УДК 004.043

О. Шпортько
Рівненський державний гуманітарний університет,
кафедра економічної кібернетики

ВИКОРИСТАННЯ ПРЕДИКТОРІВ В ПРОЦЕСІ ПРОГРЕСУЮЧОГО ІЄРАРХІЧНОГО КОНТЕКСТНО-НЕЗАЛЕЖНОГО СТИСНЕННЯ ЗОБРАЖЕНЬ БЕЗ ВТРАТ

Обґрунтована доцільність, наведений спосіб обходу пікселів та запропоновані предиктори для реалізації прогресуючого ієрархічного контекстно-незалежного стиснення зображень без втрат. Подано результати застосування запропонованих підходів для зменшення ентропії зображень набору АСТ під час попередніх перетворень.

Ключові слова: прогресуюче стиснення зображень, стиснення без втрат, предиктори, ентропія.

Argued expedience, the method of round of pixels is resulted and the predictors for realization of progressing hierarchical context-independent compression of images without losses are offered. The results of application of proposed approaches for diminishing entropy of representing the set of ACT in the process of previous transformations are presented.

Key words: progress compression of images, compression without losses, predictors, entropy.

Вступ

У сучасному світі зображення є невід'ємною складовою мультимедійної інформації, яка найчастіше створюється, нагромаджується і зберігається на цифрових носіях та передається по каналах зв'язку. Компресія відповідних файлів дає змогу пропорційно підвищити швидкість обміну інформацією по мережі та зменшити обсяги використання дискового простору. Сьогодні опрацювання яскравостей пікселів зображень у популярних графічних форматах, які виконують стиснення без втрат (наприклад, у форматі PNG [4, с. 249–317]), найчастіше здійснюється послідовно по рядках згори донизу, а у кожному рядку – поспіль зліва направо. Як наслідок, вивести стиснуте зображення у цих форматах можливо лише після декодування всіх пікселів, а декомпресія фотокарток чи малюнків з мільйонами пікселів за такого способу обходу може тривати декілька секунд незалежно від розміру області та роздільної здатності пристрою виводу.

Поряд з цим, для прискорення виводу великих зображень у форматах компресії з втратами найчастіше застосовують прогресуюче (поступальне) ієрархічне опрацювання пікселів [6, с. 176]. Під час застосування цього способу обходу зображення опрацьовують пошарово, збільшуючи щоразу роздільну здатність, причому під час послідовного оброблення даних чергового шару використовують дані попередніх шарів. Зображення з пікселів чергового шару фактично є зменшеною у декілька разів (найчастіше – у чотири) копією зображення з пікселів наступного шару, а останній шар збігається з вхідним зображенням. Тому під час прогресуючого ієрархічного декодування деталі зображення проявляються поступово. Зупинити таке декодування можливо вже після декомпресії шару з кількістю пікселів, не меншою від області виводу по кожній з осей, не очікуючи відтворення всіх пікселів зображення. Отже, розроблення методів і графічного формату компресії зображень без втрат з використанням принципів прогресуючого ієрархічного опрацювання, що є метою дослідження, є сьогодні актуальним завданням.

Аналіз останніх досліджень і публікацій.
Принципи стиснення зображень без втрат з використанням предикторів

Як відомо, стиснення зображень без втрат у графічних форматах найчастіше відбувається в три етапи: на першому яскравості пікселів перетворюються за допомогою предикторів; на другому контекстно-залежне кодування зменшує надлишковості між подібними фрагментами; на третьому контекстно-незалежне кодування усуває надлишковості між переважаючими значеннями яскравостей компонентів. Контекстно-залежне кодування може зменшувати коефіцієнт стиснення (відношення розмірів стиснутого до нестиснутого файлів зображення, надалі – КС) в декілька разів за рахунок подібних фрагментів. Але такі фрагменти рідко трапляються у фотореалістичних зображеннях, тому єдиним універсальним етапом стиснення зображень без втрат є контекстно-незалежне кодування. Основний принцип такого кодування: **довжина коду довільного елемента з більшою ймовірністю не повинна перевищувати довжину коду будь-якого елемента з меншою ймовірністю**. Цей принцип базується на фундаментальному положенні теорії інформації, згідно з яким для мінімізації довжини коду послідовності елемент s_i (в нашому випадку під елементом мається на увазі яскравість окремої компоненти кожного піксела) з ймовірністю появи $p(s_i)$ доцільно кодувати $-\log_2 p(s_i)$ бітами [2, с. 17]. Тому середня довжина коду елемента блока після застосування будь-якого контекстно-незалежного алгоритму згідно з формулою of Shannon [2, с. 611], не може бути меншою від *ентропії джерела*

