PARALELNÝ PRÍSTUP K FRAKTÁLOVEJ KOMPRESII OBRAZU A PARALLEL APPROACH TO FRACTAL IMAGE COMPRESSION

Ľubomír Dedera
Katedra informatiky a výpočtovej techniky, Vojenská akadémia v Liptovskom Mikuláši, P.O. BOX 45, 03101 Liptovský Mikuláš
E-mail: dedera@valm.sk

Abstrakt Článok sa zaoberá paralelizáciou kódovacieho a dekódovacieho algoritmu fraktálovej blokovej kompresie obrazu a prináša experimentálne výsledky porovnávajúce sekvenčné a paralelné algoritmy z hľadiska dosiahnutých časov kódovania a dekódovania a efektívnosti paralelizácie.

Summary The paper deals with a parallel approach to coding and decoding algorithms in fractal image compression and presents experimental results comparing sequential and parallel algorithms from the point of view of achieved both coding and decoding time and effectiveness of parallelization.

1. ÚVOD

Fraktálová kompresia obrazu [1], [2], [3] je po teoretickej stránke založená na teórii metrických priestorov a kontraktívnych operátorov. Patrí k metódam stratového kódovania obrazov s asymetrickými nárokmi na výpočtovú zložitosť kódovania a dekódovania: zatiaľ čo výpočtová zložitosť dekódovania nie je vysoká, kódovanie je výpočtovo veľmi náročné. V súvislosti s rozvojom paralelných výpočtových systémov, architektúr a s nimi spojeného programového vybavenia sa preto jednou z ciest znižovania reálnej časovej výpočtovej zložitosti javí paralelizácia algoritmov.
Cieľom tohto článku je poukázať na reálne možnosti paralelizácie algoritmov kódovania a dekódovania, navrhnúť a experimentálne overiť ich paralelné verzie pre multiprocesorové systémy.

2. ZÁKLADNÝ PRINCÍP KÓDOVANIA A DEKÓDOVANIA

Nech $\boldsymbol{I}$ je originálny obraz. Potom množina $\mathbf{R} = \{R_i : 1 \le i \le N\}$ navzájom disjunktných obrazových blokov takých, že $\bigcup_{1 \le i \le N} R_i = \boldsymbol{I}$, sa nazýva R-rozklad (range partition) obrazu $\boldsymbol{I}$ a jednotlivé bloky v množine $\mathbf{R}$ sa nazývajú R-bloky. R-bloky sú obrazové bloky, po ktorých sa obraz kóduje. Pre naše účely budeme predpokladať, že všetky R-bloky sú štvorcové a s rovnakým rozmerom.
Ľubovoľná množina $\mathbf{D}$ obrazových blokov získaných z obrazu $\boldsymbol{I}$ $\mathbf{D} = \{D_i : 1 \le i \le m\}$ sa nazýva D-oblasť (domain pool) obrazu $\boldsymbol{I}$ a jednotlivé bloky, ktoré ju tvoria, sa nazývajú D-bloky. Na rozdiel od R-blokov sa D-bloky môžu vzájomne prekrývať a, na druhej strane, ich zjednotenie nemusí pokryť celý obraz. V procese kódovania sa obraz $\boldsymbol{I}$ kóduje po jednotlivých R-blokoch, pričom sa na kódovanie jedného R-bloku využíva transformácia jedného D-bloku. V našich experimentoch budeme pri tom využívať D-bloky s dvojnásobným rozmerom strany ako v prípade R-blokov.
Nech je ďalej ľubovoľný R-blok reprezentovaný vektorom $R \in \mathbf{R}^n$, kde n predstavuje počet obrazových prvkov v tomto bloku a vektor R je vytvorený „poukladaním“ jednotlivých riadkov kódovaného R-bloku za sebou. D-oblasť sa doplní o bloky, ktoré sa získajú z pôvodných D-blokov aplikovaním ôsmich izometrií štvorca a nakoniec sa pomocou spriemerovania jasových úrovní rozmer blokov v D-oblasti decimuje na rozmer R-blokov. Takto získané bloky sa nazývajú kódové bloky a ich množina sa nazýva kódová kniha. Problém kódovania R-bloku R (v tvare vektora) pomocou kódového bloku D (analogicky tiež prevedeného do tvaru vektora) je potom riešiteľný metódou najmenších štvorcov [3] ako problém

