Jakub Klímek

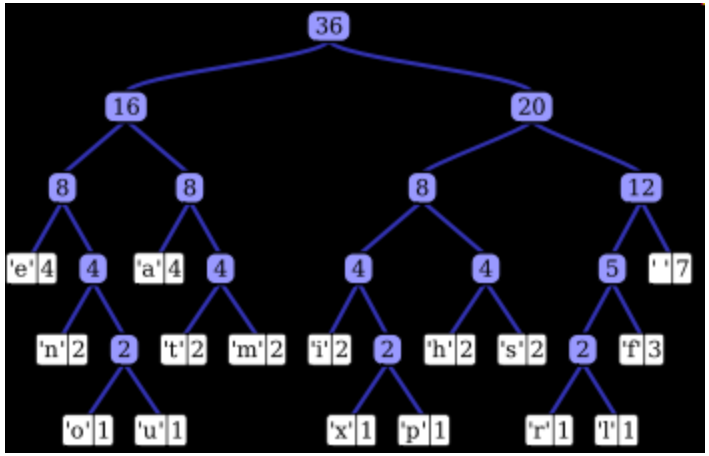
Programování v C++
Zápočtový test 1.2.2011 10:00

Huffmanovo kódování

- Kódování pro bezeztrátovou kompresi dat
- Nejlepší mezi algoritmy kódujícími symbol po symbolu
- Variabilní kódovací tabulka založená na vstupních datech
- Prefixový kód – posloupnost bitů reprezentující jeden symbol není prefixem posloupnosti bitů reprezentující jiný symbol
- Binární strom (kódovací tabulka)
  - Při kódování nula odpovídá levému synu, jednička pravému
- Uzel – symbol (pro listy), váha, levý, pravý, (otec)
  - Váha uzlu je součet vah synů
  - Váha listu je pravděpodobnost výskytu symbolu ve vstupních datech
  - Pro nás: počet výskytů ve vstupních datech

Konstrukce stromu

1. Načíst symboly a spočítat jejich četnost
2. Pro každý symbol vytvořit list (symbol + váha)
3. Nasypat listy do fronty
4. Vybrat 2 uzly s nejnižší vahou z fronty
5. Spojit je pod nový uzel, spočítat jeho váhu a dát do fronty
6. Fronta musí být prioritní
   - Priorita nepřímo úměrná váze
7. Nebo lze použít 2 obyčejné fronty, jednu na listy (naplnit setříděně), jednu na stromy



Komprese souboru

- Pomocí kódování zakódujte vstupní soubor.
- Měl by obsahovat:
  - Velikost vstupního souboru (pro dekompresi)
  - Kódovací tabulku / strom (pro dekompresi)
  - Samotný zakódovaný vstup
- Jeden z možných výsledných kódů pro vstup Simple.txt (hexadecimálně) (záleží na řazení symbolů se stejnou váhou a na pořadí výpisu jednotlivých bitů):
  - `fc bf 31 a7 4b 27 c0 e9 b0 ef d0 88 d3 53 a3 9d fe 6f 70 4e 25 00`
  - Pro zobrazení můžete použít třeba tool HexEdit

Použití

- huff –a <input> <output>
  - komprese
- huff –x <input> <output>
  - dekomprese
  - z komprimovaných souborů vytvoří zpět originály
  - vstup a výstup by měly být bit po bitu totožné
  - pro porovnání můžete použít příkaz comp
- Ošetřete špatné parametry a vstupy
- Program nesmí padat