Bezztrátová komprese dat

M. Hubička[1], L. Madron[1], J. Rajdl[2], D. Šmíd[3]

1 Gymnázium T. G. M. Hustopeče
2 Gymnázium Ledeč nad Sázavou
3 Gymnazium Bučovice

amfet@wo.cz, josef.rajdl@tiscali.cz

Abstrakt:

Cílem tohoto miniprojektu bylo seznámit se se základními typy komprese a se širokými možnostmi jejich použití. Také jsme chtěli otestovat jejich efektivitu v porovnání s komerčně používanými programy. Po odzkoušení všech metod na různých souborech (opakující se znaky, struktura, velikost) můžeme konstatovat využití různého kódování na různé typy souborů.

1 Metody pro bezztrátovou kompresi dat

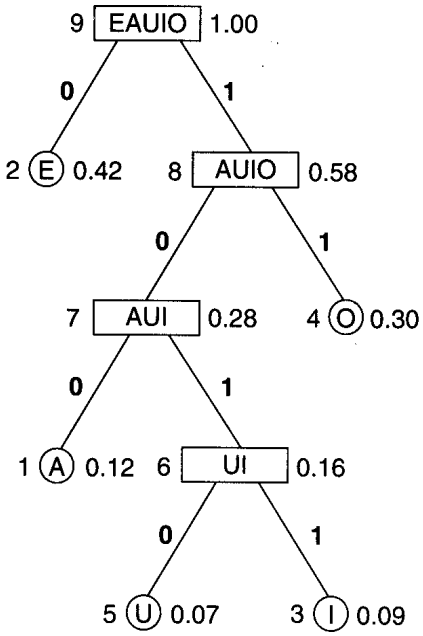
V dnešní době se prakticky setkáváme především se dvěma typy metod pro bezztrátovou kompresi dat. Jsou to:

- Statistické metody, které využívají pravděpodobností vyskytu znaků v souboru:
  - Huffmanovo kódování
  - Aritmetické kódování
- Slovníkové metody, které vytvářejí indexovaný slovník opakujících se částí kódu:
  - LZ 77
  - LZ 78, LZW

Huffmanovo kódování

Základem je nahrazení znaku (obvykle 8 bitů) kódem o určitém počtu bitů. K tomu se využívá pravděpodobnosti výskytu znaku v souboru. Vytváříme tedy schéma, ve kterém znakům s nejmenší pravděpodobností výskytu přiřazujeme kratší kódy než znakům s největší pravděpodobností. Příklad schématu je uveden na obrázku 1. Z tohoto schématu můžeme shora přečíst binární kódy nahrazující každý znak obsažený v souboru.

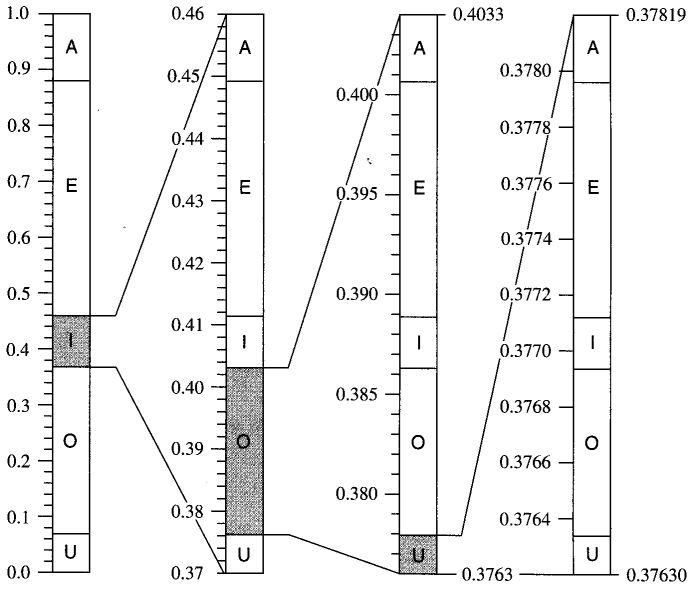
Dnes se Huffmanova komprese využívá v některých komunikačních protokolech a je součástí ztrátové komprese JPEG.



Obr. 1. Schéma kódování znaků v Huffmanově kódování
Kódy znaků čteme shora po větvích např. U = 1010 (pravděpodobnost výskytu 0.07, nejdelší kód), nebo E = 0 (pravděpodobnost výskytu 0.42, nejkratší kód).

Aritmetické kódování

Myšlenka aritmetického kódování spočívá v reprezentaci vstupního řetězce reálným číslem R pro které platí $0 \leq R < 1$. V závislosti na délce vstupního řetězce se zvyšuje počet desetinných míst, která jsou pro zakódování řetězce potřeba. Princip kódování je patrný z obrázku č. 2. Předpokládejme, že vstupní řetězec je složen pouze ze samohlásek a četnost jejich výskytu je znázorněna v prvním sloupci. Počáteční interval si rozdělíme v poměru četností výskytu znaků. Každý znak je tedy reprezentován určitým menším podintervalem. Poté v závislosti na aktuálním znaku vybereme jeden podinterval, který opět rozdělíme v poměru četností a načteme další znak. Tímto postupem se interval neustále zužuje. Z posledního intervalu vybereme jedno reprezentující číslo, jehož zápis představuje zkomprimovaný řetězec.



Obr. 2. Princip aritmetického kódování – postupné zužování intervalů

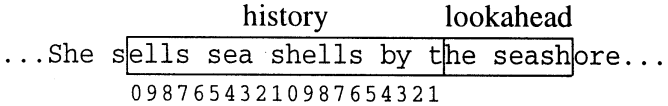
LZ77

Mezi nejznámější slovníkové metody patří metoda LZ 77, která se často používá právě u dobře známých počítačových kompresních programů (ZIP, Rar, …), které tuto metodu samozřejmě kombinují s dalšími metodami a vylepšeními čímž dosahují velmi dobrých kompresních poměrů.