$$H = -\sum_i p(s_i) \times \log_2 p(s_i) . \qquad (1)$$

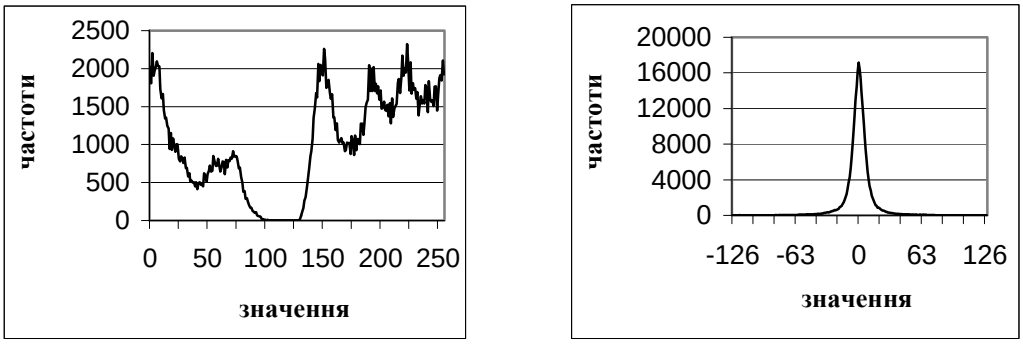
Як відомо, ентропія джерела зменшується зі збільшенням нерівномірності розподілу ймовірностей (частот) між елементами [1]. За нашими підрахунками, застосування контекстно-незалежного алгоритму в середньому зменшує КС зображень на 33 %.

Підвищити ефективність цього кодування під час стиснення зображень без втрат намагаються за допомогою *предикторів*, які під час обходу прогнозують значення яскравості кожної компоненти чергового піксела (наприклад, для найпоширеніших 24-бітних зображень – це яскравості червоної, зеленої та синьої компоненти, записані цілими числами в окремих байтах), використовуючи значення яскравостей тих самих компонентів опрацьованих раніше суміжних пікселів, оскільки ці яскравості мають між собою найбільший степінь кореляції [5, с. 675]. Під час використання цього підходу обчислюють і надалі кодують відхилення Δ_{ij} значення яскравості чергової компоненти піксела F_{ij} від прогнозованого обраним предиктором значення $predict_{ij}$, тобто

$$\Delta_{ij} = F_{ij} - predict_{ij} \qquad (2)$$

(*i* та *j* пробігають відповідно по всіх рядках та стовпцях компонентів пікселів зображення). Суміжні пікселі зображень найчастіше мають подібні кольори, а значить і близькі значення яскравостей відповідних компонентів, тому значення прогнозу часто збігається зі значенням яскравості чергової компоненти, найчастіше – є близьким до цього значення і рідко – значно відрізняється від нього (рис. 1). Тобто більшість значень Δ_{ij} виявляються близькими до нуля. Тим самим застосування предикторів найчастіше збільшує нерівномірність розподілу ймовірностей значень яскравостей і, як наслідок, зменшує ентропію (1).

Чому ж значення яскравостей компонентів пікселів відхиляються від значень, прогнозованих предикторами? Річ у тому, що ці відхилення найчастіше обумовлені двома об’єктивно існуючими основними факторами: "сильними" змінами – трендом та "слабкими" фоновими коливаннями – шумом. "Тому можливі два протилежні типи моделей: внесок шуму незначний порівняно з внеском еволюції; внесок еволюції незначний порівняно з внеском шуму. У першому випадку ... будемо передбачати значення ..., враховуючи ... тенденцію, що склалася, у другому – як рівне середньому арифметичному ... попередніх елементів" [3, с. 59].



а *б*

Рис. 1. Розподіл частот значень зеленої компоненти зображення Lena.bmp:
а – до застосування предиктора (H=7,59 bpb); б – після застосування Left-предиктора (H=5,34 bpb)

Крім того, під час послідовного обходу пікселів предиктори можуть використовувати для прогнозування лише значення яскравостей з попередніх рядків та зліва у черговому рядку (рис. 2), що зменшує ефективність їх застосування.

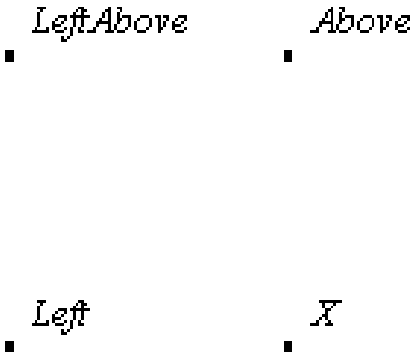


Рис. 2. Позначення суміжних елементів до елемента X
при послідовному способі обходу пікселів зображення

