VYSOKÉ UČENIE TECHNICKÉ V BRNE
FAKULTA INFORMAČNÝCH TECHNOLOGIÍ

Kódování a komprese dat (KKO)

2017/2018

Adaptivní Huffmanovo kódování

Ľubomír Gallovič (xgallo03)

Február 2018

1 Úvod

Cieľom projektu bolo implementovať knižnicu a aplikáciu na kódovanie a dekódovanie textových súborov pomocou algoritmu Adaptive Huffman Coding na 8 bitových symboloch.

2 Adaptívne Huffmanovo kódovanie

Huffmanovo kódovanie funguje na princípe zistenia početnosti jednotlivých symbolov v kódovanom texte, zostrojením Huffmanovho stromu a zakódovania znakov ako ciest v danom strome. Pri kódovaní je potrebné urobiť dva priebehy textu, jeden na vytvorenie Huffmanovho stromu, a druhý na samotné kódovanie. Do kódovaného textu je takisto potrebné uložiť Huffmanov strom, podľa ktorého bude prebiehať dekódovanie.

Adaptívne Huffmanovo kódovanie sa od klasického líši v tom, že kódovanie textu a vytváranie Huffmanovho stomu prebieha súčasne, vďaka čomu potrebuje len jeden priebeh textom. Kódy jednotlivých symbolov sú získavané z postupne sa zväčšujúceho a meniaceho Huddmanovho stromu, vďaka čomu dosahuje výsledný súbor menšiu veľkosť. Aby tento algoritmus fungoval, je potrebné zaručiť vytváranie ekvivalentného Huffmanovho stromu aj pri spracovaní kódovaného textu, pričom na rovnakých miestach v texte musia byť stromy pri kódovaní a dekódovaní ekvivalentné. Aby bol pri dekódovaní strom správne namapovaný, je potrebné pri prvom výskyte každého znaku na túto skutočnosť upozorniť, a následne tam tento znak uložiť vo svojej ASCII forme. Na to slúži špeciálny symbol v Huffmanovom strome, ktorý upozorňje, že nasledujúcich 8 bitov je nekódovaný znak.

3 Huffmanov strom

Každý uzol má definovanú svoju hodnotu, váhu a poradie. Navyše majú všetky uzly okrem koreňa definovaného svojho predka, a všetky nelistové uzly majú definovaného svojho ľavého a pravého potomka. Listy majú hodnotu ich znaku a ostatné uzly špeciálnu hodnotu UNDEFINED. Strom vždy obsahuje špeciálny list s hodnotou NEWNODE, ktorý sa používa pri namapovaní nových znakov. Váha uzlu u listov definuje dosiaľ zaznamenaný počet výskytov daného znaku, a u ostatných uzlov definuje súčet ich potomkov. Poradie uzlov začína na hodnote 512, pričom každý nový uzol má pridelenú nižšiu hodnotu. Musí platiť, že uzly s vyššou váhou majú vyššie poradie ako uzly s nižšou váhou, a hodnota poradí je využitá pri presúvaní uzlov v rámci stromu.

V prípade výskytu nového znaku sa využije list NEWNODE, ktorý vytvorí dvoch nových potomkov: pravý potomok má hodnotu nového znaku, váhu jedna, a poradie o jedno nižšie ako rodičovský uzol; ľavý potomok je nový NEWNODE s váhou nula a poradím o dva nižším ako rodič. V prípade pridania nového znaku alebo zmeny počtu výskytov je potrebné zabezpečiť, aby boli zachované vlastnosti Huffmanovho stromu. U relevantných uzlov je potrebné overiť správnosť ich pozície a prípadne ich zameniť s iným uzlom. Kontrolovaný je každý predok od novo vytvoreného alebo zmeneného uzlu až ku koreňu. Najprv je nájdený uzol s najvyšším poradím z uzlov s rovnakou váhou ako kontrolovaný uzol, a v prípade, že to nie je kontrolovaný uzol, alebo jeho rodič dochádza k zámene týchto uzlov. Cieľom je dosiahnúť, aby uzol, ktorému sa zvýši váha mal najvyššie poradové číslo z jeho váhovej kategórie, pretože inak by po navýšení jeho hmotnosti prestalo platiť, že uzly s vyššou hmotnosťou majú vyššie poradia. Cesta v strome je uložená ako postupnosť bitov, pričom 0 znamená ľavý potomok a 1 pravý potomok.

4 Implementácia

Kódovanie textu prebieha vo funkcii AHEDEncoding. Vstup je čítaný znak po znaku, pričom v prípade, že sa na vstupe nachádza prvý výskyt nejakého znaku je na výstup vypísaná cesta k uzlu NEWNODE nasledovaná ôsmimi bitmi ASCII kódu. Ak sa znak na vstupe už vyskytol, je vypísaná cesta k nemu v strome. Do stromu je následne doplnený nový výskyt/znak a strom je následne upravený algoritmom popísaným vyššie. Keďže cesty v strome nie sú častokrát zarovnané na 8 bitov, sú tieto bity uložené najprv do bufferu, a po dosiahnutí dostatočnej dĺžky vypísané na výstup ako byty. Po prečítaní celého vstupu je na koniec kódovaného súboru ešte uložená cesta k NEWNODE nasledovaná EOF, čo indikuje správne ukončenie zakódovaného súboru.

Dekódovanie prebieha vo funkcii AHEDDecoding, v ktorej sú zo vstupu postupne prečítané byty, a následne sú uložené na koniec bufferu. Program vezme vždy prvý bit z bufferu a podľa jeho hodnoty zmení pozíciu v Huffmanovom strome na ľavého alebo pravého potomka. Pokiaľ dôjde k vyprázdneniu bufferu je prečítaný nasledjúci znak. Ak je v strome dosiahnutý listový uzol, je jeho nekódovaná hodnota zapísaná na výstup. V prípade, že ide o NEWNODE, je načítaných nasledujúcich 8 bitov ako plaintext a uložených na výstup. Po doplnení výstupu dôjde k aktualizácii Huffmanovho stromu. Dekódovanie je považované za správne ukončené, ak posledný prijatý znak je NEWNODE.

5 Testovanie

Program bol testovaný na viacerých vstupoch, vrátane na súbore poskytnutého v zadaní. Pri kódovaní bolo dosiahnuté výrazné zmenšenie veľkosti súboru a pri dekódovaní sa text zhodoval so vstupným textom. Doba kódovania aj dekódovania textu poskytnutého v zadaní trvá približne sekundu.

6 Záver

V tomto projekte bola implementovaná a otestovaná knižnica na kódovanie a dekódovanie súborov 8 bitových znakov Adaptívnym Huffmanovým kódovaním. Poznatky potrebné pre riešenie projektu boli získané z prednášok KKO a z webovej stránky Duke Computer Science - Adaptive Huffman Coding. Program by mal fungovať podľa špecifikácie v zadaní a pri jeho testovaní neboli odhalené žiadne nedostatky.