Anna Krawet
Paweł Kramarski
Tomasz Krześniak
Grzegorz Mroziewicz
Marcin Ziółek

Warszawa, dn. 15.12.2005r.

Projekt z KODA

Semestr Zima 2006

Temat:

„Kodek słownikowy (własna koncepcja)”

Prowadzący:
Dr hab. inż. Artur Przelaskowski

1. Wstęp

Realizując koncepcję kodeka słownikowego modyfikowaliśmy istniejący algorytm LZW. Modyfikacja nasza polegała na zmianie sposobu tworzenia słownika. Bazując na znanym algorytmie LZW opracowaliśmy 4 rodzaje kodeków słownikowych:

- usuwane są ze słownika najrzadziej używane słowa;
- słownik po przepełnieniu jest całkowicie czyszczony;
- nowe słowa tworzone są z dwóch istniejących słów;
- dynamiczne usuwanie elementów ze słownika;

Każdy z tych kodeków zostanie dokładniej omówiony w dalszej części dokumentacji.

W celach porównawczych przy testowaniu wykorzystaliśmy dostępne, już zaimplementowane, algorytmy: LZW ze stałą długością słowa – 12 bitów, LZW ze zmienną długością słowa – 15 bitów, LZW wraz z kodem Huffmana, LZ 77. Zaimplementowaliśmy również sam algorytm LZW, ale przy zastosowaniu drzewa, dzięki czemu otrzymaliśmy odmienne wyniki szybkości działania, które prowadzi do tego samego celu.

2. Algorytm usuwania najrzadziej używanych słów ze słownika

2.1. Opis algorytmu

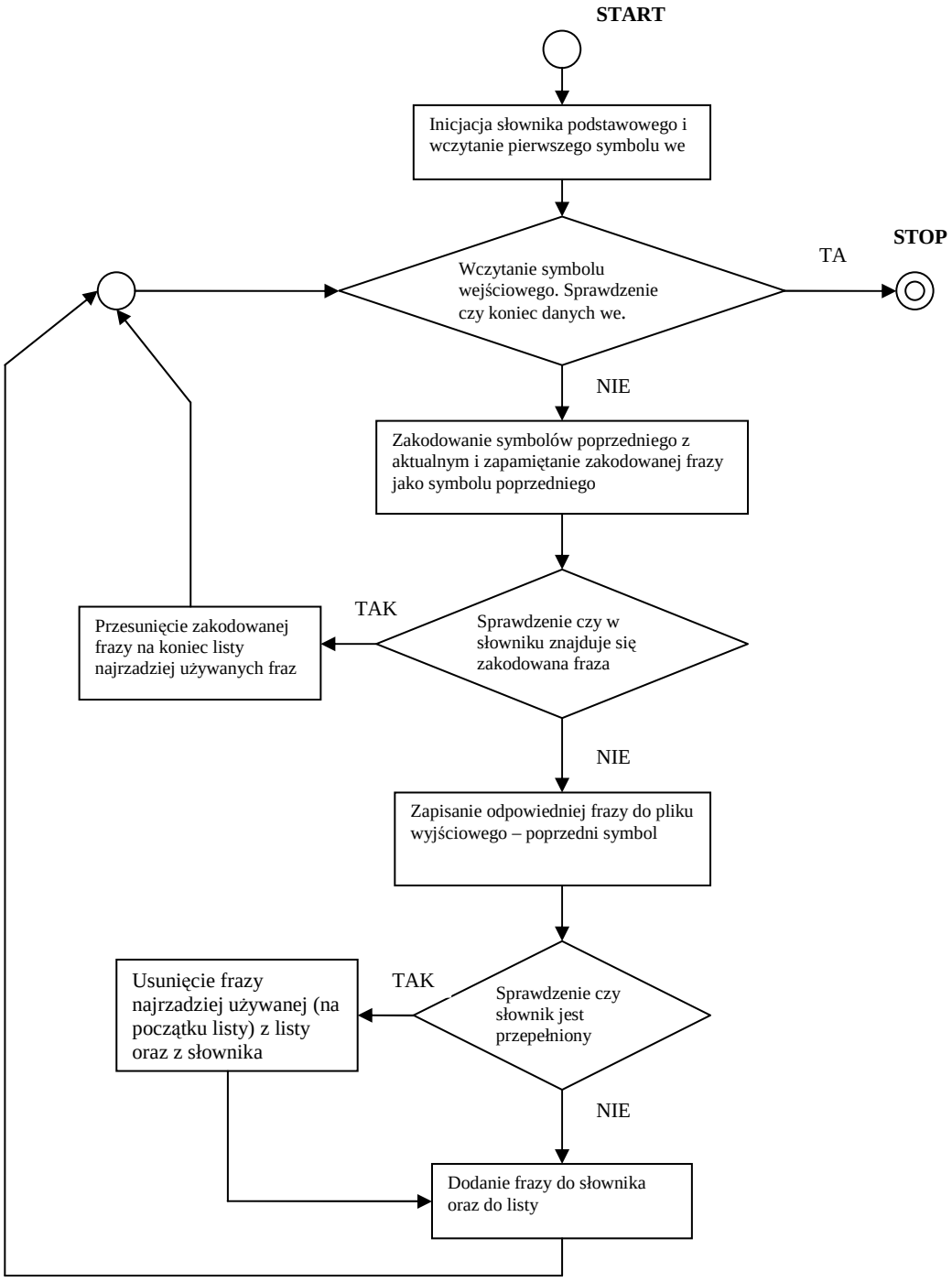
Algorytm ten jest modyfikacją algorytmu LZW (dokładnej: modyfikacją słownika w LZW), który polega na usuwaniu najrzadziej występujących zakodowanych fraz w przypadku przepełnienia słownika.

Algorytm używa dodatkowej listy najrzadziej używanych fraz.

Każda nowo utworzona fraza jest zapisywana jak w LZW do słownika, po czym zapisywana jest na listę. W przypadku powtarzania się danej frazy, zostaje ona przerzucana na koniec listy (frazy które się powtarzają zbiegają na koniec). Dzięki temu frazy, które są najrzadziej używane będą na początku listy. W momencie przepełnienia się słownika, wyszukiwana jest najrzadziej używana fraza, która może

zostać usunięta (nie każda fraza może zostać usunięta, gdyż istnieją powiązania między nimi i dla utrzymania spójności algorytmu usuwana jest nie pierwsza najrzadziej używana tylko odpowiednio wyszukana najrzadziej używana, która może zostać usunięta np.: fraza AB-256, fraza ABA-258 – fraza 256 nie może zostać usunięta, gdyż jest częścią składową frazy 258), a dodawana jest aktualna. Wyszukiwanie takiej frazy zajmuje niewiele czasu.

2.2. Algorytm blokowy kompresji



Na początku następuje inicjalizacja słownika podstawowego – elementów od 0 – 255 oraz wczytanie i zapamiętanie pierwszego symbolu do zakodowania (na schemacie jest założone, że istnieje przynajmniej jeden symbol do zakodowania), po czym następuje wczytanie kolejnego symbolu. Jeśli wczytany symbol oznacza koniec danych algorytm kończy działanie, w przeciwnym przypadku zostaje przeprowadzone kodowanie symboli poprzedniego z aktualnym i zapamiętanie zakodowanej frazy jako symbol poprzedni. Kolejny krok to sprawdzenie czy w słowniku znajduje się zakodowana fraza, jeśli tak, to przesuwamy zakodowaną frazę na koniec listy (najrzadziej używane frazy będą nie przesuwane – będą na początku listy), po czym odczytujemy kolejny symbol do zakodowania. Jeśli zakodowana fraza nie znajduje się w słowniku, zapisujemy poprzednią frazę do pliku wyjściowego, po czym sprawdzamy zajętość słownika. Jeśli słownik nie został przepełniony, dodajemy zakodowaną frazę do słownika i umieszczamy ją na liście, w przeciwnym przypadku przed dodaniem usuwamy frazę najrzadziej używaną z listy oraz ze słownika, po czym przechodzimy do pobrania kolejnego symbolu. Kryterium stopu algorytmu następuje po wczytaniu wszystkich danych symboli wejściowych. Dekompresor będzie działał analogicznie – wykorzystując listę najrzadziej używanych fraz.

2.3. Spostrzeżenia

Przy tak skonstruowanym algorytmie kodek jest efektywniejszy, w przypadku kompresji niż LZW. Dla plików tekstowych np.: o rozmiarze 1MB kompresja dawała wynik ok. 250KB, natomiast LZW ok. 400KB. Lepsza jest kompresja niż w przypadku LZW, ale czas wykonania jest dłuższy.

3. Algorytm czyszczenia słownika po przepełnieniu

Algorytm ten jest prostą modyfikacją algorytmu LZW, nie wymaga większych wyjaśnień. Procedura działania jest analogiczna jak w LZW, z tym, że przy przepełnieniu słownika następuje jego wyczyszczenie, po czym tworzony jest nowy słownik. Ten algorytm również poprawia kompresję w przypadku plików tekstowych.

4. Algorytm tworzenia nowych słów z dwóch już istniejących słów

4.1. Opis algorytmu

Cechy algorytmu:

- słownik po zapełnieniu nie jest już zmieniany,
- nowy element budowany jest z dwóch już istniejących elementów,
- słowa są zapisywane do pliku przy użyciu stałej ilości bitów (od 8 do 16)

Cech implementacji:

- słownik jest pamiętany w postaci drzewa, gdzie każdy element tego drzewa pamięta wartość indeksu innego elementu oraz zna swojego rodzica. Fraza pamiętana w elemencie składa się z frazy pamiętanej przez rodzica i przez element, którego indeks jest zapamiętany,

Zmiany w stosunku do pierwotnego algorytmu LZW:

LZW:

- Pomiędzy kolejnymi iteracjami pętli w pamięci pamiętany jest tylko jeden znak.

Proponowany algorytm:

- Pomiędzy kolejnymi iteracjami w pamięci pamiętany jest indeks elementu w słowniku (który reprezentuje sobą ciąg znaków).

LZW:

- W każdej iteracji czyta plik sprawdzając czy: znak w pamięci + to co przeczytał istnieje w słowniku. Czyta, dopóki taki element istnieje. Jeśli nie istnieje, dodaje ten nieistniejący element do słownika.

