УДК 537.8

І. В. Хіміченко, аспірант Міжнародного науково-навчального центру інформаційних технологій та систем НАН України і МОН України

ПРАКТИЧНА РЕАЛІЗАЦІЯ АЛГОРИТМУ ФРАКТАЛЬНОГО СТИСНЕННЯ ЗОБРАЖЕНЬ, ЩО Є ЕФЕКТИВНИМ ЗА ЧАСОМ

Наведений опис програмної реалізації алгоритму фрактального стиснення зображень за допомогою просторово-чутливого хешування. Дані комплексні критерії якості алгоритмів стиснення і відновлення растрових зображень; проведено практичне порівняння з іншими сучасними алгоритмами стиснення зображень на основі комплексного критерію якості.

Вступ. Результати теоретичного аналізу часової ефективності фрактального стиснення дозволяють говорити про доцільність застосування методу для вирішення задач фрактального стиснення зображень.

Мета. Метою статті є практична реалізація алгоритму фрактального стиснення статичних растрових зображень, що є ефективним за часом на основі його порівнянь з іншими сучасними алгоритмами стиснення зображень.

Ідея. Програмна реалізація алгоритму фрактального стиснення складається з двох основних етапів: на етапі попередніх обчислень для всіх векторів, що представляють доменні блоки, обчислюється їх ортонормована проекція $D_k^{\perp} = \phi(D_k)$ на ортодоповнення $\Im^{\perp}$. Визначення оператора $\phi(\cdot)$ було дано в розділі 2.1. Далі, для отриманих точок $D_k^{\perp} \in R^n \setminus \Im$ обчислюються і зберігаються значення хеш-функцій $g_i(D_k^{\perp})$ за формулою (3.3.1).

Етап попередніх обчислень

На етапі попередніх обчислень для кожного доменного блоку $D_j^{\perp}$ з доменного пулу $D_{1...n_d}$ обчислюється значення оператора $\phi(D_k^{\perp})$, після чого обчислюються і зберігаються L значень просторово-чутливих хеш-функцій від вектора $\phi(D_k^{\perp})$.

Алгоритм А: етап попередніх обчислень.
ВХІД: Доменний пул $D_{1...n_d}$, кількість хеш-таблиц L.
ВИХІД: Хеш-таблиці T_i, $i = 1,...,L$.
ДЛЯ КОЖНОГО $i = 1,...,L$
Ініціалізувати хеш-таблицю T_i шляхом генерації випадкової хеш-функції $g_i(\cdot)$
ДЛЯ КОЖНОГО $i = 1,...,L$

ДЛЯ КОЖНОГО $j = 1,...,n_d$

Зберегти точку $D_j^{\perp}$, в наборі $g_i(D_j^{\perp})$ хеш-таблиці T_i.

Результатом етапу попередніх обчислень є заповнення L-множин, кожна з яких містить обчислені значення просторово-чутливої хеш-функції для доменних блоків з доменного пулу. Надалі ця інформація використовується для швидкого зіставлення рангових і доменних блоків. Завдяки властивостям просторово-чутливих хеш-функцій, значення хеш-функції для схожих рангового і доменного блоку з великою вірогідністю збігатимуться. А використання L різних хеш-функцій додатково збільшують цю вірогідність.

Реалізація проектуючого оператора

Розглянемо реалізацію проектуючого оператора $\phi(\cdot)$. Нагадаємо, що в розділі 2.1 оператор $\phi(\cdot)$ визначений таким чином: передбачимо, що ранговий блок R_m має унікальне розкладання $R_m = OR_m + PR_m$, де P - ортогональний проекційний оператор, який проектує $\Re^n$ в підпростір $\Im$ з базисом, що складається з фіксованих базисних блоків $B_1, B_2, ..., B_n$, а оператор $O = I - P$ проектує свій операнд на ортодоповнення $\Im^{\perp}$. Тоді для $Z = (z_1, ..., z_n) \in \Re^n \setminus \Im$ визначимо оператор:

$$\phi(Z) = \frac{OZ}{\|OZ\|}.$$

Математичний сенс даного оператора полягає в наступному: спочатку ранговий блок R_m розглядається як вектор в лінійному векторному просторі $\Re^n$ з базисом, що складається з одиничних взаємно ортогональних векторів. Для $\Re^{16}$ початковий базис задається матрицею:

