METODY KOMPRESE DAT

Petr Hanáček

Motto: "Omit needless words! Omit needless words! Omit needless words!"
Will Strunk, Jr.

Klíčová slova: KOMPRESE DAT, LZW, RLE, HUFFMANOVO KÓDOVÁNÍ, ODSTRAŇOVÁNÍ REDUNDANCE

1. Úvod

Komprese dat je proces, pomocí kterého zakódujeme zprávu S1 takovým způsobem, že výsledná zpráva S2 je (co do počtu bytů) kratší. Původní zprávu S1' pak dostaneme inverzním postupem, který se nazývá dekomprese. Zpravidla požadujeme, aby zpráva S1' byla totožná se zprávou S1, tj. aby kódování bylo transparentní. Tento požadavek splňují všechny popsané algoritmy vyjma algoritmu kosinusové transformace.

2. Charakter dat

Účinnost kompresních algoritmů výrazně závisí na charakteru komprimovaných dat. Pro naše potřeby si rozdělíme data na následující skupiny:

- **Textová data** - obecné texty, zdrojové texty programů, databázové soubory, soubory tabulkových kalkulátorů. Často obsahují posloupnosti stejných symbolů (např. několik mezer po sobě) a skupiny symbolů, které se často opakují (např. často používaná slova). Jedním symbolem je zpravidla jeden byte.
- **Binární data** - spustitelné programy, knihovny v binární formě, binární soubory. Někdy obsahují posloupnosti stejných symbolů.Jedním symbolem je jeden byte.
- **Čárová grafická data** (*line art images*) - obrazy vytvořené uměle na počítači programy typu DRAW nebo PAINT nebo černobílé obrazy sejmuté scannerem. Obsahují dlouhé posloupnosti stejných symbolů (jednobarevné plochy) a opakující se posloupnosti podobných symbolů (jednotlivé řádky obrazu se od sebe liší málo). Jedním symbolem je jeden bit.

- **Obrazy ve stupnici šedi** (gray scale images) - obrazy (černobílé nebo barevné) sejmuté scannerem nebo televizní kamerou s více odstíny. Obsahují posloupnosti mírně se od sebe lišících symbolů. Jednotkou informace je zpravidla jeden byte.

3. Rozdělení kompresních algoritmů

Všechny kompresní algoritmy pracují na principu snižování redundance (nadbytečnosti) komprimovaných dat. Např. běžný text v anglickém jazyce obsahuje 70-80% redundantní informace. (Viz známý pokus Claude Shannona ([2], [4]) který předčítal posluchačům neznámý text a někde uprostřed textu přestal číst. Posluchači měli uhádnout písmeno, které bude v textu následovat. V 60 případech z 90 posluchači správně uhádli následující písmeno - z toho Shannon odvodil, že dvě třetiny písmen v textu jsou zbytečné.)

Podle způsobu zmenšování redundance se kompresní metody dělí na slovníkové (dictionary based) a **statistické.**

Slovníkové metody využívají toho, že se ve zprávě vyskytují vícekrát stejné skupiny symbolů (např. v textu stejná slova). Takovou skupinu symbolů stačí přenést pouze při jejím prvním výskytu a při dalších výskytech se na ni pouze odkážeme. Obecně slovníkové metody nahrazují skupiny symbolů symbolem jediným a nepožadují znalost statistických vlastností (frekvence jednotlivých symbolů, podmíněná pravděpodobnost výskytu určitého symbolu) vstupní zprávy. Naproti tomu **statistické metody** kódují každý symbol vstupní zprávy na jeden symbol výstupní zprávy, jehož délka závisí na pravděpodobnosti tohoto symbolu. Symboly s velkou pravděpodobností jsou zakódovány malým počtem bitů, symboly s malou pravděpodobností velkým počtem bitů. Pravděpodobnost následujícího symbolu ve zprávě závisí jednak na frekvenci jednotlivých symbolů, jednak na kontextu (na několika předcházejících symbolech). Podle délky uvažovaného kontextu se určuje **řád statistické metody.** U metod řádu 0 pravděpodobnost závisí pouze na frekvenci symbolů, u metod řádu 1 závisí navíc na jednom předchozím symbolu atd.