За такого способу обходу пікселів, використовуючи позначення суміжних елементів для елемента *X* у відповідності до рис. 2, найпоширеніші сьогодні предиктори мовою С записують так:

```
typedef unsigned char ubyte;
ubyte LeftPredict(ubyte Left, ubyte Above, ubyte LeftAbove)
 {return Left;}
ubyte AbovePredict(ubyte Left, ubyte Above, ubyte LeftAbove)
 {return Above;}
ubyte AveragePredict(ubyte Left, ubyte Above, ubyte LeftAbove)
 {return (Left+Above)/2;}
ubyte PaethPredict(ubyte Left, ubyte Above, ubyte LeftAbove)
 {int pp=Left+Above-LeftAbove;
  int pa, pb, pc;
  pa=abs(pp-Left); pb=abs(pp-Above); pc=abs(pp-LeftAbove);
  if (pa<=pb && pa<=pc) return Left;
  else if (pb<=pc) return Above;
       else return LeftAbove;}
ubyte MedPredict(ubyte Left, ubyte Above, ubyte LeftAbove)
```

```
{if (LeftAbove>=max(Left, Above)) return min(Left, Above);
 else if (LeftAbove<=min(Left, Above)) return max(Left, Above);
      else return Left+Above-LeftAbove; }
```

Предиктор *LeftPredict* прогнозує значення чергового елемента таким, що дорівнює значенню зліва, предиктор *AbovePredict* – значенню згори, предиктор *AveragePredict* – середньому арифметичному цих значень. Ці три предиктори належать до лінійних статичних предикторів, по суті розраховують середнє арифметичне значень окремих суміжних елементів і тому описують шумову модель. Наступні два предиктори належать до нелінійних статичних предикторів, враховують тенденції відносно значення, спрогнозованого у припущенні рівностей фонових коливань діагональних елементів, а, отже, описують змішану трендово-шумову модель.

Предиктор Піфа *PaethPredict* розраховує значення у точці *X*, враховуючи площину, що проходить через точки *Left*, *Above* та *LeftAbove* у тривимірному просторі і прогнозує одне з цих трьох значень у напрямку найменшого приросту стосовно розрахованого значення.

Предиктор *MedPredict* намагається адаптуватися до локальних горизонтальних та вертикальних ребер. Значення *Left* найчастіше повертається при виявленні горизонтального, а *Above* – при виявленні вертикального ребра. Якщо ребро не виявлено, то повертається значення площини над точкою *X*, що проходить у тривимірному просторі через точки *Left*, *Above* та *LeftAbove*. Перші чотири з наведених предикторів використовуються, наприклад, у форматі PNG, четвертий – в архіваторі WinRAR, останній – у форматі стиснення JPEG-LS. Опис інших предикторів, які використовуються під час послідовного обходу пікселів, можна віднайти в [1].

Прогресуюче стиснення зображень без втрат. Симетричні ієрархічні предиктори

Поступальне ієрархічне стиснення зображень дає змогу, з одного боку, прискорити декодування, а з іншого, врахувати під час використання предикторів значення попередніх опрацьованих елементів з чотирьох, а не лише двох різних боків. Саме тому для досягнення мети дослідження нами розроблено дієву схему обходу пікселів та відповідні предиктори, які реалізують прогресуюче ієрархічне стиснення зображень без втрат. Зокрема, для прогресуючого ієрархічного обходу ми пропонуємо схему, за якою на першому шарі піксели зображення опрацьовуються послідовно, починаючи з першого, по рядках згори донизу, а у кожному рядку – підряд зліва направо з кроком $h_1 = 2^k$, де *k* визначається з умови $k = \left\lfloor \log_2\left(\frac{\max(\min(height; width); 16) - 1}{15} \right) \right\rfloor$, *height* – кількість рядків, *width* – кількість стовпців пікселів зображення. Цей крок забезпечує опрацювання на першому шарі принаймні 16 пікселів (за наявності) по кожній з осей (як у піктограмах), якщо зображення має неменші розміри. На наступних шарах ($l = \overline{2, k+1}$) проміжні піксели зображення обробляються в два проходи: на першому послідовно опрацьовуються ті з них, які містяться на перетині діагоналей квадратів з вершинами у суміжних пікселах попереднього шару з кроком $h_l = 2^{k+2-l}$ як по рядках, так і по стовпцях, а на другому необроблені піксели обходяться між суміжними пікселами попереднього шару і першого проходу з тим самим кроком по стовпцях і з удвічі зменшеним – по рядках (рис. 3).

Запропонована послідовність обходу пікселів зображення дає змогу не лише прискорити декодування, коли розміри області виводу значно менші від розмірів зображення, а й застосовувати ієрархічні предиктори для прогнозування значення кожного елемента чергового піксела (на рис. 4 позначено через *X*) на всіх шарах, починаючи з другого. Для опису цих предикторів позначимо значення яскравостей аналогічних компонентів **найближчих** (суміжних) опрацьованих раніше пікселів з попереднього і чергового шару чи проходу за допомогою позначень *a*, *b*, *c*, *d*, *ab*, *ad* (рис. 4).