$$E = \begin{vmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{vmatrix}$$

Оператор $\phi(R_m)$ проектує вектор R_m в підпростір $\Im^{\perp}$, що має базис E^*, що складається з векторів, ортогональних до всіх фіксованих базисних блоків $B_1, B_2, ..., B_n$ і потім нормалізує результат. У *FracLSH* використовується тільки один фіксований базисний блок $\mathbf{1} = (1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)$. Тому в якості базиса $E^{'}$ було вибрано наступну множину векторів:

$$E' = \begin{vmatrix}
-1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & -1 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & -1 & -1 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & -1 & -1 & -1 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & -1 & -1 & -1 & -1 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & 7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 9 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 10 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 11 & 0 & 0 & 0 & 0 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 12 & 0 & 0 & 0 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 13 & 0 & 0 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 14 & 0 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 15 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{vmatrix}$$

Для зручності базис E' наведений в своєму ненормалізованому варіанті. Нормалізація E' виконується автоматично під час ініціалізації програмного забезпечення алгоритму.

Таким чином, оператор $O = I - P$ проектує свій операнд на лінійний векторний простір, заданий векторами $E'_0 ... E'_{14}$. Для обчислення $\phi(R_m)$ нам досить перейти від представлення вектора R_m в базисі E, до представлення R_m в базисі E', відкинути останню координату вектора, що представляє проекцію R_m на фіксований базисний блок, і нормувати результат.

Як відомо з курсу лінійної алгебри, матриця переходу від базису E до E' має вигляд $A = E^T$. Нижче наведений псевдокод алгоритму, що виконує обчислення оператора $\phi(R_m)$.

Алгоритм В: Обчислення проектуючого оператора.

```
void convert_range_basis()
BYTE *pSrcVec, double *pDstVec, size_t size;
    {
        size_t i,j;

        for (i=0; i<size; i++)
        {
        pDstVec[i]=0;

        for(j=0; j<size;j++)
        {
                if (TransitionMatrix[i][j]==0)
                        break;
                        pDstVec[i]+=
(double)pSrcVec[j]* TransitionMatrix[i][j];                    }
}

    pDstVec[size-l]= 0.;
    normalize_vector(pDstVec, pDstVec, size-1);
    }
```

Тут pSrcVec – початковий ранговий блок R_m, pDstVec - його нормалізована проекція $\phi(R_m)$, TransitionMatrix - матриця переходу від базису E до E', а функція normalize_vector() виконує нормалізацію вхідного вектора.

Обчислення оператора $\phi(D_k)$ для заданого доменного блоку D_k виконується аналогічним чином. Тепер у нас є всі необхідні інструменти для обчислення проектуючого оператора і створення допоміжної структури даних хеш-таблиць, і можна приступати до пошуку доменно-рангових відповідностей.

Етап пошуку доменно-рангових відповідностей

На етапі пошуку доменно-рангових відповідностей для даного рангового блоку обчислюється ортонормована проекція $R_m^{\perp} = \phi(R_m)$ на ортодоповнення $\Im^{\perp}$. Обчислюється значення хеш-функцій $g_i(R_m^{\perp})$, і проводиться лінійний пошук в таблицях хеш-функцій доменних блоків, обчислених раніше.

Алгоритм С: пошук доменно-рангових співвідношень.

ВХІД: Ранговий блок $R_m^{\perp}$, M (максимальне число доменних блоків-кандидатів, які повинен повернути алгоритм).

ВИХІД: M (або менше) доменних блоків-кандидатів.

$\Omega \leftarrow \otimes$

ДЛЯ КОЖНОГО $i = 1,...,L$

$\Omega \leftarrow \Omega \cup$ {точки, знайдені в множині $g_i(R_m^{\perp})$ хеш-таблиці T_i }

Повернути M найближчих сусідніх елементів з множини Ω (знаходяться лінійним пошуком серед всіх кандидатів).

