí podposloupnost:
  - „dynamické programování“
  - pro každou pozici si pamatuji délku nejdelší podposloupnosti končící v tom místě
  - přímočaře: kvadratická složitost $O(n^2)$
  - efektivní implementace $O(n \log n)$

Nejdelší společný podřetězec, podposloupnost

- vstup: dva řetězce s_1, s_2
- výstup: nejdelší společný úsek/podposloupnost (rozdíl podobně jako u předchozího)
- příklad:
  - BDCABA
  - ABCBDAB

Nejdelší společná podposloupnost



i		j	0	1	2	3	4	5	6
			y_j	B	D	C	A	B	A
0	x_i		0	0	0	0	0	0	0
1	A		0	↑ 0	↑ 0	↑ 0	↖ 1	← 1	↖ 1
2	B		0	↖ 1	← 1	← 1	↑ 1	↖ 2	← 2
3	C		0	↑ 1	↑ 1	↖ 2	← 2	↑ 2	↑ 2
4	B		0	↖ 1	↑ 1	↑ 2	↑ 2	↖ 3	← 3
5	D		0	↑ 1	↖ 2	↑ 2	↑ 2	↑ 3	↑ 3
6	A		0	↑ 1	↑ 2	↑ 2	↖ 3	↑ 3	↖ 4
7	B		0	↖ 1	↑ 2	↑ 2	↑ 3	↖ 4	↑ 4

T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein: Introduction to Algorithms.

Nejdelší společná podposloupnost

„dynamické programování“

Algorithm 7.1 LCS
Input: Two strings A and B of lengths n and m, respectively, over an alphabet Σ.
Output: The length of the longest common subsequence of A and B.

```
1.  for i ← 0 to n
2.      L[i, 0] ← 0
3.  end for
4.  for j ← 0 to m
5.      L[0, j] ← 0
6.  end for
7.  for i ← 1 to n
8.      for j ← 1 to m
9.          if a_i = b_j then L[i, j] ← L[i − 1, j − 1] + 1
10.         else L[i, j] ← max{L[i, j − 1], L[i − 1, j]}
11.         end if
12.     end for
13. end for
14. return L[n, m]
```

M. H. Alsuwaiyel: Algorithms, Design Techniques and Analysis.

Komprese textu

- vstup: text
- výstup: bezeztrátová komprese (text lze rekonstruovat)
- aplikace: šetření místa (viz např. zip)
- příklad: dlouhý textový dokument, bitmapový obrázek – jak byste komprimovali?

Komprese textu: přístupy

- „Run-length encoding“ (RLE):
  - nahrazení opakovaných výskytů stejného znaku
  - např. bitmapový obrázek
- (statické) slovníkové metody
- statistické metody – kódování podle textu