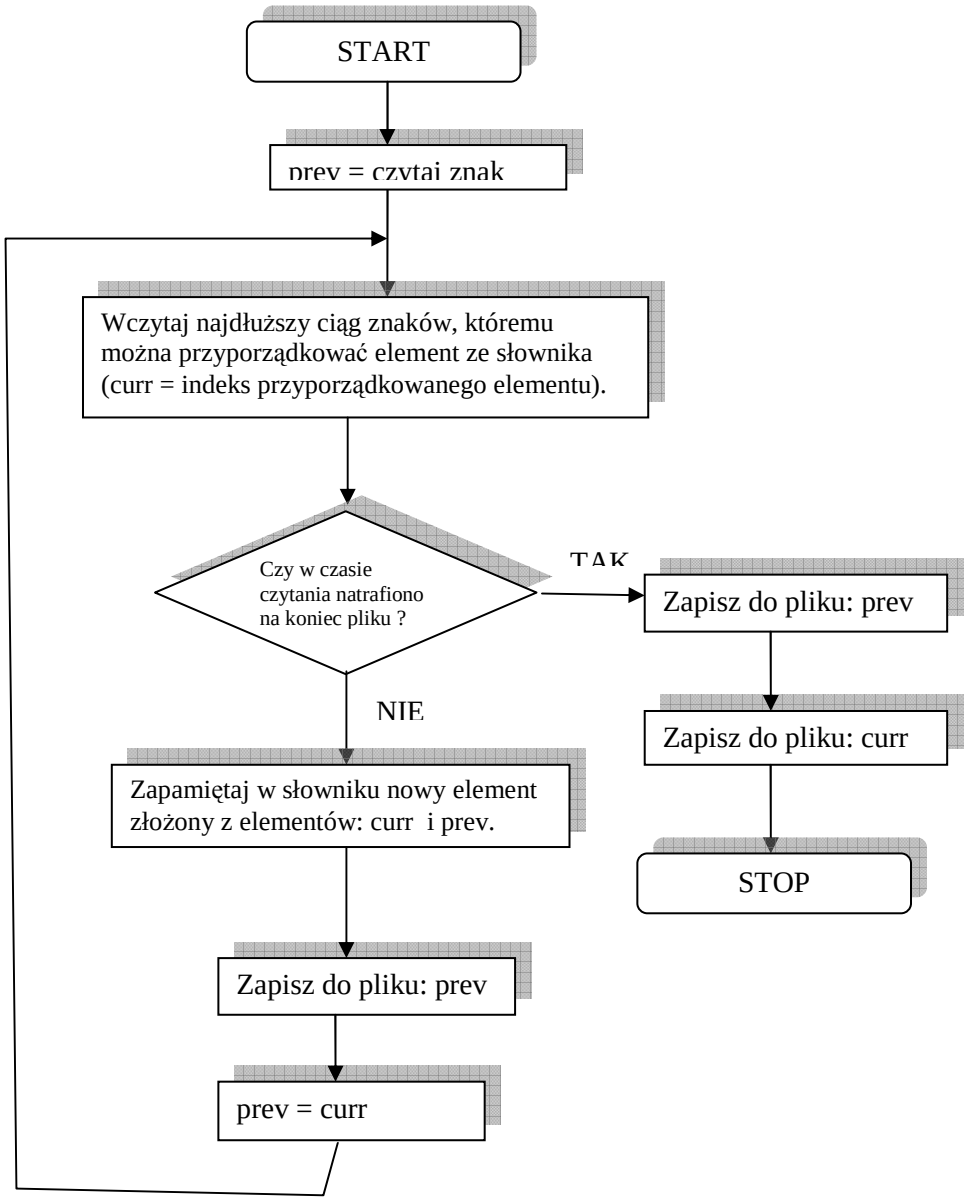
Proponowany algorytm:

- W każdej iteracji czyta z pliku najdłuższy ciąg znaków, dla którego może znaleźć istniejący element w słowniku. Przy czym nie ma tutaj jakiegokolwiek porównania z elementem zapamiętanym w pamięci. Wczytanie najdłuższego ciągu, dla którego się da znaleźć element w słowniku da gwarancję , że nowy element, który dopiero zostanie dodany, jeszcze nie wystąpił w słowniku.

Główna zmiana w moim algorytmie w porównaniu z LZW dotyczy tworzenia nowych elementów w słowniku. W LZW kolejne elementy różnią się od poprzedników (nie w sensie kolejności dodawania do słownika) jedynie tym, że posiadają o jeden znak więcej. W moim algorytmie nowy element składa się z dwóch już istniejących elementów.

Takie podejście powoduje to, że w słowniku szybciej się pojawiają dłuższe elementy. Jednak koszt czasowy przeszukiwania słownika przy czytaniu kolejnych danych z pliku wejściowego jest bardzo duży.

4.2. Algorytm blokowy



4.3. Przykłady kodowania

Przy porównaniu słowników powstałych przy kodowaniu ciągu **BOBAS BOBEK BOBCIO**, można zauważyć, że w proponowanym algorytmie powstały 2 elementy 4-znakowe, podczas, gdy w słowniku powstałym w wyniku działania LZW w ogóle nie ma elementów 4-znakowych.

Ciąg podlegający kodowaniu: **BOBAS BOBEK BOBCIO**

i	Sekwencja wejściowa	Sekwencja wyjściowa	Pamięć	Indeks (w słowniku)	Fraza (w słowniku)
0	-	-	-	[0] - [255]	Kolejne symbole alfabetu
1	„B”	-	Ind(B)	-	-
2	„O”	Ind(B)	Ind(O)	[256]	„BO”
3	„B”	Ind(O)	Ind(B)	[257]	„OB”
4	„A”	Ind(B)	Ind(A)	[258]	„BA”
5	„S”	Ind(A)	Ind(S)	[259]	„AS”
6	„_”	Ind(S)	Ind(_)	[260]	„S_”
7	„BO”	Ind(_)	256	[261]	„_BO”
8	„B”	256	Ind(B)	[262]	„BOB”
9	„E”	Ind(B)	Ind(E)	[263]	„BE”
10	„K”	Ind(E)	Ind(K)	[264]	„EK”
11	„_BO”	Ind(K)	261	[265]	„K_BO”
12	„B”	261	Ind(B)	[266]	„_BOB”
13	„C”	Ind(B)	Ind(C)	[267]	„BC”
14	„I”	Ind(C)	Ind(I)	[268]	„CI”
15	„O”	Ind(I)	Ind(O)	[269]	„IO”
16	-	Ind(O)	-	-	-

1. Dla porównania, słownik powstały w czasie kodowania ciągu znaków: **BOBAS BOBEK BOBCIO,** metodą LZW:

Indeks (w słowniku)	Fraza (w słowniku)	Indeks (w słowniku)	Fraza (w słowniku)
256	„BO”	263	„BE”
257	„OB”	264	„EK”
258	„BA”	265	„K_”
259	„AS”	266	„_BO”
260	„S_”	267	„OBC”
261	„_B”	268	„CI”
262	„BOB”	269	„IO”

Różnica w działaniu proponowanego algorytmu w stosunku do LZW jest również widoczna przy kodowaniu ciągu: **/WED/WE/WEE/WET/WEB.** Najbardziej przy tworzeniu elementu w słowniku o indeksie [262].

i	Sekwencja wejściowa	Sekwencja wyjściowa	Pamięć	Indeks (w słowniku)	Fraza (w słowniku)
0	-	-	-	[0] - [255]	Kolejne symbole alfabetu

1	„/”	-	Ind(/)	-	-
2	„W”	Ind(/)	Ind(W)	[256]	„/W”
3	„E”	Ind(W)	Ind(E)	[257]	„WE”
4	„D”	Ind(E)	Ind(D)	[258]	„ED”
5	„/W”	Ind(D)	256	[259]	„D/W”
6	„E”	256	Ind(E)	[260]	„/WE”
7	„/WE”	Ind(E)	260	[261]	„E/WE”
8	„E/WE”	260	261	[262]	„/WEE/WE”
9	„T”	261	Ind(T)	[263]	„E/WET”
10	„/WE”	Ind(T)	260	[264]	„T/WE”
11	„B”	260	Ind(B)	[265]	„/WEB”
12	-	Ind(B)	-	-	-

4.4. Skutki wprowadzonych zmian

4.4.1. Czas obliczeń

Do negatywnych skutków zmian wprowadzonych w moim algorytmie trzeba zaliczyć **wydłużenie czasu kompresji**. Jest to spowodowane większą złożonością obliczeniową przeszukiwania słownika przy czynności wyszukiwania istniejącego elementu słownika, który by zawierał przeczytany nowy ciąg znaków z pliku wejściowego. Powodem tej zwiększonej złożoności jest fakt, że w LZW (w wersji pamiętającej słownik w postaci drzewa) elementy słownika pozostające w relacji rodzic-potomek różniły się tylko 1 dodatkowym znakiem po stronie potomka. W proponowanej metodzie (która przechowuje słownik również w postaci drzewa) elementy pozostające ze sobą w takiej relacji różnią się od siebie wieloma znakami, co powoduje, ze konieczne się staje porównywanie wielu znaków przy przechodzeniu od rodzica do potomka oraz sprawia, że koszt porównania elementu i niedopasowania go do ciągu wejściowego jest o wiele większy. Różnice czasów kompresji mogą być bardzo duże. Zaproponowany algorytm może się wykonywać nawet kilkadziesiąt razy dłużej niż LZW.

4.4.2. Stopień kompresji

Pozytywną stroną tego algorytmu jest nieco lepsza kompresja plików w stosunku do LZW. Algorytm najlepiej radzi sobie przy plikach tekstowych lub dokumentach. W takich przypadkach skompresowany plik przynajmniej nieznacznie był mniejszy od pliku skompresowanego metodą LZW. Największe różnice w wielkości skompresowanych plików można zauważyć przy kompresowaniu danych tekstowych, m.in. takich jak pliki .txt czy .rtf.

Przykład kompresji pliku RTF:

Bez kompresji	3.188 MB
Kodek ZIP	0.972 MB
LZW	2.802 MB
Proponowana metoda	1.552 MB

Najmniejsze różnice występowały przy kompresji takich rodzajów plików jak dźwięk, obraz czy pliki skompresowane innymi kodekami. M.in. w przypadku plików *.mp3, gdzie stopień kompresji pliku kodekiem ZIP był niewielki, stopień kompresji metodą LZW i zaproponowaną przeze mnie był porównywalny. Obie metody radziły sobie równie źle, tworząc pliki podobnej wielkości, jednakże większe od pierwotnego.

Przykład kompresji pliku MP3:

Bez kompresji	0.921 MB:
ZIP	0.900 MB
LZW	1.265 MB
Proponowana metoda	1.205 MB

4.5. Wnioski płynące z przeprowadzonych testów

Stopień kompresji

1. Zaproponowany algorytm spisuje się dobrze przy kompresji **danych tekstowych lub dokumentów**. W tych przypadkach czasami widoczna była jego przewaga nad pozostałymi testowanymi algorytmami. Powodem tego jest powtarzalność dłuższych ciągów znaków, gdyż algorytm charakteryzuje się możliwością szybkiego (po dodaniu niewielkiej ilości elementów do słownika) stworzenia w słowniku stosunkowo długich ciągów.
2. W przypadku **obrazów** algorytm zazwyczaj sprawuje się gorzej od pozostałych algorytmów, ale podobnie do LZW. Jedynym (pozytywnym) wyjątkiem jest przypadek kompresji czarno-białej bitmapy, kiedy to algorytm osiągał przy kompresji jedne z niższych wartości bitrate. Świadczy to tylko o tym, z jakim rodzajem danych algorytm daje sobie najlepiej radę.

   Powodem tak słabej kompresji (a zarazem podobnej do tej przy LZW) w ogólnym przypadku kompresji obrazów jest fakt występowania dużej liczby kombinacji ciągów znaków w kodowanym pliku. Sprawia to, że algorytm ten zachowuje się nieco podobnie jak LZW. W takich przypadkach nowy element

jest tworzony na podstawie istniejących 2 elementów jakie się udało dopasować, przy czym ten drugi jest często bardzo krótki. W przypadku, gdy ten drugi element jest jednoznakowy mamy do czynienia z przypadkiem podobnym (lecz nie identycznym ze względu na to, że w pamięci przechowujemy indeks elementu słownika a nie ostatnio wczytany znak) dodawania elementu występującego w algorytmie LZW. Szybsze tworzenie dłuższych elementów słownika powoduje również, że istnieje mniejsze prawdopodobieństwo powtórnego ich wykorzystywania. Słownik jest wtedy mało przydatny Po zapełnieniu takiego słownika może zaistnieć sytuacja, że algorytm najczęściej wykorzystuje elementy przechowujące krótkie ciągi znaków.