$$\min_{x \in \mathbf{R}^2} \|R - \mathbf{A}x\|, \tag{1}$$

kde $\mathbf{A}$ je matica s rozmerom $n \times 2$ so stĺpcami $D, (1,...,1)^T$, $x = (a,b)^T \in \mathbf{R}^2$ je hľadaný vektor koeficientov a $\|.\|$ predstavuje L_2-normu vektora. Optimalizačný problém (1) má riešenie [3]

$$\bar{x} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T R, \tag{2}$$

pričom matica

$$\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T \tag{3}$$

sa nazýva pseudoinverzná matica k matici $\mathbf{A}$.
V spojitosti s tým kód R-bloku R bude tvorený indexom použitého optimálneho (v zmysle hodnoty (1)) D-bloku D (v procese kódovania je teda potrebné nájsť aj najvhodnejší D-blok D), použitou izometriou iz a dvojicou koeficientov a,b získaných riešením (2). Koeficient a reprezentuje zmenu kontrastu a koeficient b posun jasových úrovní. Aby sa zabezpečila konvergencia procesu dekódovania, ktorá je teoreticky založená na vete o pevnom bode, je potrebné počítať len s tými prípadmi, keď $|a| < 1$. V súvislosti s tým

môže byť kód obrazu $\boldsymbol{I}$ formálne definovaný ako množina usporiadaných pätíc

$$C_I = \{(R_i, D_i, \mathrm{iz}_i, a_i, b_i), 1 \le i \le N\}\ , \quad (4)$$

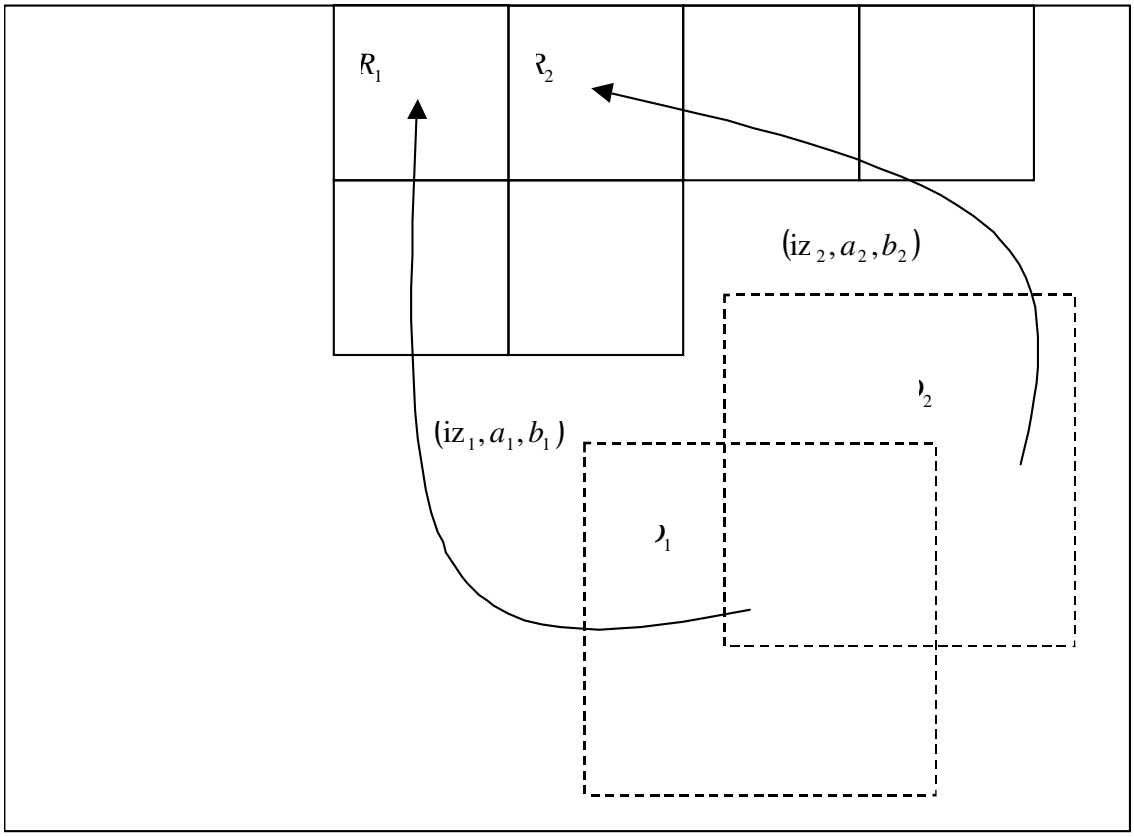
kde N je počet R-blokov v obraze $\boldsymbol{I}$ (obr. 1), R_i sú súradnice kódovaného R-bloku, D_i sú súradnice D-bloku využitého pri kódovaní R_i, iz_i je použitá izometria a a_i, b_i sú nájdené transformačné koeficienty.

