STUDIA INFORMATICA 2001
Volume 22 Number 3 (45)

Marcin BRODKA
Politechnika Śląska, Biblioteka Główna

BEZSTRATNY ALGORYTM PROGRESYWNEJ TRANSMISJI OBRAZÓW

Streszczenie. W artykule przedstawiono bezstratny algorytm progresywnej transmisji obrazów. Pokazano sposób redukcji liczby obliczeń w algorytmie oraz uzupełniania obliczeń wcześniejszych lub członów obliczeń zredukowanych na wcześniejszym etapie. Zaprezentowano łatwy sposób przechodzenia między kolejnymi etapami algorytmu transmisji. Metoda może być stosowana do szybkiego, progresywnego kodowania obrazów.

LOSSLESS ALGORITHM FOR PROGRESSIVE IMAGE TRANSMISSION

Summary. A lossless progressive algorithm of image transmission is presented in this paper. The paper also incorporates a method for calculation reduction using pruning and for completes of previous transform results with reduced calculation effort. An easy pass between subsequent transformation levels is also presented. This method can be applied to fast progressive image coding and another type of FFT transform.

1. Opis metody

Progresywna transmisja obrazów (*ang. PIT– Progressive Image Transmission*) związana jest z budowaniem obrazu po stronie odbiorczej (oglądaniem) przez progresywne poprawianie jego wierności. Na każdym poziomie wierności (dokładności) interaktywny obserwator decyduje o konieczności dalszych poprawek lub o przerwaniu budowy obrazu i odpowiednio informuje o tym stronę nadawczą. Transmisja PIT może być na przykład stosowana w systemie przeglądania obrazów pracującym w kanale transmisji o małej przepustowości [5].

Transmisja PIT transformaty sygnału odbywa się zazwyczaj w następujący sposób. Najpierw każdy blok pikseli wysyłanego obrazu poddawany jest odpowiedniej pełnej transformacji (*2D*). Przetransformowane i poddane kwantyzacji współczynniki są następnie zapamiętywane w buforze o rozmiarach dopasowanych do obrazu. Kody zapamiętanych współczynników transmisji przetwarzane są w trakcie wielokrotnych operacji przejść skanowania odpowiadającym wielu poziomom tworzenia obrazu. Każda operacja skanowania koduje i wysyła nowy zbiór współczynników transformaty z każdego bloku. Przykładem tego rodzaju schematu działania jest koder progresywny standardu JPEG [5] wykorzystujący metodę selekcji widmowej. Metoda ta nie jest efektywna z następujących powodów:

– początkowe transmisje wykorzystują względnie mało współczynników z każdego przetransformowanego bloku, ale transmisja początkowa nie może się rozpocząć, dopóki nie zostaną wyliczone i poddane kwantyzacji wszystkie współczynniki transformat wszystkich bloków,

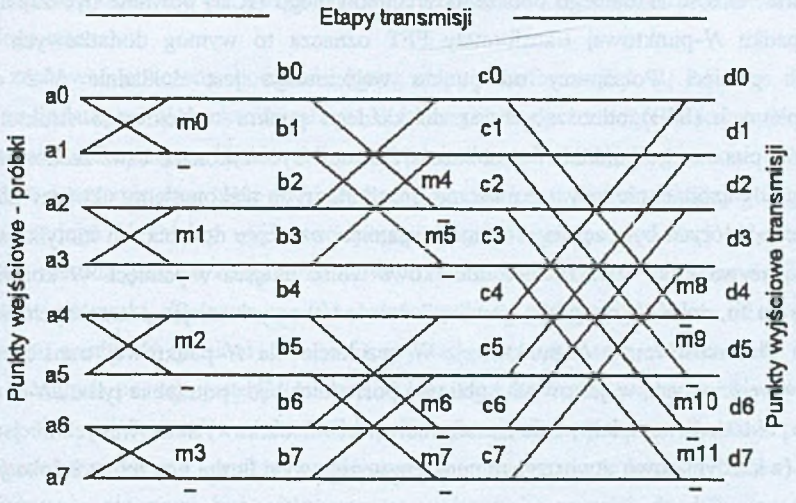
– pełna progresja może nie dojść do skutku w ogóle, jeśli obserwator po stronie dekodującej zdecyduje o przerwaniu kodowania na wczesnym etapie. W tym przypadku wysiłek włożony w wyliczenie większości współczynników zostaje zmarnowany.