3. W przypadku plików z **dźwiękiem**, tak jak dla obrazów, algorytm osiągał podobne wyniki jak LZW. Można więc wnioskować, że powód jest podobny jak dla obrazów.

Czasy wykonywania

1. W każdym przypadku algorytm ten wykonuje się dłużej niż pozostałe algorytmy, często wielokrotnie dłużej. Związane jest to z większym kosztem poszukiwania elementów w słowniku, które pasowałyby do wczytanej kolejnej porcji danych z pliku kompresowanego. Na pewno duże znaczenie ma tu sposób zapamiętania słownika w pamięci a tym samym sposób jego przeszukiwania. Tak więc wydaje się, że skrócenie czasów kompresji będzie można osiągnąć poprzez lepszą implementację algorytmu.
2. Czas kompresji i również jej stopień często różni się pomiędzy plikami tego samego typu i podobnego rozmiaru. Jeden plik może być kompresowany kilkukrotnie dłużej od innego pliku o podobnej wielkości.

W przeprowadzonych testów wynika, że pomysł tworzenia nowych elementów słownika z dwóch już istniejących znajduje zastosowanie tylko dla wąskiej grupy danych. Dane tekstowe czy bitmapy o niewielkiej ilości kolorów mogą zostać skompresowane podobnie lub lepiej niż w przypadku pozostałych testowanych algorytmów. Jednak dla większości danych taki algorytm nie sprawdza się. Dodatkową wadą tego algorytmu jest trudność jego implementacji w taki sposób, aby szybkość jego działania była porównywalna np. z algorytmem LZW. To wszystko sprawia, że algorytm ten wypada słabo w testach na tle pozostałych testowanych algorytmów.

5. Algorytm dynamicznie usuwający słowa ze słownika

5.1. Cechy algorytmu

1. algorytm bazuje na algorytmie LZW jest więc algorytmem adaptacyjnym
2. liczba bitów jest zmienna od 9 (bo zaczynamy od kodu 256) do określonej maksymalnej liczby bitów (**parametr nr 2**, od 9 do 16)
3. dodatkowo gdy słownik jest pełny, a stopień kompresji spada zmniejszany jest rozmiar słownika tak by ”zaoszczędzić” na liczbie bitów oraz dodatkowo zrobić miejsce na nowe słowa w słowniku
4. moment w którym należy oczyścić słownik wybierany jest poprzez pomiary stopnia kompresji w określony odstępach, a następnie obliczanie różnicy nowego wyniku z poprzednim; gdy co najmniej dwie różnice są ujemne i ich suma jest mniejsza niż zadana (**parametr nr 6** np. 0) następuje oczyszczenie
5. odstęp co jaki prowadzone są wszystkie obliczenia może być różny (**parametr nr 5** np.10)
6. w czasie pracy zliczana jest częstotliwość wykorzystania poszczególnych kodów i przechowywana w tablicy
7. oczyszczanie w wersji podstawowej polega na tym, że usuwane są wszystkie kody ,których częstotliwość wystąpienia jest nie większa od zadanej (**parametr nr 3** np.0); po czyszczeniu słownika wszystkie częstotliwości są zerowane
8. dodatkowo wprowadzić można ograniczenie ,że słownik po kompresji nie będzie większy niż 2^n, gdzie liczba n (**parametr nr 4** np. 10, 0-brak ograniczenia) jest parametrem (>=8) ,co powinno przyspieszyć działanie
9. nie wykorzystywane są kody mające w formie dwójkowej postać samych jedynek ,są one kodami kończącymi działanie algorytmu dekompresji
10. można również wyłączyć wszystkie modyfikacje (**parametr nr 1** , 0 - bez mod., 1 - z mod.) i zostanie algorytm LZW na zmiennej liczbie bitów

Optymalne parametry:

- P1=1

- P2=12...16 stosunek czas/jakość
- P3=0
- P4=0
- P5=10-100-1000
- P6=0

5.2. Opis działania algorytmu

1. kompresja

```
STRING = pobierz znak
WHILE są znaki do pobrania DO
CHARACTER = pobierz znak
        IF STRING+CHARACTER jest w słowniku THEN
                STRING = STRING+character
        ELSE
                wypisz kod STRING(ACTUAL BITES)
                IF jest miejsce w słowniku THEN
                        dodaj STRING+CHARACTER do słownika
                        aktualizuj  ACTUAL BITES
                END IF
                zwiększ częstotliwość słowa STRING
                Sprawdz_warunek ()
                STRING = CHARACTER
        END IF
END WHILE
wypisz kod STRING(ACTUAL BITES)
```

2. dekompresja

```
pobierz OLD_CODE
wypisz CHARACTER = OLD_CODE
WHILE SA znaki do pobrania DO
        pobierz NEW_CODE(ACTUAL BITES)
```

```
        IF NEW_CODE nie ma w słowniku THEN
                STRING = zdekoduj OLD_CODE
                STRING = STRING+CHARACTER
        ELSE
                STRING = zdekoduj NEW_CODE
        END IF
        wypisz STRING
        CHARACTER = pierwszy znak w STRING
        IF jest miejsce w słowniku THEN
                dodaj OLD_CODE + CHARACTER do słownika
                aktualizuj  ACTUAL BITES
        END IF
        zwiększ częstotliwość słowa STRING
        IF  Sprawdz_warunek() THEN
                pobierz OLD_CODE(ACTUAL BITES)
                STRING = zdekoduj OLD_CODE
                CHARACTER = pierwszy znak w STRING
                częstotliwość słowa STRING
        END IF
        OLD_CODE = NEW_CODE
END WHILE
```

3. Sprawdz_warunek()

```
IF (licznik++)%skok=0 THEN
        oblicz znacznik
        IF znacznik mniejszy niż określony próg THEN
                Czysc_slownik()
        END IF
END IF
```

4. Czysc_slownik()

```
FOR  wszystkie słowa w słowniku
        IF aktualna liczba słow. > założona  liczba słow. THEN
                przerwij
```

END IF

IF częstotliwość większa od 0 THEN

zdekoduj słowo i wszystkim podsłowom tez nadaj nowe numer

zostaw słowo i nadaj nowy numer

ELSE

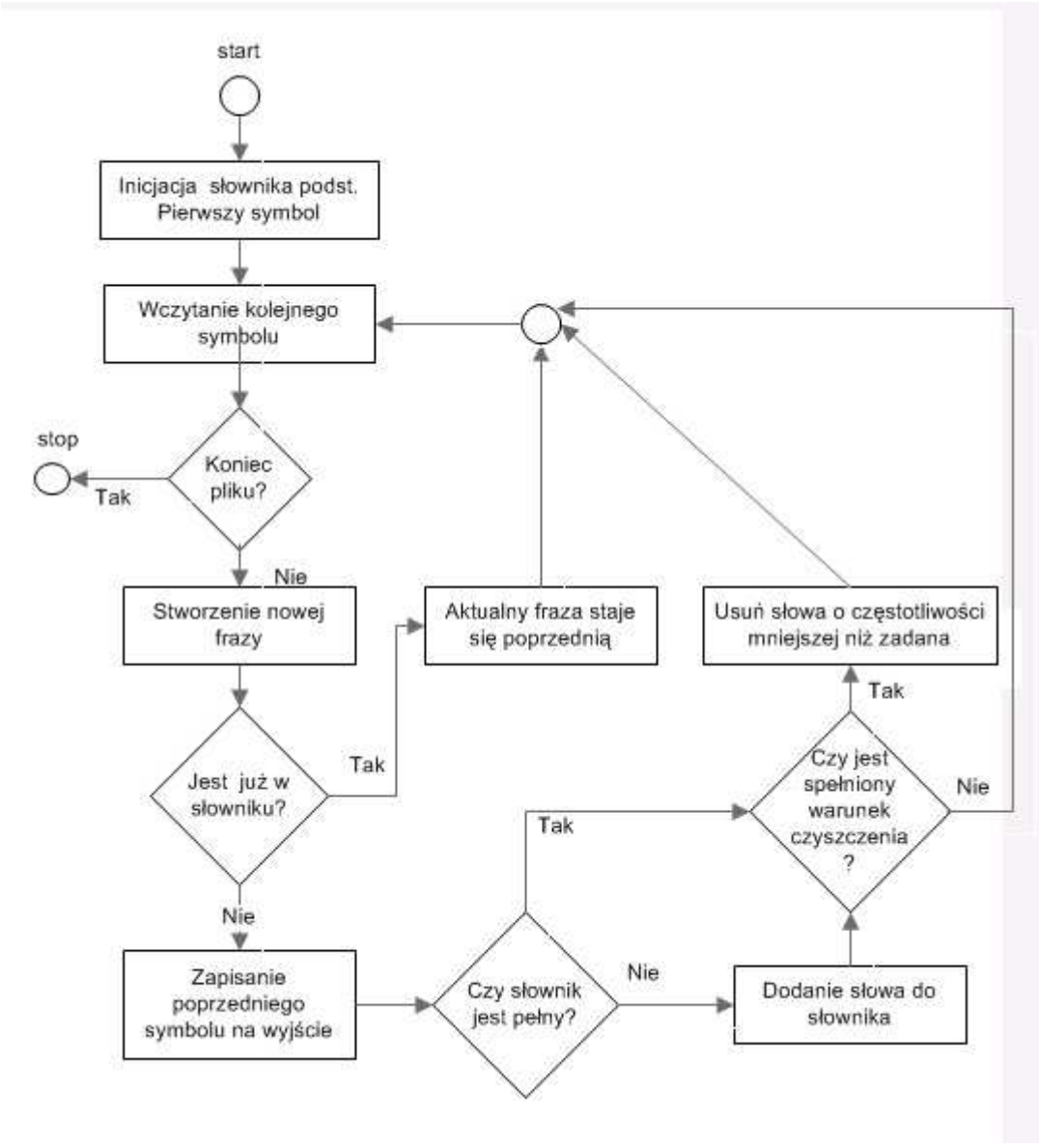
usuń słowo

END IF

wyzeruj wszystkie częstotliwości

END FOR

5.3. Algorytm blokowy



6. Wyniki testów

Oznaczenia:

- LZW12 – algorytm LZW przy stałej długości słowa 12 bitów;
- LZW15 – algorytm LZW przy zmiennej długości słowa 15 bitów;
- LZ77 – algorytm LZ77;
- LZH – algorytm LZW + Huffman;
- LZWT – realizacja algorytmu LZW na elementach drzewa;
- LZN – algorytm opisany w punkcie 2;
- LZP – algorytm opisany w punkcie 3;
- LZE – algorytm opisany w punkcie 4;
- LZD – algorytm opisany w punkcie 5;

6.1. Scenariusz testów

Cel :

Zespół był zainteresowany następującymi zagadnieniami:

- Dla jakich typów danych zaimplementowane algorytmy są skuteczne(czas,bitrate) - podejście niezależne od formatu danych wejściowych.
- Dla jakich typów danych zaimplementowane algorytmy są skuteczne(czas,bitrate) - podejście użytkowe uwzględniające format danych
- Jak wypadają zaimplementowane algorytmy na tle standardowych metod słownikowych
- Jak zmienia się jakość kompresji w zależności od liczby bitów słownika
- Jak zmienia się jakość kompresji w zależności od wielkości danych poddawanych kompresji
- Jak zmienia się czas kompresji w zależności od liczby bitów słownika
- Złożonością czasową kompresji od wielkości danych

Testy wstępne:

Dobór ostatecznych danych testowych został wykonany na podstawie serii testów próbnych . Podstawą testów wstępnych była ciekawość – podejście : „weźmy i skompresujmy”

Seria testów próbnych została przeprowadzona na różnych typach danych najczęściej poddawanych kompresji przez „zwykłych użytkowników”, w celu ogólnej oceny jakości algorytmów:

- dokumenty .doc
- pliki .txt
- pliki z muzyka w formacie .wav (muzyka , mowa)
- pliki graficzne w formacie .bmp
- pliki graficzne skompresowane .jpg
- dokumenty .pdf

Do fazy testów ostatecznych nie zostały przeznaczone formaty .jpg ,.pdf oraz inne typy danych poddane wcześniejszej kompresji, ze względu na nieefektywność kompresji tych danych za pomocą metod słownikowych, zarówno zaimplementowanych przez zespół, jak i standardowych będących punktem odniesienia.