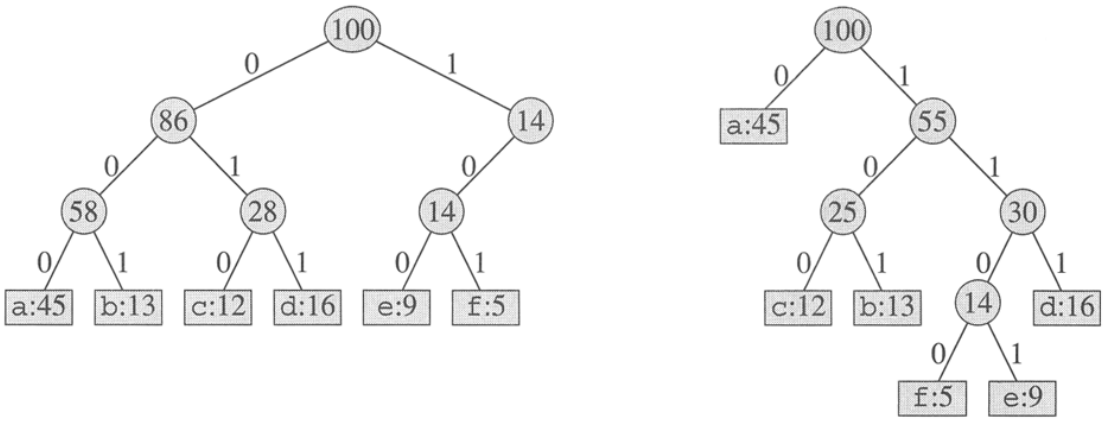
Kódování: příklad

	a	b	c	d	e	f
Frequency (in thousands)	45	13	12	16	9	5
Fixed-length codeword	000	001	010	011	100	101
Variable-length codeword	0	101	100	111	1101	1100

T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein: Introduction to Algorithms.

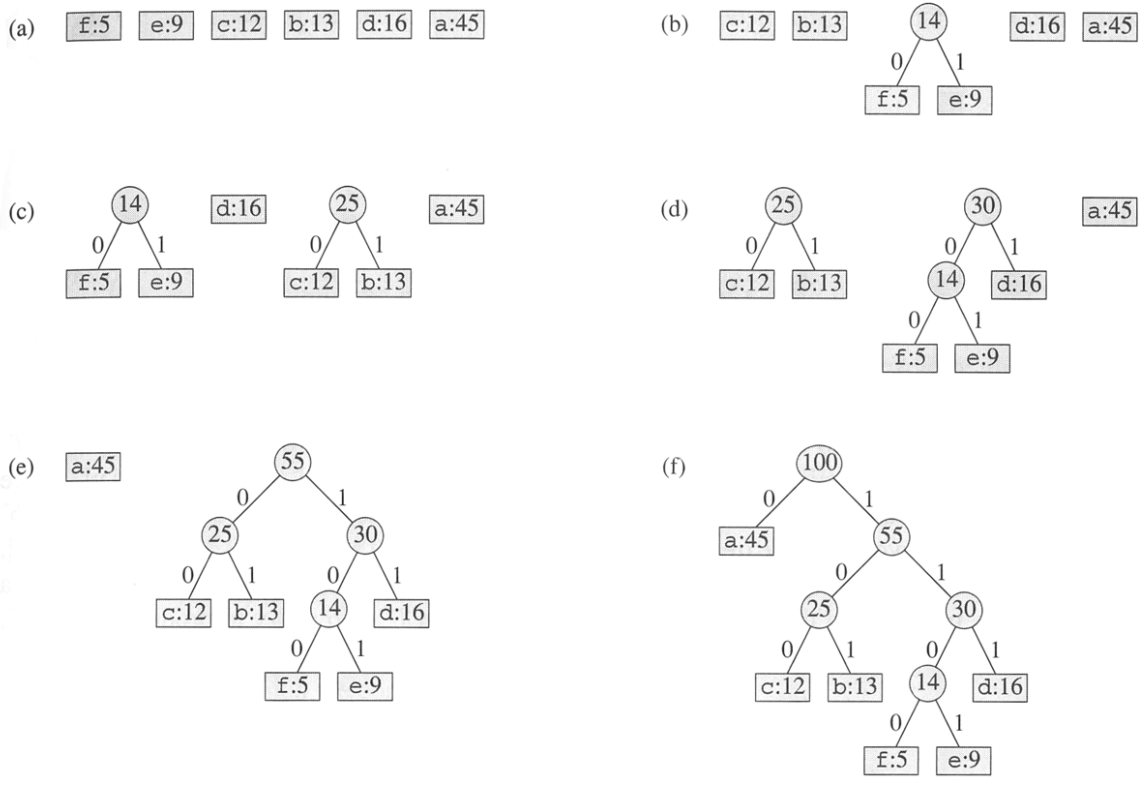
proměnlivá délka – kód musí být „bez-prefixový“

Huffmanovo kódování



T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein: Introduction to Algorithms.

Huffmanovo kódování



T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein: Introduction to Algorithms.