П	2	П	2	П	…
2	1	2	1	2	…
П	2	П	2	П	…
2	1	2	1	2	…
П	2	П	2	П	…
…	…	…	…	…	…

Рис. 3. Схема прогресуючого ієрархічного опрацювання пікселів зображення на шарах, починаючи з другого: П – пікселі попереднього шару; 1 – пікселі першого проходу чергового шару; 2 – пікселі другого проходу чергового шару

а *б*

Рис. 4. Схеми розміщення суміжних опрацьованих елементів для елемента X на шарах, починаючи з другого: а) для першого проходу; б) для другого проходу

Використовуючи ці позначення, дослідимо принципи прогнозування двох основних симетричних ієрархічних предикторів, **орієнтованих на базове середнє арифметичне** (шумова складова) **тих двох протилежних елементів з найближчих чотирьох, які найменше між собою відрізняються** (трендова складова), тобто найімовірніше належать одному об'єкту на зображенні. У випадку, коли більший вплив на яскравості компонентів близьких пікселів мають слабкі фонові коливання, спрогнозуємо значення чергового елемента за допомогою трендово-шумового предиктора *ProgresPredict1*, який повертає базове середнє арифметичне. Якщо ж переважають сильні коливання яскравостей пікселів, то спрогнозуємо значення чергового елемента за допомогою шумово-трендового предиктора *ProgresPredict2*. Цей предиктор серед чотирьох суміжних опрацьованих елементів *a*, *b*, *c*, *d* визначає і повертає найближче значення до базового середнього арифметичного, коли таке значення єдине. Якщо ж серед суміжних опрацьованих елементів є два найближчих рівновіддалених значення до базового середнього арифметичного, то серед них повертається те, яке частіше повторюється, а коли й кількість їх повторень однакова – то **менше** з цих двох значень. Загальна орієнтація на менші значення дає змогу змістити ненульові відхилення Δ_{ij}, обчислені згідно з (2), в бік додатних значень і цим самим підвищити нерівномірність їх розподілу. Мовою С ці предиктори записуються так:

```
ubyte ProgresPredict1(ubyte a, ubyte b, ubyte c, ubyte d)
 {if (abs(a-c)<=abs(b-d))  return (a+c)/2
  else  return (b+d)/2; }

ubyte ProgresPredict2(ubyte a, ubyte b, ubyte c, ubyte d)
 {ubyte absac, absbd, maxac, minac, maxbd, minbd, prognozn;
  if (a<=c)   {absac=c-a; maxac=c; minac=a;}
  else   {absac=a-c; maxac=a; minac=c; }
```

```
if (b<=d)   {absbd=d-b; maxbd=d; minbd=b; }
else   {absbd=b-d; maxbd=b; minbd=d; }
if (absac<=absbd)
 {if (minbd>=minac && minbd<=maxac)   return minbd;
  if (maxbd>=minac && maxbd<=maxac)   return maxbd;
  return minac; }
else
 {if (minac>=minbd && minac<=maxbd)   return minac;
  if (maxac>=minbd && maxac<=maxbd)   return maxac;
  return minbd; }}
```

Проаналізуємо результати застосування запропонованих предикторів ієрархічного обходу (табл. 1) для зменшення ентропії компонентів пікселів стандартного тестового набору Archive Comparison Test (ACT), який містить як синтезовані (№№ 1 (з шумами), 2, 7), так і фотореалістичні (всі інші) зображення (для порівняння у рядку *NonePredict* цієї таблиці наведена ентропія яскравостей компонентів пікселів без застосування предикторів, а нижче – ентропія цих же яскравостей після використання найпоширеніших предикторів послідовного обходу). Завантажити TIFF-версії цих зображень можна, наприклад, з http://www.compression.ca/act/act-files.html чи з http://www.compression.ru/arctest/act/act-tif.htm.

Таблиця 1

Ентропія яскравостей компонентів пікселів зображень набору ACT після застосування різних предикторів, bpb