Mimo tyto dvě skupiny kompresních algoritmů se používají také metody kombinované, které spojují výhody obou principů.

4. Metody statistické

Nejznámější statistickou metodou je **Huffmanovo kódování.** je to metoda řádu 0. Její princip spočívá v tom, že symboly vstupní abecedy seřadíme podle jejich frekvence ve zprávě a přiřadíme jim kódy různé délky - symbolům s největší čet-

ností kód nejkratší. Postup je popsán např. v [3]. Musí však platit, že žádný kód netvoří začátek jiného kódu. Na Obr. 1 je příklad pro vstupní abecedu se čtyřmi symboly x1-x4, které mají pravděpodobnosti výskytu P. Např. symbolu x1 s pravděpodobností 0.6 přísluší kód o délce 1 bit. Pro dostatečně dlouhou zprávu dosáhneme komprese na 80% původní délky. Dekomprese se provádí za pomoci dekódovacího stromu (Obr. 1), ve kterém každému symbolu původní zprávy odpovídá jeden list. Huffmanovo kódování není vhodné pro kompresi krátkých souborů, neboť s každým komprimovaným souborem musíme přenášet i dekódovací strom o velikosti 0.5 až 1 KB.

Variantou Huffmanova kódování je kódování pomocí **Shannon-Fanovy** metody. Rozdíl spočívá v jiném způsobu vytváření značek kódu (viz [3]) a v tom, že není třeba přenášet celou strukturu dekódovacího stromu, ale pouze informace o délkách jednotlivých značek.

Aritmetické kódování je podobné Huffmanovu kódování, rozdíl spočívá v tom, že každý vstupní symbol kódujeme na symbol o různém, ale ne celistvém počtu bitů. Je to umožněno tím, že celá vstupní zpráva je zakódována jako jediné číslo v pohyblivé řádové čárce v intervalu $0 <= x < 1$. Postup kódování ukažme na příkladu. Máme zprávu "AX:=BL+CH;". Nyní vytvořme tabulku pravděpodobnosti výskytu jednotlivých znaků a interval $0 <= x < 1$ rozdělme mezi těchto 9 znaků tak, že znak s větší pravděpodobností výskytu dostane větší subinterval (viz. Tab. 1). Nyní se postupuje podle následujícího algoritmu:

```
Dolní:=0;
Horní:=1;
while jsou_na_vstupu_symboly do
   begin
   Interval:=Horní-Dolní;
   Horní:=Dolní+Interval*Symbol_Horní(symbol);
   Dolní:=Dolní+Interval*Symbol_Dolní(symbol);
   end;
Dej_na_výstup (Dolní);
```

Pro naši zprávu bude obsah proměnných Horní a Dolní následující - viz. Tab. 2. Akce zapsaná v algoritmu jako Dej_na_výstup(Dolní) znamená, že vybereme z intervalu <Horní,Dolní) takové číslo, které se zakóduje nejmenším počtem bitů a to pošleme jako výstupní zprávu. Algoritmus dekódování je inverzní:

```
Získej_ze_vstupu (Číslo);
repeat
  Pošli_na_výstup(symbol, do jehož intervalu spadá Číslo);
  Interval:=Symbol_Horní(Symbol)-Symbol_Dolní(Symbol);
  Číslo:=Číslo-Symbol_Dolní(Symbol);
  Číslo:=Číslo/Interval;
until Symbol je znak konce zprávy;
```

Metoda sama o sobě nepozná konec zprávy, proto je třeba přidat na konec vstupní zprávy nějaký speciální symbol nebo přenášet i délku originální zprávy.

Aritmetické kódování ve své původní podobě je v praxi nepoužitelné, neboť bychom museli provádět operace s čísly v pohyblivé řádové čárce s obrovskou přesností. Byl však nalezen způsob, jak implementovat tuto metodu pomocí aritmetiky v pevné řádové čárce a s pevnou délkou čísla. Popis implementace (viz [1], [2]) by však přesahoval rámec tohoto příspěvku.