Princip metody spočívá v postupném prohledávání celého souboru tak, že vždy část rozdělíme na dvě „okénka“, kde první tvoří historii kterou prohledáváme a druhým okénkem se dívám dopředu a hledám zda v něm není posloupnost znaků, která se už jednou vyskytuje v okénku historie. Pokud takovou posloupnost najdu, nahradím ji uspořádanou dvojicí:

(offset o kolik znaků jdu zpět, délka sekvence)

Např. zápis (10,2) znamená že mám jít o deset znaků zpět a vzít odsud 2 znaky – viz. schéma



Při konečném zápisu zkomprimovaných dat potom rozlišuji zápis samotného písmena a odkazu do minulosti použitím tzv. identifikačního bitu.

Identifikační bit		celkem
0	písmeno (8 bitů)	9 bitů
1	12 bitů offset + 4 bity délka	17 bitů

LZW

Slovníková metoda LZW funguje, jak už název napovídá, na principu tvorby slovníku, do kterého se ukládají opakující se znaky. Jakmile se znaky objeví v souboru znovu, jsou okamžitě nahrazeny číslem, odkazující na ony znaky ve slovníku. Důležité je ještě poznamenat, že slovníky se neukládají do zkomprimovaného souboru (jsou většinou velké a komprimace by neměla smysl), nýbrž jsou dělány tak důmyslně, že se při dekódování tvoří znovu ze zakódovaných souborů.

Zakódování textu: /wed/we/wee/web/wet

Vs	Vý	Slov.	Vs	Vý	Slov.
/	-	-	E	5	7=/wee
W	/	1=/w	/	-	-
E	W	2=we	W	6	8=e/w
D	E	3=de	E	-	-
/	D	4=d/	B	2	9=web
W	-	-	/	B	10=b/
E	1	5=/we	W	-	-
/	E	6=e/	E	-	-
W	-	-	T	5	11=/wet
E	-	-	EOF	T	-

Zakódovaný text pak vypadá: /wed(1)e(5)(6)(2)b(5)t

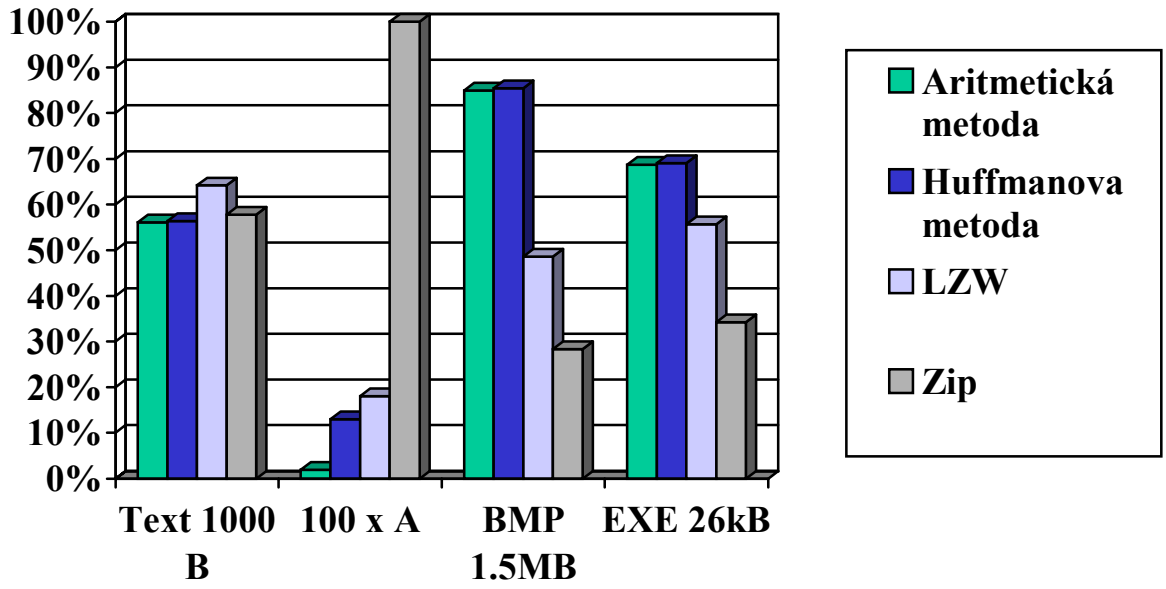
Vs-vstup, Vý-výstup, Slov.-slovník

2 Srovnání jednotlivých metod

Jednotlivé typy kompresí byly testovány na čtyřech typech souborů:

Anglický textový soubor	1000 B
Soubor s opakujícím se znakem	100 B
Obrázek formátu .BMP	1.5 MB
Spustitelný soubor	26 kB

Výsledky jsou zobrazeny v grafu.



Poděkování

Poděkování za finanční a materiální podporu Katedře matematiky FJFI, za konzultace Mgr. Františkovi Gemperlemu, Ph.D.

Reference:

[1] NELSON, M:. *LZW Data Compression* Dr. Dobb's Journal, 1989.

[2] BOZDĚCH, F. *Bezpečné zobrazování reliéfu železniční stanice* KM FJFI ČVUT, 2002.

[3] HANKERSON, D. – HARRIS, G.A. – JOHNSON, P.D.: *Introduction to Information Theory and Data Compression* CRC Press LLC, 1998.

[4] TEUKOLSKY, W.H. – VETTERLING, W.T. – FLANNERY, B.P.: *Numerical Recipes in Fortran* Cambridge University Press, 1992.