Geometrické algoritmy

- typicky problémy v rovině (případně 3D prostoru)
- vstupem množina bodů, složitějších objektů
- výstup body, objekty, splňující zadané požadavky
- kromě algoritmického nápadu často nutný i „geometrický vhled“

Základní operace

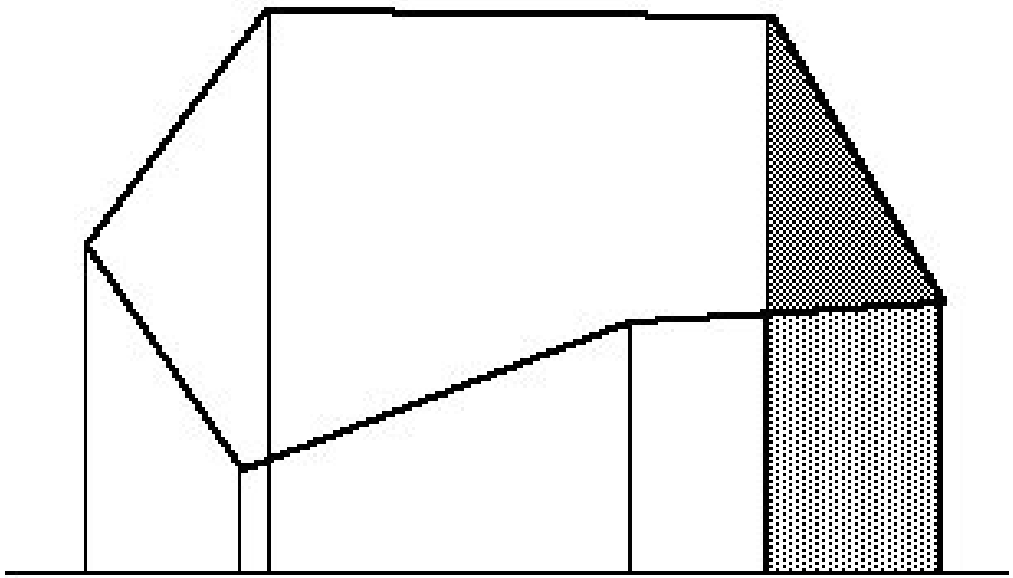
- průsečík dvou úseček
- obsah trojúhelníku
- nad/pod test: bod vzhledem k přímce
- uvnitř test: bod vzhledem ke kruhu

Obsah polygonu

- vstup: polygon (ne nutně konvexní)
- výstup: obsah

Obsah polygonu

- vstup: polygon (ne nutně konvexní)
- výstup: obsah
- algoritmus: „diskrétní integrování“



Konvexní obal

- vstup: množina bodů v rovině (souřadnice x_i, y_i)
- výstup: uspořádaný seznam bodů tvořících konvexní obal

