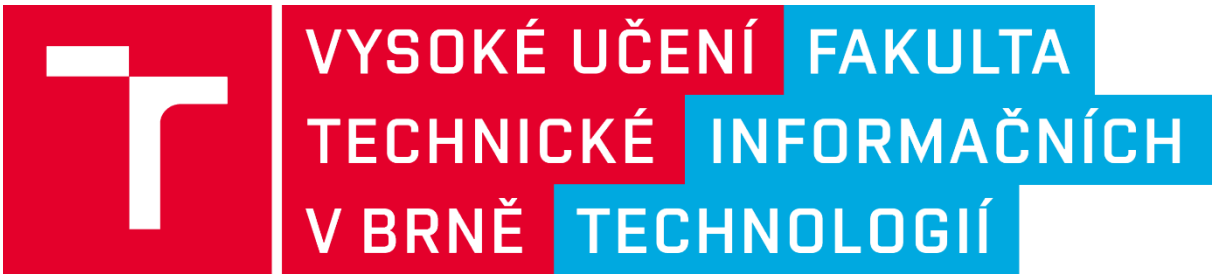


Komprese obrazových dat s využitím Huffmanova kódování

Kódování a komprese dat
2022/2023

Autor: Vojtěch Mimochodek (xmimoc01)
Datum: 5. května 2023

Úvod

Cílem tohoto projektu bylo vytvořit program v jazyce C/C++, který by měl podobu konzolové aplikace. Ta by na vstupu přijala obrázek ve formátu *.raw* a provedla kompresi s pomocí Huffmanova kódování, a to konkrétně s využitím kanonického Huffmanova kódu. Výsledná aplikace by mimo kompresi měla provádět také dekompresi. Obě operace by mělo být možné provádět pomocí statického či adaptivního průchodu. Další možností je aplikace modelu v podobě diference sousedů.

Implementace

Projekt je naimplementován v jazyce C/C++ a sestává z následujících souborů, do nichž byl kód rozdělen:

- *main.cpp*,
- *huffman.cpp, huffman.h,*
- *adaptivescanner.cpp, adaptivescanner.h,*
- *files.cpp, files.h,*
- *helper.cpp, helper.h,*
- *submatrixinfo.cpp, submatrixinfo.h,*
- *symbol.cpp* a *symbol.h*

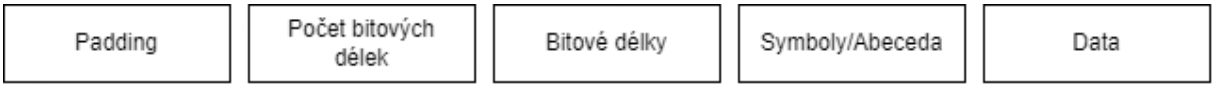
V souboru *main.cpp* jsou nejprve zpracovány vstupní argumenty z příkazové řádky. Dle zadaných argumentů se provádí akce jako komprese souboru, dekomprese souboru, vypsání nápovědy nebo vypsání chybové hlášky v případě nevalidního vstupu. Naimplementovány jsou dvě varianty skenování obrazu, a to varianta statická a adaptivní. Na základě vstupního parametru je také možno sepnout a aplikovat tak model v podobně diference sousedních pixelů.

Při kompresi je nejprve otevřen vstupní soubor a načten obsah do vektoru typu *uint8_t*. Dále je zkontrolováno, zdali je program spuštěn v módu adaptivního nebo statického skenování. V případě statického je komprese prováděna najednou nad celým načteným souborem v horizontálním směru. V případě adaptivního je zadaný obraz rozdělen na menší bloky, přičemž je každý blok komprimován jak ve vertikálním, tak v horizontálním směru. Pro získání matice ve vertikálním směru je horizontální matice transponována. Jaký výsledek je nakonec použit závisí od toho, který směr pro konkrétní matici dává lepší kompresní poměr. Byl-li spuštěn program s přepínačem pro model, je před kompresí na komprimovaný obsah aplikován model. Bloky u adaptivního skenování mají standardní velikost 32x32. V případě, že obrázek není dělitelný velikostmi bloků beze zbytku, je možné, že krajní matice budou mít rozměry menší. Velikost 32x32 byla zvolena na základě provedených experimentů. Vyzkoušeny byly i velikosti 8x8 nebo 16x16. Ty však dávaly horší výsledky.