Szybkość działania algorytmu progresywnej transmisji obrazów (PIT) można zwiększyć przy założeniu, że niektóre z punktów na wejściu lub wyjściu posiadają zerowe wartości, a ilość informacji użytą w algorytmie można zmniejszyć wykonując trywialnie proste operacje. W następstwie tego transmisję można określić jako zredukowaną, a przyrost prędkości całej operacji będzie zależał od poziomu redukcji. Redukcja może być stosowana w transformacjach takich jak FFT (*ang. Fast Fourier Transform*), DCT (*ang. Discrete Cosine Transform*) itp., pod warunkiem występowania określonych sekwencji zerowych [7], [8].

2. Progresywna transmisja z użyciem transformacji cząstkowej

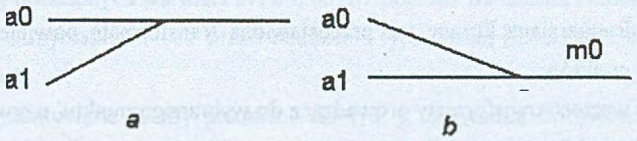
Jednowymiarowa transformata struktury przetwarzania typu PIT dla przypadku 8-punktowego (stosowana również w algorytmie FFT) pokazana jest na rys. 1. Normalny algorytm FFT przeprowadza pełne operacje motylkowe dla każdej grupy motylków każdego etapu. W rezultacie wszystkie wartości wyjściowe generowane są jedne po drugich. Generacja następuje w ostatnim etapie obliczeń motylkowych. Procedura taka nie pozwala na wykonanie obliczeń w sposób klasyczny. Transmisja zredukowana wylicza część wartości wyjściowych poprzez śledzenie linii przejściowych, które prowadzą od punktów (próbek) wejściowych do punktów wyjściowych (wartości). Jednak rozwinięcie redukcji do następnego poziomu jest

problematyczne, ponieważ algorytmy redukcji nie korzystają z wyników uzyskanych wcześniej w etapach pośrednich.



Rys. 1. Obliczenia 8-punktowego algorytmu transmisji z użyciem motylkowego algorytmu FFT

Fig. 1. Computation of an 8–point transmission algorithm using FFT transform



Rys. 2. Niekompletne obliczenia motylkowe (ICB) typu 1(*a*) i typu 2(*b*)

Fig. 2. Incomplete butterflies type 1 (a) and 2 (b)

Z powyższych rozważań wynika, że potrzebny jest algorytm redukcji, który mógłby pracować zarówno opierając się na punktach wejściowych, jak i obliczeniach pośrednich. Aby to osiągnąć, każdy punkt wyjściowy musi być uzyskany poprzez niekompletne obliczenia motylkowe (ICB – *ang. InComplete Butterflies* – patrz rys. 2). Ścieżka dostępu od punktów wejściowych do jednego punktu wyjściowego składa się ze zredukowanych obliczeń motylkowych (ICB) typu 1 (rys. 2*a*) i typu 2 (rys. 2*b*). Obliczenia zredukowane (ICB) pokazane na rys. 2 prowadzą do pierwszego punktu wyjściowego (poziomu zerowego). W uproszczeniu, jeżeli podstawowy moduł obliczenia byłby kompletnym obliczeniem motylkowym (patrz rys. 4), musiałyby być wykonane niektóre dodatkowe obliczenia pośrednie, niepotrzebne natychmiast do obliczenia punktów wyjściowych. Z drugiej strony,