V priebehu kódovania každého R-bloku R je potrebné spravidla mnohonásobne opakovať postup opísaný v predchádzajúcom odstavci, až kým sa k nemu nenájde taký kódový blok D_R, pre ktorý je hodnota výrazu (1) minimálna alebo aspoň menšia ako určitá vopred definovaná konštanta. Z uvedeného je zrejmé, že v priebehu procesu kódovania je potrebné mnohonásobne prehľadávať kódovú knihu. Ak sa v kódovej knihe nachádza *m* blokov, potom časová zložitosť každého prehľadania bude lineárna, tzn. rovná *O*(*m*). Na jej redukciu bolo navrhnutých niekoľko metód [2], [3], [4], [5], [6], pričom jednou z možností redukcie výpočtovej zložitosti je aj využitie paralelných výpočtových prostriedkov a paralelných algoritmov.

Dekódovanie obrazu reprezentovaného podľa (4) možno realizovať iteračným algoritmom, ktorý môže začať svoju činnosť na ľubovoľnom obraze a kde sa po jednotlivých iteráciách ($k = 1, 2, ...$) postupne po jednotlivých R-blokoch R_i počítajú hodnoty ich obrazových prvkov podľa vzťahu

$$R_i^{k+1} = a_i\ \mathrm{iz}_i\left(\mathrm{dec}(D_i^k)\right) + b_i\,, \quad (5)$$

kde dec reprezentuje decimáciu D-blokov na rozmer R-blokov a symboly a_i, iz_i, D_i, b_i boli popísané pri (4).



Obr. 1. Štruktúra fraktálového kódu

Fig. 1. The structure of fractal code

3. PARALELIZÁCIA KÓDOVANIA

Pre paralelizáciu základného sekvenčného algoritmu kódovania popísaného v predchádzajúcej kapitole je možné v zásade zvoliť dve stratégie [5]:

1) Paralelizácia na základe R-blokov;
2) Paralelizácia na základe D-blokov.

Zamerajme sa teraz podrobnejšie na paralelizáciu na základe R-blokov. V rámci tohto prístupu je potrebné, aby bolo možné prinajmenšom celý obraz uložiť do lokálnej pamäte procesora. Na základe obrazových dát je možné zostrojiť úplnú kódovú knihu. Každému procesoru je možné priradiť pomernú časť R-rozdelenia, a to buď staticky alebo dynamicky. Každý procesor pri tom stanoví parametre optimálnej kódujúcej

transformácie pre všetky R-bloky, ktoré sú mu priradené tak, ako v prípade sekvenčného algoritmu.

