Fakultet elektrotehnike i računarstva

Zavod za elektroničke sustave
i obradbu informacija

Lempel – Ziv vizualizacija

Autori: Petar Juroš
Mario Srbiš
Ivor Grubišić

Siječanj, 2006.

1. SADRŽAJ

1. SADRŽAJ..... 2
2. UVOD..... 3
3. OPIS..... 4
    3.1. PRIMJER LEMPEL-ZIV KODIRANJA.....5
    3.2. LEMPEL-ZIV-WELCH KODIRANJE..... 8
4. ZAKLJUČAK..... 9
5. LITERATURA..... 10

2. UVOD

Od pojave prvih računala susrećemo se sa jednim problemom koji nas prati do danas. To je problem oganičene memorije. Sa vremenom količina memorije u računalima se povećava, ali ostaje činjenica da je ona uvijek ograničena i prilikom korištenja računala to uvijek treba imati na umu. Programeri moraju omogućiti da sa tim ograničenim resursima računala obrađuju velike količine podataka. Sasvim prirodno je da su se počele javljati razne ideje kako što učinkovitije smjestiti podatke u memoriju i jedna od njih je kompresija podataka.

Kompresiju podataka možemo podijeliti u dvije kategorije. Prva kategorija je ona gdje dolazi do gubitka originalnih informacija, ali je ušteda u memorijskom prostoru tolika da su ti gubici prihvatljivi. U engleskoj terminologiji to se naziva “lossy data compression”'. Takvu kompresiju primjenjujemo kod podataka gdje možemo dozvoliti određeni gubitak informacija, kao zvuk, video i slike. Dio informacija mijenjamo za memorijsku uštedu koja nam je znatno važnija nego sami podaci. Naravno, postoje i podaci kod kojih gubitak informacija nije prihvatljiv i gdje nakon dekompresije moramo imati identične podatke početnima. U engleskoj terminologiji takva vrsta kompresije se naziva “lossless data compression”.

Razvijanjem učinkovitih vrsta kompresije zadovoljavamo rastuće potrebe za memorijskim prostorom bez potrebe za razvijanjem novih tehnologija za izradu same memorije. Razvoj industrija kao što su filmska ili muzička bio bi nezamisliv bez upotrebe neke vrste kompresije. Potreba za memorijskim kapacitetima uvijek će postojati, a kompresija podataka je tu potrebu učinila znatno manjom.

3. OPIS

Do kraja 70-tih godina prošlog stoljeća istraživanja na polju kompresije podataka bila su usmjerena uglavnom na poboljšavanje Huffman algoritma kodiranja podataka. Kako se taj algoritam temelji na određivanju frekvencije ponavljanja određenih simbola bilo je potrebno te simbole predvidjeti ili unaprijed pročitati podatke. Dva Izraelska teoretičara informacije Abraham Lempel i Jacob Ziv 1977. godine predstavljaju radikalno drukčiji pristup kompresiranju podataka. Kompresiju kod koje nije potrebno predviđati frekvenciju ponavljanja simbola i bespotrebno unaprijed čitati podatke.

Osnovna ideja Lempel-Ziv kodiranja podataka je kreirati riječnik od podataka koji su procesirani i zamjena simbola (ili cijelog niza simbola) sa indeksima iz toga riječnika. Tako u kompresiranom nizu podataka pohranjujemo samo pokazivače prema frazama koje se nalaze u riječniku. Sama metoda kreiranja riječnika u Lempel-Ziv kodiranju je razlog što je ova vrsta kodiranja vrlo učinkovita. Svoju metodu Lempel i Ziv su predstavili u dvije varijante koje su poznatije pod imenima LZ77 i LZ78. Iz tih medota kompresiranja razvile su se razne podvarijante koje unosne neznatne promjene u sam algoritam i uglavnom opisuju tehničku stranu realizacije algoritma u praksi.