Doposud jsme popisovali statistické metody řádu 0. V praxi jsou používány také **metody vyšších řádů.** Je však třeba si uvědomit, že paměťová náročnost M algoritmu vyššího řádu je $M = n^{(O+1)}$ kde n je počet symbolů vstupní abecedy a O je řád algoritmu. Například na počítači IBM PC lze pod operačním systémem MS-DOS používat algoritmy řádu 1 (pro textové soubory i řádu 2), při použití rozšířené paměti nebo pod operačním systémem UNIX maximálně algoritmy řádu 3. Nyní ukážeme příklad algoritmu řádu 1, nazývaného také **metoda následníků** (Follower Sets Method). Popíšeme fázi dekomprese, což pro pochopení postačí.

Na začátku zkomprimované zprávy se nachází tzv. tabulka následníků, ve které je pro každý symbol vstupní abecedy (je jich obvykle 256) definována množina jeho nejčastějších následníků (o mohutnosti obvykle 0-64 prvků). Vlastní komprimovaná zpráva sestává z posloupnosti symbolů. Jeden symbol se skládá ze dvou částí - z jednobitového návěští L a z hodnoty V. Je-li L=0, pak to znamená, že V je přímo hodnotou dalšího symbolu ve zprávě (nastal méně pravděpodobný případ, kdy V není v množině následníků aktuálního symbolu). Položka V nebývá obvykle číslo s pevným počtem bitů nýbrž bývá zakódována Huffmanovým kódováním. Povšimněme si i u jiných metod - jakmile přenášíme ve zkomprimované zprávě čísla s nerovnoměrným rozložením (pořadová čísla, délky, offsety) jsou to poten-ciální kandidáti na kompresi Huffmanovou nebo Shannon-Fanovou metodou.

U všech statistických metod jsme potřebovali znát statistiku vstupní zprávy a tato statistika musela být i součástí zkomprimované zprávy. Existují však i modifikace statistických metod, které statistiku znát nepotřebují. Nazývají se metody **adaptivní, dynamické** nebo metody **statistického modelování.** Tyto metody na

počátku kódování předpokládají stejnou pravděpodobnost všech symbolů vstupní abecedy, nezávislou na kontextu. Po každém zpracovaném symbolu zprávy si tabulku pravděpodobností dynamicky upravují, takže v každém okamžiku odpovídá statistice dosud přenesené části zprávy a spoléhají na to, že statistika dosud nepřenesené části zprávy se od ní příliš nebude lišit. Za cenu větších výpočetních nároků (např. u Huffmanovy metody se musí při výraznější změně pravděpodobností změnit dekódovací strom) se adaptivní metody dokázaly zbavit nutnosti přenášet ve zkomprimované zprávě i statistiku. To teprve umožnilo prakticky provozovat metody vyšších řádů (u řádu 2 a vyšších má statistická informace již neúnosně velký objem dat).

Statistické metody jsou vhodné pro kompresi obecných binárních dat. Pro kompresi textových a grafických dat se obvykle kombinují s některou metodou slovníkovou.

5. Slovníkové metody

Nejjednodušší a nejrozšířenější slovníkovou metodou, která vlastně žádný slovník nemá, je metoda **RLE** (Run Length Encoding, někdy překládáno jako metoda kódování běhů). Tato metoda je velmi rychlá a většina složitějších metod bývá s touto metodou kombinována. Její princip (viz Obr. 2.) spočívá v tom, že posloupnost stejných symbolů ve vstupní zprávě je nahrazena speciálním symbolem, který říká, kolikrát se opakuje, následovaným symbolem, který se opakuje. Existuje několik možných řešení, jak zakódovat tento speciální symbol a počet opakujících se symbolů, proto existuje také několik modifikací této metody. Tato metoda může být použita jako doplněk jiné metody pro kompresi textových dat (např. ve zdrojových textech komprimuje odsazovací mezery), ale hlavně se používá pro kompresi čárových grafických dat, kde výhodně komprimuje jednobarevné plochy. V telefaxových zařízeních se obraz komprimuje upravenou metodou RLE (podle doporučení CCITT/3), která kóduje střídavě posloupnosti bílých a černých bodů. Délky posloupností jsou kódovány Huffmanovým kódováním.