jeżeli obliczenia (ICB) byłyby wykonane dla ścieżki dostępu wcześniejszego punktu, obliczona wartość powinna być użyta, co oznacza, że podczas obliczeń dla każdej wartości wyjściowej otrzymana wartość aktualnego obliczenia zredukowanego (ICB) powinna być zapamiętana. W przypadku N–punktowej transformaty FFT oznacza to wymóg dodatkowych $N\log_2 N$ komórek pamięci. Począwszy od punktu wejściowego jest dokładnie $N-1$ obliczeń niekompletnych (ICB) odnoszących się do każdego punktu wyjściowego struktury FFT, a więc dla pierwszego punktu, $N-1$ obliczeń ICB musi być wykonane bądź zachowane tylko wtedy, gdy wcześniej nie były zapamiętane. Jeżeli algorytm niekompletny ukończy obliczenia motylkowe, których był częścią, 2 komórki pamięci używane do obliczeń motylkowych nie będą już używane, co stwarza dwa dodatkowe wolne miejsca w pamięci. W konsekwencji wygląda na to, że $N-1$ jest górną granicą ilości dodatkowych miejsc potrzebnych w trakcie obliczeń dla transformaty N–punktowej. W rezultacie dla N–punktowej transformaty dla punktów wejściowych, wyjściowych i obliczeń pośrednich będą potrzebne tylko $2N$–punktowe matryce. W trakcie kompletowania transformaty liczba dodatkowych używanych miejsc maleje do zera (z każdym nowo utworzonym punktem wyjściowym liczba potrzebnych lokacji maleje o jeden).

3. Algorytm Rangarajana [4]

Algorytm Rangarajana liczący tak przedstawioną transformatę powinien przebiegać wg następujących punktów:

– obliczyć wartość transformaty prowadzącą do wybranego punktu, używając do tego celu niekompletne obliczenia motylkowe typu *a*,

– jeżeli ICB będący w ścieżce dostępu dla punktu wyjściowego był wcześniej obliczony, należy podstawić zapamiętany wynik,

– zapamiętać obliczoną wartość dla każdego aktualnego niekompletnego obliczenia motylkowego (ICB) w ścieżce dostępu prowadzącej do punktu wyjściowego,

– w przypadku, kiedy ICB kończy obliczenia motylkowe, 2 miejsca zawierające informacje wyjściowe nie będą więcej używane, a więc utworzone zostaną dwa nowe wolne miejsca.

Ta metoda generacji kodów wyjściowych na podstawie zawartości pamięci stanu poprzedniego oraz niekompletnych obliczeń motylkowych znana jest jako redukcja sukcesywna (SP – *ang. Succesive Pruning*). W głównym algorytmie zastosowano potrójnie zagnieżdżoną pętlę wykorzystującą standardowe algorytmy FFT, ale warunki rozgałęzień na poszczególnych poziomach są różne. Normalny algorytm transmisji jest algorytmem skomplikowanym, oblicza bowiem transformaty wykonując pętle obejmujące liczbę stanów,

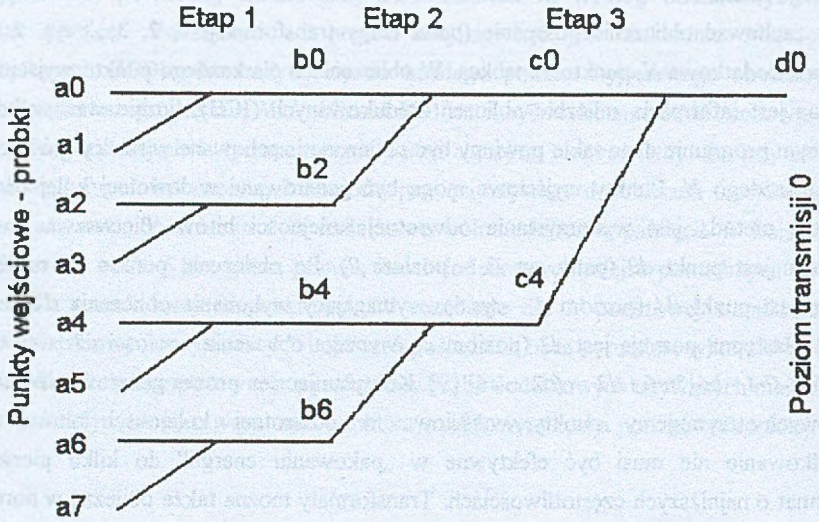
liczbę grup motylków w danym etapie i liczbę motylków obliczanych dla każdej grupy. Przy redukcji sukcesywnej pętle uwzględniają współczynniki obliczone dla każdego poziomu redukcji, liczbę poziomów dla wszystkich współczynników i obliczenia zredukowane (ICB) dla każdego stanu.