Samotná komprese probíhá v následujících krocích. Nejprve jsou pro načtený obsah vstupního souboru spočítány frekvence výskytů jednotlivých symbolů, které soubor obsahuje. Na základě těchto frekvencí je naprogramován algoritmus Hirschberg-Sieminski, který byl zmiňován v jedné z přednášek v tomto kurzu.

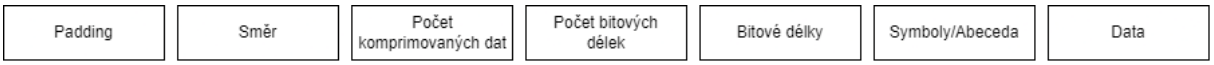
Algoritmus funguje tak, že s pomocí frekvencí výskytů symbolů jsou spočítány délky kódových slov. Pro implementaci tohoto algoritmu bylo využito datové struktury ze standardní knihovny *std::heap*. Jakmile jsou správně získány délky kódových slov, je možno sestavit Kanonický Huffmanův kód. Tento kód je variantou Huffmanova kódování, která má několik výhod oproti klasickému Huffmanovu kódu. Zatímco klasický Huffmanův kód vytváří optimální prefixový kód pro daný soubor dat na základě statistiky výskytu symbolů, kanonický Huffmanův kód přidává další úroveň optimalizace tím, že definuje jednotný způsob reprezentace stromu kódování. Před sestavením Kanonického Huffmanova kódu je nezbytné symboly nejprve seřadit podle vypočítané délky kódových slov. Poté je možné kanonický kód sestavit. Sestavení kanonického kódu je realizováno dle algoritmu vysvětleném na přednášce. Jakmile je pro všechny symboly, jenž se v souboru vyskytují, nalezen jejich kanonický kód je prováděna komprese. Postupně je procházen načtený obsah vstupního souboru a pro každý symbol je do výstupního souboru uložen jeho kanonický kód. Každý komprimovaný soubor je opatřen hlavičkou, která nese informace potřebné pro dekompresi. U adaptivně komprimovaného souboru nese každý komprimovaný blok svoji hlavičku.

Hlavička u statického přístupu skenování obrazu má následující formát:



Obrázek č. 1: Formát hlavičky u komprese se statickým skenováním obrazu

Hlavičky každé matice u adaptivního přístupu skenování mají následující formát:



Obrázek č. 2: Formát hlavičky u komprese s adaptivním skenováním obrazu pro každou matici

U adaptivního varianty je navíc na začátku souboru globální hlavička, která nese informace o šířce a výšce původního souboru.

Při dekompresi je nejprve otevřen a načten obsah ze vstupního souboru. Podle toho, v jakém režimu byl původní soubor komprimován, je důležité spouštět dekomprimaci se stejnými parametry. Tedy i zde je zkontrolováno, zdali se jedná o adaptivní přístup. Pokud ne, a je dekomprimován soubor, který byl komprimován statickým průchodem, je načtený obsah dekomprimován najednou a výsledek uložen do výstupního souboru. Pokud ano, je z prvních 4 bajtů přečtena šířka a výška původního souboru. Dále je obsah předán dekompresoru, který po dílčích částech dekomprimuje celý soubor. Dekomprimovaný obsah je nutné rekonstruovat do původního sestavení, aby byl obraz totožný jako před komprimací. Výsledek je nakonec zapsán do výstupního souboru.

Dekomprimace jako taková probíhá následujícím způsobem. Nejprve je zpracována hlavička s důležitými informacemi. Na základě těchto informací je pomocí algoritmu z přednášky provedena dekomprimace. Je využito dvou vektorů *first_code* a

first_symbol. U adaptivní varianty je dekomprimace prováděna pro každou matici jednotně. U každé matice je využíváno třídy *Submatrixinfo*, která dopočítává a poskytuje údaje o očekávané šířce a výšce konkrétně zpracovávané matice. Ty matice, které byly komprimovány ve vertikálním směru jsou zpětně transponovány. Byla-li dekomprimace spuštěna s přepínačem pro model, je na dekomprimovaný výstup aplikován model. Výsledek je nakonec uložen do výstupního souboru.

