Transformata Fermata

(analiza, implementacja)

Krzysztof Cyran, 218405
Politechnika Wrocławska, wydział Elektroniki
Wrocław, Polska
218405@student.pwr.edu.pl

Paweł Mularczyk, 218190
Politechnika Wrocławska, wydział Elektroniki
Wrocław, Polska
218190@student.pwr.edu.pl

***Streszczenie*—Projekt opisuje analizę Transformaty Fermata w przetwarzaniu wideo oraz implementację algorytmu liczenia transformaty oraz transormaty odwrotnej dla zadanych parametrów, w arkuszu kalkulacyjnym MS Excel.**

Słowa kluczowe — Transformata Fermata, Teoretyczna Transformata Numeryczna, Wideo Przetwarzanie

I. Wprowadzenie

W opisywanym przez nas projekcie głównym zastosowaniem Transformaty Fermata jest usprawnienie procesu filtracji wideo. Transformata Fermata ułatwia proces wykonywania skomplikowanych operacji na dużych liczbach, dzięki użyciu transformaty obliczenia znacznie się ułatwiają. Wsród transformat wyróżnia się wiele ich rodzajów. Transformata Fermata (FNT) jest specjalnym przypadkiem teoretycznej transformaty numerycznej (NTT), a ta zaś bazuje na transformacie Fouriera. Podstawą Transformaty Fermata są tzw. liczby Fermata, które z definicji zapisuje się jako naturalne liczby postaci $F_n = 2^{2^n} + 1$, gdzie n jest nieujemną liczbą całkowitą. Transformata Fermata wykorzystuje jako modulo q odpowiednie liczby Fermata. W celu zachowania odpowiednich właności transformata musi być tworzona z odpowiednio dobranych parametrów, co zostało opisane w punkcie *B.*

A. *Transformata Fermata i jej własności*

Transformata Fermata wyrażana jest wzorem:

$$X_k = \sum_{n=0}^{N-1} x_n \omega^{kn}, \quad k = 0, \dots, N-1$$

Wzór.1

Gdzie w powyższym wzorze N jest długością transformaty, x_n jest sekwencją liczb podanych do transformowania oraz ω jest jądrem transformaty.

Odwrotna transformata Fermata liczona jest według wzoru:

$$x_n = N^{-1} \sum_{k=0}^{N-1} X_k \, \omega^{-kn} \, mod \, q$$

Wzór. 2

B. *Dobór parametrów Transformaty Fermata*

Transformatę Fermata definijuemy jako 3 liczby *N*, *q*, *ω*, gdzie kolejno *N* jest długością transformaty, *q* jest liczbą Fermata, ω jest jądrem transformacji. Ogólny zapis liczby Fermata ma postać:

$$q_t = 2^b + 1, gdzie \; b = 2^t$$

Wiadomo też, że w przypadku dobierania wartości długości transformaty – N, powinno być ono potęgą dwójki:

$$N = 2^m, gdzie \; m \le b$$

Kolejnym warunkiem koniecznym który musi zachodzić jest:

$$\omega^N = 1 \; (mod \; q)$$

Dzięki tym własnościom, jesteśmy w stanie dobrać odpowiednie parametry niezbędne to wykonania Transformaty.

II. Implemntacja

Najważniejszą częścia projektu było zaimplementowanie transformaty Fermata oraz transformaty odwrotnej tak, by po wpisaniu odpowiednich parametrów otrzymać zgodne z prwadą wyniki. Nasze pierwsze próby rozpoczęliśmy w języku C++, gdzie poprawnie zaimplementowaliśmy obliczanie transformaty Fermata. Problemem jednak okazało się liczenie transformaty odwrotnej, a po wielugodzinnych próbach musieliśmy poszerzyć naszą wiedzę o dodatkowe materiały [2], [3]. Dzięki dokładnemu przestudiowaniu artykułów autorstwa Agarwala i Burrusa zrozumieliśmy, że dotychczasowe błędy brały się z niezrozumienia istoty operacji modulo ujemnych potęg.