Aby zachować obliczenia pośrednie (patrz etapy transformacji 1, 2, 3, – rys. 2.), jest wymagana dodatkowa *N*–punktowa tablica. W obliczeniach dla każdego punktu wyjściowego niezbędna jest informacja o liczbie obliczeń zredukowanych (ICB), liczbie stanów itd. We wdrożonym programie dane takie powinny być obliczone i zachowane w tablicy głównej jako stałe dla każdego *N*. Punkty wyjściowe mogą być generowane w dowolnej kolejności, ale najszybszą metodą jest wykorzystanie odwrotnej kolejności bitów. Pierwszym punktem obliczanym jest punkt *d0* (patrz rys. 3 – poziom *0*). Po obliczeniu punktu *d0* następnym punktem jest punkt *d4* (poziom *4* – rys.5), wymagający wykonania obliczenia *d4 = (c0 - c4)*m8*. Następną pozycją jest *d2* (poziom *2*), wymaga obliczenia wartości *c2 = (b0 - b2) *m4, c6 = (b4 - b6)*m6 i d2 = (c2 + c6)* [9]. Kontynuując ten proces generowania punktów wyjściowych otrzymujemy punkty wyjściowe w odwrotnej kolejności bitów. Takie uporządkowanie nie musi być efektywne w „pakowaniu energii" do kilku pierwszych transformat o najniższych częstotliwościach. Transformaty można także obliczać w normalnej kolejności. W takim przypadku po *d0* obliczone będzie *d1*, a zachowane obliczenia pośrednie nie są natychmiast potrzebne do obliczenia *d0*. Transformata *d1* musi być obliczona za pomocą wcześniej obliczonych wartości *b1*, *b3*, *b5*, *b7* poprzez obliczenia zredukowane (ICB) typu 2, następnie otrzymanie *c1* i *c5* przez obliczenia typu 2 i na koniec otrzymanie *d1* przy pomocy ICB typu 1.

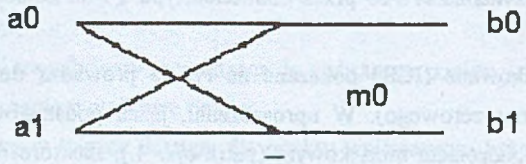
Obliczenia zredukowane (ICB) pokazane na rys. 2 prowadzą do pierwszego punktu wyjściowego (poziomu zerowego). W uproszczeniu, jeżeli podstawowy moduł obliczenia byłby kompletnym obliczeniem motylkowym (patrz rys. 4.), niektóre dodatkowe obliczenia pośrednie, niepotrzebne natychmiast do obliczenia punktów wyjściowych, musiałyby być jednak wykonane. Z drugiej strony, jeżeli obliczenia (ICB) byłyby zrealizowane dla ścieżki dostępu wcześniejszego punktu, obliczona wartość powinna być użyta, co oznacza, że podczas obliczeń dla każdego punktu wyjściowego powinna być zapamiętana wartość aktualnego ICB [9].

Różne techniki obliczeń FFT można teraz zróżnicować i porównać. Załóżmy, że operacja kodowania jest modelowana przez procedurę transformacyjną *proc()* (dwa puste nawiasy informują program, że funkcja ta nie pobiera żadnych argumentów), pracującą na podstawie *N* – punktowej tablicy wejściowej, co daje jednowymiarową *N* – punktową transformację typu FFT tablicy wyjściowej od *0* do *N–1*. W normalnej pełnej transformacji funkcja transformująca dawałaby wszystkie punkty w tablicy wyjściowej (od *0* do *N–1*). W algorytmie transformacji

zredukowanej byłoby to modelowane od 0 do q punktów, gdzie q jest mniejsze od N. W algorytmie redukcji sukcesywnej zarówno początek, jak i koniec strefy bądź grupy obliczanej są zmienne, generalnie w tablicy wyjściowej jest obliczane od p do q punktów [8].