LZ77 varijante	LZR	LZSS	LZB	LZH		
LZ78 varijante	LZW	LZC	LZT	LZMW	LZJ	LZFG

***Tablica 1** - LZ77 i LZ78 varijante kodiranja*

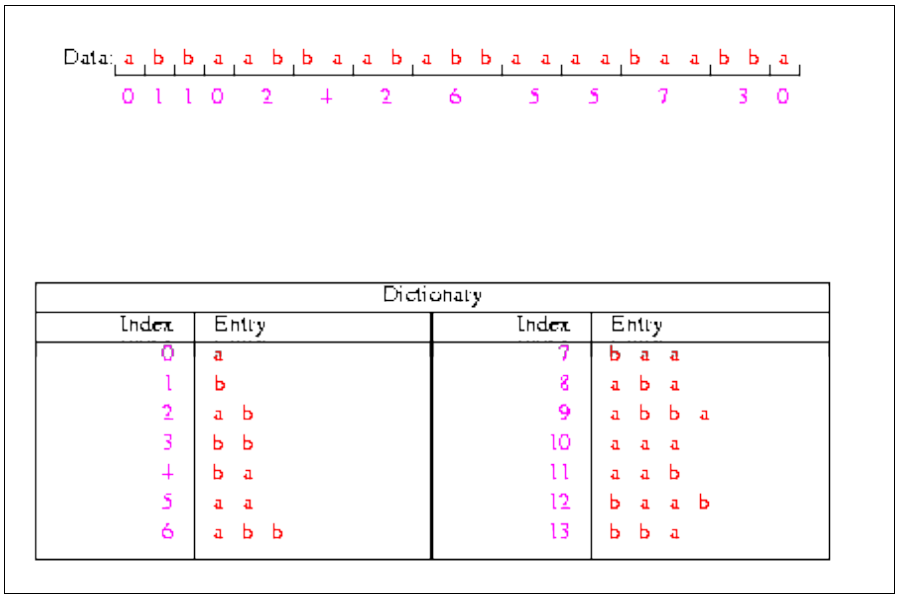
LZ77 algoritmi se uglavnom nalaze u programima koji kompresiraju tekst i opće podatke, dok se LZ78 uglavnom koristi prilikom kompresiranja binarnih podataka, kao što su slike i slično.

3.1. PRIMJER LEMPEL-ZIV KODIRANJA

Lempel-Ziv algoritam kodiranja podataka vrlo je učinkovit kada se upotrebljava na nizovima velike dužine. Na nizovima male dužine teško je prikazati njegovu učinkovitost ako se upotrebljavaju realni nizovi. Zato je u primjeru prikazan vrlo jednostavan niz 'abbaabbaababbaaaabaabba' koji ima puno znakova koji se ponavljaju kako bi kompresija bila učinkovitija.

Sam proces kodiranja može se podijeliti u 5 koraka:

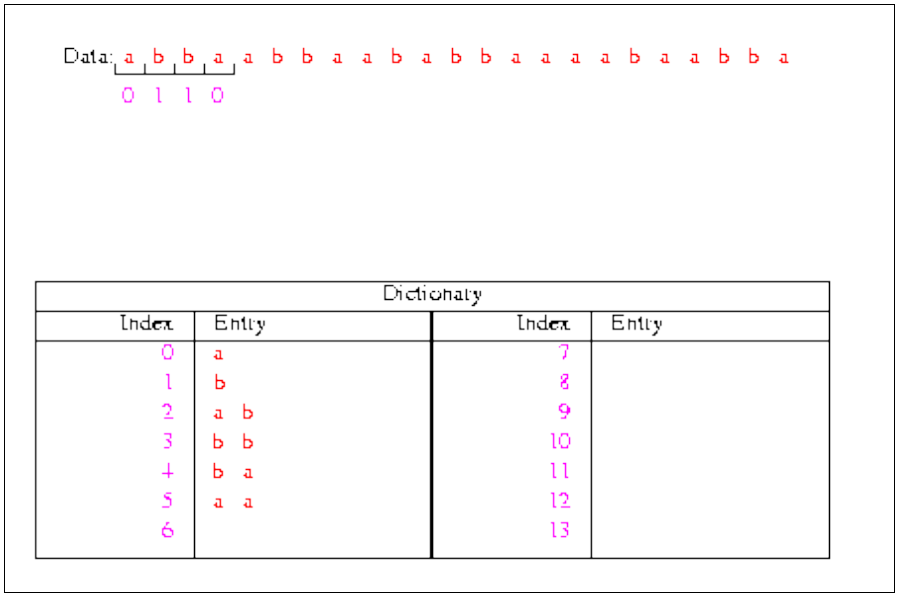
1. Inicijalizacija riječnika. U riječnik se unose svi blokovi dužine 1.
2. Traženje najdužeg bloka W unutar niza kojeg imamo unesenog u riječnik.
3. Zapisati index bloka W u kodirani niz.
4. Dodati novi zapis u riječnik koji se sastoji od bloka W i sljedećeg znaka.
5. Ponoviti korak 2.



