Algoritmy a datové struktury I

7. cvičení

čtvrtek 7. 4. 2016 9:00

Přehled datových struktur

1. stromy
   - typicky $\Theta(\log n)$ na operaci
   - vyvážené binární: AVL a červeno-černé stromy
   - (a, b)-stromy: korespondence mezi $(2, 4)$-stromy a (left-leaning) červeno-černými stromy
   - splay stromy
2. písmenkové stromy (trie)
3. haldy

	binární	binomiální	fibonacciho
delete min	$\mathcal{O}(\log n)$	$\mathcal{O}(\log n)$	amort. $\mathcal{O}(\log n)$
insert	$\mathcal{O}(\log n)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
decrease	$\mathcal{O}(\log n)$	$\mathcal{O}(\log n)$	amort. $\mathcal{O}(1)$
merge	$\mathcal{O}(n)$	$\mathcal{O}(\log n)$	$\mathcal{O}(1)$

4. hashování
   - prvky ukládáme do N přihrádek, přihrádka je určena hodnotou hashovací funkce na daném prvku
   - pokud je $n = \mathcal{O}(N)$, tak v každé přihrádce je průměrně $\mathcal{O}(1)$ prvků
   - umí insert, delete a find v $\mathcal{O}(1)$ průměrně
   - neumí operace následník a předchůdce
   - často užívané hashovací funkce: $x \mapsto ax \bmod b$ pro vhodné a a b

Třídění

- hloupé algoritmy ($\mathcal{O}(n^2)$): Bubble sort, Insert sort, Select sort, ...
- chytré ($\mathcal{O}(n \log n)$): Merge sort, Heap sort, Quick sort
- dolní odhad $\Omega(n \log n)$ pro algoritmy, které pouze porovnávají prvky
- třídění prvků z malého univerza: Bucket sort (přihrádkové třídění, $\mathcal{O}(n)$)
- opakovaným užitím Bucket sortu můžeme třídit n-tice, celá čísla (Radix sort) či řetězce

---

Informace o cvičení viz `http://iuuk.mff.cuni.cz/~husek/vyuka/ads1516.html`