Janusz JABŁOŃSKI, Julita TUROS
UNIWERSYTET ZIELONOGÓRSKI, WYDZIAŁ MATEMATYKI INFORMATYKI I EKONOMETRII

Efektywna metoda przetwarzania macierzy rzadkich

Dr inż. Janusz JABŁOŃSKI

W latach 1994 – 2004 asystent a od 2003 adiunkt w Instytucie Informatyki i Elektroniki Uniwersytetu Zielonogórskiego. Od 2004 adiunkt Wydziału Matematyki, Informatyki i Ekonometrii UZ. Prowadzi zajęcia z Inżynierii Oprogramowania. Posiada również doświadczenie dydaktyczne w zakresie architektury komputerów oraz logiki programowalnej. Prowadzi badania związane z wykorzystaniem arytmetyki resztowej oraz układów FPGA w akceleracji obliczeń.

e-mail: J.Jablonski@wmie.uz.zgora.pl

Mgr Julita TUROS

W latach 1997 – 2001 uczennica Liceum Ekonomicznego w Zespole Szkół Spożywczych w Nowej Soli. W latach 2001r. do 2006 r. studentka Uniwersytetu Zielonogórskiego, kierunek Informatyka i Ekonometria, specjalność Systemy Informacyjne. Od stycznia 2007r. niepracująca absolwentka Uniwersytetu Zielonogórskiego.

e-mail: julitkaturos@interia.pl

Streszczenie

Wiele problemów optymalizacyjnych sprowadzanych jest do przetwarzania macierzy rzadkich o bardzo dużych rozmiarach. Tradycyjne metody operacji na macierzach o tak dużych rozmiarach jest czasochłonne dlatego poszukiwane są rozwiązania gwarantujące większą efektywność i krótszy czas przetwarzania. W artykule omówiono wybrane metody i zaproponowano autorską metodę wykorzystania kompresji w poprawie efektywności przetwarzania macierzy rzadkich.

Słowa kluczowe: kompresja, macierze rzadkie.

Effective method of processing of sparse matrix

Abstract

Many optimization problems can be imported to processing of thin matrices about very large sizes. The traditional methods of operation on the matrices about so it is the large sizes time-consuming therefore the guaranteeing the larger efficiency solutions be in the demand and the shorter time of processing. This paper presented the author's method of utilization in improvement of efficiency the compression of processing of sparse matrix in presented study was introduced.

Keywords: compression, sparse matrix.

1. Wstęp

Postęp technologiczny w dziedzinie rejestracji obrazu i dźwięku, połączony z gwałtownym rozwojem Internetu pozwala na wytworzenie oraz przechowywanie ogromnych ilości informacji. Jednakże, ograniczenia pojemności pamięci, przepustowości kanałów transmisyjnych lub mocy obliczeniowej komputerów są często spotykanymi problemami w przetworzeniu tych danych w rozsądnym czasie.

Obecnie szereg zagadnień teorii sieci jak również problemów optymalizacyjnych - niemających rozwiązania dokładnego, sprowadzanych jest do znajdowania najkorzystniejszego ze względu na zadane kryterium rozwiązywania lub rozwiązania najbliższego dokładnemu – określanego jako optymalne. Takie problemy występują w wielu problemach naukowych ale napotykane są również w medycynie (tomografia komputerową, radioterapia) [12] oraz innych dziedzinach takich jak meteorologia, astronomia, poligrafia, grafika komputerowa [5]. Rozwiązania problemów optymalizacyjnych, specyfikowane rozbudowanymi układami równań, często sprowadzają się do przetwarzania macierzy dużych wymiarów (rzędu kilka lub kilkadziesiąt tysięcy). Wówczas składowanie oraz przetwarzanie (przesyłanie, czy też inne operacje) tak dużych zbiorów danych jest bardzo zasobo i kosztochłonne. Jedną z możliwych dróg „radzenia" sobie z tego typu problemami – zasobochłonności, jest wykorzystanie metod kompresji.