Algoritmus uvedený na obr. 2 predstavuje aplikáciu uvedeného prístupu na multiprocesorových MIMD architektúrach. Po formálnej stránke jeho riadiaca štruktúra **forparallel** vyjadruje možný paralelný výpočet kódu pre všetky R-bloky. Reálnu implementáciu paralelného spracovania je možné realizovať pomocou technológie vlákien (threads). Vlákno v operačných systémoch postavených na tzv. machovskej architektúre (medzi takéto operačné systémy patrí napr. Microsoft Windows NT, Sun Solaris, Compaq Tru64 Unix a.i.) predstavuje elementárnu jednotku, ktorej sa prideľuje procesor. Preto ak má výpočtový proces v jednom časovom okamihu aktívnych viacero vlákien a súčasne výpočtový systém disponuje viacerými procesormi, môže sa na riešení jednej úlohy naraz zúčastňovať viacero procesorov. Tento prístup je v súčasnosti najčastejším praktickým riešením bežne dostupným pre multiprocesory. Jeho výhodou je, že vlákna nie sú viazané na počet procesorov a programy, ktoré ich využívajú, pracujú korektne aj na jednoprocesorových systémoch. O prideľovanie procesora jednotlivým vláknam sa nestará programátor, ale plánovač operačného systému. Elementárnou úlohou je v tomto prípade kódovanie jedného R-bloku, pričom medzi elementárnymi úlohami nie je potrebná žiadna komunikácia. Z dôvodu režijných nákladov operačných systémov na vytvorenie vlákien a tiež z dôvodu, že počet procesorov v multiprocesoroch sa pohybuje maximálne v desiatkach (typické sú skôr hodnoty 2 až 4), pri praktickej realizácii bol počet paralelných úloh (vlákien) obmedzený na 20 a jedno vlákno je zodpovedné za spracovanie pomerného počtu R-blokov (agregácia). Naviac, na zabezpečenie riadeného zápisu do spoločných údajových štruktúr (výstupný kód) viacerými vláknami boli v experimentoch použité synchronizačné prostriedky operačného systému typu mutex.

```
MIMDParalelneKodovanieObrazu(Obraz o, int rs, int cs)
//o – kódovaný obraz, rs – rozmer R-blokov, cs – počet blokov v kódovej knihe
{
      RRozdelenie *rp;
      RBlok R;
      KodovaKniha *cb;
      KodovyBlok D;
      KodObrazu kod;

      rp = VytvorRRozdelenie(rs);
      cb = VytvorKodovuKnihu(rs, cs);

      forparallel (všetky R-bloky R v rp) { // pre všetky R-bloky
            for (všetky kódové bloky D v cb) {

                  // Určí vektor transformačných koeficientov x (vzťahy (2) a (3))
                  x = D.pseudoinv_matica_A * R;

                  // Na základe x určí aproximovaný blok k ...
                  k = D.matica_A * x;

                  // ... a porovná kvalitu aproximácie (1)
                  if (||R – k|| je menšia ako doteraz nájdená najmenšia metrika)
                        Zapamätaj hodnotu metriky, súradnice kódového bloku D, použitú izometriu a
                        vektor transformačných koeficientov x;
            }
            Ulož súradnice nájdeného optimálneho kódového bloku, použitú izometriu
            a vektor transformačných koeficientov pre R-blok R do kódu obrazu kod;
      }
      return kod;
}
```