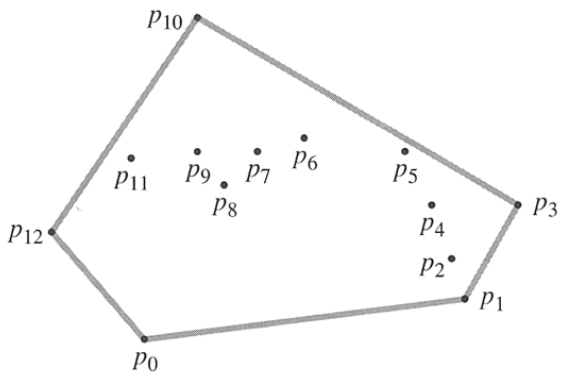
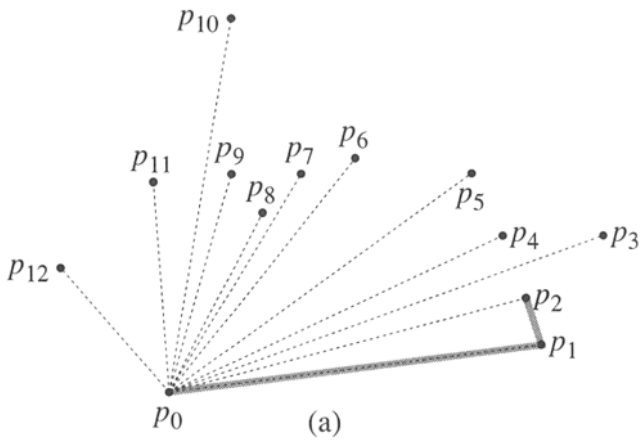
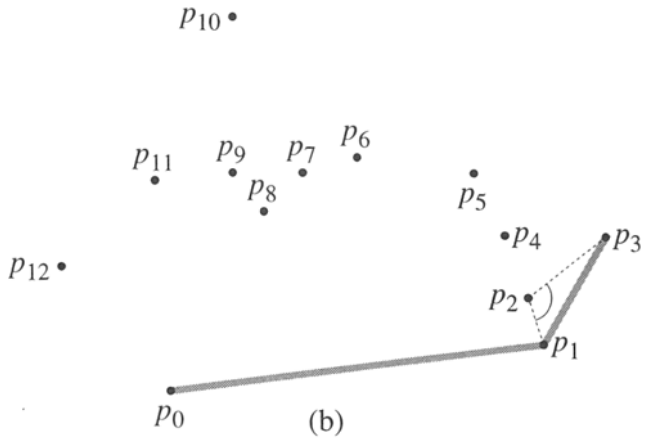


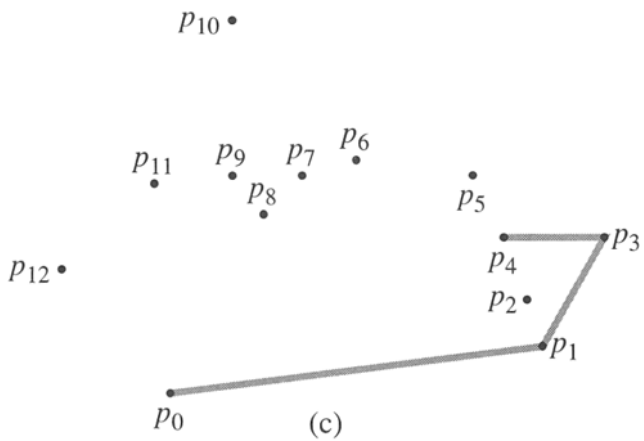
Figure 33.6 A set of points $Q = \{p_0, p_1, \ldots, p_{12}\}$ with its convex hull $\mathrm{CH}(Q)$ in gray.

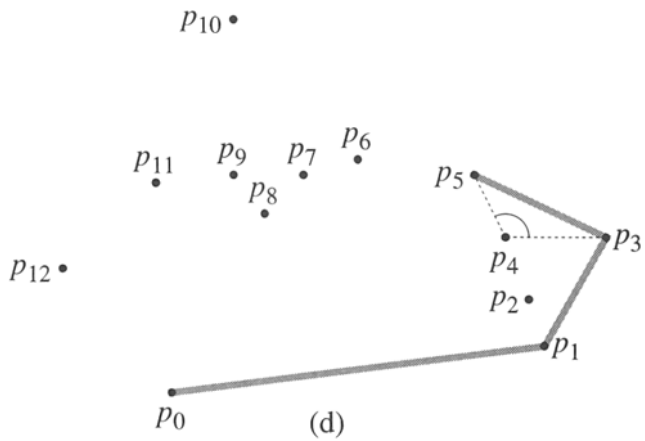
T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein: Introduction to Algorithms.