Slika 1. *– Lempel-Ziv primjer kodiranja*

Na slici 1. prikazan je niz koji želimo kodirati, rezultat kodiranja i prikaz riječnika koji smo dobili. U prvom koraku smo inicijalizirali riječnik tako da smo unijeli sve blokove dužine 1 i njima dodijelili određene indekse. Tako smo dobili riječnik D={a,b} sa pripadajućim indeksima. Daljni postupak se temelji na zamjeni blokova u nizu koji želimo kodirati sa pripadajućim indeksima iz riječnika. Paralelno sa tom zamjenom nadograđujemo riječnik sa sve složenijim blokovima, tako da nam indeksi predstavljaju blokove sve veće dužine i tako postižemo kompresiju podataka. U našem primjeru bi to značilo da zamjenjujemo prvi blok 'a' sa indexom '0' (tako nam piše u riječniku) i dodajemo

novi zapis u riječnik 'ab' kojemu dodjeljujemo sljedeći indeks '2'. Sada zamjenjujemo blok 'b' sa indeksom '1' i dodajemo novi zapis u riječnik 'bb' sa indexom '3'. Slijedi zamjena bloka 'b' sa indeksom '1' i dodavanje novog zapisa u riječnik 'ba' sa indexom '4', te zamjena bloka 'a' sa indexom '0' i dodavanje novog zapisa u riječnik 'aa' sa indexom '5'. U ovom trenutku imamo situaciju koja je prikazana na slici 2.



Slika 2. *– Lempel-Ziv primjer kodiranja - međukorak*

Sada se događa vrlo bitna stvar u kojoj je sadržana cijela filozofija Lempel-Ziv kodiranja podataka. Prilikom unošenja novih zapisa u riječnik oni postaju sve složeniji i pripadajući indeksi zamjenjuju sve veće blokove podataka. U sljedećem koraku našega primjera zamjenujemo blok 'ab' (koji je dužine 2) sa pripadajućim indeksom '2' i unosimo novi zapis u riječnik 'abb' kojemu dodjeljujemo index '6'. Primjetite da zapisi u riječniku imaju sve veće blokove i da zapisi u riječniku nisu nasumce uneseni nego su to blokovi koji se pojavljuju u nizu koji želimo kodirati. Također je vrlo bitna star i činjenica da se riječnik kreira po točno određenim pravilima, što se pokazuje kao bitna stvar prilikom dekodiranja podataka.

Nakon završetka procesa kodiranja, niz 'abbaabbaababbaaaabaabba' smo zamijenili sa nizom '0110242655730' koji ima znatno kraću dužinu. Dekodiranje se ostvaruje na vrlo jednostavan način, pomoću riječnika zamjenimo indekse sa pripadajućim blokovima podataka i dobijemo polazni niz. Važno je napomenuti da ovo predstavlja samo jednostavan teoretski primjer u kojemu su neke stvari pojednostavljene kako bi se na tako kratkom primjeru pokazao učinak kompresije prilikom kodiranja niza podataka pomoću Lempel-Ziv algoritma.

Teoretski veličina riječnika bi mogla biti beskonačna, međutim u praksi se ona

ograničava i kada se dostigne maksimalna veličina riječnika on se više ne proširuje sa novim zapisima. Također postoji mogućnost da se koristi varijabilno numeriranje indeksa u riječniku i tada govorimo o 'variable-to-variable length' verziji Lempel-Ziv algoritma.

U praksi Lempel-Ziv algoritam kodiranja radi vrlo dobro i dovodi do vrlo velike kompresije podataka pod uvjetom da su ulazni podaci dovoljno veliki i da imaju određeni stupanj redudancije. Mnogi često korišteni programi koriste Lempel-Ziv kodiranje, tako na Unix platformama imamo gzip i gunzip, a na Windows platformama WinZip, Arj, Pkzip i slične programe koje koristimo za kompresiju podataka i koji su vrlo raširen.