Z pomocą prowadzącego ustaliliśmy, iż zapis ω^{-1} mod q jest używany do oznaczania odwrotności multiplikatywnej, a więc wynikiem tej operacji jest taka wielokrotność ω, że jej modulo q jest równe 1. Wiedza ta okazała się kluczowa do poprawnej implementacji odwrotnej transformaty Fermata. Na polecenie prowadzącego dokonaliśmy implementacji obliczeń w arkuszu kalkulacyjnym excel. Najistotniejszym fragmentem tej implementacji jest moment obliczania odwrotności multiplikatywnej modulo q. Wykorzystana przez nas metoda zaprezentowana jest poniżej.

PODAJ.POZYCJĘ(1;INDEKS(MOD(WIERSZ(ADR.POŚR("1:"&C4))*POTĘGA(D4;C7*E7);C4);0);0)

gdzie C4 jest adresem liczby q, D4 adresem wartości ω, a iloczyn C7 i E7 jest ujemną potęgą ω (we wzorze na transformatę odwrotną *-kn).*

Posługując się tym sposobem dokonaliśmy poprawnej implementacji transformaty Fermata oraz odwrotnej do niej dla parametrów N=4, q=17, ω=4. Kierując się przykładem przedstawionym przez Agarwala i Burrusa[3] potwierdziliśmy poprawność otrzymywanych wyników. Transformaty liczone w ten sposób zachowały także swoje właściwości. Po obliczeniu transformat dla dwóch ciągów, a następnie dokonaniu ich iloczynu i obliczenia transformaty odwrotnej otrzymany ciąg odpowiadał splotowi dwóch ciągów wejściowych, co wykazane zostało dzięki kalkulatorowi do liczenia splotów[4] (ang. Convolution calculator).

N(długość)	Fn	a (alfa)	m		2^-m
4	17	4	2		13
Ciąg wejściowy	k	Ciąg wyjściowy	Ciąg wyjściowy 2		Transformata odwrotna
2	0	1	1		2
-2	1	10	10		-2
1	2	5	5		1
0	3	9	9		0

Rys. 1 – Zdjęcie z arkusza kalkulacyjnego

W badanym przez nas przykładzie ciągi reprezentowały się następująco: x=(2;-2;1;0), h=(1;2;0;0). Wyliczone wartości transformat to odpowiednio: X=(1;10;5;9), H=(3;9;16;10). Ich iloczyn wynosi Y=(3,5,12,5) (mod 17), a transformata odwrotna do niego to y=(2,2,-3,2). Otrzymany wynik jest identyczny zarówno do tego wyliczonego przy użyciu kalkulatora splotu, jak również zgadza się on z obliczeniami zaprezentowanymi przez autorów.

Kolejnym problemem, którego nie udało się nam przezwyciężyć okazało się liczenie transformaty dla ciągów o długosci dłuższej niż 4. Z racji, iż we wzorze na transformatę znajduje się współczynnik ω^{kn}, należy zauważyć, iż dla ciągu o długości 8 wykładnik potęgi wynosi maksymalnie 49. Taka potęga jest już zbyt duża, by mieścić się w zakresie arkusza kalkulacyjnego. Okazało się także, iż wyliczanie modulo *q* z tej wartości nie daje zamierzonego efektu, otrzymane w ten sposób wartości nie są prawidłowe. Rozwiązaniem byłby tutaj algorytm Radix-2 i diagram motylkowy (ang. butterfly diagram), jednak pomimo wielu prób nie udało nam się poprawnie zaimplementować tego algorytmu w excelu.

Autorzy tekstu [1] wspominają także o możliwości użycia look-up table, jednak jest to process trudny i czasochłonny.

Całość implementacji jest dostępna pod adresem internetowym wskazanym w punkcie V.

III. Obliczenia Ręczne

Cały opisywany algorytm został również wykonany ręcznie, bez pomocy komputera. Obliczenia na kartce obejmowały ukazanie kolejnych kroków otrzymywania wyniku transformaty

Przykładowe obliczenie wykonamy dla następujących, zadanych parametrów:

- Długość Transformaty $N = 8$
- Liczby Fermata równej $F = 17$
- Współczynnika α = 2

Użyty zostanie do tego wzór. 1. Aby wykonywać operacje przy użyciu transformaty zachowujemy warunek $k = 0,\dots,N\text{-}1$. Zatem w tym przypadku dla naszego $N = 8$, k będzie przyjmowało wartości od 0 do 7. W procesie liczenia transformat Fermata, tworzone są macierze, których wymiar jest zależny również od tej wartości. Dla zadanej wartości $N = 8$, macierz będzie miała wymiary *8x8*. W ten sposób otrzymane macierze mnożymy przez zadany ciąg wejściowy liczb.