2. Metody kompresji

Historycznie przykładami kompresji danych są alfabet Morse'a oraz szeroko rozpowszechnione alfabet Braille'a, które powstało w XIX w. Pismo Brail'a umożliwia kompresję opartą na częstości wystąpień całych słów [6]. Kodowanie liter w alfabecie Mores'a odbywa się przy pomocy kropek i kresek, a zmniejszenie rozmiaru tekstu jest konsekwencją przypisania częściej występującym znakom „krótszych" kodów. Dla małej różnorodności zbioru danych występują „seryjnie" powtarzające się elementy, wówczas w kompresji wykorzystywane są metody i algorytmy „kodowania długości serii". Odmienną technikę wykorzystuje się w kompresji mowy. Konstrukcja strun głosowych człowieka decyduje o dźwięku, jaki może być wydawany podczas mówienia. Zamiast transmisji samej mowy, można przesłać informacje o strukturze „skrzynki głosowej", które są wykorzystane w odtworzenia rozmowy. Informacje o „skrzynce głosowej" wymagają mniej pamięci aniżeli wartości związane z próbkowaniem mowy [6]. Jednym z pierwszych przykładów zastosowania tej metody kompresji było urządzenie o nazwie vocoder (ang. voice coder). Obecnie powszechnie wykorzystywanymi metodami eliminacji nadmiarowych danych jest wykorzystanie transformacji. Transformaty jako metody polegające na przejściu z dziedziny czasu w dziedzinę częstotliwości (tzw. Transformaty: FFT, DCT, Falkowe, lub metody widmowe) są efektywnymi metodami w przetwarzaniu dźwięku oraz obrazu [1]. Proste przykłady wykorzystania transformat to generowanie głosu w „mówiących zabawkach", czy transmisja głosu poprzez mobilne radio lub kompresja obrazów metodą JPEG.

Obecnie coraz częściej kompresja danych, będąca przez wiele lat przedmiotem zainteresowań stosunkowo niewielkiej grupy naukowców, jest wykorzystywana w bardzo wielu dziedzinach a efektywne metody przetwarzania danych były, są i jest bardzo prawdopodobnie, iż nadal będą niezwykle interesującym oraz aktualnym zagadnieniem.

Większość nowoczesnych systemów wymiany informacji, aby zachować swą „atrakcyjność" wymaga zastosowania skutecznej w danym przypadku metody kompresji danych. Często w celu zmniejszenia współczynnika zasobochłonności stosowane jest łączenie technik kompresji i metod kodowania, jako przykład można podać metodę kompresji i kodowania sygnału wizyjnego, fonii i danych dodatkowych w standardzie MPEG. Istotnym aspektem jest tu także odniesienie korzyści finansowych związanych z oszczędnościami czasu i nośnika [5].

2.1. Kompresja bezstratna

W ogólnym przypadku algorytmy kompresji można podzielić na dwie klasy różniące się wymaganiami związanymi z rekonstrukcją informacji. Kompresja bezstratna, w której wymagane jest, aby odtworzone i oryginalne dane były identyczne, oraz kompresja stratna, gdzie dopuszcza się, aby dane zrekonstruowane różniły się (w kontrolowany sposób) od danych oryginalnych.

Jedną ze stosowanych współcześnie metod kompresji bezstratnej jest metoda RLE (ang. Run Length Encoding). Jest to prosty i skuteczny algorytm sprawdzający się przy kompresji bloków danych, w których występują obok siebie identyczne znaki, jak w plikach graficznych. Metoda RLE jest nieefektywna w kodowaniu ciągów poniżej trzech znaków, efektywne jest kodowanie ciągów od czterech występujących po sobie znaków wzwyż [4].