U tablici 2. prikazana je usporedba kompresiranja različitih vrsta podataka pomoću adaptivne verzije Huffman kodiranja i Lempel-Ziv kodiranja. Podaci su kompresirani upotrebom programa 'compact' za Huffman kodiranje i programa 'compress' za Lempel-Ziv kodiranje.

Vrsta podataka	Adaptivni Huffman	Lemple-Ziv
Tekst	66 %	44 %
Govor	65 %	64 %
Slika	94 %	88 %

Tablica 2. *– Usporedba Huffman i Lempel-Ziv kodiranja*

3.2. LEMPEL-ZIV-WELCH KODIRANJE

1984. godine Terry Welch je modificirao LZ78 algoritam kodiranja kako bi ga implementirao u vrlo brze diskovne kontrolere, što je rezultiralo nastajanjem Lempel-Ziv-Welch algoritma (poznatijeg kao LZW algoritam) koji je danas vrlo raširen. Ova metoda kompresiranja podataka bez gubitka informacija se koristi u nekoliko formata sa pohranu slika (GIF i TIFF), dio je V.42bis modemskog kompresijskog standarda, te ju nalazimo u PostScript Level 2 formatu.

LZW je algoritam za kompresiju podataka koji može kompresirati bilo koju vrstu podatka, a najznačajnije odlike su mu brzina kompresiranja i dekompresiranja, te ne zahtijeva upotrebu aritmetike sa pomičnim zarezom, pa je to i glavni razlog njegove brzine. Zanimljivo je i to da algoritam daje jednake izlaze i na big-endian i little-endian sustavima.

Kako je nastao od LZ78 algoritma LZW također koristi riječnik kako bi zamijenio veće nizove podatak sa indeksima i tako postigao efekt kompresije. Njegova velika prednost pred ostalim algoritmima koji se temelje na riječnicima, je to da za njegovu dekompresiju nije potrebno prenositi riječnik, već je algoritam tako napravljen da se riječnik može konstruirati iz kodiranih podataka, a to pridonosi značajnom povećanju stupnja kompresije. Početni zapisi riječnika se inicijaliziraju pomoću 8-bitne ASCII tablice koja je jednoznačno definirana, tako da i koder i dekoder kreću od istog početnog riječnika i time eliminiraju potrebu prenošenja riječnika.

4. ZAKLJUČAK

Lempel i Ziv su razvili vrlo jednostavan algoritam, a vrijeme je pokazalo da je to jedan od najučinkovitijih načina kompresiranja podataka koji poznajemo. Njegova primjena je vrlo raširena i danas je gotovo nemoguće naći osobu koja nije barem jednom koristila kompresiju podataka temeljenu na Lempel-Ziv algoritmu. Cijeli niz algoritama i metoda temelji se na osnovnim principima koje je postavio Lempel-Ziv algoritam, a njegov razvoj ne prestaje i svakim danom se nadograđuje sa novim mogućnostima i modifikacijama.

Snaga Lempel-Ziv algoritma nalazi se u njegovoj jednostavnosti i brzini izvođenja što su jedni od glavnih preduvijeta da neka tehnologija postane prihvaćena od strane korisnika. Na jednostavan način ovaj algoritam dokazuje u kojem pravcu treba razvijati nove tehnologije i kako je uz pomoć osnovnih matematičkih operacija moguće razvijati velike stvari.

5. LITERATURA

http://www.data-compression.com/lempelziv.html

Animacija koja na jednostavan način prikazuje Lempel-Ziv kodiranje.

http://www.cs.cf.ac.uk/Dave/Multimedia/node214.html

Detaljan opis Lempel-Ziv-Welch metode kodiranja.

http://www.eee.bham.ac.uk/WoolleySI/All7/ziv_1.htm

Detaljan opis Lempel-Ziv kompresije sa primjerima.

http://en.wikipedia.org/wiki/Lempel-Ziv

Lempel-Ziv kodiranje objašnjeno na jednostavan način.

http://www.compression-links.info/LZW

Za one koji žele znati više ovdje mogu naći iscrpan popis dokumentacije vezane kako za Lempel-Ziv kodiranje tako i za ostale algoritme kodiranja koji se koriste prilikom kompresije podataka.

http://www.compression-links.info/LZW

Upotrijebite moćni Google i pronađite beskrajnu količinu Lempel-Ziv materijala.