Testy ostateczne:

Jako dane podstawowe zostały wybrane:

- zdjęcia z wakacji w formacie .bmp (kolor 24RGB)

- dokumenty .doc (książka w j.polskim „Sto lat samotności” G.G.Marquez’a bez obrazków)
- pliki .txt (ponownie ta sama książka)
- plik muzyczny w formacie .wav (instrumenty żywe)

Jako dane dodatkowe pozwalające ukazac , szczegółowe różnice między algorytmami oraz w prosty sposób zinterpretowac wyniki kompresji zostały wybrane:

- obrazek tulipany.bmp (kolor 24RBG rozmiar 2,25MB)

- obrazek mgla.bmp (kolor 24 RGB, rozmiar 576KB)

- obrazek mgla_cb.bmp(skala szarości , rozmiar 193KB)

- obrazek kogut.bmp(skala szarości , rozmiar 1,84MB)

- obrazek lena.bmp (kolor 24RBG , 192 KB)

6.2. Testy bitrate

Dokumenty .doc:

Zależnośc bitrate[bit] od liczby bitów słownika[bit] przy ustalonej wielkości pliku[MB]

0,2MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							3,053
Lz7							3,283
Lzh							2,457
15v							2,445
Lwt	5,113	4,594	3,052	2,746	2,697	2,621	
Lzn	2,91	2,681	2,555	2,495	2,508	2,621	
Lzp	3,564	3,228	3,031	2,881	2,737	2,621	
Lze	5,712	4,841	2,849	2,562	2,481	2,347	
Lzd	2,99	2,738	2,601	2,514	2,466	2,445	

0,5MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							3,123
Lz7							3,285
Lzh							2,466
15v							2,403
Lwt	5,244	4,759	3,123	2,701	2,579	2,491	
Lzn	2,922	2,677	2,518	2,4	2,318	2,301	
Lzp	3,598	3,237	3,011	2,834	2,687	2,588	
Lze	5,713	5,001	2,938	2,541	2,41	2,294	
Lzd	3,003	2,74	2,57	2,431	2,334	2,277	

0,7MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							3,276
Lz7							3,279
Lzh							2,449
15v							2,373
Lwt	5,271	4,765	3,276	2,727	2,591	2,468	
Lzn	2,917	2,668	2,51	2,374	2,591	2,233	
Lzp	3,598	3,24	2,996	2,807	2,677	2,547	
Lze	5,696	4,985	2,976	2,553	2,412	2,273	
Lzd	2,992	2,725	2,548	2,406	2,289	2,223	

1MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							3,364
Lz7							3,28
Lzh							2,45
15v							2,337
Lwt	5,286	4,795	3,362	2,712	2,558	2,427	

Lzn	2,914	2,658	2,495	2,353	2,242	2,169	
Lzp	3,599	3,236	2,996	2,799	2,64	2,509	
Lze	5,904	4,98	3,079	2,538	2,427	2,241	
Lzd	2,986	2,713	2,539	2,382	2,267	2,173	

1,5MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							2,76
Lz7							3,321
Lzh							2,487
15v							2,372
Lwt	3,734	2,995	2,76	2,62	2,523	2,439	
Lzn	2,956	2,698	2,528	2,387	2,27	2,187	
Lzp	3,622	3,269	3,025	2,844	2,684	2,541	
Lze	3,587	2,925	2,751	2,595	2,463	2,357	
Lzd	3,028	2,755	2,572	2,42	2,293	2,194	

2MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							5,011
Lz7							3,269
Lzh							2,43
15v							2,308
Lwt	5,309	4,802	5,011	2,802	2,605	2,516	
Lzn	2,902	2,637	2,468	2,336	2,215	2,123	
Lzp	3,592	3,221	2,972	2,78	2,615	2,47	
Lze	6,324	5,023	3,879	2,674	2,489	2,347	
Lzd	2,975	2,688	2,51	2,366	2,24	2,144	

Dokumenty txt:

Zależnośc bitrate[bit] od liczby bitów słownika[bit] przy ustalonej wielkości pliku[MB]

0,07MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							4,226
Lz7							5,217
Lzh							4,061
15v							3,934
Lwt	4,793	4,481	4,226	4,089	4,089	4,34	
Lzn	4,828	4,442	4,208	4,064	4,088	4,34	
Lzp	5,726	5,268	4,912	4,67	4,491	4,34	
Lze	5,06	4,697	4,442	4,271	4,228	4,468	
Lzd	4,986	4,535	4,227	4,037	3,945	3,933	

0,22MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							4,21
Lz7							5,161
Lzh							4,009

15v							3,814
Lwt	4,803	4,493	4,21	4,009	3,845	3,776	
Lzn	4,798	4,418	4,154	3,931	3,792	3,755	
Lzp	5,701	5,244	4,89	4,585	4,367	4,083	
Lze	5,059	4,727	4,461	4,212	4,03	3,888	
Lzd	4,954	4,513	4,197	3,947	3,771	3,662	

0,31MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							4,221
Lz7							5,138
Lzh							3,986
15v							3,812
Lwt	4,813	4,505	4,221	4,006	3,813	3,696	
Lzn	4,794	4,41	4,133	3,895	3,725	3,643	
Lzp	5,702	5,247	4,891	4,577	4,295	4,1	
Lze	5,058	4,749	4,481	4,218	4,009	3,829	
Lzd	4,954	4,506	4,176	3,923	3,62	3,594	

0,48MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							4,215
Lz7							5,134
Lzh							3,984
15v							3,788
Lwt	4,805	4,497	4,214	3,98	3,774	3,623	
Lzn	4,783	4,397	4,117	3,874	3,683	3,558	
Lzp	5,689	5,233	4,882	4,566	4,31	4,062	
Lze	5,045	4,747	4,48	4,2	3,978	3,761	
Lzd	4,939	4,495	4,169	3,906	3,697	3,54	

0,94MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							4,236
Lz7							5,124
Lzh							3,972
15v							3,781
Lwt	4,808	4,513	4,236	3,999	3,781	3,612	
Lzn	4,767	4,379	4,096	3,855	3,655	3,497	
Lzp	5,676	5,216	4,867	4,558	4,292	4,035	
Lze	5,058	4,777	4,524	4,249	4,023	3,795	
Lzd	4,924	4,476	4,157	3,898	3,684	3,51	

1,83MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							4,23
Lz7							5,123
Lzh							3,971
15v							3,782
Lwt	4,807	4,51	4,23	3,987	3,756	3,563	
Lzn	4,767	4,377	4,093	3,846	3,639	3,463	

Lzp	5,676	5,216	4,864	4,556	4,288	4,037	
Lze	5,056	4,772	4,511	4,222	3,962	3,664	
Lzd	4,923	4,476	4,156	3,896	3,677	3,491	

3,66MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							4,228
Lz7							5,123
Lzh							3,97
15v							3,783
Lwt	4,807	4,509	4,227	3,981	3,744	3,539	
Lzn	4,767	4,377	4,09	3,842	3,631	3,446	
Lzp	5,676	5,218	4,864	4,556	4,281	4,038	
Lze	5,056	4,769	4,505	4,208	3,932	3,599	
Lzd	4,923	4,475	4,156	3,895	3,672	3,481	

6,8MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							4,88
Lz7							4,979
Lzh							3,889
15v							3,719
Lwt	5,219	5,082	4,88	4,699	4,586	4,463	
Lzn	4,635	4,258	3,984	3,756	3,567	3,403	
Lzp	5,525	5,079	4,737	4,443	4,19	3,968	
Lze	5,585	5,483	5,392	5,175	5,13	5,006	
Lzd	4,789	4,355	4,049	3,807	3,608	3,443	

Dźwięk wav:

Zależnośc bitrate[bit] od liczby bitów słownika[bit] przy ustalonej wielkości pliku[MB]

1,1MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							10,774
Lz7							7,553
Lzh							6,266
15v							7,426
Lwt	9,091	9,923	10,774	11,451	12,175	8,27	
Lzn	7,923	7,895	7,597	7,316	7,207	7,335	
Lzp	8,388	8,574	8,476	8,21	7,999	7,938	
Lze	9,143	8,816	9,037	11,539	12,352	8,963	
Lzd	8,054	8,047	7,733	7,313	7,028	7,144	

2,4MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							11,347
Lz7							7,974
Lzh							6,597
15v							7,858

Lwt	9,511	10,425	11,347	12,14	12,953	8,864	
Lzn	8,439	8,46	8,175	7,838	7,633	7,702	
Lzp	8,875	9,142	9,089	8,808	8,533	8,387	
Lze	9,543	10,474	11,406	12,236	13,111	9,579	
Lzd	8,551	8,62	8,347	7,854	7,426	7,477	

5MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							11,64
Lz7							8,237
Lzh							6,801
15v							8,131
Lwt	9,736	10,685	11,64	12,495	13,372	9,371	
Lzn	8,791	8,88	8,635	8,254	7,904	7,879	
Lzp	9,181	9,536	9,548	9,281	8,923	8,678	
Lze	9,765	10,734	11,706	12,58	13,515	9,969	
Lzd	8,863	9,026	8,848	8,338	7,682	7,629	

11MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							11,806
Lz7							8,436
Lzh							7,002
15v							8,508
Lwt	9,856	10,829	11,806	12,686	13,6	10,136	
Lzn	9,068	9,273	9,133	8,775	8,352	8,108	
Lzp	9,395	9,858	9,993	9,806	9,435	9,078	
Lze	9,878	10,863	11,85	12,755	13,724	10,695	
Lzd	9,075	9,348	9,329	8,94	8,226	7,855	

18MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							11,717
Lz7							8,408
Lzh							6,993
15v							8,561
Lwt	9,774	10,744	11,717	12,543	13,486	10,331	
Lzn	9,042	9,295	9,211	8,879	8,443	8,106	
Lzp	9,336	9,841	10,03	9,889	9,528	9,135	
Lze	9,795	10,771	11,748	12,639	13,596	10,845	
Lzd	9,074	9,392	9,468	9,155	8,445	7,926	

Bitmapy bmp:

Zależnośc bitrate[bit] od liczby bitów słownika[bit] przy ustalonej wielkości pliku[MB]

7,5MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							10,155
Lz7							8,374
Lzh							7,534

15v							8,925
Lwt	9,577	10,063	10,155	9,936	9,5	9,078	
Lzn	8,952	9,548	9,938	9,416	8,83	8,255	
Lzp	9,2	9,858	10,412	10,43	10,074	9,523	
Lze	9,601	10,134	10,348	10,328	10,151	9,951	
Lzd	8,917	9,425	9,747	9,579	9,051	8,348	

10MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							10,059
Lz7							8,209
Lzh							7,421
15v							8,554
Lwt	9,504	9,951	10,059	9,891	9,592	9,29	
Lzn	8,732	9,299	9,674	9,064	8,451	7,88	
Lzp	9,011	9,623	10,155	10,091	9,685	9,128	
Lze	9,549	10,055	10,299	10,343	10,25	10,133	
Lzd	8,732	9,213	9,473	9,212	8,633	7,945	

12,5MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							10,104
Lz7							7,874
Lzh							7,082
15v							8,064
Lwt	9,548	9,938	10,104	10,08	9,958	9,814	
Lzn	8,333	8,863	9,12	8,539	7,959	7,417	
Lzp	8,609	9,183	9,655	9,552	9,14	8,605	
Lze	9,592	10,04	10,345	10,551	10,613	10,628	
Lzd	8,342	8,793	8,981	8,692	8,12	7,47	

15,5MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							10,066
Lz7							7,59
Lzh							6,785
15v							7,611
Lwt	9,448	9,896	10,066	10,112	10,132	10,14	
Lzn	7,998	8,478	8,6	8,055	7,496	6,99	
Lzp	8,279	8,81	9,201	9,054	8,636	8,121	
Lze	9,526	10,104	10,395	10,645	10,883	11,068	
Lzd	8,018	8,421	8,518	8,195	7,641	7,033	

18MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							10,16
Lz7							7,308
Lzh							6,513
15v							7,229
Lwt	9,532	9,909	10,159	10,39	10,506	10,609	
Lzn	7,685	8,126	8,153	7,639	7,107	6,636	

Lzp	7,968	8,462	8,796	8,62	8,206	7,713	
Lze	9,603	10,096	10,47	10,879	11,18	11,416	
Lzd	7,71	8,076	8,104	7,771	7,241	6,674	

Bitmapy bmp:

Zależnośc bitrate[bit] od wielkości pliku [MB]

12 bitów					
	7,5	**10**	**12,5**	**15,5**	**18**
Lzw	10,155	10,059	10,104	10,066	10,16
Lz7	8,374	8,209	7,874	7,59	7,308
Lzh	7,534	7,421	7,082	6,785	6,513
15v	8,925	8,554	8,064	7,611	7,229
Lwt	10,155	10,059	10,104	10,066	10,159
Lzn	9,938	9,674	9,12	8,6	8,153
Lzp	10,412	10,155	9,655	9,201	8,796
Lze	10,348	10,299	10,345	10,395	10,47
Lzd	9,747	9,473	8,981	8,518	8,104

15 bitów					
	7,5	**10**	**12,5**	**15,5**	**18**
Lzw	10,155	10,059	10,104	10,066	10,16
Lz7	8,374	8,209	7,874	7,59	7,308
Lzh	7,534	7,421	7,082	6,785	6,513
15v	8,925	8,554	8,064	7,611	7,229
Lwt	9,078	9,29	9,814	10,14	10,609
Lzn	8,255	7,88	7,417	6,99	6,636
Lzp	9,523	9,128	8,605	8,121	7,713
Lze	9,951	10,133	10,628	11,068	11,416
Lzd	8,348	7,945	7,47	7,033	6,674

Dokumenty txt:

Zależnośc bitrate[bit] od wielkości pliku [MB]

12 bitów								
	0,07	**0,22**	**0,31**	**0,48**	**0,94**	**1,83**	**3,66**	**6,8**
Lzw	4,226	4,21	4,221	4,215	4,236	4,23	4,228	4,88
Lz7	5,217	5,161	5,138	5,134	5,124	5,123	5,123	4,979
Lzh	4,061	4,009	3,986	3,984	3,972	3,971	3,97	3,889
15v	3,934	3,814	3,812	3,788	3,781	3,782	3,783	3,719
Lwt	4,226	4,21	4,221	4,214	4,23	4,23	4,227	4,88
Lzn	4,208	4,154	4,133	4,117	4,093	4,093	4,09	3,984
Lzp	4,912	4,89	4,891	4,882	4,864	4,864	4,864	4,737
Lze	4,442	4,461	4,481	4,48	4,511	4,511	4,505	5,392
Lzd	4,227	4,197	4,176	4,169	4,156	4,156	4,156	4,049

15 bitów								
	0,07	**0,22**	**0,31**	**0,48**	**0,94**	**1,83**	**3,66**	**6,8**
Lzw	4,226	4,21	4,221	4,215	4,236	4,23	4,228	4,88
Lz7	5,217	5,161	5,138	5,134	5,124	5,123	5,123	4,979
Lzh	4,061	4,009	3,986	3,984	3,972	3,971	3,97	3,889
15v	3,934	3,814	3,812	3,788	3,781	3,782	3,783	3,719
Lwt	4,34	3,776	3,696	3,623	3,612	3,563	3,539	4,463
Lzn	4,34	3,755	3,643	3,558	3,497	3,463	3,446	3,403
Lzp	4,34	4,083	4,1	4,062	4,035	4,037	4,038	3,968
Lze	4,468	3,888	3,829	3,761	3,795	3,664	3,599	5,006
Lzd	3,933	3,662	3,594	3,54	3,51	3,491	3,481	3,443

Dzwięki wav:

Zależnośc bitrate[bit] od wielkości pliku [MB]

12 bitów					
	1,1	**2,4**	**5**	**11**	**18**
Lzw	10,774	11,347	11,64	11,806	11,717
Lz7	7,553	7,974	8,237	8,436	8,408
Lzh	6,266	6,597	6,801	7,002	6,993
15v	7,426	7,858	8,131	8,508	8,561
Lwt	10,774	11,347	11,64	11,806	11,717
Lzn	7,597	8,175	8,635	9,133	9,211
Lzp	8,476	9,089	9,548	9,993	10,03
Lze	9,037	11,406	11,706	11,85	11,748
Lzd	7,733	8,347	8,848	9,329	9,468

15 bitów					
	1,1	**2,4**	**5**	**11**	**18**
Lzw	10,774	11,347	11,64	11,806	11,717
Lz7	7,553	7,974	8,237	8,436	8,408
Lzh	6,266	6,597	6,801	7,002	6,993
15v	7,426	7,858	8,131	8,508	8,561
Lwt	8,27	8,864	9,371	10,136	10,331
Lzn	7,335	7,702	7,879	8,108	8,106
Lzp	7,938	8,387	8,678	9,078	9,135
Lze	8,963	9,579	9,969	10,695	10,845
Lzd	7,144	7,477	7,629	7,855	7,926

Dokumenty doc:

Zależnośc bitrate[bit] od wielkości pliku [MB]

12 bitów								
	0,2	**0,5**	**0,7**	**1**	**2**	**4**	**8**	**16**
Lzw	3,053	3,123	3,276	3,364	5,011	5,221	5,157	4,847
Lz7	3,283	3,285	3,279	3,28	3,269	3,282	3,282	3,096
Lzh	2,457	2,466	2,449	2,45	2,43	2,437	2,431	2,221

15v	2,445	2,403	2,373	2,337	2,308	2,314	2,316	2,168
Lwt	3,052	3,123	3,276	3,362	5,011	5,221	5,157	4,847
Lzn	2,555	2,518	2,51	2,495	2,468	2,471	2,466	2,266
Lzp	3,031	3,011	2,996	2,996	2,972	2,982	2,981	2,766
Lze	2,849	2,938	2,976	3,079	3,879	5,419	5,374	5,055
Lzd	2,601	2,57	2,548	2,539	2,51	2,513	2,509	2,296

15 bitów								
	0,2	**0,5**	**0,7**	**1**	**2**	**4**	**8**	**16**
Lzw	3,053	3,123	3,276	3,364	5,011	5,221	5,157	4,847
Lz7	3,283	3,285	3,279	3,28	3,269	3,282	3,282	3,096
Lzh	2,457	2,466	2,449	2,45	2,43	2,437	2,431	2,221
15v	2,445	2,403	2,373	2,337	2,308	2,314	2,316	2,168
Lwt	2,621	2,491	2,468	2,427	2,516	2,654	2,775	3,752
Lzn	2,621	2,301	2,233	2,169	2,123	2,102	2,093	1,976
Lzp	2,621	2,588	2,547	2,509	2,47	2,471	2,473	2,315
Lze	2,347	2,294	2,273	2,241	2,347	2,499	2,657	3,839
Lzd	2,445	2,277	2,223	2,173	2,144	2,127	2,115	2

6.3. Testy czasu kompresji

Dokumenty doc:

zależnośc czasu kompresji[s] od ilości bitów słownika przy ustalonej wielkości pliku[MB]

0,2MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							0,281
Lz7							0,375
Lzh							0,359
15v							0,25
Lwt	0,093	0,093	0,093	0,109	0,078	0,109	
Lzn	0,14	0,14	0,14	0,14	0,125	0,125	
Lzp	0,125	0,093	0,093	0,093	0,078	0,078	
Lze	1,265	0,859	0,453	0,468	0,5	0,531	
Lzd	0,156	0,14	0,109	0,125	0,093	0,093	

0,5MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							0,39
Lz7							0,859
Lzh							0,703
15v							0,296
Lwt	0,187	0,171	0,187	0,203	0,171	0,234	
Lzn	0,328	0,265	0,312	0,328	0,343	0,39	

Lzp	0,187	0,14	0,187	0,171	0,14	0,171	
Lze	3,437	2,343	1,312	1,296	1,484	1,765	
Lzd	0,375	0,25	0,25	0,281	0,234	0,281	

0,7MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							0,546
Lz7							1,218
Lzh							1
15v							0,359
Lwt	0,234	0,25	0,218	0,234	0,25	0,296	
Lzn	0,421	0,421	0,437	0,453	0,531	0,593	
Lzp	0,25	0,234	0,218	0,234	0,218	0,25	
Lze	4,234	2,937	1,625	1,625	1,937	2,453	
Lzd	0,484	0,375	0,343	0,343	0,328	0,406	

1MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							0,531
Lz7							1,828
Lzh							1,39
15v							0,5
Lwt	0,343	0,375	0,343	0,359	0,375	0,437	
Lzn	0,578	0,609	0,625	0,64	0,796	0,921	
Lzp	0,312	0,328	0,328	0,343	0,343	0,375	
Lze	6,89	4,562	2,656	2,5	3,265	3,875	
Lzd	0,671	0,562	0,484	0,515	0,531	0,593	

2MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							1,125
Lz7							3,515
Lzh							2,531
15v							0,75
Lwt	0,625	0,703	0,812	0,64	0,718	0,89	
Lzn	1,14	1,109	1,156	1,203	1,484	1,75	
Lzp	0,578	0,546	0,546	0,562	0,593	0,671	
Lze	14	8,921	6,437	4,953	6,14	7,734	
Lzd	1,25	0,984	0,921	0,937	0,968	1,156	

Dokumenty txt:

zależnośc czasu kompresji[s] od ilości bitów słownika przy ustalonej wielkości pliku[MB]

0,07MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							0,515
Lz7							0,25

Lzh							0,234
15v							0,281
Lwt	0,203	0,046	0,046	0,046	0,062	0,046	
Lzn	0,093	0,062	0,062	0,062	0,078	0,078	
Lzp	0,046	0,062	0,046	0,046	0,046	0,046	
Lze	0,265	0,203	0,218	0,265	0,328	0,375	
Lzd	0,093	0,078	0,062	0,062	0,062	0,046	

0,22MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							0,265
Lz7							0,484
Lzh							0,343
15v							0,359
Lwt	0,078	0,078	0,093	0,109	0,093	0,125	
Lzn	0,171	0,156	0,171	0,171	0,203	0,218	
Lzp	0,093	0,093	0,203	0,093	0,093	0,109	
Lze	0,578	0,562	0,593	0,734	1	1,39	
Lzd	0,203	0,171	0,14	0,14	0,14	0,156	

0,31MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							0,296
Lz7							0,64
Lzh							0,375
15v							0,296
Lwt	0,109	0,109	0,109	0,14	0,14	0,187	
Lzn	0,218	0,203	0,203	0,296	0,265	0,312	
Lzp	0,109	0,125	0,125	0,125	0,14	0,14	
Lze	0,781	0,781	0,812	1,015	1,421	1,953	
Lzd	0,265	0,218	0,171	0,187	0,046	0,25	

0,48MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							0,39
Lz7							0,921
Lzh							0,546
15v							0,328
Lwt	0,156	0,156	0,171	0,171	0,187	0,25	
Lzn	0,328	0,296	0,312	0,328	0,421	0,5	
Lzp	0,187	0,156	0,171	0,171	0,187	0,296	
Lze	1,218	1,265	1,265	1,656	2,218	3,078	
Lzd	0,39	0,343	0,296	0,296	0,312	0,359	

0,94MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							0,531
Lz7							1,781
Lzh							0,921
15v							0,531
Lwt	0,312	0,281	0,296	0,312	0,343	0,437	
Lzn	0,609	0,578	0,593	0,64	0,796	1,031	

Lzp	0,343	0,375	0,312	0,312	0,343	0,437	
Lze	2,343	2,328	2,5	3,14	4,406	6,109	
Lzd	0,734	0,578	0,515	0,531	0,562	0,656	

1,83MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							0,859
Lz7							3,562
Lzh							1,625
15v							0,968
Lwt	0,546	0,546	0,546	0,609	0,687	0,875	
Lzn	1,156	1,125	1,203	1,265	1,64	2,078	
Lzp	0,625	0,625	0,578	0,593	0,671	0,734	
Lze	4,671	4,734	4,937	6,171	8,625	12,156	
Lzd	1,453	1,14	1,015	1,078	1,109	1,312	

3,66MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							1,687
Lz7							6,796
Lzh							3,156
15v							1,718
Lwt	1,062	1,109	1,109	1,234	1,343	1,796	
Lzn	2,296	2,25	2,312	2,515	3,281	4,234	
Lzp	1,265	1,218	1,14	1,218	1,281	1,484	
Lze	9,296	9,234	9,968	12,14	17,109	23,578	
Lzd	2,906	2,25	2,031	2,109	2,218	2,718	

6,8MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							2,968
Lz7							12,671
Lzh							5,718
15v							3
Lwt	2,156	2,14	2,171	2,375	2,546	3,156	
Lzn	4,234	4,109	4,281	4,609	5,984	7,828	
Lzp	2,25	2,171	2,109	2,171	2,343	2,718	
Lze	19,203	19,5	21,265	25,015	33,828	47,921	
Lzd	5,203	4,093	3,671	3,796	4,015	5,031	

Dźwięki wav:

zależnośc czasu kompresji[s] od ilości bitów słownika przy ustalonej wielkości pliku[MB]

1,1MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							1,5
Lz7							2,234
Lzh							1,375

15v							0,921
Lwt	0,531	0,468	0,468	0,484	0,5	0,687	
Lzn	1,125	1,25	1,218	1,312	1,64	2,921	
Lzp	0,562	0,64	0,671	0,703	0,765	0,921	
Lze	9,218	8,39	8,125	9,734	9,984	12,5	
Lzd	1,156	0,937	0,89	0,937	0,921	1,046	

2,4MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							2,078
Lz7							4,343
Lzh							2,156
15v							1,812
Lwt	0,921	0,921	0,953	1,015	1,015	1,531	
Lzn	2,39	2,562	2,703	2,921	3,562	6,468	
Lzp	1,187	1,296	1,421	1,515	1,687	2,062	
Lze	17,968	18,265	19,062	19,75	20,125	26,171	
Lzd	2,312	1,968	1,89	1,984	1,984	2,234	

5MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d.**
Lzw							3,921
Lz7							9,296
Lzh							4,265
15v							3,593
Lwt	1,875	1,937	1,937	2,046	2,093	3,187	
Lzn	5,015	5,484	5,921	6,609	7,796	13,875	
Lzp	2,5	2,734	2,984	3,328	3,703	4,562	
Lze	32,875	33,89	34,468	37,312	38,171	54,734	
Lzd	4,89	4,203	4,062	4,359	4,312	4,828	

11MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							9,203
Lz7							20,015
Lzh							8,703
15v							7,656
Lwt	4,031	4,203	4,187	4,484	4,5	7,015	
Lzn	10,671	11,859	13,25	15,718	19,218	32,046	
Lzp	5,234	5,75	6,421	7,406	8,765	11,421	
Lze	56,593	58,453	60,296	70,296	72	116,921	
Lzd	10,625	9,25	8,984	9,718	9,89	11,093	

18MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							12,812
Lz7							32,156
Lzh							14,421
15v							13,562
Lwt	6,781	6,875	6,984	7,421	7,515	11,453	
Lzn	17,39	19,359	21,921	26,609	32,906	53,656	
Lzp	8,546	9,312	10,453	12,156	14,656	19,281	

Lze	89,671	90,703	91,937	112,328	113,875	182,156	
Lzd	17,39	15,125	14,812	16,093	16,609	19,046	

Bitmapy bmp:

zależnośc czasu kompresji[s] od ilości bitów słownika przy ustalonej wielkości pliku[MB]

7,5MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							5,875
Lz7							12,906
Lzh							5,859
15v							5,031
Lwt	3,031	3,234	3,531	4,406	5,765	9,859	
Lzn	6,234	6,609	8,421	9,39	12,953	23,765	
Lzp	3,187	3,328	3,64	3,953	4,859	7,031	
Lze	20,875	29,593	33,296	41,187	60,203	94,953	
Lzd	7,328	6,281	5,953	6,421	6,937	8,265	

10MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							7,046
Lz7							18,75
Lzh							8,109
15v							6,656
Lwt	3,953	4,25	4,687	5,656	7,218	11,14	
Lzn	8,296	8,781	9,859	12,078	16,453	28,921	
Lzp	4,312	4,421	4,89	5,171	6,156	8,718	
Lze	22,796	37,437	42,453	51,515	74,968	114,281	
Lzd	9,843	8,375	7,968	8,531	9,234	10,968	

12,5MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							9,546
Lz7							23,171
Lzh							11,125
15v							7,859
Lwt	4,828	5,218	5,703	6,687	8,046	11,89	
Lzn	10,046	10,578	11,703	14,218	19,343	33,453	
Lzp	5,093	5,281	5,562	6,125	7,218	10,031	
Lze	32,359	42,64	47,593	58,281	84,281	128,281	
Lzd	11,812	10,125	9,531	10,281	10,937	12,921	

15,5MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							10,75
Lz7							28,609
Lzh							14,343
15v							10,421

Lwt	5,859	6,234	6,796	7,625	8,875	11,546	
Lzn	12,109	12,671	13,968	16,656	22,593	38,171	
Lzp	6,14	6,437	6,64	7,203	8,453	11,609	
Lze	32	46,75	56,546	67,015	90,234	130,89	
Lzd	14,25	12,187	11,468	12,25	12,984	15,453	

18MB							
	10	**11**	**12**	**13**	**14**	**15**	**n.d**
Lzw							13,515
Lz7							31,89
Lzh							17,64
15v							10,171
Lwt	6,828	7,156	7,734	8,515	9,89	12,421	
Lzn	13,843	14,468	15,718	18,609	25,234	42,203	
Lzp	6,984	7,171	7,5	8,125	9,656	13	
Lze	32,781	46,171	58,031	70,218	96,125	139,875	
Lzd	16,296	13,906	13,109	13,875	14,718	17,531	

Dokumenty doc:

zależnośc czasu kompresji [s] od wielkości pliku [MB]

12 bitów

	0,2	**0,5**	**0,7**	**1**	**2**	**4**	**8**	**16**
Lzw	0,281	0,39	0,546	0,531	1,125	2,046	4,156	7,046
Lz7	0,375	0,859	1,218	1,828	3,515	7,593	14,359	28,406
Lzh	0,359	0,703	1	1,39	2,531	4,687	9,484	19,406
15v	0,25	0,296	0,359	0,5	0,75	1,593	2,734	5,234
Lwt	0,093	0,187	0,218	0,343	0,812	1,531	2,843	5,765
Lzn	0,14	0,312	0,437	0,625	1,156	2,296	4,515	8,968
Lzp	0,093	0,187	0,218	0,328	0,546	1,109	2,125	4,265
Lze	0,453	1,312	1,625	2,656	6,437	18,859	44,671	91,046
Lzd	0,109	0,25	0,343	0,484	0,921	1,812	3,578	7,031

15 bitów

	0,2	**0,5**	**0,7**	**1**	**2**	**4**	**8**	**16**
Lzw	0,281	0,39	0,546	0,531	1,125	2,046	4,156	7,046
Lz7	0,375	0,859	1,218	1,828	3,515	7,593	14,359	28,406
Lzh	0,359	0,703	1	1,39	2,531	4,687	9,484	19,406
15v	0,25	0,296	0,359	0,5	0,75	1,593	2,734	5,234
Lwt	0,109	0,234	0,296	0,437	0,89	1,906	3,984	11,046
Lzn	0,125	0,39	0,593	0,921	1,75	3,765	7,781	16,156
Lzp	0,078	0,171	0,25	0,375	0,671	1,359	2,671	5,406
Lze	0,531	1,765	2,453	3,875	7,734	17,25	38,593	106,546
Lzd	0,093	0,281	0,406	0,593	1,156	2,5	4,968	9,968

Dokumenty txt:

zależnośc czasu kompresji [s] od wielkości pliku [MB]