Предиктор	Шар	Прохід	№ файла								Середня ентропія
			1	2	3	4	5	6	7	8	
NonePredict	1	1	7.54	4.65	7.75	7.50	7.66	7.32	5.99	7.66	7.01
LeftPredict	1	1	3.57	1.68	5.26	4.57	4.60	5.67	1.74	5.13	4.03
AbovePredict	1	1	3.68	2.23	4.92	4.60	4.72	5.97	1.86	4.94	4.12
AveragePredict	1	1	4.66	2.88	4.86	4.27	4.40	5.59	2.58	4.74	4.25
PaethPredict	1	1	1.90	1.51	4.90	4.26	4.29	5.50	1.35	4.61	3.54
MedPredict	1	1	1.90	1.54	4.84	4.15	4.20	5.43	1.37	4.50	3.49
ProgresPredict1	2	1	7.46	6.63	6.93	6.83	7.17	7.28	7.08	7.33	7.09
	2	2	7.05	5.80	6.19	6.43	6.07	6.96	6.11	7.10	6.46
	3	1	7.26	6.36	6.43	6.44	6.61	7.12	6.34	6.91	6.68
	3	2	6.70	5.66	5.80	6.02	5.69	6.81	5.27	6.53	6.06
	k+1	1	5.65	2.88	4.80	4.11	4.44	5.60	2.85	4.53	4.36
	k+1	2	3.99	1.89	4.45	3.61	3.83	4.93	1.73	3.87	3.54
	Разом		4.89	2.56	4.73	4.04	4.22	5.41	2.47	4.42	4.09
ProgresPredict2	2	1	7.40	5.68	6.99	6.89	7.21	7.34	6.39	7.32	6.90
	2	2	7.04	5.18	6.18	6.49	6.12	6.98	5.22	7.02	6.28
	3	1	7.00	5.36	6.54	6.56	6.71	7.24	5.59	6.94	6.49
	3	2	6.69	4.97	5.88	6.11	5.82	6.88	4.27	6.52	5.89
	k+1	1	4.73	2.07	5.03	4.49	4.65	5.84	1.90	4.95	4.21
	k+1	2	0.23	1.19	4.68	4.03	4.11	5.24	0.96	4.39	3.10
	Разом		2.75	1.81	4.94	4.41	4.50	5.66	1.64	4.83	3.82
ProgresPredict3	2	1	7.37	5.73	6.87	6.68	7.10	7.25	6.29	7.28	6.82
	3	1	7.08	5.44	6.27	6.46	6.50	7.13	5.28	6.90	6.38
	k+1	1	4.91	1.61	4.88	4.30	4.51	5.71	1.70	4.76	4.05
	Разом		2.79	1.66	4.89	4.35	4.44	5.62	1.55	4.77	3.76
ProgresPredict4	2	1	7.46	5.80	7.02	6.86	7.10	7.41	6.15	7.44	6.91
	3	1	7.19	5.36	6.62	6.61	6.66	7.27	5.53	7.12	6.55
	k+1	1	2.55	1.82	5.17	4.71	4.78	6.03	1.44	5.22	3.97
	Разом		2.22	1.74	4.99	4.48	4.54	5.73	1.50	4.91	3.76

Як і варто було очікувати, застосування шумово-трендового предиктора виявилося ефективнішим для синтезованих і на деяких початкових проходах окремих (№№ 3, 8) фотореалістичних зображень, оскільки для них характерні різкі перепади яскравостей опрацьованих пікселів на межах зображених об'єктів (переважає вплив тренду), а трендово-шумового – загалом для фотореалістичних зображень, адже їх суміжні пікселі найчастіше мають близькі, але неоднакові кольори (переважає вплив шуму). Використання симетричних предикторів істотно зменшує ентропію на останніх шарах у порівнянні з іншими відомими сьогодні предикторами послідовного обходу, оскільки ці предиктори враховують вплив чотирьох рівновіддалених пікселів з різних боків, а не лише пікселів зліва і згори, як у випадку послідовного обходу. При цьому кожен опрацьований піксел чергового шару в середині зображення використовується для оброблення восьми пікселів наступного шару (а не для максимум трьох наступних пікселів, як для предикторів послідовного обходу).

Рівень кореляції яскравостей компонентів суміжних пікселів на останніх шарах істотно вищий від рівня кореляції цих яскравостей на перших шарах, особливо для синтезованих зображень. Ось чому застосування предикторів послідовного обходу (а інколи – і відмова від їх використання) забезпечують меншу ентропію від прогресуючих на початкових шарах, хоча ефективність останніх зі збільшенням номера шару зростає. Тому, щоб загалом не збільшувати ентропію, для синтезованого зображення № 2 доцільно відмовитися від застосування предикторів як на другому, так і на третьому шарі, а для зображення № 7 – на першому проході другого шару.

Крім цього, під час виконання оптимізованого варіанта шумово-трендового предиктора *ProgresPredict2* в середньому здійснюється 6.33 операцій порівняння, а трендово-шумового предиктора *ProgresPredict1* – лише 3 таких операції. Тобто використання трендово-шумового предиктора замість шумово-трендового зменшує кількість таких операцій у понад два рази, тому предиктору *ProgresPredict1* слід надавати перевагу перед *ProgresPredict2* при однакових значеннях ентропії після їх застосування. Для порівняння: під час виконання нелінійного статичного предиктора послідовного обходу *PaethPredict* в середньому виконується 5.66, а *MedPredict* – 5 операцій порівняння. Тобто застосування трендово-шумового предиктора *ProgresPredict1* навіть замість нелінійних предикторів послідовного обходу прискорює кодування/декодування.

Використання асиметричних ієрархічних предикторів

Підвищити ефективність шумово-трендових предикторів можливо за рахунок врахування на першому проході кожного шару значень суміжних опрацьованих елементів зліва і згори цього ж проходу (на рис. 4 а вони позначені відповідно через *ad* і *ab*). Адже кольори пікселів найчастіше подібні до кольорів близьких пікселів саме по горизонталі чи вертикалі і значно рідше – до кольорів близьких пікселів по діагоналі. Цим (а не лише більшими відстанями до прогнозованого елемента) пояснюється значно нижча ефективність застосування симетричних предикторів на першому проході відносно другого для кожного шару довільного зображення (див. табл. 1). Врахувати ж значення аналогічних елементів справа і знизу відносно елемента *X* неможливо, оскільки під час застосування предикторів до цього елемента вони ще не опрацьовані (саме тому описані нижче предиктори належать до асиметричних).