Obr. 2. Základný kódovací algoritmus na multiprocesorových MIMD architektúrach

Fig. 2. Basic coding algorithm for MIMD architectures

Ďalej nás bude zaujímať efektívnosť paralelného algoritmu v porovnaní so sekvenčným algoritmom. Keďže uvedený algoritmus nevyžaduje žiadne nároky na komunikáciu medzi jednotlivými paralelne

vykonávanými elementárnymi úlohami, možno teoreticky očakávať efektívnosť paralelizácie definovanú vzťahom

$$E = \frac{t_s}{t_p p} \tag{6}$$

blízku 1, pričom t_s je čas realizácie sekvenčného algoritmu, t_p čas realizácie paralelného algoritmu a p počet procesorov. Jej skutočná hodnota je však ovplyvnená jednak tým, že na vytvorenie kódovej knihy paralelizácia použitá nebola, jednak réžiou operačného systému pri vytváraní vlákien a jednak známym faktom, že v prípade pridania väčšieho počtu procesorov do multiprocesorového systému sa nedosiahne tomu úmerné zvýšenie celkovej výpočtovej kapacity.

Porovnanie režijných nákladov a efektívnosti paralelného algoritmu oproti sekvenčnému algoritmu možno získať ich otestovaním na jednoprocesorovom systéme. Pre experiment bol zvolený monochromatický obraz Lena s rozmerom 512×512 obrazových prvkov a s hĺbkou 8 bitov. Obidva algoritmy realizovali jeho kódovanie pri použití izometrií a rovnakej kódovej knihy obsahujúcej 1600 blokov. Počet vlákien podieľajúcich sa na paralelnej verzii algoritmu bol 20. Čas kódovania pre rôzne veľkosti R-blokov na jednoprocesorovej PC zostave s procesorom AMD Duron 750 MHz, 256 MB RAM a OS Windows 2000 Professional uvádza tab. 1. Z porovnania vyplýva, že rozdiely medzi obidvomi algoritmami sú zanedbateľné a je možné ich pripísať jednak pseudonáhodným vplyvom činnosti operačného systému, častejšiemu prideľovaniu procesora aplikácii v prípade, že má súčasne viacero aktívnych vlákien, očakávanej vyššej efektívnosti spracovania pri viacerých súčasne aktívnych vláknach a, samozrejme, menším implementačným odlišnostiam v obidvoch prípadoch. Dôležitý záver je, že využívanie technológie vlákien umožňujúcej o. i. činnosť viacerých procesorov na tej istej úlohe neprináša so sebou nijaké významnejšie režijné náklady.

Tab. 1. Porovnanie časov kódovania sekvenčného a paralelného algoritmu na obraze Lena, jednoprocesorový systém

Tab. 1. Comparison of coding times of sequential and parallel algorithms for image Lena, uniprocessor system

Rozmer R-blokov (kódová kniha vo všetkých prípadoch 1600 blokov, s izometriami)	**Čas kódovania [s]**	
	Sekvenčný algoritmus	**Paralelný algoritmus 1 procesor**
4x4	140	139
6x6	130	130
8x8	123	122
10x10	124	125
12x12	122	123

Tab. 2. Porovnanie časov kódovania sekvenčného a paralelného algoritmu na obraze Lena, dvojprocesorový systém

Tab. 2. Comparison of coding times of sequential and parallel algorithms for image Lena, dual-processor system

Rozmer R-blokov (kódová kniha vo všetkých prípadoch 1600 blokov, s izometriami)	**Čas kódovania [s]**		**Efektívnosť paralelizácie**
	Sekvenčný algoritmus	**Paralelný algoritmus 2 procesory**	
4x4	337	204	0,82
6x6	317	185	0,85
8x8	303	174	0,87
10x10	308	187	0,82
12x12	304	172	0,88

Tab. 2 porovnáva časy kódovania sekvenčného a paralelného algoritmu na dvojprocesorovej zostave 2 × Pentium II 233 MHz, 128 MB RAM. Z výsledkov možno vidieť, že zistená efektívnosť paralelizácie algoritmu (6) sa pohybuje v intervale 0,82 až 0,88, čo možno považovať za veľmi dobrý výsledok. Pripomíname, že teoreticky najvyššia dosiahnuteľná hodnota efektívnosti je 1. Je potrebné na tomto mieste taktiež spomenúť, že hodnota efektívnosti sa môže od systému k systému meniť, jej výraznejšie priblíženie k 1 možno očakávať pri použití rýchlejších pamäťových čipov a naopak, zníženie pri použití väčšieho počtu procesorov.