Przydatna okazuje się tutaj odwrotność multiplikatywna liczby Fermata oraz jądra transformaty. Zapisywana ona jest w formie:

$$ord_{q_t}\omega$$

Wykonując ten wzór dążymy do tego by szukana wartość *k* była taka, aby ω podniesiona do tej wartości dawało wartość 1, po wykonaniu na tej liczbie działania modulo *q*. Dla wypisanych wcześniej przykładowych wartości otrzymujemy przykładowo:

$$
\begin{gathered}
2^8 = 1 \; mod \; 17 \\
2^7 = 9 \; mod \; 17 \\
2^6 = 13 \; mod \; 17 \\
2^5 = 15 \; mod \; 17 \\
2^4 = 16 \; mod \; 17 \\
2^3 = 8 \; mod \; 17 \\
2^2 = 4 \; mod \; 17 \\
2^1 = 2 \; mod \; 17 \\
2^0 = 1 \; mod \; 17
\end{gathered}
$$

Posiadając tak wyliczone wartości jesteśmy teraz w stanie podstawiając odpowiednia dane do wzoru, obliczyć macierz, która jest jednym z kroków liczenia transformaty.

Otrzymana w tym przypadku macierz, wygląda następująco:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 & 16 & 15 & 13 & 9 \\ 1 & 4 & 16 & 13 & 1 & 4 & 16 & 13 \\ 1 & 8 & 13 & 2 & 16 & 9 & 4 & 5 \\ 1 & 16 & 1 & 16 & 1 & 16 & 1 & 16 \\ 1 & 15 & 4 & 9 & 16 & 2 & 13 & 8 \\ 1 & 13 & 16 & 4 & 1 & 13 & 16 & 4 \\ 1 & 9 & 13 & 15 & 16 & 8 & 4 & 2 \end{bmatrix}$$

Tak otrzymaną macierz możemy mnożyć przez nasz ciąg wejsciowy liczb. W przypadku liczenia takich transformat dla dwóch ciągów wejsciowych, wykonujemy ich późniejszy splot. Splot jest mnożeniem otrzymanych ciągów liczb w liczeniu Transformaty.

IV. Wnioski

- Transformata Fermata oraz odwrotna do niej są szczególnie przydatne w przetwarzaniu wideo.
- Prawidłowa implementacja Transformaty jest problematyczna i należy do zadań na pograniczu informatyki i matematyki.
- W celu zachowania własności transformat konieczne jest szczególnie ostrożne dobranie parametrów.

V. Licencja

Wszystkie wykonane w tym projekcie program wydane są na licencji General Public License w wersji 3 (GPL3). Pozwala ona na dowolne kopiowanie, modyfikowanie oraz użytkowanie każdego elementu projektu. Pliki źródłowe można znaleźć na: www.github.com/cyrsztof

Spis Rysunków

Literatura

[1] T. Toivonen, J. Heikkilä, "Video Filtering With Fermat Number Theoretic Transforms Using Residue Number System", IEEE transactions on Circuits and System for Video Technology, Vol. 16, No.I, January 2006

[2] R.C.Agarwal, C.S.Burrus, "Fast Convolution Using Fermat Number Transforms with Applications to Digital Filtering", IEEE Transations on Acoustics, Speech, and Signal Processing, Vol. ASSP-22, NO. 2, April 1974

[3] R.C.Agarwal, C.S.Burrus, "Number Theoretic Transforms to Implement Fast Digital Convolution", Proceding of the IEEE, Vol. 63, No. 4, April 1975

[4] Internetowy kalkulator splotów ciągów, RapidTables, stan na 13.06.17, http://www.rapidtables.com/calc/math/convolution-calculator.htm