Розглянемо принципи прогнозування двох розроблених нами асиметричних шумово-трендових предикторів. Перший з них, *ProgresPredict3*, повертає найближче значення до базового середнього арифметичного не лише серед чотирьох найближчих суміжних опрацьованих елементів *a*, *b*, *c*, *d*, як *ProgresPredict2*, а й додатково серед *ad* і *ab*, надаючи їм перевагу над іншими елементами за однакових відстаней. Для прискорення обчислень найменша відстань до базового середнього арифметичного визначається неявно, використовуючи той факт, що точка *z* тим ближче до середини між точками *x* та *y* ($x \le y$), чим меншою є різниця $\left|(y-z)-(z-x)\right| = \left|y-2z+x\right|$.

Другий з асиметричних предикторів, *ProgresPredict4*, визначає найменший за абсолютною величиною діагональний приріст відносно елементів *ad* і *ab* та повертає значення у напрямку цього приросту відносно елемента *X*, якщо даний приріст менший від половини меншого з діагональних

приростів найближчих елементів ($\min(|a-c|,|b-d|)/2$). При цьому серед двох однакових діагональних приростів перевага надається тому, який пов'язаний з меншим прямолінійним приростом навколо елемента *X*. Інакше предиктор серед чотирьох найближчих суміжних опрацьованих елементів *a*, *b*, *c*, *d* визначає три найближчих до базового середнього арифметичного (найімовірніше належать одному зображеному об'єкту) та повертає серед них середнє значення. Мовою С дані асиметричні ієрархічні предиктори записуються так:

```
ubyte ProgresPredict3(ubyte a, ubyte b, ubyte c, ubyte d, ubyte ad, ubyte ab)
 {ubyte absac, absbd, maxac, minac, maxbd, minbd, prognozn, s, mins;
  if (a<=c)   {absac=c-a; maxac=c; minac=a; }
  else   {absac=a-c; maxac=a; minac=c; }
  if (b<=d)   {absbd=d-b; maxbd=d; minbd=b; }
  else   {absbd=b-d; maxbd=b; minbd=d; }
  if (absac<=absbd)
   {prognozn=ad; mins=abs(maxac+minac-2*ad);
    s=abs(maxac+minac-2*ab);
    if (s<mins)   {prognozn=ab; mins=s; }
    s=abs(maxac+minac-2*minbd);
    if (s<mins)   {prognozn=minbd; mins=s; }
    s=abs(maxac+minac-2*maxbd);
    if (s<mins)   {prognozn=maxbd; mins=s; }
    if (minac>=prognozn || prognozn>maxac)   return minac;
    return prognozn; }
   else
    {prognozn=ad; mins=abs(maxbd+minbd-2*ad);
    s=abs(maxbd+minbd-2*ab);
    if (s<mins)   {prognozn=ab; mins=s; }
    s=abs(maxbd+minbd-2*minac);
    if (s<mins)   {prognozn=minac; mins=s; }
    s=abs(maxbd+minbd-2*maxac);
    if (s<mins)   {prognozn=maxac; mins=s; }
    if (minbd>=prognozn || prognozn>maxbd)   return minbd;
    return prognozn; }}

ubyte ProgresPredict4(ubyte a, ubyte b, ubyte c, ubyte d, ubyte ad, ubyte ab)
 {ubyte absac, absbd, maxac, minac, maxbd, minbd, prognozn, minprD, minprGV, prD, prGV;
  if (a<=c)   {absac=c-a; maxac=c; minac=a; }
  else   {absac=a-c; maxac=a; minac=c; }
  if (b<=d)   {absbd=d-b; maxbd=d; minbd=b; }
  else   {absbd=b-d; maxbd=b; minbd=d; }
  minprD=abs(ab-a); minprGV=abs(a-d); prognozn=d;
  prD=abs(ab-b);
  if (prD<=minprD)
   {prGV=abs(b-c);
    if (prD<minprD || prGV<minprGV)   {minprD=prD; minprGV=prGV; prognozn=c; }}
  prD=abs(ad-a);
  if (prD<=minprD)
   {prGV=abs(a-b);
    if (prD<minprD || prGV<minprGV)   {minprD=prD; minprGV=prGV; prognozn=b; }}
  prD=abs(ad-d);
  if (prD<=minprD)
```

```
  {prGV=abs(d-c);
   if (prD<minprD || prGV<minprGV)   {minprD=prD; minprGV=prGV; prognozn=c; }}
 if (2*minprD<min(absac, absbd))   return prognozn;
 if (absac<=absbd)
  {if (abs(a+c-2*b)<=abs(a+c-2*d))   prognozn=b;
   else   prognozn=d;
   if (maxac<=prognozn)   return maxac;
   if (minac>=prognozn)   return minac;
   return prognozn; }
 else
  {if (abs(b+d-2*a)<=abs(b+d-2*c))   prognozn=a;
   else   prognozn=c;
   if (minbd>=prognozn)   return minbd;
   if (maxbd<=prognozn)   return maxbd;
   return prognozn; }}
```