12 bitów								
	0,07	**0,22**	**0,31**	**0,48**	**0,94**	**1,83**	**3,66**	**6,8**
Lzw	0,515	0,265	0,296	0,39	0,531	0,859	1,687	2,968
Lz7	0,25	0,484	0,64	0,921	1,781	3,562	6,796	12,671
Lzh	0,234	0,343	0,375	0,546	0,921	1,625	3,156	5,718
15v	0,281	0,359	0,296	0,328	0,531	0,968	1,718	3
Lwt	0,046	0,093	0,109	0,171	0,296	0,546	1,109	2,171
Lzn	0,062	0,171	0,203	0,312	0,593	1,203	2,312	4,281
Lzp	0,046	0,203	0,125	0,171	0,312	0,578	1,14	2,109
Lze	0,218	0,593	0,812	1,265	2,5	4,937	9,968	21,265
Lzd	0,062	0,14	0,171	0,296	0,515	1,015	2,031	3,671

15 bitów								
	0,07	**0,22**	**0,31**	**0,48**	**0,94**	**1,83**	**3,66**	**6,8**
Lzw	0,515	0,265	0,296	0,39	0,531	0,859	1,687	2,968
Lz7	0,25	0,484	0,64	0,921	1,781	3,562	6,796	12,671
Lzh	0,234	0,343	0,375	0,546	0,921	1,625	3,156	5,718
15v	0,281	0,359	0,296	0,328	0,531	0,968	1,718	3
Lwt	0,046	0,125	0,187	0,171	0,437	0,875	1,796	3,156
Lzn	0,078	0,218	0,312	0,312	1,031	2,078	4,234	7,828
Lzp	0,046	0,109	0,14	0,171	0,437	0,734	1,484	2,718
Lze	0,375	1,39	1,953	1,265	6,109	12,156	23,578	47,921
Lzd	0,046	0,156	0,25	0,296	0,656	1,312	2,718	5,031

Dźwięki wav:

zależnośc czasu kompresji [s] od wielkości pliku [MB]

12 bitów					
	1,1	**2,4**	**5**	**11**	**18**
Lzw	1,5	2,078	3,921	9,203	12,812
Lz7	2,234	4,343	9,296	20,015	32,156
Lzh	1,375	2,156	4,265	8,703	14,421
15v	0,921	1,812	3,593	7,656	13,562
Lwt	0,468	0,953	1,937	4,187	6,984
Lzn	1,218	2,703	5,921	13,25	21,921
Lzp	0,671	1,421	2,984	6,421	10,453
Lze	8,125	19,062	34,468	60,296	91,937
Lzd	0,89	1,89	4,062	8,984	14,812

15 bitów					
	1,1	**2,4**	**5**	**11**	**18**
Lzw	1,5	2,078	3,921	9,203	12,812
Lz7	2,234	4,343	9,296	20,015	32,156
Lzh	1,375	2,156	4,265	8,703	14,421
15v	0,921	1,812	3,593	7,656	13,562
Lwt	0,687	1,531	3,187	7,015	11,453
Lzn	2,921	6,468	13,875	32,046	53,656
Lzp	0,921	2,062	4,562	11,421	19,281
Lze	12,5	26,171	54,734	116,921	182,156

Lzd	1,046	2,234	4,828	11,093	19,046

Bitmapy bmp:

zależnośc czasu kompresji [s] od wielkości pliku [MB]

12 bitów					
	7,5	**10**	**12,5**	**15,5**	**18**
Lzw	5,875	7,046	9,546	10,75	13,515
Lz7	12,906	18,75	23,171	28,609	31,89
Lzh	5,859	8,109	11,125	14,343	17,64
15v	5,031	6,656	7,859	10,421	10,171
Lwt	3,531	4,687	5,703	6,796	7,734
Lzn	8,421	9,859	11,703	13,968	15,718
Lzp	3,64	4,89	5,562	6,64	7,5
Lze	33,296	42,453	47,593	56,546	58,031
Lzd	5,953	7,968	9,531	11,468	13,109

15 bitów					
	7,5	**10**	**12,5**	**15,5**	**18**
Lzw	5,875	7,046	9,546	10,75	13,515
Lz7	12,906	18,75	23,171	28,609	31,89
Lzh	5,859	8,109	11,125	14,343	17,64
15v	5,031	6,656	7,859	10,421	10,171
Lwt	9,859	11,14	11,89	11,546	12,421
Lzn	23,765	28,921	33,453	38,171	42,203
Lzp	7,031	8,718	10,031	11,609	13
Lze	94,953	114,281	128,281	130,89	139,875
Lzd	8,265	10,968	12,921	15,453	17,531

Dalsze porównania i przeprowadzone testy dotyczą wyłącznie metody dynamicznie usuwanych słów ze słownika. Wszystkie dalsze testy zostaną przeprowadzone na pliku typu bmp o rozmiarze 1,84 MB. Plik ten przyjęliśmy jako wzór dla testów.

W poszczególnych tabelach umieszczane będą wyniki obliczeń w zależności od zmiennych 6 parametrów wejściowych dla tego algorytmu opisanych w punkcie 5. W przypadku, gdy zmieniamy jeden z parametrów reszta pozostaje ustawiona na swoje wartości domyślne:

- modyfikacja – TAK;
- liczba bitów – 12;

- częstotliwość – 0;
- liczba bitów po kompresji – 0;
- skok – 10;
- granica – 0;
-

Wpływ zmian częstotliwości na wyniki

Częstotliwość	Czas [s]	Bit rate
0	1,231	6,48
1	3,234	6,64
2	2,693	6,72
3	2,383	6,8
4	2,653	6,56
5	2,443	6,72
6	2,683	6,56
7	2,313	6,48
8	2,283	6,48
9	2,062	6,64
10	2,303	6,48

Wpływ zmian liczby bitów po kompresji na wyniki

Ilość bitów na wyjściu	Czas [s]	Bit rate
Brak ograniczenia	2,253	6,48
8	1,912	6,88
9	1,972	6,8
10	2,153	6,64
11	2,293	6,48
12	2,273	6,48

Wpływ zmian skoku na wyniki

Skok	Czas [s]	Bit rate
1	2,363	7,5
5	2,263	7,5
10	2,263	7,5
15	2,253	7,5
20	2,233	7,5
30	2,203	7,5
40	2,173	7,5
50	2,133	7,5
60	2,093	7,5
70	2,032	7,5
80	2,032	7,5
90	2,002	7,5
100	1,982	7,5

Wpływ zmian granicy na wyniki

Parametr granicy	Czas [s]	Bit Rate
-9	1,361	8,32
-8	1,341	8,32
-7	1,331	8,32

-6	1,392	8,32
-5	1,392	8,24
-4	1,371	8,24
-3	1,412	8,8
-2	1,572	8,8
-1	1,301	6,88
0	2,483	6,48
1	2,513	6,48

7. Wnioski

Oznaczmy odpowiednio:

- LZN– algorytm z punktu 2 (usuwanie najrzadziej używanych);
- LZP– algorytm z punktu 3 (pełne usuwanie po przepełnieniu);
- LZE– algorytm z punktu 4 (tworzenie słowa z 2 istniejących);
- LZD– algorytm z punktu 5(dynamiczne czyszczenie słownika);

Wnioski są wyprowadzone na podstawie przeprowadzonych testów, które ułatwiają przemyślenie „dlaczego gorzej ?” – „dlaczego lepiej ?”.

7.1. Porównania

7.1.1. **LZN** vs LWT

a. Czas

Algorytm LZN jest na pewno gorszy czasowo od LWT. Widoczne to jest na wykresach. Dzieje się tak, ponieważ algorytm ten wymaga stworzenia dodatkowej listy używanych elementów oraz jej przetwarzania.

b. Stopień kompresji

Kosztem czasu, zyskaliśmy lepszą kompresję poprzez usuwanie najrzadziej występujących fraz kodowych w słowniku. I tak np.:

i. Dla obrazków wykresy uwidoczniają, iż kompresja poprawia się. Stopień poprawy oczywiście zależy od ułożenia pixeli w tym obrazku tak, jak np.: dla obrazka *mgła.bmp* stopień kompresji jest widoczny, gdyż w LWT na początku utworzony jest słownik z frazami, które pożniej mało są używane. Natomiast przy LZN słownik się zmienia i te frazy najrzadziej używane są usuwane w

przypadku przepełnienia słownika. W przypadku obrazka *tulipany.bmp* początkowo stworzony słownik jest później częściej wykorzystywany niż w przypadku obrazka *mgła.bmp* i dlatego różnica w kompresji między LZN a LWT jest mniejsza.

ii. Z plikami typu wav będzie podobnie – zależy od początkowego słownika, jaki zostanie stworzony w LWT. Jeśli będzie często wykorzystywany później, różnice między LWT i LZN będą niewielkie.

iii. Dla większości plików tekstowych stopień kompresji jest wysoki. Spowodowane jest to tym, że w tekstach wiele fraz się powtarza, a frazy najmniej używane są usuwane w LZN w przeciwieństwie do LWT, gdzie tworzony jest początkowy słownik i taki już zostaje – jeśli są tam elementy, które są mało używane, wtedy kompresja jest dużo gorsza niż w LZN, gdyż wykorzystanie takiego słownika staje się nieefektywne.

7.1.2 **LZP** vs LWT

c. Czas

LZP ma zbliżone parametry czasowe do LWT choć po przepełnieniu słownika jest on czyszczony i tworzony jest nowy, natomiast w LWT pierwszy stworzony jest jedynym stałym słownikiem.

d. Stopień kompresji

Algorytm LZP kompresuje lepiej niż LWT, jak pokazują testy.

Dzieje się tak, gdyż po przepełnieniu słownika zostaje on wyczyszczony i budowany nowy. W przypadku kiedy większość elementów w słowniku LWT nie jest później wykorzystywana algorytm ten będzie dawał słabą kompresję, natomiast w przypadku LZP słownik ten będzie zmienny. I tak np.:

i. Dla obrazków kompresja poprawia się. Oczywiście tak, jak w przypadku poprzedniego porównania stopień kompresji zależy to od ułożenia pixeli w obrazku

ii. Dla pozostałych plików można wnioskować analogicznie jak z poprzedniego porównania.

7.1.3. **LZN vs LZP**

Te właściwości można wywnioskować z dwóch powyższych porównań. LZN będzie kompresją lepszą od LZP, jeśli chodzi o stopień kompresji i gorszą, jeśli chodzi o czas.