Druhou možnou stratégiou paralelizácie kódovacieho algoritmu je paralelizácia na základe D-blokov. Tento prístup je vhodné použiť vtedy, keď nie je možné uložiť celú kódovú knihu v lokálnej pamäti jedného procesora. Preto je potrebné kódovú knihu distribuovať v lokálnych pamätiach jednotlivých procesorov. Experimentálne výsledky získané týmto prístupom možno nájsť v [4].

4. PARALELIZÁCIA DEKÓDOVANIA

Pri paralelizácii iteračného dekódovacieho algoritmu možno s výhodou využiť fakt, že v rámci jednej iterácie môžu byť nové hodnoty jasových úrovní jednotlivých obrazových prvkov počítané nezávisle jeden od druhého. Z uvedeného dôvodu je možné založiť paralelizáciu na elementárnych úlohách reprezentujúcich výpočet hodnoty jasovej úrovne jediného obrazového prvku. Pretože v prípade MIMD architektúr možno očakávať podstatne menší počet procesorov ako počet obrazových prvkov dekódovaného obrazu, bude účelné pristúpiť k aglomerácii spomenutých elementárnych úloh. Keďže rovnaké parametre (D-blok, izometria, transformačné koeficienty) sa aplikujú pri výpočte jasových úrovní obrazových prvkov v rámci celého R-bloku, je vhodné založiť paralelizáciu na R-blokoch (obr. 3) alebo na skupinách viacerých R-blokov.

```
Obraz MIMDParalelneDekodovanieObrazu(KodObrazu kod, int pocetiteracii)
// kod – kód obrazu, pocetiteracii – požadovaný počet iterácií pri dekódovaní obrazu
{
      Obraz o;

      Inicializuj všetky prvky obrazu o na 0 (čierna farba);

      for (i = 0; i < pocetiteracii; i++)
            forparallel (všetky kódy R-blokov kr v kod) {
                  // kr obsahuje položky:      x, y – súradnice kódovaného R-bloku
                  // dx, dy – súradnice kódujúceho D-bloku
                  // iz – použitá izometria
                  // a, b – transformačné koeficienty

                  Nech d1 predstavuje na veľkosť R-bloku decimovaný D-blok obrazu o so súradnicami dx, dy;
                  Aplikuj na blok d1 izometriu iz a výsledok ulož ako d2;
                  Do obrazu o ulož nový R-blok so súradnicami x, y a jasovými úrovňami obrazových prvkov
                  a * d2 + b (vzťah (5));
            }
            return o;
}
```

Obr. 3. Iteračný dekódovací algoritmus na multiprocesorových MIMD architektúrach

Fig. 3. Iterative decoding algorithm for multiprocessor MIMD architectures

Tab. 3. Porovnanie časov dekódovania sekvenčného a paralelného algoritmu na obraze Lena, 20 iterácií, 20 opakovaní, dvojprocesorový systém

Tab. 3. Comparison of decoding times of sequential and parallel algorithms for image Lena, 20 iterations, 20 repetitions, dual-processor system

Rozmer R-blokov	Čas dekódovania [s]		Efektívnosť paralelizácie
	Sekvenčný algoritmus	Paralelný algoritmus 2 procesory	
4x4	108	68	0,79
6x6	82	51	0,80
8x8	66	41	0,81
10x10	64	40	0,80
12x12	63	38	0,82

Keďže pri výpočte jasových úrovní obrazových prvkov v R-bloku je potrebné poznať hodnoty jasových úrovní obrazových prvkov zodpovedajúceho D-bloku, ktorý môže byť lokalizovaný kdekoľvek v obraze, vhodnou cieľovou architektúrou sa javí multiprocesor so zdieľanou pamäťou, a to preto, lebo pre všetky procesory zabezpečuje prístup k obrazovým dátam za rovnakých podmienok v konštantnom čase. Ak by bola cieľová architektúra založená na distribúcii obrazových dát medzi jednotlivými procesorovými jednotkami, procesor dekódujúci daný R-blok na základe od neho geometricky vzdialenejšieho D-bloku by sa k jasovým úrovniam obrazových prvkov D-bloku dostal s vyššou, nekonštantnou komunikačnou zložitosťou. Keďže elementárne úlohy sú výpočtovo nenáročné, možno očakávať, že komunikačná zložitosť by prevýšila prínosy z paralelizácie.