Результати застосування запропонованих асиметричних ієрархічних предикторів для зменшення ентропії компонентів пікселів набору зображень ACT на першому проході окремих шарів наведені у нижній частині табл. 1 (нагадаємо, що на другому проході при цьому використовується, як і раніше, шумово-трендовий предиктор *ProgresPredict2*, який враховує горизонтальні та вертикальні коливання яскравостей опрацьованих суміжних елементів). З даних цієї таблиці бачимо, що застосування на першому проході кожного шару асиметричних предикторів замість симетричних дало змогу зменшити ентропію синтезованих зображень на 0.24 bpb в основному за рахунок першого проходу останнього шару, де ентропія для таких зображень зменшилася на 0.98 bpb. Але зрозуміло, що *ProgresPredict3* та *ProgresPredict4* використовують для прогнозування значення шести, а не чотирьох опрацьованих елементів та виконують відповідно в середньому щоразу 11.33 та 18.83, а не 6.33 операції порівняння, як *ProgresPredict2* і тому розраховуються довше. Крім цього, предиктор *ProgresPredict3* виявився ефективнішим для синтезованого зображення № 2, а *ProgresPredict4* – для зображень №№ 1, 7. Тобто для різних синтезованих зображень, і навіть для різних їх фрагментів та проходів ефективнішими можуть виявитися різні асиметричні ієрархічні предиктори.

Комплексне застосування ієрархічних предикторів

Наведені результати дослідження вказують на те, що предиктори слід використовувати лише з того проходу і того шару, коли їх застосування починає зменшувати ентропію, а серед предикторів необхідно обирати той, який забезпечує найменшу ентропію після свого застосування. Більше того, рівень впливу і характер шумів та трендів для різних фрагментів зображення може відрізнятися, тому для них можуть виявитися ефективними різні ієрархічні предиктори. Підкреслимо, що відмовлятися від застосування предикторів слід для цілого проходу, а не для окремих його фрагментів, оскільки значення яскравостей компонентів зображень без та після використання предикторів мають різний характер нерівномірностей розподілу (див. рис. 1). Тому під час прогресуючого ієрархічного стиснення для кожного проходу чергового шару доцільно спочатку встановити, чи застосування предикторів загалом зменшує ентропію яскравостей компонентів його пікселів, після чого, у разі прийняття рішення про застосування предикторів, обрати для кожного однорідного фрагмента пікселів проходу саме той ієрархічний предиктор, який максимально зменшує його ентропію.

Для прийняття рішення про застосування предикторів на кожному проході чергового шару ми використовували симетричні ієрархічні предиктори, оскільки вони розраховуються швидше, а у випадку зменшення ними ентропії обирали **для кожного рядка цього проходу** найефективніший серед симетричних предикторів і додатково для перших проходів – найефективніший між всіма наведеними ієрархічними предикторами. Результати застосування цих способів комбінування

ієрархічних предикторів для зменшення ентропії компонентів пікселів набору зображень ACT наведені у табл. 2–4. Для порівняння на початку цих таблиць наведені результати аналогічного застосування до цих же зображень комбінацій послідовних предикторів, які для кожного окремого рядка забезпечують найменшу ентропію.

Таблиця 2

Ентропія яскравостей компонентів пікселів зображень набору ACT після застосування різних комбінацій предикторів, bpb

Комбінація	№ файла								Середня ентропія
	1	2	3	4	5	6	7	8	
Послідовні предиктори	1.89	1.46	4.80	4.15	4.20	5.43	1.33	4.50	3.47
Прогресуючі симетричні предиктори	2.75	1.80	4.73	4.04	4.21	5.41	1.64	4.42	3.63
Прогресуючі симетричні і асиметричні предиктори	2.26	1.69	4.73	4.04	4.19	5.41	1.53	4.42	3.53

Таблиця 3

Час кодування різними комбінаціями предикторів пікселів зображень набору ACT, с

Комбінація	№ файла								Середній час
	1	2	3	4	5	6	7	8	
Послідовні предиктори	6.16	10.22	2.30	3.41	2.26	3.46	4.18	3.41	4.43
Прогресуючі симетричні предиктори	4.83	7.91	1.81	2.69	1.76	2.80	3.30	2.74	3.48
Прогресуючі симетричні і асиметричні предиктори	6.04	10.00	2.30	3.35	2.36	3.46	4.18	3.41	4.39

Таблиця 4

Час декодування різними комбінаціями предикторів пікселів зображень набору ACT, с