7.1.4. **LZE**

e. Czas kompresji
   i. W każdym przypadku algorytm ten wykonuje się dłużej niż pozostałe algorytmy, często wielokrotnie dłużej. Związane jest to z większym kosztem poszukiwania elementów w słowniku, które pasowałyby do wczytanej kolejnej porcji danych z pliku kompresowanego. Na pewno duże znaczenie ma tu sposób zapamiętania słownika w pamięci a tym samym sposób jego przeszukiwania. Tak więc wydaje się, że skrócenie czasów kompresji będzie można osiągnąć poprzez lepszą implementację algorytmu.
   ii. Czas kompresji i również jej stopień często różni się pomiędzy plikami tego samego typu i podobnego rozmiaru. Jeden plik może być kompresowany kilkukrotnie dłużej od innego pliku o podobnej wielkości.

f. Stopień kompresji
   i. Zaproponowany algorytm spisuje się dobrze przy kompresji **danych tekstowych lub dokumentów** małej wielkości – tzn. słownik nie jest jeszcze zapełniony. W tych przypadkach czasami widoczna była jego przewaga nad pozostałymi testowanymi algorytmami. Powodem tego jest powtarzalność dłuższych ciągów znaków, gdyż algorytm charakteryzuje się możliwością szybkiego (po dodaniu niewielkiej ilości elementów do słownika) stworzenia w słowniku stosunkowo długich ciągów. Zapełnienie się słownika jest momentem krytycznym algorytmu, który od tej pory przybliża się w wynikach testów do LZW. Modyfikacja opróżniająca słownik pozwoliłaby osiągac dobre wyniki bitrate także dla dużych rozmiarów danych.

ii. W przypadku **obrazów** algorytm zazwyczaj sprawuje się gorzej od pozostałych algorytmów, ale podobnie do LZW. Jedynym (pozytywnym) wyjątkiem jest przypadek kompresji czarno-białej bitmapy, kiedy to algorytm osiągał przy kompresji jedne z niższych wartości bitrate. Świadczy to tylko o tym, z jakim rodzajem danych algorytm daje sobie najlepiej radę. Powodem tak słabej kompresji (a zarazem podobnej do tej przy LZW) w ogólnym przypadku jest fakt występowania dużej liczby kombinacji ciągów znaków w kodowanym pliku oraz tylko jednokrotne tworzenia słownika ,który nie jest już później modyfikowany. Sprawia to, że algorytm ten zachowuje się podobnie jak LZW. W takich przypadkach nowy element jest tworzony na podstawie istniejących 2 elementów jakie się udało dopasować, przy czym ten drugi jest często bardzo krótki. W przypadku, gdy ten drugi element jest jednoznakowy mamy do czynienia z przypadkiem dodawania elementu występującego w algorytmie LZW.

iii. W przypadku plików z **dźwiękiem**, tak jak dla obrazów, algorytm osiągał podobne wyniki jak LZW. Można więc wnioskować, że powód jest podobny jak dla obrazów – powstający słownik jest podobny do tego powstałego w algorytmie LZW.

g. Ogólne

i. W przeprowadzonych testów wynika, że pomysł tworzenia nowych elementów słownika z dwóch już istniejących znajduje zastosowanie tylko dla wąskiej grupy danych. Dane tekstowe czy bitmapy o niewielkiej ilości kolorów mogą zostać skompresowane podobnie lub lepiej niż w przypadku pozostałych testowanych algorytmów. Jednak dla większości danych taki algorytm nie sprawdza się. Dodatkową wadą tego algorytmu jest trudność jego implementacji w taki sposób, aby szybkość jego działania była porównywalna np. z algorytmem LZW. To wszystko sprawia, że algorytm ten wypada słabo na tle pozostałych testowanych algorytmów.

7.1.5. **LZD**

h. Czas kompresji

i. Algorytm LZD ma czas kompresji komoresji podobny do LZW w większości przypadków. Czasem parametr ten jest nieco gorszy dla LZD gdy kompresowany jest obraz ,który wymaga wielokrotnego czyszczenia słownika. Z kolei jednak w zwykłym LZW po zapełnieniu słownika cały czas następuje szukanie wzorca w pełnym słowniku , natomiast w LZD co jakiś czas powracamy do słowkia o małym rozmiarze lub właściwie pustego co powoduje,że w tym momencie jest on szybszy od LZW z pełnym słownikiem.

ii. Różnice w czasie działania w bardzo duży sposób zależą od rodzaju danych wejściowych , gdyż to od nich zależy częstotliwość czyszczenia słownika.

iii. W testach została użyta wersja gdzie dość często następuje sprawdzanie czy należy wyczyścić słownik.Dopiero później przeprowadziliśmy testy gdzie okazało się,że można by sprawdzać tą możliwość rzadziej ,co poprawia jeszcze parametry czasowe,a właściwie nie pogarsza stopnia kompresji.

i. Stopień kompresji

i. Stopień kompresji dla algorytmu LZD w większości przypadków jest najlepszym jaki udało nam się uzyskać w naszych rozwiązaniach. Lepszy zazwyczaj jest tylko algorytm LZH ale to przez kompresje Huffmana, która dodana do algorytmu LZD mogłaby sprawić,że byłby on jeszcze lepszy.

ii. Z wyjątkiem obrazka tulipany.bmp ,który jest szczególnym przypadkiem bitmapy, z którą nawet kompresja jpg radzi sobie stosunkowo słabo, algorytm LZD daje dość dobre efakty. Lepsze są one oczywiście dla danych o mniejszym rozrzucie wartości wejściowych takich jak obrazy w skali szarości niż kolorowe. Przyczyną tego ,że w przypadku obrazu z tulipanami nie udało się osiągnąć dobrego wyniku jest właśnie to,że charakteryzuje się on bardzo dużą zmiennością. W algorytmie LZD słownik może być czyszczony dopiero gdy się zapełni,a dla tego obrazu

jest to zbyt rzadko.Można by próbować czyścić go częściej ale powoduje to wzrost czasu działania.

iii. Ogólnie jednak algorytm LZD bazuje na próbach dostosowania słownika tak by był on aktualny przez cały czas. Działą więc lepiej niż standardowe algorytmy typu LZW czy LZ77, dla danych gdzie lokalne fragmenty nie mają odwzorowania w danych globalnych. Szczególnie dla dłuższych plików. Dla danych gdzie nie ma takich różnic między danymi lokalnymi, a globalnymi jak np. teksty nie ma już tak dużej różnicy.

j. Możliwości polepszenia

i. W ramach prób polepszenia działania algorytmu gdy zmienność danych jest bardzo duża można by próbować wprowadzić dodatkowy krok czyszczenia słownika jeszcze przed jego zapełnieniem. Trzeba by jednak liczyć dodatkowe współczynniki by nie wykonywać go w normalnych sytuacjach, gdyż dla innego typu obrazów powoduje to pogorszenie wyników.

k. Wpływ różnych parametrów

i. Algorytm LZD testowany był z użyciem różnych parametrów , by sprawdzić jak najlepiej powinny byc dobrane

1. granica czyszczenia (mówi o tym czy już należy wyczyścić słownik) – im mniejsza tym gorszy stopien kompresji bo słownik nie jest czyszczony ale też lepszy czas.Optymalna dla stopnia kompresji jest liczna zero co widać z testów.
2. liczba bitów po kompresji (dodatkowe ograniczenie jak duży ma być słownik po czyszczeniu) – z testów wynika,że gdy się go sztucznie ogranicza poniżej 11 bitów to następuje nieznaczna poprawa czasu ale znaczny spadek kompresji
3. częstotliwość wystąpienia słowa(ile razy jest wykorzystane dane słowo by nie zostało usunięte ze słownika) – po wynikach testów widać, że najlepszą opcją jest zostawianie słów ,które zostały użyte choć raz. Gdy

zwięszamy tę liczbę czas nieznacznie maleje,a le stopień kompresji zachowuje się dość losowo.

4. skok (co ile iteracji sprawdzamy czy w ogóle czyścić słownik) – ten parametr został przetestowany na końcu. Widać,że im rzadziej sprawdzamy tym czas jest lepszy. Kompresja znacząco się nie zmienia dlatego nie zamieściliśmy jej wykresu. Widać jednak z wykresu czasu ,że gdyby reszta testów robiona była dla skoku 100 lub 1000 zamiast 10 to czas byłby mniejszy niż ten z wykresów.

l. Ogólne

i. Próby dążenia do czyszczenia słownika w sposób zależny od danych i uzyskiwanego stponia kompresji wydają się być droga w dobrym kierunku. Dają zdecydowanie lepsze wyniki kompresji niż algorytmu ze stałym słownikiem ,a róznice w czasie są nieznaczne.

7.2. Ogólne podsumowanie zastosowanych technik

1. W całym zestawieniu testów , spośród algorytmów wzorcowych najsłabiej prezentują się standardowe lzw oraz lz77. W przypadku lzw krytyczną cechą jest przepełnienie słownika, dla lz77 – lokalność tworzonego słownika (z zaznaczeniem, że lokalność nie zawsze jest zła – zależy to od rodzaju danych). Powyższe cechy słabiej lub mocniej widoczne są w zaimplementowanych algorytmach. Algorytmy , które nie opróżniają słownika (lwt,lze) tracą siłę kompresji, ze wzrostem kompresowanych danych – zwiększanie liczby bitów słownika może tylko pomóc przesunąć próg zapełnienia słownika.

Wyłania się więc pytanie – jak opróżniać , modyfikować słownik. Z testów wynika,że należy kierować się w stronę przemyślanego podejścia probabilistycznego ponieważ:

- nieopróżnianie słownika ogranicza zastosowanie algorytmu do małych rozmiarów danych, bądź specyficznych typów danych dla których słownik zbudowany na pierwszych próbkach pozostaje

aktualny do końca kompresji (lokalne dane nie różnią się od globalnych np. tekst)

- pełne opróżnianie słownika (lzp), wprowadza znaczną poprawę kompresji do lzw jednak widać, że nie wykorzystuje siły leżącej w rozkładzie danych. Podobnie jest z lz77, w którym zmiana słownika chociaż wykorzystuję lokalną powtarzalność kodowanych fraz, nadal jest sztywna i niepodparda metadanymi.
- wyrzucanie ze słownika najrzadziej występujących fraz (lzn) oraz dynamiczna modyfikacja wielkości słownika połączona z usuwaniem najrzadziej występujących fraz – a więc podejścia uwzględniające rozkład danych przy określaniu konstrukcji słownika ,odnoszą najlepsze wyniki , czasami nawet lepsze od lzh i lv15

2. Zwiększenie liczby bitów słownika pomaga uzyskać lepszy bitrate dla większości przypadków testowych
3. Czas kompresji w zależności od liczby bitów słownika wzrasta – co wiąże się z wielkością słownika i jego przeszukiwaniem. Rozwiązania zaimplementowane przez zespół działają z czasami podobnymi do algorytmów odniesienia, a często nawet szybciej.Jedynie algorytm lze dość wyraźnie odstaje, kompresuje zbyt wolno.
4. Rozważając dla jakich formatów danych wejściowych algorytmy wypadają pomyślnie , widać siłę w kompresji plików w formatach doc i txt, format wav wydaje sie być poza zasięgiem przy zadanych parametrach – tylko algorytm lzd(na 15 bitach słownika) kompresował duże pliki a wynik osiągniety przez lzh pozwala na zaoszczędzenie jeszcze jednego bitu na sybol.
Dla bitmap bmp lepsze wyniki zostały osiągnięte dla skali szarości. Bitmapy 24RGB kompresowały się również poza pewnymi typami np. tulipany.bmp . Widać jednak wyraźnie, że lepiej radziły sobie z nimi algorytmu ,które w jakiś sposób miały możliwiść uaktualniania słownika po zapełnieniu go.
5. Podsumowując, przebadaliśmy dokładnie różne drogi poprawy algorytmu LZW oraz przetestowaliśmy zarówno nasze modyfikacje jak i już implementacje już istniejących algorytmów. Dało nam to duże pojęcie o tym jak można wpływać na poprawę stopnia kompresji oraz, że należy znaleźc dobry stosunek między jakością kompresji , a czasem jej trwania.