Graham scan: ilustrace

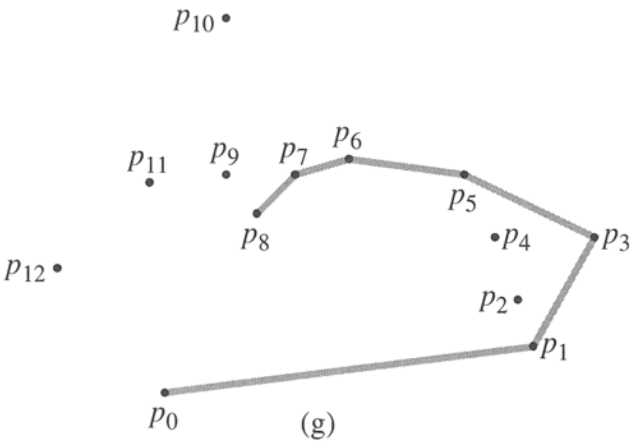


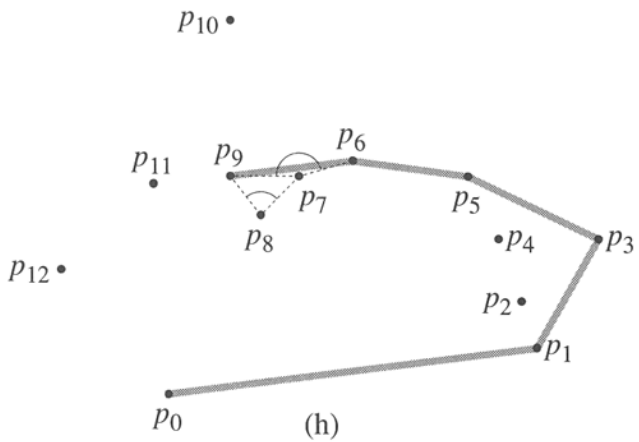


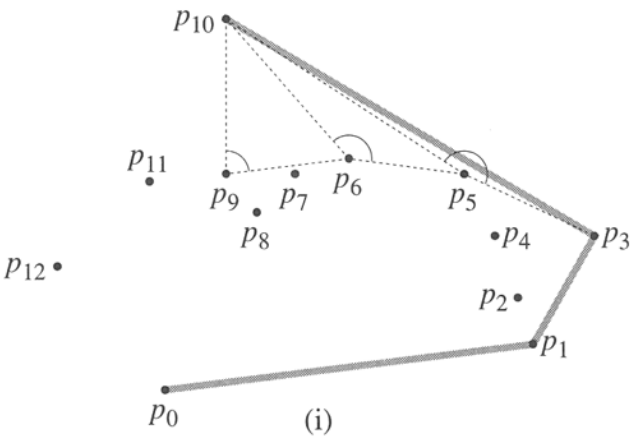