Rys. 3. Transformacja prowadząca do wyznaczenia składowej stałej – poziom 0
Fig. 3. Transformation leading to DC coefficient – level 0

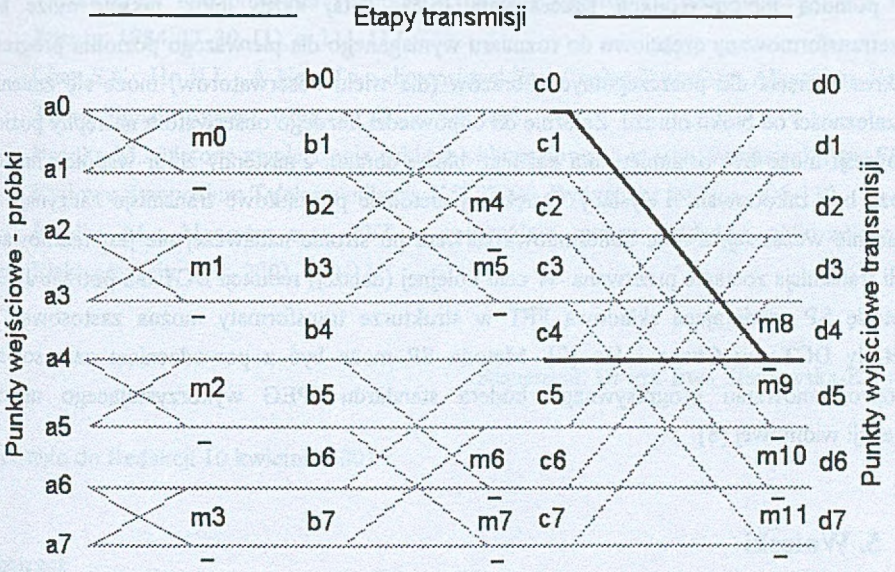


Rys. 4. Kompletne obliczenia motylkowe
Fig. 4. Complete butterflies

Tablica 1 przedstawia zestawienie nakładu obliczeniowego dla algorytmów jednowymiarowych (1–D). Przy założeniu, że czas dodawania jest taki sam, jak czas mnożenia i są to czasy stanowiące jeden cykl zegarowy, normalnie uporządkowana redukcja sukcesywna – NOSP (*ang. Normal Ordered Succesive Pruning* – redukcja sukcesywna z normalną kolejnością bitów) zabiera mniej jednostek czasu wymaganych do osiągnięcia każdego poziomu redukcji niż zwykła, pełna transformacja FFT.

Dlatego algorytm SP może być używany do uzupełnienia transformacji zredukowanej bądź do przejścia z jednego poziomu redukcji do drugiego. W obliczeniach pełnej transformacji algorytm uzupełnia transformaty w takim czasie, jak algorytm normalnej FFT. Porządek

NOSP może być bezpośrednio odwrotnie transformowany przez redukcję odwrotną. Porządek BOSP (*ang. Bit reversal Ordered Succesive Pruning* – redukcja sukcesywna z odwróconą kolejnością bitową) może być odwrotnie transformowany przez schemat szybkiej rekonstrukcji Takikawy [6] bądź odwrotną redukcją sukcesywną.



Rys. 5. Graf przepływowy prowadzący do wyliczenia 4 transformaty
Fig. 5. SFG leading to 4th order transform coefficient

Tabela 1

Nakład obliczeniowy dla algorytmów jednowymiarowych

Jednostki czasu potrzebne na przeprowadzenie obliczeń algorytmu transmisji			
Współczynniki Wyjściowe	Redukcja sukcesywna z odwróconą kolejnością bitów (BOSP)	Normalny algorytm redukcji sukcesywnej (NOSP)	Normalny algorytm transmisji (algorytm FFT)
0	7	7	25
1	9	18	26
2	14	23	27
3	16	28	28
4	27	30	30
5	29	32	32
6	34	34	34
7	36	36	36

4. Zastosowanie do progresywnego kodowania obrazów

Lepsze wykorzystanie mocy obliczeniowej w serwerze odpowiedzialnym za kodowanie i szybszy czas przetwarzania z podziałem w czasie dla wielu użytkowników można osiągnąć za pomocą metody redukcji sukcesywnej (SP). Tutaj każdy blok pikseli może być przetransformowany częściowo do rozmiaru wymaganego dla pierwszego poziomu progresji. Zakres obcięcia dla poszczególnych obrazów (dla wielu obserwatorów) może się zmieniać w zależności od bloku obrazu. Zależnie od odpowiedzi każdego obserwatora następny poziom progresji może być osiągnięty dla każdego bloku obrazu, a następny zbiór współczynników może być zakodowany i wysłany. Dzięki tej metodzie początkowe transmisje zaczynają się znacznie wcześniej, a moc obliczeniowa serwera po stronie nadawczej nie jest marnowana, jeśli transmisja zostanie przerwana. W celu kolejnej (dalszej) redukcji DCT dla potrzeb JPEG, metodę SP zawierającą składową FFT w strukturze transformaty można zastosować do metody DCT wg Chana i Ho [7]. Metoda SP może być z powodzeniem zastosowana w oprogramowaniu progresywnego kodera standardu JPEG wykorzystującego metodę selekcji widmowej [8].

