Vybrané kapitoly z teoretickej informatiky (1), 2-AIN-105

Vyučujú:
Jozef Šiška, I-7, siska@ii.fmph.uniba.sk
Michal Kováč, Michal Anderle, František Ďuriš

Web: http://compbio.fmph.uniba.sk/vyuka/vkti1/

Literatúra:

Brassard, Bratley: Fundamentals of Algorithmics, Prentice Hall 1995
Pardubská: Vybrané kapitoly z teoretickej informatiky (1)
Bentley: Programming Pearls, ACM Press 1999
Cormen, Leiserson, Rivest, Stein: Introduction to Algorithms, MIT Press 2009

O čom je tento predmet?

- Pre daný problém, úlohou je nájsť **efektívny algoritmus**, ktorý tento problém rieši.
- Rozpoznať, kedy efektívny algoritmus **neexistuje**.

Osnova predmetu:

- Úvod, výpočtová zložitosť
- Techniky tvorby efektívnych algoritmov
  (greedy algoritmy, dynamické programovanie, rozdeľuj a panuj)
- NP-ťažké problémy
- Nevypočítateľné problémy

V ďalšom semestri:

- Ako sa vysporiadať s problémami, o ktorých vieme že sú ťažké?

Hodnotenie predmetu

- 30%: Domáce úlohy (cca každé dva týždne)
  (vr. jedného programátorského príkladu)
- 20%: Midterm
- 50%: Písomná skúška
- zo skúšky je potrebné získať aspoň 50% bodov
- 90+ = A, 80+ = B, 70+ = C, 60+ = D, 50+ = E

Opisovanie

- Budeme kruto trestať:
  - -100% príslušného bodového hodnotenia
  - disciplinárna komisia
- **Podporujeme** diskusiu o domácich úlohách, **ale:**
  - Nerobte si poznámky
  - Počkajte niekoľko hodín, kým začnete spisovať vlastné riešenie

Bentleyho problém: Riešenie 1

```
max:=0;
for i:=1 to n do // start of subarray
  for j:=i to n do  // end of subarray
    // compute sum of subarray A[i]..A[j]
    sum:=0;
    for k:=i to j do
      sum:=sum+A[k];
    // compare to maximum
    if sum>max then max:=sum;
```

Zložitosť: $O(n^3)$

Bentleyho problém: Riešenie 2a

```
max:=0;
for i:=1 to n do
  sum:=0;
  for j:=i to n do
    sum:=sum+A[j];
    // sum is now sum of subarray A[i]..A[j]
    // compare to maximum
    if sum>max then max:=sum;
```

Zložitosť: $O(n^2)$

Bentleyho problém: Riešenie 2b

```
// precompute B[i]=A[1]+...+A[i]
B[0]:=0;
for i:=1 to n do
   B[i]:=B[i-1]+A[i];

max:=0;
for i:=1 to n do
  for j:=i to n do
    // compare to maximum
    if B[j]-B[i-1]>max then
      max:=B[j]-B[i-1];
```

Zložitosť: $O(n^2)$

Bentleyho problém: Riešenie 4a

```
maxsol:=0; tail:=0;
for i:=1 to n do
  tail:=max(tail+A[i],0);
  maxsol:=max(maxsol,tail);
```

Zložitosť: $O(n)$

Bentleyho problém: Riešenie 4b

```
maxsol:=0; prefix:=0; min_prefix:=0;
for i:=1 to n do
  prefix:=prefix+A[i];
  min_prefix=min(min_prefix,prefix);
  maxsol:=max(maxsol,prefix-min_prefix);
```

Zložitosť: $O(n)$

Čas potrebný na riešenie problému veľkosti...

	Sol.4 $O(n)$	**Sol.3** $O(n \log n)$	**Sol.2** $O(n^2)$	**Sol.1** $O(n^3)$	**Sol.0** $O(2^n)$
10	ε	ε	ε	ε	ε
50	ε	ε	ε	ε	2 weeks
100	ε	ε	ε	ε	2800 univ.
1000	ε	ε	0.02s	4.5s	—
10000	ε	0.01s	2.1s	75m	—
100000	0.04s	0.12s	3.5m	52d	—
1 mil.	0.42s	1.4s	5.8h	142yr	—
10 mil.	4.2s	16.1s	24.3d	140000yr	—

Najväčšia veľkosť problému, ktorý zvládneme vyriešiť za...

	Sol.4 $O(n)$	**Sol.3** $O(n \log n)$	**Sol.2** $O(n^2)$	**Sol.1** $O(n^3)$	**Sol.0** $O(2^n)$
1s	2.3 mil.	740000	6900	610	33
1m	140 mil.	34 mil.	53000	2400	39
1d	200 bil.	35 bil.	2 mil.	26000	49

O koľko viac času potrebujeme, ak sa n zvýši...

	Sol.4 $O(n)$	**Sol.3** $O(n \log n)$	**Sol.2** $O(n^2)$	**Sol.1** $O(n^3)$	**Sol.0** $O(2^n)$
$+1$	—	—	—	—	$\times 2$
$\times 2$	$\times 2$	$\times 2+$	$\times 4$	$\times 8$	—