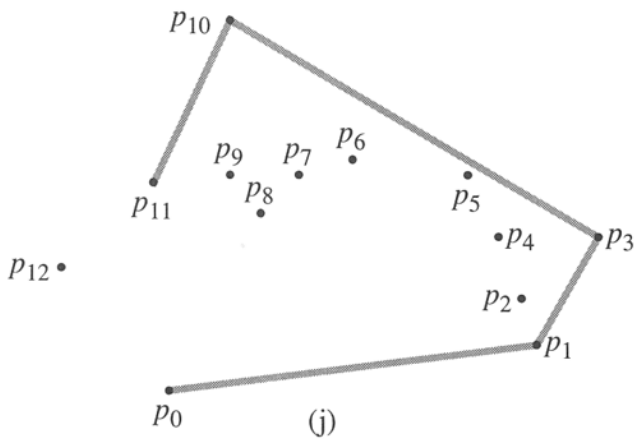


Graham scan: ilustrace









Jarvis march: ilustrace

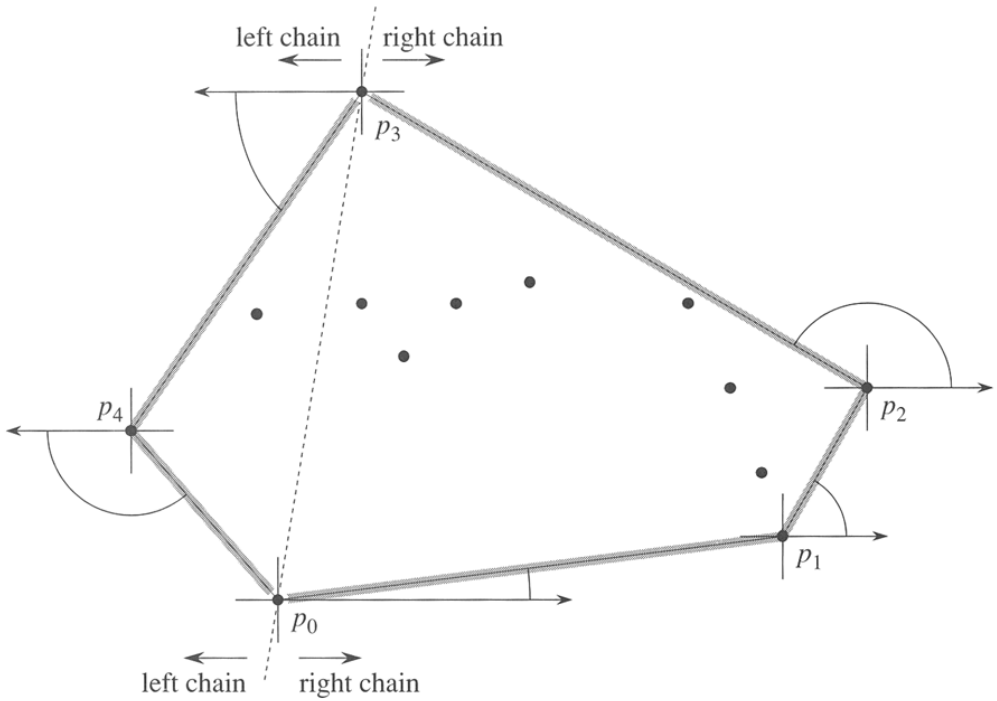
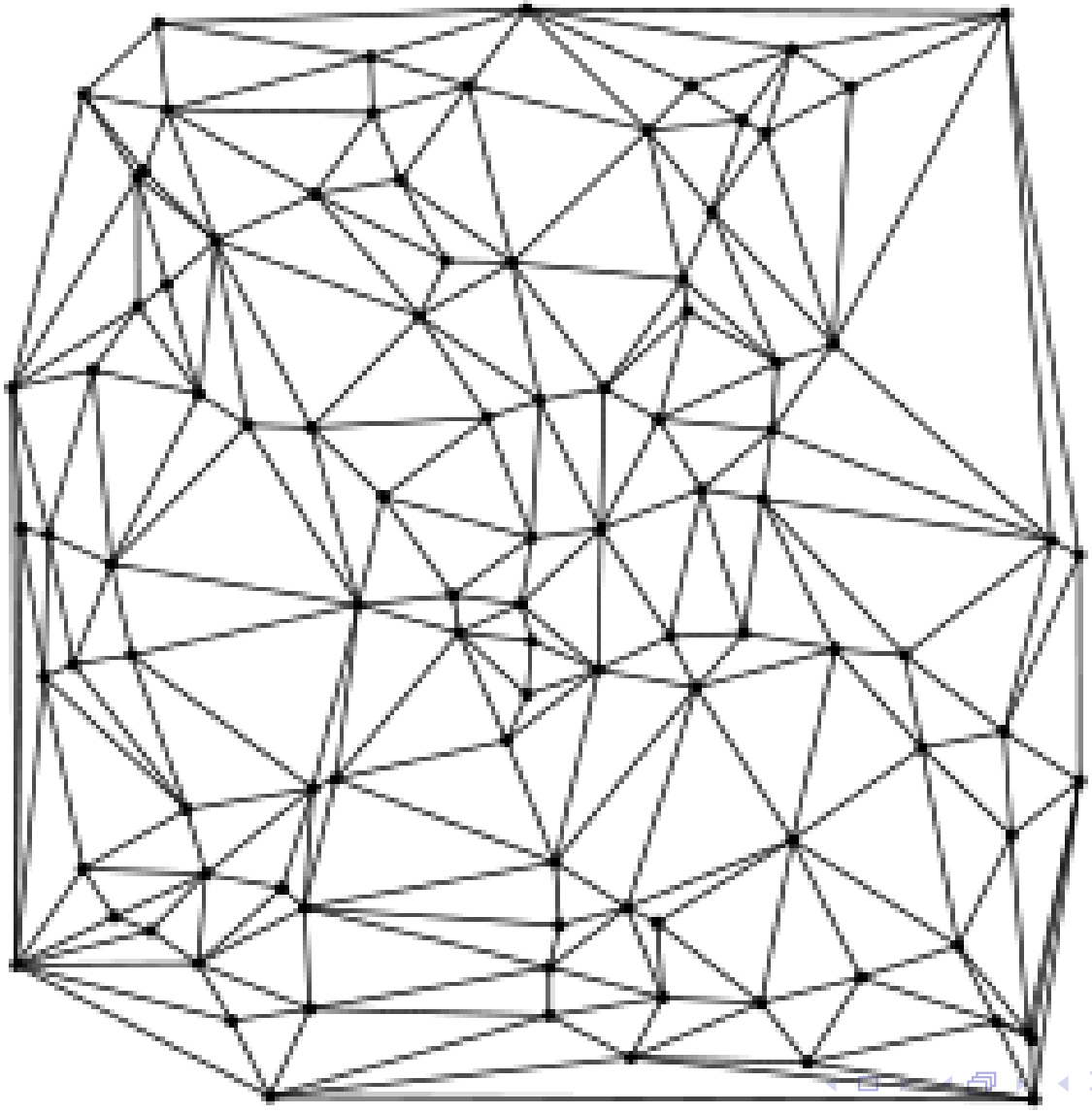