Další velmi rozšířenou slovníkovou metodou je metoda **LZW** (Lempel-Ziw-Welch). Vychází z následující myšlenky: co kdybychom každý symbol vstupní abecedy (který se kóduje 8 bity) zakódovali pomocí 12 bitů tak, že symboly 0-255 by odpovídaly původním znakům a ostatní (cca 4000) symboly by vyjadřovaly pořadové číslo ve slovníku se slovy, která se ve zprávě používají nejčastěji. Problémem je, jak vybrat oněch 4000 nejpoužívanějších slov. Použije se tento postup: Kompresní a dekompresní strana si buduje identický slovník z posloupností dosud přenesené zprávy. Ve slovníku nejsou samozřejmě všechny posloupnosti, ale platí, že slovo o N znacích se objeví ve slovníku po N-1 výskytech ve zprávě. Například mějme slovo BEGIN. Po

prvním výskytu je ve slovníku posloupnost BE, po druhém BEG, po třetím BEGI atd. Postup vytvoření slovníku je znázorněn na Obr. 3. pro posloupnost "This is a", která se zakóduje na posloupnost "This <258> a" kde <258> je symbol s hodnotou258.

Po vyčerpání všech možností kódu (po naplnění slovníku) se do slovníku další posloupnosti už nepřidávají. Metoda LZW má některé modifikace pro zvýšení účinnosti. **Dynamická LZW** metoda nemá délku kódu 12 bitů, ale zvětšuje ji dynamicky podle zaplnění slovníku (začíná se s 9-bitovým kódem a ten se zvětšuje až na 13 bitů). **LZW s adaptivním mazáním** (Adaptive Reset) dovoluje po zaplnění slovníku vyslat speciální znak, který znamená vymazání celého slovníku. Od tohoto okamžiku se slovník naplňuje znovu. **LZW s částečným mazáním** (Partial Clearing) při zaplnění slovníku nemaže celý slovník, ale pouze posloupnosti, které se v textu vyskytly méně než dvakrát (pokud si slovník představíme jako strom, odstraňuje listy stromu). Posloupnosti, které se v textu vyskytly víckrát, zůstávají ve slovníku.

Metoda LZW je hodně používaná a je velmi účinná pro kompresi textů, zvláště pro zdrojové texty programů.

Další slovníkovou metodou je metoda s **klouzajícím slovníkem** (Sliding Dictionary, Sliding Buffer, Lempel-Ziw-Storer-Szymanski). Tato metoda si nebuduje žádný explicitní slovník. Jako slovní používá poslední úsek již dekomprimovaného textu (např. o velikosti 4KB) - viz Obr. 4. Přenášený symbol je tvořen buďto bitem s hodnotou 0, za kterým následuje přímá hodnota výstupního symbolu V, který se pošle na výstup, nebo bitem s hodnotou 1, za kterým následuje dvojice offset S a délka L. V tomto druhém případě se na výstup pošle posloupnost symbolů, která v klouzajícím slovníku začíná na offsetu S a má délku L symbolů. Pro zvýšení účinnosti bývají hodnoty S a L (někdy i hodnota V) komprimovány Huffmanovou nebo Shannon-Fanovou metodou. Metoda s klouzajícím slovníkem je vhodná pro textová data a čárová grafická data.

6. Speciální metody

Následující tři metody se používají pro kompresi obrazů ve stupnici šedi a obrazů získaných barevnou kamerou. Metoda **kosinusové transformace** (DCT) je ze zde uvedených metod jediná, která provádí kompresi s jistou ztrátou informace. Princip metody je následující: Obraz rozdělíme na čtvercové bloky f(x,y) (např. 8x8 bodů), které podrobíme dvourozměrné diskrétní kosinusové transformaci (podrobnosti viz [6]). Získáme 64 hodnot F(u,v), z nichž F(0,0) je střední hodnota jasu bloku ("stejnosměrná složka") a ostatní hodnoty F(u,v) jsou "harmonické". Tyto hodnoty zpracujeme vhodným kvantizátorem (zde nastává ztráta informace) a zjistíme, že hodno-

ty F(u,v), pro u,v<>0, jsou zpravidla malá čísla, což využijeme pro zakódování Huffmanovým kódem. Dekomprese je inverzní postup. Tato metoda je zajímavá tím, že jsou komerčně vyráběny koprocesory pro kompresi obrazů tímto způsobem.