5. Wnioski

Przedstawiono metodę zmniejszenia nakładu obliczeniowego we współczesnych algorytmach kodowania i transmisji obrazów. Metody te są ogólnie stosowane dzięki możliwościom redukcji nakładu obliczeń w algorytmach obliczeniowych typu FFT, możliwości wcześniejszego zakończenia redukcji transformat lub – ogólnie – przejścia z jednego poziomu redukcji do drugiego [8]. Przedstawiono zastosowanie tej metody do szybkiej transmisji obrazu, co w dobie intensywnego wykorzystywania sieci komputerowej typu Internet wydaje się problemem ważnym.

LITERATURA

1. Cormen T.H., Leiserson Ch.E., Rivest R.L.: Wprowadzenie do algorytmów. WNT, Warszawa 1998.
2. Oppenheim V., Schafer R. W.: Cyfrowe przetwarzanie sygnałów. WKŁ, Warszawa 1979.
3. Markel J. D.: FFT Pruning. IEEE Trans. on Audio and Elect. Vol. AU-19, Dec. 1971, s. 305-311.

4. Rangarajan S. R., Srinivasan S.: Generalized Method for Pruning an FFT Type of Transform. IEE Proc.-Vis Image Signal Process, Vol. 144, No. 4 Aug.1997, s. 189-192.
5. Wallace G. K.: The JPEG Still Picture Compression Standard. ACM ,.1991, 39,(2), s. 481-185.
6. Takikawa K.: Fast progressive reconstruction of a transformed image. IEEE Trans. Inf. Theory, 1984, IT-30, (1), s. 111-117.
7. Chan S.C., Ho K.L.: A New Two-dimensional Fast Cosine Transform Algorithm. IEEE Trans. Signal Process., 1991, 39, (2), s. 481-485.
8. Brodka M.: Metoda zmniejszenia nakładu obliczeniowego w transformacjach typu FFT. Krajowe Sympozjum Telekomunikacji, KST'2000, Bydgoszcz 2000, s. 106-112.
9. Brodka M.: Algorytm typu FFT o szczególnie małym nakładzie obliczeniowym. Elektronizacja, nr 3, 2001, s. 9-11.

Recenzent: Dr inż. Ewa Starzewska-Karwan

Wpłynęło do Redakcji 10 kwietnia 2001 r.

Abstract

Lossless method for progressive image transmission is proposed. Method is able to prune for a FFT type of transform structure and being able to prune the transmission. Progressive image transmission involves the build-up of an image at the receiving (viewing) and by progressively improving its fidelity. At each level of fidelity, the interactive viewer decided either in favour of favour improvements, or upon aborting the build-up, and informs the transmission end accordingly. PIT can, for example, be applied in an image browsing system designed over a low it rate channel. Transform domain PIT usually proceeds in the following manner. Firstly, each pixel block of the image to be sent is subjected to a suitable full transform. The transformed and quantised coefficients are the stored in an image sized buffer. The entropy codlings of the coefficient and transmission proceed in multiple scans/pass, corresponding to the multiple levels of build-up of the image. Each scan/pass encodes and sends the new set of transform coefficient from each block. This method is show for example with computation an 8-point transform an FFT type of structure. In this paper a more generalised pruning method is proposed. In this paper a more generalised pruning method is proposed an FFT type of structure, which can progress efficiently from one level of transform completion to next level. A single pruned transform or full transform are special cases of this

algorithm. Is show in fig. 1 as 8-points transform using FFT type of structure. Pruned computation to leading DC coefficient shown in fig. 3 and SFG leadind to 4^{th} level transform in fig. 5. Incomplete butterflies is shown in fig. 2. and complete butterflies in fig. 4. The method is generalised of pruning an FFT type of transform can be applied to fast adaptive image compression.