Завдяки властивостям просторово-чутливих хеш-функцій, при збігу хеш-значень для рангового і доменного вектора, існує дуже висока вірогідність того, що дані вектори лежать близько один до одного в заданому (у нашому випадку в Евклідовому) метричному просторі. Для M знайдених доменних блоків-кандидатів проводиться обчислення помилки $E\left(D_l, R_m\right)$, $l = 1,...,M$ за формулою (1.3.3) і вибирається найбільш відповідний блок. Таким чином, ми уникаємо необхідності вирішувати задачу для кожної пари доменно-рангових блоків, а відбираємо лише декілька кандидатів, які з великою вірогідністю дадуть вирішення, близьке до оптимального. Цим і досягається істотне підвищення часової ефективності пропонованого алгоритму.

Оцінка часової ефективності алгоритму

Оцінка часової ефективності фрактального стиснення з використанням просторово-чутливого хешування ґрунтується на існуючих оцінках алгоритму пошуку найближчого сусіднього елементу в багатовимірному метричному просторі. Час обробки одного запиту на пошук відповідного доменного блоку для даного рангового блоку визначається кількістю операцій обчислення функції відстані $\Delta\left(R_m^{\perp}, D_k^{\perp}\right)$ між проекціями рангового і доменного блоків, і кількістю операцій обчислення хеш-функції $g_i(D_k^{\perp})$. Загальна кількість запитів дорівнює кількості рангових блоків. Відомо, що при відповідному виборі параметрів алгоритму *LSH* для одного запиту кількість операцій обчислення функції відстані визначається як $O(n_d^{\rho})$, а кількість операцій обчислення хеш-функції, відповідно, як $O(n_d^{\rho} \log_{1/p_2} n_d)$, де n_d – загальна кількість доменних блоків, $\rho = \dfrac{\ln(1/p_1)}{\ln(1/p_2)}$ для (r_1, r_2, p_1, p_2) – чутливої хеш-функції, p_1 визначена як вірогідність того, що відстань між двома точками в заданому метричному просторі не більша за r_1 за умови, що хеш-функція дає колізію, p_2 – це вірогідність того, що відстань між двома

точками більша за r_2 за умови, що колізії не відбувається. Алгоритм LSH для побудови проміжних структур даних вимагає $O(dn_d + n_d^{1+\rho})$ слів пам'яті, де d – це розмірність ортонормованої проекції $D_j^{\perp}$ доменного блоку на ортодоповнення $\Im^{\perp}$.

Таким чином, досягається сублінійний час обробки одного рангового блоку, що є великим кроком вперед в порівнянні з класичними алгоритмами фрактального стиснення.

Висновок. У даній статті розглянута програмна реалізація алгоритму фрактального стиснення зображень за допомогою просторово-чутливого хешування. Результати показують, що на основі запропонованого методу можлива побудова високоефективного алгоритму фрактального стиснення зображень, який працює в декілька сотень разів швидше за оригінальний алгоритм Джеквіна, і, за певних умов, перевершує існуючі алгоритми фрактального стиснення по співвідношенню час роботи/якість» вихідного зображення.

1. *Monro D.M., Dudbridge F.* Fractal approximation of image blocks, in: Proceedings of ICASSP-1992 IEEE International Conference on Acoustics, Speech and Signal Processing, Vol. 3, pp. 485-488, 1992.
2. *Motwani R., Naor A., Panigrahy R.* Lower bounds on Locality Sensitive Hashing. Proceedings of the Symposium on Foundations of Computer Science, 2005.
3. *Friedman J.H., Bentley J.L., Finkel R.A.* An algorithm for finding best matches in logarithmic expected time, ACM Trans. Math. Software 3,3 (1977) 209-226.
4. *Frigaard C., Gade J., Hemmingsen T., Sand T.,* Image compression based on fractal theory, Institute for Electronic Systems, Aalborg University, Denmark, 1994.
5. *Forte B., Vrscay E.R.* Solving the inverse problem for function/image approximations using iterated function systems, 1. Theoretical basis, Fractals 2,3 (1994) 325-334.
6. *Хіміченко І.В.* Підхід до фрактального стиснення зображень з використанням просторово-чутливого хешування як метода підвищення часової ефективності при фрактальному стисненні / САИТ, XI международная научно-техническая конференция УНК ИПСА НТУУ «КПИ», 2009
7. *Хіміченко І.В.* Роль пошуку найближчого сусіднього елементу при фрактальному стисненні зображень / ИАИ, IX международная научная конференция, НТУУ «КПИ», 2009