Figure 33.9 The operation of Jarvis's march. The first vertex chosen is the lowest point p_0. The next vertex, p_1, has the smallest polar angle of any point with respect to p_0. Then, p_2 has the smallest polar angle with respect to p_1. The right chain goes as high as the highest point p_3. Then, the left chain is constructed by finding smallest polar angles with respect to the negative x-axis.

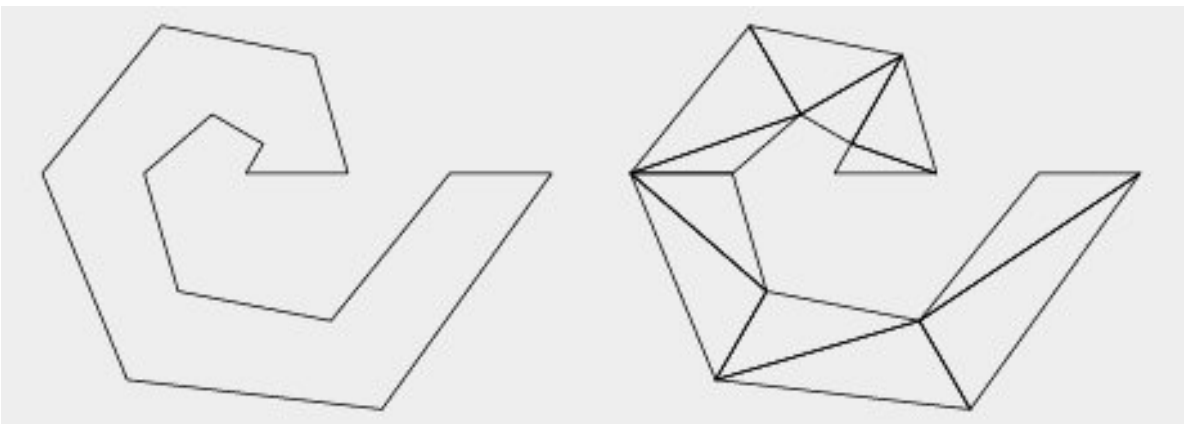
T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein: Introduction to Algorithms.