Experimenty

Následující tabulka obsahuje výsledky z požadovaných měření aplikace. U obrazů, nad kterými byly provedeny testy je spočítána entropie zdroje. Dále jsou uvedeny časy výpočtu a efektivita komprese pro všechny varianty přepínačů. Časy jsou v jednotkách vteřin. V buňkách pro jednotlivé varianty přepínačů lze vidět počet bitů výsledného komprimovaného souboru a efektivitu v bitech na symbol (bps). Veškerý vývoj i testování bylo prováděno na školním serveru **merlin**. Časy běhu byly změřené pomocí nástroje *time*.

Název	Entropie	-- bps	-m bps	-a bps	-a -m bps	Čas -- Čas -m	Čas -a Čas -a -m
df1h.raw	8	262410 8.00	32772 1.00	174596 5.32	35972 1.09	0.871 s 0.219 s	0.856 s 0.390 s
df1hvx.raw	4.5139	150096 4.58	60180 1.83	120452 3.67	52208 1.59	0.452 s 0.241 s	0.607 s 0.413 s
df1v.raw	8	262410 8.00	32773 1.00	174596 5.32	36164 1.10	0.814 s 0.245 s	0.874 s 0.338 s
hd01.raw	3.8255	127140 3.88	111607 3.40	135043 4.12	120071 3.66	0.427 s 0.353 s	1.158 s 1.000 s
hd02.raw	3.6359	121401 3.70	109196 3.33	131011 4.02	119238 3.63	0.353 s 0.357 s	1.301 s 0.996 s
hd07.raw	5.5769	183921 5.61	126375 3.85	155199 4.73	117634 3.58	0.496 s 0.368 s	1.052 s 0.758 s
hd08.raw	4.2108	138826 4.23	115549 3.52	129110 3.94	112033 3.41	0.341 s 0.315 s	0.927 s 0.855 s
hd09.raw	6.6211	218231 6.65	153539 4.68	199861 6.09	148354 4.52	0.556 s 0.437 s	1.389 s 1.046 s
hd12.raw	6.1657	203171 6.20	143901 4.39	181659 5.54	133805 4.08	0.613 s 0.395 s	1.449 s 0.966 s
nk01.raw	6.4729	213276 6.50	198883 6.06	217071 6.62	197930 6.04	0.513 s 0.476 s	1.329 s 1.287 s

Tabulka č. 1: výsledky měření

Návod

Před spuštěním aplikace je nutné daný program nejprve přeložit. K tomuto účelu je předpřipraven soubor Makefile. Překlad je možno provést pomocí příkazu *make*. Příkaz *make clean* provede smazání objektových souborů vytvořených při překladu a spustitelného souboru.

Spuštění programu se provádí z příkazového řádku pomocí příkazu:

./huff_codec [-c|-d] [-a] [-m] [-i {název_souboru}] [-o {název_souboru}] -w {šířka (int)}

Např. chceme-li provést komprimaci pomocí adaptivního průchodu s modelem nad vstupním souborem *hand.raw* , který má rozměry *232x344* a výstup chceme uložit do souboru *hand_acm.raw* použijeme následující příkaz:

./huff_codec -c -a -m -w 232 -i hand.raw -o hand_acm.raw

Závěr

Úspěšně byl naimplementován program v jazyce C/C++, který nad vstupním souborem s obrazovými daty v RAW formátu provede komprimaci a dekomprimaci. Ty je možno provádět ve dvou variantách skenování – statické a adaptivní. U obou variant je možno využít aplikace modelu.

Projekt mi osobně nepřišel příliš jednoduchý a řadím jej k těm složitějším projektům, se kterými jsem se během studia setkal. O to víc jej však vnímám jako přínosnější a obohacující. Celkově byla látka a naimplementované algoritmy velmi zajímavé. Své znalosti a schopnosti programování v C/C++ jsem určitě zdokonalil a rovněž nabyté zkušenosti z oblasti komprese a kódování dat beru jako velké plus.