Комбінація	Кількість шарів	№ файла								Середній час
		1	2	3	4	5	6	7	8	
Послідовні предиктори	4	0.44	1.22	0.35	0.53	0.34	0.44	0.42	0.44	0.52
	всі	0.44	1.22	0.35	0.53	0.34	0.44	0.42	0.44	0.52
Прогресуючі симетричні предиктори	4	0.11	0.06	0.05	0.05	0.06	0.11	0.11	0.05	0.08
	всі	2.09	3.46	0.77	1.10	0.83	1.10	1.43	1.10	1.49
Прогресуючі симетричні і асиметричні предиктори	4	0.17	0.05	0.05	0.06	0.05	0.11	0.11	0.11	0.09
	всі	2.64	4.39	0.77	1.10	0.82	1.15	1.87	1.16	1.74

Як свідчать дані всіх таблиць, застосування комбінацій всіх ієрархічних предикторів зменшує ентропію відносно найефективнішого з цих окремих предикторів в середньому по набору ACT на 0.23 bpb переважно за рахунок синтезованих зображень. Додаткове застосування асиметричних предикторів хоча й у середньому сповільнює кодування на 26 % і декодування на 17 %, але зменшує ентропію на 0.1 bpb. Тому для швидкого ієрархічного стиснення зображень доцільно застосовувати комбінування симетричних предикторів, а для забезпечення максимальної компресії – додатково на перших проходах кожного шару комбінування симетричних та асиметричних предикторів. Застосування ієрархічних предикторів хоча й у середньому забезпечує меншу ентропію від послідовних на 0.27 bpb для синтезованих зображень, але досягає кращих значень по цьому показнику на 0.06 bpb для фотореалістичних знімків. І головне: прогресуючий ієрархічний спосіб обходу пікселів дає змогу виконати декодування значно швидше від послідовного обходу, коли розміри області виводу значно менші розмірів зображень (наприклад, для заповнення області виводу 128 x 128 пікселів (4 шари) таке декодування виконується в середньому у 5.78 рази швидше, див. табл. 4), оскільки час ієрархічного декодування залежить від

меншого серед розмірів області виводу та розмірів зображення (про це наголошувалося ще в [6, с. 175]), а послідовного – лише від розмірів зображення.

Висновки і перспективи подальших досліджень

1. У нових версіях графічних форматів та нових форматах компресії зображень без втрат доцільно реалізувати прогресуюче ієрархічне стиснення, оскільки це дозволяє істотно прискорити декодування, коли розміри області виводу менші від розмірів зображення.

2. Зменшення розмірів стиснутих прогресуючим ієрархічним способом зображень досягається переважно на останніх шарах, оскільки пікселі, які використовують предиктори, мають з прогнозованим пікселом в середньому найвищий рівень кореляції відносно інших шарів.

3. На першому проході кожного шару в процесі ієрархічного стиснення підвищити ефективність застосування симетричних предикторів, які враховують яскравості опрацьованих найближчих пікселів по діагоналі, можливо за рахунок врахування яскравостей опрацьованих на цьому ж проході найближчих суміжних пікселів по горизонталі та вертикалі.

4. Під час контекстно-незалежного стиснення без втрат для однорідних фрагментів зображень доцільно обирати той предиктор з декількох альтернативних, який дає змогу максимально зменшити ентропію.

Надалі, з метою додаткового зменшення розмірів файлів стиснутих зображень без втрат і прискорення декодування, планується адаптувати контекстно-залежні методи компресії [3] до ієрархічного обходу пікселів зображень та підвищити ефективність застосування симетричних і асиметричних предикторів за допомогою різницевих колірних моделей [7].

1. Бредихин Д. Ю. Сжатие графики без потерь качества [Електронний ресурс] / Д. Ю. Бредихин. – 2004. – Режим доступу: http://www.compression.ru/download/articles/i_lless/bredikhin_2004_lossless_image_compression_doc.rar. 2. Гонсалес Р. Цифровая обработка изображений / Р. Гонсалес, Р. Вудс. – М.: Техносфера, 2005. – 1072 с. 3. Методы сжатия данных. Устройство архиваторов, сжатие изображений и видео / Д. Ватолин, А. Ратушняк, М. Смирнов, В. Юкин. – М.: ДИАЛОГ-МИФИ, 2003. – 384 с. 4. Миано Дж. Форматы и алгоритмы сжатия изображений в действии: Учеб. пособ. / Дж. Миано. – М.: Триумф, 2003. – 336 с. (Серія: Практика программирования). 5. Прэтт Э. Цифровая обработка изображений: Пер. с англ. / Э. Прэтт. – М.: Мир, 1982. – Кн. 2, 480 с. 6. Сэломон Д. Сжатие данных, изображений и звука / Д. Сэломон. – М.: Техносфера, 2006. – 368 с. – (Серія: Мир программирования: цифровая обработка сигналов). 7. Шпортько О. В. Використання різницевих кольорових моделей для стиснення RGB-зображень без втрат / О. В. Шпортько // Відбір і обробка інформації. – 2009. – № 31 (107). – С. 90–97.