Triangulace



Triangulace

- vstup: množina bodů, příp. polygon
- výstup: triangulace



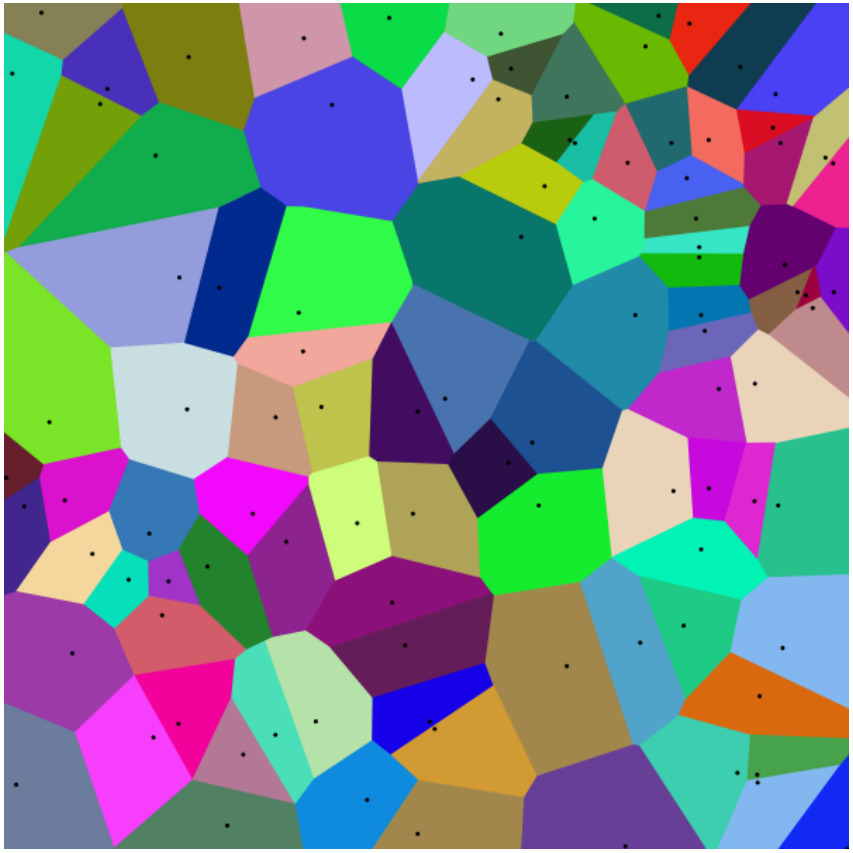
Triangulace

- variace:
  - Delaunay: „pěkné úhly“ (minimalizace maximálního úhlu)
  - minimalizace vzdáleností
  - 3D: trojúhelníky $\longrightarrow$ čtyřstěny
- aplikace: počítačová grafika
- algoritmy:
  - konvexní polygon: přímočaré
  - další variace postupně komplikovanější

Voroného diagram

- vstup: množina bodů
- výstup: rozdělení roviny na oblasti „ke kterému bodu je odsud nejblíž“
- aplikace:
  - vyhledávání
  - umisťování nových bodů (např. restaurace)
  - plánování cest (např. roboti)
  - triangulace

Voroného diagram



animace algoritmu:
http://www.diku.dk/hjemmesider/studerende/duff/Fortune/

Shrnutí

- algoritmické problémy z různých oblastí
- teorie čísel, řetězce, geometrické algoritmy
- přehled:
  - o co v problémech jde
  - myšlenky základních algoritmů