Metoda **DPCM** (Differential Pulse Code Modulation) využívá faktu, že v obrazech ve stupnici šedi se zpravidla hodnota pixelu příliš neliší od hodnot okolních pixelů a je tedy možno předpovědět hodnotu pixelu Px,y podle hodnot pixelů již přenesených (viz Obr. 5, Hodnota pixelu Px,y se předpoví z hodnot pixelů Px-1,y-1, Px,y-1, Px+1,y-1, Px-1,y). V případě, že na kompresní i dekompresní straně máme shodný prediktor (viz Obr. 6.) můžeme místo hodnoty každého pixelu přenášet pouze odchylku mezi hodnotou pixelu a hodnotou předpovězenou. Přenášíme sice stejný počet hodnot, kolik je pixelů, ale odchylky jsou poměrně malá čísla, která můžeme výhodně komprimovat Huffmanovým kódováním.

Metoda komprese dat s použitím **Hilbertovy křivky** je modifikací předchozí metody. Rozdíl je v tom, že u klasických metod přenosu obrazu jsou přenášeny posloupností pixelů po řádcích (tak jako v televizním signálu). Tato metoda přenáší posloupnosti pixelů podle tzv. Hilbertovy křivky (viz Obr. 7.). Tato křivka patří mezi fraktály a má tu vlastnost,že dokáže vyplnit plochu. Výhodou této křivky je to, že pixely v posloupnosti dané touto křivkou spolu "blíže sousedí", tedy jasové změny mezi nimi budou menší, odchylky prediktoru budou také menší a komprese dat účinnější.

7. Účinnost komprese dat

Různé testy a statistiky účinnosti kompresních algoritmů nejsou příliš objektivní, neboť účinnost komprese nezávisí pouze na použitém algoritmu, ale také na délce komprimovaných dat (kratší soubory se zkomprimují méně, delší více) a charakteru komprimovaných dat (na jejich redundanci). Přesto uvedu alespoň orientační údaje. Soubory textového typu (obecné texty, zdrojové texty programů, databázové soubory) se komprimují na 30-60% původní délky, binární soubory na 50-90% původní délky. U obrazových dat závisí účinnost na typu obrazu a jeho obsahu.

8. Závěr

Tento přehled algoritmů pro kompresi dat samozřejmě není a nemůže být úplný. Stále vznikají nové algoritmy a jejich modifikace s cílem zvýšit účinnost komprese. Přestože se kapacity paměťových médií stále zvětšují a jejich ceny klesají, komprese dat má stále své opodstatnění. Například většina výrobců a prodejců software distri-

buje své produkty v komprimované podobě. Komprese dat se také stala důležitou součástí takových aplikací, jako je zpracování obrazu či telefax.

Literatura

[1] Okumura Haruhiko: Data Compression Algorithms of LARC and LHARC, PV-VAN, 1989

[2] Nelson, M.R.: Arithmetic Coding and Statistical Modeling, Dr. Dobb's Journal, February 1991

[3] Šebesta, V., Vrba, K.:Teorie přenosu zpráv, SNTL Praha, 1980

[4] Shannon, C.E., Weaver, W.: The Mathematical Theory of Communication, Urbana, Illinois, University of Illinois Press, 1949

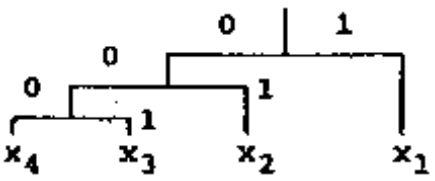
[5] Vivian, R.H.: DPCM Studies using Edge Prediction, Microprocessing and Microprogramming 21, 1987

[6] Image Compression for High-Speed Network Transmission, Circuit Cellar, August/September 1990

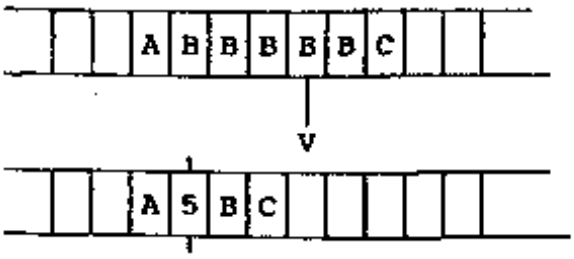
[7] Apiki S.: Lossless Data Compression, BYTE, March 1991

Autor: Ing. Petr H a n á č e k
Katedra informatiky a výp. techniky, FE VUT
Božetěchova 2, 612 66 Brno 12
tel. 05 - 746 111/kl. 31