W metodzie Huffmana wykorzystywane są zasady zbliżone do RLE. Pomysł polega na przypisywaniu kodów o różnej liczbie bitów poszczególnym elementom kompresowanego zbioru danych na podstawie częstości ich występowania – krótsze kody częściej występującym znakom. Przy kompresji strumienia danych stosowany jest efektywniejszy tzw. dynamiczny algorytm Huffmana, który uaktualnia, w miarę napływania informacji, określoną liczbę znaków, budowane przez algorytm drzewo binarne reprezentujące rozkład częstości występowania elementów w zbiorze wejściowym. Metoda ta, okazała się na tyle efektywna, że w latach 80. konstruowano na jej podstawie układy sprzętowe służące do zwiększania pojemności dysków twardych w stacjach graficznych [4]. Metody Huffmana są szeroko stosowane w większości programów kompresujących zmniejszających rozmiar danych wejściowych nawet o 20-40% [1]. Ich atutem jest duża efektywność niezależnie od typu danych wejściowych oraz fakt, iż nie bazują na zależnościach pomiędzy kolejnymi danymi. Najczęściej algorytmy Huffmana wykorzystuje się w połączeniu z innymi metodami kompresji. W standardzie JPEG, w którym wykonywane są operacje na macierzach, algorytm ten wykorzystywany jest dopiero w ostatnim etapie kompresji, gdzie wcześniej strumień danych poddaje się kwantyzacji oraz uporządkowaniu „zig-zag". Wadą metod Huffmana jest ich czasochłonność, szczególnie przy odmianie adaptacyjnej, wymagającej skomplikowanych operacji na strukturze drzewa. Dodatkowy problem dotyczy podstawowej jednostki informacji (bit), którego nie można bezpośrednio zapisywać lub odczytywać, lecz „odzyskiwać" z większej jednostki informacji, czyli słowa (bajt) za pomocą operacji logicznych. Ponadto, są to metody bardzo czułe na zakłócenia w transferze danych, błędne odczytanie tylko choćby jednego bitu może spowodować nieodwracalne błędy przy dekompresji.

Z szerokiej gamy stosowanych metod kompresji szczególny przypadek stanowią formy reprezentacji macierzy w szczególności macierzy rzadkich [3] odzwierciedlających dane wejściowe w zagadnieniach optymalizacyjnych oraz z teorii sieci.

Celem kompresji jest zmniejszenie rozmiaru przekazywanej informacji, która w pewnym sensie jest formą zapisu macierzy pozwalającą na oszczędność zajmowanej pamięci oraz czasu przetwarzania. Złożoność obliczeniowa prostej operacji dodawania dwu macierzy A i B rozmiaru n wynosi $O(n^2)$ natomiast dla macierzy rzadkich i odpowiedniej reprezentacji tylko $O(nzA + nzB)$, gdzie nzX – liczba niezerowych elementów macierzy.

Spośród wielu znanych form reprezentacji macierzy rzadkich, jak dotąd tylko kilka wprowadza znaczące ograniczenie rozmiaru danych – czyli również rozmiaru pamięci oraz czasu wymaganego do przetwarzania z wykorzystaniem dostępnych urządzeń [8]. W dalszej części przedstawione zostały metody, które są najczęściej wymieniane, stanowiące podstawę propozycji autorskiego opracowania.

2.2. Kompresja macierzy rzadkich

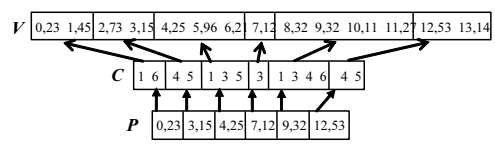
Formy zapisu macierzy rzadkich - zawierających wiele elementów (zerowych) nie wpływających na wynik przetwarzania, pozwalają na oszczędność zajmowanej pamięci [9]. Można to zilustrować następującym przykładem: przechowywanie w pamięci komputera macierzy kwadratowej o rozmiarze 1000x1000 w postaci tablicy dwuwymiarowej wymaga pamiętania miliona elementów, z których w zależności od formatu każdy zajmuje od 4 do 16 bajtów. Przy założeniu, iż każdy z elementów zajmuje 4 bajty, tablica zajmuje 4MB. Zakładając, że będzie to macierz rzadka z 4000 tysiącami elementów znaczących (różnych od zera) wymagane jest zaledwie 16kB pamięci. Jednakże, wymagane są dodatkowe zasoby związane z informacją na temat lokalizacji w tablicy poszczególnych wartości.