Podobne ako pri kódovaní bol počet súbežne vytvorených vlákien pri praktických experimentoch obmedzený na 20, pričom každé vlákno zabezpečovalo dekódovanie pomerného počtu R-blokov. Na zabezpečenie synchronizácie jednotlivých iterácií medzi vláknami boli použité systémové prostriedky typu event. Konkrétne časy dekódovania obrazu Lena s rozmerom 512x512 obrazových prvkov pre rôzne rozmery R-blokov, pri 20 dekódovaniach a pri 20 iteráciách na dvojprocesorovom systéme 2 × Pentium II 233 MHz, 128 MB RAM zachytáva tabuľka tab. 3. Efektívnosť paralelného algoritmu sa v tomto prípade pohybuje v intervale 0,79 až 0,82.

5. ZÁVER

Algoritmy fraktálovej kompresie obrazu sú vo všeobecnosti vhodné pre paralelnú implementáciu. V prípade kódovacích algoritmov je vhodnosť použitia SIMD aj MIMD architektúr umocnená skutočnosťou, že jednotlivé R-bloky možno kódovať paralelne, s minimálnymi nárokmi na komunikáciu medzi jednotlivými elementárnymi procesmi resp. procesorovými jednotkami. Táto skutočnosť sa potvrdila aj v experimentálnych výsledkoch, kde v prípade dvojprocesorového systému efektívnosť paralelného algoritmu dosahovala hodnoty 0,82 až 0,88. Pri dekódovacích algoritmoch boli v prípade multiprocesorových MIMD architektúr dosiahnuté podobné výsledky efektívnosti. Vzhľadom na komunikačnú zložitosť sa nedajú predpokladať v prípade dekódovania podobné výsledky pri SIMD architektúrach, hoci tento prípad nebol priamo študovaný.

LITERATÚRA

[1] JACQUIN, A. E.: A Fractal Theory of Iterated Markov Operators with Applications to Digital Image Coding. PhD Thesis, Georgia Institute of Technology, Atlanta, Georgia, USA,1989.

[2] JACQUIN, A. E.: Fractal Image Coding Based on a Theory of Iterated Contractive Image Transformations. In Proc. of SPIE Symposium on Visual Communication and Image Proc., Vol. 1360 (1990), s. 227-239.

[3] SAUPE, D., HAMZAOUI, R.: Complexity Reduction Methods for Fractal Image Compression. In I.M.A. Conf. Proc. on Image Processing; Mathematical Methods and Applications, J. M. Blackledge (ed.), Oxford University Press, Oxford, UK, 1995.

[4] HUFNAGL, C., HÄMMERLE, J., POMMER, A., UHL, A., VAJTERSIC, M.: Fractal Image Compression on Massively Parallel Arrays. In Proc. of International Picture Coding Symposium (PCS'97), volume 143, Berlin, Germany, September 1997, s. 77-80.

[5] HÄMMERLE, J., UHL, A.: Fractal Image Compression on Multiprocessors and Multicomputers. In Proc. of the Internat. Conference on Parallel Processing and Applied Mathematics, Zakopane, Poland, 1997, s. 433-441.

[6] DEDERA, Ľ.: Fraktálová kompresia obrazu a jej paralelné algoritmy. Habilitačná práca, Vojenská akadémia v Liptovskom Mikuláši, Liptovský Mikuláš, 2002.