Symbol	P	Kód
x_1	0.6	1
x_2	0.2	01
x_3	0.1	001
x_4	0.1	000



Obr. 1. Příklad Huffmanova dekódovacího stromu



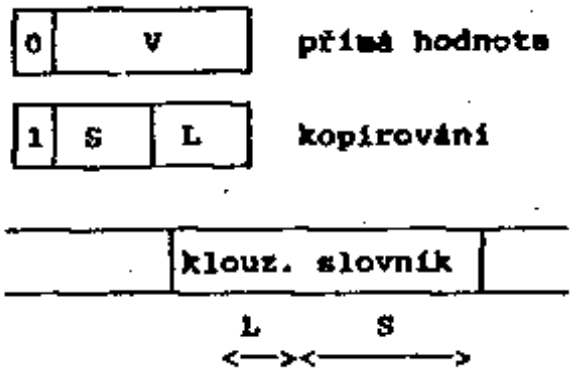
Obr. 2. Příklad komprese RLE

Vstup	Slovník	Výstup
T		
h	256=T+h	T
i	257=h+i	h
s	258=i+s	i
' '	259=s+' '	s
i	260=' '+i	' '
s		
' '	261=258+' '	258
a	262=' '+a	' '
		a

Obr. 3. Příklad metody LZW

Znak	Pravděpodobnost	Interval
B	1/10	0.00-0.10
+	1/10	0.10-0.20
A	1/10	0.20-0.30
H	1/10	0.30-0.40
L	1/10	0.40-0.50
X	1/10	0.50-0.60
=	2/10	0.60-0.80
/	1/10	0.80-0.90
C	1/10	0.90-1.00

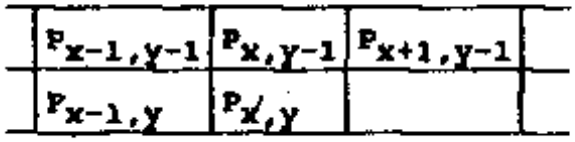
Tab. 1. Rozdělení intervalů mezi znaky



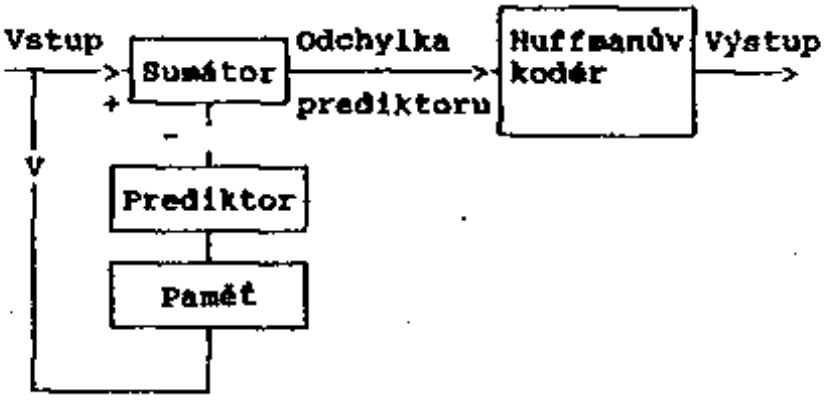
Obr. 4. Klouzající slovník

Znak	Dolní	Horní
	0.0	1.0
A	0.2	0.3
X	0.25	0.26
=	0.256	0.258
=	0.2572	0.2576
B	0.25720	0.25724
L	0.257216	0.257220
+	0.2572164	0.2572168
C	0.25721676	0.2572168
H	0.257216772	0.257216776
/	0.2572167752	

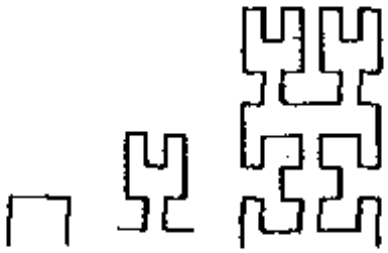
Tab. 2. Postup aritmetického kódování



Obr. 5. Predikce hodnoty pixelu



Obr. 6. Princip metody DPCM - komprese



Obr. 7. Fáze vytváření Hilbertovy křivky