Kompresowane za pomocą metody CRS (ang. *Compressed Row Storage*) lub analogicznie CSC (ang. *Compressed Column Storage*) macierze rzadkie przechowujące dane w tablicy dwuwymiarowej (rys.1), zastępowane są trzema tablicami (rys.2) jednowymiarowymi [2, 11]. Tablica V (rys. 2) jest tablicą wartości, w której przechowywane są kolejne niezerowe elementy macierzy. Jej rozmiar jest równy liczbie niezerowych elementów macierzy A (rys. 1). Tablica kolumn C (rys. 2) przechowuje informacje, w której kolumnie macierzy A (rys. 1) znajduje się każda z niezerowych wartości. Indeksy w tablicy kolumn C odpowiadają indeksom w tablicy wartości V, tablice te mają takie same rozmiary. Tablica P zawiera informacje o tym, który z kolei niezerowy element macierzy, (czyli która wartość z tablicy V) jest pierwszą niezerową wartością w każdym z wierszy macierzy A. Rozmiar tej tablicy P jest równy liczbie wierszy w macierzy A.

$$A = \begin{bmatrix} 0{,}23 & 0 & 0 & 0 & 0 & 1{,}45 \\ 0 & 0 & 0 & 2{,}73 & 3{,}15 & 0 \\ 4{,}25 & 0 & 5{,}96 & 0 & 6{,}21 & 0 \\ 0 & 0 & 7{,}12 & 0 & 0 & 0 \\ 8{,}32 & 0 & 9{,}32 & 10{,}11 & 0 & 11{,}27 \\ 0 & 0 & 0 & 12{,}53 & 13{,}14 & 0 \end{bmatrix}$$

Rys. 1. Obraz macierzy A dla metody CRS
Fig. 1. View of matrix A for CRS method

Ponieważ, zwykle wierszy jest mniej niż niezerowych wartości, to występuje tu pewna oszczędność pamięci (w stosunku do tego, gdyby przechowywać pełne współrzędne każdego niezerowego elementu macierzy - zarówno numer kolumny, jak i wiersza).



Rys. 2. Graficzna reprezentacja formatu CRS
Fig. 2. View of CRS format

Efektywniejszym, ze względu na rozmiar danych, jest wykorzystanie formatu RWEI (ang. *Row-wise Ellpack-Itpack format*). Metoda ta polega na przechowywaniu w pamięci komputera wartości elementów niezerowych macierzy wraz z numerami kolumn, w których one występują, wykorzystując do tego celu dwie tablice dwuwymiarowe [7]. W tablicy V (rys. 4) zapisane są wartości niezerowych elementów macierzy B (rys. 3), natomiast tablica C przechowuje numer kolumny, w której odpowiedni element występuje.

$$B = \begin{bmatrix} 732 & 251 & 0 & 0 & 0 & 0 \\ 789 & 213 & 324 & 0 & 0 & 0 \\ 0 & 930 & 596 & 0 & 0 & 0 \\ 210 & 0 & 231 & 350 & 421 & 0 \\ 0 & 659 & 0 & 0 & 897 & 127 \\ 0 & 0 & 873 & 0 & 514 & 534 \end{bmatrix}$$

Rys. 3. Macierz B dla metody RWEI
Fig. 3. Matrix B for RWEI format

$$V = \begin{bmatrix} 732 & 251 & - & - \\ 789 & 213 & 324 & - \\ 930 & 596 & - & - \\ 210 & 231 & 350 & 421 \\ 659 & 897 & 127 & - \\ 873 & 514 & 534 & - \end{bmatrix} \qquad C = \begin{bmatrix} 1 & 2 & - & - \\ 1 & 2 & 3 & - \\ 2 & 3 & - & - \\ 1 & 3 & 4 & 5 \\ 2 & 5 & 6 & - \\ 3 & 5 & 6 & - \end{bmatrix}$$

Rys. 4. Graficzna reprezentacja formatu RWEI
Fig. 4. View of RWEI format

3. Proponowane rozwiązanie

Na podstawie analiz metod kompresji macierzy rzadkich, w szczególności metod: CRS i RWEI można zaproponować zmodyfikowany format kompresji MRWEI (ang. *Modified Row-Wise Ellpack-Itpack*) dedykowany macierzom rzadkim. Wprowadzając do RWEI modyfikację zaczerpniętą z CRS, nie trzeba budować odrębnych macierzy V i C, ale utworzyć na podstawie macierzy wejściowej dwie tablice jednowymiarowe, czyli dwa wektory V i CI, które będą przechowywały potrzebne informacje o elementach niezerowych.

Proponowana metoda jest dwuetapowa i polega na:

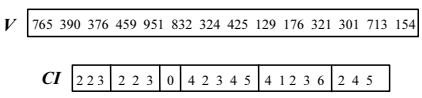
1) wczytywaniu wierszami i zapisywaniu wartości elementów niezerowych macierzy rzadkiej (rys. 5) do wektora V (rys. 6), którego rozmiar nie będzie przekraczał nxn, gdzie n- liczba wierszy (kolumn),
2) utworzeniu wektora CI (rys. 6) o rozmiarze nieprzekraczającym $n*n+n+1$, w którym są zapisywane: liczba elementów niezerowych w wierszu oraz indeksy kolumn, w których owe elementy niezerowe występują, ostatni wyraz wektora zawiera liczbę wierszy macierzy wejściowej, stąd rozmiar wektora.

Ostatni element wektora CI (rys. 6) jest opcjonalny, ponieważ rozmiar można przyjąć jako wartość największego współczynnika w tym wektorze.

$$A = \begin{bmatrix} 0 & 765 & 390 & 0 & 0 & 0 \\ 0 & 376 & 459 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 951 & 832 & 324 & 425 & 0 \\ 129 & 176 & 321 & 0 & 0 & 301 \\ 0 & 0 & 0 & 713 & 154 & 0 \end{bmatrix}$$

Rys. 5. Obraz macierzy A dla metody MRWEI.
Fig. 5. View of matrix A for MRWEI method.

Ponieważ format CRS wymaga przechowywania w tablicy P (rys. 2) wartości elementu początkowego, która z reguły wymaga formatu zmiennoprzecinkowego, (czyli więcej pamięci aniżeli typ integer), więc uzasadnione jest przypuszczenie, iż wymagane będzie mniej zasobów pamięci na zapis macierzy w formacie MRWEI.



Rys. 6. Graficzna reprezentacja formatu MRWEI
Fig. 6. View of MRWEI format

Ponadto można oczekiwać, iż przetwarzanie dwóch tablic pozwoli na korzystanie ze zmiennych typu „register", co również powinno wpływać na zwiększenie efektywności przetwarzania.

4. Wyniki eksperymentalne

W celu weryfikacji oraz sprawdzenia efektywności proponowanego rozwiązania przygotowane zostały odpowiednie aplikacje implementujące zaproponowane rozwiązanie.

Zestawienie otrzymanych wyników przetwarzania w środowisku Matlab, dla macierzy T o rozmiarze nxn = 1000x1000 oraz współczynniku wypełnienia 0,6 ÷ 0,9, przedstawia tab. 1.

Tab. 1. Wyniki eksperymentalne Matlab
Tab. 1. Experiment results in Matlab

Wsp. wyp macierzy	Czas kompresji *[s]*		Czas mnożenia *[s]*	
	CRS	MRWEI	CRS	MRWEI
0,6	0,147	0,112	1490	1455
0,7	0,142	0,110	1056	880
0,8	0,117	0,066	679	372
0,9	0,104	0,057	355	197

Tabela tab. 2 przedstawia uzyskane wyniki związane z czasem przetwarzania macierzy T w środowisku C++.

Tab. 2. Wyniki eksperymentalne w C++
Tab. 2. Experiment results in C++

Wsp. wyp macierzy	Czas kompresji *[s]*		Czas mnożenia *[s]*	
	CRS	MRWEI	CRS	MRWEI
0,6	0,156	0,124	48,7	49,3
0,7	0,133	0,115	43,2	44,0
0,8	0,115	0,104	37,8	36,0
0,9	0,109	0,097	25,3	25,4

Testy zostały przeprowadzone na komputerze klasy PC (Pentium HT IV 3.0 GHz z 2GB RAM).

5. Podsumowanie

Metody kompresji wykorzystywane w poprawie efektywności przetwarzania macierzy rzadkich opierają się na podobnych zasadach, tj. większość metod przekształca zapis macierzy z postaci dwuwymiarowej tablicy do postaci kilku wektorów lub kilku list, w których przechowywane są niezbędne informacje na temat wartości niezerowych elementów macierzy oraz informacji pozwalających na jednoznaczną lokalizację elementów tych elementów w tablicy (tzn. elementów znaczących, których współrzędne ulokowania mają wpływ na postać macierzy wynikowej).

Analiza otrzymanych wyników, dla zaproponowanej metody, potwierdza rozważania teoretyczne. Zaproponowana metoda kompresji MRWEI pozwala na uzyskanie zmniejszenia rozmiaru przetwarzanych danych oraz pozwala na uzyskanie poprawy efektywności przetwarzania. W przypadku macierzy o rozmiarze 1000x1000 i wypełnieniu elementami zerowymi 60÷90%, kompresja wpływa znacząco na zmniejszenie zasobów niezbędnych w celu składowania oraz transmisji informacji, jak również może mieć wpływ na przyspieszenie operacji mnożenia. Ponadto, jak wskazują wyniki zaprezentowane w tab.1. i tab. 2. kompresja wykorzystująca zaproponowane rozwiązanie może być realizowana znacznie szybciej, przy czym, współczynnik wypełnienia elementami zerowymi ma duże znaczenie. Wraz ze wzrostem wypełnienia elementami zerowymi te różnice zmniejszają się oscylując w granicach 20%. Otrzymane wyniki pozwalają przypuszczać, iż zaproponowane rozwiązanie może znaleźć zastosowania.

6. Literatura

[1] Heim K.: Metody kompresji danych, MIKOM, Warszawa 2000.
[2] Herman G. T.: Image Reconstruction From Projection: The Fundamentals of Computarized Tomography (Aaemic Press 1980).
[3] http://www-math.cudenver.edu/~jmandel/mri/Parallel-sparse-matrix.pdf.
[4] Kuniszewski S.: Wielki ścisk, artykuł z magazynu „Chip" nr 05/2004.
[5] Przelaskowski A.: Kompresja danych obrazowych. Zarys zagadnień istonych. Politechnika Warszawska.
[6] Sayood K.: Kompresja danych, Wydawnictwo RM, Warszawa 2002.
[7] Wyrzykowski R., Zrównoleglenie algorytmów przetwarzania macierzy rzadkich, Politechnika Częstochowska.
[8] Duff, I. S.: M. A. Heroux, and R. Pozo. (2002). An Overview of the Sparse Basic Linear Algebra Subroutines: The New Standard from the BLAS Technical Forum," ACM Transactions on Mathematical Software, 28, 239-267.
[9] Im E. J.: Optimizing the performance of sparse matrix-vector multiplication. Ph.D. thesis, University of California, May 2000.
[10] Roger Koenker and Pin Ng. SparseM: A sparse matrix package for R. J. of Statistical Software, 8 (6), 2003.
[11] D'Azevedo, E.F., Fahey, M.R., Mills, R.T.: Vectorized sparse matrix multiply for compressed row storage format. In: Proceedings of the 5th International Conference on Computational Science, Atlanta, USA, Springer-Verlag (2005).
[12] Boldi P. and Vigna S.: The WebGraph framework I: Compression techniques. In The Thirteenth International World Wide Web Conference, pages 595–602, New York, NY, USA, 2004.

Artykuł recenzowany